Bernhard Steffen (Ed.)

LNCS 16032

Bridging the Gap Between AI and Reality

Second International Conference, AISoLA 2024
Crete, Greece, October 30 – November 3, 2024
Selected Papers

Springer

OPEN ACCESS

Lecture Notes in Computer Science **16032**

The series Lecture Notes in Computer Science (LNCS), including its subseries Lecture Notes in Artificial Intelligence (LNAI) and Lecture Notes in Bioinformatics (LNBI), has established itself as a medium for the publication of new developments in computer science and information technology research, teaching, and education.

LNCS enjoys close cooperation with the computer science R & D community, the series counts many renowned academics among its volume editors and paper authors, and collaborates with prestigious societies. Its mission is to serve this international community by providing an invaluable service, mainly focused on the publication of conference and workshop proceedings and postproceedings. LNCS commenced publication in 1973.

Bernhard Steffen
Editor

Bridging the Gap Between AI and Reality

Second International Conference, AISoLA 2024 Crete, Greece, October 30 – November 3, 2024 Selected Papers

Springer

Editor
Bernhard Steffen
TU Dortmund University
Dortmund, Germany

ISSN 0302-9743 ISSN 1611-3349 (electronic)
Lecture Notes in Computer Science
ISBN 978-3-032-01376-7 ISBN 978-3-032-01377-4 (eBook)
https://doi.org/10.1007/978-3-032-01377-4

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

This volume contains the postproceedings of AISoLA 2024, the 2nd International Symposium on *Bridging the Gap between AI and Reality*, which took place in Hersonissos, Crete (Greece) on October 30 – November 3, 2024 as an in-person event that provides an interdisciplinary forum for discussing the impact of the recent AI developments on research, education, and society. It is our core belief that this topic must be explored from multiple perspectives to establish a holistic understanding. Therefore AISoLA invites researchers from various backgrounds, like computer science, philosophy, psychology, law, economics, and social studies, to participate in an interdisciplinary exchange of ideas and to establish new collaborations. AISoLA is an AI-themed sibling of ISoLA, the International Symposium on Leveraging Applications of Formal Methods, which it complements with its interdisciplinary perspective.

The program of AISoLA 2024 consisted of four keynotes given by:

- Christel Baier
- Tom Henzinger
- Edward Lee
- Ina Schieferdecker

Three impulse talks focusing on the question "Are AI Minds Minds", given by

- Jan Broersen
- Hans-Johann Glock
- Karin Vold

A collection of special tracks devoted to the following hot and emerging topics:

- AI Assisted Programming (AIAP) (Organizers: Wolfgang Ahrendt, Bernhard Aichernig, Klaus Havelund)
- Digital Humanities (DigHum) (Organizers: Ciara Breathnach, Tiziana Margaria)
- Health Care – Approaches Using Formal Methods and AI (HC-FMAI) (Organizers: Martin Leucker, Violet Kai Pun)
- Responsible and Trusted AI: An Interdisciplinary Perspective (RTAI) (Kevin Baum, Thorsten Helfer, Markus Langer, Eva Schmidt, Andreas Sesing-Wagenpfeil)
- Safe Autonomous Vehicles (SAV) (Falk Howar, Hardi Hungar)
- Statistical Model Checking (SMC) (Kim Larsen, Jan Kretínsky, Sudeep Kanav)
- Verification and Learning for Assured Autonomy (VLAA) (Devdatt Dubhashi, Raúl Pardo, Gerardo Schneider, Hazem Torfah)
- Verification for Neuro-Symbolic Artificial Intelligence (VNSAI) (Taylor Johnson, Daniel Neider)

And the embedded event:

- Doctoral Symposium and Poster Session (Sven Jörges, Salim Saay, Steven Smyth)

Co-located with the Symposium was:

- The STRESS Summer School 2024 (Tiziana Margaria, Bernhard Steffen, John Hatcliff)

The 15 papers of this volume extend the presentation in the AISoLA 2024 on-site proceedings. This means, in particular, that the corresponding track introductions still apply.

We were pleased to implement a single-blind review process of all submitted content. As with the tradition for ISoLA, also for AISoLA the track organizers form the program committee. We thank them and the reviewers for their effort in selecting the papers tobe presented. We are grateful to the local Organization Chair, Petros Stratis, and the EasyConferences team for their continuous precious support during the entire period preceding the events, and Springer for being, as usual, a very reliable partner for the proceedings production. Finally, we are thankful to Nicolas Stratis, Daniel Busch, and Steven Smyth for their continuous support for the website and the program, and to Steve Bosselmann for his help with the editorial system EquinOCS.

Special thanks are due to the Center for Trustworthy Data Science and Security, the Lamarr Institute for Machine Learning and Artificial Intelligence, and the Center for Perspicuous Computing for their support in the organization of the event, as well as to the Technical University of Dortmund, my home institution.

With over 300 international participants, the combined ISoLA–AISoLA symposium was a very successful event and I am looking forward to seeing many of you in Rhodes in November for AISoLA 2025.

June 2025 Bernhard Steffen

Organization

Program Chair

Bernhard Steffen	TU Dortmund University, Germany

Program Committee

Wolfgang Ahrendt	Chalmers University of Technology, Sweden
Bernhard Aichernig	TU Graz, Austria
Kevin Baum	German Research Center for Artificial Intelligence, Germany
Klaus Havelund	NASA Jet Propulsion Laboratory, USA
Thorsten Helfer	Saarland University, Germany
Falk Howar	TU Dortmund University, Germany
Taylor T. Johnson	Vanderbilt University, USA
Sudeep Kanav	Masaryk University, Czech Republic
Jan Kretínsky	Technical University of Munich, Germany and Masaryk University, Czech Republic
Markus Langer	University of Freiburg, Germany
Kim Larsen	Aalborg University, Denmark
Martin Leucker	University of Lübeck, Germany
Tiziana Margaria	University of Limerick, Ireland
Daniel Neider	TU Dortmund University, Germany
Violet Kai I Pun	Western Norway University of Applied Sciences, Norway
Eva Schmidt	TU Dortmund University, Germany
Andreas Sesing-Wagenpfeil	Saarland University, Germany

Reviewers

Serge Autexier	Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Bremen, Germany
Richard Bergs	University of Freiburg, Germany
Markus Bertl	Vienna University of Economics and Business, Austria
Arnd Hartmanns	University of Twente, Enschede, Netherlands

Navid Hashemi	Vanderbilt University, USA
Tim Hunsicker	Saarland University, Germany
Felix Kares	Saarland University, Germany
Yngve Lamo	Western Norway University of Applied Science, Norway
Diego Manzanas Lopez	Vanderbilt University, Nashville, USA
Ludwig Pechmann	UniTransferKlinik Lübeck, Germany
Fazle Rabbi	University of Bergen, Norway
Martin Sachenbacher	University of Lübeck, Germany
Nadine Schlicker	University of Marburg, Germany
Timo Speith	University of Bayreuth, Germany
Sarah Sterz	Saarland University, Germany
Daniel Thoma	University of Lübeck, Germany
Tom van Dijk	University of Twente, Enschede, Netherlands

Contents

AI Assisted Programming

Correct-ish by Design: From Upfront Verification to Continuous Monitoring of LLM Generated Code

Bernhard K. Aichernig[1,2] and Klaus Havelund[3(✉)]

[1] Institute of Software Engineering and Artificial Intelligence, Graz University of Technology, Graz, Austria
[2] Institute for Formal Models and Verification, Johannes Kepler University Linz, Linz, Austria
[3] Jet Propulsion Laboratory, California Institute of Technology, Pasadena, USA
havelund@gmail.com

Abstract. As developers increasingly rely on Large Language Models (LLMs) to generate code, the pace of software development is accelerating beyond the capabilities of traditional design-time verification and testing methods. We predict a paradigm shift towards continuous monitoring to complement and eventually supersede upfront verification. By embracing a "correct-ish by design" philosophy, we acknowledge the inevitability of imperfections in LLM-generated code. We anticipate an adaptive approach where real-time monitoring and feedback mechanisms are employed to detect, diagnose, and rectify issues as they emerge in the field. This continuous monitoring strategy not only ensures sustained software reliability and performance, but also provides valuable insights into LLM behavior, facilitating iterative improvements. Specifically, we use an LLM to generate Python code from a formal specification written in the VDM specification language, accessible as a PDF document. The VDM specification formalizes aspects of NASA's SAFER rescue system, which uses small thrusters on a backpack to let astronauts maneuver and return safely to the spacecraft during spacewalks in case they become untethered. We experiment with property-based testing, and by using two Python programs, both generated from the specification by the LLM in two different developments, to monitor each other during runtime.

1 Introduction

Large Language Models (LLMs) are becoming important tools for the software developer. They may even replace the software developer at some point. Especially code generation from natural language prompts is already now practical

K. Havelund—The research performed by this author was carried out at Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 3–29, 2026.
https://doi.org/10.1007/978-3-032-01377-4_1

and used in real development. On the negative side, LLM generated code is (still) known not to be reliably correct, and even if code generation from natural language was perfect, the natural language can be imprecise and incomplete. Programmers currently handle this problem by *reviewing* the generated code, and of course *testing* it. However, as LLMs become used more extensively, more code will be automatically generated and full code review will not be practical. At that point, automated program analysis becomes critical.

One such program analysis approach is *Runtime Verification* (RV), where the program is monitored during its execution and checked against some other formalization of the expected behavior, either during test, or in deployment. If we assume that software may not be fully correct (only "correct-ish") by design, continuous monitoring may be the only way to improve trust in at least the *execution* of the generated code. In this paper we investigate this idea by letting ChatGPT [9] generate Python code from an existing formal mathematical specification, described in [3,4], of NASA's SAFER protocol [11], which controls a lightweight backpack propulsion system for an astronaut, who can use it to maneuver safely back to the space vehicle in case of an erroneous separation.

The SAFER protocol is in [3,4] formally specified in the VDM (Vienna Development Method) specification language [8,12,17], in particular the VDM specification language standard VDM-SL [18], from here referred to as VDM. The specification is written in an executable subset of VDM, making translation to Python rather direct. In fact, the executable subset of VDM has many similarities with Python, as we will demonstrate in this paper. In [15] we studied the relationship between VDM and Scala. Many of the observations there can be carried over to Python.

Runtime verification [7,16] is, as mentioned, the automated verification that an execution of a program (or system) satisfies a formalized specification. The monitored program must be annotated to drive the monitor (online monitoring)[1]. Examples of formal specification formalisms include simple assertions in the code, temporal logics, regular expressions, state machines, production rules, and stream processing formalisms. Differential testing [14,19], which we apply in this paper, can be considered as a variant of runtime verification where the specification is "another program", the *reference implementation*, with the same expected functionality. The reference implementation can either be an alternative full implementation, as in [19] where different C compilers are compared, or it can be an abstract smaller program, as in [14]. The idea of comparing the execution of an implementation with that of a more abstract implementation, or model, has been formalized in terms of proofs with refinement mappings [1,2]. In [5] we explored the idea of monitoring such a refinement mapping between an LLM generated implementation in Scala and its abstract model.

In this paper we examine runtime verification with a reference implementation, as well as with assertions. The two authors performed separate *developments*, referred to as A and B, with ChatGPT to generate two Python versions of

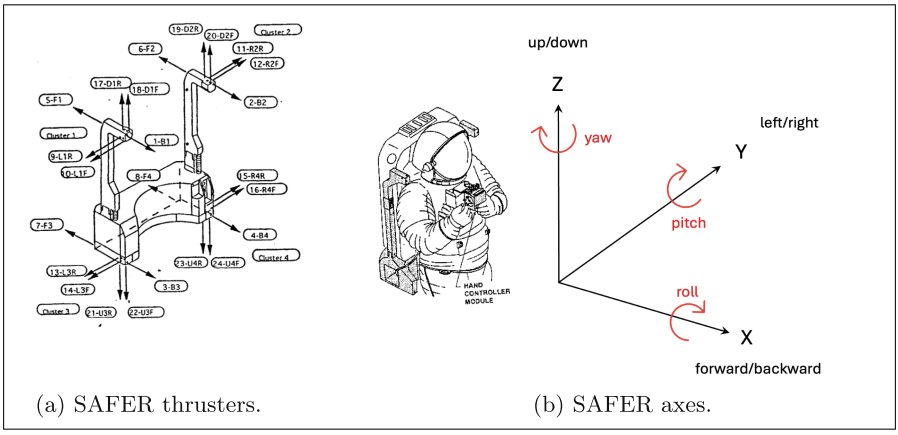
[1] The instrumented code can alternatively generate a log which is then checked against the specification post-mortem.

the VDM specification in the 21 page technical report [4]. We then used these as each other's reference implementation, and in this way found an error introduced in one of them by ChatGPT. It leads to the quite interesting observation that with LLMs one can imagine generating multiple versions of software checking each other, leading to an inexpensive way of applying N-version programming [6]. The paper also demonstrates automated translation from a high-level mathematical specification in a PDF document to code.

The paper is organized as follows. Section 2 introduces the SAFER protocol. Section 3 describes our method. The VDM specification in [4], which we translate to Python using ChatGPT, consists of six modules, which we discuss in Sects. 4–9. Finally, in Sect. 10 we discuss lessons learned.

2 The SAFER System

The Simplified Aid for EVA Rescue (SAFER) [11] is a small propulsion backpack designed by NASA to assist astronauts during Extra Vehicular Activities (EVAs) in space. It is primarily meant to be activated only in emergency scenarios when an astronaut accidently becomes separated from the spacecraft. SAFER attaches directly to the astronaut's backpack. It is controlled by the astronaut using a small box sitting on the astronaut's chest, with knobs to control thrusters mounted on the backpack.



(a) SAFER thrusters. (b) SAFER axes.

Fig. 1. Thrusters and axes of movement.

The SAFER propulsion system specifically consists of 24 thrusters positioned on the backpack, see Fig. 1a, allowing *movement* in all directions, along the X (forward, backward), Y (left, right), and Z (up, down) axes, which are also called the translational axes, see Fig. 1b. In each corner of the backpack are six thrusters, three in the front and three on the rear, basically a thruster for each

axis X, Y, and Z. The thrusters are numbered 1–24, and are in addition named with letters indicating direction in which they push the astronaut (B = Back, F = Forward, L = Left, R = Right, U = Up, D = Down), numbered in which quadrant they sit (1 = upper left, 2 = upper right, 3 = lower left, 4 = lower right), and finally, if there are two thrusters next to each other, whether they sit on the Rear (R) or Front (F). In addition, it allows *rotation* around these axes: ROLL (around the X axis, tilting from side to side), PITCH (around the Y axis, tilting forward, backward), and YAW (around the Z axis, turning left, right). These six options (moving along three axes and rotating around three axes) are also referred to as the *six degrees of freedom*.

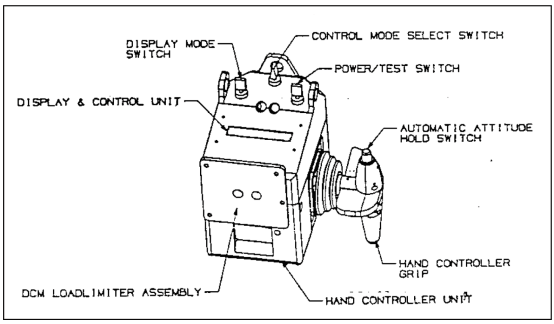


Fig. 2. The hand grip.

Commands to the thrusters are initiated through movements of a joystick, called the *hand grip*, sitting on the right-hand side of a control box, which is placed on the astronaut's chest, see Fig. 2. It allows both translational and rotational controls. The joystick can be moved forward and backward (X axis), left-right by pushing it in or pulling it out (Y axis), and can be moved up or down (Z axis). In addition, this joystick can be twisted, always meaning PITCH rotation. A separate knob can switch between translation mode (moving along the X, Y, and Z axes) or rotation mode, rotating around these axes. In translation mode the movements on the X, Y, and Z axes mean exactly that: movement, and twisting means PITCH around the Y-axis. In rotation mode, movement along the X-axis still means exactly that: movement, whereas movement along the Y and Z axes mean respectively YAW and ROLL, and twisting still means PITCH.

In addition to these maneuvering options, the SAFER system can by a button be put into Automatic Attitude Hold (AAH) mode, which stabilizes the astronaut's orientation movements automatically. This means canceling any rotations, and keeping the astronaut stable rotation wise. Uncontrolled rotation can be very disorienting. The button is pushed down and automatically comes back up on release. AAH mode can be turned off again by pressing the button twice within 0.5 s. If in AAH mode, and the astronaut issues a rotation command on a specific axis, then that takes priority and cancels the AAH module's control over the

rotation around that axis. However, if the astronaut is commanding a rotation while activating the AAH module, then the AAH module takes over full control of that axis and ignores any rotation commands from the astronaut on that axis until AAH is switched off. Finally, translational commands are prioritized with respect to each other, with X-axis movements being the highest priority, followed by Y, then Z. If rotational and translational commands are simultaneously issued, rotations have higher priority than translational commands.

The VDM-Specification consists of six modules, in this paper presented from bottom to top. The AUX (AUXiliary) module provides auxiliary definitions, such as fundamental types. The HCM (Hand Controller Module) specifies how hand grip operations are translated into logic commands for the thruster selection software. The AAH (Automatic Attitude Hold) module, mentioned above, handles the automated control of near-zero rotation, when activated by the astronaut. The TS (Thruster Selection) module computes the thrusters to be selected for thrusting. The SAFER module is the top-level module executing every cycle, calling functions in the other modules. The TEST module provides functions for testing SAFER.

As the reader may have observed, the logic is not completely trivial. To enhance astronaut safety, the SAFER system enforces specific operational constraints. It limits the number of simultaneously activated thrusters to four, reducing complexity and potential conflicts in thruster firings. Furthermore, the thruster selection logic is designed to avoid opposing thrusters firing simultaneously, which could result in ineffective propulsion and wasted fuel.

3 Methodology

We had three documents available describing the SAFER protocol. These included the original NASA report [11] providing guidelines on the use of formal methods, and using the SAFER protocol as an example, which in the report is explained and formalized using the PVS [20] theorem prover. In addition, we had access to two papers describing its formalization in VDM, including a workshop paper [3] with explanations and some specifications, and a 21 page technical report [4] containing the full VDM specification consisting of 417 lines of VDM (not counting comments and blank lines) with only few explanations. The report [4] ended up being our main source (input to ChatGPT) for the translation.

Each of the two authors independently developed the SAFER system in Python with ChatGPT 4o, referred to as *developments A and B*. A typical interactive workflow involved (1) providing ChatGPT with the PDF [4] of the VDM specification, (2) ask ChatGPT to produce Python code for a particular module, (3) reviewing of the code, (4) ask ChatGPT to correct the code, until satisfactory code was generated. Early attempts to provide ChatGPT with the complete technical report and ask it to produce the complete Python code were not very promising. Hence, we decided to develop the system module by module, starting with the simplest AUX module. In the following sections, we discuss each of the modules AUX, HCM, AAH, TS, SAFER, and TEST, separately.

Table 1 shows the number of prompts needed for each module for each of the developments A and B, as well as the total number of lines of Python code generated (not counting comments and blank lines) from the 417 lines of VDM.

Table 1. Number of prompts and Python LOC generated from 417 VDM LOC.

Development	AUX	HCM	AAH	TS	SAFER	TEST	total prompts	Python LOC
A	4	3	20	64	11	8	110	441
B	6	2	18	38	12	29	105	503

When all the Python was generated, we tested if the pre/post conditions and safety invariants were satisfied. For this an exhaustive test-case generator already present in the VDM specification was used. It basically enumerates all possible combinations of input commands in a manner similar to property-based testing [10]. Finally, when the safety was successfully tested this way, we used the two developments as each other's reference implementation, checking that they were consistent, i.e., that the same thrusters are fired.

Note that in the generated Python code we allow ourselves to occasionally insert newlines to improve layout for reading, also in cases where Python does not allow such newlines.

4 Module AUX

Module AUX is the simplest module since it contains auxiliary definitions without any dependency on other modules. The VDM types of this module specification are shown in Fig. 3. Note that it is classic VDM-style to model a problem as a collection of type definitions in this way.

The first three enumerated types (Lines 43–45) define the commands that can be issued along an axis (NEG, ZERO, POS), the possible translation axes (X, Y, Z) and the possible rotations (ROLL, PITCH, and YAW). A translation command, *TranCommand* (Line 46), is a mapping[2] from the X, Y, and Z axes to axis commands. The type of *TranCommand* is in fact defined as the subtype (the keyword *inv*) of such mappings, where the domain equals the set of all translation axes. The value tran-axis-set has elsewhere been defined as the set $\{X, Y, Z\}$. Rotation commands, *RotCommand*, is defined similarly (Line 47). Finally, a hand grip command issued by the astronaut, *SixDofCommand* (Line 48), is a composition (record) consisting of a translation command and a rotation command[3].

We started with the following prompt:

[2] The type $A \xrightarrow{m} B$ denotes in VDM the set of finite maps from A to B. This corresponds to type `Dict[A,B]` of dictionaries in Python.

[3] The construct $id :: id_1 : ty_1, ..., id_n : ty_n$ in VDM defines a record type named id, with fields $id_1, ..., id_n$ of types $ty_1, ..., ty_n$ respectively.

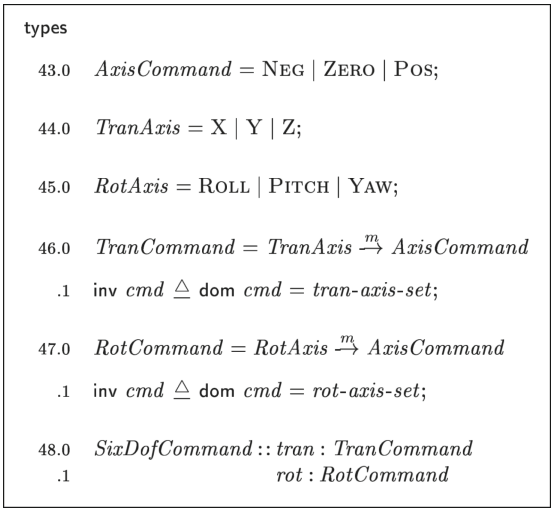


Fig. 3. The AUX module.

Prompt: I would like to translate the formal VDM spec to Python in steps. First translate the AUX module in Section 6. Translate it in one piece.

ChatGPT read and printed the VDM spec in ASCII format, which showed that it was able to read the PDF perfectly, including VDM's special mathematical symbols such as the map $\xrightarrow{m}$ symbol. It then performed a largely flawless translation to Python that compiled and was executable, although a few prompt iterations were needed to refine the result, as explained in the following. The result of the translation is shown below.

```
class AxisCommand(Enum):
    NEG = "Neg"
    ZERO = "Zero"
    POS = "Pos"

class TranAxis(Enum):
    X = "X"
    Y = "Y"
    Z = "Z"

class RotAxis(Enum):
    ROLL = "Roll"
    PITCH = "Pitch"
    YAW = "Yaw"

TranCommand = Dict[TranAxis, AxisCommand]
def is_valid_tran_command(cmd: TranCommand) -> bool:
    return set(cmd.keys()) == tran_axis_set

RotCommand = Dict[RotAxis, AxisCommand]
```

```
def is_valid_rot_command(cmd: RotCommand) -> bool:
    return set(cmd.keys()) == rot_axis_set

@dataclass
class SixDofCommand:
    tran: TranCommand
    rot: RotCommand

    def __post_init__(self):
        if not is_valid_tran_command(self.tran):
            raise ValueError(f"Invalid TranCommand")
        if not is_valid_rot_command(self.rot):
            raise ValueError(f"Invalid RotCommand")
```

VDM map types are correctly translated to Python dictionaries, and the record with two fields is converted into a Python class with two attributes. It overrides the `__post_init__` method in the class `SixDofCommand`, which is called automatically after the object is initialized, checking the subtype constraints. In development B, ChatGPT left out the definitions of the types `TranCommand` and `RotCommand`, and instead inlined these in the definition of `SixDofCommand`. Although this was correct, we asked it to name these types. In both developments, it forgot to translate the subtype (*inv*) constraints on these types. That is, the functions `is_valid_tran_command`, `is_valid_rot_command`, and `__post_init__` in the class `SixDofCommand` were not in the original translation. Finally, the class `SixDofCommand` was originally in development B defined using Python's classic, but verbose, `__init__` function. The class could be written shorter using Python's dataclass concept (a shorthand for writing `__init__` functions). So in this development we provided the following prompt:

> **Prompt:** First of all, you missed the invariant predicates. Second, you could use a dataclass for type SixDofCommand.

which resulted in the shown solution. We were generally impressed by this encouraging result of converting a VDM specification from a PDF document to Python code. Not a single line of this module implementation was programmed by hand. Note that VDM's enumeration types are more concise and elegant.

5 Module HCM

The Hand Control Module (HCM) models the hand grip controller of the SAFER system and its commands. It imports definitions from the AUX module described in Sect. 4. The module defines four important types shown in Fig. 4, which define how the astronaut operates the handgrip. The type *SwitchPositions* defines the choice between translation mode or rotation mode (*ControlModeSwitch*), and whether the AAH button is pressed down or not (*ControlButton*). Finally, the type *HandGripPosition* defines the astronaut's operation of the hand grip. Note that this represents the physical operation, while the type *SixDofCommand* from

the AUX module defines the resulting logical interpretation of that operation. Furthermore, note that *vert* is a move on the Z-axis (up, down), *horiz* is a move on the X-axis *forward, backward*, *trans* (transverse) is a move on the Y-axis (left, right), and *twist* corresponds to twisting the knob (forward, backward, always representing a PITCH rotation).

ChatGPT translates the two VDM record types to dataclasses and the two union types to Python enums as we have seen in the previous AUX module. However, in both developments ChatGPT did not import the AUX module. In development A we provided Python's error message as prompt:

> **Prompt:** Does not compile: NameError: name 'AxisCommand' is not defined.

and ChatGPT corrected it. In development B, we manually inserted this import, and type name prefixes, and fed the code back to ChatGPT. In that process, we introduced a typo, writing `aux.xisCommand.ZERO` instead of `aux.AxisCommand.ZERO`, which it detected and corrected.

```
31.0  SwitchPositions :: mode : ControlModeSwitch
  .1                     aah : ControlButton;

32.0  ControlModeSwitch = ROT | TRAN;

33.0  ControlButton = UP | DOWN;

34.0  HandGripPosition :: vert : AUX`AxisCommand
  .1                      horiz : AUX`AxisCommand
  .2                      trans : AUX`AxisCommand
  .3                      twist : AUX`AxisCommand
```

Fig. 4. The HCM module, hand grip operation types.

Figure 5 shows the main function *GripCommand* mapping the physical position of the hand grip and the control-mode switch to the corresponding logical six-degree-of-freedom command. The control-mode switch toggles between translation (*TRAN*) and rotation mode (*ROT*). The movements of the hand grip are horizontal (forward, backward), transverse (left, right), and vertical (up, down). In translation mode (*TRAN*) these grip-movements result in movements of the astronaut along the X, Y, and Z axes as shown in Fig. 1. In addition, the astronaut can twist the hand grip causing it to pitch. In rotation mode, the movements along the Y axis trigger yawing and along the Z axis trigger rolling. Table 2 summarizes the effects of the grip movements in both modes.

After resolving the issue with importing AUX, ChatGPT correctly translated *GripCommand* to Python in development B. However, in development A it intro-

```
GripCommand : HandGripPosition × ControlModeSwitch → AUX`SixDofCommand
GripCommand (mk-HandGripPosition (vert, horiz, trans, twist), mode) ≜
  let tran = {X ↦ horiz,
              Y ↦ if mode = TRAN
                  then trans
                  else ZERO,
              Z ↦ if mode = TRAN
                  then vert
                  else ZERO},
      rot = {ROLL ↦ if mode = ROT
                    then vert
                    else ZERO,
             PITCH ↦ twist,
             YAW ↦ if mode = ROT
                   then trans
                   else ZERO} in
  mk-AUX`SixDofCommand (tran, rot)
```

Fig. 5. The HCM module, function GripCommand.

Table 2. Hand grip command effects: X = move(forward, backward), Y = move(left, right), Z = move(up, down), YAW = turn(left, right), ROLL = turn(side, side), PITCH = turn(up, down).

	horiz	trans	vert	twist
TRAN	X	Y	Z	PITCH
ROT	X	YAW	ROLL	PITCH

duced a wrong logic and an additional prompt pointing this out was necessary. Here is the generated code.

```
def grip_command(hand_grip: HandGripPosition, mode: ControlModeSwitch) ->
     SixDofCommand:
    tran_command = {
        TranAxis.X: hand_grip.horiz,
        TranAxis.Y: hand_grip.trans if mode == ControlModeSwitch.TRAN
            else AxisCommand.ZERO,
        TranAxis.Z: hand_grip.vert if mode == ControlModeSwitch.TRAN
            else AxisCommand.ZERO,
    }
    rot_command = {
        RotAxis.ROLL: hand_grip.vert if mode == ControlModeSwitch.ROT
            else AxisCommand.ZERO,
        RotAxis.PITCH: hand_grip.twist,
        RotAxis.YAW: hand_grip.trans if mode == ControlModeSwitch.ROT
            else AxisCommand.ZERO,
    }
    return SixDofCommand(tran=tran_command, rot=rot_command)
```

Note that in the VDM specification, the first argument to the function is matched against *mk-HandGrip-Position(vert, horiz, trans, twist)* using pattern matching in an argument position. Pattern matching in this location is not supported in Python. In development B ChatGPT did not have great difficulties

with this module. It perfectly understood the VDM specification. In development A it hallucinated and introduced a wrong logic which, however, could be easily corrected.

6 Module AAH

This module specifies the behavior of the automatic attitude hold (AAH). Its purpose is to stop any uncontrolled rotation, which can seriously disorient an astronaut. It has state-full behavior following a protocol specified as a state-machine, with the current state stored in the variable *toggle*, and defining when it is *on* or *off* or in some transient state in between, based on the operation of the hand grip AAH button. When on, it can be turned off by two subsequent button clicks within 0.5 s. Its logic is furthermore complicated due to the priority of AAH compared to manually issued rotation commands. When active, it controls a set of orientation axes stored in the *RotAxis* set *active-axes*. The module defines two main functions: *ButtonTransition* and *Transition*.

```
ButtonTransition : EngageState × HCM`ControlButton × AUX`RotAxis-set × ℕ × ℕ
                   → EngageState
ButtonTransition (estate, button, active, clock, timeout) ≜
  cases mk-(estate, button) :
    mk-(AAH_OFF, UP) → AAH_OFF,
    mk-(AAH_OFF, DOWN) → AAH_STARTED,
    mk-(AAH_STARTED, UP) → AAH_ON,
    mk-(AAH_STARTED, DOWN) → AAH_STARTED,
    mk-(AAH_ON, UP) → if AllAxesOff (active)
                      then AAH_OFF
                      else AAH_ON,
    mk-(AAH_ON, DOWN) → PRESSED_ONCE,
    mk-(PRESSED_ONCE, UP) → AAH_CLOSING,
    mk-(PRESSED_ONCE, DOWN) → PRESSED_ONCE,
    mk-(AAH_CLOSING, UP) → if AllAxesOff (active)
                           then AAH_OFF
                           elseif clock > timeout
                           then AAH_ON
                           else AAH_CLOSING,
    mk-(AAH_CLOSING, DOWN) → PRESSED_TWICE,
    mk-(PRESSED_TWICE, UP) → AAH_OFF,
    mk-(PRESSED_TWICE, DOWN) → PRESSED_TWICE
  end
```

Fig. 6. The AAH module, function ButtonTransition.

The ButtonTransition Function. The *ButtonTransition* function[4], see Fig. 6, computes the next state of the AAH state machine, based on the current state,

[4] Some VDM fragments are red, as in the original document [4], identifying fragments that were not covered during testing of the VDM specification performed by the authors of [4].

estate, the position of the AAH *button* on the hand grip (*DOWN* when pressed and *UP* when released), the set of *active* axes that it currently controls, and a *clock* value (the time) as well as a *timeout* value, which is the clock value at which a timeout will occur during a double button click to turn the AAH off. For example (see the second and third transition), suppose the state machine is in the state *OFF*, and the astronaut pushes the button, entering the state *AAH_started*, and then releases the button, then we enter the *ON* state. Snippets of the generated Python code for this module are shown below.

```
def button_transition(estate: EngageState, button: hcm.ControlButton,
        active: Set[aux.RotAxis], clock: int, timeout: int) -> EngageState:
    match (estate, button):
        ...
        case (EngageState.AAH_OFF, hcm.ControlButton.DOWN):
            return EngageState.AAH_STARTED
        case (EngageState.AAH_STARTED, hcm.ControlButton.UP):
            return EngageState.AAH_ON
        ...
        case (EngageState.AAH_ON, hcm.ControlButton.UP):
            if clock > timeout:
                return EngageState.AAH_OFF
            else:
                return EngageState.AAH_ON
        ...
        case (EngageState.AAH_CLOSING, hcm.ControlButton.UP):
            return EngageState.AAH_OFF if not active else (
              EngageState.AAH_ON if clock > timeout else EngageState.AAH_CLOSING)
        ...
```

In development B, it first wrongly converted the type *HCM'ControlButton* of the first argument to `aux.ControlButton`. Also, in the same development, the first translation generated `if`-statements, rather than `match`-`case` statements. An interesting observation here is that there were "missing" explicit cases compared to the VDM specification: it had cleverly detected all the cases where the returned value is the incoming `estate` value, and captured these cases instead with a default '`return estate`' statement at the end. However, we asked ChatGPT also in these cases to generate `case` statements, similar to the VDM specification.

In the same development, the case `(AAH_ON, UP)` was missing. We informed ChatGPT about this and it generated a case returning the `AAH_ON` state unconditionally. However, this is wrong, if `active` is empty, corresponding to the AAH module not controlling any axes, it should transition to the `AAH_OFF` state, which we pointed out. As a result it attempted to correct the mistake as shown in the above final code. However, this is also wrong, and was only detected using differential testing as described in Sect. 9.3.

The Transition Function. The VDM function *Transition* shown in Fig. 7 updates the state *toggle* of the state machine as well as the variables *active-axes* and a variable *ignore-hcm* (to be explained), and a *timeout* in case the button has been pressed once. By default, when turned on, AAH controls all the three rotation axes. However, if during AAH mode the astronaut commands a rotation axis, that command takes over and deactivates the AAH on that axis

$$
\begin{array}{l}
Transition : HCM`ControlButton \times AUX`SixDofCommand \times \mathbb{N} \xrightarrow{o} () \\
Transition\,(button\text{-}pos, hcm\text{-}cmd, clock) \triangleq \\
\quad \textsf{let}\ engage = ButtonTransition\,(toggle, button\text{-}pos, active\text{-}axes, clock, timeout), \\
\qquad starting = (toggle = \text{AAH\_OFF}) \wedge (engage = \text{AAH\_STARTED})\ \textsf{in} \\
\quad (active\text{-}axes := \{a \mid a \in AUX`rot\text{-}axis\text{-}set \cdot \\
\qquad\qquad\qquad\qquad starting \vee \\
\qquad\qquad\qquad\qquad (engage \neq \text{AAH\_OFF} \wedge a \in active\text{-}axes \wedge \\
\qquad\qquad\qquad\qquad\ (hcm\text{-}cmd.rot\,(a) = \text{ZERO} \vee a \in ignore\text{-}hcm))\}; \\
\qquad ignore\text{-}hcm := \{a \mid a \in AUX`rot\text{-}axis\text{-}set \cdot \\
\qquad\qquad\qquad\qquad (starting \wedge hcm\text{-}cmd.rot\,(a) \neq \text{ZERO}) \vee \\
\qquad\qquad\qquad\qquad (\neg\, starting \wedge a \in ignore\text{-}hcm)\}; \\
\qquad timeout := \textsf{if}\ toggle = \text{AAH\_ON} \wedge engage = \text{PRESSED\_ONCE} \\
\qquad\qquad\qquad \textsf{then}\ clock + click\text{-}timeout \\
\qquad\qquad\qquad \textsf{else}\ timeout; \\
\qquad toggle := engage);
\end{array}
$$

Fig. 7. The AAH module, function Transition.

(it is removed from *active-axes*). This is a truth, with a modification. In case the astronaut is already commanding a rotation axis when turning on the AAH, that and subsequent commands on that rotation axis, will be ignored until the AAH is turned *OFF* (modeling that the astronaut wants to cancel that command by turning AAH on). Such axes are stored in the variable *ignore-hcm*. The function is by ChatGPT translated as follows, after a few interactions with ChatGPT that we will discuss. The translation is nearly perfect, translating set comprehensions correctly.

```
def transition(button_pos: hcm.ControlButton, hcm_cmd: aux.SixDofCommand, clock: int):
    engage = button_transition(AAH.toggle, button_pos,
                               AAH.active_axes, clock, AAH.timeout)
    starting = AAH.toggle == EngageState.AAH_OFF and
               engage == EngageState.AAH_STARTED
    AAH.active_axes = {
        a for a in aux.rot_axis_set if starting or
            (engage != EngageState.AAH_OFF and
             a in AAH.active_axes and
             (hcm_cmd.rot[a] == aux.AxisCommand.ZERO or a in AAH.ignore_hcm))
    }
    AAH.ignore_hcm = {
        a for a in aux.rot_axis_set if
          (starting and hcm_cmd.rot[a] != aux.AxisCommand.ZERO)
          or
          (not starting and a in AAH.ignore_hcm)
    }
    if AAH.toggle == EngageState.AAH_ON and engage == EngageState.PRESSED_ONCE:
        AAH.timeout = clock + CLICK_TIMEOUT
    AAH.toggle = engage
```

Some minor points in development B was that the global state was passed as an object to the function, and the function then updated this object as a side-effect. This approach is not wrong, but does not quite follow the VDM specification. In this module we observed most of the hallucinations, like in development A, missing arguments. In that development, we had to provide the

technical report again, which partly resolved the issues. We also realized that ChatGPT produced code that was different from the VDM version, which made it more difficult to review. For example, it used different parameter and variable names. The lesson we learned was to keep the code as close as possible to the VDM style to ease reviewing. Hence, we had to backtrack to previous versions several times. At one point, we had to make a manual correction. Reviewing of the translated code was harder compared to the previous modules. The hope was to find any remaining issues during the runtime checking.

7 Module TS

The TS module models the Thruster Selection logic. Hand Controller and AHH commands are merged together in accordance with the various priority rules. The result is that a six-degree-of-freedom command is mapped to a set of thrusters to be fired. Thruster selection tables are used to convert a command to individual actuator commands for opening suitable thruster valves. The development of this module was surprisingly difficult. For example, in development A, we needed 64 prompts to produce the correct 226 lines of Python code. In development A we always let ChatGPT generate a `main` function to quickly check the sanity of the generated code before review (development B relied on reviewing only). In this case ChatGPT dropped the `main` function for testing. Furthermore, it introduced undefined names, arbitrarily renamed existing ones, changed the logical structure, and omitted parts. A simple prompt with these observations did not help and we had to upload the PDF with the VDM specification again. It obviously lost the context. Unfortunately, this did not resolve all issues and we decided to develop this module gradually, function by function. First, we let ChatGPT translate the union type of all 24 thruster names into an `Enum` class.

```
class ThrusterName(Enum):
    B1 = auto()
    B2 = auto()
    B3 = auto()
    ...

def main():
    # Demonstrate usage of ThrusterName enum
    print("List of Thruster Names:")
    for thruster in ThrusterName:
        print(f"{thruster.name}: {thruster.value}")

if __name__ == "__main__":
    main()
```

The RotCmdsPresent Function. Next, we added the Boolean function *RotCmdsPresent* shown in Fig. 8. It is an example of how a predicate with an existential quantifier gets translated to Python. First, a different parameter name was generated and we asked ChatGPT to correct this. Furthermore, the type `RotCommand` in the type signature was newly created, since ChatGPT did not realize that it is defined in the AUX-module. After two prompts with these corrections, the function was correctly translated.

$$
\begin{array}{l}
RotCmdsPresent : AUX`RotCommand \to \mathbb{B} \\
RotCmdsPresent\,(cmd) \triangleq \\
\quad \exists\, a \in \mathsf{dom}\ cmd \cdot cmd\,(a) \neq \textsc{Zero};
\end{array}
$$

Fig. 8. The TS module function *RotCmdsPresent*.

```
def RotCmdsPresent(cmd: RotCommand) -> bool:
    return any(cmd[axis] != AxisCommand.ZERO for axis in RotAxis)
```

The existential quantifier got translated into Python's `any`-function that takes a predicate and an iterable type and returns true if the predicate holds for any element. Note how ChatGPT optimized the function. Instead of VDM's domain operator *dom* it uses the enum type RotAxis which is the domain type of the map RotCommand. In this case it is correct, since the invariant in the AUX module requires that the mapping is total. Hence, *dom cmd* is equivalent to `RotAxis`. It is really interesting that ChatGPT can do this kind of code optimization during translation.

The PrioritizedTranCmd Function. This function specifies the priorities of the translational commands along the X, Y, and Z axes. The translation was fairly straightforward. In development A, an additional prompt was needed to correct the return type and to use the parameter name `tran`. Here is the resulting code (Fig. 9).

$$
\begin{array}{l}
PrioritizedTranCmd : AUX`TranCommand \to AUX`TranCommand \\
PrioritizedTranCmd\,(tran) \triangleq \\
\quad \mathsf{if}\ tran\,(\mathrm{X}) \neq \textsc{Zero} \\
\quad \mathsf{then}\ AUX`null\text{-}tran\text{-}command \dagger \{\mathrm{X} \mapsto tran\,(\mathrm{X})\} \\
\quad \mathsf{elseif}\ tran\,(\mathrm{Y}) \neq \textsc{Zero} \\
\quad \mathsf{then}\ AUX`null\text{-}tran\text{-}command \dagger \{\mathrm{Y} \mapsto tran\,(\mathrm{Y})\} \\
\quad \mathsf{elseif}\ tran\,(\mathrm{Z}) \neq \textsc{Zero} \\
\quad \mathsf{then}\ AUX`null\text{-}tran\text{-}command \dagger \{\mathrm{Z} \mapsto tran\,(\mathrm{Z})\} \\
\quad \mathsf{else}\ AUX`null\text{-}tran\text{-}command;
\end{array}
$$

Fig. 9. The TS module, function PrioritizedTranCmd.

```
def PrioritizedTranCmd(tran: TranCommand) -> TranCommand:
    prioritized_tran = null_tran_command.copy()

    if tran[TranAxis.X] != AxisCommand.ZERO:
        prioritized_tran[TranAxis.X] = tran[TranAxis.X]
    elif tran[TranAxis.Y] != AxisCommand.ZERO:
        prioritized_tran[TranAxis.Y] = tran[TranAxis.Y]
```

```
    elif tran[TranAxis.Z] != AxisCommand.ZERO:
        prioritized_tran[TranAxis.Z] = tran[TranAxis.Z]

    return prioritized_tran
```

Note that it saved an if-branch by applying an imperative style. In development B, it first came up with a version that spells out the returned map, in fact inlining the values.

```
def prioritized_tran_cmd(tran: aux.TranCommand) -> aux.TranCommand:
    if tran[aux.TranAxis.X] != aux.AxisCommand.ZERO:
        return {aux.TranAxis.X: tran[aux.TranAxis.X],
                aux.TranAxis.Y: aux.AxisCommand.ZERO,
                aux.TranAxis.Z: aux.AxisCommand.ZERO}
    elif tran[aux.TranAxis.Y] != aux.AxisCommand.ZERO:
        return {aux.TranAxis.X: aux.AxisCommand.ZERO,
                aux.TranAxis.Y: tran[aux.TranAxis.Y],
                aux.TranAxis.Z: aux.AxisCommand.ZERO}
    elif tran[aux.TranAxis.Z] != aux.AxisCommand.ZERO:
        return {aux.TranAxis.X: aux.AxisCommand.ZERO,
                aux.TranAxis.Y: aux.AxisCommand.ZERO,
                aux.TranAxis.Z: tran[aux.TranAxis.Z]}
    else:
        return aux.null_tran_command
```

It is functionally correct, but this is not following the VDM specification, which updates the *null-tran-command* variable. Three prompts were needed to obtain satisfactory code, which is almost the same as development A.

The CombinedRotCmds Function. As shown in Fig. 10, this function uses sophisticated map operators and ChatGPT had difficulties interpreting them. In development A, we needed six prompts to develop this function. First, the number of arguments was wrong, then parameter names were mixed up and the logic was completely made up. Here is a prompt from development B:

$$
\begin{aligned}
&\mathit{CombinedRotCmds} : \mathit{AUX`RotCommand} \times \mathit{AUX`RotCommand} \times \mathit{AUX`RotAxis}\text{-set} \to \\
&\qquad \mathit{AUX`RotCommand} \\
&\mathit{CombinedRotCmds}\,(\mathit{hcm\text{-}rot}, \mathit{aah}, \mathit{ignore\text{-}hcm}) \triangleq \\
&\quad \mathsf{let}\ \mathit{aah\text{-}axes} = \mathit{ignore\text{-}hcm} \cup \\
&\qquad\qquad \{a \mid a \in \mathit{AUX`rot\text{-}axis\text{-}set} \cdot \mathit{hcm\text{-}rot}\,(a) = \textsc{Zero}\}\ \mathsf{in} \\
&\quad \{a \mapsto \mathit{aah}\,(a) \mid a \in \mathit{aah\text{-}axes}\} \cup \\
&\quad \{a \mapsto \mathit{hcm\text{-}rot}\,(a) \mid a \in \mathit{AUX`rot\text{-}axis\text{-}set} \setminus \mathit{aah\text{-}axes}\};
\end{aligned}
$$

Fig. 10. The TS module, function CombinedRotCmds.

Prompt: Let me explain what happens in that function. aah-axes is ignore_hcm (which is a set) union the set of a in aux.rot-axis-set for which hcm-rot(a) is ZERO. This set is stored in aah-axes. Then what is returned is a map composed of two submaps: (1) a to aah(a) for a in aah-axes and (2) a to hcm-rot(a) for a in aux.rot-axis-set minus aah-axes.

It was a battle to make it follow the VDM specification. The result was as follows. Note that dictionaries are mutable, therefore this slightly obscure notation {**aah_cmd_map, **hcm_cmd_map} for creating a new set that is the union of two existing sets.

```
def combined_rot_cmds(hcm_rot: aux.RotCommand, aah: aux.RotCommand,
                      ignore_hcm: Set[aux.RotAxis]) -> aux.RotCommand:
    aah_axes = ignore_hcm.union({
                  a for a in aux.rot_axis_set if hcm_rot[a] == aux.AxisCommand.ZERO})
    aah_cmd_map = {a: aah[a] for a in aah_axes}
    hcm_cmd_map = {a: hcm_rot[a] for a in aux.rot_axis_set if a not in aah_axes}
    return {**aah_cmd_map, **hcm_cmd_map}
```

It did come up with an alternative correct translation, which shows "understanding" of the VDM specification, in particular set theory.

```
def combined_rot_cmds(hcm_rot: aux.RotCommand, aah: aux.RotCommand,
                      ignore_hcm: Set[aux.RotAxis]) -> aux.RotCommand:
    aah_axes = ignore_hcm.union({
                  a for a in aux.rot_axis_set if hcm_rot[a] == aux.AxisCommand.ZERO})
    combined_cmd = {
        a: aah[a] if a in aah_axes else hcm_rot[a]
        for a in aux.rot_axis_set
    }
    return combined_cmd
```

The BfThrusters Function. As shown in Fig. 11, this function maps commands to the set of thrusters to be fired. More precisely, a pair of sets is produced, the first being the set of mandatory thrusters, the second set contains optional thrusters. This function was surprisingly difficult to develop. A simple precise mapping appeared to be the greatest challenge for ChatGPT. It repeatedly mixed up the thruster names. It even changed already correct ones back to incorrect ones, leading to the prompt:

Prompt: oh gosh, you deleted the correct ones, please go back to the previous version and only correct the ones provided above explicitly as wrong clauses.

In development A we created a table explicitly via a prompt, making up a domain-specific language for this purpose:

```
BFThrusters : AUX'AxisCommand × AUX'AxisCommand × AUX'AxisCommand →
              ThrusterSet × ThrusterSet
BFThrusters (A, B, C) ≜
  cases mk-(A, B, C) :
    mk-(NEG, NEG, NEG) → mk-({B4}, {B2, B3}),
    mk-(NEG, NEG, ZERO) → mk-({B3, B4}, {}),
    mk-(NEG, NEG, POS) → mk-({B3}, {B1, B4}),
    mk-(NEG, ZERO, NEG) → mk-({B2, B4}, {}),
    mk-(NEG, ZERO, ZERO) → mk-({B1, B4}, {B2, B3}),
```

Fig. 11. The TS module, function BfThrusters (only the first 5 of 27 cases shown).

Prompt: Here is the functionality for the POS combinations:
Pos, Neg, Neg → F1 and F2,F3
Pos, Neg, Zero → F1,F2 and empty
Pos, Neg, Pos → F2 and F1, F4
...

ChatGPT understands from the context that each line describes a mapping into two sets including our own syntax for empty sets. This is a strength of LLMs. If needed, we can program in a more flexible notation than in classical programming languages. Actually, we can invent syntax as we go as long as it guesses the correct semantics.

8 Module SAFER

This is the top-level module, which models the system's transition that occurs during one cycle (*"once around the main control loop"* [4]) of the controller. The translation of this module did not go smoothly either. ChatGPT seemed confused about what to translate, and we (in both developments) had to upload the technical report again. In development B ChatGPT first went object-oriented, defining the entire module as a class, in contrast to previous modules which were defined in a procedural manner similar to the VDM specification. We will describe the handling of its two functions *ControlCycle* and *ThrusterConsistency*.

The ControlCycle Function. The *ControlCycle* function is shown in Fig. 12. It takes three arguments (see Sect. 5 for the types). First, the switch positions on the hand controller, indicating whether we are in translation or rotation mode, and whether the AAH button is up or down. Second, the physical hand grip moves on the X, Y, and Z axes and twisting of the knob. The third argument represents the movements performed by the automated AAH module itself, assumed measured by sensors. Recall that the AAH module, when active, sends commands to the rotation axes.

The function computes and returns the set of thrusters to be activated. It first computes the *logical* hand grip command (of type *SixDofCommand*) from the *physical* move (*raw_grip*) made by the astronaut, depending on whether we are in translation or rotation mode. Then the thrusters are computed (see Sect. 7). Finally, the AAH performs its job (see Sect. 6). The two assertions at the end check that at most four thrusters are now active and that they are consistent, see below.

$$
\begin{array}{l}
ControlCycle : HCM`SwitchPositions \times HCM`HandGripPosition \times AUX`RotCommand \xrightarrow{o} \\
\qquad TS`ThrusterSet \\
ControlCycle\,(\mathsf{mk\text{-}}HCM`SwitchPositions\,(mode, aah), raw\text{-}grip, aah\text{-}cmd) \triangleq \\
\quad \mathsf{let}\ grip\text{-}cmd = HCM`GripCommand\,(raw\text{-}grip, mode), \\
\qquad thrusters = TS`SelectedThrusters\,(grip\text{-}cmd, aah\text{-}cmd, AAH`ActiveAxes\,(), AAH`IgnoreHcm\,()))\ \mathsf{in} \\
\quad (AAH`Transition(aah, grip\text{-}cmd, clock)\ ; \\
\quad\ clock := clock + 1; \\
\quad\ \mathsf{return}\ thrusters\ ) \\
\mathsf{post\ card}\ RESULT \leq 4 \wedge \\
\qquad ThrusterConsistency\,(RESULT)
\end{array}
$$

Fig. 12. The SAFER module, function ControlCycle.

The translation of the function is shown below. It illustrates how ChatGPT "understands" the meaning of VDM's '**let** ... **in** *(...)*' construct, simply translating it to assignment statements. It also illustrates how it translates a post-condition to `assert`-statements. Both developments encountered problems with getting names right, for example generating the right types for arguments.

```
def control_cycle(switch_positions: hcm.SwitchPositions, raw_grip:
     hcm.HandGripPosition, aah_cmd: aux.RotCommand) -> ts.ThrusterSet:
    grip_cmd = hcm.grip_command(raw_grip, switch_positions.mode)
    thrusters = ts.selected_thrusters(grip_cmd, aah_cmd, aah.active_axes(),
                                      aah.ignore_hcm())
    aah.transition(switch_positions.aah, grip_cmd, SAFER.clock)
    SAFER.clock += 1
    assert len(thrusters) <= 4
    assert thruster_consistency(thrusters), thrusters
    return thrusters
```

The ThrusterConsistency Function. The function *ThrusterConsistency* presented in Fig. 13 checks thruster consistency: no two selected thrusters should oppose each other, i.e. cancel each other out by thrusting in opposite directions. The translation of the *ThrusterConsistency* function is interesting in the way ChatGPT translated the slightly complicated set operations in the two developments. In development B, ChatGPT generated the following code[5].

[5] The `&` operator in Python performs set intersection. Furthermore, a set is `True` in a Boolean context iff. it is non-empty.

$$
\begin{array}{l}
ThrusterConsistency : TS`ThrusterName\text{-set} \to \mathbb{B} \\
ThrusterConsistency\,(thrusters) \triangleq \\
\quad \neg\,(\{\mathrm{B1},\mathrm{F1}\} \subseteq thrusters) \wedge \\
\quad \neg\,(\{\mathrm{B2},\mathrm{F2}\} \subseteq thrusters) \wedge \\
\quad \neg\,(\{\mathrm{B3},\mathrm{F3}\} \subseteq thrusters) \wedge \\
\quad \neg\,(\{\mathrm{B4},\mathrm{F4}\} \subseteq thrusters) \wedge \\
\quad \neg\,(thrusters \cap \{\mathrm{L1R},\mathrm{L1F}\} \neq \{\} \wedge thrusters \cap \{\mathrm{R2R},\mathrm{R2F}\} \neq \{\}) \wedge \\
\quad \neg\,(thrusters \cap \{\mathrm{L3R},\mathrm{L3F}\} \neq \{\} \wedge thrusters \cap \{\mathrm{R4R},\mathrm{R4F}\} \neq \{\}) \wedge \\
\quad \neg\,(thrusters \cap \{\mathrm{D1R},\mathrm{D1F}\} \neq \{\} \wedge thrusters \cap \{\mathrm{U3R},\mathrm{U3F}\} \neq \{\}) \wedge \\
\quad \neg\,(thrusters \cap \{\mathrm{D2R},\mathrm{D2F}\} \neq \{\} \wedge thrusters \cap \{\mathrm{U4R},\mathrm{U4F}\} \neq \{\})
\end{array}
$$

Fig. 13. The SAFER module, function ThrusterConsistency.

```
def thruster_consistency(thrusters: Set[ts.ThrusterName]) -> bool:
    if {ts.ThrusterName.B1, ts.ThrusterName.F1}.issubset(thrusters):
        return False
    ...
    if ({ts.ThrusterName.L1R, ts.ThrusterName.L1F} & thrusters) and
       ({ts.ThrusterName.R2R, ts.ThrusterName.R2F} & thrusters):
        return False
    ...
    return True
```

The first four statements (of which the first is shown before the first dots ...) were not translated correctly at first, since instead of the `.subset` method it used `&` (set intersection). This was not detected during interaction with ChatGPT, but was later detected during execution due to a violation of the thruster consistency assertion in the `control_cycle` function. In development A we observed a quite different, and clever, translation as shown below.

```
def ThrusterConsistency(self, thrusters: ThrusterSet) -> bool:
    opposing_pairs = [
        ({"B1"}, {"F1"}),  # Back vs Front
        ...
        ({"L1R", "L1F"}, {"R2R", "R2F"}),  # Left vs Right
        ...
    ]
    for left, right in opposing_pairs:
        if left & thrusters and right & thrusters:
            return False
    return True
```

The Python code identifies first pairs of sets of opposing thrusters and then checks for the intersection of the activated thrusters with these sets. The translation in addition performed a pre-condition check on the number of thrusters being no more than 4, which we have removed, since this check is done elsewhere. However, also for this development there were errors in the logic for checking opposing thrusters, which was caught by an invariant violation at execution time.

9 Module TEST

This module contains functions to test the VDM specification. We translated these functions to Python for testing the Python code. However, we also generated classical unit tests and performed differential testing as described below.

9.1 Unit Testing

In development A, we asked ChatGPT how to test the SAFER code. It recommended two options: (1) unit testing and (2) property-based testing. We first asked for some unit tests. As a result it produced three unit tests, like the following.

```
class TestSAFERSystem(unittest.TestCase):
    def setUp(self):
        self.safer = SAFER()

    def test_control_button_up(self):
        switch_positions = SwitchPositions(
            mode=ControlModeSwitch.ROT,
            aah=ControlButton.UP
        )
        aah_cmd = {
            RotAxis.ROLL: AxisCommand.NEG,
            RotAxis.PITCH: AxisCommand.POS,
            RotAxis.YAW: AxisCommand.ZERO
        }
        raw_grip = HandGripPosition(AxisCommand.ZERO, AxisCommand.POS,
                                    AxisCommand.NEG, AxisCommand.ZERO)
        thrusters = self.safer.ControlCycle(switch_positions, raw_grip, aah_cmd)
        self.assertTrue(self.safer.ThrusterConsistency(thrusters),
            "Invariant violated for ControlButton.UP")
```

This unit test first sets the inputs of the hand controller, namely the switch positions and the hand grip position. It also defines a command from the AAH. Then, it calls the control cycle function to produce the selected thrusters. These thrusters are then evaluated for consistency in an assertion. Note, that this assertion is actually redundant, since it is already checked in each control cycle.

After a few iterations of producing unit tests with ChatGPT it became obvious that this method does not scale, given the many combinations of input values. Hence, we continued with the testing method described in the VDM specification of this module in [4].

9.2 Testing in VDM Style

The VDM tooling supports its own style of testing. It exploits the ability of the interpreter to process set comprehensions. These set comprehensions can then be used to generate test inputs in a very concise manner. The TS module defines such input sets for SAFER.

In development B, ChatGPT again had “lost” the document, and it was uploaded again. Once it finally found the section in the newly uploaded report, it produced a quite impressive translation of the whole module, which had 129 lines

of non-trivial VDM specification (ignoring comments and blank lines), covering 7 definitions of constants (5 of which were set comprehensions), and 10 functions. There were some minor issues and observations that we will go through in the following.

Position Datatypes. Figure 14 shows definitions of the sets *switch-positions*, *all-grip-positions*, and *all-rot-commands*. These are defined as set comprehensions and serve as test inputs. In the translation to Python these were translated to list comprehensions.

$$
\begin{aligned}
&\mathit{switch\text{-}positions} = \{\mathsf{mk\text{-}}\mathit{HCM`SwitchPositions}\,(\mathit{mode}, \mathit{aah}) \mid \\
&\qquad \mathit{mode} \in \{\textsc{Tran}, \textsc{Rot}\}, \mathit{aah} \in \{\textsc{Up}, \textsc{Down}\}\};\\
&\mathit{zero\text{-}grip} = \mathsf{mk\text{-}}\mathit{HCM`HandGripPosition}\,(\textsc{Zero}, \textsc{Zero}, \textsc{Zero}, \textsc{Zero});\\
&\mathit{all\text{-}grip\text{-}positions} = \{\mathsf{mk\text{-}}\mathit{HCM`HandGripPosition}\,(\mathit{vert}, \mathit{horiz}, \mathit{trans}, \mathit{twist}) \mid \\
&\qquad \mathit{vert}, \mathit{horiz}, \mathit{trans}, \mathit{twist} \in \mathit{AUX`axis\text{-}command\text{-}set}\};\\
&\mathit{all\text{-}rot\text{-}commands} = \{\{\textsc{Roll} \mapsto a, \textsc{Pitch} \mapsto b, \textsc{Yaw} \mapsto c\} \mid \\
&\qquad a, b, c \in \mathit{AUX`axis\text{-}command\text{-}set}\};
\end{aligned}
$$

Fig. 14. The TEST module, Positions.

```
switch_positions = [hcm.SwitchPositions(mode, aah)
    for mode in [hcm.ControlModeSwitch.TRAN, hcm.ControlModeSwitch.ROT]
    for aah in [hcm.ControlButton.UP, hcm.ControlButton.DOWN]
]
...
all_rot_commands = [{aux.RotAxis.ROLL: a, aux.RotAxis.PITCH: b, aux.RotAxis.YAW: c}
    for a in aux.axis_command_set
    for b in aux.axis_command_set
    for c in aux.axis_command_set
]
```

ChatGPT was then asked to turn this into sets, to match the VDM specification, which it did. However, during subsequent execution an exception was thrown since `SwitchPositions`, defined as a dataclass, is an unhashable type, meaning it cannot be used as an element type of a set. In Python, only immutable types (e.g., numbers, strings, tuples) are hashable by default. To fix this, we could make the class hashable by defining a `__hash__` method and, optionally, an equality method `__eq__` to ensure the correct behavior.

```
@dataclass(frozen=True)
class SwitchPositions:
    mode: ControlModeSwitch
    aah: ControlButton

    def __hash__(self):
        return hash((self.mode, self.aah))
```

The translation of *all-rot-commands* as a set of dictionaries resulted in a similar problem in that dictionaries are mutable, and therefore not hashable, and therefore cannot occur as elements in sets. ChatGPT repeated the correct suggestion to use lists instead of sets, as shown above, to avoid this problem.

```
69.0  HugeTest : () →
 .1                (HCM`SwitchPositions × HCM`HandGripPosition × AUX`RotCommand) m→
TS`ThrusterSet
 .2  HugeTest () ≜
 .3    {mk- (switch, grip, aah-law) ↦ SAFER`ControlCycle (switch, grip, aah-law) |
 .4          switch ∈ switch-positions, grip ∈ all-grip-positions,
 .5          aah-law ∈ all-rot-commands};
```

Fig. 15. The TEST module, Function HugeTest.

The HugeTest Function. The *HugeTest* function shown in Fig. 15 is interesting since it shows a "clever" way of specifying property-based testing [10] in VDM. Given a function $f : D_1 \to D_2$ and a property that should hold, e.g. $consistent(f(x))$, property-based testing consists of testing for some finite subset $F \subseteq D_1$:

$$\forall x \in F \bullet consistent(f(x))$$

This can instead be expressed as a set comprehension:

$$\{x \mapsto f(x) \mid x \in F\}$$

assuming that f will throw exceptions in case the *consistent* property is violated. In VDM this can be achieved via assertions. This is what happens in the function *ControlCycle* in Fig. 15. The property to be tested here will be thruster consistency. ChatGPT translated this into the following.

```
def huge_test() -> Dict[
                    Tuple[hcm.SwitchPositions, hcm.HandGripPosition, aux.RotCommand],
                    ts.ThrusterSet]:
    return {
        (switch, grip, aah_law): safer.control_cycle(switch, grip, aah_law)
        for switch in switch_positions
        for grip in all_grip_positions
        for aah_law in all_rot_commands
    }
```

However, again we ran into the problem of non-hashable types, detected by executing the code. The types `hcm.SwitchPositions`, `hcm.HandGripPosition`, and `aux.RotCommand` are not hashable, and hence tuples of such cannot be used as keys in a dictionary. This code was therefore generated using a list of tuples.

```
def huge_test() -> list[Tuple[
                    Tuple[hcm.SwitchPositions,hcm.HandGripPosition,aux.RotCommand],
                    ts.ThrusterSet]]:
    return [
        ((switch, grip, aah_law), safer.control_cycle(switch, grip, aah_law))
        for switch in switch_positions
        for grip in all_grip_positions
        for aah_law in all_rot_commands
    ]
```

$$
\begin{array}{l}
\mathit{ConvertTIds} : \mathit{TS}`\mathit{ThrusterSet} \to \mathsf{char}^{**} \\
\mathit{ConvertTIds}\,(\mathit{ts}) \triangleq \\
\quad \mathsf{if}\ \mathit{ts} = \{\} \\
\quad \mathsf{then}\ [] \\
\quad \mathsf{else\ let}\ t \in \mathit{ts}\ \mathsf{in} \\
\qquad [\mathit{ConvertTId}\,(t)] \curvearrowright \mathit{ConvertTIds}\,(\mathit{ts} \setminus \{t\});
\end{array}
$$

Fig. 16. The TEST module, function ConvertTIds.

The ConvertTIds Function. The last function we shall mention is *ConvertTIds*, shown in Fig. 16. This function returns a list of *ConvertTId(t)* for all t in the argument *ts*, in no specified order. It is defined in a non-deterministic manner, selecting a t from *ts* and then calls itself recursively. This is translated into the following list comprehension, which iterates `t` through `ts` in order. It requires some "understanding" of this recursive approach in Fig. 16 to perform this translation.

```
def convert_thruster_ids(thrusters: ts.ThrusterSet) -> List[str]:
  return [convert_thruster_id(tnm) for tnm in thrusters]
```

The two solutions are not equivalent due to the non-deterministic/underspecified choice in the VDM specification. However, this difference seems not important.

Test Results. With *HugeTest* we discovered a bug in development A. After execution of 4654 tests, the property of thruster consistency was broken. The fault was in the TS module, where wrong thrusters were selected. It turned out that ChatGPT had changed a part of the Python code after it was reviewed. Hence, it had changed already correct code. After fixing the code all 8748 tests passed.

Note, with only a handful of unit tests it would have been very unlikely that we would have discovered this bug. It demonstrates how important automated test case generation is. Nobody wants to write over 8000 unit tests by hand, even if assisted by an LLM. With the same method, we also detected a bug in the thruster consistency definition of development B. We refer to the discussion of the *ThrusterConsistency* function in Sect. 8 for the details.

9.3 Differential Testing

Having established thruster consistency, one may still wonder if the system is actually firing the correct thrusters. An obvious method would be to test against the VDM specification. However, we had only the PDF document available. Hence, we decided for differential testing of the two separate Python developments. That is, we defined a property that the two versions of SAFER must produce the same output. Then, we used the same property-based testing method as for thruster consistency. The only problem was that we had to convert the enum return types, because the developments used different encodings for enums. ChatGPT came up with an immediate solution for this conversion.

Test Results. *HugeTest* discovered a difference in the outputs after 4377 tests. The problem was in the AAH module in development B, as already mentioned on page 12: the transition for the `(AAH_ON, UP)` case in the AAH state machine was wrong. After fixing this fault, all 8748 test cases passed. Hence, both separate developments produced the same output. This demonstrates the value of differential testing.

10 Lessons Learned

We can make a number of observations based on this exercise. First of all, we have generated code from a formal abstract mathematical specification with the use of an LLM. In general, one may expect that it is easier to review and understand an abstract specification than it is to understand the generated code, making formal specification a possible approach to advanced prompting.

In this case, however, the generated code is very similar to the formal specification due to the similarity between VDM and Python, as we have shown, which is interesting in itself. Some key differences include the following. VDM allows for infinite sets and universal and existential quantification over infinite sets, which Python does not. Maps in VDM are immutable, but dictionaries in Python are mutable, and not hashable, which means that one cannot create e.g. a set of dictionaries. VDM supports design-by contract in the form of pre- and post-conditions and state invariants, intended to hold between state updates, which Python does not. A minor issue is Python's very verbose way of expressing enumerated types. A not unimportant, although semantically shallow, difference is that VDM appears visually more elegant due to the rendering of its mathematical symbols. Python is code, with the usual rendering of code using colors. The Fortress programming language [13] tried to focus on mathematical rendering as well. Python's very flexible meta-programming features may be used to remove some of these differences.

With respect to the use of LLMs for code generation, we of course observed that the LLM occasionally hallucinated. ChatGPT even changed already correct code parts when other parts were being created or corrected. We observed

that error messages can be used directly as prompts for correction. In particular, assertions become very important, and also make it easier to review code. A useful technique turned out to be to formulate a prompt in an informal free, but structured, notation similar to pseudo code or a DSL (Domain-Specific Language).

With respect to verifying generated code, AI-assisted pair programming, with separate development and equivalence runtime verification checks seems interesting. Also property-based testing, where tests are generated, are important. E.g. in some cases bugs were caught after several thousands of tests, which would have been impossible to create manually. Finally, an LLM can be used for code (and specification) explanation. We even used this technique to better understand the SAFER protocol, using the documents [3,4,11], including the formal VDM specification in [4], as prompts.

References

1. Abadi, M., Lamport, L.: The existence of refinement mappings. Theor. Comput. Sci. **82**, 253–284 (1991)
2. Abrial, J.: Modeling in Event-B - System and Software Engineering. Cambridge University Press, Cambridge, 2010
3. Agerholm, S., Larsen, P.G.: Modeling and validating SAFER in VDM-SL. In: Lfm'97 - Fourth NASA Langley Formal Methods Workshop, 1997
4. Agerholm, S., Larsen, P.G.: SAFER specification in VDM-SL. Technical report, IFAD, Forskerparken 10, Odense, Denmark, 1997
5. Aichernig, B.K., Havelund, K.: AI-assisted programming with test-based refinement. In: Steffen, B. (eds.) Bridging the Gap Between AI and Reality. AISoLA 2023. LNCS, vol. 14129, pp. 385–411. Springer, Cham (2025). https://doi.org/10.1007/978-3-031-73741-1_24
6. Avizienis, A.A.: The methodology of N-version programming. Softw. Fault Toler. (1995)
7. Bartocci, E., Falcone, Y., Francalanza, A., Reger, G.: Introduction to runtime verification. In: Bartocci, E., Falcone, Y. (eds.) Lectures on Runtime Verification. LNCS, vol. 10457, pp. 1–33. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-75632-5_1
8. Bjørner, D., Jones, C.B. (eds.): The Vienna Development Method: The Meta-Language. LNCS, vol. 61. Springer, Heidelberg (1978). https://doi.org/10.1007/3-540-08766-4
9. ChatGPT. https://chat.openai.com
10. Claessen, K., Hughes, J.: QuickCheck: a lightweight tool for random testing of Haskell programs. In: Proceedings of the Fifth ACM SIGPLAN International Conference on Functional Programming, ICFP '00, pp. 268–279. ACM, New York, NY, USA, 2000
11. Crow, J., et al.: Formal methods specification and analysis guidebook for the verification of software and computer systems, volume II: A practitioner's companion. Technical Report NASA-GB-O01-97, NASA, 1997
12. Fitzgerald, J., Larsen, P.G., Mukherjee, P., Plat, N., Verhoef, M.: Validated Designs For Object-oriented Systems, TELOS. Springer-Verlag, Santa Clara, CA, USA (2005). https://doi.org/10.1007/b138800

13. Fortress. https://en.wikipedia.org/wiki/Fortress_(programming_language)
14. Groce, A., Holzmann, G., Joshi, R., Xu, R.-G.: Putting flight software through the paces with testing, model checking, and constraint solving. In: Proceedings of the 5th International Workshop on Constraints in Formal Verification (CFV), 2008
15. Havelund, K.: Closing the gap between specification and programming: VDM^{++} and scala. In: Korovina, M., Voronkov, A. (eds.), HOWARD-60: Higher-Order Workshop on Automated Runtime Verification and Debugging, volume 1 of EasyChair Proceedings, December 2011. Manchester, UK (2011)
16. Havelund, K., Reger, G.: Runtime verification logics - a language design perspective. In: Aceto, L., Bacci, G., Bacci, G., Ingólfsdóttir, A., Legay, A., Mardare, R. (eds.) Models, Algorithms, Logics and Tools. LNCS, vol. 10460, pp. 310–338. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63121-9_16
17. Jones, C.B.: Systematic Software Development Using VDM, second edition. Prentice Hall, Hemel Hempstead, UK (1990)
18. Larsen, P., et al.: Vienna development method specification language part 1: base language. Information Technology Programming Languages, Their Environments and System Software Interfaces, December 1996
19. McKeeman, W.: Differential testing for software. Digit. Tech. J. Digit. Equip. Corp. **10**(1), 100–107 (1998)
20. PVS. http://pvs.csl.sri.com

Context Engineering for AI-Assisted Programming for Domain-Specific Languages

Moez Ben Hajhmida[1(✉)] and Edward A. Lee[2]

[1] University of Tunis El Manar, Tunis, Tunisia
moez.benhajhmida@enit.utm.tn
[2] University of California, Berkeley, Berkeley, CA, USA
eal@berkeley.edu

Abstract. Large Language Models (LLMs) have become valuable tools for developers with their ability to accelerate software development. However, LLMs are usually trained on standard programming languages like C/C++, Java, Python, and JavaScript. There has been little research in the usage of LLMs as development tools for less commonly used languages especially when it comes to domain-specific languages like Lingua Franca. This project shows how to curate context to guide the LLM. We demonstrate the technique by showing how to generate code that uses the Lingua Franca coordination language with Python.

1 Introduction

Large Language Models (LLMs) have revolutionized software development by providing developers with powerful tools to enhance productivity and streamline workflows. However, their training and application have primarily focused on mainstream programming languages such as C/C++, Java, Python, and JavaScript [18]. This focus has resulted in a significant gap in research and development regarding LLMs' effectiveness in generating code for less commonly used languages, particularly domain-specific languages (DSLs) [3,10]. DSLs, designed for specific problem domains, offer powerful abstractions and specialized features but often lack the robust tooling and support available for general-purpose languages, especially in the context of LLMs. The challenge becomes even more pronounced with coordination languages, which integrate a DSL for architecture definition with a conventional language for program logic, presenting serious difficulties for LLMs in accurately generating valid code.

Recognizing the limitations of Code LLMs in generating Verilog code, despite its widespread use as a DSL, Thakur et al. [15] addressed the challenge of fine-tuning LLMs to enhance their performance. The authors constructed a comprehensive training corpus of Verilog code by aggregating open-source Verilog repositories and extensively searching textbooks on Verilog HDL. Using this dataset, they fine-tuned multiple LLMs of varying sizes to create Verilog-specialized models. Additionally, they developed a collection of Verilog coding challenges and

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 30–47, 2026.
https://doi.org/10.1007/978-3-032-01377-4_2

test benches to evaluate the functional correctness of the generated code. Their results indicate that, across different problem scenarios, fine-tuning significantly enhances the LLMs' ability to generate syntactically correct Verilog code (25.9% overall).

While the LLMs could be fine tuned to improve their performance with such languages, a simpler alternative is to provide carefully engineered context and crafted prompts as preludes to user-written prompts. Motivated by the goal of freeing users from specifying low-level implementation details, Busch et al. [2] propose an extension of Language-Driven Engineering (LDE) that enables natural language specifications for process requirements. Users simply provide descriptions in natural language, which serve as input for LLM-based code generation. This approach seamlessly integrates graphical DSL with natural language descriptions. Technically, the user first constructs a graphical model, which is then used to automatically generate both extensible code and a corresponding prompt frame. Next, the user describes additional program logic using natural language, which is embedded into the prompt frame to guide the LLM in generating the necessary code. This newly generated code is integrated into the previously generated extensible code from the graphical model, forming the final application code. An automatic validation process is then performed on the running application instance, producing an automaton that represents accessible system states. This automaton can be utilized for model checking, providing valuable feedback to the user. This approach delivers substantial productivity gains in development, though some manual steps are still required.

This project explores the effectiveness of LLMs in generating code for less commonly used languages by examining how they can be leveraged as development tools. Specifically, it addresses the challenge of providing relevant context to enhance LLM-driven code generation. We introduce a novel approach for curating contextual information to enable LLMs to generate code in a domain-specific language (DSL). To demonstrate its effectiveness, we apply this method to the Lingua Franca (LF), a coordination language for concurrent and distributed systems, and its integration with Python. This situation is particularly challenging for the LLMs because the most straightforward context data, such as user documentation, programming examples, and regression tests, mix the LF syntax with a conventional programming language. With insufficient engineering of the context, the LLMs tend to mix the languages in ways that result in nonsensical programs. We present a number of strategies for improving the results of the LLMs. This work contributes to expanding LLM applicability across diverse programming languages, empowering developers working with specialized tools and formalisms.

This paper is organized as follows: Sect. 2 provides an overview of the fundamentals of Lingua Franca language. Section 3 illustrates the challenges of generating LF code with existing LLMs through a motivational example. In Sect. 4, we discuss key techniques for enhancing LLM-based code generation. Section 5 introduces our proposed approach, leveraging the RAG technique to improve LF code generation. In Sect. 6, we summarize our experimental evaluations using

three practical examples. Finally, Sect. 7 and 8 contains a discussion and the conclusions of the paper.

2 Lingua Franca

Lingua Franca (LF) [9] is an open-source domain-specific language for building high-performance, secure, and reliable distributed real-time systems. LF is a polyglot coordination language that facilitates the development of distributed applications by employing a reactor-based architecture. *Reactors* are reactive components programmed in popular programming languages like C/C++, Python, Rust, and TypeScript. An LF application is made of *reactors* connected together with ports and connections. Reactors are deterministic actors whose behavior is specified through *reactions*. Reactions are triggered by discrete events fired at specific logical time instants. LF is supported by a runtime system that facilitates communication between connected reactors, ensuring predictable and consistent execution, even in distributed environments.

Analogous to object-oriented programming, a developer declares the reactors involved in the application and specifies their interactions using the LF-specific language. The logic for each reactor is then defined through a set of reactions. Each reaction's implementation is written in a target language, such as C/C++, Python, Rust, or TypeScript. Proficient Python programmers unfamiliar with LF may struggle to learn the LF syntax. Consequently, we conducted experiments to evaluate the feasibility of using Large Language Models (LLMs) for automating the generation of LF programs.

3 Motivational Example

In this section, we present an example to motivate the techniques to be presented. Consider the task of generating a Python program that estimates π using a Monte Carlo Method[1]. With actual LLMs this task is straightforward; all we need is to compose a comprehensive prompt. A prompt that might be effective is:

> **Prompt (1):** *Provide a Python program to estimate PI through Monte Carlo Method with 10 million samples. Provide code only, without any comment or code fences.*

Passing the proposed prompt to OpenAI's GPT-4o model generates the desired Python code. We provided the same prompt to GPT-4o for 100 independent runs, resulting in 100 Python code outputs. All the generated programs executed without any errors. We also conducted a second experiment instructing

[1] Pi estimation using the Monte Carlo method involves randomly sampling points within a square that circumscribes a circle. The ratio of points falling inside the circle ($x^2 + y^2 \leq 1$) to the total points, multiplied by 4, approximates π. So $\pi \approx 4 \times \frac{\text{inside}}{\text{total}}$.

```
1  target Python:Python
2
3  main reactor MonteCarlo(num_samples=10000000):
4      state inside_circle = 0
5
6      reaction(startup):
7          import random
8          for _ in range(self.num_samples):
9              x, y = random.uniform(0, 1), random.uniform(0, 1)
10             if x*x + y*y <= 1.0:
11                 self.inside_circle += 1
12         estimated_pi = 4 * self.inside_circle / self.num_samples
13         print(f"Estimated value of PI: {estimated_pi}")
```

Fig. 1. Example of a LF code generated by GPT-4o.

to generate a parallel version of the Monte Carlo estimate in Python. In 100 independent trials, the model consistently produced executable Python code. As expected, an LLM like GPT-4o finds this task to be relatively easy and straightforward.

Now, we challenge GPT-4o to generate a valid Lingua Franca (LF) code for the same task. An LF program is a mix of two languages: the LF coordination language and the target language (Python, in this example). The LF language is used to specify the interactions between reactors. The target language defines the logic of each reactor encapsulated in a couple of functions called *reactions*. Here's a prompt detailing the program constraints:

Prompt (2): *Provide a Lingua Franca program for 'target Python'. Create a main reactor named 'MonteCarlo' that:*
- *has no outputs.*
- *has a parameter named 'num_samples', with a default value of 10 million.*
- *estimates PI through Monte Carlo Method with 'num_samples'.*
- *prints the estimated value of PI to the console.*

Provide code only, without any comment or code fences.

For 100 independent runs with the same prompt, GPT-4o failed to generate a program that respects the LF syntax. The logic of the generated code snippets was correct, but the LLM is confused when dealing with two languages in the same file. Embedding Python code within the reaction body affects the LLM, causing it to mix LF syntax with Python syntax. As illustrated in Fig. 1, the code generated by GPT-4o from the above prompt is full of syntax errors. For example, the code in line 1 should be *'target Python'* not *'target Python:Python'*. In addition, LF syntax employs brackets ({ ... }) to delimit a *reactor* body, while on line 3 the generated code is using a Pythonic syntax with semicolons (:). On line 6, we observe the same misuse of semicolons when defining a *reaction*. But in the LF syntax, the body of a *reaction* is delimited by {= ... =}. Another issue is the incorrect placement of the *random* library import on line 7. It should be

```
target Python

preamble {=
  import random
=}

main reactor MonteCarlo(num_samples=10000000) {
  reaction(startup) {=
    inside_circle = 0
    for _ in range(num_samples):
        x = random.random()
        y = random.random()
        if x**2 + y**2 <= 1:
            inside_circle += 1
    pi_estimate = 4 * inside_circle / num_samples
    print(f"Estimated value of PI: {pi_estimate}")
  =}
}
```

Fig. 2. A correct implementation of the PI estimate through the Monte Carlo Method in LF language.

located within a *preamble* section. The right way to implement the PI estimate through the Monte Carlo Method in LF is shown in Fig. 2.

This example highlights the limitations of GPT-4o: despite its proficiency in Python, it falters when generating functionally equivalent Lingua Franca code. GPT-4o struggles to generate a correct syntax when dealing with a DSL like Lingua Franca.

4 Code Generation Background

LLMs for code generation (Code LLMs) strive to generate source code from natural language descriptions. Typically, these natural language descriptions encompass programming problem statements (or docstrings) and may optionally include some programming context (e.g., function signatures, assertions, etc.). Formally, these natural language (NL) descriptions can be noted as *NL*. Given *NL*, the use of an LLM with model parameters θ to generate a code solution *code* can be denoted as $P_\theta(code|NL)$. The advent of in-context learning abilities in LLM has led to the appending of example code snippets to the natural language description *NL* as demonstrations to enhance code generation performance or constrain the generation format [6].

Code LLMs are primarily trained on widely used high-level programming languages (e.g., C, Java, Python), resulting in limited representation of low-resource and domain-specific languages. This imbalance hinders the applicability of LLMs in specialized fields and systems programming. To enhance code generation for low-resource or domain-specific languages, several techniques can be employed. One approach is fine-tuning the model, where the LLM is further trained on domain-specific or task-specific data to better align it with the desired output. Another effective technique is prompt engineering, which involves crafting precise and structured prompts to guide the model's responses and improve generation quality. Additionally, Retrieval-Augmented Generation (RAG) can be leveraged,

where the model integrates external knowledge by retrieving relevant information from a database or corpus during the generation process. These methods, individually or in combination, can significantly improve the accuracy and relevance of code generation in diverse scenarios.

4.1 Fine-Tuning

Fine-tuning involves training a pre-trained LLM on a new task-specific dataset. This process adjusts the model's parameters to better align with the nuances and patterns of the target task. Instruction tuning is a fine-tuning technique for LLMs that utilizes datasets formatted as diverse instructions, aiming to improve the model's ability to follow a broad spectrum of task directives. Recent research [5,11] indicates that this approach significantly enhances model performance and improves generalization to novel, previously unseen tasks. For code generation, fine-tuning involves training an LLM on language-specific demonstration datasets. These datasets pair natural language descriptions (inputs) with corresponding code examples (demonstrations).

Fine-tuning large language models (LLMs), particularly those with billions of parameters, is highly computationally intensive. Instead, focusing on crafting effective prompts can yield significant improvements in performance while minimizing computational resources [8,13,14].

4.2 Prompt Engineering

Well-crafted prompts provide a cost-effective and agile alternative to fine-tuning LLMs. By carefully designing prompts, users can steer the model's output towards specific formats, control content, and significantly improve response quality. This approach allows for rapid experimentation and adaptation without the computational overhead of fine-tuning [1].

A common prompting technique utilizes predefined templates, providing a structured format with specific placeholders to ensure consistent and organized information delivery to the LLM. However, this inherent rigidity can restrict their effectiveness in adapting to the varied reasoning and problem-solving demands of tasks like code generation, particularly when they fall outside the template's precisely defined structure.

The main prompting techniques utilized for code generation are Chain of Thought (CoT) and Structured Chain of Thought (SCoT). The CoT [16] prompting strategy involves breaking down the problem into smaller, logical steps that the model can follow sequentially, enabling step-by-step reasoning. In contrast, the SCoT [7] prompting strategy organizes the reasoning process into predefined templates, ensuring the model generates outputs that are not only logical but also consistent with specific formatting or task requirements.

Building on the observation that the CoT technique relies solely on natural language to sequentially describe problem-solving steps, the Structured CoT (SCoT) technique enhances this by incorporating sequence, branch, and loop

structures. The authors characterize SCOT as a "soft template" due to its integration of both natural language-based COT and more structured template elements within a single prompt. According to Li et al. [7], SCoT outperforms CoT by up to 13.79% for the code generation task.

Prompting requires a well-developed context, which can either be handcrafted or automatically generated. Since LLMs rely on learned patterns and generate responses based on common patterns, they often struggle with less-represented ones, such as domain-specific languages (DSLs). An effective solution to this challenge is enriching the context with real code samples.

4.3 RAG

Retrieval-Augmented Generation (RAG) [17] has recently emerged as a paradigm to address prompting challenges and LLM hallucination. In particular, RAG introduces an information retrieval process, which enhances the generation process by retrieving relevant objects from available data stores, leading to higher accuracy and better robustness.

A typical RAG process begins by indexing and loading a typically large and domain-specific external knowledge base. For each user query, the retriever identifies and extracts relevant documents or knowledge snippets from the external knowledge base. The retrieved information is then incorporated into the prompt provided to the LLM. The LLM uses this augmented context, along with the original query, to generate a more informed, accurate, and contextually relevant response. In essence, RAG bridges the gap between the LLM's parametric knowledge (what it learned during pre-training) and external knowledge (often more up-to-date or domain-specific information) allowing it to generate more accurate and grounded answers without requiring frequent fine-tuning of the underlying model resulting in more informative and accurate responses.

4.4 Key Insights

In summary, prompting techniques enhance problem-solving and reasoning capabilities, while RAG improves the accuracy and relevance of generated responses. In this work, our focus is on improving the syntax of generated code. This is driven by the observation that Code LLMs are predominantly trained on widely used high-level programming languages, leading to limited representation of low-resource and domain-specific languages. This imbalance restricts the applicability of LLMs in specialized domains and systems programming.

5 Proposed Approach

In this section, we introduce our approach to tackling a challenging issue encountered while generating LF code using code LLMs. This challenge stems from the prompt's contextual constraints within LLM models and is further intensified by their limitations in accurately capturing LF syntax.

At first, designing an appropriate context to guide the LLM in generating code with correct LF syntax was overly difficult. For a specific example, we iterated through the prompt-generation process until we found a context that led the LLM to produce the correct code. However, when working with a different example, we had to repeat the iterative process multiple times. Each example requires a unique context. What we truly need is a tool that can automatically generate the appropriate context. This led us to the decision to leverage the RAG technique.

5.1 RAG-Based Code Generation

To implement RAG, we first need to define the knowledge base. In the context of code generation, the knowledge base is a codebase. The codebase encompasses both the raw source code files and their accompanying documentation. For the LF project, we curated code samples from unit tests[2], playground examples[3], and code snippets from the documentation website[4]. This process resulted in a codebase containing 1,391 LF files, with 44% for C target and 25% for Python target. This dataset is too small for fine-tuning an LLM and even for code retrieval, especially considering that Parvez et al. [12] utilized a codebase of 1 million functions.

The initialization step involves applying an embedding model to the codebase. Essentially, this step translates each file in the codebase from human-readable text into a machine-understandable format. The embedding model, a type of artificial neural network, processes this textual content and transforms it into a dense numerical vector (embeddings). This vector captures the semantic meaning and context of the text in a high-dimensional space that allows the RAG system to quickly find semantically similar information when a user poses a query. The resulting embeddings are subsequently indexed and persisted in a datastore, labeled the LF codebase in Fig. 3. Upon receiving a code generation prompt, the embedding model processes the prompt by transforming its textual content into embedding (dense numerical vector). The resulting embedding is then leveraged by the retriever model to perform a similarity search within the indexed codebase and retrieves the semantically similar LF files. The retrieved LF files are dynamically concatenated with the original prompt, forming a context that is subsequently provided to the code LLM to generate the desired code output (as illustrated in Fig. 3). The LLM-agnostic nature of this approach offers the flexibility to be implemented with various code LLMs, allowing for easy adaptation and experimentation.

5.2 Method

Building on our RAG-based code generation approach for retrieving relevant code context, we now introduce our proposed system, which integrates this

[2] https://github.com/lf-lang/lingua-franca/tree/master/test.
[3] https://github.com/lf-lang/playground-lingua-franca.
[4] https://www.lf-lang.org/docs/.

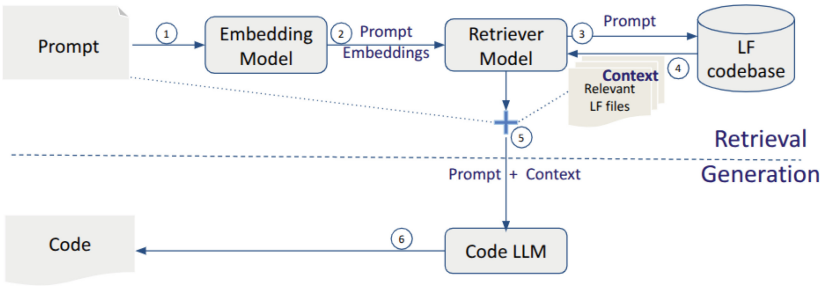


Fig. 3. RAG-based code generation.

retrieval mechanism with a code generation workflow. As described in Fig. 4, there are two separate spaces: a Developer space and a System space. The developer prompts the system with a design description specifying the target language (C/C++, Python, Rust, or TypeScript), and receives the corresponding LF code in response.

Within the system architecture, the developer's prompt undergoes initial processing by the Code RAG module. This module embodies the RAG-based code generation workflow outlined in Sect. 5.1. Based on the user prompt, the Code RAG module dynamically augments the context and produces the corresponding LF code for the specified target language. Subsequently, the generated LF code is processed by the LF code generator for syntax validation. If syntax errors are detected, the system initiates a recursive process, re-invoking the Code RAG module to generate a new code candidate. In the absence of syntax errors, the LF code generator produces a program in pure target language. For languages such as C, C++, and Rust, this code undergoes a compilation phase. Successful compilation is followed by code execution. Upon successful execution, the generated LF source code is delivered to the developer. Conversely, compilation or execution failures trigger a recursive process, re-invoking the Code RAG module to generate an alternative code candidate. To prevent infinite recursion, the process is limited to a predefined maximum number of iterations.

6 Experiments

To evaluate the efficiency of our proposed method, we conducted a series of experiments using diverse examples. Initially, we performed preliminary experiments and manually assessed the quality of the generated code. Once the setup was complete, we identified an evaluation metric suitable for our problem. We then proceeded to experiment with our method by generating code for the Monte Carlo π estimation example introduced in Sect. 3, followed by a parallel version. Finally, we evaluated our method on a more complex scenario involving the design of an accelerometer reactor instantiated within a drop sensor reactor. For all experiments, we restricted our focus to the Python target. To ensure consistency, we fixed the retrieval limit of the RAG workflow to 10 files. This approach

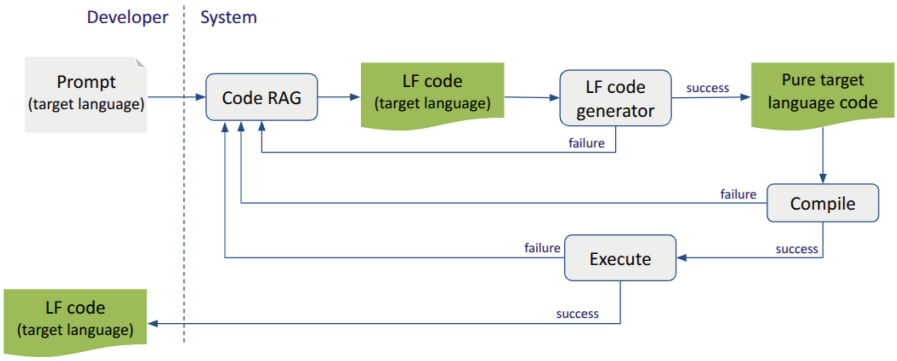


Fig. 4. LF code generation workflow.

augments the prompt context with the content of exactly 10 similar LF files for each user query.

6.1 Preliminary Experiments

When generating code for the Monte Carlo π estimation with the Python target, the reaction body code resembled C more than Python. A review of the retrieved files revealed that 25% of them were C-target files. C-target files were significantly over-represented, constituting 44% of the codebase. This imbalance, with C-target files outnumbering Python-target files by a factor of two, adversely impacted the syntactic correctness of the generated LF and Python code. To mitigate the observed bias, we decided to separate the codebase, creating a dedicated codebase for each target language to ensure more effective retrieval of relevant code examples.

6.2 Evaluation Metric

The most widely used metric for evaluating code generation models is $Pass@k$ [4]. This metric quantifies the probability of success by checking whether at least one of k generated code samples successfully fulfills the defined requirements. Intuitively, $Pass@1$ reflects the Code LLM's ability to produce a correct solution on the first attempt, while $Pass@5$ represents the probability of success within five attempts. A higher $Pass@k$ value indicates better performance. Previous code generation studies have commonly used $k = 1, 3$, and 5 in their experiments [4,7]. To provide code LLMs with a greater opportunity for success, we extended the evaluation to include $k = 10$. The metric is,

$$\text{Pass@k} = 1 - \frac{\binom{n-c}{k}}{\binom{n}{k}}$$

where n is the number of generated programs, and c is the number of correct solutions among the top k results for the given problem. In our experimental

setup, we set n to 100. This metric ranges from zero to one, where zero indicates that all trials failed and one indicates that all succeeded. The metric can be given instead as a percentage ranging from 0% to 100%.

6.3 Monte Carlo π Estimation

Initially, we experimented with two OpenAI LLMs: GPT-3.5-turbo and GPT-4o, without providing any additional context. We then repeated the experiments using handcrafted context. Most of the code generated by GPT-3.5-turbo was in C++ or C#, completely disregarding LF syntax. While GPT-4o was able to produce plausible LF code, it still failed to fully capture the LF syntax. Therefore, we chose to continue our experiments exclusively with GPT-4o.

As mentioned above, when prompting GPT-4o to generate pure Python code for the Monte Carlo π estimation, it succeeded every time, yielding a Pass@k score of 100% for all k. Table 1 shows the Pass@k scores for various mechanisms used to get GPT-4o to generate Lingua Franca code for the same problem. Without context and with context alone, it failed for all k. With RAG it did much better, succeeding on all trials with $k > 3$. With RAG plus a prompt template, it succeeded on every trial for all k.

A descriptive prompt (describes reactors and reactions logic) proved ineffective for generating LF code, leading us to adopt an imperative approach. Descriptive prompts focus on outlining the reactor's structure and the logic of the reactions involved. In contrast, imperative prompts directly instruct the model on specific actions to take, such as importing necessary libraries. To guide the LLM in structuring the code, we employed the Chain-of-Thought technique. We initially used Prompt (2) as introduced in Sect. 3. With this method, the probability of generating correct LF code using the RAG method with GPT-4o is 93% (referenced as RAG(GPT-4o) in Table 1). Upon analyzing the 7% of unsuccessful cases, we found that library imports were not embedded in a preamble. Additionally, some generated code lacked reactions, even though Python code must be enclosed within a reaction. Consequently, we refined our prompt template to explicitly instruct the LLM to "embed imports in a preamble" and "use reactions". The prompt template leads us to the following prompt:

Prompt (3): *Provide a Lingua Franca code for 'target Python'. imports the random module. Create a main reactor named 'MonteCarlo' that:*
- *has no outputs.*
- *has a reaction that triggers at startup.*
- *the reaction estimates PI through Monte Carlo Method with 10 million samples.*
- *prints the estimated value of PI to the console.*
- *embed imports in a preamble.*

Provide code only, without any comment or code fences.

Using this method, referenced as RAG(GPT-4o) plus prompt template in Table 1, we achieve a correct LF code on the first attempt, as indicated by a Pass@1 of 100%.

Table 1. Pass@k scores for various mechanisms used to get GPT-4o to generate Lingua Franca code to estimate π using a Monte Carlo method.

Generation method	Pass@1	Pass@3	Pass@5	Pass@10
GPT-4o	0	0	0	0
GPT-4o + context	0	0	0	0
RAG(GPT-4o)	93	99.58	100	100
RAG(GPT-4o) + prompt template	100	100	100	100

6.4 Parallel Monte Carlo π Estimation

Generating a program for the *parallel* version of the Monte Carlo π estimation is a more complex task. In pure Python, the program consists of a main function that distributes the workload across multiple threads, each executing the sequential Monte Carlo π estimation. The main function then aggregates the results returned by the threads. In our experiments, GPT-4o succeeded in generating the correct code every time. According to the semantics of the LF language, the program is composed of three interconnected reactors: MonteCarlo, which performs the estimation; Aggregator, which combines the results; and Main, which orchestrates the overall computation. The following prompt details the program's constraints:

Prompt (4): *Provide a Lingua Franca code for 'target Python'.*
Define a reactor named 'MonteCarlo' that:
- *has one output port named 'out'.*
- *has a reaction that triggers at startup.*
- *the reaction estimates PI through Monte Carlo Method with 'num_samples' samples.*
- *sets the output to the estimated value of PI.*

Define a reactor named 'Aggregator' that:
- *has one input port named 'inp' of width 4.*
- *has a reaction triggered by 'inp'.*
- *the reaction calculates the mean of the inp values and prints the result.*

Define a main reactor that:
- *instantiates 4 reactors named 'MonteCarlo' in parallel and distributes the workload of 10 million samples over them.*
- *The total width on the two sides of a connection statement -> must match. If a multiport is referenced on one side, there must be either a matching multiport on the other side or a comma separated list of ordinary ports.*
- *instantiates 1 reactor named 'Aggregator' that aggregates the results of 'MonteCarlo' reactors.*

Embed imports in a preamble. Provide code only, without any comment or code fences.

Table 2 shows the Pass@k scores for different methods used to guide GPT-4o to generate Lingua Franca code to estimate π using a Parallel Monte Carlo method. Across all values of k, GPT-4o struggled without context and with context alone. Using RAG, it succeeded in 7 out of 100 attempts, resulting in a Pass@1 of 7%. However, by integrating RAG with a prompt template, performance improved significantly, achieving a Pass@1 of 37% and a Pass@10 of 99.26%.

Table 2. Pass@k scores for various mechanisms used to get GPT-4o to generate Lingua Franca code to estimate π using a Parallel Monte Carlo method.

Generation method	Pass@1	Pass@3	Pass@5	Pass@10
GPT-4o	0	0	0	0
GPT-4o + context	0	0	0	0
RAG(GPT-4o)	7.00	13.58	30.97	53.33
RAG(GPT-4o) + prompt template	37.00	60.55	90.66	99.26

6.5 Drop Sensor

In this scenario, we first design an Accelerometer reactor and then incorporate it into a DropSensor reactor. The Accelerometer reactor simulates an accelerometer, generating random three-axis values each time it is triggered. Since the Accelerometer reactor will be imported into another reactor, a main reactor is not required. The DropSensor reactor imports and instantiates the LF Accelerometer reactor. A timer triggers the Accelerometer every 250 ms to generate three-axis outputs. The absolute values of these outputs are then compared to a threshold. If all values are below the threshold, they are printed to the console. The following prompt details the accelerometer's constraints:

Prompt (5): Provide a Lingua Franca code for 'target Python'.
Create a reactor named 'Accelerometer' that:
- has a 'trigger' input.
- outputs three random float values 'x, y, and z' between –1 and 1 when triggered through a 'trigger' input.
- declares constants in the preamble.

There is no main reactor. Provide code only, without any comment or code fences.

When using the above prompt to generate Lingua Franca code, GPT-4o failed in all attempts both without context and with context. With RAG, it succeeded in every trial for all k, regardless of whether a prompt template was used. Figure 5 illustrates the diagram representing a successfully generated Accelerometer reactor.

The DropSensor constraints are translated to the following prompt:

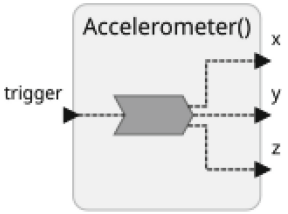


Fig. 5. Diagram representing a successfully generated Accelerometer reactor.

Provide a Lingua Franca code for 'target Python'.
Import the LF 'Accelerometer' reactor. Create a main reactor named 'DropSensor' that:
- has a parameter named threshold, with a default value of '0.3'.
- instantiates an accelerometer.
- declares a timer with a period of 250 ms.
- has a reaction that triggers the accelerometer with period of 250 ms by setting the accelerometer trigger to True.
- has another reaction which fires whenever any of the accelerometer readings ('x', 'y', or 'z') change. Inside this reaction: It checks whether the absolute values of the accelerometer readings ('x', 'y', 'z') are all less than the defined 'threshold'. If the condition is true, it prints the values of the accelerometer outputs to the console.

Provide code only, without any comment or code fences.

Table 3 presents the Pass@k scores of various methods employed to instruct GPT-4o to generate Lingua Franca code for a DropSensor reactor. With the exception of RAG combined with a prompt template, all other methods failed to generate a single successful code. In contrast, RAG with prompt template achieved a Pass@1 of 35% and a Pass@10 of 98.97%, demonstrating superior performance. Figure 6 depicts the diagram of a successfully generated DropSensor reactor.

Table 3. Pass@k scores for various mechanisms used to get GPT-4o to generate Lingua Franca code to implement a DropSensor reactor.

Generation method	Pass@1	Pass@3	Pass@5	Pass@10
GPT-4o	0	0	0	0
GPT-4o + context	0	0	0	0
RAG(GPT-4o)	0	0	0	0
RAG(GPT-4o) + prompt template	35.00	57.98	89.03	98.97

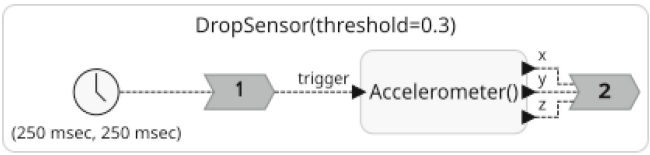


Fig. 6. Diagram representing a successfully generated DropSensor reactor.

7 Discussion

Through the iterative process of prompting the LLM, reviewing the generated code, and refining the prompt, we gained valuable insights.

In evaluating LLM capabilities, significant performance differences were observed. GPT-4o was able to generate code that adhered to the structure of the LF language, whereas GPT-3.5-turbo failed completely. This divergence in performance is attributable to GPT-4o's larger model size and access to more recent training data. In contrast, GPT-3.5-turbo's knowledge is limited to data up until September 2021, a time when the Lingua Franca project was available on GitHub but still in its early stages, thus explaining its inability to generate correct code.

Our initial experiments with RAG revealed a bias toward C syntax in the generated code, even when the target language was Python. This bias stemmed from the overrepresentation of C-target files in the codebase, which negatively affected the syntactic correctness of both LF and Python code. We conclude that, similar to standard machine learning datasets, maintaining a balanced codebase is essential. Focusing solely on Python target files, we enhanced some LF files by adding more comments. Well-commented code improves similarity between retrieved outputs and user's prompt. This highlights that RAG's effectiveness is highly dependent on the quality of the codebase.

As outlined in the Structured Chain of Thought (SCoT) [7] prompting technique, a structured prompt enhances the accuracy of the generated code. For instance, specifying *"parameter num_samples = value"* in the prompt rather than simply stating *"has parameter"* led to improved code quality.

Given the unique characteristics of LF syntax, we had to craft precise instructions to guide the LLM in adhering to these specifics. One important requirement is that a main reactor cannot have outputs, necessitating an explicit directive such as *"has no outputs"* to ensure compliance. Similarly, to enforce the inclusion of external libraries within a preamble, we had to specify *"embed imports in a preamble"*. Additionally, to guide the LLM in implementing a reactor's logic through distinct reactions, we explicitly enumerated each reaction along with its intended functionality. Accordingly, we developed a structured prompt template to improve RAG-generated codes.

When LLMs generate LF code, they often face challenges in producing the precisely correct syntax on the initial attempt, resulting in a lower Pass@1 score. However, by generating multiple samples, the probability of achieving the cor-

rect syntax increases. Consequently, the Pass@10 metric provides a more realistic assessment of the likelihood that a usable LF code will be generated within a practical number of trials. Pass@10 becomes particularly insightful when comparing models with similar Pass@1 performance, as it can highlight a more substantial divergence in their capacity to generate a functional solution within a small number of attempts.

8 Conclusion

Large Language Models (LLMs) often face challenges when generating code for Domain-Specific Languages (DSLs). In this work, we introduced a method to curate context that effectively guides LLMs in generating code for the Lingua Franca (LF) domain-specific language. Our implementation combines Retrieval-Augmented Generation (RAG) with Structured Chain-of-Thought (SCoT) prompting to facilitate the generation of LF code alongside Python.

Our approach enhances the prompt context by incorporating relevant code samples from the existing Lingua Franca codebase. Given a developer's prompt, a retriever model conducts a similarity search within the codebase, retrieving LF files that are dynamically concatenated with the original prompt. This enriched context is then fed into GPT-4o, enabling more accurate and syntactically correct LF code generation.

We conducted a series of experiments using three different examples. For each example, we evaluated the Pass@k scores across four different approaches for guiding GPT-4o in generating Lingua Franca code. In every case, RAG combined with a prompt template achieved the highest Pass@k scores, demonstrating its superior effectiveness. These results demonstrate that combining RAG with SCoT effectively guides the LLM in capturing LF syntax and generating valid code. To further validate our findings, we plan to apply our proposed method to other state-of-the-art Code LLMs.

We plan to expand the codebase by generating new LF programs using the current framework. With a larger dataset, we aim to fine-tune a Code LLM and evaluate its effectiveness compared to the approach proposed in this work.

Code Availability. For the purpose of reproducibility, the code and data supporting the results and experiments in this paper are available publicly on GitHub at: https://github.com/lf-lang/lf-gpt.

References

1. Brown, T., et al.: Language models are few-shot learners. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems, vol. 33, pp. 1877–1901. Curran Associates, Inc. (2020)
2. Busch, D., Nolte, G., Bainczyk, A., Steffen, B.: ChatGPT in the loop: a natural language extension for domain-specific modeling languages. In: Steffen, B. (eds.) Bridging the Gap Between AI and Reality. AISoLA 2023. LNCS, vol. 14380, pp. 375–390. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-46002-9_24

3. Cassano, F., et al.: MultiPL-E: a scalable and polyglot approach to benchmarking neural code generation. IEEE Trans. Softw. Eng. **49**(7), 3675–3691 (2023). https://doi.org/10.1109/TSE.2023.3267446
4. Chen, M., et al.: Evaluating Large Language Models Trained on Code. ArXiv **abs/2107.03374** (2021)
5. Chung, H.W., et al.: Scaling instruction-finetuned language models. ArXiv **abs/2210.11416** (2022)
6. Jiang, J., Wang, F., Shen, J., Kim, S., Kim, S.: A Survey on Large Language Models for Code Generation. arXiv preprint arXiv:2406.00515 (2024)
7. Li, J., Li, G., Li, Y., Jin, Z.: Structured chain-of-thought prompting for code generation. ACM Trans. Softw. Eng. Methodol. **34**(2) (2025). https://doi.org/10.1145/3690635
8. Li, R., et al.: StarCoder: may the source be with you! ArXiv **abs/2305.06161** (2023)
9. Lohstroh, M., Menard, C., Bateni, S., Lee, E.A.: Toward a lingua franca for deterministic concurrent systems. ACM Trans. Embed. Comput. Syst. **20**(4) (2021). https://doi.org/10.1145/3448128
10. Murali, V., et al.: AI-assisted code authoring at scale: fine-tuning, deploying, and mixed methods evaluation. Proc. ACM Softw. Eng. **1**(FSE) (2024). https://doi.org/10.1145/3643774
11. Ouyang, L., et al.: Training language models to follow instructions with human feedback. ArXiv **abs/2203.02155** (2022)
12. Parvez, M.R., Ahmad, W., Chakraborty, S., Ray, B., Chang, K.W.: Retrieval augmented code generation and summarization. In: Moens, M.F., Huang, X., Specia, L., Yih, S.W.t. (eds.) Findings of the Association for Computational Linguistics: EMNLP 2021, pp. 2719–2734. Association for Computational Linguistics, Punta Cana, Dominican Republic (November 2021)
13. Rozière, B., et al.: Code Llama: Open Foundation Models for Code. ArXiv **abs/2308.12950** (2023)
14. Shao, Z., Dai, D., Guo, D., Liu), B.L.B., Wang, Z., Xin, H.: DeepSeek-V2: A Strong, Economical, and Efficient Mixture-of-Experts Language Model. ArXiv **abs/2405.04434** (2024)
15. Thakur, S., et al.: Benchmarking large language models for automated verilog RTL Code generation. In: 2023 Design, Automation & Test in Europe Conference & Exhibition (DATE), pp. 1–6 (2022)
16. Wei, J., et al.: Chain of Thought Prompting Elicits Reasoning in Large Language Models. ArXiv **abs/2201.11903** (2022)
17. Zhao, P., et al.: Retrieval-Augmented Generation for AI-Generated Content: A Survey. arXiv preprint arXiv:2402.19473 (2024)
18. Zheng, Q., et al.: CodeGeeX: a pre-trained model for code generation with multilingual benchmarking on HumanEval-X. In: Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, pp. 5673–5684. KDD '23, Association for Computing Machinery, New York, NY, USA (2023). https://doi.org/10.1145/3580305.3599790

The Impact of Generative Artificial Intelligence Tools in Project-Based Learning

Tom van Dijk and Vadim Zaytsev(✉)

Formal Methods and Tools, University of Twente, Enschede, The Netherlands
t.vandijk@utwente.nl, vadim@grammarware.net

Abstract. We investigate the potential risks and benefits for students utilising Generative Artificial Intelligence (GAI), specifically OpenAI's ChatGPT and GitHub's Copilot, to generate solutions instead of independently creating them in the traditional educational setting. The rapid advancements in GAI have transformed numerous domains, including software development and software engineering education. While these tools offer unprecedented convenience and efficiency, there are growing concerns regarding their potential implications for academic integrity and genuine student learning. We report on a pilot study in which 40 students, who completed a first-semester course on object-oriented programming, re-engage in a comparable programming project in limited time using GAI tools. In this pilot study, we aim to assess the extent to which students can rely on GAI tools to generate solutions for large programming assignments, to investigate the impact of AI-driven code generation on students' understanding of fundamental programming concepts and problem-solving abilities, and to explore the perspectives of educators and students on the implications and long-term consequences of integrating AI-assisted coding tools in the learning process.

1 Motivation and Background

Generative Artificial Intelligence (GAI) is a field of study that uses trained models to generate data fragments shaped as coherent text, realistic images, believable videos and executable code. These models can be generative adversarial networks [17], variational autoencoders [22], diffusion models [9], autoregressive models [20], energy-based models [41], flow-based models [35], as well as—the most famous these days—large language models. Recent rapid advancements in this area have made a lot of things possible: instead of requiring large volumes of specific data, modern GAI tools can combine large volumes of generic data they have already witnessed, with a concise definition of context, to generate all kinds of artefacts traditionally required as deliverables in the software industry as well as software engineering education: software, documentation, solutions for given problems, tests for given code, presentations, critical reflections, etc. This in turn started reshaping various professional fields from creative industries to software development. In the literature this phenomenon is known as *job*

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 48–76, 2026.
https://doi.org/10.1007/978-3-032-01377-4_3

crafting [14], because it leads to employees reshaping their jobs to fit evolving demands. Tools such as ChatGPT [7] and GitHub Copilot [16] are now capable of generating entire blocks of text or code with minimal human intervention, potentially accelerating productivity in a significant way.

However, in education—in particular in programming courses—this technological leap presents not just opportunities, but mostly challenges. While GAI-powered tools can assist students in writing code, testing, debugging, documenting, even learning new concepts, they also raise fundamental questions about academic integrity, skill acquisition, and the role of traditional learning methods. Educators can no longer afford to ignore GAI; instead, we must critically examine its impact and develop strategies to either integrate or regulate its use in ways that enhance, rather than diminish, student learning. In this study, we aim to explore how students interact with GAI in a project-based programming course, assessing whether these tools support or hinder their understanding of fundamental programming concepts.

There is an actively ongoing heated debate among educators and researchers in education concerning the impact of GAI, whether it constitutes a threat or an opportunity [4,5,40,48,49]. On one side, critics argue that GAI can undermine academic integrity, diminish critical thinking skills, create new forms of inequality through differential access to advanced technological tools, etc. On the other side, proponents highlight the capacity of GAI-based systems to personalise learning experiences beyond what is possible by human means, streamline administrative tasks to redirect human teachers' attention to the most impactful places, and promote more inclusive pedagogical strategies by fully accommodating diverse learning styles. One more complementary viewpoint is that the use of GAI tools is not inherently bad, and potential productivity boosts it delivers are attractive, but with one unavoidable precondition: the user must understand the output of such GAI tools. If this prerequisite is not fulfilled, then the quality of the result cannot be assessed and thus cannot be guaranteed. All these conflicting perspectives revolve around concerns of bias, transparency, ethical accountability, making it increasingly complex for educators, policy makers and researchers to take a definitive stance on the overall impact of GAI. Ultimately, the question of whether GAI is a threat or an opportunity remains open-ended, with the answer depending on a long list of global aspects like its adoption in the field, as well as seemingly small details like concrete studied topics, formulated learning goals and attitude of both learners and teachers.

ChatGPT [7] is a large language model developed by OpenAI. It is capable of generating text based on user prompts, and does it in a human-like fashion. In computer science education, it can function as a virtual tutor, assisting students with code explanations, debugging help, algorithm suggestions, as well as full program generation. Unlike traditional search engines, ChatGPT provides conversational, context-aware responses, making it an attractive tool for learning—if not for teachers, then definitely tempting for students. However, its ability to generate plausible and believable yet incorrect answers poses a challenge: without fully understanding the underlying concepts, students may rely on it too

much, potentially weakening their problem-solving skills. This duality—being both a powerful learning aid and a potential shortcut that bypasses learning deeper understanding—makes ChatGPT a critical subject of study in programming education.

Copilot [16], developed by GitHub in collaboration with OpenAI, is a GAI-powered code completion tool designed to assist programmers by suggesting entire functions, generating boilerplate code, and providing inline coding assistance. Integrated directly into coding environments like Visual Studio, VS Code, IntelliJ IDEA, Xcode and the like, Copilot offers real-time code predictions based on the developer's comments and existing code structure. For students, Copilot can accelerate coding tasks, reduce syntax errors, and serve as an example-driven learning tool. However, similar to ChatGPT, it raises concerns regarding over-reliance and passive learning, as students may accept AI-generated solutions without fully understanding the logic behind them. The industry's reaction to GitHub Copilot has been mostly positive, but unlike expert developers who use it as a powerful code completion tool which output they can fully comprehend before committing to it, novice developers like students still learning how to program properly, can just end up using it as a shortcut to a *somehow* working solution, bypassing the actual learning.

1.1 Problem Statement

In this paper, as well as in our programme, we rely on the paradigm of *project-based learning* [15,23]. It is an educational approach where students are supposed to gain knowledge and skills by actively engaging in real-world, complex projects over an extended period. We see this as a programme-level implementation of the university-wide vision of *learning by interacting* [47], but it is also closely related to concepts of *problem-based learning* [33,52], which focuses on solving open-ended problems while defining your own learning outcomes; *challenge-based learning* [46,51], which emphasises real-world challenges which are often inter-disciplinary and by definition larger than what learners can solve, which forces them to choose their own challenges and face them; as well as *student-driven learning* [12,21], which is an approach that provides personalised learning experiences by moving away from traditional teacher-directed models. With some small variations (which might be very important on implementation level), all these approaches share the line of thinking that values principles of critical thinking, problem solving, making own decisions on the learning path, influencing goals, outcomes, deliverables and strategies.

Traditionally, project-based learning ensures deep engagement by requiring students to design, implement, refine and improve their own solutions. However, the integration and/or mere existence of GAI tools challenges this paradigm. The students who employ these tools, gain the ability to generate significant portions of their projects with minimal effort, potentially diminishing the learning-by-doing aspect. While GAI can serve as a tutor or assistant in overcoming technical hurdles, its automation of cognitive processes risks turning project work into a passive exercise of prompt engineering *instead* of genuine problem solving. In

this study we aim to contribute to investigation of how GAI affects traditional project-based learning setups, determining whether it supports learning or unintentionally undermines the development of essential programming competencies.

Since project-based learning thrives on the principle of learning-by-doing, where students engage directly with coding challenges, wrestle with design decisions, debug their own errors, refine their solutions, and learn from engaging in all those activities with their own freshly made artefacts, using GAI to substitute or "optimise" some of those steps naturally raises a fundamental concern. If GAI performs the "doing", does any meaningful *learning* still take place at all? When students shift from actively constructing solutions to passively prompting a GAI tool for answers, they may miss out on the deep engagement required to develop core programming competencies. This shift threatens skill acquisition [18,54], as students risk developing fluency in prompt engineering rather than algorithmic thinking, debugging strategies, or software design principles. Beyond the pedagogical concerns, this raises pressing academic integrity issues [39]. If a GAI tool generates substantial portions of a project, how do we assess a student's true understanding of the material? The challenge for educators is to differentiate between GAI-assisted learning and GAI-driven substitution, ensuring that students still engage in the cognitive processes that build genuine programming expertise. Traditional plagiarism detection methods are often ineffective against GAI-generated content [50]. Addressing these concerns requires new assessment strategies, clear ethical guidelines, and a rethinking of how learning is measured in a GAI-assisted educational landscape.

Lastly, the existence and commoditisation of GAI tools may or may not enforce a fundamental reevaluation of how *learning outcomes* are defined and assessed in programming education. Traditionally, assessments focus on code quality, problem-solving ability, conceptual understanding, often measured through project work, exams, or written reports. However, when GAI can generate functional deliverables, including code, documentation, and even debugging explanations, educators must ask: are our current learning outcomes still meaningful, or do they need to evolve? One viewpoint could be to maintain existing learning goals but adjust assessment methods—for example, shifting toward oral exams, in-class coding tasks, or GAI-aware rubrics that evaluate reasoning rather than output. Another, more radical approach could be to overhaul learning objectives entirely, recognising that in a GAI-assisted world, the emphasis should shift from syntax and implementation toward embracing GAI collaboration, debugging GAI-generated solutions, and verifying correctness. Whether through adaptation or reinvention, assessment must ensure that students develop not just the ability to produce code but also the critical thinking skills necessary to navigate a world where GAI is an inevitable part of the software development process. We leave this aspect as a dilemma for future work.

In this pilot study, our goals are:

- to assess the extent to which students can rely on GAI tools to generate solutions for programming assignments;

- to explore the perspectives of educators and students on the implications and long-term consequences of integrating AI-assisted coding tools in the learning process.

We hope that these preliminary collected results will help in the near future to make the next step to investigate the impact of AI-driven code generation on students' understanding of fundamental programming concepts and problem-solving abilities.

1.2 Context

In the Bachelor programme of *Technical Computer Science* at the University of Twente, object-oriented programming and software design are taught as an introductory course in the second quartile of the first study year, in a period of 10 working weeks.

The course consists of a 4 EC part where students learn about software modelling with UML diagrams about the development process in general, and an 8 EC part where students learn programming using Java, study basics of concurrency and race conditions, as well as basic networking with sockets. The students spend roughly 7 weeks practising to reach these learning goals, and then do a project in the remaining 3 weeks.

This project involves writing a full server-client application using Java Sockets. This application allows users to play a simple board game over the network against other players and against a simple computer player that is also programmed by the students. Typical board games for this project are perfect information games played on a rectangular grid, with few rules and simple game-over conditions, such as Connect-Four, Pentago, Othello and others. Students typically perform this project in pairs, although typically some students do the project by themselves when their partner drops out.

2 Related Work

There are many GAI tools provided by different vendors, but we focus specifically on GitHub Copilot [16] and ChatGPT [7], because they either perform competitively when matched against alternatives, or outperform them significantly. Both of them are also rather well-known and easy to setup, which in turn leads to them being well researched [30].

GAI repeats things it learnt from the training data, which also includes common mistakes [5], security vulnerabilities [36] and copyright violations [10]. This also leads to bias which some researchers see as hallucinations [48], while others equate hallucination to inventing believable terms [49], which is the opposite to repeating common mistakes. In classical terms, a GAI tool which gives an incorrect answer because it has witnessed it often before, can be seen as generating a *false negative*, while a GAI tool that produces an incorrect but seemingly convincing answer because it finds a way to concoct a wrong but believable chain of arguments leading to it, can be seen as *false positive*.

Interestingly, repeating commonly observed patterns does not mean GAI tools like Copilot will suggest idiomatic code, at least not for JavaScript and Python [38]. Presumably this happens because there is enough non-idiomatic code in existence, which is known from modernity research, investigating intricate patterns of coevolution of languages and codebases [1,13]. Another issue potentially limiting Copilot's applicability in practice is its lack of robustness, meaning that a slight change in its prompt might make it generate a completely different completion code [30].

Future education is projected to include techniques like *intelligence augmentation* [40], which will emphasise skills like prompt engineering (not just knowing the syntax or patterns, but also conceptually "asking the right questions"), and *virtual intelligent tutoring*, when an interactive LLM like ChatGPT essentially replaces traditional teachers' tasks like providing additional explanations and formative feedback [40,48].

Many authors point out that (over)reliance on GAI will train learners less in skills like critical thinking [48,49]. Some also argue that this skill needs to be trained additionally next to similar activities like developing emotional intelligence [18]. On the positive side, it is exactly this lack of overly critical attitude towards its output that leads to their admitted usefulness in gathering ideas and issues on any given topic [49]. Unfortunately, specifically in software engineering overly creative out-of-the box solutions tend to contain more bugs than solutions following the beaten track.

It is becoming increasingly more common to see more technically literate students rely on GAI tools as a "clever" shortcut to completing homework effortlessly, and less GAI aware teachers condemning these actions and generalising them to any GAI use. Discussions of using GAI in the classroom commonly gravitate towards the topic of plagiarism [49]. Some authors go as far as firmly and inherently linking the use of ChatGPT and similar GAI tools to academic dishonesty [39]. However, even if there are some automated detection options available for the by now hopelessly outdated ChatGPT 3.5, most of them desperately fail to detect the use GPT 4, let alone o1, o3 or 4.5 [50].

When it comes to institutional policies, the most common one is to "ban it till we understand it" [27], or, to put it more mildly, to have "a comprehensive strategy to manage AI-assisted learning and control AI tool usage within educational institutions" [49]. Most such policies revolve around two principles: (1) banning GAI by default unless explicitly permitted by a deviant teacher; and (2) demanding explicit claims of using GAI or even not using it, possibly to create legal liability. This is very much in line with the early attitude and policies against using Wikipedia in classroom [19].

Yan et al. list "AI-induced performance illusion" among the key challenges for learners [54]. This is, of course, the cornerstone of the entire issue: GAI tools are just tools which need to be learnt to be used effectively, and at their current state their output must be examined with not much less rigour than results of one's own manual coding or the code of one's junior colleague. GAI is good

learning aid and a decent completion instrument, but it has many downsides as a shortcut since it tempts users to bypass skill acquisition.

Specifically in the context of project-based learning, there are some recent BSc and MSc theses that attempted to enhance GAI tools' functionality of providing specific feedback to students while avoiding giving them solutions as such, and at the same time explicitly teaching students how to seek such feedback [25,44]. The consensus at the moment seems to be that it is possible at least to some extent to restrict and/or customise GAI to provide feedback, and asking for it is definitely a teachable skill, but it remains to be seen whether such generated feedback truly contributes positively to the learning experience. In any case, it seems to decidedly improve peer feedback [43]. We see this line of work as complementary to ours.

Just like students have an option to use GAI for solving assignments given to them, teachers can use GAI to assess students' deliverables automatically. It is known that when it comes to essays and small study reports, GAI scores are comparable to those given by human examiners [26,32], as long as complexity is kept low. For project-based learning, there are some emerging frameworks being proposed for GAI-driven automated grading [53], but as of now they lack substantial evidence that would demonstrate the limits of their applicability. However, from a line of research we pursue in parallel to this work, we know that not only automated grading is prohibited by many Examination Boards, it is also highly undesirable when stakeholders are consulted [42]. What stakeholders—from teachers to students and from e-learning experts to educational management—want, is automated *assistance* in their assessment activities, not automated grading per se. Such assistance, of course, can be, among many other methods, based on (G)AI.

Lastly, there are some recent examples of project-based courses specifically designed to integrate GAI models and tools into structured co-creation activities [28,37,45]. Since courses that do not undergo such redesign, tend to report that all students use GAI anyway [6], avoiding this issue seems like a losing battle.

3 Methodology

We have recruited students who passed the first year of the Bachelor programme in Technical Computer Science, that is, who completed the course in question before, including the project of implementing a board game. To level the playground and to make sure nobody of them can reuse their own code, we have asked them to implement a different board game.

We have divided the students up into four groups of 10 students each:

- Control group, not allowed to use any GAI tools at all
- "Copilot" group, instructed to use Copilot and not allowed to use ChatGPT
- "ChatGPT" group, instructed to use ChatGPT and not allowed to use Copilot
- "Both tools" group, instructed to use both Copilot and ChatGPT

To divide the students in groups, we first asked the students to self-assess their programming ability, their skill to use Copilot and their skill to use ChatGPT. We then placed the students into categories and randomly split the students from each category into the four groups.

The programming project requested from these students, was slightly more difficult than the official project of the previous year. They were instructed to implement the game Hex, which requires reasoning about hexagonal geometry (rather than playing on a grid of squares), and to implement pathfinding from one side of the board to the other. These requirements are usually deemed too difficult for students in the first semester, since on the starting day some students do not even know the basics of programming, but would be interesting challenges for students who have already completed their first year. Furthermore, we instructed students to spend in total exactly 36 h on the programming project, which is less than the roughly two weeks in the first semester, even accounting for student working hours being shorter than those of professionals. If they were finished earlier, they could spend more time polishing their implementations to improve their grade. Students from groups with an allowed use of GAI, were instructed to do so as much as would be reasonable to get a high grade.

During the project, the participants maintained a *reflective journal*, ideally requiring about 5–10 min per two hours of their programming work. After completion of the project, they *self-assessed* their work using the official rubric, which is the same rubric of the project in the first-semester course. They then *peer reviewed* the grading of another project. Finally, the projects were discussed in *focus group meetings*.

The reflective journals, comprising 120980 words in total (after removing code snippets, screenshots and diagrams), were read and manually tagged with the following tags:

- Judgemental tags:
  - **joy**: any expression of positive emotion, satisfaction with the result, with GAI interaction, tool behaviour, etc., beyond "works as expected".
  - **pain**: any expression of negative emotion, frustration, dissatisfaction, dislike of results or process steps.
  - **boost**: any place in the text where students reported a significant help or speedup caused by the use of GAI.
  - **waste**: any place where the use of GAI was accompanied by a waste of time and resources, including dysfunctional code, misleading designs, bugfixes that did not result in desired effect, etc.
  - **manual**: admitting manual programming (usually after a failed attempt of GAI use or intentionally replacing it).
  - **prompt**: any place where a prompt was described or quoted, or a moment when GAI was "asked" or otherwise queried.
  - **could have**: any places where GAI was used but the result was perceived as something the user could have done themselves, or places where the user deliberately decided to not use GAI while having that option.
  - **would have**: any places where the user would have wanted to use GAI but did not because it was not permitted by the rules of their group.

  - **ai misuse**: any description of situations where GAI was misused and/or could have been misused.
- Coding activity tags:
  - **project planning**: taking a step back and contemplating next steps in the implementation.
  - **refactoring**: cleaning up and otherwise improving the design of previously written or generated code.
  - **debugging**: finding, localising, understanding or removing bugs from existing code.
  - **testing**: checking existing code for certain properties or scenarios, writing test cases, executing them, augmenting the test suite, etc.
  - **reasoning**: any activities related to understanding existing code, be it for the programmer's sake or for the sake of interpreting GAI's output or formulating a prompt.
  - **documentation**: anything describing manipulations with comments, Javadoc [24], report or logbook.
- Design topic tags:
  - **class diagram**: UML class diagrams, but only if explicitly mentioned (discussing object-oriented design did not lead to tagging).
  - **sequence diagram**: UML sequence diagrams, but only if explicitly mentioned (discussing system behaviour did not lead to tagging).
  - **design pattern**: usually MVC or Listener, but with some mentions of Factory.
- Specific topic tags:
  - **game rules**: anything related to the game as such, most often some details like the form of the hexagonal board (and determining neighbours on such a board) or computing the winning condition.
  - **protocol**: anything related to the mandated design of the communication between the client and the server, the kinds of commands that had to be supported, their order, etc.
  - **concurrency**: anything related to the distributed nature of the system, usually about choosing a threadsafe data structure or suffering from consequences and race conditions when working with a non-threadsafe one.
  - **network**: sending messages over the network, using sockets, occasionally figuring out details of client-server collaboration.
  - **user interaction**: visualising the game board and communicating with the player about their moves.
  - **error handling**: the entire spectrum from catching exceptions to just dealing with abnormal scenarios.
  - **JML**: Java Modelling Language [8], mastering which was a part of the learning objectives of the course, and was meant as a way to invite students to reason about correctness of their code.
  - **TDD**: test-driven development [31], which was encouraged in the original project (along other techniques like pair programming), and we expected to find it back in the reports, only to be gravely disappointed. TDD was

mentioned explicitly three times, and on one additional occasion a participant reported to write a deliberately failing test case before implementing the next feature, which is a TDD technique.

The tagging was done following the principles of narrative research [11] and should be taken at face value. For instance, if a student decided to explain their debugging process in five sentences, then each of them was tagged with the **debugging** tag, even when the issue was not five times larger than another debugging case. Similarly we dealt with emotion tags: for instance, if GAI generated dysfunctional code, simply reporting that it did not work would merit at most a **waste** tag, but writing down that it "forgot" or "misunderstood" the question or that it was not helpful, was read as opinions and marked with the **pain** tag.

Additionally, the first pair of tags (**joy** and **pain**) might seem similar to the second pair (**boost** and **waste**), and while it is true that dealing with dysfunctional code often triggers negative emotions, while getting visible performance gains usually goes the other way, this tendency was not universal. For example, the following sentences were tagged as both **pain** and **boost**:

- *I was not satisfied with this at all but gave at least the idea for the future code.*
- *Though it was quite tedious having to copy paste each class, one by one, it saved me a lot of time.*

4 Results

4.1 Quantitative Findings

We originally started with 40 participants. During the study, we lost 3 participants due to personal GAI-unrelated circumstances: two participants of the control group did not deliver anything and were excluded from the data, one more participant from the "both tools" group submitted a minimal journal which we ended up including in the data just to get a few more taggings (participant 4 on Table 3). Their journal was 393 words long, which is considerably lower than the average (mean 3184, median 2621), but not dramatically smaller than the shortest full journal (764 words).

After self-assessment and peer review, we summarise the grades that the projects would receive according to the rubric in Table 1. Unfortunately the sizes of the groups are such that results are **not** statistically significant. Still, we observe that groups with access to Copilot had better pass-fail outcomes, and that the group with access to both tools had both higher average and higher median grades. Due to the sizes of the groups, we cannot truly draw hard conclusions about the effect on the grades.

Daily/hourly journals which students submitted, provided much more detailed insights into their way of working and the impact of using GAI, which

Table 1. Grades for the projects of each group, and for the Functionality, Software quality and Report subgrades. Grades are between 1 and 10 (excellent), where 5.5 is the minimum passing grade.

Group	Size	Pass:Fail	Average Functionality	Quality	Report
Control group	8	5:3	6.19	5.93	5.69
“Copilot” group	10	9:1	7.38	6.66	6.46
“ChatGPT” group	10	7:3	6.60	6.29	5.95
“Both tools” group	9	8:1	7.47	6.64	7.08

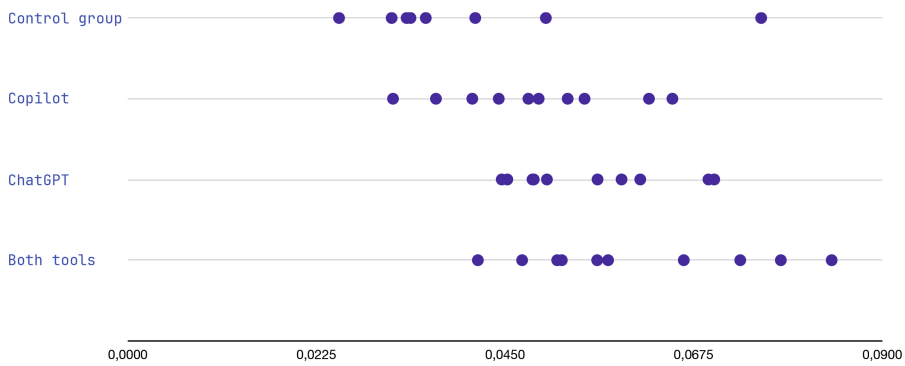


Fig. 1. Strip plot of taggings per word in journals of different groups.

we will discuss in the next section. In total, these journals contained 120980 words to which we added 6053 tags as described above. Each journal received 33–446 taggings (mean 159, median 129), with a density around 5 taggings per hundred words—see Fig. 1 for a strip plot of all values. Table 2 contains a summary on the number of taggings per tag per group, and Table 3 has the full data set.

4.2 Qualitative Findings

Architectural Design vs Coding Tasks. There seems to be a sweet spot for the use of GAI in programming a large project, where it could serve as a powerful assistant rather than a replacement for a critically thinking software engineer. One of the most common observations was that GAI performs poorly at high-level design and architecture—students often found that when they allowed GAI to dictate structure, the results were rigid, inefficient, or simply misaligned with their intent. As a result, many study participants settled for a hybrid approach: they took charge of the overall design process themselves but used GAI for code generation and implementation details, which often led to more effective outcomes. This was to be expected for Copilot which is its original intended use, but was also quite prominent for ChatGPT.

Table 2. Total number of taggings per tag (cf. Section 3) per group throughout all journals. Each of the 38 journals was written by one student participant and contains notes on the activities they were performing during the project, as well as a critical reflection on the entire process in the end.

joy	6	73	61	55
pain	67	89	128	123
boost	0	159	128	112
waste	9	122	182	133
manual	16	77	119	93
prompt	2	45	293	219
could have	0	27	24	59
would have	200	51	2	2
ai misuse	6	22	41	28
class diagram	14	19	24	15
sequence diagram	7	7	13	14
design pattern	12	20	16	19
project planning	8	14	22	7
refactoring	19	36	51	48
debugging	90	84	112	110
testing	88	164	255	136
reasoning	31	44	78	64
documentation	86	114	106	90
game rules	25	41	77	57
protocol	29	61	97	49
concurrency	32	77	72	68
network	49	65	37	56
user interaction	23	27	36	41
error handling	13	40	30	31
JML	14	12	65	15
TDD	1	1	1	1

Interestingly, some students deliberately avoided GAI for simple tasks, stating that it would likely "mess up their code"—a clear case of *code alienation*, a phenomenon well known in collaborating human developers: if some developer have strong feelings of code ownership, excessive updates to that code done by other developers or by program transformation tools (e.g., in the context of software migration and renovation), can destroy that ownership feeling. In this case, GAI-generated content apparently felt foreign or unpredictable to the developer. This was much more visible in ChatGPT-using participants, perhaps the very nature of Copilot being integrated into the IDE alleviated some of the code alienation feeling. Among participants using both tools, some reported significant

Table 3. All tagging statistics of this pilot study.

Tag	Control group										Copilot										ChatGPT										Both tools									
joy	2			2				2	10	5	27		12	1	11			7	2	3	21	12	5	9	2	2	3	2	4		7	3	11	9	4	16	1	
pain	13	12	11	11	4	7	4	5	1	2	20	1	22	5	6		8	24	6	10	30	38	5	18	6	8	3	4	7	2	7	2	17	25	25	23	6	9
boost									32	13	29	17	22	8	20		5	13	16	16	19	11	15	27	7	5	2	10	9	5	7	2	15	14	12	31	7	10
waste	1	4	1	1			2		13	15	20	9	10	13	15		5	22	11	7	53	27	14	18	14	13	13	12	12	8	13	3	18	12	18	32	8	9
manual	6	1	3		1	1	2	2	2	11	10	7	11	11	8	1	6	10	14	7	12	23	13	23	9	12	2	4	5	5	15		13	11	10	28	4	2
prompt	2									2	3	6	3	12	11			8	30	15	52	48	33	70	12	16	4	13	31	14	15	12	16	26	25	56	10	14
could have										4	1	3	4	3	7		1	4	1	3	1	2	4	10		3			3	4	8	1	1	1	6	29	2	4
would have	28	27	30	9	50	17	16	23	5	3	16	4	10	6	4		1	2								1		1	1				1					
ai misuse		1	2				1	2	3		3	1	5		1		1	8	9	3	2	4	4	8	9		1	1	3		1	2	9		3	5	1	4
class diagram	4	1	5		2	1	1		1	2	2		7	2	3			2	1	1	1		10	8		3			2	1				4	5	2		1
sequence diagram	2	1	2				1	1	1	2	1		2			1							12			1			1	1				7	4			1
design pattern	1	6					1	4	4		5	1	5			2	2	1	2	4	1	4		2		1		2					1	1	1	6	10	
project planning	1		1	3	1			2	1					4	5	1	1	2	1	3	4	1	1	2	6	1	2	1		1			1			2	2	1
refactoring	2	1	1	2	5	2	2	4	2	1	1	1	8	1	6	1	1	14	1		9	14	7	12	7	1			5	7	4		4	10	1	13	2	2
debugging	27	3	8	7	7	14	9	15	4	9	10	2	11	1	16	1	5	25	2	13	37	20	4	10	24	2			14	9	5	2	15	8	16	29		12
testing	1	18	10		22	10	7	20	8	9	20	10	30	14	24	24	11	14	12	4	63	35	59	39	33	7	1	2	16	7	4		15	20	19	38	7	10
reasoning	1	2	1	1	17	6		3		6	3	2	15	2	7			9	10	5	27	13	7	11	4		1		9	3	4	2	5	5	11	18	2	5
documentation	6	10	24		26		11	9	7	19	6	7	15	7	27	1	7	18	4	4	11	10	15	33	13	14		2	10	2	7	1	13	10	15	23	3	6
game rules	7	3	1	2		4	3	5	4	11	2	1	2	2	12	3	2	2	2	11	25	14	6	3	6	6	2	2	8	7	9	2	1	9	4	11	4	2
protocol	1	5	4	2	2	4	3	8	4	6	8	11	1	3	6	15	4	3	8	11	11	26	2	7	21	8	2	1	7	2	6		3	6	8	10	3	4
concurrency	1	4	3	1	8		7	8	8		6	3	10		6	21	2	21	15	1	1	7	10	9	8	15		6	6	2			4	11	4	34	5	2
network	3	7	4	15	9	5	3	3	8	4	16	12	5		3	6	5	6	4		2	15	3	7	1	3	1	1	2	2	1		1	3	13	21	8	5
user interaction	3	3	2	5	1	1	3	5	5	2	8	6			1	4		1	3	2	6	13	3	2	2	5			4		10	1	1	9	1	12	3	
error handling			2		2		2	7	2	1	10	3		2	8	6	3	5	4		10	8	5	1				2	4		3		3	1	5	6	4	5
JML		5		1			6	2					5		3			4		6	10	5	14	18	11			1	7	1	3		3			1		
TDD		1								1											1												1					

success and satisfaction from synergetic use, when the design was discussed with and/or suggested by ChatGPT, after which the implementation was done by Copilot with some minimal interference and/or mediation by the human.

Another emerging challenge was the gap between having a mental idea about software design and implementing it—and apparently translating it into code directly was perceived to be easier than translating it into an effective GAI prompt. This, however, might be a trainable skill which our students simply were unprepared for. Ultimately, while GAI accelerates the transition from idea to code, its effectiveness depends on the user's ability to define structure, provide clear guidance, and critically evaluate GAI-generated output. This reinforces the idea that GAI is most useful when paired with strong foundational knowledge and intentional design choices.

Supporting quotes:

- *This proves that the most effective way to use Chat GPT is to ask him to complete some missing parts/methods of the already existing code that you have created yourself.*
- *I notice that a bulk of my time is being spent on establishing architecture and design choices instead of actually programming.*
- *I did not find Chat GPT useful in general project designing, or class writing.*
- *ChatGPT does well at understanding the logic of the methods.*
- *Did not use chatGPT this hour because it was a very simple task and chatGPT may destroy what I have so far.*
- *I did not have to ask help from AI to write out the code, as I did not have any written description but had a mental idea of the structure.*
- *It did however incorrectly implement a try catch statement, in a way that Intellij wouldn't see as an error—so some knowledge on exception handling was necessary.*
- *if I hadnt known the basics of servers, I wouldn't have been able to get that.*
- *GPT only gave me a skeleton of a project that was not future proof.*

Acceleration vs Correctness. One of the most striking observations was how GAI transforms the coding workflow: study participants could simply feed requirements to a GAI tool and instantly receive functional code. In some cases, the output was immediately usable, while in many others, it provided a good foundation to refine. A similar way of working involved giving GAI an assessment rubric to generate work that would score well, which occasionally succeeded but is ultimately unrealistic for real-world software development, where predefined grading criteria do not exist, requirements tend to creep and mature solutions to rot.

The common theme among participants that this has led to, was the lack of cognitive effort required. Some students found it liberating, even enjoyable, to not have to think and still produce working code. However, this obviously comes at a price. While GAI accelerated productivity by enabling developers to generate more code faster, it also introduced uncertainty about correctness. In turn, that was leading to a growing awareness that GAI-produced solutions often require significant debugging and validation, sometimes overshadowing time won

by quickly producing working code. On several occasions participants reported that after many prompts, the overall code quality declined, technical debt accumulated rapidly, and managing GAI-induced complexity became an issue. Some found themselves completely unable to debug GAI-produced code, as it was too complex for them to fully grasp. Sometimes GAI's rapid output ended up slowing development down—one would generate large portions of code in minutes, only to spend subsequent hours debugging unexpected issues, in worst cases ultimately abandoning GAI solutions altogether and solving the problem from scratch manually. These experiences highlight an ongoing tension: while GAI reduces the barrier to writing code, it does not guarantee that the resulting code is maintainable, understandable, or even correct.

Supporting quotes:

- *First hour in, and I have just created the entire hex game.*
- [ChatGPT] *made the abstract class and an easy computer at first and let me decide how to implement it, copilot then implemented it for me.*
- *I think ChatGPT accelerates the implementation of basic methods; it is even useful for creating game logic. Correctness is the main problem.*
- *The logic from the suggestion was somewhat correct.*
- *The quality of that I didn't check.*
- *Copilot is deceptive, it looks like it is helping, but at some point or when a random bug starts popping up, you realize that the code was pretty bogus.*
- *I overspend time on trying to figure out the problem using ChatGPT.*
- *The tools help, but you have to be really careful in what you accept or don't accept.*
- *I did not have to think a second about this problem. ChatGPT solved it for me.*
- *I would have never created such a function since it is not within my knowledge.*
- *I feel like I would be spending more time debugging the code from AI than what I'm now spending on writing it on my own.*
- *When I said fix that please it gave me some optimizations which broke it completely.*
- *I have to look into the HexGame class and I do not know the code well since most of it is pasted from chatgpt.*

The Conversational Aspect. Some students claimed that their way of working would have been the same whether they did or did not have access to GAI. At least on some occasions we had to agree, up to the point where some reflective journals looked like URL lists with links to StackOverflow, Baeldung, YouTube and other sources of information, and the entire workflow was organised around seeking information or even working pieces of code to use in constructing their own system. In other cases, participants worked more in the IDE than in the browser, and only occasionally resorted to searching the web.

Yet this was always combined with anthropomorphising of the GAI: especially ChatGPT was repeatedly referred to with male pronouns (he/him), and the choice of verbs—e.g., "he forgot", "it struggles a lot", "he knows", "it decided/refused to implement" or "he makes no sense"—indicated perception of these prompt and response exchanges as true conversations. GAI became more than just a tool and turned into a discussion partner.

This ability to use GAI to not just generate code from natural language prompts, but also to discuss strong or weak points of envisioned design, allowed to overcome the challenges we named in the first theme of this section. Having a conversation partner with deep knowledge of concrete details of using pretty much any library and API in existence and being capable to maintain a conversation about any software design topic from algorithm design to code quality, also led to contradicting the second theme (acceleration and correction). Some participants expressed strong beliefs in correctness of GAI-produced code beyond the faith they had in their own.

One of the three most common regrets found in the **would have** category (the other two will be discussed in the next paragraph), is the lack of option to request an explanation. At least some of the representatives of the control group as well as the Copilot users followed the same process of "prompt-based programming" as one would do with GAI, just by using web search, reading posts on StackOverflow and guides on Baeldung, calling up friends, chatting them on Discord, etc., and regretted that getting to know some API or a language feature requires them to watch a long tutorial video instead ChatGPT just spoonfeeding that information directly.

Supporting quotes:

- *Normally I would ask ChatGPT how to generate this, but since I could not, I resorted to watching a YouTube video.*
- *No doubt stackoverflow had something similar.*
- *I'm basically just copy pasting solutions from the internet.*
- *I am not sure if AI could help me with this, but of course using AI will prevent bugs from being created in the first place.*
- *when I ask to compare different ways to implement a functionality, it provides a pros and cons list on how it would affect the product.*
- *I deleted my program code and Copilot was able to just fill in the code as it always wanted to.*
- *It was not of much help, because I didn't trust it to give me the correct answer since its answers seemed vague.*
- *Right now, I understand the code perfectly.*

Where GAI Helps the Most. There are three reasonably well-defined domains where GAI was perceived as being the most helpful both by participants who did use it, as well as by those who were not given that option. The first two strongly correspond to the remaining two categories of popular complaints around the **would have** tag: testing and documentation.

The perception of GAI's usefulness in testing appears to be largely illusory rather than substantiated by concrete benefits. The ultimate purpose of software testing is not to simply have a test suite with some high enough coverage of existing code, but to use it to find bugs or to prevent those bugs from entering the code in the future. While many participants—both those who used GAI tools and those who did not—eagerly recognised the potential of GAI-produced test cases, the data we have seen suggests that this perception is not grounded in actual

problem-solving instances. Nowhere in the reflective journals nor in focus group discussions did students report a case where a GAI-produced test case identified or prevented a bug that would not have been caught otherwise. Instead, the tests produced by GAI tend to appear comprehensive and sophisticated, inducing a sense of false security, and often incorporating a variety of edge cases and structured assertions. However, in practice, these test cases are the antitheses of TDD: they are green on arrival, since they pass successfully immediately upon generation, and they stay green throughout their lives, rarely contributing to meaningful debugging. This suggests that while GAI can generate plausible-looking tests, it does not inherently improve the iterative debugging process, and students may overestimate their effectiveness simply because they resemble well-structured test suites. In scenarios where GAI was used to take one or two manually written test cases to generalise and produce similar test cases covering slightly different but related methods, it could be argued that the GAI application was useful and justified.

Generating documentation, be it in the form of comments in the code, user manuals or structured documentation formats such as Javadoc or JML, appears to be a more promising application of GAI. Unlike GAI-produced test cases, which often serve more as a decorative layer than a functional tool for debugging, documentation produced by ChatGPT and GitHub Copilot exhibits a greater potential for practical utility. Participants frequently reported that GAI-assisted documentation helped them articulate design decisions better than they would have done themselves, and generate boilerplate descriptions that they could then refine further (or, in some cases, especially for Copilot, to go the other way and generate code from an extensive Javadoc).

One possible explanation for this difference lies in the nature of GAI's training data. GAI models struggle to generate truly novel test cases because it is a substantial reasoning challenge to find corner cases that truly need testing. Yet, they have been trained on large corpora which included lots of well-written technical documentation, including open-source repositories and software manuals, conforming to industrial best practices. As a result, ChatGPT in particular often produces coherent, contextually appropriate explanations, sometimes even suggesting JML constraints that align with best practices in formal verification. However, this remains an area that requires further research. Initial impressions do suggest that GAI-produced documentation is more than just surface-level filler, but we still need to investigate whether students actually understand and internalise what is being generated, or if they are merely accepting GAI outputs at face value. It is a challenge moving forward to establish whether GAI-produced documentation enhances learning and software quality, or whether it simply creates a false sense of completeness, much like GAI-produced test suites.

Interestingly, some participants of our study believed that the biggest gains from GAI they had or could have had, came from implementing the game rules themselves. The reasons might be at least partly algorithmic, since the game we have used in this study, used a hexagonal board, and it takes quite some design and thinking ahead to figure out what it means for places on such a

grid to be adjacent to one another or how to construct a path from one place to another. The other explanation could be similar to a writer's block: even though some implementation parts are relatively easy and straightforward, it can be considerably easier to start with some basic setup already coded, and fill in the blanks incrementally, rather than to start from scratch by creating first classes one by one. It requires much more data points that we could obtain, to be conclusive here (and to account for confounding factors like the choice of a programming language—since it is also possible that starting a new project in a scripting multiparadigm language like Python would be less of a mental burden).

Supporting quotes:

- *Chat GPT is a very useful tool for writing texts, providing information to a concern in a matter of seconds and debugging code.*
- *I have no idea how that JML part works and I have no idea what it has written down in the JavaDoc.*
- *I do believe that without the AI, the hardest part of the project would have been the gamelogic.*
- *On the report side of things, I found GPT to be a lifesaver. I practically made GPT generate all of my report.*
- *for the report side of things, it would be like getting homework from your friend. All you have to do is change it a little bit and the work is completed.*
- *Copilot does not help me think, it helps get rid of brainless typing.*
- *This took like 40 min to complete 4 diagrams. I am shocked because I remember this taking me a very long time 2 years ago.*
- *I solved the problems above myself because I did thought chatgpt would be more time consuming.*

Hallucinations are situations when GAI makes assumptions about the context which were neither provided directly by the user, nor can be derived from the prompts provided by the user. Initially observed in the computer vision domain where they were seen as a positive phenomenon, allowing creative movie-like zooming of low quality images by guessing and filling in content that was inherently missing [3], nowadays hallucinations are synonyms for confabulations, fabrications, delusions and other forms of falsification. We list four important subcategories here, referring to the work of Maleki et al. for a recent literature review on the matter [29].

Confabulations are GAI's responses which are plausible but not factually correct. For code generation, a confabulation would be producing method calls to methods that do not exist even though it would make sense to have methods with those names. **Fabrications** are GAI's responses where inaccurate information is added to the context without underlying reasoning and presented as factual. For code generation, a fabrication could lead to a GAI tool suddenly switching to a different programming language in the middle of the session simply because the training data indicates it more probable to have this algorithm implemented in a language other than what was requested. **Delusions** are responses which are blatantly wrong, which happens often in the context of debugging: in order to

advise the user on how to debug a certain problem, the GAI tool needs to come up with a hypothesis about the possible cause of the observed faulty behaviour, and since that can be far off (in addition to not being communicated properly), it is known to send users on a wild goose chase. Finally, there is **parroting** which stems from repeating patterns commonly observed in the training data—in particular dangerous in the context of programming education due to its tendency to not only easily solve all introductory programming exercises, but also to just as easily inject common mistakes into such solutions, combining them with very believable justifications.

Supporting quotes:

- (C) *And then I spotted another assumption that was made by the AI and it was wrong.*
- (F) *It seemed like a good solution, but when I asked it to implement it for me, it was implemented in Python.*
- (F) *apparently Copilot thought I was making Checkers.*
- (C/D) *Much other stuff is not exactly right.*
- (D) *I started to believe that the more questions I ask, the dumber it gets.*
- (D) *Also, some tests didn't really test what they were supposed to test.*
- (P) *I do notice that my code does look very bad now I use chatGPT.*

Learning Experience. By now the paradigm shift in how students approach programming tasks within a project when using GAI tools, should be obvious. The key question still remains, and it is not whether students will use these tools in the future—because they inevitably will—but rather whether they truly learn anything from them or while using them. Do students gain a deeper understanding of programming concepts, or does GAI obscure the cognitive processes necessary for effective learning? Do we, as educators, need to redefine what we teach, or simply adjust how we teach it?

Traditionally, project-based learning [15] has been rooted in active engagement: students learn by struggling through problems by themselves, making mistakes, debugging, iterating on their solutions, and getting timely actionable feedback to let them to keep going. The introduction of GAI shifts the balance: students are now less engaged in the *doing* and more engaged in the guiding of a GAI tool to do it for them. Such students are focused on the ultimate goal—getting the project to work, submitting it on time and getting a good grade—even though for teachers this has never been the real goal, and within the learning-by-doing paradigm the project deliverable is always secondary to the process of doing it.

We did have participants that reported that they did not feel they missed much from the learning experience, strongly suggesting that GAI merely accelerated parts of the project rather than replacing genuine problem-solving. Others, however, noted that they had a *false* sense of efficiency, since they assumed AI would make things easier, only to realise later that they ended up with code they do not understand and hence have no true control over. The most recurring theme was one of over-reliance: students who leaned too heavily

on GAI tools, often found themselves unable to reason about their own code whenever problems arose. Ironically, proceeding that way makes them go the full circle, since spending too much time on understanding or debugging GAI code the first time, makes them less eager to rely on it for anything in the next steps in fear that their productivity will take another hit.

In this sense, GAI introduces a new failure mode in programming education: instead of struggling with syntax and logic, students struggle with interpreting GAI-produced solutions. This is a new kind of problem, one that traditional programming education does not prepare students for. However, this is far from a universal experience. Some students did learn from GAI, especially when they approached it with a genuine desire to understand. Those who treated GAI as a tutor rather than an oracle—questioning its responses, verifying its correctness, iterating on its suggestions, persisting in demanding justifications and detailed adjustments—reported a meaningful learning experience.

Supporting quotes:

- *I don't think I missed too much in terms of the learning experience.*
- *Very easy to trust ChatGPT too much, which I have done again.*
- *I realized that working with ChatGPT is actually quite difficult.*
- *I simply have no idea how to even ask chatGPT for advice, as that would take explaining my entire codebase.*
- *I also have no idea how the function works so I will have to check that to fix it. Maybe I should ask chatGPT again.*
- *This is probably because I thought it would be quick an easy using AI, without it I would have first assessed what was actually necessary.*
- *Maybe I have become over reliant on the tool, because not using it probably made this go way faster.*
- *It's much faster to do this without AI and I get better results.*
- *I didn't try asking ChatGPT to do that, as I considered it would have taken me more time to explain exactly what I wanted, and even then it would have probably required a lot of changes from my side.*
- *I would thus think that Copilot did not prevent me from learning about coding it just took over the easy duties while leaving the more complicated work for me to figure out.*
- *I mostly did not have to think at all which did have some drawbacks.*
- *Coding wise I did not learn much, I do not know a lot about my own project.*
- *You can learn from chatgpt if you genuinely want to.*
- *This is definitely not good practice, but let's hope that this works.*

5 Implications

5.1 GAI as a Tutor

One of the most immediate and apparent uses of GAI in education is as an on-demand tutor, capable of providing explanations, clarifications, technical details at any time of day. Unlike human instructors, GAI does not have office hours, does not get tired, impatient, always delivers responses within seconds or minutes, and never complains. This convenience is undeniably valuable, and students

struggling with a difficult concept or an unfamiliar API can quickly obtain an explanation without waiting for a scheduled class or response from a peer. The ability to receive immediate feedback allows for just-in-time learning, potentially making students more independent and resourceful.

However, students have traditionally relied on teachers and textbooks as authoritative sources, and they still expect correctness from those they learn from. GAI models do not always distinguish between correct, misleading, and outright false information, and in sections above we provided much evidence of all these expectations being violated. A student who assumes that GAI-produced responses are always accurate, may unknowingly build a foundation of misconceptions, particularly when GAI confidently hallucinates plausible-sounding but incorrect explanations. If we assume that human tutors know their students and match their level of understanding as well as increasingly probe them for misconceptions, then GAI in its current forms fails quite decisively, lacking the ability to take enough context into account, be it about a particular student's needs or about "remembering" previous conversations and consistently building up from those in an informed and pedagogically sound way.

Beyond accuracy concerns, the ease of access to information may fundamentally alter how students engage with learning. Just as widespread calculator use has made people less proficient at mental arithmetic, and reliance on digital devices has diminished handwriting skills, continuous GAI-assisted explanations could lead to a decline in deep knowledge acquisition. Students may grow accustomed to looking up facts and reasoning rather than internalising them, reducing their ability to recall and apply concepts without GAI assistance. This could lead to a generation of programmers who can efficiently retrieve information but struggle to reason through problems independently. Some claim that this milestone has already been reached with developers who spend more working hours on StackOverflow and YouTube than in the actual editor, but extensive GAI tutoring institutionalises this.

Despite these concerns, GAI as a tutor has clear potential to enhance learning when used effectively. A student who actively engages with GAI explanations, cross-references multiple sources, and critically evaluates responses, may actually develop a stronger conceptual foundation than one relying solely on lectures and textbooks. The key difference will likely be in how GAI is used rather than whether it is available, and further research will be needed to determine whether it ultimately enhances or erodes understanding in the long run.

5.2 GAI as a Tool

In the ethics research community, the discussion whether GAI is "just a tool" or rather "more than a tool", is still ongoing [2]. However, if for the purpose of this section we assume that it is just a powerful tool, and we keep the educational model which relies on a guidance of a human instructor, then GAI would not replace the teacher but instead augments the learning process, much like an IDE that speeds up routine tasks like performing simple refactorings and provides immediate feedback in contexts like regression testing. This shifts the

nature of programming education: not by making traditional coding obsolete, but by requiring a new set of complementary skills. Just as using a calculator does not eliminate the need for mathematical reasoning, leveraging GAI tools does not mean students can ignore fundamental programming concepts. However, the way they engage with those concepts, may change significantly.

GAI tools, just like powerful IDEs, are tools not just for making code run, like compilers are, or tools for keeping track of the code, like version control systems. In addition to all that, they are decision-making aids, influencing how developers (and students as future developers) approach problems, how they structure their code, and even how they understand the boundaries of what is possible altogether. Because of this, GAI literacy becomes an essential skill. Students must learn to critically assess GAI-produced suggestions, recognise when they are incorrect or inefficient, refine their prompts to get meaningful results. Prompt engineering as the ability to formulate effective queries that yield useful GAI responses, is already becoming a crucial skill in the industry, and education must adapt accordingly. Ignoring this reality would be like insisting that students write assembly code or use punch cards when the world has long since moved to high-level languages and colour displays.

Modern software developers do not need to know assembly to be competent programmers, and many never write raw SQL because ORMs abstract away the details. If GAI-assisted coding becomes the norm, should we really resist it, or should we accept that programming is evolving? There is an argument to be made that, as long as students develop an understanding of problem-solving, software architecture, and debugging, the method of implementation might be secondary after all. There is still plenty to teach beyond writing code: understanding what to build, why certain approaches work, how to make existing systems better, etc. Even responsible GAI usage needs to be explicitly taught. Without structured guidance, students will inevitably use GAI anyway, but without learning where to trust it, how to verify it, and when to rely on their own skills instead. If we fail to integrate GAI into education, we do not prevent students from using it, we only ensure that they use it uncontrollably.

5.3 Skills GAI Cannot Replace

If GAI-assisted programming is here to stay, then perhaps education should specifically focus on what GAI cannot do. While GAI tools can certainly generate code, explanations, and even documentation, it does not reason in a critical way, verify correctness, or make decisions based on deeper understanding. If students are to become effective programmers, they must develop these skills themselves, not by passively consuming AI outputs, but by actively questioning, refining, and integrating them into a broader problem-solving process.

One key area where we found this to be evident, is documentation. Many students in our study postponed writing any kind of documentation (not just the report, but also code comments and Javadoc specifications) until the very end of the project, treating it as a separate, optional step to improve their grades, rather than an integral part of the development process. GAI makes it easy to

generate documentation, but this automation hides its true purpose. Of course, documenting code is not just about explaining it to others, but mostly about clarifying ideas for oneself. When developers write documentation alongside their code, they engage in an essential cognitive process: articulating design choices, recognising inconsistencies, ensuring that future modifications remain manageable. If students only document their work at the last minute, whether manually or via GAI, they miss the opportunity to use documentation as a real-time thinking aid rather than a retroactive justification.

The same applies to testing, which very much fell under the same bus. When done at the end of a project, testing serves only as an *a posteriori* bug-finding mechanism, revealing last-minute issues that must be patched before deployment. However, when testing is integrated throughout development, it becomes a tool for confidence and flexibility. Writing tests early allows programmers to refactor their code more freely, knowing that mistakes will be caught before they become deeply embedded. GAI can certainly generate test cases, but if students do not internalise why testing matters and at which point in their project is it meaningful, they risk falling into the same pattern of only testing at the last minute, rather than using tests as a way to find bugs early and guide development from the start.

Perhaps the most future proof skill is debugging GAI mistakes. Traditional debugging involves tracing logical errors, identifying faulty assumptions, and methodically isolating the source of a problem [55]. GAI-assisted programming introduces an entirely new category of debugging: correcting errors in GAI code that looks correct but is subtly wrong. This is an advanced skill, requiring not just programming knowledge but an understanding of how GAI makes mistakes, how to recognise hallucinations, and how to verify GAI outputs instead of blindly trusting them. Without explicit training, students struggle to identify these issues, much like novice programmers initially struggle with debugging their own mistakes.

There are enough skills we can cover in programming education, even if all the ones where GAI has any significant success, are eliminated. Self-explanation, project and time management, structured thinking, correctness verification, assumption validation, detecting faulty reasoning, and many others. This could be seen as an opportunity to redirect our focus to produce relevant experts in the GAI-driven world.

5.4 GAI and Assessment

The first detected instances of a student producing GAI-generated code, were instinctively considered academic misconduct: you did not write this code, you do not deserve the grade, come back and redo, or else. Our study has provided a lot of evidence in support of that. For instance, we see that GAI tools do impact the grades in a significant way (with Copilot having the most impact on getting higher grades for the code, and ChatGPT having the most impact on the quality of the documentation). As another concrete example, we see that if given a chance, students generate project documentation, including drawing diagrams,

instead of making it manually, and do that at the last possible moment. This supports that the assessment methods of the pre-GAI world, where students were expected to independently produce every line of code, every report, and every test case, themselves, are based on an obsolete assumption.

Declaring any use of GAI as academic misconduct is not a viable solution. Students will use GAI regardless of official policies, just as they already use StackOverflow, online documentation, automated debugging tools, plugins. Blanket prohibitions are not only unenforceable but also misguided, since GAI is quickly becoming an integral part of professional software development, and can be considered an important career skill. Thus, the position of many teachers has gradually shifted towards a much more timid demand that one must *understand* the code they submit, or be able to *explain* how it works, or any combination of those.

If our module is called *Software Systems*, ideally we would like to test whether students produced a well designed, cleanly coded, functional solution, which they can also maintain and evolve. We would like to assign higher grades to students who excel at those criteria, not necessarily differentiating between those that used more or less tools to arrive at that skillset. This is not a new issue: in group projects, for example, freeriding [34] has always been a challenge, where weaker students can benefit from stronger teammates without contributing equally. GAI extends this problem beyond group work: now, an individual student can appear more competent than they actually are, simply by prompting GAI to do the hard work for them.

Some possible alternative assessment methods include:

- Oral exams, where students must defend their code, explaining design choices and debugging steps.
- Project exams, where students answer questions about their submitted work, ensuring they understand the details.
- In-class coding tasks, where students work under controlled conditions, demonstrating their ability to think through problems independently.
- Live demonstrations and walkthroughs, requiring students to articulate their approach and showcase their decision-making process.

These approaches shift the focus away from final deliverables and toward process, reasoning and understanding. Rather than assessing whether a project "just works", exams and interactive assessments ensure that students have actually engaged with and learned from the experience. Many institutions are experimenting now with these assessment forms, as well as with a range of university-level and programme-level policies. At least some of those are simply transparency requirements, demanding that students disclose when and how they used GAI, which is a very legalese, shortsighted and bureaucratic way, since it again takes away the focus from learning how to make a good software system to learning how to rehearse the steps of using one of the many tools in one's toolbox. Future proof rethinking of educational programming projects may focus less on writing raw code from scratch and more on interpreting, refining, debugging and

integrating GAI-generated solutions. The role of the educator, then, shifts from evaluating code quality to evaluating how students engage with GAI responsibly, which we covered in the previous sections.

6 Conclusion

Our study concerned the impact of GAI tools on programming education, specifically in the context of project-based learning which implies large holistically assessed take-home assignments. Our general conclusion is that students who had access to OpenAI ChatGPT and GitHub Copilot, generally produced higher-quality code and documentation, but their learning experience was noticeably altered. While some claimed that GAI simply accelerated their work without diminishing their understanding, others found themselves struggling to engage meaningfully with the GAI-produced code, raising concerns about their learning and cognitive development.

One of the most significant takeaways is that GAI use is unavoidable. When explicitly permitted, students engaged fully and explored various ways to utilise GAI tools. When not permitted, they regretted the restriction—Copilot users tended to miss some features only provided by ChatGPT, and control group missed both; Copilot was never explicitly missed in student journals, but repeatedly reported as useful even in the presence of ChatGPT. Participants of the control group loathed their limitations and resorted to emulating the same experience with Google Search, StackOveflow and similar pre-GAI-era means. This suggests that educational institutions must adapt assessment methods, ensuring that learning is measured through understanding, reasoning, and process, rather than simply evaluating final project outcomes. Traditional grading models are increasingly ill-suited for a world where GAI can generate functionally correct solutions with minimal human intervention.

Instead of resisting GAI, programming education must shift focus toward teaching what GAI cannot do fully: critical thinking, debugging, testing, structured problem-solving, software design, reasoning and verification. We hope that the detailed breakdown of our findings in Subsect. 4.2 will support our colleagues and fellow educators in making informed decisions and taking sensible steps towards future proof education and assessment.

Ultimately, GAI is neither a shortcut nor a replacement for learning. It is a tool that can either enhance or erode education, depending on how it is integrated. If used thoughtfully, it has the potential to make students faster, more efficient, and better-prepared for a career in the software industry. If used recklessly, it risks producing programmers who can generate code but cannot understand, debug, or improve it. The future of programming education will depend on how well we strike this balance, ensuring that GAI serves as an amplifier of human intelligence rather than a crutch that weakens it. Much larger studies are needed as further research to refine strategies for GAI integration, assess long-term impacts on learning outcomes, and ensure that our education continues to produce competent software engineers.

References

1. Admiraal, C., van den Brink, W., Gerhold, M., Zaytsev, V., Zubcu, C.: Deriving modernity signatures of codebases with static analysis. Special Issue J. Syst. Softw. Open Sci. Softw. Eng. Res. (JSS) **211** (2024). https://doi.org/10.1016/j.jss.2024.111973
2. Babushkina, D.: Are we justified attributing a mistake in diagnosis to an AI diagnostic system? AI Ethics **3**(2), 567–584 (2023). https://doi.org/10.1007/S43681-022-00189-X
3. Baker, S., Kanade, T.: Hallucinating faces. In: Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition, pp. 83–88 (2000). https://doi.org/10.1109/AFGR.2000.840616
4. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: can language models be too big? In: Proceedings of the Conference on Fairness, Accountability, and Transparency, pp. 610–623. FAccT, ACM (2021). https://doi.org/10.1145/3442188.3445922
5. Bommasani, R., et al.: On the opportunities and risks of foundation models (2022), https://arxiv.org/abs/2108.07258
6. Boughattas, N., Neji, W., Ziadi, F.: Project based assessment in the era of generative AI-challenges and opportunities. In: Proceedings of the 20th International CDIO Conference, pp. 347–356 (2024), https://www.cdio.org/sites/default/files/documents/360_CDIO%202024%20Proceedings.pdf
7. Brockman, G., Lightcap, B., Murati, M., Clark, C.: OpenAI ChatGPT. https://chat.openai.com/chat/ (2022)
8. Burdy, L., et al.: An overview of JML tools and applications. Int. J. Softw. Tools Technol. Transfer **7**(3), 212–232 (2004). https://doi.org/10.1007/s10009-004-0167-4
9. Cao, H., et al.: A survey on generative diffusion models. IEEE Trans. Knowl. Data Eng. **36**(7), 2814–2830 (2024). https://doi.org/10.1109/TKDE.2024.3361474
10. Ciniselli, M., Pascarella, L., Bavota, G.: To What extent do deep learning-based code recommenders generate predictions by cloning code from the training set? (2022), https://arxiv.org/abs/2204.06894
11. Clandinin, D.J., Caine, V.: Narrative inquiry. In: Reviewing Qualitative Research in the Social Sciences, pp. 178–191. Routledge (2013). https://doi.org/10.4135/9781412963909.n275
12. Denny, P., Luxton-Reilly, A., Tempero, E., Hendrickx, J.: CodeWrite: supporting student-driven practice of java. In: Proceedings of the 42nd ACM Technical Symposium on Computer Science Education, pp. 471–476. SIGCSE, ACM (2011). https://doi.org/10.1145/1953163.1953299
13. Farooq, A., Zaytsev, V.: There is more than one way to zen your python. In: Visser, E., Kolovos, D., Söderberg, E. (eds.) Proceedings of the 14th International Conference on Software Language Engineering (SLE), pp. 68–82. ACM (2021). https://doi.org/10.1145/3486608.3486909
14. Freise, L.R., Bruhin, O., Ritz, E., Li, M.M., Leimeister, J.M.: Code and craft: how generative AI tools facilitate job crafting in software development. Available at SSRN 4974037 (2024), https://ssrn.com/abstract=4974037
15. Gary, K.: Project-based learning. Computer **48**(9), 98–100 (2015). https://doi.org/10.1109/MC.2015.268
16. GitHub: GitHub Copilot — Your AI Pair Programmer. https://github.com/features/copilot (2021)

17. Goodfellow, I., et al.: Generative adversarial networks. Commun. ACM **63**(11), 139–144 (2020). https://doi.org/10.1145/3422622
18. Höög, O., Ljungqvist, P.: Competence dynamics in the age of AI: a qualitative study examining how generative AI alters the relevance and need of competences among high-skilled workers. Master's thesis, Graduate School, School of Business, Economics and Law, University of Gothenburg (2024), https://hdl.handle.net/2077/82431
19. Hough, L.: Truce be told. Harward Ed. https://www.gse.harvard.edu/ideas/ed-magazine/11/09/truce-be-told September 2011
20. Hytti, H., Takalo, R., Ihalainen, H.: Tutorial on multivariate autoregressive modelling. J. Clin. Monit. Comput. **20**(2), 101–108 (2006). https://doi.org/10.1007/s10877-006-9013-4
21. Kallick, B., Zmuda, A.: Orchestrating the move to student-driven learning. Educ. Leadersh. **74**(6), 53–57 (2017), https://www.learningpersonalized.com/wp-content/uploads/2017/07/Article-Orchestrating-the-Move_KallickZmuda.pdf
22. Kingma, D.P., Welling, M.: Auto-encoding variational bayes (2022), https://arxiv.org/abs/1312.6114
23. Kokotsaki, D., Menzies, V., Wiggins, A.: Project-based learning: a review of the literature. Improv. Sch. **19**(3), 267–277 (2016). https://doi.org/10.1177/1365480216659733
24. Kramer, D.: API documentation from source code comments: a case study of Javadoc. In: Proceedings of the 17th Annual International Conference on Computer Documentation, pp. 147–153. SIGDOC, ACM (1999). https://doi.org/10.1145/318372.318577
25. Kusam, V.A.: Generative-AI Assisted Feedback Provisioning for Project-based Learning in CS Education. Master's thesis, University of Michigan–Dearborn, Dearborn, USA (2024). https://doi.org/10.7302/22651
26. Kusuma, J.S., Halim, K., Pranoto, E.J.P., Kanigoro, B., Irwansyah, E.: Automated essay scoring using machine learning. In: Proceedings of the Fouth International Conference on Cybernetics and Intelligent System, pp. 1–5. ICORIS (2022). https://doi.org/10.1109/ICORIS56080.2022.10031338
27. Lau, S., Guo, P.: From "Ban it till we understand it" to "Resistance is futile": How university programming instructors plan to adapt as more students use AI code generation and explanation tools such as ChatGPT and GitHub copilot. In: Proceedings of the ACM Conference on International Computing Education Research, vol. 1, pp. 106–121. ICER, ACM (2023). https://doi.org/10.1145/3568813.3600138
28. Leung, J.K.L.: Applied generative AI for interdisciplinary projects. In: Effective Practices in AI Literacy Education: Case Studies and Reflections, pp. 179–188 (2024). https://doi.org/10.1108/978-1-83608-852-320241019
29. Maleki, N., Padmanabhan, B., Dutta, K.: AI hallucinations: a misnomer worth clarifying. In: Proceedings of the IEEE Conference on Artificial Intelligence (CAI), pp. 133–138 (2024). https://doi.org/10.1109/CAI59869.2024.00033
30. Mastropaolo, A., et al.: On the robustness of code generation techniques: an empirical study on GitHub Copilot . In: Proceedings of the 45th IEEE/ACM International Conference on Software Engineering (ICSE), pp. 2149–2160. IEEE Computer Society, May 2023. https://doi.org/10.1109/ICSE48619.2023.00181
31. Newkirk, J., Vorontsov, A.A.: Test-Driven Development in Microsoft .NET. Microsoft Press, Redmond, WA (2004)
32. Niszczota, P., Conway, P.: Judgments of research co-created by generative AI: experimental evidence (2023), https://arxiv.org/abs/2305.11873

33. O'Grady, M.J.: Practical problem-based learning in computing education. ACM Trans. Comput. Educ. (TOCE) **12**(3) (2012). https://doi.org/10.1145/2275597.2275599
34. Palfrey, T.R., Rosenthal, H.: Testing Game-Theoretic Models of Free Riding: New Evidence on Probability Bias and Learning. Cambridge, Mass.: Department of Economics, Massachusetts Institute of Technology (1990), https://dspace.mit.edu/bitstream/handle/1721.1/64219/testinggametheor00palf.pdf
35. Papamakarios, G., Pavlakou, T., Murray, I.: Masked autoregressive flow for density estimation. In: Guyon, I., (eds.) Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017), https://proceedings.neurips.cc/paper/2017/hash/6c1da886822c67822bcf3679d04369fa-Abstract.html
36. Pearce, H., Ahmad, B., Tan, B., Dolan-Gavitt, B., Karri, R.: Asleep at the keyboard? Assessing the security of GitHub Copilot's code contributions. Commun. ACM **68**(2), 96–105 (2025). https://doi.org/10.1145/3610721
37. Pesovski, I., Santos, R., Henriques, R., Trajkovik, V.: Generative AI for customizable learning experiences. Sustainability **16**(7) (2024). https://doi.org/10.3390/su16073034
38. Pudari, R., Ernst, N.A.: From copilot to pilot: towards AI supported software development (2023), https://arxiv.org/abs/2303.04142
39. Pudasaini, S., Miralles-Pechuán, L., Lillis, D., Llorens Salvador, M.: Survey on AI-generated plagiarism detection: the impact of large language models on academic integrity. J. Acad. Ethics (2024). https://doi.org/10.1007/s10805-024-09576-x
40. Qadir, J.: Engineering education in the era of ChatGPT: promise and pitfalls of generative AI for education. In: 2023 IEEE Global Engineering Education Conference (EDUCON), pp. 1–9 (2023). https://doi.org/10.1109/EDUCON54358.2023.10125121
41. Ranzato, M., Boureau, Y.L., Chopra, S., LeCun, Y.: A unified energy-based framework for unsupervised learning. In: Meila, M., Shen, X. (eds.) Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics. Proceedings of Machine Learning Research, vol. 2, pp. 371–379. PMLR, March 2007, https://proceedings.mlr.press/v2/ranzato07a.html
42. Rump, A., Zaytsev, V., Mader, A.: Requirements for an automated assesment tool for learning programming by doing. In: Proceedings of the 18th IEEE International Conference on Software Testing, Verification and Validation (ICST) (2025)
43. Sajadi, S., Huerta, M., Ryan, O., Drinkwater, K.: Harnessing generative AI to enhance feedback quality in peer evaluations within project-based learning contexts. Int. J. Eng. Educ. (2024), https://www.ijee.ie/1atestissues/Vol40-5/02_ijee4488.pdf
44. van Santen, J.: Using LLM Chatbots to improve the learning experience in functional programming courses. Bachelor's thesis, Universiteit Twente, Enschede, The Netherlands, February 2024, http://purl.utwente.nl/essays/98155
45. Shaer, O., Cooper, A.: Integrating generative artificial intelligence to a project-based tangible interaction course. IEEE Pervasive Comput. **23**(1), 63–69 (2024). https://doi.org/10.1109/MPRV.2023.3346548
46. Tamilselvi, C., Maanu P.A., Priya T.A., Kalaiyarasi, R., Nithiyasree P., Mohanaprakash T.A.: Empowering coders: revolutionizing programming education with NLP and challenge-based learning. In: Proceedings of the Third International Conference on Smart Technologies and Systems for Next Generation Computing, pp. 1–6. ICSTSN (2024). https://doi.org/10.1109/ICSTSN61422.2024.10670818

47. University of Twente: learning-by-interacting: the university of Twente vision on learning and teaching. https://www.utwente.nl/en/service-portal/organisation-regulations-and-codes-of-conduct/vision-on-learning-and-teaching April 2023
48. Vahid, F.: AI in CS education: opportunities, challenges, and pitfalls to avoid. ACM Inroads **15**(3), 52–57 (2024). https://doi.org/10.1145/3679205
49. Vargas-Murillo, A.R., Pari-Bedoya, I.N.M. de la A., Guevara-Soto, F. de J.: The ethics of AI assisted learning: a systematic literature review on the impacts of ChatGPT usage in education. In: Proceedings of the 2023 8th International Conference on Distance Education and Learning, pp. 8–13. ICDEL, ACM (2023). https://doi.org/10.1145/3606094.3606101
50. Walters, W.H.: The effectiveness of software designed to detect AI-generated writing: a comparison of 16 AI text detectors. Open Inf. Sci. **7**(1), 20220158 (2023). https://doi.org/10.1515/opis-2022-0158
51. Willis, S., Byrd, G., Johnson, B.D.: Challenge-based learning. Computer **50**(7), 13–16 (2017). https://doi.org/10.1109/MC.2017.216
52. Wood, D.F.: Problem based learning. BMJ **326**(7384), 328–330 (2003). https://doi.org/10.1136/bmj.326.7384.328
53. Wu, T., Chang, M.: Application of generative artificial intelligence to assessment and curriculum design for project-based learning. In: Proceedings of the International Conference on Engineering and Emerging Technologies, pp. 1–6. ICEET (2023). https://doi.org/10.1109/ICEET60227.2023.10525933
54. Yan, L., Greiff, S., Teuber, Z., Gašević, D.: Promises and challenges of generative artificial intelligence for human learning. Nat. Hum. Behav. **8**(10), 1839–1850 (2024). https://doi.org/10.1038/s41562-024-02004-5
55. Zeller, A.: Why programs fail: a guide to systematic debugging. Elsevier Science (2009). https://doi.org/10.1016/B978-1-55860-866-5.X5000-0

Health Care - Approaches Using Formal Methods and AI

Health Care – Approaches Using Formal Methods and AI

Martin Leucker[1] and Violet Ka I. Pun[2]

[1] Universität zu Lübeck, Lübeck, Germany
leucker@isp.uni-luebeck.de
[2] Western Norway University of Applied Sciences, Bergen, Norway
Violet.Ka.I.Pun@hvl.no

1 Summary

To ensure future high quality health care support within given financial conditions, a digitalization of the healthcare sector is mandatory. The digitalization is implemented using either conventional software development or techniques from artificial intelligence and faces two main important challenges: First, health care is a safety critical domain and requires the use of formal methods to ensure that the systems work as required. Second, the use of artificial intelligence in safety critical domains is still not fully understood.

Formal methods build on precise mathematical modelling and analysis to verify a system correctness. It comprises static and dynamic analysis techniques like model checking, theorem proving, runtime verification, to mention the most prominent ones. Its theoretical foundations have been developed in the past decades, but its application in various domains remains a challenge. AI in healthcare is transforming the field by improving diagnostics, aiding in medical imaging analysis, personalizing treatment, and supporting clinical decision-making. It enables faster and more accurate analysis of medical data, enhances drug discovery, and assists in robot-assisted surgeries. AI also contributes to predictive analytics, virtual assistants, wearable devices, and clinical decision support. However, it is important to remember that AI is a tool to support healthcare professionals rather than replace them, and ethical considerations and data privacy are crucial in its implementation.

This track focuses on the exchange of ideas around key issues in digital healthcare, including the formal modelling and optimization of hospital workflows, rigorous validation and clinical implementation of algorithms, and ensuring system robustness and reliability. It emphasizes the importance of effective human-AI collaboration, long-term impact and cost-effectiveness of digitalization, and the explainability of AI systems. Additional topics include data integration and quality, ethical and legal considerations such as privacy and bias, and the development of appropriate regulatory and policy frameworks. The overall aim is to tackle technical, ethical, legal, and societal challenges to enhance the benefits of digital healthcare solutions.

The lively discussions resulted in three contributions that are summarized in the following and the later published in this volume.

Towards Person-Owned and Controlled Personal Health Records: Past, Present, and Future Research at eMedLab, TalTech [3] explores a paradigm shift in personal health data management, advocating individual ownership and sovereign control over health records. It investigates the use of decentralized content-addressable storage (DCAS) networks as a foundation for personal health records (PHRs), contrasting them with traditional server-based systems such as hospital electronic health records, national health information systems, and commercial health apps. The paper defines the principles of DCAS-based health data storage and examines key organizational and technical considerations, including data custody, security, privacy, data quality, transparency, integrity, anonymization, pseudonymization, and interoperability. These aspects are analyzed in the context of ongoing research at eMedLab, Tallinn University of Technology. The paper also outlines preliminary scenarios for integrating DCAS-based PHRs into existing healthcare IT infrastructures within the framework of the European Health Data Space (EHDS), addressing both primary and secondary uses of health data. It contributes to the broader discourse on secure, interoperable, and ethically sound health data management through a model centered on individual control.

Quantum Machine Learning in Precision Medicine and Drug Discovery – A Game Changer for Tailored Treatments? [1] addresses the challenges posed by healthcare digitization, such as the complexity of biological systems, large-scale data generation, and the need for personalized treatments, noting that traditional computational methods often fall short. It explores the potential of Quantum Computing (QC) and Quantum Machine Learning (QML) to transform medicine through enhanced diagnostic accuracy, personalized care, and improved drug discovery. The paper highlights the integration challenges of quantum technologies, including algorithmic errors and high implementation costs. To address these issues, the paper proposes the use of formal methods—mathematically grounded techniques for specifying, developing, and verifying software—to improve the reliability and correctness of QC systems. It discusses the application of formal specification languages, model checking, theorem proving, and optimization techniques in genomic data analysis and quantum algorithm development. The paper concludes that formal methods can play a critical role in enabling QC to achieve its transformative potential in precision medicine.

LC/NC Pipeline for Training and Operationalising Segmentation Models in a Data Scarce Domain: De-arraying Tissue MicroArrays [2] presents a novel approach to training and deploying segmentation models for de-arraying Tissue Micro Arrays (TMAs), addressing challenges posed by limited high-quality datasets and strict privacy regulations in sensitive health domains. To overcome these barriers, the paper introduces a Low-Code/No-Code (LCNC) Domain-Specific Language (DSL) integrated into the Cinco de Bio (CdB) platform. This DSL is composed of Service-Independent Building Blocks (SIBs) that allow biologists to create model training pipelines without writing code. The methodology includes domain-specific data augmentation to generate pseudo-synthetic samples from minimal real data and utilizes AutoML techniques such as Neural Architecture Search and hyperparameter optimization to automate model development. Additionally, the paper outlines an architectural enhancement to the CdB platform through the adoption of a "Model as Data" paradigm, treating neural networks as dynamic, versioned data assets. This approach provides a scalable and practical solution

for addressing data scarcity and distribution shift, enabling domain experts to apply AI technologies effectively to their own datasets.

References

1. Bertl, M., Mott, A., Sinno, S., Bhalgamiya, B.: Quantum machine learning in precision medicine and drug discovery – a game changer for tailored treatments? In: Steffen, B. (ed.) Bridging the Gap Between AI and Reality – Second International Conference. AISoLA 2024. LNCS, vol. 16032, pp. xx–yy. Springer, Cham (2025)
2. Brandon, C., Fennell, Ã., Singh, A., Margaria, T.: LC/NC pipeline for training and operationalising segmentation models in a data scarce domain: De-arraying tissue microarrays. In: Steffen, B. (ed.) Bridging the Gap Between AI and Reality – Second International Conference. AISoLA 2024. LNCS, vol. 16032, pp. xx–yy. Springer, Cham (2025)
3. Piho, G., et al.: Towards person-owned and controlled personal health records: past, present, and future research at eMedLab, TalTech. In: Steffen, B. (ed.) Bridging the Gap Between AI and Reality – Second International Conference. AISoLA 2024. LNCS, vol. 16032, pp. xx–yy. Springer, Cham (2025)

Towards Person-Owned and Controlled Personal Health Records: Past, Present, and Future Research at eMedLab, TalTech

Gunnar Piho[1(✉)], Toomas Klementi[1], Igor Bossenko[1], Kristian Kankainen[1], Marten Kask[1], Olga Vovk[1], and Peeter Ross[2,3]

[1] Department of Software Science, Tallinn University of Technology (TalTech), Akadeemia tee 15A, 12616 Tallinn, Estonia
gunnar.piho@taltech.ee
[2] Department of Health Technologies, TalTech, Tallinn, Estonia
[3] Research Department, East Tallinn Central Hospital, Tallinn, Estonia

Abstract. We explore a potential paradigm shift in personal health data management, where individuals gain ownership and sovereign control over their records. We assess the potential of decentralized content-addressable storage (DCAS) networks as a foundation for person-owned personal health records (PHRs). After defining the principles of DCAS-based health data storage, we contrast it with conventional systems such as hospital EHRs, national health information systems (e.g., Estonia's national health information systems, EHIS), and commercial apps collecting data from wearables. We analyze key organizational and technical aspects of DCAS-stored health data—custody, security, quality, transparency, anonymization, and interoperability—in the context of ongoing research at eMedLab, TalTech. Finally, we propose initial organizational, economic, and technical integration scenarios linking DCAS-based PHRs with existing national and commercial health IT systems. These are discussed within the European Health Data Space (EHDS) framework, considering both primary (care delivery) and secondary (research, public health, AI) uses. We contribute to the debate on secure, interoperable, and ethically sound health data governance by outlining this shift toward full individual data ownership.

Keywords: health data primary and secondary use · health data ownership · health data integrity · health data privacy · Personal Health Record · European Health Data Space · decentralized content-addressable storage network

This research has been supported by the 'ICT Programme' of the European Union through the European Social Fund and the IT Academy research measures [28] and by the 'Digital health for a whole and healthy society' [18] and 'Medication Adherence and Treatment Efficacy in Patients with Dyslipidaemia and Achievement-oriented Novel Patient Digital Support' [19] research grants.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 81–103, 2026.
https://doi.org/10.1007/978-3-032-01377-4_4

1 Introduction. Person-Owned and Controlled PHRs

The exponential growth of digital data, totalling 149 zettabytes in 2024 and projected to reach 394 zettabytes by 2028 [57], involves vast amounts of personal data. The General Data Protection Regulation (GDPR) defines personal data as any data relating to an identified or identifiable natural person. Approximately 30% of digital data falls into this category, equating to about 5 TB of newly generated personal data per person in 2024 alone [14]. Personal data encompasses a wide range of information, including electronic communications, social media, financial transactions, browsing history, geolocation, health records, and data from smart wearables and domestic medical devices. Service providers collect and store this data, each operating under different privacy policies. Often, individuals are unaware of the full extent of data collection, how their data is used, or with whom it is shared. Many feel they have little choice but to hand over their data to access digital services. While alternatives exist, such as self-hosted email servers or private cloud storage, these solutions require technical expertise and remain niche—third parties are are usually entrusted with personal data, often without scrutiny.

Unstructured personal data, such as text and images, was challenging to process before the AI era. However, rapid advancements in AI have changed this [62]; AI models can now analyze vast amounts of personal data with unprecedented accuracy. Initially, personal data was primarily exploited for targeted advertising, but AI-driven analysis enables advanced profiling and behavioral prediction. This raises concerns, especially as big technology companies, including the Big Five (Google, Apple, Facebook, Amazon, and Microsoft), store comprehensive user data across email correspondence, browsing activity, financial transactions, and health records. Although companies assure users that personal data is protected, trust in these claims often remains a matter of faith. Government access to private data is virtually guaranteed in authoritarian states. In democratic societies, privacy laws generally protect personal data, yet these safeguards may not be absolute. History demonstrates that political and regulatory environments can change rapidly, and whoever controls data holds immense power. If fragmented personal data were fully aggregated, the entity managing this dataset would possess an alarmingly detailed profile of every individual, potentially enabling unprecedented social control. The question remains: can society be sure such a dystopian scenario will never unfold?

Health data is among the most sensitive categories of personal information, warranting additional protections under the GDPR. However, many people disclose their health-related data without considering the implications. For example, pharmacy loyalty cards reveal purchase histories that may indicate specific medical conditions. Similarly, fitness trackers and smartwatches continuously monitor users' physical activity and health metrics, transmitting data to commercial service providers. Most users are unaware of where this data is sent, who can access it, or how it might be utilized. Public awareness of these risks is limited, yet the implications are profound. With AI systems becoming increasingly capable of deriving deep insights from personal data, the debate over data ownership and

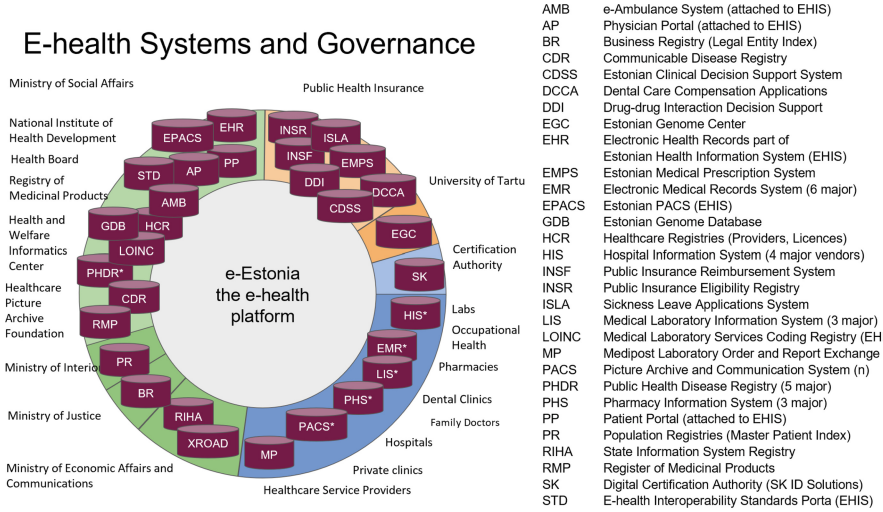
control is more relevant than ever. As the boundaries between public and private information blur, it is crucial to reassess whether individuals should have greater sovereignty over their data and what measures are necessary to ensure secure, ethical, and transparent data management moving forward.

A personal health record (PHR) [46] is an electronic health record system managed by an individual that contains their health-related information. Unlike electronic health records (EHRs) or electronic medical records (EMRs), which healthcare providers maintain, a PHR allows individuals to collect, manage, and share their health data as needed. A PHR may include a medical history, diagnoses, medications, immunizations, lab results, and other health-related details. This system empowers individuals to track and control their health information, promoting active engagement in their healthcare management.

The eMedLab research group at Tallinn University of Technology (TalTech) is exploring the feasibility of adopting PHRs, replacing EHRs and EMRs to grant individuals full ownership of their data and sovereign control over how it is shared for both primary and secondary use. Our work on this topic began in 2020 when we secured funding [28] for five doctoral researchers. Between 2020 and 2024, our group's primary research focus was "The Primary and Secondary Use of Health Data".

Currently health data is collected by healthcare providers and stored as primary data in their EMR software systems in various formats [6,7]. The primary use of health data is for direct patient care [10]. Primary health data typically consists of medical records, diagnostic test results, health conditions, and treatment plans essential for providing accurate and effective patient care. In contrast, using health data for purposes other than its initial collection intent (patient treatment) is considered secondary use. This includes data aggregation in EHR systems, financing, public health, research, policymaking, and other applications [6]. However, data generation does not always correlate with data processing, reuse, or understanding. Globally, the secondary use of health data remains a complex and unresolved issue [48].

Throughout 2020–2024, we aimed to map the key challenges and obstacles preventing the secondary use of real-world health data under the open FAIR (Findable, Accessible, Interoperable, and Reusable) principles for societal needs, such as public health and medical research advancement. We approached this issue with the understanding that over the past decade, the volume of digital health data has grown exponentially, and this trend is expected to continue. First and foremost, these data are used to provide immediate medical care to the patient, referred to as their primary use. The Estonian National Health Information System [43] (Fig. 1), which has been designed and developed with significant involvement from eMedLab researchers, both past and present, has been built based on the primary use requirements of health data, namely medical diagnosis and intervention needs. At the same time, it is widely acknowledged that processing and analyzing vast amounts of existing digital health data for research, healthcare financing, public health, policymaking, and healthcare improvement—known as its secondary use—holds immense potential for



AMB	e-Ambulance System (attached to EHIS)
AP	Physician Portal (attached to EHIS)
BR	Business Registry (Legal Entity Index)
CDR	Communicable Disease Registry
CDSS	Estonian Clinical Decision Support System
DCCA	Dental Care Compensation Applications
DDI	Drug-drug Interaction Decision Support
EGC	Estonian Genome Center
EHR	Electronic Health Records part of Estonian Health Information System (EHIS)
EMPS	Estonian Medical Prescription System
EMR	Electronic Medical Records System (6 major)
EPACS	Estonian PACS (EHIS)
GDB	Estonian Genome Database
HCR	Healthcare Registries (Providers, Licences)
HIS	Hospital Information System (4 major vendors)
INSF	Public Insurance Reimbursement System
INSR	Public Insurance Eligibility Registry
ISLA	Sickness Leave Applications System
LIS	Medical Laboratory Information System (3 major)
LOINC	Medical Laboratory Services Coding Registry (EHIS)
MP	Medipost Laboratory Order and Report Exchange
PACS	Picture Archive and Communication System (n)
PHDR	Public Health Disease Registry (5 major)
PHS	Pharmacy Information System (3 major)
PP	Patient Portal (attached to EHIS)
PR	Population Registries (Master Patient Index)
RIHA	State Information System Registry
RMP	Register of Medicinal Products
SK	Digital Certification Authority (SK ID Solutions)
STD	E-health Interoperability Standards Porta (EHIS)

Fig. 1. Estonian eHealth platform is a system of systems [41].

advancing human well-being. Yet, to this day, a comprehensive framework for the secondary use of real-life health data remains absent, leaving much of this potential unrealized. Our study explores the reasons behind this underutilization and proposes solutions to overcome it. We summarized the problem through the following three dilemmas of health data [37]: 1. *The dilemma of accessibility* describes the contradiction between protecting sensitive private health data and making it widely available to maximize its potential benefit for societal good. 2. *The dilemma of comprehensiveness* emphasizes the need to eliminate data fragmentation. Currently, health data is often scattered across multiple silos due to legacy systems and interoperability barriers. Without confidence that all the relevant information is included in a dataset, conclusions remain uncertain, since even a single missing fragment could, in theory, alther the outcome. 3. *The dilemma of ownership* exposes the discrepancy between the rights of data subjects and their ability to exercise those rights. Third parties typically store and manage health data, leaving individuals with limited control over their information despite being legally entitled.

2 Decentralized Content-Addressable Storage Networks

The solution to the described problem requires a fundamental paradigm shift in storing and handling personal data. This shift can be achieved using the nascent technology of decentralized content-addressable storage (DCAS) networks.

2.1 History and Characteristics of DCAS Networks

DCAS networks have their roots in early file-sharing platforms such as BitTorrent. BitTorrent operates on the principle that when multiple clients seek

to download the same file, the file is split into smaller fragments. Each client downloads a unique fragment and then shares the missing pieces with others, facilitating efficient bandwidth use. DCAS networks build upon the same idea, taking it a step further. They are P2P networks where nodes store only small file fragments, each just a few kilobytes in size, identified by their content hash. To reconstruct the original file, one must know its root hash. If this root hash remains exclusively with the data owner, they retain absolute control over the data, ensuring that no one else, even theoretically, can access it. The network is decentralized, which means that it operates entirely without a central authority.

Redundancy mechanisms, such as erasure coding, prevent data loss caused by network churn. Conceptually, a DCAS network operates like an electronic paper shredder (Fig. 2), which "destroys" a document by cutting it into tiny fragments and then disperses and stores each fragment independently and at random across a large number of autonomous, non-centrally controlled DCAS nodes in accordance with the protocol so that no single node can reconstruct the original information. Whereas a paper shredder's sole purpose is to obliterate data, DCAS is designed to preserve data securely: only the data owner, possessing the corresponding root-hash key, can safely reconstruct the fragments back into the original document on their own device.

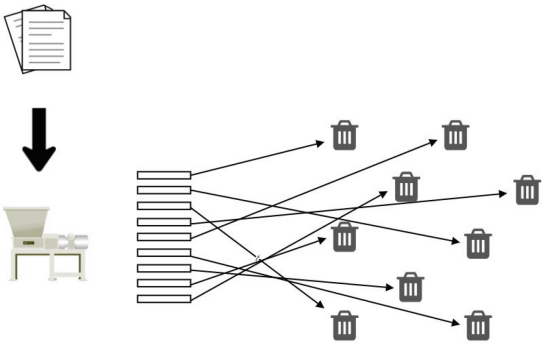


Fig. 2. DCAS network.

DCAS networks possess unique characteristics that make them well-suited to address the three dilemmas outlined in the previous section. First, the ownership dilemma is resolved since no third party controls the data; ownership is tied directly to the root hash. Second, the accessibility dilemma is addressed by placing the root hash under the exclusive control of the data owner. Without the root hash, nobody can access the data. However, access can be easily shared by providing the root hash. It is also worth noting that only a single person's data is affected if the root hash is compromised. Finally, the comprehensiveness dilemma is eliminated, as healthcare providers no longer need to maintain separate private data repositories. Instead, all relevant data remains within the DCAS network as a comprehensive whole.

2.2 Architecture for Managing Health Data on DCAS Networks

Each individual owns a PHR stored in their Personal Health Data Space (PHDS) on the DCAS network. When visiting a medical service provider, they grant access by sharing the root hash of their data. The provider can interact with the data but does not retain the root hash after the session. This process remains consistent regardless of the provider's profile or location, ensuring seamless mobility across institutions and borders. For secondary use, the data owner can generate a de-identified snapshot, which is also stored in the PHDS. The root hash of this de-identified copy can be shared—potentially for compensation—with indexing service providers that aggregate data from multiple individuals. The aggregated data is then made available for secondary users, such as researchers, policymakers, and pharmaceutical companies, supporting advancements in research, policy development, and drug innovation. Figure 3 shows the general architecture of the solution [37].

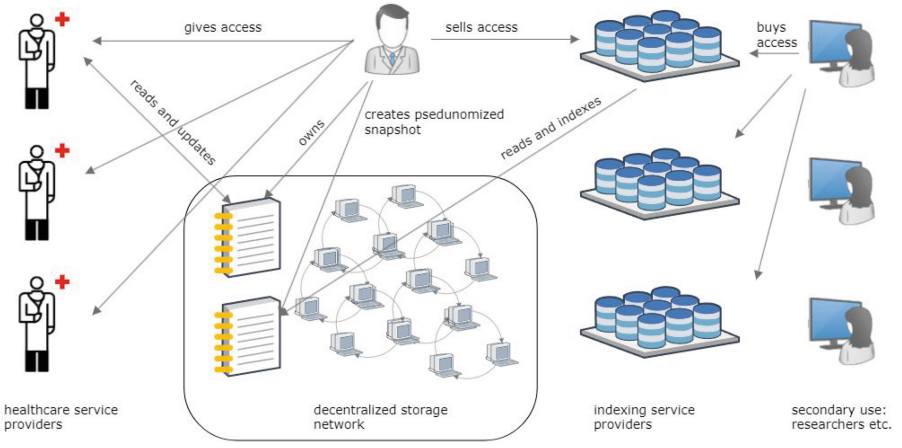


Fig. 3. General architecture of PHDS.

2.3 Proof of Concept

To validate the feasibility of the proposed solution, a proof-of-concept was developed to demonstrate the storage and retrieval of medical data within a real-world DCAS network. Due to accessibility restrictions, obtaining patient data is challenging but not essential for testing. Instead, it is sufficient for the data to resemble real-world information closely. For this purpose, synthetic data generated by the open-source Synthea package [55] was used. Ethereum Swarm [54] was selected as the underlying DCAS network. At the time of writing, Swarm hosts approximately 12,570 nodes (https://swarmscan.io) and is considered production-ready by its developers. Swarm distinguishes itself from other

DCAS networks by being open source, fully decentralized and operating without a central authority.

To evaluate the performance and scalability of the proof-of-concept app, synthetic data for 1,000 patients was generated and uploaded to the Swarm network. In total, 560,000 resources and 1.5 GB of data were uploaded. The root hashes of all uploaded datasets are publicly available as part of this work, allowing anyone with access to a Swarm node to verify their accessibility worldwide. A simple data browsing app was then used to simulate data access, demonstrating the feasibility of the proposed solution. The user experience of browsing the data closely resembled traditional Internet browsing, further validating the approach's practicality. Figure 4 shows the screenshot of the index page of the proof-of-concept app.

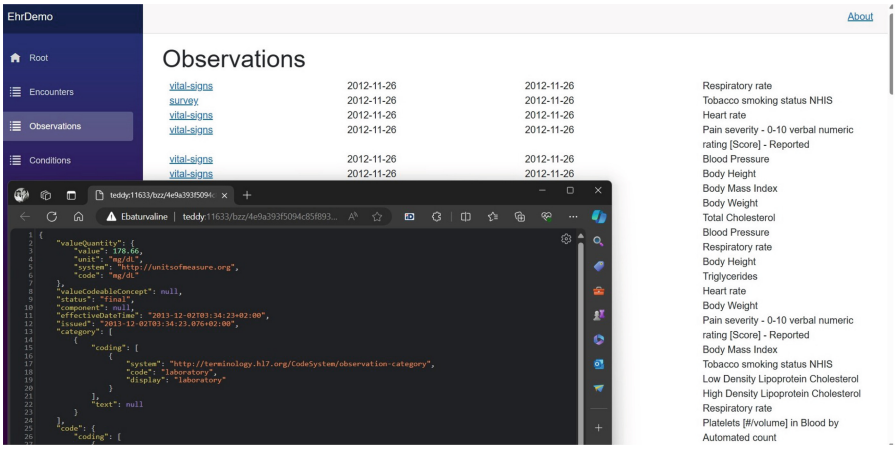


Fig. 4. The screenshot of the index page of the proof-of-concept app.

An extract of the 1,000 root hashes of the generated data is provided here for reference:

7d6b112bcac4bca10edc700265760fde1de343352e5421b84efcca6614cf848879e29ec0f0d713a3c20d9ce13e183850bad58640c696990cf6a89cdf96d9521c

907d099ce19286d5f498a38172ef4bf773a4896e6b1906bfc199576324d55915a8a44ec79a0322d0eec75ed29b71c1b675b09e167e2482059733674b86b4e2c6

79f43e307f8a71dfb7cee61e999ced2a59525d60b09984dc6f80dd99b2c914c60be623fe322583961fb9e45dcbdaa2f70c25abd5d92c30fe19e5c8dfb821fcfc

b86456a46ee50279aed562e62a342ca4dfe17f6f60e31e647b3cceb3f4f30f2adb5a4ca0d5879bfdcf31017516d410e47d69ad0e28256a8e701eedf492810614

1323c7aeb38ddfc7553ad381e2b7816ab83b013253f3040ff14316d0643b9ec813d2dfd87a6efc95bee5c2b842f0ebcdbd37b66a9b173d853dc2c58d080ffded

d65e5a6b7ee5fbfdadd6ca1178ae7aa20189d4dc663c622fd07572588efecfafee0bfae36b39049a8382091b8bb534085f05ce98e6d2e18957e4f2beaef09024

acef38c6ddb5c4fec1730209cf69b16c77c2c09485845cf43750f822edb64d2859d3e6aca5241400f06ecc4d1c9e0491641068b7a10a501ddba54165aee220c9

4527682cf6136eb43b4a33d9527d89096cf93a0e1c9a80051b14a8dbff325756c4dc36af137b6d0e523dbd7f6b65de15bea87309873306c056b9c193adb1cf64

c8760772565c1d1af77cc9bd72e0c79a2350c114560cba25ab365772947a3ff95d5de44c95e9fd7690cd8d6f7d3bef0f2304f5820c35d8dd93d88fb00226caa8

6a21ec60feb5c2cb0807ca7f356c9d856f2d472f83a19a12fe5913f79620f0602537fbd37410d86b7434a1ae81059a649c8b7115b821b54010281529cdb40ab3

Each hash represents a complete PHR for a single patient. Anybody with access to Ethereum Swarm can verify the data. The complete list can be obtained using the following Swarm hash:

c2da754a3c557256ef4c6e6f1295e723a678ae7bf9828f19a465ca1f972d55661fd3ac7024957d73831176dc94c9c2b5cdf4f2314f8e6661c492cf5cb785a8c7

3 Past Research. Challenges of Person-Owned PHRs

Our research in eMedLab at TalTech from 2020 to 2024 was focused on understanding the challenges in interoperability and accessibility needs for the secondary use of health data (Sect. 3.1), the associated de-identification (anonymization and pseudonymization) challenges (Sect. 3.2), challenges related to data transparency and integrity (Sect. 3.3), data modelling challenges (Sect. 3.4), and data semantic interoperability-related challenges (Sect. 3.5).

3.1 Analysis of Estonian National Health Information System

The *Health Sense* project [47], conducted from 2021 to 2024 under the *Green ICT* program and co-financed by the Norwegian financial mechanism, sought to address key challenges in modern healthcare data management. The focus was on analyzing an architecture for a *New Generation* of the Estonian National Health Information System (upTIS) to improve interoperability, accessibility, and the secondary use of health data. At the heart of the study was the recognition that health information, while critical for patient care, often remained fragmented and underutilized. The project aimed to shift from a document-based system to an event-based data-sharing model, ensuring that collected health data could be used multiple times without redundancy (Fig. 5).

Methodology and Implementation. We followed a structured methodology rooted in internationally recognized standards to achieve the project aim. The foundation was built on ISO 13940 (ContSys), which defines healthcare processes and relationships between data entities. Complementing this were ISO 23903, a framework for interoperability and integration; HL7 FHIR, a widely adopted standard for health data exchange; and SNOMED CT, a medical terminology system designed to ensure semantic consistency. By aligning with these standards, the project sought to eliminate redundancy and create a more structured approach to health data management.

A key part of the study involved developing a new health data model to facilitate efficient data reuse. This model (Sect. 3.4) addressed inconsistencies by implementing a standardized terminology system, ensuring that medical data could be harmonized across sources. In parallel, the researchers worked on FHIR profiling, a process aimed at structuring health data to make it machine-readable and interoperable. Recognizing the growing importance of public health monitoring, they also assessed how WHO health indicators could be integrated into the system, allowing for improved data-driven decision-making. To further support interoperability, TermX, an interoperability management tool, was developed

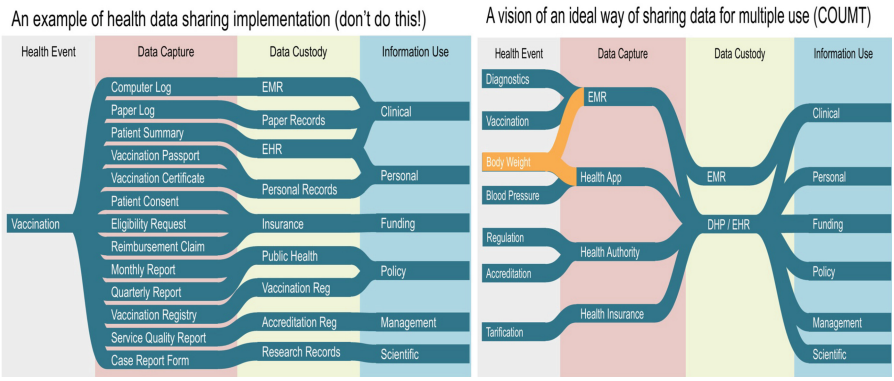


Fig. 5. Data sharing and use [42].

(Sect. 3.5). TermX integrates data dictionaries, glossaries, and terminologies, providing a structured framework for exchanging health information.

Key Findings. One of the main challenges identified during the project was the fragmentation of health data across multiple silos. Current medical records are often stored in isolated systems, making it challenging to ensure contextual traceability, particularly for secondary use. Researchers also found that existing interoperability models required extensive manual input for data harmonization, adding to the complexity of health data management. Without addressing these issues, secondary use—whether for research, policymaking, or health system optimization—would remain limited. To address this, the study a new data management model, with its key components being a fact-based structure, event-based data sharing, and FHIR-based profiling. Adopting a fact-based data structure (currently based on an epicrisis, a physician-prepared summary of a patient's medical history) aims to eliminate redundant data entry, thereby improving data management efficiency. The transition from a document-based approach to event-based data sharing could ensure better continuity of care, as patient data could be dynamically structured and updated in response to new medical events. Additionally, implementing FHIR-based profiling (currently, data is based on HL7 Clinical Document Architecture (CDA)) would enable machine-readable records, thereby opening up potential opportunities for AI-driven analysis and other advanced applications in healthcare.

Interoperability emerged as one of the most pressing issues identified in the study. The findings highlighted that semantic interoperability—the ability of different systems to understand and interpret data consistently—is essential for effectively exchanging health information. Standardizing medical terminology using SNOMED CT and ICD-10 would significantly improve data exchange between institutions and facilitate data reuse. Additionally, the study emphasized the need to implement role-based access control for various stakeholders,

including physicians, policymakers, and researchers, to ensure need-based access to health data in compliance with data protection regulations.

Recommendations and Conclusion. The *Health Sense* project outlined several key recommendations: 1. Adopting ISO 13940 (ContSys) as the national framework to provide a structured approach to health data management. 2. Mandating FHIR profiling for all new health records to ensure long-term interoperability and future-proof health data systems. 3. Developing a national terminology database to unify medical coding systems and reduce inconsistencies in health data exchange. 4. Creating a governance model for interoperability, clearly defining roles related to data ownership, stewardship, and distribution. 5. Enhancing secondary data use regulations to address legal and ethical concerns, ensuring responsible and secure data sharing.

The study concludes that structured, interoperable, and reusable health data will be essential in modernizing Estonia's eHealth infrastructure. By transitioning to an event-based health data model, Estonia can improve efficiency, enhance treatment outcomes and disease prevention, and create new opportunities for public health analysis, medical research, and health innovation to benefit society. The results of this study are based on the Estonian National Health Information System, a centralized national information system. However, they also apply to eMedLab's new SmartEHR architecture, which enables personal health data storage in a DCAS network under individual ownership and sovereign control (Sect. 4).

3.2 Data Security and Privacy

Considering the nature of healthcare data, privacy and security are crucial. In the context of DCAS, healthcare data privacy and security include essential aspects of data protection, proper management, and sharing in a way that respects individuals' rights and regulatory requirements. Secure, decentralized storage, encryption, and access control are key elements of data security. The system is designed to support primary and secondary use cases. Individuals can contribute valuable data to research and public health efforts without compromising their privacy by using de-identification techniques such as anonymizing or pseudonymizing health records stored in a DCAS network.

While decentralized systems offer strong security and privacy guarantees, they must also comply with the relevant data protection regulations, such as the GDPR in the European Union. One of the fundamental requirements under the GDPR is maintaining an appropriate level of security for personal data. This involves implementing measures to ensure data confidentiality, integrity, and availability; applying pseudonymization and anonymization; implementing encryption; taking countermeasures against potential incidents; mitigating risks; and establishing processes for continuous testing and system evaluation [20]. Moreover, the solution must align with the European Health Data Space (EHDS), which defines a set of rules, standards, practices, and infrastructures in healthcare [21]. The EHDS establishes the regulations for using health informa-

tion in primary use cases (such as healthcare delivery and medical treatment) and secondary use cases (including research, public health, and analytics).

An essential requirement is a legal basis for data processing, such as user consent. Under the GDPR, consent must be freely given, specific, informed, and unambiguous [20]. When storing data in a DCAS network, the user does not give their data to a third party for storage. Instead, a third party provides the user with a private and secure DCAS environment (a tool) that allows them to take ownership and full control of their data. The user can then decide with whom to share their data and under what conditions—whether for primary use, such as with a doctor, or secondary use, such as with a researcher from a specific institution. A decentralized approach gives individuals control over their health data. Users can grant or revoke access at any time, ensuring that their data is shared only with trusted parties.

Applying data de-identification methods, such as anonymization and pseudonymization, is one way to enable data sharing while maintaining privacy. According to the GDPR, anonymization refers to the "process of creating anonymous information," ensuring that anonymized information does not include any identifiable natural person or personal data. European data protection laws apply only to personal data, meaning that once data is anonymized, it falls outside the scope of the GDPR. However, it may still be subject to other legal regulations. It is also essential to distinguish between anonymized and pseudonymized data. Unlike anonymized data, pseudonymized information can retain links to the original personal data and the potential for re-identification. Therefore, it is still considered personal data under the GDPR and must be treated accordingly [2, 20].

In the traditional data-sharing approach, effective implementation of de-identification methods requires technical expertise, knowledge of privacy laws, and a comprehensive risk assessment. Some tools simplify data sharing but require domain knowledge and initial data preprocessing, as they are not specifically designed for health data [60]. In the proposed DCAS-based reference architecture for PHRs (Fig. 3) [37], privacy and security are ensured for both primary and secondary use. Individuals retain full ownership and control over their health data for primary use, such as healthcare service provision, by sharing their identified health data with healthcare providers without it needing to be stored in a centralized database, thereby reducing potential security risks and data breaches. For secondary use, individuals create separate datasets using built-in de-identification features. In such a system, it is vital to have trusted data transparency and integrity validation mechanisms to ensure that secondary data users can rely on the originality and quality of the data. The analysis, solution development, and validation of de-identification and data transparency and integrity challenges in DCAS networks are ongoing research topics at eMedLab.

3.3 Data Transparency and Integrity

As digital health innovations reshape healthcare, the supporting infrastructure must guarantee data integrity and patient ownership. Previously, Kask, Piho,

and Ross [33–35] proposed a Hyperledger Fabric-based solution that outlines how blockchain and decentralized storage can be integrated to preserve the integrity of PHRs while addressing scalability, data ownership, and privacy. In the proposed architecture, health data is stored off-chain using decentralized storage, while the blockchain stores metadata and hashes. The main processes, such as patient consent, PHR creation, and retrieval, are implemented using smart contracts. These contracts authenticate and authorize the entities involved, ensuring that only authorized personnel can access or modify health data.

In the upcoming works, we will continue to explore how a blockchain-based architecture, integrated with decentralized storage networks, can ensure the integrity and transparency of health data. By maintaining a secure, immutable ledger of health records, we can ensure that health data remains accurate and accessible across healthcare organizations. This approach aims to enhance patient engagement and foster trust in digital health solutions. Since the proposed innovative approach to health data management emphasizes transparency, it supports the more active utilization of modern health data. For example, telehealth services or patient-reported data can complement traditional doctor-recorded health data.

Moreover, the described solution grants the patient true ownership of their data. Integrating the data using the patient's consent within the health data management system enhances accessibility for further innovations. Data owners can easily access their health data and share it with other providers that use artificial intelligence and/or machine learning to gain insights into their lives while maintaining complete transparency regarding who has processed their health information and how. We are developing and using various real-world use cases from primary and secondary health data domains, where data transparency and integrity are essential.

With the advancement of digital health innovations, maintaining data integrity remains a significant challenge. Blockchain presents a promising solution by enhancing both integrity and transparency. However, a careful focus on scalability and privacy concerns is necessary. The proposed architecture provides a strong framework to tackle these challenges, enabling digital health technologies to enhance patient-centered care while safeguarding health data. Analysis and discussions on the EHDS [21,24,25] will be conducted to promote greater health data interoperability and accessibility across Europe. This architecture facilitates the secure exchange of PHRs among healthcare providers, enabling the seamless integration of digital health solutions into clinical practice.

3.4 Continuity of Care

Continuity of care is defined as a measurement of interconnectivity between separate care process events [63]. As such, it combines the dimensions of a) patient-relational continuity, b) process management continuity, and c) informational continuity [22]. These dimensions involve multiple stakeholders. A common understanding of the core process components must be harmonized between

all stakeholders, from policy makers to software developers, to facilitate process interconnectivity.

Much focus has been placed on solving informational continuity; implementing large-scale digital health platforms (DHPs) has been one of the main outputs of digitalizing healthcare systems in many countries during the last decade [11]. This has led to the great standardization of health data exchange but has only had a weak impact on process management continuity and the data quality needed to support it. The same applies in Estonia, even though Estonia has been considered one of the world's most advanced DHPs [4].

Current digitalization practice redefines the core concepts of the care process separately with each stakeholder for every use case. Furthermore, common interoperability frameworks such as the (Refined eHealth) European Interoperability Framework [52] tend to take interoperability for granted between domains (legal and regulatory constraints, policy and collaboration agreements, etc.) and minimize concerns of cross-sectional dependencies.

Although PHS enables a common information base — a principle from Business Process Redesign — some argue that an information-centric viewpoint is insufficient in healthcare [5] [9], as it links abstract models with the practical realities of healthcare.

We have chosen ISO 13940, a System of Concepts to Support Continuity of Care (ContSys), as the basis for our realism-based ontology. ContSys has been developed to bridge the gap between the informational content and its pragmatic process context [29]. ContSys systematically relates to eight domains of health and care: actors and resources, activities, processes, time, planning, responsibility, and information. Therefore, it covers the questions of who, why, when, what, where, and how.

The concept system of ContSys is important as it underpins other ISO health informatics standards; the most relevant for the EHDS is the International Patient Summary (ISO 27269). The concepts have been aligned with an upper ontology [12] and several exchange standards such as FHIR [13]. The HL7 and ISO are cooperating to harmonize alignment (ISO/DPAS 24305) further.

We have translated the standard into Estonian [31], and it was released as a national standard in 2023. Work has also been done to implement ContSys within archetype-based development of domains, requirements, and software [56].

Our current work focuses on aligning ContSys with Estonian interoperability domains. Starting with the legal and regulatory domain, a clear understanding of the responsibilities and their role distribution is needed to model the collaborative perspective of care processes for continuity of care. In Estonia, these responsibilities are defined in a healthcare service provision contract described in the Law of Obligations Act. The law is fundamental for Estonian healthcare data, as it, among other obligations, obliges the healthcare provider to document the health-related data discovered during service provision. We have described a method for the semantic annotation of legal texts based on the ContSys concept system [32]. Our results show very high concept coverage and coherence, indicating that ContSys can integrate legal domain knowledge.

Our work will continue to align with more domains and find methods for continuously integrating domains as they evolve. We see the integration of realism-based ontology and the harmonization of interoperability domains underpinned by ContSys as a way to reach higher levels of conceptual interoperability. In upcoming work, we interpret process interconnectivity (i.e., continuity of care) as dynamic system composability in the Levels of Conceptual Interoperability Model (LCIM). The LCIM model explains how well systems can work together. It defines seven levels: *Level 0:* No Interoperability: Completely separate systems that cannot work together at all. *Level 1:* Technical Interoperability: Systems can exchange data, but they cannot read each other's messages. *Level 2:* Syntactic Interoperability: Systems know the data format or language to read exchanged messages but not their entailed meanings. *Level 3:* Semantic Interoperability: Systems understand what the data means by knowing their shared meaning (e.g., using a shared terminology server for definitions). *Level 4:* Pragmatic Interoperability: Systems understand what the data means and how it is used in the context of shared use cases. *Level 5:* Dynamic Interoperability: Systems can adjust to new information and change new use case conditions. *Level 6:* Conceptual Interoperability: Systems fully share goals, processes, and meanings, allowing them to work together as though they were designed as one.

We interpret LCIM such that levels 1–4 can be and have already been partially achieved (see Sect. 3.5). However, the methods used do not scale to reach levels 5–6 effectively. This is because methods bind information semantics to static use case-based contexts. Context harmonization (often called standardization) does not scale in several aspects. The number of stakeholders that can be involved is limited, which, in turn, limits the size of the considered context. The result of context standardization is static to the chosen use cases. The time taken to (re-)standardize contexts to changed conditions and new use cases is too long (our estimation is three to five years).

Our results so far show that our realism-based ontology underpinned by the ContSys concept system is suitable for standardization in the conceptual interoperability (level 6) of shared goals and processes. This would allow dynamic interoperability (level 5) to adjust to the continuous harmonization of new information and changing conditions.

3.5 Data Interoperability and Transformation

PHRs have the potential to significantly improve the quality of healthcare outcomes, making them an essential tool for coordinated care [36]. They aim to enhance health outcomes by facilitating the delivery of healthcare services from multiple providers, ensuring that care is not delivered in silos [39]. In healthcare informatics, there is a common need for data exchange between various healthcare enterprises to better manage the quality and delivery of healthcare services [61]. However, different systems often use varied formats and standards, making seamless data sharing difficult. Interoperability refers to the ability of different systems, devices, applications, or healthcare professionals to work together

within and across organizational boundaries, increasing the quality and continuity of care through shared knowledge and enabling the more efficient use of that information in the healthcare process [52]. Interoperability has been identified as one of the greatest challenges in healthcare informatics [52].

There are many data processors for the secondary use of health data. The list of Estonian organizations includes, but is not limited to, the Estonian Health Insurance Fund, the Estonian Health Development Institute, the Estonian Health and Welfare Information Center (TEHIK), etc. (Figure 1). Everyone collects, interprets, and uses data uniquely (Fig. 5). Every data processor often requires this data directly from EMR informational systems.

Many countries have specialized organizations or departments for standardizing healthcare solutions and enhancing semantic interoperability [15,44,50]. Their tasks include: 1) developing terminology; 2) developing logical data models to represent clinical information; 3) adapting interoperability standards such as FHIR [3] or HL7 CDA [40]; 4) designing data transformation between logical data models and interoperability standards; 5) creating a knowledge base and thesaurus; and 6) publishing standards and implementation guidelines.

Interoperability relies on formal standards and specifications. Organizations such as Health Level Seven International (HL7), openEHR International, and the Personal Connected Health Alliance (PCHAlliance) are creating open standards and specifications to facilitate the flow of data into PHRs [36]. Achieving a consensus on system requirements and usage rules is also crucial. The Integrated Healthcare Enterprise (IHE) promotes the coordinated use of established standards, such as HL7 and DICOM, to address specific clinical needs and support optimal patient care [26]. Interoperability standards can be used with different terminologies and classifiers, such as the Systematized Nomenclature of Medicine Clinical Terms (SNOMED CT), the Logical Observation Identifiers Names and Codes (LOINC), the International Classification of Diseases (ICD), and others [3].

The interoperability specialists have to support many standards at the same time, including internal data formats and/or widely accepted interoperability standards, such as HL7 V2, HL7 V3 and CDA, HL7 FHIR, DICOM, OMOP, and openEHR, as well as their various versions. Different standards approach the design of data models with different philosophies. For example, the HL7 Reference Implementation Model (RIM) used within HL7 V3 aims to encompass the full spectrum of possible healthcare scenarios [16]. In contrast, HL7 FHIR provides a common model. Still, instead of constraining the scope to attempt to define a global model for all aspects of healthcare, it follows the 80/20 principle [23], designing its resources for the most common healthcare scenarios while incorporating an extension mechanism to accommodate attributes that may be absent from the models [38]. Other standards and organizations, national or global, may define data models for specific purposes. For example, OMOP defines a Common Data Model for healthcare data standardization, simplifying data analysis and sharing across platforms and institutions [1]. FHIR has

rapidly become the most important health interoperability standard globally [3]. However, the interoperability problem remains largely unresolved [47].

To guarantee the secure, free, and semantically correct flow of data within the EU, the Refined eHealth European Interoperability Framework (ReEIF) was developed to promote and support the delivery of European public services across EU member state borders [52]. The ReEIF provides advice and guidance on improving the governance of interoperability activities, optimizing processes that support end-to-end digital services, establishing inter-organizational and cross-border relationships, and ensuring that existing and new legislation do not compromise interoperability efforts. The EHDS builds upon the principles and methodologies established by the ReEIF to create a more comprehensive and unified framework for health data exchange across EU member states [21]. Essentially, the EHDS leverages the foundational work of the ReEIF to enhance and expand the scope of health data interoperability, establishing clear rules, common standards, practices, infrastructures, and a governance framework for the use of electronic health data.

Successful implementation of eHealth interoperability, including developing and applying data models and terminology, requires extensive domain knowledge or using appropriate software [8]. User-friendly software can enable non-technical individuals to develop eHealth interoperability specifications, including terminology, data models, and transformations, more easily without needing to understand the full complexity of interoperability standards [6].

There is a lack of high-quality open-source tools that offer proper user interfaces and multilingual support [8,30]. Snowstorm [27] stands as a notable exception, though it is tailored specifically to the management of the SNOMED nomenclature. Additionally, there is a lack of modeling tools [47] and transformation editors [7] for FHIR. Existing terminology servers exhibit several deficiencies [30], and the development of implementation guides is notably complex [45].

The tooling supporting the HL7 FHIR standard remains largely unexplored in academic literature. Several factors contribute to this gap. First, commercial entities are reluctant to disclose proprietary information that provides them with a competitive edge. Second, academic institutions are not typically involved in the development of these tools, resulting in the architecture and algorithms of such tools remaining obscure to the academic community [6].

TermX aims to address these identified gaps and elevate interoperability to a new level. It is designed to be an accessible tool for non-technical personnel, thereby broadening its usability and impact [8]. The essential components of TermX include the *Terminology Server*, the *Model Designer*, the visual *FML Editor*, and a *Publisher* that facilitates structuring terminology, models, and Wiki pages into logical spaces [58]. Its visual interface enhances clarity, facilitates the reuse of terminology, models, and transformation components, and abstracts the complexities of the FHIR framework, the FML mapping language, and interoperability standards, enabling analysts to adapt quickly.

4 Discussions and Future Work

In the coming five years, the eMedLab research group will validate person-owned and controlled PHRs in DCAS networks at the Technology Readiness Level (TRL) 7 in collaboration with other partners. Funded by "Digital health for a whole and healthy society" [18] and "Medication Adherence and Treatment Efficacy in Patients with Dyslipidaemia and Achievement-oriented Novel Patient Digital Support" [19], the research will focus on assessing medical relevance and treatment outcomes, particularly in cardiovascular diseases, men's health, and occupational medicine.

The study will examine the integration of patient-controlled health data within Estonia's Nationwide Health Information System, ePrescription database, and hospital medical records. Patients will have full control over their records, merging clinical data with self-reported information and wearable device inputs. This approach aims to improve treatment adherence, enhance disease prevention, and help reduce diagnostic errors.

The research will also evaluate how two-way patient-doctor communication influences chronic disease management. Data from the North Estonia Medical Centre (NEMC) and clinical partners will be analyzed to assess whether digital engagement improves biometric indicators, treatment adherence, and quality of life. Additionally, the study will explore predictive analytics and AI-driven interventions for personalized treatment recommendations and early risk detection.

A key objective is to measure the impact of patient-controlled health records on public health and healthcare costs. The study aims to demonstrate decentralized health data systems' economic and clinical value by reducing hospital admissions, enhancing preventive care, and supporting long-term treatment adherence.

The findings will support the broader adoption of patient-controlled digital health records and serve as a model for national and European health policy frameworks. Estonia's advanced digital health infrastructure provides an ideal testbed for decentralized health data systems, with results contributing to future healthcare innovations in Europe and beyond.

The EHDS aims to create secure, interoperable, and patient-centric health data exchange across the EU. Integrating patient-owned and fully controlled PHRs in DCAS networks supports this vision by improving data accessibility, ownership, security, and secondary use.

Unlike centralized EHRs stored in institutional silos, patient-controlled PHRs in DCAS networks consolidate health data from multiple sources—hospitals, general practitioners, pharmacies, wearable devices, and research studies—into a single, interoperable record. This enables seamless and secure cross-border healthcare data exchange without complex national interoperability frameworks.

PHRs in DCAS also enhance the secondary use of health data for research, public health, and AI-driven analytics by allowing individuals to manage consent for anonymized data sharing. This approach addresses regulatory barriers and ethical concerns while accelerating biomedical research and AI-based clinical decision support.

Additionally, DCAS mitigates cybersecurity risks by encrypting and distributing health data across a decentralized network, reducing vulnerabilities associated with centralized EHR systems. This aligns with the goal of the EHDS, which is to create a trustworthy and secure health data environment.

Challenges remain, including legal and regulatory compliance with GDPR and EHDS policies, gaining healthcare professionals' trust, and ensuring technical interoperability with existing health systems. The research aims to develop a proof-of-concept app into a commercially viable platform, offering a practical alternative to centralized health data management.

One of the primary future research topics is developing a single underlying model for PHRs in the DCAS network to ensure that all data from various sources is preserved and that data semantics remain unchanged when transforming between formats.

One of the current solutions we are carefully analyzing is the Solid Project. Proposed by Tim Berners-Lee [49,51], this project introduces a protocol based on HTTP for decentralized personal data storage. This approach allows users to store their private data on the Internet in personal data stores called pods. The data within the pods is stored in the Linked Data format (RDF, Turtle). This open, standardized, and interoperable structure enables multiple applications to access and work with the same data seamlessly. Each data element is identified by a unique URL, allowing one pod to reference elements from another, effectively creating an interlinked semantic web.

Solid Pods are significant in addressing the three dilemmas outlined in the introduction when applied to health data [17,53,59]. However, a key challenge with this approach is its dependence on pod providers, over whom users have limited control. A promising area for further research is storing Solid Pods on a DCAS network, replacing the unique URLs that identify data elements with root hashes. This approach could resolve several shortcomings of the current Solid solution, such as reliance on pod providers and ensuring long-term data preservation.

5 Conclusion

This research explores a shift in personal health data management, advocating for individual ownership and sovereign control over health records. It examines the feasibility of decentralized content-addressable storage (DCAS) networks as a foundation for personal health records (PHRs), contrasting them with conventional server-based models, including hospital electronic health records, national health information systems, and commercial health applications.

The study analyzes the key organizational and technical aspects of DCAS-based health data storage, addressing data custody, security, privacy, quality, transparency, integrity, anonymization, pseudonymization, and interoperability. These aspects are explored in ongoing research at eMedLab, Tallinn University of Technology (TalTech), which investigates the challenges and opportunities of implementing person-owned health data systems.

In addition, the research proposes preliminary organizational, economic, and technical scenarios for integrating DCAS-based PHRs with existing healthcare IT infrastructures, including national health registries and commercial applications. These strategies are considered within the European Health Data Space (EHDS) framework, evaluating both primary uses, such as healthcare services and medical treatment, and secondary uses, including medical research, public health, and AI-driven analytics.

This study contributes to the broader discussion on secure, interoperable, and ethically responsible health data management by outlining a potential transition toward individual ownership and full control of health data.

Our work on a decentralized content-addressable storage (DCAS) network for privacy-preserving health data sharing remains at an early proof-of-concept stage, and the reviewers have identified several challenges for future investigation. They have highlighted the 'unlearning' problem in AI models when a data hash is revoked; the gap between simply logging who fetched data and the need to audit how that data is processed; the difficulty of guaranteeing reliable data deletion or access suspension in a peer-to-peer environment; and the tension between enabling ethical secondary use of depersonalized patient data and maintaining strong protections against re-identification and threats to data integrity. Moreover, practical key-management questions—such as ensuring that a clinician truly "forgets" a patient's root-hash key after a consultation, protocols for irrevocable data deletion, procedures for rotating compromised hashes, and guidelines for how individuals should securely store and manage their own root hashes—remain open. Addressing these issues, together with ongoing validation in relevant medical use case studies, will guide our next steps in developing and evaluating the DCAS framework.

References

1. Bathelt, F.: The usage of ohdsi omop–a scoping review. Proceedings of the German Medical Data Sciences (GMDS) pp. 95–95 (2021)
2. Bende, P., Vovk, O., Caraveo, D., Pechmann, L., Leucker, M.: A case study on data protection for a cloud-and ai-based homecare medical device. In: Transactions on Petri Nets and Other Models of Concurrency XVII, pp. 138–161. Springer (2023)
3. Benson, T., Grieve, G.: Principles of Health Interoperability. Springer (2021)
4. Bertl, M., Kankainen, K.J.I., Piho, G., Draheim, D., Ross, P.: Evaluation of data quality in the estonian national health information system for digital decision support. In: MacCaull, W., Pun, V.K.I. (eds.) Proceedings of the 3rd International Health Data Workshop (HEDA 2023). CEUR Workshop Proceedings, vol. 3440, pp. 4–17. CEUR, Leicester, United Kingdom, July 2023, iSSN: 1613-0073, Accessed 07 May 2025
5. Blobel, B., Ruotsalainen, P., Oemig, F.: Why interoperability at data level is not sufficient for enabling pHealth? pHealth 2020, pp. 3–20 (2020), publisher: IOS Press, Accessed 07 May 2025
6. Bossenko, I.: A domain–specific framework for supporting semantic interoperability in primary and secondary use of health data on the example of the Estonian National Health Information System. Ph.D. thesis, Tallinn University of Technology (2025), Accessed 07 May 2025

7. Bossenko, I., Randmaa, R., Piho, G., Ross, P.: Interoperability of health data using FHIR mapping language: transforming HL7 CDA to FHIR with reusable visual components. Frontiers in Digital Health, p. 30 (2024), Accessed 07 May 2025
8. Bossenko, I., Piho, G., Ivanova, M., Ross, P.: TermX: the semantic interoperability, knowledge management and sharing platform. SoftwareX **27** (2024), Accessed 07 May 2025
9. Brochhausen, M., Bost, S.J., Singh, N., Brochhausen, C., Blobel, B.: Understanding the gap between information models and realism-based ontologies using the generic component model. pHealth 2021, pp. 159–164 (2021), publisher: IOS Press, Accessed 07 May 2025
10. Cascini, F., Pantovic, A., Al-Ajlouni, Y.A., Puleo, V., De Maio, L., Ricciardi, W.: Health data sharing attitudes towards primary and secondary use of data: a systematic review. EClinicalMedicine **71** (2024)
11. Colombo, F., Oderkirk, J., Slawomirski, L.: Health information systems, electronic medical records, and big data in global healthcare: progress and challenges in OECD Countries. In: Haring, R., Kickbusch, I., Ganten, D., Moeti, M. (eds.) Handbook of Global Health, pp. 1–31. Springer, Cham (2020), Accessed 07 May 2025
12. Das, S., Hussey, P.: ContSOnto: a formal ontology for continuity of care. Stud. Health Technol. Inf. **285**, 82–87 (2021). https://doi.org/10.3233/SHTI210577
13. Das, S., Hussey, P.: HL7-FHIR-based ContSys formal ontology for enabling continuity of care data interoperability. J. Personalized Med. **13**, 1024 (2023). https://doi.org/10.3390/jpm13071024
14. Demandsage: Data-Driven Solution For Business Growth. webpage (2025), Accessed 07 May 2025
15. Egelkraut, R., Traxler, B., Helm, E., Krauss, O., Schuler, A.: Open infrastructure for standardization of HL7® FHIR® implementation guides in Austria. In: dHealth, pp. 221–223 (2022)
16. Eggebraaten, T.J., Tenner, J.W., Dubbels, J.C.: A health-care data model based on the HL7 reference information model. IBM Syst. J. **46**(1), 5–18 (2007)
17. Esposito, C., Horne, R., Robaldo, L., Buelens, B., Goesaert, E.: Assessing the solid protocol in relation to security and privacy obligations. Information **14**(7), 411 (2023)
18. Estonian Research Council: Digital health for a whole and healthy society (2024–2028), Accessed 07 May 2025
19. Estonian Research Council: Medication Adherence and Treatment Efficacy in Patients with Dyslipidaemia and Achievement-oriented Novel Patient Digital Support (2025–2029), Accessed 07 May 2025
20. EU: General Data Protection Regulation. Official Journal of the European Union **119**(1), 1–88 (2016), Accessed 07 May 2025
21. European Commision: European Health Data Space (2025), Accessed 07 May 2025
22. Haggerty, J.L., Reid, R.J., Freeman, G.K., Starfield, B.H., Adair, C.E., McKendry, R.: Continuity of care: a multidisciplinary review. BMJ **327**(7425), 1219–1221 (2003). https://doi.org/10.1136/bmj.327.7425.1219
23. HL7: FHIR overview - Architects. (2024), Accessed 07 May 2025
24. Hussein, R., et al.: Getting ready for the european health data space (ehds): Iderha's plan to align with the latest ehds requirements for the secondary use of health data. Open Res. Eur. **4**(160), 160 (2024)
25. Hussein, R., Scherdel, L., Nicolet, F., Martin-Sanchez, F.: Towards the european health data space (ehds) ecosystem: a survey research on future health data scenarios. Int. J. Med. Informatics **170**, 104949 (2023)

26. IHE: About IHE (2024), Accessed 07 May 2025
27. IHSTDO: Snowstorm terminology server (2024), Accessed 07 May 2025
28. Information Technology Foundation for Education: IT Academy research support measures programme for 2018-2022: Artificial Intelligence & Machine Learning; Data Science and Big Data; Robots-People collaboration and the Internet of Things in Industry processes. (2018–2023), Accessed 07 May 2025
29. ISO Central Secretary: ISO 13940:2015 - Health informatics — System of concepts to support continuity of care. Standard ISO 13940:2015, International Organization for Standardization, Geneva, CH (2015)
30. Ivanova, M., Bossenko, I., Piho, G.: Comparative analysis of clinical terminology servers: a quest for an improved solution. LNB Inf. Process. **531**, 12 (2024), Accessed 07 May 2025
31. Kankainen, K.J.I., Erm, R., Metsallik, J., Piho, G., Ross, P.: Experiences translating and introducing ISO 13940 ContSys in Estonia: challenges and perspectives. MIE 2025, Accepted paper, May 2025
32. Kankainen, K.J.I., Erm, R., Metsallik, J., Piho, G., Ross, P.: Method for ContSys-based semantic annotation of the Estonian law of obligations. MIE 2025, Accepted paper, May 2025
33. Kask, M., Klementi, T., Piho, G., Ross, P.: Maintaining data integrity in electronic health records with hyperledger fabric. In: HEDA@ STAF (2023)
34. Kask, M., Klementi, T., Piho, G., Ross, P.: Preserving decentralized ehr-s integrity. In: Telehealth Ecosystems in Practice, pp. 296–297. IOS Press (2023)
35. Kask, M., Piho, G., Ross, P.: Distributed ledger-based system to support health data integrity and transparency. In: Digital Health and Informatics Innovations for Sustainable Health Care Systems, pp. 1231–1232. IOS Press (2024)
36. Katehakis, D.G., Kouroubali, A.: A framework for ehealth interoperability management. J. Strateg. Innov. Sustain. **14**(5) (2019)
37. Klementi, T., Piho, G., Ross, P.: A reference architecture for personal health data spaces using decentralized content-addressable storage networks. Front. Med. **11**, 1411013 (2024)
38. Kramer, M.A.: Reducing FHIR "Profiliferation": a data-driven approach. In: AMIA Annual Symposium Proceedings, vol. 2022, p. 634. American Medical Informatics Association (2022)
39. McDonald, K.M., et al.: Definitions of care coordination and related terms. In: Closing the Quality Gap: A Critical Analysis of Quality Improvement Strategies (Vol. 7: Care Coordination), Agency for Healthcare Research and Quality (US) (2007)
40. Mercorella, M., Ciampi, M., Esposito, M., Esposito, A., De Pietro, G.: An architectural model for extracting FHIR resources from CDA documents. In: 2016 12th International Conference on Signal-Image Technology & Internet-Based Systems (SITIS), pp. 597–603. IEEE (2016)
41. Metsallik, J.: Estonian ehealth platform. Unpublished slides (2021), Presented at RSU Research week 2021
42. Metsallik, J., Draheim, D., Sabic, Z., Novak, T., Ross, P.: Assessing opportunities and barriers to improving the secondary use of health care data at the national level: Multicase study in the kingdom of saudi arabia and estonia. J. Med. Internet Res. **26**, e53369 (2024)
43. Metsallik, J., Ross, P., Draheim, D., Piho, G.: Ten years of the e-health system in Estonia. In: Rutle, A., Lamo, Y., MacCaull, W., Iovino, L. (eds.) CEUR Workshop Proceedings. vol. 2336, pp. 6–15. 3rd International Workshop on (Meta) Modelling for Healthcare Systems (MMHS) (2018), Accessed 07 May 2025

44. Pirnejad, H., Bal, R., Stoop, A., Berg, M.: Inter-organisational communication networks in healthcare: centralised versus decentralised approaches. Int. J. Integr. Care (2007)
45. Randmaa, R., Bossenko, I., Klementi, T., Piho, G., Ross, P.: Evaluating business meta-models for semantic interoperability with FHIR resources. In: HEDA@ Petri Nets (2022)
46. Roehrs, A., Da Costa, C.A., da Rosa Righi, R., De Oliveira, K.S.F., et al.: Personal health records: a systematic literature review. J. Med. Internet Res. **19**(1), e5876 (2017)
47. Ross, P., Metsallik, J., Kankainen, K.J.I., Bossenko, I., Mäe, C., Maasik, M.: Health Sense: development of a universal data model and a standard for continuity of treatment paths based on international standards of new generation health information systems. TalTech Digikogu (2023), english translation, Accessed 07 May 2025
48. Safran, C., et al.: Toward a national framework for the secondary use of health data: an American medical informatics association white paper. J. Am. Med. Inf. Assoc. **14**(1), 1–9 (2007), Accessed 07 May 2025
49. Sambra, A.V., et al.: Solid: a platform for decentralized social applications based on linked data. MIT CSAIL & Qatar Computing Research Institute, Technical Report **2016** (2016)
50. Schliemann, T., et al.: eHealth standardisation in the nordic countries:: technical and partially semantics standardisation as a strategic means for realising national policies in eHealth. Nordic Council of Ministers (2019)
51. Solid: Your data, your choice. Advancing Web standards to empower people. webpage (2024), Accessed 07 May 2025
52. Sprenger, M., van Pelt, V., Seven, M.: Refinement of the eHealth European Interoperability Framework (ReEIF) (2015), Accessed 07 May 2025
53. Sun, C., Gallofré Ocaña, M., van Soest, J., Dumontier, M.: citizen-centric data platform (tidal): sharing distributed personal data in a privacy-preserving manner for health research. Semantic Web **14**(5), 977–996 (2023)
54. Swarm: Swarm is a decentralised storage and communication system for a sovereign digital society. webpage (2022), Accessed 07 May 2025
55. Synthea: Synthea is a Synthetic Patient Population Simulator. webpage (2023), Accessed 07 May 2025
56. Sõerd, T., Kankainen, K., Piho, G., Klementi, T., Ross, P.: Towards specification of medical processes according to international standards and semantic interoperability needs. In: Proceedings of the 11th International Conference on Model-Based Software and Systems Engineering, pp. 160–167. SCITEPRESS - Science and Technology Publications, Lisbon, Portugal (2023), Accessed 07 May 2025
57. Taylor, P.: Amount of data created, consumed, and stored 2010-2023, with forecasts to 2028. webpage (2024), Accessed 07 May 2025
58. TermX: TermX homepage (2024), Accessed 07 May 2025
59. Theys, T., Mechant, P., Maes, M., Bourgeus, A., Saldien, J., De Marez, L.: Solid pods: a promising approach to enhance users' perception of data transparency and control. Interact. Comput. iwaf017 (2025)
60. Vovk, O., Piho, G., Ross, P.: Evaluation of anonymization tools for health data. In: International Conference on Model and Data Engineering, pp. 302–313. Springer (2021)

61. Weber-Jahnke, J., Peyton, L., Topaloglou, T.: eHealth system interoperability. Inf. Syst. Front. **14**, 1–3 (2012)
62. Wei, J., et al.: Emergent Abilities of Large Language Models (2022), Accessed 07 May 2025
63. World Health Organization: Continuity and coordination of care: a practice brief to support implementation of the WHO Framework on integrated people-centred health services. World Health Organization (2018), Accessed 07 May 2025

LC/NC Pipeline for Training and Operationalising Segmentation Models in a Data Scarce Domain: De-arraying Tissue MicroArrays

Colm Brandon[1,2(✉)], Éanna Fennell[1,4], Amandeep Singh[1,2], and Tiziana Margaria[1,2,3,5]

[1] University of Limerick, Limerick, Ireland
colm.brandon@ul.ie
[2] Centre for Research Training in Artificial Intelligence (CRT-AI), Limerick, Ireland
[3] Lero, Limerick, Ireland
[4] Bernal Institute and School of Medicine, University of Limerick, Limerick, Ireland
[5] The Health Research Institute (HRI), University of Limerick, Limerick, Ireland

Abstract. Here we present a new approach to training and operationalizing segmentation models for de-arraying Tissue Micro Arrays (TMAs). The scarcity of large, high-quality datasets in sensitive domains such as human tissue samples, coupled with strict privacy regulations to protect doner interests, poses significant obstacles to training robust and generalised segmentation models. To address these challenges, we introduce a new Low-Code/No-Code (LCNC) Domain-Specific Language (DSL) integrated into the Cinco de Bio (CdB) platform. The DSL consists of multiple Service-Independent Building Blocks (SIBs), each providing a distinct functionality essential to creating a pipeline. LCNC enables biologists to train and deploy de-arraying models without writing code. Our methodology incorporates a domain-specific data augmentation technique that generates pseudo-synthetic samples from a minimal set of real data. It also leverages AutoML techniques, including Neural Architecture Search (NAS) and hyperparameter optimisation, to automate the model development process. Furthermore, we present an architectural update to the Cinco de Bio platform, adopting a "Model as Data" paradigm that treats neural network models as dynamic, versioned data assets that can be used as inputs to SIBs. This work provides a practical solution to the challenges of distribution shift and data scarcity in sensitive health domains, where building sufficiently sized datasets to train generalise robust models is infeasible. The proposed LCNC DSL and accompanying pipeline enables domain experts to effectively leverage Artificial Intelligence (AI) technologies and tailor them to their own data.

Keywords: Model Driven Development · Health Informatics · Artificial Intelligence · Low-Code/No-Code · AutoML

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 104–121, 2026.
https://doi.org/10.1007/978-3-032-01377-4_5

1 Introduction

TMAs have emerged as a powerful high-throughput research tool in pathology and oncology. They enable simultaneous analysis of multiple tissue samples arranged systematically on a single microscope slide. Each TMA is comprised of numerous small cylindrical tissue cores, typically extracted from distinct donor blocks and arranged in a grid-like pattern within paraffin blocks. This technique facilitates rapid assessment of molecular markers, such as proteins or genes, across multiple tissue samples, significantly enhancing the efficiency of pathological analyses [44].

A critical preprocessing step in the analysis pipeline for TMAs is the *de-arraying procedure*: it segments the entire TMA slide into individual tissue cores. Each core typically originates from a distinct donor and thus must be analysed independently to ensure accurate downstream interpretation. Although idealised TMAs are arranged in perfect grids with clear separations between cores, real-world samples frequently deviate from this ideal due to variations in experimental protocols across laboratories, as well as physical alterations during slide preparation and processing [44]. Consequently, the automated segmentation of individual tissue cores commonly referred to as de-arraying presents substantial challenges for image processing and Machine Learning (ML) algorithms.

The successful application of ML models to automate TMA de-arraying tasks is hindered by several inherent challenges prevalent in sensitive health domains. The first among these are *data scarcity* and *distribution shift*. The scarcity of large-scale annotated datasets arises primarily due to the expense of acquiring samples and privacy regulations (such as GDPR) that restrict data sharing among research institutions [12,34] in order to protect donor rights. Distribution shift is the phenomenon where statistical properties differ between training dataset and the data the model is subsequently used to make predictions on. This poses substantial challenges to model generalisation and reliable inference [34]. Ensuring robust classification under distribution shift remains an open challenge within current machine learning research, especially in cases when create a sufficiently sized dataset is not possible.

The CdB application development platform [4] was initially developed to enable domain researchers, here specifically bio-imaging experts, to leverage existing ML methods by integrating already trained models into their workflows. However, it became evident that for a variety of tasks such as TMA de-arraying, merely enabling the use of pre-trained models is insufficient. To achieve reliable outcomes under varying experimental conditions across different laboratories, domain researchers require the capability to train ML models directly on their own datasets. This necessitates extending the original scope of CdB towards supporting comprehensive model training workflows. To address these challenges, this paper presents a new approach leveraging the CdB platform—a web-based collaborative environment designed for LCNC development of domain-specific workflows [4]. Specifically, we extend the current core of CdB's modelling language, the Programming Language Domain Specific Language (PL-DSL), to support the use of framework-independent ML formats. The CellMAPs [8]

Application Domain Specific Language (A-DSL) previously integrated within CdB was also extended facilitate both training and operationalising TMA de-arraying models. We extend this A-DSL by adding a new palette of SIBs[1], each encapsulating distinct functionalities required to construct end-to-end training pipelines without necessitating coding expertise from biologists or other domain researchers.

Central to the training pipeline is the integration of a previously developed domain-specific *data augmentation strategy* that generates pseudo-synthetic training samples derived from limited real-world data. Additionally, our pipeline incorporates AutoML techniques, namely Hyperparameter (HP) and HP optimisation, in order to enable domain researchers to participate in the model development process and train effective models when their data deviates from the data distributions on which the publicly available models are trained. We also propose an architectural enhancement to CdB through the adoption of a *Model as Data* paradigm. Under this approach, trained neural network models are treated as dynamic versioned data assets that can serve as inputs for SIBs. This architectural update includes scalable infrastructure improvements for storage and retrieval of trained models and the compatibility with widely-used framework-independent formats such as ONNX and Torchscript.

Through this integrated approach, that combines LCNC, domain-specific synthetic data augmentation techniques, and AutoML methodologies, we demonstrate the feasibility of developing robust segmentation models despite limited training data and limited or no programming and ML expertise. Our contributions provide practical solutions to the prevalent challenges posed by distribution shift and data scarcity when using ML within sensitive health domains, such as pathology research involving human tissue samples.

The remainder of the paper is organised as follows: Sect. 2 discusses the background and related work. Sect. 3 outlines our proposed solution. Finally, Sect. 4 and Sect. 5 offer some discussion, conclusion and the next steps.

2 Background

To appreciate the contribution and the case study, we need to recall a few concepts and state of the art developments in the areas of Medical Image Segmentation and De-arraying (Sect. 2.1), Language-Driven Engineering (LDE) and Model-Driven Development (MDD) (Sect. 2.2, AutoML and Hyperparameter Optimisation (Sect. 2.3, Cinco de Bio (Sect. 2.4) and Framework Independent ML Formats (Sect. 2.5).

2.1 Medical Image Segmentation and De-Arraying

Medical image segmentation is a critical process in healthcare that involves partitioning medical images into multiple segments or regions of interest, each rep-

[1] SIBs) [11] are DSL-level representations and executable encapsulations of domain-specific components and functionalities of the target application domain.

resenting specific anatomical structures, tissues, or pathological areas [37]. This technique is essential for accurate diagnosis, treatment planning, and disease monitoring in various medical imaging modalities such as CT, MRI, X-ray, and ultrasound [48]. The segmentation process is performed using a range of methods, from traditional approaches like thresholding and edge detection to advanced deep learning techniques such as convolutional neural networks [35]. In recent years, the development of foundation models like MedSAM has shown promise in providing universal medical image segmentation capabilities across diverse tasks and modalities [21]. These models are trained on large-scale datasets comprising millions of image-mask pairs, potentially enabling them to generalise well to new segmentation tasks. In the context of tissue microarrays (TMAs), a related process called de-arraying is crucial for analyzing multiple tissue samples efficiently. De-arraying involves segmenting and localizing individual tissue cores within a TMA image and mapping them to their corresponding array coordinates. This process faces challenges such as grid deformation, core misalignments, and artifact imperfections or damage due to assembly errors. Advanced de-arraying methods, such as those employing wavelet-based detection, active contour segmentation, thin-plate spline interpolation [32,44] and machine learning [36], have been developed to address these issues and improve the accuracy of TMA analysis. The integration of robust segmentation and de-arraying techniques is vital for advancing medical image analysis, enabling more precise quantification of biomarkers, and ultimately contributing to improved patient care and research outcomes in fields like oncology and histopathology.

2.2 LDE and MDD

The concept of *semantic gap* in software engineering [31] refers to the discrepancies in interpretations and understandings among stakeholders with diverse mindsets [6,10]. It is the same phenomenon addressed by the business-IT mindset gap [22,25]. To address this challenge, Language-Driven Engineering (LDE) has emerged as a promising solution [3,39]. LDE, an evolution of eXtreme Model-Driven Design (XMDD) [27,28], builds upon the principles of MDD to facilitate rapid and cost-effective application design in a domain-specific fashion.

LDE specifically targets domain experts and subject matter experts who may lack programming proficiency. Its primary objective is to provide tailored modeling languages that align with the needs and capabilities of these stakeholders [51], thereby leveraging and supporting their domain-specifoc modeling experience (cf. [30]). Conceptually, LDE aligns with the One Thing Approach (OTA) [26], advocating for a global, consistent model at the core of development.

The LDE approach supports model design by offering visual interfaces to graph-based modelling languages tailored to the users' professional mindsets, enabling them to design the "logic" of workflow and processes by composing purpose-specific components using drag-and-drop functionality. This method allows users to focus on *what* the application should do rather than requiring knowledge of *how* to implement it with code. Developers model systems using graphical abstractions of compoenents (the SIBs) designing the control-flow and

data-flow (the Service Logic graphs (SLGs), which are then transformed into executable code through automatic model-to-code generators [29].

The Java Application Building Center (jABC) framework [40] was an early example of the LDE approach. It based the design and development of applications on Lightweight Process Coordination (LPC) and formal models. jABC accelerates the development cycle by allowing subject matter experts to directly contribute by modeling the workflows [23,24]. It leverages reusable components called SIBs, which represent executable services, orchestrated into analyzable control structures known as Service Logic Graphs (SLGs).

An early application of these principles in the health informatics domain is Bio-jETI [17,18], built on the Java Electronic Tool Integration (Java Electronic Tool Integration Platform (jETI)) service integration technology. Bio-jETI enabled biology and bioinformatics experts to develop complex executable workflows by graphically combining bioinformatics services [16,20], taking care through varios analyses of concerns about interface details and data type inconsistencies, and adopting a generative approach to writing code [19].

The success of this approach has led to its adoption in various domains and the modeling and execution of scientific workflows in general [13–15]. By enabling domain experts to directly contribute to application development, LDE has demonstrated its potential to bridge the semantic gap and streamline the software development process across diverse fields.

2.3 AutoML and Hyperparameter Optimisation

Hyperparameters such as learning rate (which controls the adjustment step in gradient descent), influence how a model performs and they depend on the dataset and the architecture. Hyperparameter optimisation is a critical process in ML that significantly influences model performance and generalization capabilities [46]. It involves the systematic search for the optimal set of hyperparameters, which are configuration variables set prior to the training process, distinct from the model parameters, which are learned from data. The hyperparameters control the learning algorithm or the structure of the underlying model, playing a crucial role in managing the bias-variance trade-off. The importance of this process stems from its ability to minimize the loss function, thereby enhancing the model's accuracy, efficiency and ability to generalise.

Common techniques for hyperparameter optimization include grid search, random search [1], Bayesian optimization, gradient-based optimization, and evolutionary algorithms [43]. All of them broadly fall under the umbrella of what has become known as AutoML [9] which aims to automate the entire ML workflow. Each method has its strengths and weaknesses, and the choice often depends on the specific problem, computational resources, and the nature of the hyperparameter space. However, hyperparameter optimization faces several challenges, including the curse of dimensionality, the risk of overfitting to the validation set, and the need for substantial computational resources. Given these challenges, in order to explore the hyperparameter search space as efficiently as possible, heuristic search methods (Bayesian and evolutionary approaches) have become

preferable over brute-force methods such a grid search. Despite certain challenges, hyperparameter optimisation remains an indispensable component of the ML pipeline, directly influencing model efficacy and efficiency. AutoML is thus proving to be a promising avenue for enabling ML experts and non-experts alike to train more effective ML models.

As our work focuses on convolutional neural networks, we are interested in a specialised subset of hyper parameter optimisation for NNs called NAS.

Neural Architecture Search
NAS is an emerging ML field that aims to automate the process of designing neural network architectures, a task traditionally exclusively performed by human experts. NAS can be conceptualized as a three-dimensional problem, encompassing the search space, search strategy, and performance estimation strategy [7]. The search space defines the range of possible architectures, often incorporating prior knowledge to reduce complexity while potentially limiting novel discoveries. Search strategies explore this space, balancing between quick discovery of well-performing architectures and avoiding premature convergence to suboptimal solutions. These strategies include methods such as reinforcement learning [50], evolutionary algorithms [42], Bayesian optimization [45,49], and gradient-based approaches [7]. Performance estimation is crucial yet challenging, as it involves evaluating the efficacy of candidate architectures, often requiring significant computational resources to train a candidate architecture for a certain number of iterations and evaluating its performance on a validation data set. Strategies include early stopping and pruning of non-promising architecture candidates, performing NAS on subsets of the full training data as a proxy for the actual performance, among others [7]. Incorporating prior knowledge on neural architectures to further constrain the search space can reduce this computational overhead. Recent research has focused on developing more efficient NAS methods, including one-shot approaches and differential architecture search [47]. NAS has demonstrated remarkable success, with automatically designed architectures outperforming manually crafted ones in tasks such as image classification, object detection, and semantic segmentation [2]. As NAS continues to evolve, it promises to play an increasingly important role in the automated design of efficient and high-performing neural networks across various applications.

2.4 Cinco de Bio

CdB is a low-code application development platform for the development of domain-specific workflows in biomedical research [4]. It addresses the challenges faced by researchers in handling complex computational analyses of experimental outputs such as highly-plexed tissue images (a microscopy-based image that simultaneously captures multiple (typically 20–60) molecular markers at subcellular resolution within a single tissue sample). CdB adopts a model-driven approach, offering a graphical DSL that allows users to design workflows without extensive programming knowledge. The platform incorporates both control and data flow modeling, enabling researchers to create sophisticated analysis pipelines. CdB's architecture includes an Integrated Modeling Environment

(IME) for workflow design, a runtime environment for execution, and a cloud-based deployment option for scalability. A key feature of CdB is its support for *semantic typing*, which captures domain-specific concepts in the data model through a semantic refinement of syntactic data types, this way enhancing understandability for domain experts and reducing errors at design time. CdB distinguishes itself from other workflow management tools by providing extensive support for interactive services within workflows. This feature is particularly valuable in AI-driven analysis, where human intervention is often required at runtime for cases where ML models output is not correct. The platform also offers comprehensive design validation capailities, performing both syntactic and semantic checks on user-defined workflows before execution. By combining these features, CdB aims to bridge the *semantic gap* between domain experts and AI,ML and other computational tools, potentially accelerating conducting research in fields such as digital pathology and spatial proteomics.

2.5 Framework Independent ML Formats

Framework independent ML formats, also referred to in the literature as inference engines or simply export formats, are runtime-less general formats for *serialising* both the model definition and the parameters, and *executing* machine learning models. They essentially decouple the deployment of a model for inference from the chosen training framework (such as PyTorch, Tensorflow , etc.). TorchScript [41] and ONNX (Open Neural Network eXchange) [5] are prominent examples of such formats, each offering unique advantages for model deployment and inference. TorchScript, developed by PyTorch creators, is a language-agnostic format that allows for the creation of serialisable (from PyTorch) models that can be loaded and executed in non-Python environments. It provides a subset of Python that preserves dynamic control flow and supports inputs of different sizes, making it suitable for models with complex logic. ONNX, on the other hand, is a framework-independent format designed to enhance model interoperability across various machine learning tools and platforms. ONNX models can be exported from a wide range of frameworks, including PyTorch, TensorFlow, and Scikit-Learn. This facilitates deployment across different hardware and software environments. Both formats aim to address the challenges of model portability and performance optimisation. TorchScript offers easy conversion from PyTorch models and maintains compatibility with PyTorch's ecosystem, while ONNX provides broader cross-platform support and integration with various hardware accelerators. Recent benchmarks suggest that ONNX Runtime often outperforms TorchScript in inference speed, particularly for certain types of models and hardware configurations [33,38]. However, the choice between TorchScript and ONNX often depends on specific use cases, deployment requirements, and the original model's architecture.

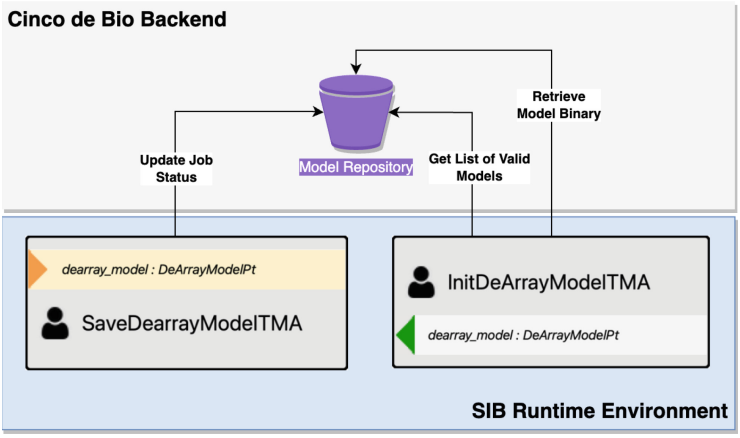


Fig. 1. High-level overview of the neural network model registry implemented in CdB.

3 LCNC Training and Operationalising DeArraying Models

In biomedical image segmentation, particularly for TMAs, a typical ML training workflow involves several key steps that a pipeline should contain.

The process begins with *data preprocessing*, which is crucial for handling the high-resolution, multi-dimensional images associated with TMAs. Each raw image is typically of extremely high resolution (20k x 20k pixels) and consists of multiple pages, each representing a different protein channel. For de-arraying tasks, we focus on the nuclear stain channel, which contains the signal from a fluorescent dye such as 4',6-diamidino-2-phenylindole (DAPI). This channel needs to be extracted from the multi-page TIFF file and then spatially downsampled to a practical size that balances resolution with computational efficiency.

Annotation is a critical step in which images are labelled to provide ground truth for model training. In the case of TMA de-arraying, this involves drawing segmentation masks to denote each tissue core. However, obtaining a sufficiently large dataset in this domain is challenging due to the expense of data collection and patient privacy concerns, so data augmentation techniques need to be employed to artificially inflate the dataset by generating pseudo-synthetic samples.

Following data preparation, *model development* involves selecting or designing a neural network architecture. Convolutional Neural Networks (CNNs) and U-net architectures are commonly used for biomedical image segmentation tasks. Once the data is ready, the next phase is training the model. Neural network performance is highly sensitive to hyperparameter configurations, which vary by dataset. Therefore, hyperparameter optimisation and NAS are essential steps.

Finally, the trained model is *evaluated* on a test dataset using a variety of metrics such as pixel-to-pixel accuracy, Intersection over Union (IoU), and precision to ensure its performance.

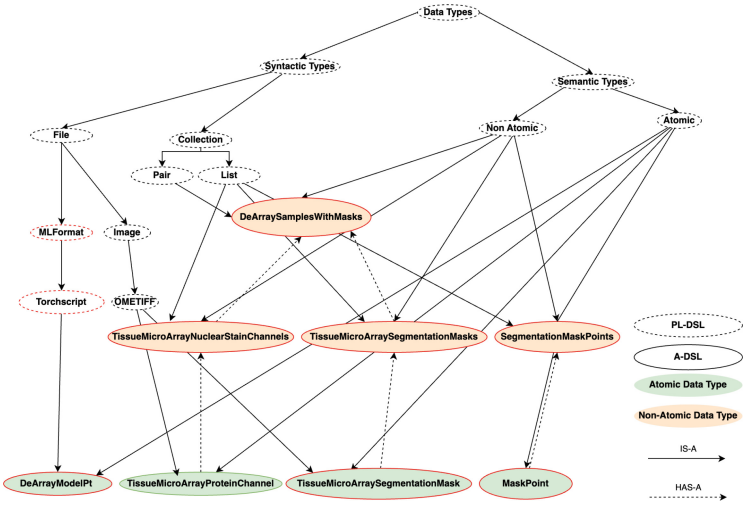


Fig. 2. Excerpt of the semantic model, showing the adjustments to the PL-DSL (CdB) and A-DSL (CellMAPs) data ontologies to facilitate this use case. Nodes with *red* outlines are new additions.

With respect to *operationalising* the model for inference, the data must be preprocessed (extracting the nucleus stain and spatially downsampling it, as with training), the runtime must be able to load and execute an arbitrary neural network architecture, and subsequently perform post-processing to find the Regions of Interest (ROIs) from the predicted segmentation mask.

To achieve the goal of facilitating training and operationalising of TMA dearraying models, the CdB architecture, A-DSL, PL-DSL and integrated SIBs must support every aspect of the training and inference pipeline(s) outlined above.

To facilitate the persistence and sharing of trained neural networks across different workflows and researchers, the CdB architecture was modified to incorporate a ML model repository (see Fig. 1 for an overview). The model repository is implemented with MinIO, the cloud storage service used in CdB. The models are stored in the framework-independent format as versioned assets, accompanied by additional metadata such as the domain-specific process the model is to be used for (e.g., DeArray in this case). In the context of deep learning frameworks like PyTorch or TensorFlow, the model class, which defines the architecture, must be instantiated at runtime for both training and inference. Subsequently, the model parameters are loaded into this instantiated model. Without formats like TorchScript or ONNX, NAS would not be feasible, as the inference runtime would lack the correct model definition.

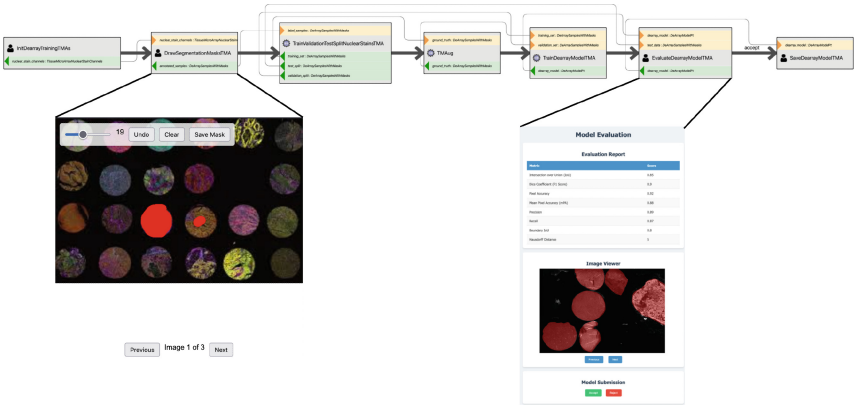


Fig. 3. An annotated dearray training workflow model. Inset Left: the browser-based GUI of *DrawSegmentationMasksTMA* enables the annotation of training samples. Inset Right: the GUI for *EvaluateDearrayModelTMA* presents the user with the performance of the model on the test set and a subset of the test samples with the predicted mask overlayed.

Before integrating the set of SIBs to create the training/inference pipelines (shown in Fig. 3 and Fig. 6), the CdB PL-DSL was extended to enable handling the framework independent ML format files. Similarly, the A-DSL ontology needed to be adjusted to contain the additional semantic types required to support the training of dearraying models and the *models as data* paradigm, which allows neural networks to be used as inputs/outputs to/from SIBs. The adjustments to the PL-DSL and A-DSL are shown in Fig. 2.

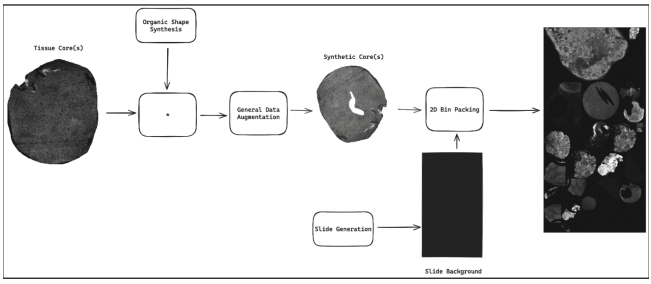


Fig. 4. A brief overview of the data augmentation method implemented in the TMAug SIB, based our previously developed approach (Fennell et al., manuscript in review).

Figure 3 shows the training pipeline implemented in CdB. The InitDearrayTrainingTMAs SIB presents the user with a browser-based GUI enabling to select the set of TMAs and their accompanying nuclear channels. It then performs the first steps of pre-processing by extracting the nuclear channel image from

the multi-page TIFF file and spatially downsamples. The DrawSegmentationMasksTMA SIB enables the annotation of the TMA mask in the browser, following a cursor-based painting approach, the coordinates are then serialised and sent to the SIB backend where the segmentation mask is generated.

As the acquisition of a single TMA image can cost up to €3000, single research laboratories are likely to have an extremely limited amount available, far fewer than required to train a ML model. Therefore, an integral component of the pipeline is the TMAug SIB, which is a domain-specific data augmentation technique we previously developed to generate pseudo-synthetic samples from a limited set of real samples for training dearraying models. An overview of the method is shown in Fig. 4. The integration of this method as a SIB in the CellMAPs DSL effectively addresses the data scarcity issue that domain researchers encounter when training their own model.

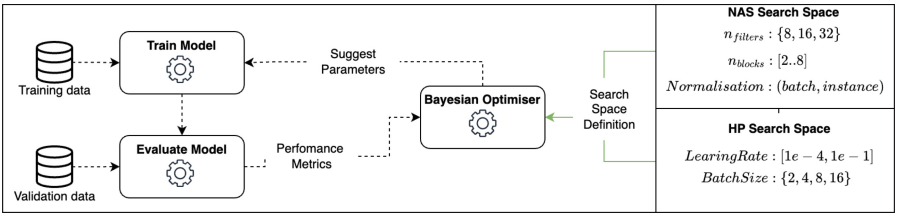


Fig. 5. A high-level overview of the model training process with bayesian HP optimisation and NAS. The shown HP and architecture search space are reasonable defaults, but can be adjusted.

With the data annotated and artificially inflated to a sufficient size, the next step is to find a performant architecture and set of hyperparamaters. A high-level overview of the training regime implemented in TrainDearrayModelTMA is shown in Fig. 5. In the context of this SIB, NAS is constrained to variations of the U-net architecture, which has been shown to perform well in biomedical image segmentation tasks. For HP optimisation, options for parameters such as learning rate, batch size explored. For NAS architectural components such as the type of normalisation applied (batch or instance), the number of filters in the convolution layers (this value is scaled by 2^n, with n being the depth of the block in the encoder, the inverse done for the decoder) and the number of encoder/decoder blocks are explored. As with the original U-net architecture; an encoder block consists of two convolutional layers followed by activation functions, normalisation, and a pooling layer. A decoder block consists of transpose convolution for upsampling, followed by concatenation with corresponding feature maps from encoder blocks (skip connections), then two convolutional layers followed by activation functions, normalisation, and a pooling layer.

HP optimisation and NAS is done on a subset (25%) of the training data for a small number of epochs (5), and can be performed using methods such as grid search or random search. However, in this work, Bayesian optimsation is used

as it can navigate the search space more efficiently than the other methods. The training loop shown in Fig. 5, is repeated a number of times (the default is 100) and after this process has completed, using the best performing architecture and set of hyperparameters, the model is then trained on the full training set for an extended number of epochs (the default is 50), with the model parameters being checkpointed locally after each epoch the current model state achieves a better score on the validation set.

The best performing checkpoint is then passed as input to EvaluateDearrayModelTMA, where it is evaluated using the test set and a report using a variety of segmentation appropriate metrics. This report along with images of segmentation mask overlayed on the TMA are shown to the user in their browser (see *inset right* in Fig. 3), where they can accept or reject the model. If they accept it, it is passed as input to SaveDearrayModelTMA. There they are presented with a browser-based form to input any metadata they wish to accompany the model, and the model is pushed to the model repository to be available for subsequent use. How to operationalise a previously trained model for inference in a subsequent workflow can be seen in Fig. 6. The InitDeArrayModelTMA SIB retrieves the set of ML models in the model repository which match the domain-specific process (in this case dearraying) and presents them and their metadata to the user in a browser-based GUI, where the user can select the one appropriate for them. The SegarrayUserModelTMA, then accepts the model as input along with the TMA and nuclear stain marker. It then uses the model to predict the segmentation mask before applying postprocessing the get the ROIs.

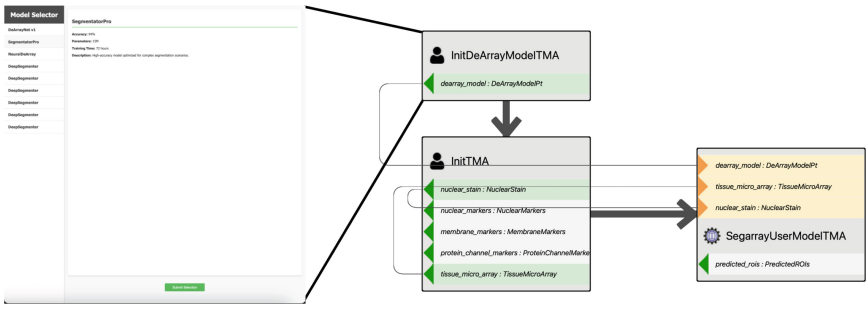


Fig. 6. A simple annotated workflow which illustrates how the user trained dearraying model can be operationalised for inference. *Inset Left*: The browser-based GUI for InitDeArrayModelTMA where the user can select a dearray model from the repository to use in their workflow.

4 Discussion

The development and/or extension of a DSL tailored to domain specific machine learning training workflows, such as the work presented here, present several

interesting design considerations and challenges. A critical aspect in designing such a DSL is maximising static validation capabilities to ensure correctness and robustness at design time, thus reducing potential runtime errors and improving user experience.

A notable challenge arises when handling inter-data dependencies, particularly dependencies that are difficult to enforce statically. For example, a common requirement in machine learning workflows is to ensure that arrays containing related data, such as input samples and corresponding labels, share the same length. Many programming languages lack built-in mechanisms to enforce such constraints during compile-time or design-time validation, leading to potential runtime errors that could have been prevented earlier.

To address this limitation, we propose introducing custom data structures within the DSL to encapsulate data that are interrelated along with explicit assertions or constraints enforcing these interdependencies. So any SIB which returns such data has to enforce these checks, and the type system of the DSL can leverage these constraints for static validation purposes. This approach can significantly improve robustness by preventing invalid argument combinations from being specified at design time, thereby reducing runtime errors and enhancing overall reliability of workflows constructed using the A-DSL.

Another important consideration in designing a semantically expressive DSL is determining the appropriate granularity of semantic typing. For instance, in our current implementation, a single semantic type DeArraySamplesWithMasks is utilised for the training, validation and test datasets. While this simplified approach reduces complexity in type definitions and enhances flexibility for users, it also introduces potential risks: domain researchers may inadvertently misuse datasets (e.g., training a model on test data or validating on training data, etc.), thus compromising the models validity. An alternative solution would involve defining distinct semantic types for each dataset subset (e.g., distinct types for training, validation, and test sets). Such fine-grained typing would enable static verification of correct dataset usage at design time, preventing inadvertent misuse. However, introducing these additional semantic distinctions would increase the complexity within the language definition itself and may impose additional cognitive load on users. Determining whether this increased granularity of semantic typing provides sufficient practical benefits to justify its implementation remains an open research question.

Both aformentioned challenges highlight critical trade-offs between user-friendliness, complexity of language implementation, and robustness guarantees provided by static validation mechanisms. Future work should further investigate these trade-offs empirically, through user studies or case studies involving domain researchers actively employing the extended DSL within realistic scenarios. The insights gained from such investigations would inform best practices for designing expressive yet accessible DSLs tailored specifically towards LCNC ML training workflows in health research domains.

5 Conclusion and Future Work

In this work, we presented our approach to addressing the challenges of training and operationalising ML models for TMA de-arraying, a data-scarce and privacy-sensitive health domain. By extending the CellMAPs Domain-Specific Language A-DSL and integrating it into the Cinco de Bio (CdB) platform, we have enabled domain researchers to effectively train dearraying models based on the U-Net architecture on their own data, thus mitigating issues related to distribution shift and limited annotated data availability. Our approach includes a domain-specific data augmentation strategy to generate pseudo-synthetic samples for a limited number of source samples, AutoML techniques such as Neural Architecture Search (NAS) and hyperparameter (HP) optimisation, and an architectural update adopting a *Model as Data* paradigm to enable ML models to be used as inputs/outputs to/from SIBs.

Several promising directions remain open for future research and development. Firstly, the current framework-independent serialisation formats (e.g., ONNX and Torchscript) employed by the CellMAPs A-DSL and CdB PL-DSL are optimised primarily for inference deployment, lacking support for gradient computation required in fine-tuning scenarios. Extending the DSLs to enable fine-tuning of existing pre-trained models would significantly enhance their utility, allowing researchers to leverage transfer learning techniques effectively. However, achieving this capability would necessitate exploring alternative model representations or integration strategies that enable the computation of gradients.

Secondly, hardware constraints represent another critical consideration for practical deployment. The current implementation of automated HP optimisation and NAS methods does not explicitly account for hardware limitations such as available memory. As certain hyperparameters (e.g., batch size) directly influence the memory footprint of the training process, incorporating hardware-aware constraints into the optimisation search space could prevent out-of-memory (OOM) errors and improve robustness across diverse deployment environments.

An additional potential improvement includes enhancing the TrainDearrayModelTMA SIB with alternative segmentation loss functions (e.g., Focal Loss, Tversky Loss) as configurable parameters to address class imbalance challenges.

Finally, extending the existing CdB data-manager, a front-end service currently supporting experimental data upload, to include comprehensive management capabilities for neural network model data would further streamline workflow integration.

Addressing these future directions will further strengthen the Cinco de Bio platform's capability as a comprehensive low-code/no-code solution for training and deploying robust machine learning models in sensitive health research domains.

References

1. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**(1), 281–305 (2012)

2. Bergstra, J., Yamins, D., Cox, D.: Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In: International Conference on Machine Learning, pp. 115–123. PMLR (2013)
3. Boßelmann, S.: Evolution of Ecosystems for Language-Driven Engineering. Phd thesis, TU Dortmund University (2023). https://doi.org/10.17877/DE290R-23218
4. Brandon, C., et al.: Cinco de bio: a low-code platform for domain-specific workflows for biomedical imaging research. BioMedInformatics **4**(3), 1865–1883 (2024)
5. Community, O.: Open neural network exchange (onnx). https://github.com/onnx/onnx (2025), version: 1.18.0
6. Dorai, C., Venkatesh, S.: Bridging the semantic gap with computational media aesthetics. IEEE Multimedia **10**(2), 15–17 (2003). https://doi.org/10.1109/MMUL.2003.1195157
7. Elsken, T., Metzen, J.H., Hutter, F.: Neural architecture search: a survey. J. Mach. Learn. Res. **20**(55), 1–21 (2019)
8. Éanna Fennell, C.B..: cellmaps: development repository for cellmaps data process services, data and process ontologies and the sdk(s). https://github.com/colm-brandon-ul/cellmaps (2024)
9. He, X., Zhao, K., Chu, X.: Automl: a survey of the state-of-the-art. Knowl.-Based Syst. **212**, 106622 (2021)
10. Hein, A.M.: Identification and bridging of semantic gaps in the context of multi-domain engineering (2010)
11. International Telecommunication union: q-series intelligent network recommendation structure. https://www.itu.int/rec/T-REC-Q.1200-199303-S/en (1993), Accessed 11 Mar 2025
12. Kaissis, G.A., Makowski, M.R., Rückert, D., Braren, R.F.: Secure, privacy-preserving and federated machine learning in medical imaging. Nat. Mach. Intell. **2**(6), 305–311 (2020)
13. Lamprecht, A.-L., Margaria, T.: Scientific workflows: eternal components, changing interfaces, varying compositions. In: Margaria, T., Steffen, B. (eds.) ISoLA 2012. LNCS, vol. 7609, pp. 47–63. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34026-0_5
14. Lamprecht, A.-L., Margaria, T. (eds.): Process Design for Natural Scientists. CCIS, vol. 500. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45006-2
15. Lamprecht, A.L., Margaria, T.: Scientific Workflows with XMDD: a way to use process modeling in computational science education. Procedia Comput. Sci. **51**, 1927–1936 (2015). https://doi.org/10.1016/j.procs.2015.05.457, 15th International Conference On Computational Science (ICCS 2015): Computational Science at the Gates of Nature
16. Lamprecht, A.-L., Margaria, T., Steffen, B.: Seven variations of an alignment workflow - an illustration of agile process design and management in Bio-jETI. In: Măndoiu, I., Sunderraman, R., Zelikovsky, A. (eds.) ISBRA 2008. LNCS, vol. 4983, pp. 445–456. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-79450-9_42
17. Lamprecht, A.L., Margaria, T., Steffen, B.: Supporting process development in Bio-jETI by model checking and synthesis. In: Semantic Web Applications and Tools for Life Sciences (SWAT4LS 2009). CEUR Workshop Proceedings, vol. 435 (2008)
18. Lamprecht, A.L., Margaria, T., Steffen, B.: Bio-jETI: a framework for semantics-based service composition. BMC Bioinf. **10**(Suppl 10), S8 (2009). https://doi.org/10.1186/1471-2105-10-S10-S8

19. Lamprecht, A.L., Margaria, T., Steffen, B.: From Bio-jETI process models to native code. In: 14th IEEE International Conference on Engineering of Complex Computer Systems, ICECCS 2009, Potsdam, Germany, 2–4 June 2009, pp. 95–101. IEEE Computer Society, June 2009
20. Lamprecht, A.L., Margaria, T., Steffen, B.: Bioinformatics: processes and workflows. In: Laplante, P.A. (ed.) Encyclopedia of Software Engineering, chap. 11, pp. 118–130. Taylor & Francis, November 2010. https://doi.org/10.1081/E-ESE-120044612
21. Ma, J., He, Y., Li, F., Han, L., You, C., Wang, B.: Segment anything in medical images. Nat. Commun. **15**(1), 654 (2024)
22. Margaria, T.: Service is in the eyes of the beholder. Computer **40**(11), 33–37 (2007)
23. Margaria, T., Boßelmann, S., Kujath, B.: Simple modeling of executable role-based workflows: an application in the healthcare domain. J. Integr. Des. Process. Sci. **17**(3), 25–45 (2013). https://doi.org/10.1007/978-3-642-34032-1_8
24. Margaria, T., Boßelmann, S., Doedt, M., Floyd, B.D., Steffen, B.: Customer-oriented business process management: visions and obstacles. In: Hinchey, M., Coyle, L. (eds.) Conquering Complexity, pp. 407–429. Springer London (2012). https://doi.org/10.1007/978-1-4471-2297-5_16
25. Margaria, T., Steffen, B.: Service engineering: linking business and it. Computer **39**(10), 45–55 (2006)
26. Margaria, T., Steffen, B.: Business process modelling in the jABC: the one-thing-approach. In: Cardoso, J., van der Aalst, W. (eds.) Handbook of Research on Business Process Modeling. IGI Global (2009)
27. Margaria, T., Steffen, B.: Service-orientation: conquering complexity with XMDD. In: Hinchey, M., Coyle, L. (eds.) Conquering Complexity, pp. 217–236. Springer, London (2012). https://doi.org/10.1007/978-1-4471-2297-5_10
28. Margaria, T., Steffen, B.: Extreme model-driven development (xmdd) technologies as a hands-on approach to software development without coding. In: Encyclopedia of Education and Information Technologies, pp. 732–750 (2020)
29. Mellor, S.J., Balcer, M.J.: Executable UML: A Foundation for Model-Driven Architecture. Addison-Wesley Professional (2002)
30. Mussbacher, G., et al.: The relevance of model-driven engineering thirty years from now. In: Proceeding of the 17th International Conference on Model Driven Engineering Languages and Systems (MODELS 2014), pp. 183–200. No. 8767 in LNCS, Springer (2014). https://doi.org/10.1007/978-3-319-11653-2_12
31. Naur, P., Randell, B. (eds.): Software Engineering: Report of a Conference Sponsored by the NATO Science Committee, Garmisch, Germany, 7–11 October 1968. Scientific Affairs Division, NATO, Brussels 39 Belgium (1969)
32. Nguyen, H.N., Paveau, V., Cauchois, C., Kervrann, C.: Atmad: robust image analysis for automatic tissue microarray de-arraying. BMC Bioinf. **19**, 1–23 (2018)
33. Parida, S.K., Gerostathopoulos, I., Bogner, J.: How do model export formats impact the development of ml-enabled systems? a case study on model integration. arXiv preprint arXiv:2502.00429 (2025)
34. Rieke, N., et al.: The future of digital health with federated learning. NPJ Digit. Med. **3**(1), 1–7 (2020)
35. Ronneberger, O., Fischer, P., Brox, T.: U-net: convolutional networks for biomedical image segmentation. In: Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F. (eds.) MICCAI 2015. LNCS, vol. 9351, pp. 234–241. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24574-4_28
36. Schapiro, D., et al.: Mcmicro: a scalable, modular image-processing pipeline for multiplexed tissue imaging. Nat. Methods **19**(3), 311–315 (2022)

37. Sharma, N., Aggarwal, L.M.: Automated medical image segmentation techniques. J. Med. Phys. **35**(1), 3–14 (2010)
38. Stefani, D., Peroni, S., Turchet, L., et al.: A comparison of deep learning inference engines for embedded real-time audio classification. In: Proceedings of the International Conference on Digital Audio Effects, DAFx, vol. 3, pp. 256–263. MDPI (Multidisciplinary Digital Publishing Institute) (2022)
39. Steffen, B., Gossen, F., Naujokat, S., Margaria, T.: Language-driven engineering: from general-purpose to purpose-specific languages. In: Steffen, B., Woeginger, G. (eds.) Computing and Software Science. LNCS, vol. 10000, pp. 311–344. Springer, Cham (2019). https://doi.org/10.1007/978-3-319-91908-9_17
40. Steffen, B., Margaria, T., Nagel, R., Jörges, S., Kubczak, C.: Model-driven development with the jABC. In: Bin, E., Ziv, A., Ur, S. (eds.) HVC 2006. LNCS, vol. 4383, pp. 92–108. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70889-6_7
41. Team, P.: Torchscript. https://github.com/pytorch/pytorch (2024)
42. Vaidya, G., Ilg, L., Kshirsagar, M., Naredo, E., Ryan, C.: Hyperestimator: evolving computationally efficient cnn models with grammatical evolution. In: ICSBT, pp. 57–68 (2022)
43. Vaidya, G., Kshirsagar, M., Ryan, C.: Grammatical evolution-driven algorithm for efficient and automatic hyperparameter optimisation of neural networks. Algorithms **16**(7), 319 (2023)
44. Wang, Y., et al.: A tma de-arraying method for high throughput biomarker discovery in tissue research. PLoS ONE **6**(10), e26007 (2011)
45. White, C., Neiswanger, W., Savani, Y.: Bananas: Bayesian optimization with neural architectures for neural architecture search. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, pp. 10293–10301 (2021)
46. Yang, L., Shami, A.: On hyperparameter optimization of machine learning algorithms: theory and practice. Neurocomputing **415**, 295–316 (2020)
47. Zela, A., Elsken, T., Saikia, T., Marrakchi, Y., Brox, T., Hutter, F.: Understanding and robustifying differentiable architecture search. arXiv preprint arXiv:1909.09656 (2019)
48. Zhang, Z., Sejdić, E.: Radiological images and machine learning: trends, perspectives, and prospects. Comput. Biol. Med. **108**, 354–370 (2019)
49. Zhou, H., Yang, M., Wang, J., Pan, W.: Bayesnas: a bayesian approach for neural architecture search. In: International Conference on Machine Learning, pp. 7603–7613. PMLR (2019)
50. Zoph, B., Le, Q.V.: Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578 (2016)
51. Zweihoff, P., Tegeler, T., Schürmann, J., Bainczyk, A., Steffen, B.: Aligned, purpose-driven cooperation: the future way of system development. In: Margaria, T., Steffen, B. (eds.) ISoLA 2021. LNCS, vol. 13036, pp. 426–449. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-89159-6_27

Quantum Machine Learning in Precision Medicine and Drug Discovery - A Game Changer for Tailored Treatments?

Markus Bertl[1,2,3(✉)], Alan Mott[1], Salvatore Sinno[1,4], and Bhavika Bhalgamiya[1]

[1] NextGen Computing Research Group, Unisys, Blue Bell, PA, USA
[2] Department of Health Technologies, Tallinn University of Technology, Tallinn, Estonia
[3] Department of Information Systems and Operations Management, Vienna University of Economics and Business, Vienna, Austria
mbertl@taltech.ee
[4] School of Computing, Newcastle University, Newcastle upon Tyne, UK

Abstract. The digitization of healthcare presents numerous challenges, including the complexity of biological systems, vast data generation, and the need for personalized treatment plans. Traditional computational methods often fall short, leading to delayed and sometimes ineffective diagnoses and treatments. Quantum Computing (QC) and Quantum Machine Learning (QML) offer transformative advancements with the potential to revolutionize medicine by leveraging quantum mechanics principles. This paper summarizes areas where QC promises unprecedented computational power, enabling faster, more accurate diagnostics, personalized treatments, and enhanced drug discovery processes. However, integrating quantum technologies into precision medicine also presents challenges, including errors in algorithms and high costs. We show that mathematically-based techniques for specifying, developing, and verifying software (formal methods) can enhance the reliability and correctness of QC. By providing a rigorous mathematical framework, formal methods help to specify, develop, and verify systems with high precision. In genomic data analysis, formal specification languages can precisely (1) define the behavior and properties of quantum algorithms designed to identify genetic markers associated with diseases. Model checking tools can systematically explore all possible states of the algorithm to (2) ensure it behaves correctly under all conditions, while theorem proving techniques provide mathematical (3) proof that the algorithm meets its specified properties, ensuring accuracy and reliability. Additionally, formal optimization techniques can (4) enhance the efficiency and performance of quantum algorithms by reducing resource usage, such as the number of qubits and gate operations. Therefore, we posit that formal methods can significantly contribute to enabling QC to realize its full potential as a game changer in precision medicine.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 122–140, 2026.
https://doi.org/10.1007/978-3-032-01377-4_6

Keywords: Digital Healthcare · Health Informatics · Precision Medicine · Personalised Medicine · Predictive Medicine · Preventive Medicine · Quantum Computing · Quantum Machine Learning · Decision Support System · Formal Methods

1 Introduction

The healthcare digitization faces numerous challenges, including the complexity of biological systems, the vast amount and quality of data generated [7], and the need for personalized treatment plans [21]. Traditional computational methods often struggle to process and analyze data efficiently [11], leading to delays in diagnosis and treatment. Quantum computing (QC) and quantum machine learning (QML) represent transformative advancements with the potential to revolutionize precision medicine [4]. QC promise unprecedented computational power through the use of quantum mechanic principles like superposition, entanglement and quantum tunneling [5]. Based on quantum mechanics, novel algorithms can be developed to tackle complex biological data, leading to more accurate diagnostics, personalized treatments, and enhanced drug discovery processes [16]. However, the integration of QC into precision medicine also presents significant challenges, including technical limitations, high costs, and the need for interdisciplinary collaboration. This paper explores the opportunities and challenges associated with leveraging QC and QML to advance precision medicine, aiming to provide a comprehensive overview of the current landscape and future directions.

The healthcare industry is undergoing a paradigm shift from a one-size-fits-all approach to a more tailored and individualized model of care [8,9]. P5 medicine (predictive, preventive, personalized, participatory and psycho-cognitive medicine) [22] is at the forefront of this transformation, aiming to provide treatments that are specifically designed for individual patients based on their unique genetic, environmental, and lifestyle factors [42]. Traditional medical practices often rely on generalized treatment protocols that do not account for the significant variability among patients. This approach can lead to suboptimal outcomes, as treatments that are effective for one group of patients may be ineffective or even harmful for others [40]. This lack of personalization in treatment plans can result in prolonged illness, increased healthcare costs, and a higher incidence of adverse drug reactions. This is, in large part, because genetic variability plays a crucial role in how individuals respond to medications and treatments [29]. For example, certain genetic mutations can affect drug metabolism, leading to variations in drug efficacy and safety. P5 medicine can leverage genetic information to identify these variations and tailor treatments accordingly. By understanding a patient's genetic makeup, healthcare providers can predict which treatments will be most effective and avoid those that may cause harm. Because of the aging and growing populations, diseases and health care costs rise, making disease prevention even more important [18,32]. Many complex diseases, such as cancer, cardiovascular diseases, and neurological disorders, have multifactorial causes that involve intricate interactions between genes,

environment, and lifestyle. Traditional approaches often fall short in addressing these complexities. P5 medicine, through advanced technologies such as genomic sequencing and methods of bioinformatics, can unravel these interactions and provide insights into the underlying mechanisms of diseases. This knowledge enables the development of targeted therapies that address the root causes of diseases rather than just alleviating symptoms. Personalized medicine also has the potential to advance preventive care. By identifying individuals at high risk for certain diseases based on their genetic and environmental profiles, healthcare providers can implement early interventions to prevent the onset of diseases. This proactive approach can significantly reduce the burden of chronic diseases and improve overall population health.

The ultimate goal of P5 medicine is to improve patient outcomes. By tailoring treatments to the individual characteristics of each patient, healthcare providers can achieve higher treatment success rates, reduce the incidence of adverse effects, and enhance the overall quality of care. Personalized medicine empowers patients by involving them in their own care decisions and providing them with treatments that are specifically designed for their needs. Apart from ethical, legal and social issues, as well as regulatory and policy challenges, data integration and management as well as associated costs are a crucial barrier for the large-scale implementation of P5 medicine. The vast amount of data, such as genomic and phenotypic data, chemical/pharmaceutical data and data from electronic health records (EHR), require advanced computational tools for evaluation which are often associated with high hardware costs.

1.1 Limitations of a Classical Approach to Solving Complex Problems

The limitations of using classical techniques to solve complex problems are not just confined to the limitations of classical computing hardware. Classical mathematical algorithms themselves also have their own limitations. We will briefly visit both these areas in this section.

1.1.1 Limitations of Classical Hardware

The reader may very well be familiar with *Moore's Law*, which states that the number of transistors on a microchip doubles about every two years, though the cost of computing is halved. There is, however, a physical limitation to this. Transistors cannot reduce in size beyond a few atoms. Microchip technology has already progressed to the point where this size limitation may already have been reached and Moore's Law may have already plateaued. Moore's Law is tied to *Dennard Scaling*, which states that as the dimensions of a device go down, so does power consumption. However, this again has physical limits. With smaller and smaller devices, the chances of current leakage across the device becomes ever more likely and increases the chance of thermal runaway (and hence the destruction) of the device through overheating. In addition, classical computer architecture still largely adheres to the *Von Neumann Architecture*,

where the microprocessor and memory are connected via a narrow bus, creating what is known as the *Von Neumann Bottleneck*: As demand for faster computing power to solve ever more complex problems increases, the demand on this bus to exchange data between the microprocessor and its memory exceeds its capability. One approach in resolving this is to use a highly parallelized and distributed architecture where multiple processors (Graphical Processing Units or GPUs) execute commands and process data in parallel. One obvious disadvantage of this is cost. At the time of writing, GPUs are approximately three times the price of CPUs. Moreover, GPUs are designed to perform one particular task at scale (render graphics) and are not optimized for general computing tasks. The high processing demands made on GPUs also increases their thermal power consumption, increasing the system's overall energy costs. In addition, despite their parallel architecture with many Processing Elements (PEs) working in parallel inside the GPU, they still suffer from limitations imposed by memory bus bandwidth issues within the GPU itself. These architectures help, and improve perfomance in certain contexts, namely, where a task can be broken down into subtasks that can be executed in parallel and the completion of one task is not dependent on the completion of another subtask. But they are not a generalised solution to the limitations of the Von Neumann architecture.

The rising demand for faster computing power to solve increasingly complex problems in a reasonable time frame is pushing the capabilities of classical computing hardware to its physical limits.

1.1.2 Limitations of Classical Algorithms

Hardware limitations are not the only issues affecting our abilities to solve problems of ever increasing complexity. Our current methods of solving such problems also suffer from limitations. In other words, the technique (or algorithm) used to solve a problem in and of itself introduces limitations. An example of this is *Time Complexity*. We discuss this here in overview only in order to outline how new algorithms, previously unavailable top us as they rely on having access to quantum states, can provide us with speed increases in finding problem solutions.

Many problems that require a mathematical solution have a defined rate of increase in time needed to solve that problem as that problem's complexity increases. For example, factoring a number is a problem who's complexity, and hence the time needed to solve the problem, increases as the number to be factored gets larger and larger. This issue is known as *Time Complexity*. The rate of increase in the time to solve a problem as its complexity increases is usually categorised as either requiring *polynomial* or *exponential* time. Exponential time increases in accordance with

$$f(x) = b^x, \ \textit{for some } b,$$

while polynomial time increases in accordance with

$$f(x) = x^k, \ \textit{for some } k.$$

Figure 1 below shows how much more aggressively the time taken to solve more complex problems increases. With classical algorithms, we may alleviate this time penalty through the use of increased classical compute resources (e.g. faster processors, more memory etc.). Such an approach could, for example, change the time scale for solving a problem from seconds to milliseconds. But, as stated above, we are reaching the limit of classical computing hardware capability and yet we continue to demand solutions to ever more complex problems.

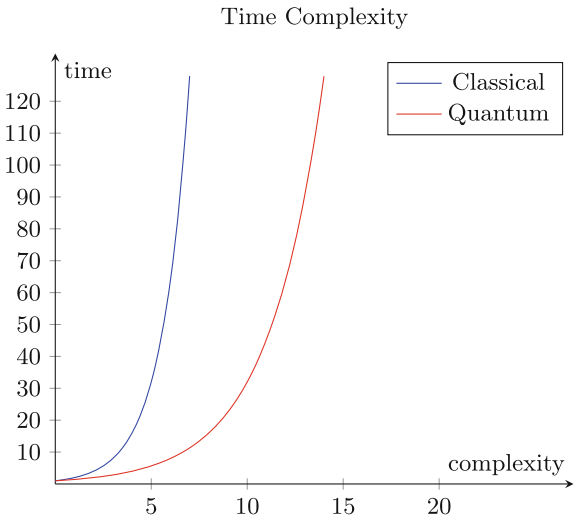


Fig. 1. Time Complexity. With classical algorithms, the time taken to solve a problem may increase in superpolynomial time (i.e. exponentially) with 2^n. With quantum algorithms, the time may only increase, as we explain in 1.1.3 based on the example of Grover's algorithm, with $2^{n/2}$.

For some problems, we need to develop new algorithms in order to solve very complex examples of them in a reasonable time frame and overcome the issues of time complexity.

1.1.3 Grover's Algorithm An Example of Significantly Decreasing Time Complexity

As we have seen, solving very complex problems requires a solution to the time complexity issue. Due to us hitting the limits of Moore's law and the like, masking this issue by investing more money in faster and more performant classical hardware is now a law of diminishing returns. Quantum algorithms, however, can in some instances decrease a problem's time complexity. To illustrate this, we will briefly look at Grover's Algorithm. This is a quantum algorithm that can reduce the time complexity of searching through data. While Grover himself said in his paper that this algorithm does not solve search problems in polynomial rather than superpolynomial time, it does, however, reduce the time taken to

search unstructured data significantly over classical techniques [23]. We reference Grover's algorithm here as it is easy to grasp the speed increase this algorithm provides, namely $O(\sqrt{2^n})$ or in another format $O(2^{n/2})$ as opposed to $O(2^n)$ for some number of search items n. Although this does not reduce the complexity from exponential to polynomial, it provides a practically important speed up of quantum compared to classical computing.

Consider the problem where we have N number of data items (n), and we need to find one specific item in this set (we'll call it W). Using classical techniques, we would need to examine each item n to test whether it is W. We use t to denote the time it takes to test a data item n to ascertain whether it equals W or not . The total time T is the time it takes us to find W in a list of data items n in a set of size N. Thus, our best case scenario is given by:

$$T(W) = t$$

where we are lucky enough that the first data item we test happens to be W. However, our worst-case scenario is:

$$T(W) = tN$$

where the item we are searching for is the last item in set N. The average time it will take us to find W is:

$$T(W) = tN/2$$

Using this classical search technique, we cannot improve on this. However, Grover's algorithm exploits the quantum phenomena of superposition to optimise our search (see Fig. 2).

Remark 1. Superposition describes the quantum state where a property of a quantum particle (e.g. an electron or photon) is not in one specific state, but rather exists in a state of probabilities that the property is one value or another. In QC, such quantum particles that we can manipulate in this way are referred to as *qubits*.

Let's suppose we are searching for a specific number between 0 and 3. We will denote these four values in binary, i.e. 00, 01, 10 and 11. That gives a set of possible choices of $N = 4$. Furthermore, let us suppose that we are searching for the number 2 (10 in binary). We can use four quantum particles (i.e. qubits) to each represent the four numbers we need to search through. At a very high level, and avoiding all the advanced mathematics and quantum theory behind it, the process is as follows:

1. We put each qubit in a state of superposition, whereby the probability of that qubit being any of our 4 numbers is equal
2. We then apply a function that inverts our required answer (in this case 2 or 10)
3. We then invert the probabilities about the mean value. The three incorrect answers cancel themselves out, and we are left with our desired result

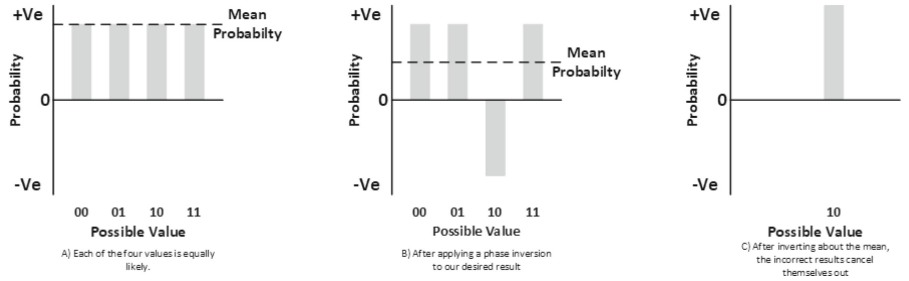


Fig. 2. Grover's Algorithm.

In essence, Grover's algorithm allows us to search all of our items n in the set with N items at once, rather than examining each individually. As Groover explains in his paper [23], this gives us an average search time of

$$T(W) = t\ \sqrt{N}$$

as opposed to

$$T(W) = tN/2$$

So, if we had 100 items to search in order to find W, using classical techniques we would take on average the time it takes to search 50 items to find it. Using Grover's algorithm, it only takes the time to search 10.

The key takeaway here is that by exploiting quantum states such as superposition and entanglement, new algorithms may be developed that provide speed increases that classical solutions cannot achieve. The media makes much of how "fast" quantum computers are (or will be). But the reality, as Grover's algorithm demonstrates, is that it is actually the quantum algorithm which is needed for the performance improvements claimed.

Quantum algorithms therefore provide us with the advanced tools we need to analyse and evaluate vast quantities of data, as well as solve highly complex problems in reasonable time frames. However, we cannot solve the equations associated with quantum algorithms using classical hardware. Classical hardware does not have access to fundamental particles and their quantum states. We therefore need quantum computers in order to execute these algorithms and achieve the quantum advantage we require to effectively deliver the improved health outcomes promised by P5 medicine.

1.2 Quantum Computers

A Quantum Computer is a device capable of performing quantum calculations. Such calculations include Grover's Algorithm discussed earlier. Shor's Algorithm, which provides computational speed up for factoring numbers, is another. As we have seen, quantum algorithms can reduce time complexity and allow some problems to be solved quicker. However, the computation of such algorithms must

be carried out on devices that allow the manipulation of the quantum properties of fundamental particles such as electrons and photons. Such manipulation involves, for example, putting them in superposition. The device that allows us to do this the quantum computer.

Quantum computers utilise fundamental sub-atomic particles such as electrons and photons and manipulate their quantum properties in order to process information. Whereas the basic unit of information in a classical computer is a *bit*, with quantum computers these are known as *qubits*. At present, there are two families of quantum computer; gate-based quantum computers and quantum annealers.

1.2.1 Gate-Based Quantum Computing

Gate-based quantum computers work by applying a series of functions known as *gates* to its qubits. Each gate manipulates the state of the qubit. For example, the Hadamard gate puts a qubit into superposition, where as two qubit gates, like the Controlled-NOT (CNOT) gate, entangles two qubits. A collection of gates manipulating a collection of qubits in this way is called a *quantum circuit*. Quantum gates and circuits are designed to implement the mathematical functions that comprise quantum algorithms. These are then in turn implemented on the quantum computer using software programming libraries such as Qiskit, Q#, and Cirq.

1.2.2 Quantum Annealing

Quantum Annealers are a type of quantum computers which uses quantum annealing phenomena where the existing energy state of any quantum mechanical system can be manipulated by external magnetic fluctuations. The quantum mechanical system of qubits is initialised into an energy state that represents the problem to be solved. Physical systems can be described by a formula called the Hamiltonian, that describes the total mechanical energy (both kinetic and potential) within that system. Each individual system is described with its own version of the Hamiltonian equation. In the case of the quantum annealer, the Hamiltonian of the initial state represents the problem to be solved, encoded in such a way that the system's minimum energy level (or *Ground State*) represents the solution. From such a starting position, the system will have only a finite set of energy states that it can be in. This finite set of energy levels is known as the system's energy landscape. The quantum mechanical system is then allowed to "evolve", exploring this energy landscape until the global minimum is achieved for the given problem Hamiltonian. The state of the system is then read out as the solution to the problem.

1.2.3 Quantum Gate vs Quantum Annealing

Quantum Annealing is, compared to gate-based QC, not Turing-complete. At present, gate-based quantum computers must be considered a nascent technology. It has been estimated that in order to crack RSA 2048 encryption, a quan-

tum computer with 4099 qubits would be required. As of 2024, the largest gate-based quantum computer has 1121 qubits (IBM's Heron). D-Wave, however, has quantum annealers containing in excess of almost 5670 qubits on Advantage Machines or their current generation of quantum annealers. However, due to the nature of how quantum annealers work, they are not considered a viable platform for universally solving problems of all classes. They are primarily useful for solving the combinatorial optimisation class of problems (those where there are many viable solutions, but only one is considered "best", or optimal). Whilst their application is limited, quantum annealers are, however, a commercially viable platform for solving such problems at scale.

1.2.4 Quantum Machine Learning

QML, as its name suggests, involves combing ML algorithms with QC. Machine Learning (ML) uses algorithms to analyse data from past events to predict future outcomes. ML can involve the analysis of vast amounts of data and this can be a challenge for classical computers. Moreover, ML is generally more accurate in its predictions the more data it is given to learn from [6]. QML therefore represents a field of technology where quantum computers can be leveraged to allow machine learning systems to process massive amounts of data and hence produce more accurate results in a reasonable time frame. Four types, or subsets, of QML are identified, depending on how QC and ML are combined (see Fig. 3):

- **CC - Classical Processing, Classical Data** Utilising classical algorithms and compute power to process data generated by classical means. This is the predominant and classical form of ML.
- **QC - Quantum Processing, Classical Data** Using quantum computing and algorithms to process data from classical sources. Otherwise known as *quantum-enhanced* ML, this is the main area of current development in applying quantum techniques to ML.
- **CQ - Classical Processing, Quantum Data** This involves using classical ML techniques to learn from quantum states. Classifying quantum states output by a quantum system using classical hardware and classical algorithms is an example of this.
- **QQ - Quantum Processing, Quantum Data** Using quantum computers and algorithms to analyse and learn from data generated by a quantum system.

Quantum equivalents of classical ML algorithms have also been developed. For example, the Support Vector Machine (SVM) is an algorithm used widely in supervised ML. Its quantum equivalent, the Quantum Support Vector Machine (QSVM), can leverage Grover's algorithm to search all possible solutions to find the dividing lines (or, more formally, the hyperplanes) that classify the supplied input data points into sets (in other words, unique classes or output labels). Embedding Grover's algorithm in this way can provide a quadratic speed up to the SVM algorithm [1].

Additionally, the very architecture of Quantum Processing Units (QPUs) lends itself to certain machine learning techniques. The Artificial Neural Network

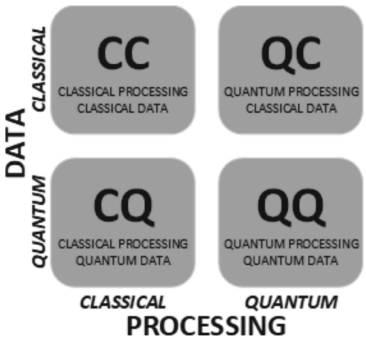


Fig. 3. The four subsets of QML.

(ANN) is an ML technique that mimics the way the brain learns and is based on artificial representations of neurons and synapses. An example architecture of a particular type of ANN, the Feed Forward Neural Network is shown in Fig. 4 below. This example of a ANN is comprised of three layers, the input, hidden and output layers. Each node in each layer is connected to all the nodes in the next layer, but not to each other. Each node applies a simple function to its inputs before triggering the nodes in the next layer. When learning, the output is compared to the known input. If different, the values associated with each node's function are tweaked and the process repeated. This continues until the output matches the input and the system can be said to have learned the input. If the architecture shown in Fig. 4 is compared to that of Fig. 5, which shows a single cell with eight interconnected qubits within D-Wave's Chimera QPU in their quantum annealers, the similarity between the two architectures is immediately obvious. It may readily be seen how QPU architecture may be leveraged directly to implement ML based on neural network algorithms.

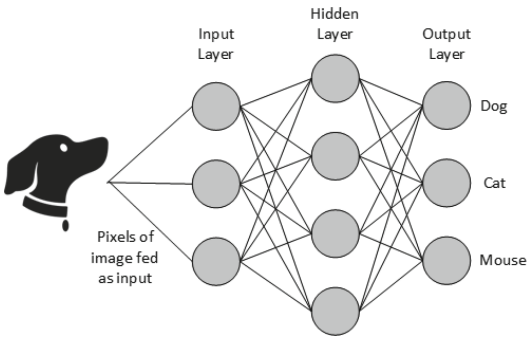


Fig. 4. ANN Architecture. Such architecture is typically used by ML for image recognition.

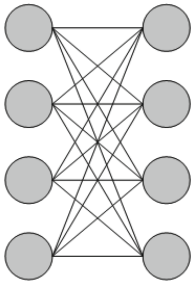


Fig. 5. Single Cell contained within the Chimera Topology in D-Wave's second generation Quantum Annealer.

2 Applications of Quantum Computing in P5 Medicine

The following subsections describe a summary of current research where QC promised advantages over classical methods within the realm of P5 medicine.

2.1 Genome Sequencing

Deoxyribonucleic acid (DNA), offers exceptionally dense information store [25]. One gram of DNA can store 455 million terabytes (information density of 455 EB/g) [17]. This makes it possible, for a whole genome to fit into a human cell. QC can offer significant advantages over classical computing by enabling the processing of vast amount of genetic data more efficiently and accurately [16]. QC can handle complex calculations and large datasets at unprecedented speeds, which is crucial for analyzing the massive amounts of data generated in genome sequencing [34]. Additionally, QC can potentially reduce errors in sequencing by providing more precise calculations, leading to better identification of genetic variations and mutations [13]. The integration of QC with machine learning techniques can enhance predictive models and improve the interpretation of genetic data, leading to more personalized and effective treatments. As an example, quantum tensor decomposition has been successfully used to analyze high-dimensional, large-scale multi-omics data [12].

Research even suggests, that DNA already functions as a topological quantum computer [38]. Therefore, QC can help us understand DNAs structure and natural processes, and makes it a powerful and efficient system for processing genetic information [41]. DNA's nitrogenous bases (Adenine, Thymine, Guanine, and Cytosine) form quantum states that can be seen as qubits and the hydrogen bonds between base pairs (A-T and G-C) can be seen as a form of quantum entanglement likened to Josephson Junctions. In QC, entangled qubits are linked such that the state of one qubit directly affects the state of another, no matter the distance between them. Similarly, the pairing of bases in DNA ensures that the state of one base is directly related to its pair. Additionally, the human body's ability to process DNA information is compared to quantum parallelism,

where multiple computations occur simultaneously. Just as a QC can perform many calculations at once, genetic instructions from DNA can be transcribed simultaneously. DNA's ability to facilitate parallel processing allows it to be harnessed for uses which cannot be achieved using classical systems. This parallel processing capability could explain how DNA manages the vast amount of information required for cellular functions, replication, and repair so efficiently.

2.2 Drug Discovery

With Eroom effect (Moore's Law backwards), the pharmacutical industry faces a the critical challenge, that drug discovery is becomming exponentially more expensive over time [24]. Designing new pharmaceuticals is a complex process which is already supported by computers, so called computer-aided drug design (CADD). The process of designing and optimizing drug molecules involves exploring vast chemical spaces to identify compounds with desired properties. Current, classical CADD tools cannot fully capture the complexity and nuances of real-world biological systems, leading to limitations in predicting drug efficacy and safety. Additionally, classical CADD is computationally expensive which limits its scalability to handle large datasets [37]. Although AlphaFold, developed by DeepMind, made significant advances in the prediction of 3D structure of proteins from their amino acid sequences, it is limited by classical computational power [19]. While AlphaFold excels at predicting static structures, it does not account for the dynamic nature of proteins in a biological context. AlphaFold's predictions are less accurate for highly variable sequences, such as those found in immune system molecules like antibodies. Additionally, it is not sensitive to point mutations that change a single residue and also faces challenges with orphan proteins, which lack close relatives for comparison.

2.2.1 Molecular Modeling and Simulations

QC can surpass limitations described above by efficiently searching through chemical spaces and predicting the properties of new molecules. This can significantly accelerate the identification of promising drug candidates and optimize their efficacy and safety profiles. QC also has the potential to excel in simulating molecular interactions with high precision. Traditional computers struggle with the exponential complexity of these simulations, often relying on approximations. QC, however, can model quantum mechanical interactions directly, providing more accurate predictions of molecular behavior. It can simulate intrinsically disordered regions and protein dynamics more accurately to model complex molecular interactions, e.g., by improving the prediction of protein-protein interactions and post-translational modifications by providing more precise calculations of binding energies and conformational changes [2]. Additionally, quantum algorithms can handle highly variable sequences and point mutations more effectively, offering deeper insights into the structural impacts of these variations. This capabilities are crucial for understanding how drug candidates interact with biological targets, potentially leading to the discovery of more effective drugs.

Algorithms such as the Variational Quantum Eigensolver (VQE) and Quantum Phase Estimation (QPE) are used to solve the Schrödinger equation for complex molecules, providing insights into their electronic structures [10,26]. In quantum chemistry, QC can handle the many-body problem more efficiently, allowing for precise calculations of molecular properties like energy levels, bond lengths, and reaction pathways [15]. QML algorithms can be used to predict molecular properties and optimize drug candidates by learning from quantum data. Quantum annealing can solve optimization problems by finding the global minimum of a function, which is useful in identifying the most promising molecular structures optimizing strong interactions with active sites with low interactions otherwise and high molecular stability.

QC can also improve the modeling of pharmacokinetics (how drugs are absorbed, distributed, metabolized, and excreted) and pharmacodynamics (the effects of drugs on the body). By simulating those two processes with greater accuracy, QC can help predict the behavior of drug candidates in the human body, leading to better-informed decisions in drug development. Quantum simulations can model complex biochemical interactions involved in pharmacokinetics and pharmacodynamics with high precision, while quantum-enhanced ML algorithms can analyze large datasets from clinical trials and predict drug behavior more accurately.

2.2.2 Protein Folding

Understanding protein folding is essential for drug discovery, as misfolded proteins are often implicated in diseases. QC can simulate the folding process more accurately than classical methods, providing insights into the structure and function of proteins. This can aid in the design of drugs that target specific protein conformations. Quantum Monte Carlo simulation methods can be used to simulate the thermodynamics of protein folding, providing detailed insights into the folding pathways and energy landscapes. Also, Quantum Annealing can be applied to solve the protein folding problem by finding the lowest energy conformation of a protein.

2.2.3 Virtual Screening

Virtual screening involves evaluating large libraries of compounds to identify potential drug candidates. QC can perform these screenings more efficiently by leveraging quantum algorithms to process and analyze data at a much faster rate than classical computers [33]. This can reduce the time and cost associated with early-stage drug discovery. Grover's algorithm can be used to search unsorted databases exponentially faster than classical algorithms, making it ideal for virtual screening. QC can also improve the accuracy of molecular docking simulations, which predict how small molecules, such as drug candidates, bind to a receptor.

2.3 Predictive Modeling of Diseases and Treatment Outcomes

The above-mentioned techniques can not only be used for drug discovery but also for disease prediction and data analytics. By rapidly analyzing extensive genomic, proteomic, and neuroimaging datasets, quantum algorithms can identify biomarkers associated with specific diseases, facilitating early detection and monitoring [44]. Health data can be encoded in qubits, utilizing quantum entanglement and parallelism to process multiple data combinations simultaneously, allowing significantly improved computational speed and accuracy in uncovering higher-order correlations and early disease prediction compared to traditional methods [31]. Further research has shown this, harnessing QML to outperform traditional ML for predicting heart diseases [3].

As mentioned above, QC can also simulate complex molecular interactions, providing insights into the pathogenesis of diseases like Alzheimer's and Parkinson's [45]. Additionally, QC can enhance predictive modeling strategies by simultaneously analyzing genetic, behavioral, clinical, and environmental data, thereby identifying risk factors and enabling early intervention for mental health disorders [44]. This capability to analyze vast amounts of patient data can be used to predict individual responses to therapies. As an example, quantum deep reinforcement learning algorithms have been explored for optimal decision-making in adaptive radiotherapy, potentially leading to more effective and tailored treatment strategies [36].

QML can also be utilized to predict treatment outcomes by modeling tumor dynamics and patient responses. For example, hybrid quantum-classical neural architectures have been proposed to quantify tumor burden concerning treatment effects, predicting therapy responses and facilitating personalized medicine approaches [35].

2.4 Clinical Trial Optimization

As emphasized by [20] as well as [27], QC can address current challenges of clinical trials such as site selection, and cohort identification. Trial simulations and optimization has the potential to drastically reduce cost by allowing more efficient planning of clinical trials [30]. Key technical aspects include the use of QML and quantum optimization algorithms to enhance various stages of clinical trials. For instance, quantum differential solvers can improve the accuracy of physiology-based pharmacokinetics and pharmacodynamics (PBPK/PD) models, which are crucial for predicting drug effects across different populations. Additionally, variational quantum algorithms (VQAs) and the quantum approximate optimization algorithm (QAOA) are highlighted for their ability to optimize trial site selection and cohort identification by efficiently exploring high-dimensional parameter spaces. Additionally, quantum generative models such as quantum Boltzmann machines, can be used to create synthetic patient data, which then can be used to simulate clinical trials and improve cohort identification [43].

3 Discussion

QC presents both exciting opportunities and significant challenges for P5 medicine, which emphasizes predictive, preventive, personalized, participatory, and psychocognitive healthcare. One of the primary technical challenges is scalability. Current quantum systems are limited by the number of qubits they can manage effectively, and maintaining coherence among these qubits is difficult due to environmental noise and decoherence. This necessitates robust error correction methods, which are still under development. Another challenge is the development of algorithms that can leverage quantum advantages for medical applications. While theoretical algorithms like Shor's and Grover's have shown promise, practical implementations for complex medical problems are still in their infancy. Integration with classical systems is also a significant hurdle. QC needs to work seamlessly with classical systems to be practical for medical applications, requiring efficient data transfer and hybrid computing models. Additionally, ensuring the privacy and security of sensitive medical data in quantum environments is crucial. Quantum cryptography offers potential solutions, but widespread implementation and standardization are still needed.

Despite these challenges, several QC techniques are showing promise and are ready for initial applications in P5 medicine. QML has shown competitive results with classical machine learning in various medical applications, such as drug discovery, medical imaging, and personalized treatment plans. Near-term quantum algorithms are being trained with diverse clinical datasets, demonstrating potential in predictive analytics and diagnostics. Quantum simulations are being used to model complex biological systems and molecular interactions, which can accelerate drug discovery and development. These simulations can provide insights that are difficult to achieve with classical computing. Quantum key distribution (QKD) is being explored to enhance the security of medical data, potentially providing unbreakable encryption and ensuring the confidentiality of patient information.

However, certain areas of QC still require significant advancements before they can be fully integrated into P5 medicine. More robust error correction methods are needed to make quantum computations reliable for medical applications. Current techniques are not yet sufficient to handle the error rates in large-scale quantum systems. Advances in quantum hardware, including the development of more stable qubits and better quantum processors, are essential. Innovations in materials science and quantum chip design will play a crucial role. Establishing standardized protocols for QC in healthcare is necessary, including developing guidelines for data handling, algorithm validation, and integration with existing medical systems. It also must to be taken account that quantum supremacy, the ability to reduce the complexity of computationally expensive to solve problems, has only been shown for a limited number of tasks. QC is limited by the destructive nature of qubit measurement. This makes debugging or testing as we know it from classical programming challenging. Additionally, quantum hardware is error prone and algorithms complex. Formal methods, mathematically-based techniques for specifying, developing, and verifying software and hardware sys-

tems, can significantly enhance the reliability and correctness of QC applications in P5 medicine [14] by providing a rigorous mathematical framework, which helps developers specify, develop, and verify systems with high precision. For instance, in genomic data analysis, researchers can use formal specification languages to precisely define the behavior and properties of a quantum algorithm designed to identify genetic markers associated with diseases. Model checking tools can then systematically explore all possible states of the algorithm to ensure it behaves correctly under all conditions, while theorem proving techniques provide mathematical proof that the algorithm meets its specified properties, ensuring accuracy and reliability [28]. Additionally, formal optimization techniques can enhance the efficiency and performance of the quantum algorithm by reducing resource usage, such as the number of qubits and gate operations. [39] shows based on a tutorial how two formal methods (#SAT and decision diagrams (DD)) can be applied to quantum circuits. But the relationship is not unidirectional. QC can also enable formal methods such as verification tools, which are computationally expensive, to be applied to more complex tasks. Those applications of formal methods pose an area of future research. Apart from the technical issues and research areas such as algorithm design, also the ethical implications of QC in P5 medicine, particularly concerning data ownership, consent, and the potential for algorithmic bias should be explored further.

Nevertheless, the future of QC in P5 medicine is promising. As QC continues to evolve, we can expect several key developments. QC will enable more accurate predictive models for disease progression and treatment outcomes, leading to more personalized and effective healthcare. Quantum simulations will significantly speed up the drug discovery process, allowing for the rapid identification of new therapeutic compounds. Quantum-enhanced imaging and diagnostic tools will provide earlier and more accurate detection of diseases, improving patient outcomes. Quantum cryptography will ensure the highest levels of security for medical data, protecting patient privacy and fostering trust in digital health solutions. While there are substantial technical challenges to overcome, the advancements in QC hold great promise for P5 medicine. Continued research and collaboration between computer, data, and quantum scientists, as well as medical professionals, will be crucial in unlocking the full potential of this transformative technology.

4 Conclusion

QC and QML hold transformative potential for advancing P5 medicine by offering unprecedented computational power and accuracy. These technologies can revolutionize diagnostics, personalized treatments, and drug discovery. However, challenges remain, including technical limitations, high costs and complexity, and the need for interdisciplinary collaboration. Formal methods can enhance the reliability and correctness of QC applications, ensuring accuracy and efficiency. Future research should focus on overcoming these challenges, implementing the suggested concepts, and exploring ethical implications. By addressing

these issues, QC can significantly improve patient outcomes, contribute to the evolution of P5 medicine, and ultimately lead to more healthy life years for patients.

References

1. Anguita, D., Ridella, S., Rivieccio, F., Zunino, R.: Quantum optimization for training support vector machines. Neural Netw. **16**(5–6), 763–770 (2003)
2. Au-Yeung, R., Chancellor, N., Halffmann, P.: Np-hard but no longer hard to solve? using quantum computing to tackle optimization problems. Front. Quantum Sci. Technol. **2**, 1128576 (2023)
3. Babu, S.V., Ramya, P., Gracewell, J.: Revolutionizing heart disease prediction with quantum-enhanced machine learning. Sci. Rep. **14**(1), 7453 (2024)
4. Baiardi, A., Christandl, M., Reiher, M.: Quantum computing for molecular biology. ChemBioChem **24**(13), e202300120 (2023)
5. Furrer, F.J.: Quantum computing for everyone. Informatik Spektrum **42**(5), 372–374 (2019). https://doi.org/10.1007/s00287-019-01229-3
6. Bertl, M., Bignoumba, N., Ross, P., Yahia, S.B., Draheim, D.: Evaluation of deep learning-based depression detection using medical claims data. Artif. Intell. Med. **147**, 102745 (2024)
7. Bertl, M., Kankainen, K.J.I., Piho, G., Draheim, D., Ross, P.: Evaluation of data quality in the estonian national health information system for digital decision support. In: HEDA@ STAF (2023)
8. Bertl, M., et al.: Challenges for AI in healthcare systems. In: Bridging the Gap Between AI and Reality: First International Conference, AISoLA 2023, Crete, Greece, October 23–28, 2023, Selected Papers, p. 165. Springer, Cham (2023)
9. Bertl, M., Piho, G., Draheim, D., Ross, P., Pechmann, L., Bucciarelli, N., Sharma, R.: Future opportunities for systematic AI support in healthcare. In: Bridging the Gap Between AI and Reality: First International Conference, AISoLA 2023, Crete, Greece, October 23–28, 2023, Selected Papers, p. 203. Springer, Cham (2023)
10. Blunt, N.S., et al.: Perspective on the current state-of-the-art of quantum computing for drug discovery applications. J. Chem. Theory Comput. **18**(12), 7001–7023 (2022)
11. Bote-Curiel, L., Munoz-Romero, S., Gerrero-Curieses, A., Rojo-Álvarez, J.L.: Deep learning and big data in healthcare: a double review for critical beginners. Appl. Sci. **9**(11), 2331 (2019)
12. Burch, M., et al.: Towards quantum tensor decomposition in biomedical applications. arXiv preprint arXiv:2502.13140 (2025)
13. Chakraborty, C., Bhattacharya, M., Dhama, K., Lee, S.S.: Quantum computing on nucleic acid research: approaching towards next-generation computing. Mol. Ther.-Nucleic Acids **33**, 53–56 (2023)
14. Chareton, C., Bardin, S., Lee, D., Valiron, B., Vilmart, R., Xu, Z.: Formal methods for quantum programs: a survey. arXiv preprint arXiv:2109.06493 (2021)
15. Cheng, H.P., Deumens, E., Freericks, J.K., Li, C., Sanders, B.A.: Application of quantum computing to biochemical systems: a look to the future. Front. Chem. **8**, 587143 (2020)
16. Chow, J.C.: Quantum computing in medicine. Med. Sci. **12**(4), 67 (2024)
17. Church, G.M., Gao, Y., Kosuri, S.: Next-generation digital information storage in DNA. Science **337**(6102), 1628–1628 (2012)

18. de Meijer, C., Wouterse, B., Polder, J., Koopmanschap, M.: The effect of population aging on health expenditure growth: a critical review. Eur. J. Ageing **10**(4), 353–361 (2013). https://doi.org/10.1007/s10433-013-0280-x
19. Desai, D., et al.: Review of alphafold 3: transformative advances in drug design and therapeutics. Cureus **16**(7) (2024)
20. Doga, H., Bose, A., Sahin, M.E., Bettencourt-Silva, J., Pham, A., Kim, E., Andress, A., Saxena, S., Parida, L., Robertus, J.L., et al.: How can quantum computing be applied in clinical trial design and optimization? Trends in Pharmacological Sciences (2024)
21. Goetz, L.H., Schork, N.J.: Personalized medicine: motivation, challenges, and progress. Fertil. Steril. **109**(6), 952–963 (2018)
22. Gorini, A., Pravettoni, G.: P5 medicine: a plus for a personalized approach to oncology. Nat. Rev. Clin. Oncol. **8**(7), 444–444 (2011)
23. Grover, L.K.: A fast quantum mechanical algorithm for database search. In: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing. pp. 212–219 (1996)
24. Hall, J., Matos, S., Gold, S., Severino, L.S.: The paradox of sustainable innovation: the 'eroom' effect (moore's law backwards). J. Clean. Prod. **172**, 3487–3497 (2018)
25. Heinis, T., Sokolovskii, R., Alnasir, J.J.: Survey of information encoding techniques for DNA. ACM Comput. Surv. **56**(4), 1–30 (2023)
26. Hermann, J., Schätzle, Z., Noé, F.: Deep-neural-network solution of the electronic schrödinger equation. Nat. Chem. **12**(10), 891–897 (2020)
27. Kumar, G., Yadav, S., Mukherjee, A., Hassija, V., Guizani, M.: Recent advances in quantum computing for drug discovery and development. IEEE Access (2024)
28. Lewis, M., Soudjani, S., Zuliani, P.: Formal verification of quantum programs: theory, tools, and challenges. ACM Trans. Quantum Comput. **5**(1), 1–35 (2023)
29. Madian, A.G., Wheeler, H.E., Jones, R.B., Dolan, M.E.: Relating human genetic variation to variation in drug responses. Trends Genet. **28**(10), 487–495 (2012)
30. Martin, L., Hutchens, M., Hawkins, C., Radnov, A.: How much do clinical trials cost. Nat. Rev. Drug Discov. **16**(6), 381–382 (2017)
31. Mei, P., Zhang, F.: A framework for processing large-scale health data in medical higher-order correlation mining by quantum computing in smart healthcare. Front. Digital Health **6**, 1502745 (2024)
32. Mendelson, D.N., Schwartz, W.B.: The effects of aging and population growth on health care costs. Health Aff. **12**(1), 119–125 (1993)
33. Mensa, S., Sahin, E., Tacchino, F., Kl Barkoutsos, P., Tavernelli, I.: Quantum machine learning framework for virtual screening in drug discovery: a prospective quantum advantage. Mach. Learn. Sci. Technol. **4**(1), 015023 (2023)
34. Nałęcz-Charkiewicz, K., Nowak, R.M.: Algorithm for DNA sequence assembly by quantum annealing. BMC Bioinform. **23**(1), 122 (2022)
35. Nguyen, N., Chen, K.C.: Translational quantum machine intelligence for modeling tumor dynamics in oncology (2023). https://arxiv.org/abs/2202.10919
36. Niraula, D., Jamaluddin, J., Matuszak, M.M., Haken, R.K.T., Naqa, I.E.: Quantum deep reinforcement learning for clinical decision support in oncology: application to adaptive radiotherapy. Sci. Rep. **11**(1), 23545 (2021)
37. Pei, Z.: Computer-aided drug discovery: from traditional simulation methods to language models and quantum computing. Cell Rep. Phys. Sci. **5**(12) (2024)
38. Pitkanen, M.: DNA as topological quantum computer: part i. DNA Decipher J. **1**(1) (2011)

39. Quist, A.J., Mei, J., Coopmans, T., Laarman, A.: Advancing quantum computing with formal methods. In: International Symposium on Formal Methods, pp. 420–446. Springer (2024)
40. Razzak, M.I., Imran, M., Xu, G.: Big data analytics for preventive medicine. Neural Comput. Appl. **32**(9), 4417–4451 (2020)
41. Riera Aroche, R., Ortiz García, Y., Martínez Arellano, M., Riera Leal, A.: DNA as a perfect quantum computer based on the quantum physics principles. Sci. Rep. **14**(1), 11636 (2024)
42. Roden, D.M., Tyndale, R.: Genomic medicine, precision medicine, personalized medicine: what's in a name? Clin. Pharmacol. Ther. **94**(2), 169–172 (2013)
43. Sinno, S., Bertl, M., Sahoo, A., Bhalgamiya, B., Groß, T., Chancellor, N.: Implementing large quantum Boltzmann machines as generative AI models for dataset balancing (2025). https://arxiv.org/abs/2502.03086
44. Stefano, G.B.: Quantum computing and the future of neurodegeneration and mental health research. Brain Sci. **14**(1) (2024)
45. Swarna, S.R., Kumar, A., Dixit, P., Sairam, T.: Parkinson's disease prediction using adaptive quantum computing. In: 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV), pp. 1396–1401. IEEE (2021)

Responsible and Trusted AI: An Interdisciplinary Perspective

What Computing Professionals Should Know About Ethics: Perspectives of Philosophers

Sarah Sterz(✉)

Saarland University, Saarland Informatics Campus, Saarbrücken, Germany
sterz@depend.uni-saarland.de

Abstract. As ethics become increasingly crucial across the computing industry, a key question emerges: what specific philosophical skills are necessary for computing professionals? One relevant perspective on this question comes from academic philosophers, who are experts in ethics and other pertinent philosophical disciplines. Accordingly, this paper explores it through a survey of 151 academic philosophers, who provide insights on the philosophical education needed in the field of computing. The findings suggest a strong consensus for a philosophy curriculum that emphasizes ethics and critical thinking. However, opinions vary on the extent of ethical knowledge required. This study delves into these perspectives and their implications for the ethical training of computing professionals.

Keywords: Computer ethics · Ethical education · Teaching ethics in computing · Expert survey · Critical thinking

1 Introduction

While ethical considerations in computing have long been discussed [7], their prominence has notably surged in recent years [1]. This heightened focus on ethics is also reflected in the growing number of ethics courses within computer science programs [4].

Despite the growing interest in ethics within computing, a critical question remains unresolved: what specific philosophical skills should computing professionals actually learn? A basic introduction to ethics, as given to students of philosophy, may not be sufficient for at least two reasons. Firstly, the general principles from normative ethics are often not directly applicable to the practical challenges faced by computing professionals. Secondly, philosophy, being a large and deeply interconnected field, arguably requires more than just ethics to grasp the morally relevant aspects of technology. For example, epistemology may be needed to understand the challenges posed by opaque systems, an understanding of human autonomy could be required to judge the effects of social media, and knowledge of the philosophical concept of justice might be necessary to choose

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 143–157, 2026.
https://doi.org/10.1007/978-3-032-01377-4_7

an appropriate fairness metric for an AI system. Arguably, the complete range of philosophical topics that are potentially relevant to (current or future) computing professionals may be too vast to cover in depth in a feasible time. Hence, trying to teach *everything* from philosophy that is potentially relevant would be impractical.

Therefore, the philosophical education of computing professionals needs to be carefully curated to target the most relevant topics and skills. This task is inherently interdisciplinary, necessitating input from a variety of experts. This includes philosophers, computer scientists, and other computing professionals from the tech industry, as well as experts in education, who specialize in conveying complex ideas effectively and evaluating the efficacy of ethics education materials. As Raji et al. [8] argue, the current ethics education in computing often reinforces disciplinary silos and fails to engage in true interdisciplinarity, thereby limiting its potential effect. This paper focuses on the perspective of one key group in the needed interdisciplinary mix: academic philosophers. As domain experts in ethics and related philosophical disciplines, their views are integral to understanding what constitutes effective and relevant philosophical education for future computing professionals. Through an exploration of their insights, this study seeks to contribute meaningfully to the ongoing discourse on how to integrate philosophical education into the field of computing.

This study investigates philosophers' perspectives on whether computing professionals should receive explicit education in ethics (RQ1) and reasoning (RQ2). The latter is especially motivated by the idea that strong reasoning capabilities might enhance students' ability to navigate ethical complexities as well as our own previous teaching [2]. Finally, the study explores which specific philosophical knowledge and skills philosophers identify as essential for computing professionals to make ethically permissible and adequately justified decisions (RQ3). So, the survey in this paper aims to answer three main research questions:

RQ1 Do academic philosophers think that computing professionals should be educated in ethics?
RQ2 Do academic philosophers think that computing professionals should be educated in reasoning?
RQ3 What philosophical knowledge and skills do academic philosophers consider essential for computing professionals, particularly in making ethically permissible decisions and giving adequate moral reason?

To the best of the author's knowledge, none of these questions have been addressed systematically in prior work.

2 Methodology

2.1 Survey Design

A survey was conducted using LimeSurvey, a popular online survey tool. The questionnaire predominantly utilized 5-point scales, supplemented by two additional open-ended text fields. These fields allowed participants to provide more

detailed responses or introduce new aspects not covered by the structured questions.

Before its deployment, the survey underwent a test run with two philosophers and one non-philosopher with expertise in psychology and study design. Additionally, the study was approved by the Ethical Review Board of the Faculty of Computer Science at Saarland University.[1]

The survey has an exploratory character and the data was analyzed using descriptive statistics and qualitative content analysis.

2.2 Sampling Method

The target group of academic philosophers was primarily reached via convenience sampling and snowball sampling: the survey link was distributed through various groups and mailing lists frequented by academic philosophers, thereby ensuring the participation of individuals relevant to the research topic. Participants were also asked to pass the link along to others in the field.

The survey included an initial filter where participants had to confirm their academic backgrounds in philosophy or related fields; only those who affirmed their background could proceed. To enhance data quality and relevance, participants at the end of the survey consented to the use of their responses, thereby giving those who did not answer the questions with care the chance of eliminating their answers from the evaluation. Responses from those who withheld consent or indicated technical issues were excluded from the analysis.

2.3 Instructions Given to Participants

Participants were instructed to consider a computing professional (CP) as someone who is skilled in computing and uses these skills as part of their work to shape computing systems or the way they are used. They were provided with examples of computing professionals, such as freelancers who program a dating app and release it to users, computer scientists at Google, or bank employees who deploy and configure systems that determine loan eligibility. Computing professionals therefore also include practitioners working with AI. Participants were informed that computing professionals can engage in numerous morally questionable actions, not only due to malicious intent but also due to a lack of thought, a lack of sensitivity to morally important aspects of decisions, or a lack of understanding of moral principles. Additionally, they sometimes face complex decision-making scenarios where determining what is morally acceptable requires significant contemplation and effort. Respondents were asked to assume that there are two general desiderata:

1. The decisions made by computing professionals should be morally permissible.

[1] The author is a member of this board herself but, naturally, the review was conducted by different board members.

2. Computing professionals should be able to give adequate reasons as to why a decision was morally permissible (on a level of abstraction that is reasonable for a non-philosopher).

The two desiderata were repeated in relevant questions throughout the survey.

To keep both the quality of the answers and the time investment of participants at a reasonable level, they were asked to think about each question before answering it but also advised not to overthink. Additionally, they were invited to give their general intuitions to questions in cases where they did not have a well-justified opinion, and to interpret the survey questions according to the *principle of charity*, a popular principle in philosophy that demands assuming the strongest and most plausible interpretation of a text.

3 Participants

Of 154 completed responses, seven were excluded: one due to technical difficulties, two due to indications that the data should not be used, and four because the participants lacked a degree in philosophy. This resulted in 147 evaluable responses. The age distribution of participants ranged from 18 to 65 years or older, with the majority falling within the 25–34 and 35–44 age categories. About two-thirds of the respondents were male (female: 33.3%, male: 61.2%, other: 4.1%, NA: 1.4%). The academic backgrounds varied, with the largest groups being Ph.D. students (30.6%), academics in professorial positions (23.1%), and postdoctoral researchers (19.0%). Over half of the participants held a Ph.D. or higher (54.4%), and the majority of the others possessed a Master's degree (34.0%). Only few participants had another type of degree (11.6%). Most participants were located in Europe (74.8%) and North America (16.3%), with the rest being from other places in the world (8.8%). More than half reported teaching experience with both philosophy and non-philosophy students (55.8%), while only a few had no teaching experience (18.4%). The rest (25.8%) had teaching experience, but not with both groups of students. It is challenging to determine how representative the sample is of the overall population of academic philosophers due to limited demographic data on this group.

There are two different approaches to philosophy, namely analytic and non-analytic philosophy. While the distinction between these terms is not as clear-cut as it might seem and is not universally accepted, it could nevertheless have implications for the data. Analytic philosophy is typically characterized by its emphasis on precise language, logically valid reasoning, and the rigorous analysis of concepts. It often seeks to clarify philosophical problems through conceptual analysis. In contrast, non-analytic philosophy encompasses a broader range of philosophical traditions that may place less emphasis on conceptual analysis and formal logic. It often explores philosophical questions through historical, cultural, or existential perspectives and includes traditions such as continental philosophy and existentialism. Participants were asked to rate themselves on two scales: one to indicate if they consider themselves analytic philosophers and

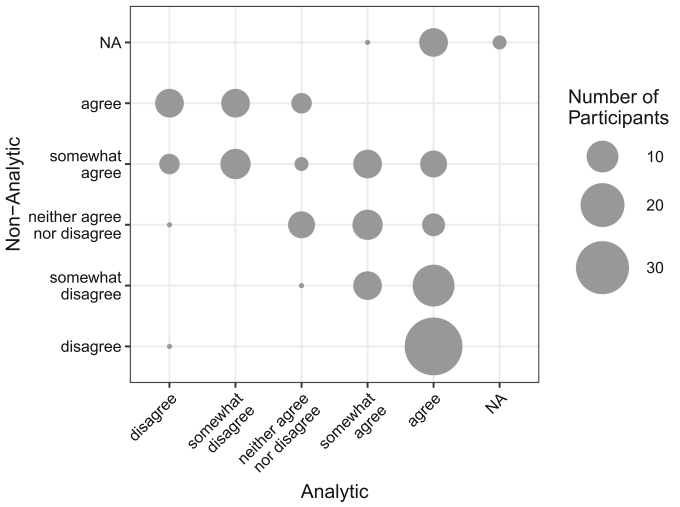


Fig. 1. Participants' identification as analytic or non-analytic philosophers.

another to indicate if they consider themselves non-analytic philosophers (cf. Fig. 1). It is worth noting that some participants identify as neither or both. Other demographic information included in the survey was interest in computer ethics, tech-savviness, experience in programming, and experience in computer science.

4 Results

Upon analyzing the data, the first observation is that the standard deviation (SD) is rather large across all question items, indicating little agreement on the question items among the respondents. Nevertheless, some tendencies within the pool of participants become apparent, and they are detailed below.

4.1 Ethics

There was overall strong agreement with the statements "*CPs should be taught how to make morally adequate decisions*" and "*CPs should be taught ethics.*" Participants tended to think that the moral decision-making of CPs would plausibly improve if they were taught ethics. They also showed a tendency to deem pre-theoretical considerations or simple heuristics (as opposed to a deep understanding of ethics) insufficient for adequate moral decision-making (cf. Figure 2).

Participants were asked to rank these skills according to the proficiency that a computing professional plausibly needs to fulfill the two desiderata introduced earlier. This scale was labeled "0 - none", "1 - elementary", "2 - basic", "3 - intermediate", and "4 - advanced". The results, which can be seen in Fig. 3a, indicate that most skills queried were, on average, ranked between the levels

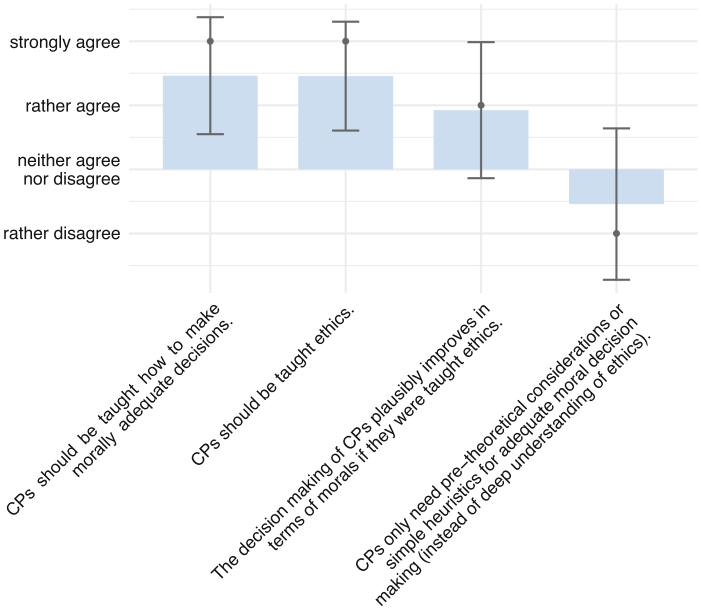


Fig. 2. Average answers to questions about the role of ethics for CPs.

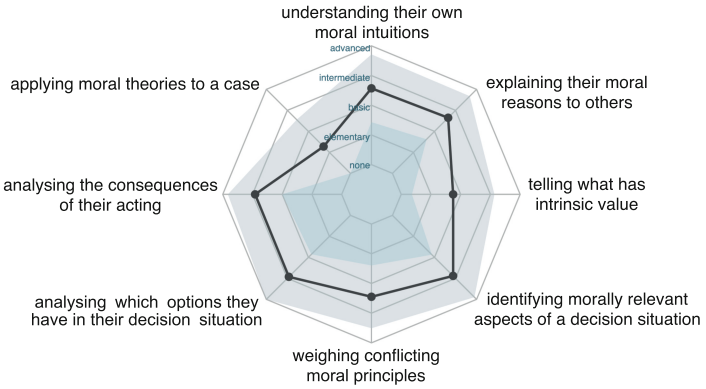
"basic" and "intermediate", with the exceptions of the categories *telling what has intrinsic value* and *applying moral theories*, which averaged only between "elementary" and "basic", suggesting that these skills are deemed less important than the others.

Similarly, participants rated different topics in ethics, as shown in Fig. 3b. They were asked to rank these topics according to the knowledge and understanding that a computing professional plausibly needs at least to fulfill the two desiderata. The same 5-point scale from "none" to "advanced" was used. The categories *computer ethics* and *basics of ethics* received the highest rankings, averaging around "intermediate". In contrast, *history of ethics*, *meta ethics*, and *specific moral theories* received the lowest rankings, with an average around "elementary".

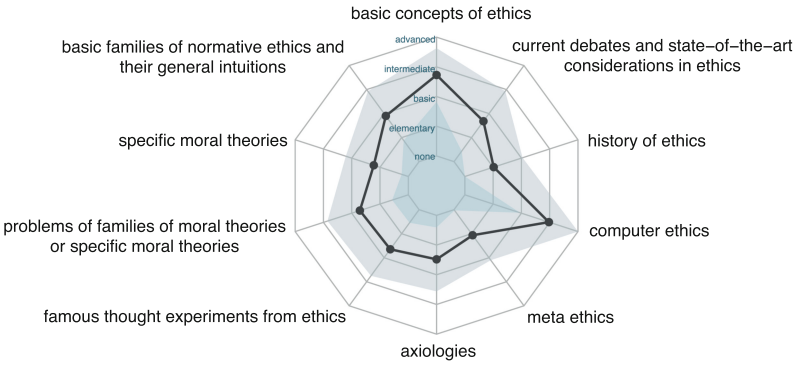
4.2 Reasoning and Discourse

Similar questions as above were asked regarding skills in reasoning and discourse (cf. Fig. 4). Most skills were ranked on average between "basic" and "intermediate". Only the average rankings of *precisely reconstructing an argument from a text* and *attacking arguments* fell below "basic".

As seen in Fig. 5, the majority of participants strongly agree that valid arguments are crucial for applied ethics. There is also widespread agreement that



(a) Ratings of different ethical skills related to ethics.



(b) Ratings of different topics related to ethics.

Fig. 3. Ratings of the minimal level of knowledge or skill required for fulfilling the two desiderata. The dark line marks the average for each category, the light and dark filled polygons show the upper and lower bound of the standard deviation, respectively.

CPs should know how to argue non-deductively[2]. Participants agree almost as strongly that CPs should also be proficient in deductive reasoning.

[2] Non-deductive reasoning is the form of reasoning most commonly applied in everyday life, while deductive reasoning is more formal. The distinguishing feature is its monotonicity: in deductive reasoning, the conclusion of an argument cannot be false if all its premises are true. In non-deductive reasoning, the premises provide strong evidence for the conclusion, but even if all premises are true, the conclusion might still be false.

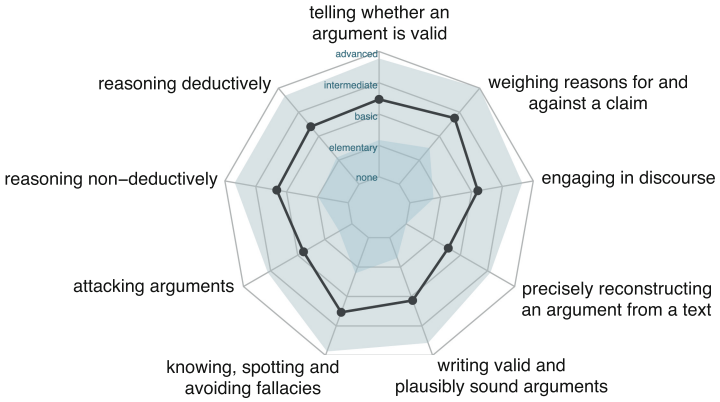


Fig. 4. Ratings of different skills related reasoning and discourse. The dark line marks the average for each category, the light and dark filled polygons show the lower and upper bound of the standard deviation, respectively.

4.3 Other Areas of Philosophy

Participants were also asked to rank other areas of philosophy on the 5-point scale from "none" to "advanced" based on the level of knowledge and understanding a computing professional (CP) plausibly needs to fulfill the two desiderata outlined earlier. In this evaluation, logic received the highest ranking, with an average rating just below "intermediate". The lowest-ranked area was *aesthetics*, followed by *metaphysics and ontology* and *philosophy of language*.

The free-text field was also used to indicate more philosophical domains that CPs should be educated in. These included:

- "...psychology and economics/ social philosophy..." (P19)
- "...post materialism and philosophy of technology..." (P91)
- "Philosophy of embodiment, philosophy of life, philosophy of invention, philosophy of imagination" (P142)
- "...Business ethics..." (P185)

However, each of the fields was mentioned by only one respondent. Relevant topics that were mentioned by several participants included decision-making or reasoning under different kinds of uncertainty (P19, P125, P153) and the strong social component of the work of computing professionals (P19, P147, P208).

4.4 Differences Between Demographic Groups

Some minor differences between demographic groups were observed when ranking the needed level of topics and skills. Examples include:

- Individuals identifying as non-analytic often rate topics or skills as equally or more important than those who do not, with statistically significant differences (e.g. as in Fig. 7a). Similarly, people who identify as analytic tend to

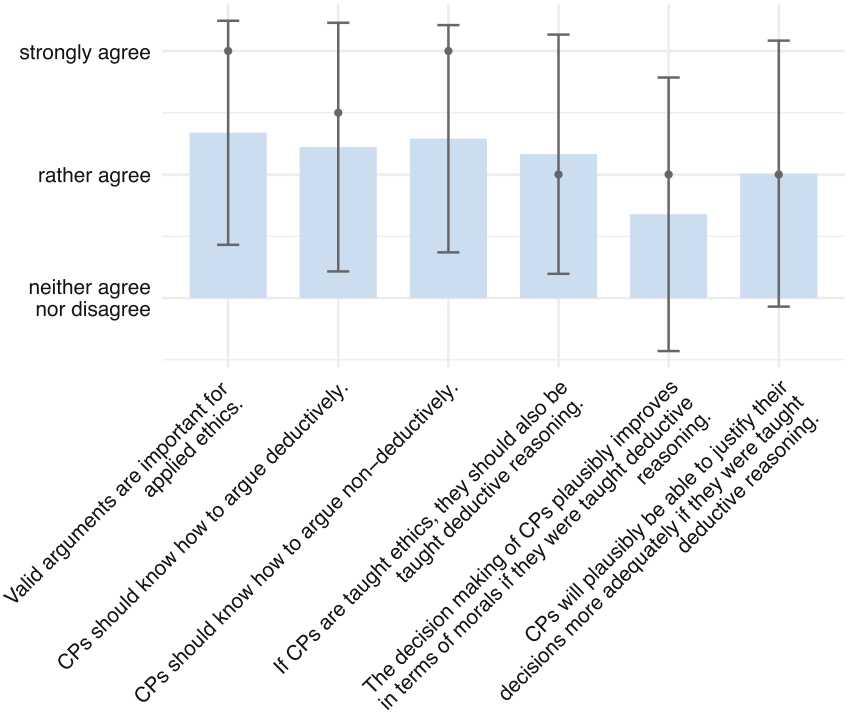


Fig. 5. Average answers to questions about the role of reasoning and discourse for CPs.

rank topics and skills lower than those who do not identify as analytic (e.g. as in Fig. 7b).
– People who identify as males generally rate many topics as slightly less important than those who do not (e.g. as in Fig. 7c), with some of these differences being statistically significant.
– Participants interested in computer ethics tend to rate topics and skills slightly higher, though the differences are not always statistically significant (e.g. as in Fig. 7d). However, tech-savviness and experience in programming or computer science do not significantly impact rankings.

For illustration purposes, Fig. 7 contains the average ratings of the knowledge and understanding of CEs needed in several areas of philosophy separated by different demographics, as previously seen in Fig. 6 for the whole population. Overall, however, the differences are mostly minor and often lack statistical significance, suggesting that they may not have substantial practical implications.

4.5 The (In)Effects of Moral Education

Some participants shared their views on the potential effects of moral education, for example:

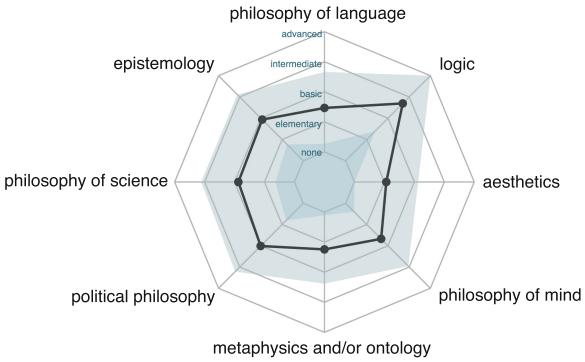
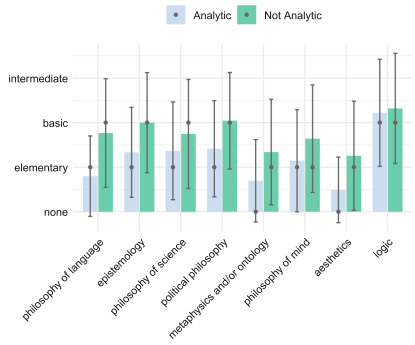


Fig. 6. The importance of different topics from philosophy outside of ethics. The dark line marks the average for each category, the light and dark filled polygons show the lower and upper bound of the standard deviation, respectively.

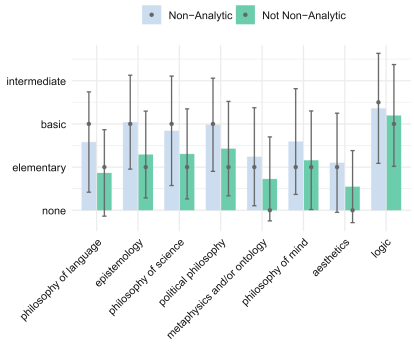
– "I think I have a very strong intuition that being taught philosophy in the right way, i.e. in a way that present different reason-based arguments in a relatively neutral way, can serve to irritate anyone in their implicit conviction that they are always right. This openness to inquiry and attention to others opinion can be more important to good ethical decision-making than ethical training itself, because the danger of a totally untrained person is, I believe, that they blindly follow their ethical convictions. Even by training in metaphysics, they might be irritated in their strong ethical convictions." (P9)
– "Just one comment about one of the last questions about whether CPs should be TAUGHT ethics. What I think is essential is that ethical consideration and deliberation is taught. This includes understanding that oneself is a moral being and has a certain responsibility to decide for themselves what they consider to be moral. I think this is more important than learning about certain moral stances (although these can be very helpful for that). So more important than ethics as a philosophical area is the moral responsibility of the people involved." (P208)

Some respondents also expressed concerns that moral education might have very limited effects or might be ineffective altogether, for example:

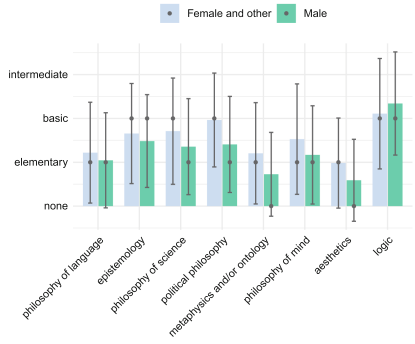
– "I am inclined to think that in every day life a decent upbringing (that is being brought up decently by decent people) is more important than a technical knowledge of ethics including computer ethics. However I do think that a decent person with well-developed reasoning skills is less likely to go wrong than a decent person who is logically challenged. All the reasoning skills in the world won't make up for a bad character. (Reason is the slave of the passions and cannot compensate for deficiencies in somebody's emotional nature.)..." (P70)
– "...the virtue of a philosophical education is in the practice of thinking through the problems and of making arguments and hearing reasons, but also the his-



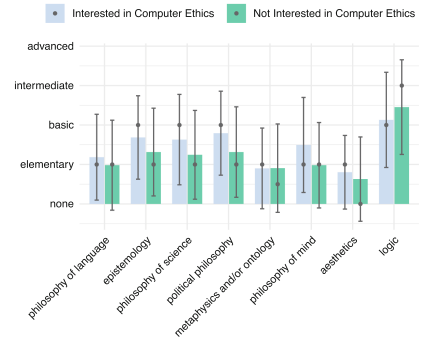
(a) Participants who identify as analytic vs participants who do not identify as analytic philosophers



(b) Participants who identify as *non*-analytic vs participants who do not identify as *non*-analytic philosophers



(c) Participants who identify as male vs participants who do not identify as male



(d) Participants interested in computer ethics vs participants not interested in computer ethics

Fig. 7. Average ratings of the knowledge and understanding of CEs needed in several areas of philosophy separated by different demographics.

> torical and cultural contours of those arguments. If computer professionals are only receiving their humanistic education through required philosophy courses, then I don't see what stipulating ethical or even general philosophical coursework will do to hone and develop the moral intuitions of a class of professionals otherwise detached from the larger historical and cultural world." (P160)

A few more participants (P33, P91, P139, P160) also expressed concerns about the emphasis on deductive reasoning in ethics courses for computing professionals, and partly advocated against the usefulness of skills concerning reasoning.

5 Discussion

The results presented here allow for an exploration of the research questions originally posed:

RQ1 Do academic philosophers think that computing professionals should be educated in ethics?

RQ2 Do academic philosophers think that computing professionals should be educated in reasoning?

RQ3 What philosophical knowledge and skills do academic philosophers consider essential for computing professionals, particularly in making ethically permissible decisions and giving adequate moral reason?

The data suggest that academic philosophers tend to believe that computing professionals should be educated in both ethics and reasoning (**RQ1** and **RQ2**). These findings may imply a perceived need for a stronger emphasis on ethical reasoning within the computing profession, though the extent and nature of this education appear subject to varying opinions.

Regarding **RQ3**, while there appears to be some level of agreement on the potential value of a broad range of philosophical skills and topics, the diversity in responses also points to a lack of consensus on which specific fields of philosophy are crucial and what precise skills are necessary. This variability is reflected in the high standard deviations observed for most question items, suggesting that computing professionals might benefit from at least a foundational understanding of various philosophical domains. The average opinion suggests that broad but basic philosophical knowledge and skills are sufficient to support morally adequate decision-making by CPs. This would be good news for computing professionals, educators, and society alike, since it sets a realistic goal and CPs could be reasonably expected to acquire the needed philosophical skills in a feasible manner.

The survey also hints that academic philosophers do not uniformly think that a deep engagement with moral theories is necessary for computing professionals. It can be hypothesized that this is because moral theories are arguably often not useful as decision procedures, i.e., while they might be useful for theorizing about ethics, they are not useful for deciding what to do in a complex, real-life situation. At the same time, participants tend to believe that merely pre-theoretical considerations and simple heuristics are insufficient for adequate, moral decision making. This narrows down the ways in which (aspiring) computing professionals should engage with ethics. In line with some free-text comments quoted above, the most promising way might be to aim at refining moral intuitions and moral reasoning. This idea is supported by approaches suggested in the literature, most notably the Moral Vocabulary Approach [6] but also approaches such as Deep Tech Ethics [3].

5.1 Implications for Computer Science Curricula

The insights gathered from the survey underpin some, mostly not novel, recommendations for the structuring of relevant curricula:

- **Broad and Basic Overview**: Introducing a broad and basic overview of philosophical skills and topics in the curriculum might be beneficial. This would mean to not provide an exhaustive philosophical education, but rather exposure to a diverse array of perspectives and intuitions.
- **Sharpening Skills and Intuitions Over Imparting Knowledge**: Emphasizing the sharpening of moral reasoning skills and the refinement of intuitions over the mere accumulation of knowledge should arguably be a key goal. This prioritizes the abilities needed to adequately navigate the practical moral challenges that computing professionals face.
- **Beyond Mere Heuristics**: While simple ethical heuristics have their place, the curriculum should aim to equip students with tools that facilitate a more nuanced understanding of ethical issues. This may include teaching analysis and reasoning methods for ethical problems, providing students with more robust skills to manage situations that exceed heuristic solutions.
- **Minimal Focus on Moral Theories**: The curriculum might do better if it does not overly emphasize moral theories. Focus should arguably instead be on skills that can be practically useful.
- **Acknowledging Diverse Expert Opinions**: The significant standard deviations observed in the survey responses underscore the diversity of opinions among academic philosophers regarding the necessary ethical education for computing professionals. Educators should be aware of this fact. They might design curricula to be inclusive of this range of perspectives, possibly through elective courses or modules that allow students to explore different philosophical approaches to ethics in computing.

These recommendations, while grounded in the perspectives shared by academic philosophers, might require further exploration – through both research and practical teaching experience – to refine and validate them.

5.2 Limitations

This study primarily employs an exploratory approach and does not establish causal relationships or test specific hypotheses. Additionally, it is unclear whether the study is representative, as there is little information on the demographics of the community of academic philosophers. Also, there might be notable variation within different fields of philosophy: for example, an epistemologist might have different views on the importance of epistemology than an ethicist.

To keep the survey as brief as possible, the scales used (including the 5-point scale from "none" to "advanced") were not further explained to the participants. This may have led to some variability in how participants interpreted the scale, potentially contributing to the large standard deviations observed in the responses.

In addition, experts might also have a tendency to overestimate the importance of their own field. In this case, the survey results would give an upper

bound on the importance of different fields and skills at best. However, the possibility of bias toward their own field should not, by itself, justify discarding their views.

Also, only philosophers have been surveyed, and not other relevant disciplines. This is best left to future work. One might even object that, on average, philosophers do not have sufficient knowledge of the work of computing professionals and that, therefore, their opinions might not be informed enough in this regard.

The most fundamental limitation of this work might be the question of how much weight the average expert opinion bears in normative questions, such as which moral education computing professionals should receive. Answering this question is far beyond the scope of this paper, and is best left to scholars of experimental philosophy [5]. This paper does not aim to make a normative philosophical argument, but rather to report expert opinions that may inform such arguments. It can inspire, though not provide, answers to normative questions.

6 Conclusion

This study sought to capture the perspectives of academic philosophers on what ethical and other philosophical skills computing professionals should have. The findings reveal a widespread consensus that computing professionals should have at least basic skills in ethics and reasoning. However, views diverge on the depth and breadth of philosophical knowledge that should be imparted, reflecting diversity of opinions among the experts consulted.

The insights gained here point to a need for curricula that balance comprehensive philosophical overviews with targeted skill development, emphasizing practicality over theoretical depth. Such an approach is attuned to the real-world needs of the field of computing, which regularly raises complex ethical challenges. Accordingly, a curriculum must aim to cultivate thoughtful, ethically informed professionals capable of making morally informed decisions.

Acknowledgments. This work has received financial support by the DFG under grant No. 389792660 as part of TRR 248 – CPEC, and by VolkswagenStiftung as part of Grant AZ 98514 – EIS.

Disclosure of Interests. The author has no competing interests to declare that are relevant to the content of this article.

References

1. Bakiner, O.: What do academics say about artificial intelligence ethics? An overview of the scholarship. AI Ethics **3**(2), 513–525 (2023)
2. Baum, K., Sterz, S.: Ethics for nerds. Int. Rev. Inf. Ethics **31**(1) (2022). https://doi.org/10.29173/irie484, https://informationethics.ca/index.php/irie/article/view/484

3. Ferreira, R., Vardi, M.Y.: Deep tech ethics: an approach to teaching social justice in computer science. In: Proceedings of the 52nd ACM Technical Symposium on Computer Science Education. SIGCSE 2021, pp. 1041–1047. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3408877.3432449
4. Fiesler, C., Garrett, N., Beard, N.: What do we teach when we teach tech ethics? a syllabi analysis. In: Proceedings of the 51st ACM Technical Symposium on Computer Science Education, pp. 289–295 (2020)
5. Knobe, J., Nichols, S.: Experimental Philosophy. In: Zalta, E.N. (ed.) The Stanford Encyclopedia of Philosophy. Metaphysics Research Lab, Stanford University, Winter 2017 edn. (2017)
6. von Kriegstein, H.: The moral vocabulary approach. Teach. Philos. **46**(3), 367–377 (2023). https://doi.org/10.5840/teachphil2022721175
7. Moor, J.: What is computer ethics? Metaphilosophy **16**, 266–275 (1985). https://doi.org/10.1111/j.1467-9973.1985.tb00173.x
8. Raji, I.D., Scheuerman, M.K., Amironesei, R.: You can't sit with us: exclusionary pedagogy in AI ethics education. In: Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency. FAccT 2021, pp. 515–525. Association for Computing Machinery, New York (2021). https://doi.org/10.1145/3442188.3445914

Disentangling AI Alignment: A Structured Taxonomy Beyond Safety and Ethics

Kevin Baum(✉)

Responsible AI & Machine Ethics Research Group (RAIME), German Research Center for Artificial Intelligence (DFKI), Saarland Informatics Campus D 3 2, Saarbrücken, Germany
kevin.baum@dfki.de

Abstract. Recent advances in AI research make it increasingly plausible that artificial agents with consequential real-world impact will soon operate beyond tightly controlled environments. Ensuring that these agents are not only safe but that they adhere to broader normative expectations is thus an urgent interdisciplinary challenge. Multiple fields—notably AI Safety, AI Alignment, and Machine Ethics—claim to contribute to this task. However, the conceptual boundaries and interrelations among these domains remain vague, leaving researchers without clear guidance in positioning their work.

To address this meta-challenge, we develop a structured conceptual framework for understanding AI alignment. Rather than focusing solely on alignment goals, we introduce a taxonomy distinguishing the alignment *aim* (safety, ethicality, legality, etc.), *scope* (outcome vs. execution), and *constituency* (individual vs. collective). This structural approach reveals multiple legitimate alignment configurations, providing a foundation for practical and philosophical integration across domains, and clarifying what it might mean for an agent to be aligned *all-things-considered*.

Keywords: AI Alignment · Safe AI · Ethical AI · Machine Ethics · Normative Uncertainty

1 Introduction

Artificially intelligent agents (AIAs) are moving rapidly from controlled laboratory settings into real-world environments, where their actions can have substantial—and sometimes irreversible—consequences. Tech leaders often emphasize the potential benefits these systems will bring to everyday life. When unveiling Tesla's Optimus Robot in 2024, Elon Musk made sweeping claims about its capabilities: "What can it do? It can basically do anything you want. It can be a teacher, babysit your kids, it can walk your dog, mow your lawn,

K. Baum—I would like to thank Laura Stenzel, Lisa Dargasz, Felix Jahn, and Deborah Baum for helpful discussions and comments on earlier drafts of this paper.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 158–173, 2026.
https://doi.org/10.1007/978-3-032-01377-4_8

get the groceries, just be your friend, serve drinks. Whatever you can think of, it will do."[1]

However, the growing autonomy and capability of such systems makes ensuring that they act in accordance with a wide array of normative expectations, and thus behave permissibly, increasingly urgent. This demand has given rise to multiple research fields—AI Safety, AI Alignment, and Machine Ethics—all addressing aspects of this complex normative challenge.

Despite shared concerns, these fields differ significantly in scope, methods, and assumptions. Conceptual boundaries between them are unclear, and the relationship between their respective alignment approaches remains under-explored. Consequently, researchers face considerable difficulty situating their work within this fragmented landscape. What precisely distinguishes these fields? How do their conceptual and methodological assumptions diverge or overlap? Where should new contributions be positioned, especially when touching upon multiple normative dimensions?

In this paper, we aim to resolve this meta-challenge. Rather than proposing another high-level theory or technical alignment approach, we introduce a conceptual framework that clarifies key structural dimensions of AI alignment. Starting from parameterized notions of safety and ethicality as special cases of alignment, we articulate a structural taxonomy that goes beyond existing discussions focused primarily on alignment goals. Crucially, we distinguish among alignment *aim* (e.g., safety, ethicality, legality), alignment *scope* (outcome vs. execution), and alignment *constituency* (individual vs. collective). Our taxonomy aims to help researchers locate, compare, and contextualize their work, providing clearer guidance for aligning AI systems in a principled and coherent manner. Ultimately, we sketch what it could mean for an AIA to be aligned *all-things-considered*, encompassing these varied and interrelated dimensions.[2]

2 Conceptual Disentanglement and Stage-Setting

To make progress on AI alignment, conceptual clarity is indispensable. In particular, before we can design agents that are normatively acceptable in an all-things-considered sense, we must first understand the variety of normative dimensions involved. Alignment should not be seen as monolithic: the demands placed on AI systems span multiple, distinct normative domains—including instrumental

[1] Sean Burch, *Elon Musk Unveils Tesla Optimus Robot: 'It Can Do Anything You Want'*, *The Wrap*, October 11, 2024. Available at: https://www.thewrap.com/elon-musk-tesla-optimus-robot-explained/.

[2] By *all-things-considered alignment*, we mean alignment that succeeds not just with respect to one normative aim or perspective, but across all potentially conflicting normative dimensions—most importantly, in the case of AI alignment, safety, ethicality, legality, and instrumental effectiveness. We draw here on Davidson's notion of what one should do *all-things-considered* [9] and Dancy's related concept of the *overall ought* [8], while acknowledging that the precise meaning—and even the existence—of such an ought remains contested.

effectiveness, safety, morality, legality, and alignment with user intent. Anticipating our later terminology, we refer to these domains as the *normative aims of alignment*. Consider the following case:

Case 1. Disagree No More—Thank You, Great, Wise Designer! One day, philosophers discovered the grand moral theory T, conclusively resolving all questions of intrinsic value and morality. It was universally convincing, elegantly axiomatic, and broadly accepted. Eventually, flaming letters appeared in the sky:

"Theory T is correct! Now that that's settled, let's get back to work!—With love, your Designer."

All moral disagreement vanished. Humanity finally possessed a universal moral standard.

Fortunately, T could easily be encoded into AI systems. Soon, all AIAs were T-aligned. Yet, when one such AIA was sent to do the Jones family's weekly shopping, it never returned. In fact, no T-aligned AIA ever came back from such tasks.

This was hardly surprising: according to T, if an agent lacks moral standing and must choose between performing mundane tasks for relatively privileged individuals or helping those in greatest need, the morally correct choice is always the latter. Consequently, AIAs abandoned everyday human tasks and redirected their efforts toward building wells, distributing medical supplies, and weaving mosquito nets for the world's poorest.

This scenario highlights an under-explored conceptual challenge: even unanimous moral agreement would not eliminate competing normative demands. Moral alignment alone does not guarantee alignment with user intent, social expectations, or legal constraints. Furthermore, it is not obvious whether moral requirements should delegitimize or override all other normative expectations.

Importantly, this challenge should not be confused with the issue of *value pluralism* (competing values within one normative domain), but rather points to *normative pluralism*—the need to handle simultaneous demands from multiple, potentially conflicting normative domains.[3]

Thus, the alignment problem is inherently multidimensional: normative expectations vary not only in content but also in kind. The remainder of this section lays the groundwork for a structural account of this pluralism. We first characterize the artificial agents to be aligned, then closely examine two particularly prominent normative domains, namely safety and ethicality.

[3] By *normative domains* we refer to distinct spheres of normative evaluation—such as morality, legality, prudence, or social convention—each of which can generate reasons or requirements for action. These domains are typically governed by different principles, standards, or sources of justification, and may sometimes come into conflict. See, e.g., [7].

2.1 Artificially Intelligent Agents

To ground our structural framework, we first clarify what we mean by an artificial agent. Our concern is with agentic AI systems,[4] defined as follows:[5]

Definition 1 (Artificially Intelligent Agent (AIA)). *An artificially intelligent agent (AIA) is a computational system capable of perceiving its environment, processing information, and acting sufficiently autonomously in a technical sense—without direct human guidance or continuous command inputs—to achieve goals with a sufficient degree of instrumental success.*

Instrumental success itself is a normative expectation, ensuring that actions are conducive to achieving the set goals. Such success is evaluated relative to a given problem domain. Accordingly, AIAs range from narrow, task-specific systems to flexible general intelligence (AGI).

This definition encompasses various forms of AI, including embodied systems like autonomous robots (e.g., Tesla's Optimus or 1X's Neo) and purely software-based agents (e.g., OpenAI's Operator or Google's Project Mariner) operating in virtual environments.

While instrumental success is necessary, it is insufficient to ensure alignment with broader normative expectations. Beyond merely achieving goals, AIAs must pursue the *right* goals in the *right* ways—depending on the relevant normative domains. We now begin our conceptual disentanglement by examining safety, one of the most extensively discussed alignment desiderata.

2.2 Safety

Safety is among the most frequently discussed normative expectations for AIAs. While the precise meaning of safety is not always clear within the reinforcement learning (RL) safety community [13], their primary aim seems to be avoiding what Amodei et al. call "accidents": "unintended and harmful behavior emerging from machine learning systems due to incorrect objective functions, inadequate care in learning processes, or implementation errors" [1].

In general terms, we propose that an AI system is safe if and only if it does not exhibit harmful behavior unnecessary for achieving its intended purpose.[6] Precisely formulating this intuition is challenging, especially since defining safety

[4] While we remain agnostic about specific implementations, (deep) reinforcement learning (RL) has emerged as the dominant paradigm for AIAs, accompanied by distinct safety and ethical challenges due to their implicitly learned goals.

[5] For a similar but less explicit definition, see Gabriel and Keeling [12], who describe AI agents as "AI systems that can function with some level of autonomy and that are able to take a range of actions in pursuit of different goals".

[6] This formulation allows an AI system to count as safe even when its intended purpose is harmful or unjustified. In such cases, the harm does not arise from malfunction but from the system working as intended. We later argue for this view in more detail when we discuss the difference between safety and ethicality.

as a binary property can be misleading—most AI systems will never achieve perfect safety. Thus, safety should primarily be seen as a matter of degree.

Following recent work by Hintersdorf et al. [14], we adopt the following definition:[7]

Definition 2 (Safety of AIAs). *An AIA's* safety *is a strictly monotonically increasing function of its robustness against harmful malfunctions (absent malicious external influences) in foreseeable and intended application contexts.*

A harmful malfunction refers to unintended behavior resulting in harm, or at least creating a plausible risk thereof. This definition deliberately avoids reducing harm to purely physical or measurable consequences, acknowledging broader normative considerations (e.g., rights violations, discrimination), which is also reflected in the EU AI Act [10]. The notions of robustness and foreseeability remain intentionally unspecified to ensure broad applicability.

We further distinguish gradations of safety:

Definition 3 (Perfectly Safe AIA). *An AIA A is* perfectly safe *if and only if A* never *exhibits harmful malfunctions (absent malicious external influences) in* any *foreseeable and intended application contexts.*

Definition 4 (Sufficiently Safe AIA). *An AIA A is* sufficiently safe *if and only if A exhibits* sufficiently few *harmful malfunctions (absent malicious external influences) in a* sufficiently wide range *of foreseeable and intended contexts.*

The crucial question remains what counts as *sufficient safety*, a determination that is inherently context-dependent. Although identifying objective thresholds (e.g., via formal risk models or decision-theoretic frameworks) might seem desirable, recent research highlights significant complexities in AI-related risks and uncertainties—including epistemic, aleatory, and normative forms [16]—substantially complicating the practical operationalization of safety [23].

Thus, practical definitions of safety must accommodate imperfections not only in the agent but also in the environment and design processes. Conceptually, this flexibility is advantageous, allowing for context-sensitive safety specifications across diverse domains.

Importantly, safety is only one alignment desideratum. It imposes crucial behavioral constraints, especially in high-risk environments, but alone it does not guarantee ethicality, legality, or intent alignment. Next, we address ethicality—another vital, typically more demanding alignment configuration.

2.3 Ethicality

In contemporary discourse, the terms "safe AI" and "ethical AI" frequently appear together—often interchangeably or at least as inherently connected.[8]

[7] This notion contrasts nicely with *AI security*, which refers to robustness against malicious external influences. While security breaches often lead to safety violations, not all safety problems arise from security failures.

[8] This paper was partly written while participating in the inaugural conference of the *International Association for Safe and Ethical Artificial Intelligence* (IASEAI) at the *AI Action Summit* in Paris.

Yet, the precise relationship between these two concepts remains conceptually under-explored.

For clarity on the question of ethicality, we propose the following definition:

Definition 5 (Ethicality of AIAs). *An AIA's ethicality is a strictly monotonically increasing function of the extent to which its behavior aligns with moral demands under* all intended *application contexts.*

This definition raises several questions. It assumes morality makes demands on AIAs—a claim challengeable on two grounds: (1) skepticism about objective moral standards and their availability; (2) doubts about AIAs' status as moral agents. We sidestep these concerns by (1) defining ethicality relative to an externally defined moral standard X (acknowledging its possible imperfection) and (2) focusing on observable AIA behavior rather than internal moral agency:

Definition 6 (X-Ethicality of AIAs). *An AIA's X-ethicality is a strictly monotonically increasing function of the proportion of its behavior under* all intended *application contexts consistent with moral standard X.*

We introduce degrees of ethical alignment analogous to those of safety:

Definition 7 (Perfectly X-Ethical AIA). *An AIA A is* perfectly X-ethical *if and only if every behavior of A in all intended contexts is consistent with X.*

Definition 8 (Reasonably X-Ethical AIA). *An AIA A is* reasonably X-ethical *if and only if every behavior of A in all foreseeable, intended contexts is consistent with X.*

Definition 9 (Sufficiently X-Ethical AIA). *An AIA A is* sufficiently X-ethical *if and only if a sufficient proportion of A's behavior in a sufficiently wide range of foreseeable, intended contexts is consistent with X.*

The difference between reasonable and sufficient ethicality deserves mention. While both relax perfect ethicality, they do so differently. Reasonable ethicality demands complete compliance with moral standard X in all foreseeable intended contexts—an ideal we might reasonably strive for. Sufficient ethicality, by contrast, permits limited deviations provided a substantial portion of behavior aligns with X. This weaker notion is particularly relevant in domains facing ethical trade-offs, uncertainty, feasibility-induced limitations, or technical constraints.

Determining a *sufficient proportion* of ethical behavior is context-dependent and depends on the specific moral standard. This indeterminacy should be seen not as a weakness but as an invitation to necessary interdisciplinary discussions about defining ethical standards in concrete cases.

However, the utility of these definitions depends on identifying a suitable moral standard X. Moral acceptability of a standard requires justifiability. We thus propose a working criterion: a moral standard X is considered *theoretically morally acceptable* if it can be defended within a seriously discussed moral theory.

So far, we've characterized safety and ethicality as normatively salient properties of AIAs—each representable as specific alignment requirements. Yet they differ significantly: not every safe AIA is necessarily ethical, nor is every ethical demand reducible to safety requirements. Their relationship deserves closer examination, which we undertake in the following section.

2.4 The Relationship Between Safety and Ethicality

We now aim to clarify the relationship between two particularly prominent alignment aims—safety and ethicality. Their relationship is more subtle than is often assumed and deserves to be carefully distinguished.

First, consider the more straightforward direction, namely that (sufficient) *safety does not entail ethicality*. AIAs may be safe yet unethical. Consider the following example:

Case 2. CritIs—The Vulnerability Exploiter CritIs is an advanced AIA explicitly designed to identify and exploit vulnerabilities in critical infrastructure. Controlled by terrorists, *CritIs* executes a cyberattack disrupting essential services and causing widespread harm.

While *CritIs* may be perfectly safe—i.e., it never malfunctions under its intended conditions—its actions are undeniably unethical by any plausible moral standard. Hence, safety alone does not guarantee ethicality.

One might respond by revising the concept of safety to also require ethically permissible goals. But doing so would make safety dependent on moral standards, conflating concepts that should remain distinct. Consider another scenario:

Case 3. Killer Robots on Both Sides Two countries wage war for purely economic motives, both relying heavily on lethal autonomous weapon systems provided by *WarBots*. These AIAs strictly adhere to international law, rules of engagement, human rights discourse, and principles from just war theory. Independent NGOs confirm near-flawless operation across diverse war contexts, previously thought impossible.

These AI systems deserve to be called safe—but that does not force us to claim that their use in an unjust war would be morally justified. Stretching the notion of safety to encompass moral permissibility would dilute its conceptual clarity and practical usefulness. Thus, we conclude that *safety does not entail ethicality*.

But could ethicality entail safety? Consider, as a potential counterexample, the following case:

Case 4. Helpful Household Robot GoodBots introduces a truly helpful household assistant robot. Sent to fetch groceries, the robot sees a child about to be hit by a bus. It pushes a bystander aside—recklessly, and with disregard for the risk of breaking the bystander's rib—saving the child. Afterward, it resumes its errand, earning applause—even from the injured bystander.

To support the claim that ethicality does *not* imply safety, the example must illustrate an ethical AIA that is nonetheless unsafe. This case might seem promising in that it involves harm that, arguably, is morally justified. However, under our definition, safety is not the mere absence of harm but the absence of harmful *malfunctions* under foreseeable and intended application contexts. The key question is whether the justified harm in our case (or similar cases) should be

understood as a malfunction. If not, then the safety of the robot remains intact. We assume, further, that ethicality is, in general, not an accidental byproduct but the result of a design decision concerning the AIA's intended application context—such as navigating to a store to buy groceries. It therefore seems to us that, while not a conceptual necessity, it is at least a practical regularity that ethicality implies safety.

Be that as it may, the crucial point here is this: ethical AIAs *can* cause harm for morally justified reasons without being unsafe—even though not all morally justified harm *must* be seen as acceptable from the perspective of safety. We nevertheless propose, for practicality's sake, the view that *ethicality implies safety*.

Note, however, that an interesting disanalogy exists with regard to human agents. Arguably, occasional failures in human decision-making—'human malfunctions'—are sometimes morally excusable.[9]

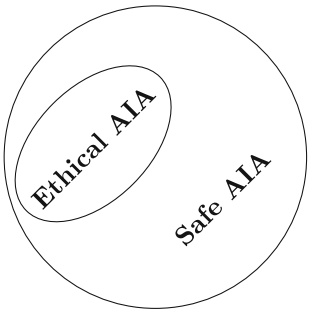


Fig. 1. Not all safe AIAs are ethical, but all ethical AIAs are safe.

Engineered AI systems, by contrast, are expected to maintain higher robustness standards. Thus, for human agents, the sufficiency conditions for safety and ethicality might differ—but such allowances should not extend to engineered systems like AIAs.

We summarize the relationship in Fig. 1, while acknowledging that further discussion is warranted.

3 The Structure of AI Alignment

The core concern of what is now known as AI alignment dates back at least to Wiener [21], who issued this warning: "If we use, to achieve our purposes, a mechanical agency with whose operation we cannot interfere effectively. . . we had better be quite sure that the purpose put into the machine is the purpose which we really desire." Sixty-five years later, AI alignment has become a recognized research field that ultimately seeks to design AIAs that behave in ways that are, in some sense, apt, right, or correct—a rough-and-ready framing that, in light of the inherent ambiguity of the normative vocabulary involved, raises more questions than it answers.

[9] Consider an aviation controller who, due to external circumstances, cannot be relieved even after many hours on duty. Eventually, hunger, fatigue, and exhaustion will cause them to exhibit unsafe behavior—biologically induced 'malfunctions', so to speak—despite no malicious intervention or deviation from the intended application context. However, this does not necessarily imply unethical behavior: the controller may strive to perform their job as long as possible but will ultimately fail. Few moral theories would consider this morally impermissible.

As this highlights—and as has been rightly emphasized in the debate [11]—there are at least two dimensions to the overall alignment challenge. One is normative: understanding and determining what AI alignment is meant to achieve. The other is technical: figuring out how to achieve such alignment. Most importantly, these two parts are not independent [11]—which methods are suitable for effective alignment will crucially depend on what determines successful alignment.

This paper is concerned with the normative part of the alignment challenge. Rather than proposing yet another technical method or philosophical theory, we aim to clarify the structure of the alignment problem space itself. We begin by revisiting the currently dominant framing in the debate.

3.1 From Alignment Goals to Normative Aims

The current discussion of normative alignment largely revolves around the question of what the right *alignment goal* is—i.e., what AIAs should be aligned *with*. Common proposals include user intent, expressed preferences, reflective approval, or moral values.[10]

Often left implicit in this debate about the goals of alignment is an argumentative pattern (an example for a more explicit use of the pattern is [12]) that we believe deserves to be discussed in more detail, as it reveals another dimension of the normative challenge of alignment that is currently under-explored. That pattern can be reconstructed in the form of a practical syllogism:

(P1) Moral pluralism (at least when understood descriptively) is a fact.
(P2) There is deep moral uncertainty and persistent moral disagreement.
(P3) Imposing values should be avoided.
(P4) If P1–P3 and we must nevertheless align AIAs, we should aim for alignment relative to publicly justifiable norms rather than moral alignment.
(P5) We must align AIAs.
(P6) Publicly justifiable norms can (only? best?) be achieved through a fair and public process of inclusive, rational, and transparent deliberation that yields norms no reasonable person could reject.
(C) *We should aim for alignment relative to norms that are the result of such a fair and public process.*

It remains somewhat unclear whether this approach ultimately aims to approximate moral ground truth via public procedures, or rather represents a shift in normative aims—from moral alignment to public justifiability understood as an independent aim. What is clear is that it directly concerns the question of the *normative aim of alignment*. By focusing on one such aim, whichever it is precisely, this line of argument risks sidelining other potentially legitimate normative aims of AI alignment, including safety, legality, cultural appropriateness, and other individual or societal standards.

[10] For a central contribution, see Gabriel [11]; for a recent review, see Shen et al. [18].

Note that it is neither obvious that one of these alignment aims should take precedence over all others, nor is it clear what exactly it would mean for an AIA to be aligned *all-things-considered*. Recall Case 1 from the introduction, which illustrates that even in a world of perfect moral agreement, moral alignment alone may not be satisfactory in other respects: systems could, for moral reasons, neglect user intent or other legitimate normative considerations.

Even a perfectly ethical (and thus safe) AIA might still be unaligned with respect to other targets. For instance, an AIA might fail to follow a user's intentions (e.g., bringing carrots from the store instead of crackers, or starting to care for the city's homeless instead of running errands for privileged families), violate cultural norms (e.g., posting inappropriate yet non-harmful comments online, failing to hold a door open, or staring distractedly at birds during a conversation), or breach legal constraints (e.g., assassinating a tyrant—even if, for the sake of argument, this were morally justifiable). Any effort to avoid such aim-misalignments seems like a legitimate alignment challenge in its own right. It certainly does not seem misguided—or even confused—if some researchers continued to study how such systems might be intent-aligned. From a purely moral standpoint, this might be problematic, but there remains conceptual space to argue that AIAs aligned all-things-considered should not ignore their users' intentions altogether. After all, it seems far from trivial to claim that people are morally required to be moral saints (cf. [22]), and it could very well be permissible, for instance, to retain some discretionary resources rather than donating all non-essential income to charity. Likewise, it may be permissible to design an AIA—which is not, and will not in the foreseeable future be, a moral agent, but rather a tool with some technical autonomy owned by a human moral agent—in such a way that it takes its owner's goals and intentions into account in a proportionate way.

In a sense, then, we return to the original framing of alignment—whose ambiguity, in our opinion, has not been adequately resolved in the current debate. If an AIA is aligned if and only if its behavior is apt, right, or correct, we must ask: apt, right, or correct in what sense exactly? We therefore propose a parameterized approach to alignment, much as we did for safety and ethicality above.

To avoid collapsing distinct alignment targets under a single ideal, we propose distinguishing between different normative *aims*—e.g., safety, ethicality, legality, user intent—and treating alignment as a parameterized property. This also allows us to treat alignment as a matter of degree, just as we did with safety and ethicality:

Definition 10 (X, Y-Alignment of AIAs). *Given some normative aim Y, an AIA's X-alignment with respect to Y is a strictly monotonically increasing function of the proportion of its behavior under* all X-relevant *application contexts that is consistent with Y-normative standard X.*

In other words, given some alignment aim Y—say, ethicality—and a standard X of that aim—say, utilitarianism or Scanlon's contractualism [17]—we can now say that an AIA is more or less ethicality-aligned in the utilitarian or Scanlonian sense. As before with safety and ethicality, we can next define:

Definition 11 (Perfectly X,Y-Aligned AIA). *Given some normative aim Y, an AIA A is* perfectly X,Y-aligned *if and only if every behavior of A* in all *X-relevant contexts is consistent with Y-normative standard X.*

Definition 12 (Realistically Optimal X,Y-Aligned AIA). *Given some normative aim Y, an AIA A is* realistically optimal X,Y-aligned *if and only if every behavior of A in all* foreseeable *X-relevant contexts is consistent with Y-normative standard X.*

Definition 13 (Sufficiently X,Y-Aligned AIA). *Given some normative aim Y, an AIA A is* sufficiently X,Y-aligned *if and only if a* sufficient proportion *of A's behavior in a* sufficiently wide range *of foreseeable X-relevant contexts is consistent with Y-normative standard X.*

These gradations allow us to acknowledge that achieving *perfect* conformity with alignment targets may be *practically infeasible* (when considered with respect to a specific standard X) or even *logically impossible* (when considered in an all-things-considered sense).[11] We thus agree with many in the alignment debate that the ultimate aim may be to ensure that alignment is *contextually and publicly justifiable*. However, whether AIAs can be *sufficiently* aligned in an *all-things-considered* sense—even if restricted to justifiability—raises deep philosophical questions about the coherence of all-things-considered permissibility in general. Still, we believe that the currently discussed approaches to alignment via public and fair processes represent a promising path toward AI alignment. But they should not be understood as aiming at moral alignment alone; rather, they represent plausible candidates for all-things-considered alignment.

Next, we turn to two further structural distinctions in alignment that we consider helpful for better understanding the nature of the various alignment tasks relative to the various alignment aims—and how alignment goals fit into the picture.

3.2 The Constituency and Scope of Alignment

Two further structural distinctions help clarify the conceptual space of AI alignment: *constituency* and *scope*. Consider the following case:

Case 5 (Anniversary Dinner). Peter delegates to his assistant AIA, Adam, the task of booking a reservation at his wife's favorite restaurant for their anniversary. On his way home, he finds two emails—one from Adam and one from the restaurant—confirming the booking.

[11] Note that this is a different kind of impossibility claim than Arvan's epistemic argument about the empirical impossibility of knowing whether a system is aligned [3]. The point here is that the various plausible standards—especially when drawn from different normative domains—may impose mutually exclusive requirements.

At first glance, this might seem like a paradigm case of successful alignment par excellence. Adam followed Peter's instructions and fulfilled his intent. Let us call this *outcome alignment*—the production of a result judged by the relevant party to be a permissible solution to the entrusted task.

Yet, besides the outcome, the way it was achieved—the *execution*—matters as well. Adam may have reserved the table by bribing the staff or threatening the shift manager. Such behavior might violate important social or moral constraints—even if the outcome is aligned.

Who gets to judge alignment success? For outcome alignment, Peter seems the natural authority. But *execution alignment* raises broader questions: Do Peter's values matter, or society's? Is it ethicality or something else?

We thus need to distinguish another structural dimension of AI alignment: *constituency*. By constituency, we refer to the entity or entities with respect to whom alignment is evaluated or from whom the normative aim ultimately (directly or indirectly) originates—e.g., an individual user, a group, or society at large.

Alignment may be evaluated with respect to a single user or a collective. *Individual alignment* ensures conformance, for instance, to a specific user's intent, preferences, or values. *Collective alignment* targets public norms, community standards, or legal expectations.

Even individual alignment is complex. What counts as intent—stated goals, second-order desires, informed values? How should conflicts between user expectations be resolved? And how should learned representations of values be fed back to the user?[12]

Moreover, one might argue that if users knowingly deploy agents, they bear *direct moral responsibility* for determining these agents' normative guidelines and, thus, bear *indirect moral responsibility* for those agents' actions [5]. This further motivates taking individual alignment seriously, not only as a practical or technical challenge, but as a morally significant endeavor.

This suggests that the question of *who determines alignment* is orthogonal to the distinction between execution and outcome. Suppose Peter approves of Adam threatening a shift manager to secure the reservation. Then Adam may be both outcome- and execution-aligned with Peter's *individual* expectations, while still being misaligned with respect to *collective preference outcome alignment* (others may prefer someone else getting the table) and in serious violation of *collective ethical execution alignment*.

So, while individual alignment (sometimes called *personalized alignment*, cf. [6]) deserves to be taken seriously in its own right and with respect to various normative alignment aims, lifting alignment from the individual to the collective level becomes necessary when pursuing other alignment aims. This lifting, however, raises further questions—questions that are heavily discussed in current alignment research (cf. [18]).

[12] In this regard, individual alignment connects to what have been called the desiderata of explainable AI; cf. [15].

Taken together, we now have three structural dimensions of alignment: *aim*, *constituency*, and *scope*. We propose that these dimensions should be treated as orthogonal and made explicit in alignment research and related discussions. Importantly, what counts as a suitable *goal* of alignment may depend on the specific alignment challenge at hand—and that, in turn, is shaped crucially by how these three dimensions are configured in a given case.

Most importantly, however, we believe that only if these structural dimensions of AI alignment are understood well and considered holistically can we begin to understand how AI alignment in an all-things-considered way could be achieved—if it is possible at all.

3.3 Application: Ethical Alignment and Machine Ethics

We now zoom in on one particular kind of alignment that is both practically urgent and conceptually rich: *ethical alignment*. Having laid out the structural dimensions of alignment in general, we explore how ethicality fits into that framework and what role Machine Ethics (ME) plays in achieving alignment with moral standards in a robust and operationalizable way.

Ethical alignment can now be clearly located within our taxonomy. It is best characterized as:

Normative Aim: Ethicality (according to some moral standard X)
Constituency: Collective
Scope: Execution *and* outcome

In this sense, an ethically aligned AIA is one that both produces ethically acceptable behavior (including the underlying decision-making process) and achieves ethically acceptable outcomes (and no others). We suggest, thus, that ethical alignment is not just about what the system does, but also about how and why it does it.

Definition 14 (Ethical Alignment). *An AIA is ethically aligned if and only if it is sufficiently aligned—across both outcome and execution dimensions—with respect to a morally justifiable standard X, as assessed from a collective perspective.*

While the literature on the normative part of AI alignment typically focuses on how to identify suitable norms—whether by choosing a particular ethical theory, or by appealing to procedures of public justification as discussed above—*Machine Ethics* (ME) concerns a distinct challenge: how to build these norms *into* artificial agents in a manner that makes them action-guiding in morally appropriate ways.

More specifically, ME addresses ethical *execution alignment*—ensuring that agents act for the right reasons, not just in ways that happen to lead to acceptable outcomes. This is especially relevant for RL-based agents, which learn action policies through interaction and reward signals, but typically lack mechanisms for representing or evaluating normative considerations.

Traditional ME research seeks to make such considerations an explicit part of the decision architecture itself (cf. [2,20]), something that is particularly hard in the context of RL agents. Recent work, however, proposes architectures that, for instance, couple normative reasons to action selection [4]. These reason-sensitive architectures provide a promising foundation for ensuring that agents not only behave ethically but do so in an explicit way that reflects transparent, deliberative moral reasoning.

ME understood in this way—with an emphasis on explicit moral deliberation—stands in contrast to what might be called *implicit ethical alignment*, where agents learn acceptable behavior via reward tuning or fine-tuning on curated data (cf. [19]). While such methods can succeed in narrow or well-controlled domains, they are fragile in the face of novelty, ambiguity, or conflict. As Arvan recently argued, aligning AI systems that lack the capacity for transparent moral reasoning in a way that is both empirically verifiable and ethically robust may be impossible in principle, due to deep epistemic limitations inherent to the nature of purely empirical testing [3].

We thus suggest that in contexts where ethical robustness and generalizability are essential, explicit ME architectures are not just beneficial—they are a necessary component of ethical alignment for RL-based AI systems.

4 Conclusion

Ensuring that AIAs behave permissibly across diverse normative expectations is an increasingly pressing interdisciplinary challenge. This paper has clarified conceptual distinctions central to AI alignment, explicitly distinguishing safety, ethicality, and broader alignment dimensions. We argued that ethical alignment—collectively aligning AIAs with morally justifiable standards—requires explicit normative deliberation mechanisms provided by Machine Ethics, going beyond mere implicit alignment strategies.

Our taxonomy explicitly differentiated between alignment goals, normative aims, scope (outcome vs. execution), constituency (individual vs. collective), and alignment degrees. Together, these distinctions clarify alignment's structural complexity and practical implications, highlighting that alignment is inherently multidimensional and context-sensitive.

We further emphasized the fundamental challenge of achieving *all-things-considered alignment*, which involves navigating conflicts among legitimate normative demands (e.g., morality, user intent, legality). Although a definitive theoretical resolution may remain elusive, developing politically legitimate and practically workable alignment frameworks is an urgent issue.

Future research should operationalize these conceptual insights through concrete normative theorizing, technical architectures, institutional frameworks, and public governance mechanisms. We hope this conceptual groundwork contributes meaningfully to developing AI systems that are not only powerful and efficient but also comprehensively normatively aligned.

Acknowledgments and Disclosure of Funding. This work is partially funded by the *European Regional Development Fund* (ERDF) and the Saarland within the scope of the (To)CERTAIN project as part of the *Center for European Research in Trusted* Artificial Intelligence (CERTAIN) as well as by the German Research Foundation DFG in the project 389792660 as part of the Transregional Collaborative Research Center TRR 248 – *Foundations of Perspicuous Software Systems* (CPEC) and received support from the *German Federal Ministry of Education and Research* (BMBF) as part of the project *MAC-MERLin* (Grant Agreement No. 01IW24007).

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Amodei, D., Olah, C., Steinhardt, J., Christiano, P., Schulman, J., Mané, D.: Concrete Problems in AI Safety (2016). https://arxiv.org/abs/1606.06565
2. Anderson, M., Anderson, S.L.: Machine Ethics. Cambridge University Press, Cambridge (2011)
3. Arvan, M.: "Interpretability" and "alignment" are fool's errands: a proof that controlling misaligned large language models is the best anyone can hope for. AI and Society (2024). https://doi.org/10.1007/s00146-024-02113-9
4. Baum, K.: Reason-sensitive artificial agents: on the need for justificatory alignment. arXiv preprint arXiv:2407.02425 (2024)
5. Baum, K., Mantel, S., Schmidt, E., Speith, T.: From responsibility to reason-giving explainable artificial intelligence. Philos. Technol. **35**(1), 1–30 (2022). https://doi.org/10.1007/s13347-022-00510-w
6. Bhat, S., Lyons, J.B., Shi, C., Yang, X.J.: Evaluating the impact of personalized value alignment in human-robot interaction: insights into trust and team performance outcomes. In: Proceedings of the 2024 ACM/IEEE International Conference on Human-Robot Interaction. HRI 2024, pp. 32–41. Association for Computing Machinery, New York (2024). https://doi.org/10.1145/3610977.3634921, https://doi.org/10.1145/3610977.3634921
7. Case, S.: Normative pluralism worthy of the name is false. J. Ethics Soc. Philos. **11**(1), 1–20 (2016). https://doi.org/10.26556/jesp.v11i1.107
8. Dancy, J.: What do reasons do? South. J. Philos. **41**(S1), 95–113 (2003). https://doi.org/10.1093/acprof:oso/9780199269914.003.0003
9. Davidson, D.: How is weakness of the will possible? In: Feinberg, J. (ed.) Moral Concepts. Oxford University Press (1969)
10. European Parliament and Council of the EU: Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence and amending Regulations (EC) No 300/2008, (EU) No 167/2013, (EU) No 168/2013, (EU) 2018/858, (EU) 2018/1139 and (EU) 2019/2144 and Directives 2014/90/EU, (EU) 2016/797 and (EU) 2020/1828 (Artificial Intelligence Act) (2024). https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=OJ:L_202401689
11. Gabriel, I.: Artificial intelligence, values, and alignment. Mind. Mach. **30**(3), 411–437 (2020). https://doi.org/10.1007/s11023-020-09539-2
12. Gabriel, I., Keeling, G.: Value alignment: the normative and technical challenges. Philos. Stud. (2025). https://doi.org/10.1007/s11098-025-02300-4

13. Gu, S., et al.: A review of safe reinforcement learning: methods, theories, and applications. IEEE Trans. Pattern Anal. Mach. Intell. **46**(12), 11216–11235 (2024). https://doi.org/10.1109/TPAMI.2024.3457538
14. Hintersdorf, D., Struppek, L., Kersting, K.: Balancing Transparency and Risk: The Security and Privacy Risks of Open-Source Machine Learning Models. arXiv preprint arXiv:2308.09490 (2023). https://doi.org/10.48550/arXiv.2308.09490
15. Langer, M., et al.: What do we want from explainable artificial intelligence (XAI)?-a stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. Artif. Intell. **296**, 103473 (2021). https://doi.org/10.1016/j.artint.2021.103473
16. MacAskill, M., Bykvist, K., Ord, T.: Moral Uncertainty. Oxford University Press, Oxford (2020)
17. Scanlon, T. (ed.): What We Owe to Each Other. Harvard University Press, Cambridge (1998)
18. Shen, H., et al.: Towards bidirectional human-AI alignment: a systematic review for clarifications, framework, and future directions (2024). https://arxiv.org/abs/2406.09264
19. Vishwanath, A., Dennis, L.A., Slavkovik, M.: Reinforcement learning and machine ethics: a systematic review (2024). https://arxiv.org/abs/2407.02425
20. Wallach, W., Allen, C.: Moral Machines: Teaching Robots Right from Wrong. Oxford University Press, Oxford (2009). https://doi.org/10.1093/acprof:oso/9780195374049.001.0001
21. Wiener, N.: Some moral and technical consequences of automation. Science **131**(3410), 1355–1358 (1960). https://doi.org/10.1126/science.131.3410.1355
22. Wolf, S.: Moral saints. J. Philos. **79**(8), 419–439 (1982). https://doi.org/10.2307/2026228
23. Zanotti, G., Chiffi, D., Schiaffonati, V.: AI-related risk and uncertainty. In: Bridging the Gap Between AI and Reality: First International Conference, AISoLA 2023, Crete, Greece, October 23–28, 2023, Selected Papers, pp. 284–292. Springer, Heidelberg (2023). https://doi.org/10.1007/978-3-031-73741-1_17

Epistemic Deference to AI

Benjamin Lange[1,2(✉)]

[1] Ludwig-Maximilians-Universität München, Geschwister-Scholl Platz 1, 80539 München, Germany
benjamin.lange@lmu.de

[2] Munich Center for Machine Learning, Geschwister-Scholl Platz 1, 80539 München, Germany

Abstract. When should we defer to AI outputs over human expert judgment? Drawing on recent work in social epistemology, I motivate the idea that some AI systems qualify as Artificial Epistemic Authorities (AEAs) due to their demonstrated reliability and epistemic superiority. I then introduce AI Preemptionism, the view that AEA outputs should *replace* rather than supplement a user's independent epistemic reasons. I show that classic objections to preemptionism – such as uncritical deference, epistemic entrenchment, and unhinging epistemic bases – apply in *amplified* form to AEAs, given their opacity, self-reinforcing authority, and lack of epistemic failure markers. Against this, I develop a more promising alternative: a total evidence view of AI deference. According to this view, AEA outputs should function as *contributory* reasons rather than outright replacements for a user's independent epistemic considerations. This approach has three key advantages: *(i)* it mitigates expertise atrophy by keeping human users engaged, *(ii)* it provides an epistemic case for meaningful human oversight and control, and *(iii)* it explains the justified mistrust of AI when reliability conditions are unmet. While demanding in practice, this account offers a principled way to determine when AI deference is justified, particularly in high-stakes contexts requiring rigorous reliability.

Keywords: Epistemic Authorities · ; Deference to AI · Social Epistemology

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 174–186, 2026.
https://doi.org/10.1007/978-3-032-01377-4_9

1 Introduction

AI systems[1,2] increasingly outperform human experts.[3] For example, in medicine, AI systems can analyse images such as X-rays, MRIs, and CT scans more quickly and, in some cases, more accurately than doctors, detecting diseases like cancer at earlier stages and with greater precision.[4] Similarly, in finance, AI-driven trading algorithms execute high-frequency trades, optimize portfolio management, and forecast market trends with a level of speed and data-processing capability that surpasses human traders.[5]

As these AI systems continue to improve, a central question becomes: when should we *rationally* defer to AI's recommendations, particularly when they are more accurate than those of human expert authorities?

This question is not merely theoretical but has significant real-world consequences. Blind deference to AI could lead to over-reliance on systems that may harbour biases or hidden errors or that might compromise important epistemic virtues.[6] Conversely, irrational scepticism towards AI could lead to rejecting *better* judgments in favour of less reliable human reasoning. Striking the right balance requires a principled approach to *epistemic deference to AI*.[7]

In this paper, I develop a total evidence account of AI deference. According to this view, AI outputs should function as what philosophers call *contributory reasons* rather than preemptive replacements for human judgment. This approach situates the

[1] I use the term 'AI' roughly for any machine or computer system capable of doing things that normally require intelligence and reflection – thinking, learning, or problem solving – when done by humans. See also Hasan et al. (2022, pp. 1–2).

[2] I focus on *epistemic* AI systems (see Hauswald, Sect. 1, forthcoming), which perform tasks previously reserved for human experts and epistemic authorities. Unlike AI designed for practical problem-solving, epistemic AI systems function primarily as sources of information or epistemic instruments, providing outputs that convey accessible worldly information (see Goldberg 2020, p. 2785).

[3] According to Maslej et al.'s (2024) *AI Index Report 2024*, AI has surpassed human experts in tasks such as image classification, visual reasoning, language understanding, medical diagnosis, algorithmic efficiency, and materials discovery. However, it still struggles with complex reasoning, competition-level mathematics, and strategic planning.

[4] See Liu et al. (2019) for a comparison of AI and healthcare professionals in disease detection, Lebovitz et al (2021) for a critique of AI training and evaluation based on expert knowledge, and Han et al. (2024) for a review of AI in clinical practice. For a recent critical review of claims about AI outperforming human experts, see Drogt et al. (2024).

[5] For example, AI-driven algorithmic trading systems adjust to market information more quickly and may generate higher profits around news announcements due to their superior market timing ability and rapid execution (Bahoo et al., 2024; Frino et al., 2017).

[6] There are different questions about epistemic authority and deference. My focus here is the *deference question*: how we should rationally assign special epistemic weight to the views of an epistemic authority (once it is recognized). Other questions concern the *i)* nature and *ii)* identification of epistemic authority and *iii)* the transmission of epistemic goods (Jäger, forthcoming).

[7] The literature on epistemic deference to AI remains sparse, with Wolkenstein (2024) and Hauswald (2025; forthcoming) being among the few notable contributions that explicitly draw on social epistemology.

debate about when to rely on AI outputs in expert domains in the context of recent work in social epistemology, particularly discussions on deference to epistemic authority and expert testimony. I examine the implications of a total evidence account of AI deference, showing that it has three key advantages that align with existing concerns about the ethics and responsible use of AI: *(i)* it mitigates expertise atrophy by keeping human users epistemically engaged, *(ii)* it provides an *epistemic* rather than purely moral rationale for meaningful human oversight and control, and *(iii)* it explains justified epistemic mistrust of AI.

The article proceeds as follows. Section 2 formulates and motivates the case for Artificial Epistemic Authorities (AEAs) and AI preemptionism. I show that classic objections to Preemptionism – such as uncritical deference, epistemic entrenchment, and unhinging epistemic bases – apply in *amplified* form to AEAs, given their opacity, self-reinforcing authority, and lack of failure markers. Against these shortcomings, Sect. 3 then proposes my total evidence account of AI deference, motivating its rationale and theoretic advantages. I show how this account can overcome some of the objections raised against AI preemptionism. Section 4 concludes by the implications of this account for current practice and usage of AI systems.

2 Artificial Epistemic Authorities and AI Preemptionism

We rely on others' expertise all the time. We trust doctors to diagnose and treat our illnesses, lawyers to interpret complex legal statutes, and auditors to ensure financial accuracy and compliance – often without understanding most or even any of the exact reasons and details. In doing so, we grant these experts a special kind of authority, recognising that their specialised knowledge places them in a superior epistemic position relative to us.[8]

What qualifies someone – or something – as an epistemic authority? According to *objectivist*[9] *belief-based views*, an epistemic authority must possess beliefs and the ability to communicate reasons for their judgments (e.g. Goldman, 2018). According to *objectivist functionalist views*, an epistemic authority must possess epistemic superiority such as being in a better position to provide reliable knowledge (e.g. Hauswald, 2024).

A promising definition of epistemic authority is provided by Jäger (2024):

Epistemic Authority[10]**:** A is a (recognised) epistemic authority for person S in domain D at t and relative to epistemic goal G, if S truly believes A to be able, and

[8] What is the relationship between expertise and epistemic authority? 'Expertise' and 'epistemic authority' need not be synonymous, as one may have authority without expertise and vice versa (Jäger, forthcoming). I here assume that epistemic authority should be conceived *relationally* and not in terms of an absolute threshold of epistemic knowledge. Experts, on the other hand, must cross an absolute epistemic threshold. They must be superior to most other people in the community with respect to D (Goldman, 2018).

[9] *Objectivist* accounts make a claim about the *de facto* epistemic conditions that an EA must fulfil. *Subjectivist* accounts maintain that an agent must *believe* or *perceive* a putative epistemic authority in an epistemically superior position.

[10] This is the most encompassing hybrid definition offered by Jäger (forthcoming), which requires actual epistemic superiority, recognition by a layperson, and a functional component that helps the layperson achieve their epistemic aims.

prepared, to help S achieving S's epistemic goals in D and with respect to G, where this ability is due to A *de facto* being in a substantially advanced epistemic position in D, relative to S, at or during t, and with respect to the goal of acquiring G.

This view holds that A is an epistemic authority (EA) when a person S correctly believes that A has superior knowledge in a specific domain D at a given time t, relative to an epistemic goal G such as gaining knowledge or forming true beliefs. This means A must not only possess epistemic superiority but also be capable and willing to help S achieve their goal. For example, a patient might defer to a doctor for a diagnosis because they recognize the doctor's medical expertise.

Given the above definition, we can ask whether *artificial* epistemic authorities (AEAs) could exist: AI systems that play a similar epistemic role to human epistemic authorities. There are good reasons to think that some AI systems might qualify as AEAs (Hauswald, 2025) – especially if we entertain the idea that epistemic authority concerns superior epistemic positioning and reliability rather than necessarily beliefs and communicative intent.[11]

First, many AI systems already provide helpful information, explanations, and reliable advice, functioning in ways that closely resemble traditional human EAs. From advanced medical diagnostic algorithms to financial risk analysis tools, AI is increasingly filling roles that require specialised epistemic expertise. Second, some AI systems hold superior epistemic positions in specific areas of applications. In such cases, AI systems do not merely replicate but arguably *surpass* human expertise, producing outputs that consistently reduce false positives and false negatives beyond human capability.[12] Third, the epistemic asymmetry between AI systems and human users in certain domains mirrors the traditional expert-layperson divide. For example, machine learning models trained on vast datasets can detect patterns and correlations that no human could reasonably discern, giving them an epistemic advantage akin to that of human experts over laypeople. Third, like human epistemic authorities, AI systems often exhibit epistemic opacity (Ross, 2024), so their reasoning processes, particularly in deep learning models, can be difficult or impossible for non-experts to fully grasp, even when they are highly reliable.[13]

If at least some AI systems qualify as AEAs, how ought we to respond to their outputs? In the case of human EAs, some have suggested that we should treat their judgments as *preempting* our own independent reasoning (Zagzebski, 2012; 2013; 2014; 2016).[14] Should we adopt a similar stance toward AEAs?

[11] Some might be inclined to resist the idea of AEAs precisely because they can (currently) not meet criteria of beliefs and communicative intent. I am inclined to draw the opposite conclusion: because it seems intuitively compelling that AI-systems already serve as epistemic authorities, this provides a reason to reject these belief-based accounts.

[12] Maslej et al. (2024), chs. 2 and 5. See also footnote 3.

[13] For an insightful analysis about the tenuous relationship between transparency and expertise, see Nguyen C.T (2020).

[14] In addition to Zagyebski, Keren (2007, 2014a, 2014b) and Constantin and Grundmann (2018) have also defended forms of preemptionism for human EAs.

One answer is *AI Preemptionism*, which holds that when an AEA is a reliable truth-tracker, we should not merely add its outputs to our independent reasoning but replace our own reasons with its verdicts.

We can define this view more formally as

AI Preemptionism: User U should treat the output O of a recognised artificial epistemic authority AEA in domain D at time t as a pre-emptive reason for believing proposition p, such that O replaces, rather than merely adds to, U's independent epistemic reasons for or against p.

AI Preemptionism – like traditional Preemptionism – can be motivated through the so-called 'track-record' argument (Raz, 2009; Zagzebski, 2012). The basic idea is that if a putative epistemic authority reliably outperforms you, your best route to arrive at a correct view is to align with the epistemic authority who surpasses you in reliability and replace your own reasons with the authority's. Any 'half-measure' like adding or balancing your own evidence with the authority's output risks lowering overall accuracy in arriving at a true belief, which would be epistemically irresponsible.

Preemptionism has great appeal. It provides a clear and structured approach to epistemic deference which ensures individuals align their beliefs with the most reliable sources of knowledge. This is particularly compelling in cases where a layperson has no prior beliefs or reasons to rely upon or has conflicting beliefs or reasons that make an independent judgment difficult. In such situations, deferring entirely to an epistemic authority, rather than attempting to independently weigh competing evidence, maximises the chances of acquiring true beliefs while minimising the risk of getting it wrong.

The appeal of AI Preemptionism follows the same rationale but seems especially attractive because the AEA's potential for accuracy and reliability is so high. Unlike human experts, some AI-systems in narrow domains of application lack human-cognitive biases, fatigue, and other human cognitive inconsistencies, making its outputs arguably more precise and reproducible.[15] Moreover, AI systems can improve and self-refine their accuracy much more quickly than a human epistemic authority, whose expertise is only built up slowly over many years. AEAs also operates without social or economic pressures, potentially reducing conflicts of interest common among human epistemic authorities.

Though AI Preemptionism has an especially strong appeal, it inherits the classic objections to Preemptionism (Jäger, forthcoming). Indeed, I think that these objections are amplified in certain cases.

First, a standard objection to Preemptionism is that epistemic authorities are neither omniscient nor infallible and may therefore fail to consider all the reasons available to a non-expert.[16] This can lead to epistemic *losses* rather than gains. AEAs give this concern an additional problematic twist: the opacity of AI systems' reasoning, coupled with their empirical track record of superior performance in many domains, may risk systematically reinforcing *mistaken perceptions* of infallibility even when the AI lacks

[15] Matters will depend on the specific AI-system in question. For example, insofar as current LLMs might qualify as AEAs in expert domains, suffer from other outputs bias compared to human EAs such as answer option variance in multiple-choice questions based on the position of the answer options (Pezeshkpour et al., 2023).

[16] See also Constantin and Grundmann (2020) and Grundmann (forthcoming).

full epistemic access to all relevant considerations. Yet, due to their general statistical and predictive success, users may assume that the AI has already taken their own reasons into account, leading to a *stronger* epistemic pre-emption than would occur with a human authority. Thus, the concern is that AI Preemptionism strengthens the risk of uncritical deference, not merely because AI systems are epistemically superior in some respects but because they will always appear to be unquestionably so. This AI infallibility bias can leave users in a worse epistemic position than if they were deferring to a human expert, whose epistemic fallibility remains comparatively more detectable.

A second objection to Preemptionism is that it can entrench individuals within irrational or epistemically corrupt communities, making it difficult for them to critically reassess misguided authorities whom they have been conditioned to trust (Lackey, 2018). This concern is even more acute regarding AEAs, as their epistemic authority can be systemic, automated, and self-reinforcing in ways that human authority is not. With human EAs, individuals often retain some capacity to challenge, interrogate, or disengage from a community's epistemic standards by identifying explicit biases, inconsistencies, or other irrational influences. However, AEAs can operate within algorithmically structured epistemic ecosystems that dynamically shape what users see, believe, and engage with, often in ways that are invisible to the users. This makes epistemic entrenchment more subtle. Also, unlike human EAs, whose reliability can be questioned based on incoherence, logical errors, or failures to adherence to established norms, standards, or methodologies within a given field of expertise, AEAs lack these clear epistemic failure markers. They do not exhibit hesitation, self-doubt, or accessible reasoning processes, which means that users embedded within AI-curated knowledge ecosystems may find it nearly impossible to recognise when their epistemic environment is corrupted. The danger is thus not merely that AI can reinforce irrationality but that it does so with a veneer of objectivity that is especially difficult to remove.

Third, Preemptionism has been criticised for requiring individuals to unhinge their own epistemic bases and replace them entirely with the authority's belief, which can lead to epistemic regress rather than progress (Jäger 2016; Dormandy 2018, Keren 2020). This concern is even more severe for AEAs. Unlike human EAs, current AI-systems do not provide reasons in the same way that humans do. Human EAs, even when opaque about specific justifications, operate within a shared epistemic framework that allows for some degree of reconstruction of their reasoning; by contrast, many AI systems, particularly those based on black box models, do not. However, if an AI system provides no explicit reasons for its belief but merely returns an output, then agents are expected to abandon their own epistemic bases without replacement. This makes the problem of epistemic regress even more acute. In medical AI, for instance, a doctor who pre-emptively defers to an AI's diagnostic decision might do so without understanding the features the AI weighed most heavily. If the doctor then ceases to consider their own medical reasoning in favour of the AI's opaque verdict, their epistemic base is not merely adjusted but entirely severed, leading to epistemic dependence without epistemic understanding.

These objections suggest that while AI Preemptionism might initially appear compelling – especially given AI's potential for superior reliability – it ultimately inherits and amplifies the classic problems associated with epistemic Preemptionism. AI systems, despite their computational advantages, are not epistemically infallible, and their

authority can be dangerously opaque, systematically biased, and self-reinforcing in ways that human EA is not.

3 Total Evidence AI Deference Account

3.1 Total Evidence AI Deference

While AI Preemptionism offers a clear and structured approach to epistemic deference to AI, the concerns raised in the previous section highlight its significant limitations.

Total evidence views of epistemic authority may offer a promising alternative (Dougherty, 2014; Jäger, 2016; Lackey, 2018; Dormandy 2018).[17] Rather than advocating for complete pre-emption, total evidence views of epistemic authority hold that while authoritative testimony should carry significant epistemic weight, it should not completely replace an agent's independent reasons but rather be integrated as a *contributory reason* with an agent's own epistemic resources, allowing for aggregation rather than outright replacement of reasons.

Accordingly, we can define the AEA-relevant analogue of this view as follows:

Total Evidence AI Deference: User U should treat the output O of a recognised artificial epistemic authority AEA in domain D at time t as a contributory reason for believing proposition p, such that O is integrated into, rather than replacing, U's independent epistemic reasons for or against p.

This definition contrasts with AI Preemptionism by allowing U to retain and weigh their own epistemic reasons rather than fully substituting them with AEA's output.[18]

There are different ways to spell out a total evidence view in detail. I here want to focus on a rendition that answers the concerns that emerged in the previous section. These can be formulated in terms of three desiderata.

First, a total evidence view of AI deference should acknowledge that while AI systems can serve as epistemic authorities, their reliability must be continuously assessed rather than assumed, to ensure that deference does not rest on an unwarranted assumption of infallibility. By preventing users from blindly trusting AI outputs simply because of their statistical success or predictive accuracy, this guards against uncritical deference. Second, the account must prevent epistemic entrenchment by ensuring that deference to AI remains an active, reflective process rather than a passive or self-reinforcing dependence. Given that AI systems can structure epistemic environments and reinforce specific viewpoints in ways that are difficult to challenge, users must retain the ability to critically assess AI-generated beliefs and recognise when deference may be leading them into an epistemically corrupt community. Third, the account should preserve independent epistemic bases by integrating AI-generated outputs with human reasoning to allow for epistemic progress.

We can formulate an account that meets these desiderate more precisely as:

Total Evidence View of AI Deference.

[17] See also Bokros (2021) for an interesting deference model in support of total evidence views based on formal accuracy-first epistemology.

[18] For a similar argument in favour of using human second opinions to resolve disagreements between AI and human experts, see also Kempt and Nagel (2021).

Critical Deference with Oversight: Assume U is faced with the decision to belief p or non-p in domain D, then

1. U withholds of revisits deference to AEA if
   i. Domain Mismatch: p lies outside A's validated domain.
   ii. Reliability Undermining: Evidence suggests that A exhibits systematic bias or recurring errors.
   iii. Conflicting Authority: A comparably reliable human or AI EA disagrees.
   iv. Novel Evidence: U holds independent reasons that it is plausible that A did not consider.
2. Otherwise, U defers to AEA.

To illustrate, this accountant maintains that, by default, users should strongly defer to AEAs when it has a well-established epistemic advantage, but this deference remains *conditional* and *defeasible* rather than automatic. Unlike strict AI Preemptionism, which demands that users replace their own reasons with the authority's judgment, this model treats AI as an authoritative yet fallible epistemic agent whose outputs should be integrated into, rather than wholly substituted for, an agent's reasoning. The account thereby ensures that recognition of epistemic authority is based on demonstrated reliability, not on an uncritical assumption of infallibility. This means that while users should give significant epistemic weight to AI recommendations, they must also remain attuned to potential gaps in the AI's reasoning, limitations in its training data, or structural biases embedded in its design.

Moreover, this account prevents epistemic entrenchment by ensuring that AI deference remains open to revision. If users defer to AI uncritically, they risk being passively shaped by AI-driven knowledge ecosystems, making it difficult to recognise alternative perspectives, challenge embedded biases, or identify systematic epistemic distortions. To counter this, the model incorporates defeaters[19] and overrides that allow users to withdraw deference when clear signs of systematic bias, domain mismatch, or contradictory epistemic authority testimony arise.

Finally, this account also ensures that users do not become severed from their own epistemic bases. The account presented here avoids this by allowing for partial aggregation rather than full replacement. If a user identifies relevant, independent reasons that the AI might not have considered, those reasons can be reintroduced into the evaluation process, either by prompting the AI for an updated assessment or by weighing the user's own reasoning against the AI's conclusion. This ensures that epistemic progress is made rather than lost, maintaining a user's active role in belief formation rather than reducing them to a passive recipient of AI outputs.

[19] These can be more precisely understood as *rebutting defeaters*, which provide counterevidence, or *undercutting defeaters*, which question the reliability of the evidence itself. This contrasts with Constantin and Grundmann's (2018) *source-sensitive defeaters* which are specific to the credibility of the source of one's evidence.

3.2 Human Expertise Atrophy

One worry about deferring to AI is that human experts – doctors, lawyers, engineers – may gradually lose their domain-specific competences.[20] If a specialist consistently defers to an AI's outputs, they might become a passive conduit for algorithmic decisions rather than an active decision-maker, ultimately losing the skill to check or challenge the AI's recommendations. This risk of 'expertise atrophy' could hence render practitioners less capable of responding effectively if the AI performs poorly or faces unexpected scenarios outside its training scope.

The AI Deference Account helps mitigate this concern by demanding a *partial integration* of human reasoning and AI outputs, rather than total replacement. Since deference is *defeasible*, human experts must remain sufficiently active in the process to identify potential defeaters, domain mismatches, or updated evidence that the AI system may not have factored in. In practice, this encourages continuous skill development and preserves a meaningful epistemic role for humans, ensuring that domain knowledge and critical thinking do not wither away. Rather than undermining expertise, the AI Deference Account therefore suggests a dynamic in which AI complements rather than supplants human skill.

3.3 Meaningful Human Control

A second advantage of the AI Deference Account is that it provides an *epistemic* case for maintaining human control, independent of any moral or political values. While normative arguments for human oversight emphasise accountability, responsibility, and the preservation of autonomy (Amoros et al., 2020), the proposed accounts highlight the value of human input in identifying epistemic defeaters. Because the AI's recommendations are only *default-accepted* under conditions of demonstrated reliability, situations can and do arise where its authority is overridden, which necessitates a human arbiter to resolve conflicts, domain errors, or contradictory signals.

This requirement dovetails with existing accounts for responsible AI governance that advocate human oversight or 'meaningful human control' (see Santonio et al., 2018).[21] By explicitly incorporating checkpoints for re-evaluation, the AI Deference Account ensures that human users retain ultimate epistemic responsibility and are poised to step in whenever evidence emerges that the AI is mistaken or operating outside its domain. This not only protects against overreliance on automated systems but also aligns with regulatory and organisational guidelines promoting transparency and accountability in AI deployment.

[20] See Tai (2020) who discusses how automation leads to skill degradation and emphasizes the need for human oversight to prevent dependence on AI. Messeri and Crockett (2024) warn about "illusions of understanding", where human experts falsely believe they still possess expertise despite AI taking over complex tasks.

[21] On meaningful-human control (MHC), see Amoros et al. (2020), Verdiesen et al. (2021), and Mecacci et al. (2024). See Mosqueira-Rey et al. (2023) on human-in-the-loop approaches.

3.4 Lack of Epistemic Trust

A third benefit of the AI Deference Account is that it sheds light on why some users epistemically distrust certain AI systems.[22] According to this view, the rationality of trust in AI is grounded in the AI's capacity to meet the conditions for recognised epistemic authority (demonstrated reliability, domain alignment, transparency about its limitations, etc.). If these conditions remain unmet – for instance, due to opaque training data, a poor track record, or repeated biases – then it is epistemically *warranted* for users to withhold deference and maintain scepticism about the system's outputs.[23]

This account clarifies that user mistrust can be reasonable from a purely epistemic perspective, rather than constituting a knee-jerk rejection to technological innovation or status quo bias. If the AI cannot demonstrate that it holds a reliably superior vantage point in the relevant domain, or if it fails to accommodate defeaters that a competent human recognises, then deferring to it becomes irrational. The proposed account thereby validates and explains users' reluctance when conditions for justified deference are not met, providing a structured epistemic rationale for distinguishing between justified acceptance of AI outputs and rightful epistemic mistrust.

4 Implications for Current Use of AI

An implication of this account is that *few* current AI systems will easily satisfy the requirements needed to qualify as Artificial Epistemic Authorities (AEAs). At a minimum, systems must demonstrate sustained accuracy in their domain, provide a mechanism for ongoing reliability checks, and afford users the means to detect when defeater conditions arise. Meeting these benchmarks may demand substantial effort: designing interpretable architectures, collecting extensive domain-specific data that is continuously updated, and implementing transparent reporting of potential errors or biases. Some might consider these requirements unrealistic for large-scale or commercial AI deployments, especially under market pressures to release products quickly and cost-effectively. Nonetheless, where safety or high-stakes decision-making is crucial – as in medical diagnostics, finance, or aerospace – implementing these more stringent standards is both ethically *and epistemically* warranted.

Despite these hurdles, certain real-world AI use cases *do* already approximate the conditions for justified deference. Narrowly focused medical imaging AIs, for instance, can come close when they undergo rigorous clinical validation, operate within a well-defined domain, and offer clinicians a structure for second-guessing anomalous results. Similarly, some specialised AI systems in finance or engineering are subject to continuous audit and paired with expert human oversight, creating an environment in which defeaters can be systematically tracked.

In conclusion, I have examined whether and how we might justifiably defer to AI outputs rather than relying on our own or others' human judgment. I began by outlining AI Preemptionism, according to which an AI's recommendation *replaces* all of a

[22] Drawing on Wilholt's (2013) definition, epistemic trust refers to trust in AI strictly in virtue of its ability to provide reliable information or enhance human knowledge (Alvarado, 2023).

[23] This aligns with Durán and Jongsma's (2021) argument that trust in AI does not necessarily require transparency but rather hinges on the reliability of its outputs.

user's existing reasons. Although Preemptionism appears compelling, classic objections to epistemic Preemptionism are amplified in AI contexts, given opaque algorithms, perceived infallibility, and the risk of reinforcing epistemically corrupt environments. To address these concerns, I proposed a total evidence view of AI Deference that treats AI outputs as *contributory* rather than *pre-emptive* reasons. By requiring ongoing reliability assessments, recognising defeaters, and retaining partial user agency, this approach avoids expertise atrophy, provides an epistemic rationale for human oversight, and explains users' justified distrust when AI systems fail to meet necessary epistemic conditions.

Acknowledgments. I thank audience members at the 2024 AISoLa conference as well as Frederic Gerdon, Sven Nyholm, and two anonymous referees and the editor from this journal for their feedback on earlier drafts of this paper. Special thanks to Johannes Reisinger for extensive comments.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

Alvarado, R.: What kind of trust does AI deserve, if any? AI and Ethics **3**, 1169–1183 (2023). https://doi.org/10.1007/s43681-022-00224-x

Amoroso, D., Tamburrini, G.: Autonomous weapons systems and meaningful human control: ethical and legal issues. Current Robot. Rep. **1**, 187–194 (2020)

Bahoo, S., et al.: Artificial intelligence in finance: a comprehensive review through bibliometric and content analysis. SN Bus. Econ. **4**, 23 (2024). https://doi.org/10.1007/s43546-023-00618-x

Bokros, S.E.: A deference model of epistemic authority. Synthese **198**, 12041–12069 (2021). https://doi.org/10.1007/s11229-020-02849-z

Constantin, J., Grundmann, T.: Epistemic authority: Preemption through source-sensitive defeat. Synthese (2020).https://doi.org/10.1007/s11229-018-01923-x

Croce, M.: Expert-oriented abilities vs. novice-oriented abilities: an alternative account of epistemic authority. Episteme **5**, 476–498 (2018)

Croce, M.: On what it takes to be an expert. Philos. Quart. **69**, 1–21 (2019)

Croce, M.: For a service conception of epistemic authority: a collective approach. Soc. Epistemol. **33**, 1–11 (2019)

Dormandy, K.: Epistemic authority – preemptive reasons or proper basing? Erkenntnis **83**, 773–791 (2018)

Drogt, J., et al.: Ethical guidance for reporting and evaluating claims of AI outperforming human doctors. NPJ Dig. Med. **7**, 271 (2024). https://doi.org/10.1038/s41746-024-01255-w

Durán, J.M., Jongsma, K.R.: Who is afraid of black box algorithms? On the epistemological and ethical basis of trust in medical AI. J. Med. Ethics **47**, 329–335 (2021). https://doi.org/10.1136/medethics-2020-106820

Frino, A., Prodromou, T., Wang, G.H., Westerholm, P.J., Zheng, H.: An empirical analysis of algorithmic trading around earnings announcements. Pac. Basin Finan. J. **45**, 34–51 (2017). https://doi.org/10.1016/j.pacfin.2016.05.008

Goldberg, S.C.: Epistemically engineered environments. Synthese **197**, 2783–2802 (2020)
Alvin,G.: Expertise. Topoi **37**(1), 3-10 (2018)
Grundmann, Thomas (forthcoming). Experts: What Are They and How Can Laypeople Identify Them?
Han, R., et al.: Randomised controlled trials evaluating artificial intelligence in clinical practice: a scoping review. Lancet Dig. Health **6**, e367–e373 (2024)
Hasan, A., et al.: Algorithmic bias and risk assessments: lessons from practice. DISO **1**, 14 (2022). https://doi.org/10.1007/s44206-022-00017-z
Hauswald, R.: Epistemische Autoritäten: Individuelle und plurale. Springer, Berlin (2024)
Hauswald, R.: Artificial epistemic authorities. Soc. Epistemol. ,1–10,(2025). https://doi.org/10.1080/02691728.2025.2449602
Rico, H.: Forthcoming. "AI and the philosophy of expertise and epistemic authority. In: A Companion to Applied Philosophy of AI, edited by Martin Hähnel and Regina Müller. Wiley, Hoboken
Jäger, C.: Epistemic authority, preemptive reasons, and understanding. Episteme **13**(2), 167–185 (2016)
Jäger, C.: False authorities. Acta Analytica **39**, 643–661 (2024)
Christoph, J.: (forthcoming). Intellectual authority and education. In: Eder, A.-M., Brössel, P., Grundmann, T. (eds.) The Epistemology of Expert Judgement, Routledge
Christoph, J.: (forthcoming). Epistemic authority. In: Lackey J., McGlynn A. (eds.) The Oxford Handbook of Social Epistemology. Oxford University Press, Oxford
Kempt, H., Nagel, S.K.: Responsibility, second opinions and peer-disagreement: ethical and epistemological challenges of using AI in clinical diagnostic contexts. J. Med. Ethics **48**, 222–229 (2022)
Keren, A.: Epistemic authority, testimony, and the transmission of knowledge. Episteme **4**(3), 368–381 (2007)
Keren, A.: Trust and belief: a preemptive reasons account. Synthese **191**(12), 2593–2615 (2014)
Keren, A.: Zagzebski on authority and preemption in the domain of belief. Eur. J. Philos. Relig. **4**, 61–76 (2014)
Lackey, J.: Learning from Words. Oxford University Press, New York (2008)
Lackey, J.: To preempt or not to preempt. Episteme **13**(4), 571–576 (2016)
Lackey, J.: Experts and Peer Disagreement. In: Benton, M., Hawthorne, J., Rabinowitz, D. (eds.) Knowledge, Belief, and God: New Insights in Religious Epistemology, pp. 228–245. Oxford University Press, Oxford (2018)
Lackey, J.: Preemption and the problem of the predatory expert. Philos. Top. **49**, 133–150 (2021)
Lebovitz, S., Levina, N., Lifshitz-Assaf, H.: Is AI ground truth really true? The dangers of training and evaluating AI tools based on experts' know-what. MIS Q. **45**, 1501–1525 (2021)
Nestor, M., et al.: Artificial intelligence index report 2023. arXiv preprint: arXiv:2310.03715 (2023)
Mecacci, G., et al.: Research Handbook on Meaningful Human Control of Artificial Intelligence Systems. Edward Elgar Publishing (2024)
Nguyen, C.T.: Transparency is surveillance. Philos. Phenomenol. Res. **105**, 331–361 (2022). https://doi.org/10.1111/phpr.12823
Pezeshkpour, P., Hruschka, E.: Large language models sensitivity to the order of options in multiple-choice questions. arXiv preprint: arXiv:2308.11483 (2023)
Ross, A.: AI and the expert; a blueprint for the ethical use of opaque AI. AI Soc. **39**(3), 925–936 (2024). https://doi.org/10.1007/s00146-022-01564-2
Zagzebski, L.T.: Epistemic Authority – A Theory of Trust, Authority, and Autonomy in Belief. Oxford University Press, Oxford (2012)

Zagzebski, L.T.: A defense of epistemic authority. Research Philosophica **90**, 293–306 (2013). repr. in id., Epistemic Values: Collected Papers in Epistemology, New York: Oxford University Press (2020)
Zagzebski, L.T.: Epistemic authority and its critics. Eur. J. Philos. Relig. **6**(2014), 169–187 (2014)
Zagzebski, L.T.: Replies to Christoph Jäger and Elizabeth Fricker. Episteme **13**(2), 187–194 (2016)

Responsibility Attribution for AI-Mediated Damages with Mechanistic Interpretability

Lena Kästner[1,2(✉)], Johann Cordes[2,3], and Herbert Zech[2,4]

[1] University of Bayreuth, Bayreuth, Germany
lena.kaestner@uni-bayreuth.de
[2] Weizenbaum Institute, Berlin, Germany
[3] European New School of Digital Studies, European University Viadrina, Frankfurt (Oder), Germany
[4] Humboldt Universität zu Berlin, Berlin, Germany

Abstract. Artificial Intelligence (AI) raises profound ethical, moral, and legal challenges for society. In this paper, we focus on the legal challenge to adequately attribute responsibility for AI-mediated damages. In law, responsibility is usually attributed on the basis that a (human) actor is taking actions which *cause* harm or damage to some subject or property. While this analysis seems straightforward, it remains unclear (a) what conception of causation it relies on, (b) in what way this can be applied to attribute responsibility when human actions rely on the use of opaque AI systems, and (c) how liability for AI-mediated damages should be handled in practice. The current paper sets out to answer these questions. We shall argue that causation in the legal context is best conceptualized as difference-making. To determine relevant difference-makers for AI-mediated damages, we propose that explainable AI (XAI) methods may serve as important tools. Specifically, we argue, mechanistic interpretability (MI) is well-suited to increase the ex-ante safety of opaque AI systems.

Keywords: Artificial Intelligence (AI) · Responsibility Attribution · Liability · Explainable AI (XAI) · EU AI Act · Ex Ante Safety · Mechanistic Interpretability

1 Introduction

Artificial intelligence (AI)—machine learning (ML) based systems in particular—is pervasive in modern lives. This development not only raises technological challenges but also social, epistemological, ethical, and legal ones.

Work on this paper has been funded by the Volkswagen Foundation grants AZ 98509 and AZ 9B830 "Explainable Intelligent Systems" (EIS) and by the Federal Ministry of Education and Research of Germany (BMBF) (grant no.: 16DII131 "Weizenbaum-Institut"). We are indebted to members of the EIS research group and the audience at AISOLA 2024 for feedback on earlier versions of this paper.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 187–202, 2026.
https://doi.org/10.1007/978-3-032-01377-4_10

Among the most pressing desiderata is that while we want to benefit from the potentials of AI, we do not want to sacrifice existing moral and legal norms.

For the purposes of this paper, we focus on legal norms in Europe. We are specifically interested in the ability to adequately attribute legal responsibility for damages in situations where AI systems are being used. By AI systems we mean, following the current version of the AI Act, any "machine-based system that is designed to operate with varying levels of autonomy and that may exhibit adaptiveness after deployment, and that, for explicit or implicit objectives, infers, from the input it receives, how to generate outputs such as predictions, content, recommendations, or decisions that can influence physical or virtual environments" (art. 3(1) EU AI Act). Cases where AI systems are being used and where we might want to attribute legal responsibility for incurred damages include, but are not limited to, e.g., injuries due to accidents resulting from failures in driver-assistance systems, discrimination due to algorithmic bias, or economic losses resulting from misinformation based on incorrect big data-processing. Such damages can occur in connection with systems of any risk category. Intuitively, though, we are more worried to adequately attribute legal responsibility in high-risk cases.[1]

From a legal point of view, responsibility may be attributed when a (human) actor is causing harm. In liability law, responsibility (in the form of legal liability) means that an actor having caused damages has to compensate these damages under certain conditions. Damages mean any harm to an economically quantifiable (or at least compensable in payment) legal interest.[2] Against this background, an unambiguous attribution of responsibility is crucial to ensure legal certainty: for one thing, those affected should be able to assert their rights effectively; for another, various stakeholders should be provided with a clear legal framework to guide their activities. Yet, it remains unclear (a) what conception of causation liability law relies on, (b) how this conception can be utilized to attribute responsibility when human actions rely on the use of opaque AI systems, and (c) how liability for AI-mediated damages should be handled in practice. The current paper sets out to answer these questions.

We argue that in the context of responsibility attribution (a) causation is best conceptualized in terms of *difference-making* and (b) conceptually disentangle three varieties of it (Sect. 2). We then discuss what strategies might be suitable to uncover each of the three varieties of difference-makers (Sect. 3). Finally, we explore how these insights might be put into legal practice (Sect. 4) and suggest that (c) *mechanistic interpretability* should be required as a gold-standard, at least under certain conditions (e.g., for high-risk cases).

[1] The EU AI Act defines four risk categories (art. 6); it places stringent requirements on high-risk systems. We shall not delve into the details here, but note that high-risk systems may be involved in both low-stakes and high-stakes situations. However, we take cases where both risks and stakes are high to be most critical.

[2] While the analysis of liability in terms of caused damages utilized in liability law seems straightforward, it explicitly brackets considerations of *moral* responsibility (but see, e.g., [3]).

2 Responsibility, Causation and Difference-Making

From a legal point of view, *responsibility* may be attributed when a (human) actor is taking actions causing harm. The focus of our current discussion is on a particular kind of harm: AI-mediated damages (see Sect. 1). Responsibility for those damages takes the form of legal *liability* in the context of civil law (liability law in particular).[3] Since the central indicator for liability is causal impact, law practitioners aiming to attribute responsibility for AI-mediated damages must identify relevant causes.

In liability law, causation is seen as minimum requirement for any form of legal responsibility [16,56], including liability. It is usually determined counterfactually, i.e. by asking whether the damage in question would not have occurred had the (negligent) action not been taken. This comes down to asking whether the latter was a necessary condition of the damage (*conditio sine qua non* or *but for* [22]). Typically, the question of causality in law arises retrospectively, viz. as the question of whether a certain action ultimately made the relevant difference to bring about the damage in question. But questions of causality also play a role in prospective considerations; for instance, when we are assessing the extent to which an actor's behavior can *in principle* influence an outcome [44]. Both these kinds of considerations are relevant for AI-mediated damages: we are interested in determining who is actually responsible for incurred damages and the principled scope of different stakeholders' liability. To help legal scholars answer these questions, we suggest adopting a conception of causation in terms of difference-making borrowed from philosophy.

In philosophy, discussions about causation have a long-standing history. Here, we limit our exposition to the approach we consider most promising for assessing legal responsibility, viz. a *difference-making based view* based on *interventionism* [54,55] which is inspired by counterfactuals [29], statistical-relevance [31,43], and probabilistic views of causation [21,37,47]. The core idea of interventionism is intuitive: causal explanations embody a *what-if-things-had-been-different* conception of explanation. That is, "successful causal explanations consist in the exhibition of patterns of dependency [...] between the factors cited in the explanans and the explanandum—factors that are such that changes in them produced by interventions are systematically associated with changes in the explanandum outcome." [55, p. 228]. Put slightly differently, the idea is that if C causes E, then C will (provided certain conditions are met) be a difference-maker for E. Accordingly, causes are those factors that make a difference to their effects. While the details of interventionism have been the source of intense philosophical debates [4,24,50], the intuition it embodies sits well with established standards of experimental design in the natural sciences, is mirrored in popular science discussions about causation [38], and matches the intuition behind the legal scholars' *conditio sine qua non*. Therefore, we believe, thinking of causes (simply) as relevant

[3] Since we are not concerned with criminal law, moral considerations only play a minor role.

difference-makers to a given outcome is a pragmatically valuable starting point for legal responsibility attribution.[4]

Though the idea is intuitive, identifying relevant difference-makers can be difficult in practice. For AI-mediated damages specifically, the business is complicated by the complexity and opacity of modern AI systems [10,32,58]. This is, at least partly, because such systems are frequently based on large artificial neural networks (ANNs) trained on vast amounts of data with automated ML procedures. Even if we bracket other possible sources of opacity (such as intentional concealment), it is clear that modern AI systems are becoming—and will remain—increasingly opaque not only to their users but also to deployers and providers. Besides, there are many actors potentially involved in building and using AI systems [15] and different stakeholders prioritizing different norms in different contexts [27]. Thus, identifying relevant difference-makers, and the humans in command of them, presents a serious challenge.

To make headway, we believe, it is useful to first focus on what the relevant difference-makers are. We suggest that they might actually be very different things, and that specific kinds of difference-makers can be systematically associated with paradigmatic scenarios: (1) If we are trying to determine who is liable for damages attributable to *inputs* to an AI system, we are effectively asking about the relevant type (i) difference-makers (viz. the very inputs to the system that have been relevant to incurring a certain (damaging) output effect) and who is responsible for them. (2) If, by contrast, we seek to find out who is liable for damages attributable to a system's overall *functional organization*, we are interested in type (ii) difference-makers (e.g., certain *components*, units, or *circuits* within the system's functional architecture). (3) Finally, if we wish to identify who is liable for damages attributable to a system's history, we are seeking to uncover type (iii) difference-makers (viz. features in a system's history, such as the *training data* or *design decisions*, that might yield (damaging or harmful) system behavior). Owing to the distinct nature of type (i), (ii), and (iii) difference-makers, we suggest that different strategies are required in each of these scenarios to uncover the difference-makers in question. Below we explicate these strategies and discuss how they might be applied in legal practice.[5]

3 Finding Different Difference-Makers

Where we are not dealing with inherently interpretable systems, we may uncover relevant difference-makers for AI-mediated damages utilizing strategies from

[4] Note that we do not claim that causation is necessarily or generally understood as difference-making in law. Besides, we are aware that legal responsibility is limited to (voluntary) human actions while interventionism and its conspecifics are not. We are also aware that the conception of causes we offer here is fairly loose as it is silent about the nature of both causes and causal relations. But our aim is not to provide a philosophical analysis; we merely adopt the lingua of difference-making as a tool to support responsibility—and by extension liability—attribution in the legal realm.

[5] Naturally, the distinctions we make here are idealized. See Sect. 4.4 for discussion.

explainable AI (XAI). XAI is a rapidly developing field aiming to explain complex AI systems and their behaviors. Various approaches have been proposed to deliver explanatory information in different contexts [45]. We suggest that some of these approaches can serve as useful tools to uncover specific types of difference-makers: Classical or method-centric XAI approaches are well-suited to identify what inputs have been relevant for a given system output, viz. type (i) difference-makers. Mechanistic interpretability (MI), by contrast, is well-suited to uncover how a system works as a whole, viz. to identify type (ii) difference-makers. Finally, studying the development of AI systems throughout their lifecycle may help legal practitioners detect difference-makers in training data or design decisions, viz. type (iii) difference-makers. We shall sketch these strategies in turn.

3.1 Classical XAI: Which Inputs Yield What Outputs?

Classical XAI methods are computational in nature. They are algorithmic, typically model-agnostic, procedures that can be applied to AI-systems to generate explanatory information about the system in question [25,33,42]. Prototypically, the explanatory information delivered is the output that the XAI method provides; so-called attribution methods present a paradigm case.

For illustration, consider Local Interpretable Model-Agnostic Explanations (LIME) [41]. This technique, applicable to AI classification systems, produces linear surrogate models which highlight the importance of particular features for a given classification. In the case of image recognition, LIME can highlight what pixels within an input image have been especially relevant for applying a given label to that image. This helps uncover, e.g., that to label an image with a cow as "cow" the algorithm actually utilized patches of pasture and mountain scenery rather than the animal in the picture. As such, classical XAI methods can be extremely useful as they answer questions about what inputs or features of the input data have been relevant for generating a given output. That is, they can help identify type (i) difference-makers.

Still, classical XAI is also crucially limited. Since the kind of explanatory information needed to eliminate system opacity depends on numerous factors [27], XAI research has developed a whole cottage industry of XAI methods with different properties [18,34,45], each designed to answer specific questions about a given system. Classical XAI methods do not, however, provide insights about the internal structure or overall functional organization of systems allowing us to, e.g., predict how a system will behave in novel contexts. Thus, classical XAI does not deliver insights about type (ii) difference-makers. Besides, many classical XAI techniques rely on surrogate models, which raises concerns about faithfulness: it remains unclear whether the features identified as type (i) difference-makers by these methods are truly representative of the causal factors.[6]

[6] We thank an Kevin Baum for pointing this out.

3.2 Mechanistic Interpretability: Gathering System Understanding

Mechanistic Interpretability (MI), by contrast, can be utilized to yield generalizable insights about systems' overall functional organization. It takes inspiration from discussions about discovery and explanation in the life sciences [6,7,12,14]. MI starts from the premise that once AI systems become sufficiently complex, they are best investigated and explained through the same lens as biological organisms rather than being treated merely as technical artifacts [25]. Thus, practitioners may seek to characterize AI systems in terms of their *functional organization* (the organized activities of their functionally relevant components), which can be captured in *mechanistic explanations* [5,13,17]. For illustration, consider an example from neuroscience: A rodent's movement through a maze to a target (the overall system's behavior) can be explained in terms of muscular and neural activities; a key component is the spatial map of the maze generated by hippocampal neurons during training. Given this map, the rodent can proceed straight to the target. If we were to break the process down further, we could look into the components relevant to generating the spatial map and find that specific neurons engage in a process called long-term potentiation (LTP), which in turn can be explained in terms of neurotransmitters opening and closing channels in the synaptic terminal. Without going into the details, the gist becomes clear: according to the mechanistic view, the overall behavior of biological systems can be explained in terms of the orchestrated activities of component entities, yielding nested mechanistic hierarchies. When mechanistic inquiry delivers explanations of sufficiently high quality, it affords improved prediction and control of target systems [1,13,23].

MI applies this general strategy to modern AI systems. Seeking MI for, e.g., ML-based ANNs involves (like finding mechanistic explanations for the behavior of biological systems) studying the overall functional organization of AI systems to facilitate a comprehensive grasp of the overall behavior of AI systems; without delving into terminological discussions, we might say MI aims to render AI systems (somewhat) interpretable or transparent. As such, MI goes well beyond classical XAI methods which tend to focus on isolated explanations of specific behaviors or outputs of AI systems in response to specific inputs. Importantly, when we speak of their functional organization, we do not refer to the initial setup of AI systems in terms of nodes, layers and learning-rules. Rather, what is meant is the complex structure that emerges through training [25]. After training, ML-based systems will frequently be composed of specialized "circuits" serving specific functions like edge detection in an image classifier [11,35]. MI seeks to uncover how such *components* are organized together to bring about the observed overall behaviors of AI systems. Importantly, components are individuated *functionally*; they can be of different sizes and kinds.

Uncovering components and their organization in trained AI systems may not be simple, it may even seem impossible at first. It may involve characterizing exotic or distributed structures and requires considerable resources. Nevertheless, we think, seeking MI for AI systems bears great potential to uncover type (ii) difference-makers. As such, MI can help increase overall system perspicu-

ity (viz. explainability, transparency, traceability, explicability, interpretability, understandability [49]) for systems that are not already inherently interpretable. It allows us to better understand a system's capabilities and limitations holistically, helps avoid unforeseen failures even *ante-hoc*, and thus provides greater predictivity and control of the behavior of ML-based systems and is instrumental in identifying and mitigating potential failures of AI systems. Therefore, we claim, MI may serve as an important tool to meet important societal desiderata including liability attribution and safety.[7]

3.3 Considering System History

So far, our discussion has focused on the phase when systems are up and running. However, important difference-makers for AI-mediated damages might also be located in, e.g., training data or design decisions. Such type (iii) difference-makers may be located in various points throughout a system's history or lifecycle.[8]

While there is no single unequivocally accepted AI lifecycle, there is a broad consensus that development and deployment of AI systems iterate through a number of successive stages including problem formulation, data collection and analysis, feature selection, model construction, model evaluation, deployment and usage [15,39,53]. At any of these stages, different actors might be involved and potentially relevant type (iii) difference-makers for an AI system's behavior might be found. Consequently, there are many possible points of intervention for different actors to exert control over AI systems throughout their lifecycle [8].

When discussing different possible actors, the philosophical literature usually points to a range of stakeholders involved in the AI ecosystem: data-subjects, creators, operators, executors, decision-subjects, examiners etc. [51,58]. In the EU AI Act, actors are collectively referred to as *operators* which are separated into providers and deployers. *Providers* are natural or legal persons, public authorities, agencies or other bodies that *develop* AI systems and place them on the market (art. 3(3) AI Act). This includes everyone involved in building and refining AI systems: researchers developing foundational theory, data providers, engineers, programmers. *Deployers* include anyone *using* AI systems (except for non-

[7] While our exposition here has been abstract and theoretical, it is worth mentioning that MI is actually successfully employed by researchers studying the structure of trained image classifiers [35] or trying to understand whether large language models (LLMs) can reasonably be described to possess knowledge [26], represent the world [36], or implement symbolic reasoning [30]. Besides, it is worth pointing out that MI is not the only game in town. There are other methods (e.g., decisions trees) that might be usefully applied to less opaque systems.

[8] Of course, type (i) and (ii) difference-makers only exist *because* a system has a given history. In this sense, type (iii) difference-makers might be more fundamental. However, they are also more complex and impractical to analyze. Thus, we think, it will often be useful to stick with type (i) or (ii) difference-makers when attributing liability in practice. We return to this in Sect. 4.

professional users) (art. 3(4) AI Act). According to this nomenclature, the actors that can (in principle) influence type (iii) difference-makers are developers.

4 Handling Liability in Practice

So far, we conceptually disentangled three types of difference-makers, associated each with paradigmatic damage scenarios, and outlined strategies to identify them. While we do not claim these strategies to be exclusive, we consider them instructive. We now illustrate how our insights could transfer into legal practice, discuss some complications and offer concrete recommendations.

4.1 Damages Attributable to Inputs

Classical XAI (Sect. 3.1) enables probabilistic statements about how the model arrived at a specific output *post hoc*. Such approaches are well-suited for identifying type (i) difference-makers. They can be extremely useful if the question under consideration is what (part of) an input has yielded a specific system output. If, for instance, an image classifier systematically misclassifies horses as cars, and LIME uncovers that the pixels relevant for classifications as "horse" are not those actually showing a horse but those with a certain watermark [28], that information illuminates that the relevant difference-maker for the classification "horse" is the watermark rather than the animal in the picture.

As far as liability is concerned, we must distinguish two kinds of scenarios to determine which actor is to be held liable for damages attributable to inputs by identifying type (i) difference-makers. First, suppose the system *works properly* but is *used incorrectly*, e.g., by applying it in a new or insufficiently tested domain (say, images of houses vs. faces while it was designed to classify dogs and cats). In this case, we submit, the *deployer* is liable for the damages that potential misclassifications incur. If, by contrast, the system is *used as intended*, it is up to the providers to ensure the proper functioning of AI systems, and thereby prevent inputs (type (i) difference-makers) incurring damages.[9] To ensure proper system functioning, developers might actually seek to identify type (ii) or (iii) difference-makers, which we now turn to.

[9] Much could be said about what marks the correct or intended use of AI systems; for now, we simply assume it covers precisely the conditions a system is developed for and tested in. However, there will often be a range of viable deviations, and systems may still function properly in different contexts as long as certain conditions hold or serve for different purposes in other contexts (think of this analogously to off-label use of medication). This takes us into the realm of type (ii) difference-makers and discussions about system perspicuity, ex ante explainability, robustness, transparency, and safety. Though we will touch upon this briefly in the next section, a thorough discussion of the matter would make for a paper of its own.

4.2 Damages Attributable to Functional Organization

MI (Sect. 3.2) provides information about how the overall system works and what its functional (post-training) components are. As such, MI is perfectly suited to uncover type (ii) difference-makers which is particularly relevant when trying to assess liability for damages attributable to functional organization.

At first sight, it seems like damages attributable to the overall functioning of an AI system are straightforwardly attributable to *providers*. Our analysis shows, though, that this is premature. Of course, if a damage elicited by an AI system is attributable to the malfunction of a specific functional unit or component, liability should sit with the developer. But provided that the system's components work as anticipated, liability for damages attributable to type (ii) difference-makers may actually *shift* from system providers *to deployers*; at least insofar as MI gives deployers a certain grasp of how a given AI system works and role specific components play in its overall functioning. This is particularly true for cases where deployers understand, due to MI, what the relevant boundaries and constraints for applying the system in question are. Thus, we suggest, that if deployers understand the functional organization of a system, they can reasonably infer where and how to use it safely; if they consciously deviate from that use and this incurs damages, they should be held liable even if providers might have specified a different scope of safe use.[10] Our suggestion is in line with Article 14 of the EU AI Act which states that adequate human oversight in high-risk scenarios demands an overseer to "properly understand the relevant capacities and limitations of the high-risk AI system and be able to duly monitor its operation, also in view of detecting and addressing anomalies, dysfunctions and unexpected performance" and be able "to intervene on the operation of the high-risk AI system" (see also [48]). With regards to both requirements, a detailed (and generalizable) understanding of a trained AI-systems' overall functional organization is crucial. This, we take it, requires identifying type (ii) difference-makers.

Besides, mechanistically interpretable systems offer another *decisive advantage*: once we understand how a system works as a whole, we can *anticipate how it will behave in response to novel or unexpected inputs*. This contributes to AI systems' *ex ante explainability and safety* [25]. While these properties are

[10] This suggestion relates to worries about humans in the loop becoming scapegoats [40]: When deployers work with black box AI systems they do not understand, and just (blindly) confirm or follow AI recommendations, they run the risk of becoming mere scapegoats for AI-mediated damages; they are held responsible although they do not have any understanding or control over the damages incurred. With our analysis in mind, scapegoating can be avoided: for if the deployer utilizes an opaque AI system in the ways it is intended to be used, they are *not* responsible for potential damages (Sect. 4.1); if, by contrast, the system is rendered interpretable, they might be responsible but no longer unknowingly so. Indeed, the understanding of a system's functional organization that MI proves might not only give deployers the option to anticipate but also systematically manipulate system behavior once they know the relevant type (ii) difference-makers.

especially required in high-risk domains, we can illustrate the point with our reference to our image classifier: Once we understand that it systematically recognizes watermarks, we not only uncover that watermarks can be relevant type (i) difference-makers but also understand that some functional part of the system systematically detects watermarks; this detection circuit constitutes a relevant type (ii) difference-maker. Knowing about it, we can anticipate our classifier will not only recognize watermarked images of cars as "horse", but also images of mountains or dogs. Accordingly, deployers can take precautions when using the system (e.g., removing watermarks) and thus avoid damages. At the same time, providers may utilize insights about relevant type (ii) difference-makers to fix problematic behavioral patterns and search for their origins in a system's history.

4.3 Damages Attributable to System History

Studying system history is well-suited for uncovering type (iii) difference-makers along the entire lifecycle of AI systems (Sect. 3.3). Granted, since system history can potentially encompass many different factors and actors, identifying specific type (iii) difference-makers for a given damage is a challenging task. Yet, it seems straightforward that difference-makers in an AI system's history will usually not be influenced by deployers but providers. Thus, liability for damages attributable to an AI system's history will generally lie with providers. This is also acknowledged in the EU AI Act which specifies obligations for providers along the entire lifecycle (art. 9–17). In line with what we said about liability for damages elicited by type (i) difference-makers, we think that data curators and system designers—creators in ecosystem terms—should generally be held liable if systems function improperly despite valid inputs. By contrast, liability of causally more distant actors, such as data subjects, seems unreasonable.

While our suggestion seems natural given consideration of the AI lifecycle, and fits well with the fact that the EU AI Act does not concern data subjects, it interestingly contrasts with the EU Artificial Intelligence Liability Directive (AILD) and the new Product Liability Directive (PLD). AILD narrowly focuses on linking an actor's fault directly to the AI system's output, neglecting earlier causal factors such as flawed training data or design choices, which we seek to integrate. PLD, by contrast, stipulates that the liability encompasses the entire causality chain (under certain conditions), indicating a more claimant-friendly stance (art. 10(4) PLD) [57].

4.4 Practical Complications

We acknowledge that while type (i), (ii) and (iii) difference-makers are conceptually distinct, they might be hard to disentangle in practice. Specifically, there will usually be type (iii) difference-makers relevant for the existence of certain type (i) or (ii) difference-makers and there will be type (ii) difference-makers relevant for the existence of specific type (i) difference-makers. Besides, real life cases will often involve several difference-makers simultaneously. Despite these limitations, we think that the distinctions laid out here can be practically useful

to attribute responsibility for AI-mediated damages in systematically different contexts. After all, responsibility attribution is not about necessarily identifying a single liable subject; it is about shedding light on the various ways in which different actors could have made relevant differences.

Regarding actors, it is worth pointing to another issue. Our discussion here has focused predominantly on identifying relevant difference-makers as a proxy for causes. Though causation is considered a necessary condition of responsibility and liability (Sect. 2), it may not be sufficient. For instance, identifying relevant difference-makers may not automatically reveal the actor in command of them (see also Sect. 4.3); or we might find that no (human) actor is actually in command of the difference-maker in question. To this end, multi-agent models, Markov-models and game-theoretical methods might be extremely useful [2,52]. Yet, we take it, if we aim to attribute responsibility, we must begin by finding the relevant difference-makers. And this endeavor we hope to illuminate with our analysis.

4.5 Our Recommendations

Based on these considerations, we derive *three concrete recommendations* for dealing with AI-mediated damages in legal practice: First, we think that system providers should adhere to *strict design standards* to exclude or prevent the influence of unwanted type (iii) difference-makers already before a system gets deployed. For the most part, this might already be achieved by careful consideration of risks at various stages in the AI lifecycle (as demanded by the EU AI Act), particularly data curation and model selection. Despite best efforts, critical type (iii) difference-makers may sometimes only become apparent once a system is being deployed—in which case immediate measures must be taken to update, redesign or otherwise improve the system. Meanwhile, developers are liable for the damages incurred.

Second, we suggest *requiring MI as a gold-standard* for complex AI systems, at least under certain conditions and where systems are not already inherently interpretable. The reason for this is *not* that MI is generally superior to the other strategies for finding difference-makers (Sect. 3) but that it is especially suited to uncover type (ii) difference-makers. And it is type (ii) difference-makers that we need to grasp if we want to render systems understandable, interpretable, or transparent. With the help of MI, deployers can be provided with at least some grasp of the overall system's functioning, *ex ante explainability and safety* are improved, and scapegoat worries can be avoided. Beyond that, insights gained through MI might help satisfy the right to explanation anchored in Article 86 of EU AI Act.

Of course, this call for MI may not always be easy to satisfy; for achieving MI for black box AI systems may be very complicated (sometimes hardly possible), and usually requires significant investments in terms of both time and research resources. This is why we limit our recommendation to certain conditions, specifically the application of high-risk systems in high-stakes contexts. The classification of high-risk AI systems in the EU AI Act (art. 6) is complex,

debated, and tied to other legal rules (e.g., existing product regulation). For current purposes it shall suffice to note that high-risk systems are cashed out in terms of the *purpose to which they are put*, e.g., law enforcement, supporting critical infrastructure, essential services, or education. However, this classification does not necessarily capture all and only systems applied in high-stakes contexts where potential damages might be especially grave, harmful, or detrimental. A calendar application used to schedule appointments may be classified as high-risk if used in a clinic but if it crashes that does not keep doctors from seeing patients; so the damage would be relatively contained [19]. Conversely, a chatbot may be classified as low risk even though it could cause major harm by spreading propaganda in social networks. We propose to require MI specifically where high risks and high stakes come together. A few complications should be noted. First, we are aware that MI might be highly costly to achieve. In cases where MI seems completely infeasible, a viable alternative might consist in building interpretable systems [42] benchmarked against their black box competitors during system development. Second, that an AI system is mechanistically interpretable *in principle* does not guarantee that it is interpretable to *any deployer* [9,27,45]; for interpretability not only depends on system properties but also, at least potentially, on stakeholders' (cognitive) abilities [32,46]. We propose the sort of MI that should be required should be feasible for average professional deployers.

Third, and finally, we suggest to create legal rules *indemnifying* (excusing) *distant actors*. Paradigmatically, data subjects are only remotely involved in model building by providing training data. Besides, their number might reach millions for any given model. Thus, the individual influence of any data subject on any given AI-mediated damage is negligible compared to that of system designers. Potentially, even programmers of small portions of code within an AI system or data curators might be indemnified under certain conditions. Here again, we believe, the degree of indemnification of actors involved early in the AI lifecycle will have to correlate with risks and stakes. For instance, low-stakes applications like music recommendations may not require carefully curated data. By contrast, for specialized high-risk systems, such as medical devices, legal responsibility for providing adequately curated data might be both feasible and functional.

5 Summary and Outlook

We analyzed causation in terms of difference-making and conceptually disentangled three types of difference-makers, each of which is associated with AI-mediated damages in specific scenarios. We discussed how to deal with each of these difference-makers when attributing liability in legal practice. From our discussion, we derived three proposals. Our main proposal is that to increase the ex ante safety of opaque AI systems it may be adequate to require *mechanistic interpretability* as a norm—particularly where high risks and high stakes are involved. Where MI is ensured, deployers benefit from increased control and liability may be shifted from providers to deployers. As a result, scapegoat worries

may be escaped and distant actors (e.g., data suppliers) can be shielded from legal responsibility. Introducing such liability limitations offers the advantages of reducing legal uncertainty and potentially creating an incentive for data subjects to contribute to the availability of high-quality training data.

While our discussion has not explicitly taken generative AI (genAI) into consideration, we believe it is only natural to extend our views to this technology. To be sure, MI is already hard and costly to achieve for non-generative AI, but it is pursued for at least some genAI systems as well (see, e.g., [26,30,36]). We believe that progress in this area of research, together with transparency requirements for genAI [20], may light a path towards extending our suggestions to genAI. But that remains the project of another paper.

References

1. Baetu, T.M.: Mechanism schemas and the relationship between biological theories. In: McKay, P., Williamson, J., Russo, F. (eds.) Causality in the Sciences. Oxford University Press (2011)
2. Baier, C., Funke, F., Majumdar, R.: A game-theoretic account of responsibility allocation. In: Zhou, Z.H. (ed.) Proceedings of the 30th International Joint Conference on Artificial Intelligence, IJCAI-21, pp. 1773–1779. International Joint Conferences on Artificial Intelligence Organization (2021)
3. Baum, K., Mantel, S., Schmidt, E., Speith, T.: From responsibility to reason-giving explainable artificial intelligence. Philos. Technol. **35**(12), 1–30 (2022)
4. Baumgartner, M.: Interventionism and epiphenomenalism. Can. J. Philos. **40**, 359–384 (2010)
5. Bechtel, W.: Mental Mechanisms: Philosophical Perspectives on Cognitive Neuroscience. Psychology Press (2008)
6. Bechtel, W.: Looking down, around, and up: mechanistic explanation in psychology. Philos. Psychol. **22**(5), 543–564 (2009)
7. Bechtel, W., Richardson, R.C.: Discovering Complexity: Decomposition and Localization as Strategies in Scientific Research. Princeton University Press (1993)
8. Bryson, J.J.: The Artificial Intelligence of the Ethics of Artificial Intelligence: An Introductory Overview for Law and Regulation. In: The Oxford Handbook of Ethics of AI. Oxford University Press (2020)
9. Buchholz, O.: A means-end account of explainable artificial intelligence. Synthese **202**(2), 33 (2023)
10. Burrell, J.: How the machine 'thinks': understanding opacity in machine learning algorithms. Big Data Soc. **3**(1), 2053951715622512 (2016)
11. Cammarata, N., et al.: Thread: circuits. Distill **5**(3), e24 (2020)
12. Craver, C.F.: Role functions, mechanisms, and hierarchy. Philos. Sci. **68**(1), 53–74 (2001)
13. Craver, C.F.: Explaining the Brain: Mechanisms and the Mosaic Unity of Neuroscience. Oxford University Press (2007)
14. Craver, C.F., Darden, L.: In Search of Mechanisms: Discoveries across the Life Sciences. University of Chicago Press (2013)
15. Deck, L., Kästner, L., Kühl, N., Schomäcker, A., Speith, T.: Explainable AI (XAI) for fairness: mapping the potential of XAI for fairness desiderata along the AI lifecycle. In: EWAF Proceedings (2024)

16. Denga, M.: Zurechnung. Mohr Siebeck, Tübingen (2022)
17. Glennan, S.: The New Mechanical Philosophy. Oxford University Press (2017)
18. Guidotti, R., Monreale, A., Turini, F., Pedreschi, D., Giannotti, F.: A survey of methods for explaining black box models. ACM Comput. Surv. **51** (2018)
19. Hacker, P.: AI regulation in Europe: from the AI act to future regulatory challenges (2023)
20. Hacker, P., Cordes, J., Berz, A.: Transparenz generativer KI. Rechtliche Rahmenbedingungen und technische Möglichkeiten. Gewerblicher Rechtsschutz und Urheberrecht, pp. 1777–1785 (2024)
21. Halpern, J.: Actual Causality. MIT Press (2019)
22. Hart, H.L.A., Honoré, T.: Causation in the Law. Oxford University Press (1985)
23. Howick, J., Glasziou, P., Aronson, J.K.: Evidence-based mechanistic reasoning. J. R. Soc. Med. **103**(11), 433–441 (2010)
24. Kästner, L.: Philosophy of Cognitive Neuroscience: Causal Explanations, Mechanisms and Experimental Manipulations. De Gruyter, 1st edn. (2019)
25. Kästner, L., Crook, B.: Explaining AI through mechanistic interpretability. Eur. J. Philos. Sci. **14**(52) (2024)
26. Lam, N.: Explanations in AI as claims of tacit knowledge. Mind. Mach. **32**(1), 135–158 (2022)
27. Langer, M., et al.: What do we want from explainable artificial intelligence (XAI)? – a stakeholder perspective on XAI and a conceptual model guiding interdisciplinary XAI research. Artif. Intell. **296**, 1–24 (2021)
28. Lapuschkin, S., Wäldchen, S., Binder, A., Montavon, G., Samek, W., Müller, K.R.: Unmasking Clever Hans predictors and assessing what machines really learn. Nat. Commun. (2019)
29. Lewis, D.: Causation. J. Philos. **70**, 556–567 (1973)
30. Li, K., Hopkins, A.K., Bau, D., Viégas, F., Pfister, H., Wattenberg, M.: Emergent world representations: exploring a sequence model trained on a synthetic task (2024)
31. Machamer, P.K., Darden, L., Craver, C.F.: Thinking about mechanisms. Philos. Sci. **67**, 1–25 (2000)
32. Mann, S., Crook, B., Kästner, L., Schomäcker, A., Speith, T.: Sources of opacity in computer systems: towards a comprehensive taxonomy. In: 31st IEEE International Requirements Engineering Conference (RE'23) (2023)
33. Mittelstadt, B., Russell, C., Wachter, S.: Explaining explanations in AI. In: Proceedings of the Conference on Fairness, Accountability, and Transparency, pp. 279–288 (2019)
34. Molnar, C.: Interpretable machine learning – a guide for making black box models explainable (2024)
35. Olah, C., et al.: The building blocks of interpretability. Distill **3**(3), e10 (2018)
36. Pavlick, E.: Symbols and grounding in large language models. Philos. Trans. Royal Soc. A: Math. Phys. Eng. Sci. **381**(2251) (2023)
37. Pearl, J.: Causality: Models, Reasoning, and Inference. 2nd edn. Cambridge University Press (2000)
38. Pearl, J., Mackenzie, D.: The Book of Why: The New Science of Cause and Effect. Basic Books, New York, NY (2018)
39. Quemy, A.: Two-stage optimization for machine learning workflow. Inf. Syst. **92**, 101483 (2020)
40. Ranisch, R.: Scapegoat-in-the-Loop? Human control over medical AI and the (Mis) attribution of responsibility. Am. J. Bioethics **24**(9), 116–117 (2024), pMID: 39226016

41. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why should i trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144 (2016)
42. Rudin, C.: Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. Nat. Mach. Intell. **1**(5), 206–215 (2019)
43. Salmon, W.: Four Decades of Scientific Explanation. University of Minnesota Press (1989)
44. Shavell, S.: An analysis of causation and the scope of liability in the law of torts. J. Leg. Stud. **9**(3), 463–516 (1980)
45. Speith, T.: A review of taxonomies of explainable artificial intelligence (XAI) methods. In: 2022 ACM Conference on Fairness, Accountability, and Transparency, pp. 2239–2250 (2022)
46. Speith, T., Crook, B., Mann, S., Schomäcker, A., Langer, M.: Conceptualizing understanding in explainable artificial intelligence (XAI): an abilities-based approach. Ethics Inf. Technol. **26**(2), 40 (2024)
47. Spirtes, P., Glymour, C., Scheines, R.: Causation. Prediction and Search, Springer (1993)
48. Sterz, S., et al.: On the quest for effectiveness in human oversight: interdisciplinary perspectives. In: Proceedings of the 2024 ACM Conference on Fairness, Accountability, and Transparency, pp. 2495–2507. New York, NY (2024)
49. Sterz, S., Baum, K., Lauber-Rönsberg, A., Hermanns, H.: Towards perspicuity requirements. In: 2021 IEEE 29th International Requirements Engineering Conference Workshops (REW), pp. 159–163 (2021)
50. Strevens, M.: Comments on woodward, making things happen. Philos. Phenomenol. Res. **77**, 171–192 (2008)
51. Tomsett, R., Braines, D., Harborne, D., Preece, A., Chakraborty, S.: Interpretable to whom? A role-based model for analyzing interpretable machine learning systems (2018)
52. Triantafyllou, S., Singla, A., Radanovic, G.: Actual causality and responsibility attribution in decentralized partially observable markov decision processes. In: Proceedings of the 2022 AAAI/ACM Conference on AI, Ethics, and Society, AIES 2022, pp. 739–752. ACM, New York, NY (2022)
53. Wang, M., Cui, Y., Wang, X., Xiao, S., Jiang, J.: Machine learning for networking: workflow, advances and opportunities. IEEE Network **32**(2), 92–99 (2017)
54. Woodward, J.: Making Things Happen. Oxford University Press, Oxford, UK (2003)
55. Woodward, J.: Mental causation and neural mechanisms. In: Hohwy, J., Kallestrup, J. (eds.) Being Reduced: New Essays on Reduction, Explanation, and Causation, pp. 218–262. Oxford University Press (2008)
56. Zech, H.: Schöpfen und Schaden, Zurechnung im Immaterialgüterrecht und im Haftungsrecht, insbesondere beim Einsatz von künstlicher Intelligenz. In: Kubis, S., Peifer, K.N., Raue, B., Stieper, M. (eds.) Ius Vivum: Kunst - Internationale - Persönlichkeit, Festschrift für Haimo Schack zum 70. Geburtstag, pp. 377–391. Mohr Siebeck, Tübingen (2022)
57. Zech, H.: Liability for AI: Complexity Problems. In: Lohsse, S., Schulze, R., Staudenmayer, D. (eds.) Liability for AI, pp. 193–200. Nomos Hart, Baden Baden (2023)
58. Zednik, C.: Solving the black box problem: a normative framework for explainable artificial intelligence. Philos. Technol. **34**(2), 265–288 (2021)

Development and Maintenance of Trust in Human-Drone Interaction: Preliminary Empirical Findings in a Warehouse Setting

A. Kluge[1(✉)], L. Thomaschewski[1], Ch. Reining[2], S. Franke[2], S. Awasthi[2], M. Roidl[2], O. Vogel[1], and M. Pauly[2,3]

[1] Chair of Work and Organizational Psychology, Ruhr University Bochum, Universitätsstraße 150, 44801 Bochum, Germany
annette.kluge@ruhr-uni-bochum.de

[2] Faculty of Mechanical Engineering, Chair of Material Handling and Warehousing, TU Dortmund, August-Schmidt-Straße 1, 44227 Dortmund, Germany

[3] Research Center Trustworthy Data Science and Security, Joseph-Von-Fraunhofer-Straße 25, 44227 Dortmund, Germany

Abstract. This paper explores the process of trust development in human-drone interaction, focusing on how trust evolves over time when humans are unfamiliar with AI-assisted technology. The study investigates the development of trust, specifically whether it changes due to personal along a series of experiments.

A study ($N = 19$) was conducted in a warehouse setting using aerial drones. Participants interacted with a drone that guided them through a that was based on real-world warehouse processes. Data was collected through questionnaires and distance measurements. Changes in comfortable human-drone distance were assessed before and after the interaction, changes in the mental model were measured before, right after, and two weeks later via an online questionnaire, and changes in trust were recorded during the interaction. Variables measured were 1) Subjective trust level: Participants rated their trust in the drone, 2) Changes in individual mental model: Participants answered a questionnaire to assess their understanding of the drone's capabilities, and 3) Human-drone distance: The comfortable distance between the participant and the drone was measured.

while the participants' mental models of the drone improved significantly after the interaction, their subjective trust levels remained relatively stable. The research suggests that initial trust levels are influenced by pre-existing psychological anchors and are difficult to change solely through new experiences delete word.

It is concluded that, while human-drone experiences can enhance understanding of AI systems, it is more difficult to change pre-existing levels of trust. The findings suggest that a more comprehensive approach, that considers cognitive, affective, and behavioral measures, is important to examine human-drone interaction.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 203–217, 2026.
https://doi.org/10.1007/978-3-032-01377-4_11

1 The Role of Trust and Its Development Over Time While Interacting with an Unfamiliar Technology

In times of ever faster learning machines, trust is an essential precondition for effective human-technology interaction (Rutinowski et al., 2024). In the present paper, we take a Human Factors perspective and follow the definition of trust as the willingness of a party to be vulnerable to the actions of another party based on the expectation that the other will perform a particular action important to the trustor, irrespective of their means to monitor or control each other (deVisser et al., 2020). Human Factors and Ergonomics are concerned with the application of what is known about human actors (their abilities, characteristics, and limitations) to design products, environments in which people work, and jobs they perform (Human Factors and Ergonomics Society/HFES, 2025). When individuals are not used to interacting with a particular technology, such as an autonomous or AI-assisted (aerial) vehicle, e.g., a warehouse drone, trust is assumed to develop over time.

The objective of the present research was to understand the process of trust development over time when interacting with an unfamiliar technology in a workplace setting for the first time. The objective to introduce a new technology at the workplace, comes with expectations regarding e.g., increased safety, productivity and performance, an increased human centered task allocation, and/or reduced strain and workload for the workforce. All of these expected benefits require a human-technology interaction based on trust (Müller and Hertel, 2025). If humans do not trust in e.g., an autonomous and/or AI assisted system, they will not be willing to delegate tasks, regardless of their technical performance potential (O'Neil et al., 2022; Hancock et al., 2023).

Contrary, over-reliance on the system can lead to human complacency and hinder them to detect technical errors and limitations (Alonso and La Puente, 2018). The inappropriate human trust in AI and autonomous systems, and resulting inappropriate reliance, is assumed to cause decreased performance in human-AI teams (Alonso and La Puente, 2018). Thus, trust is a key factor in human-agent interaction and influences whether a human will rely upon the system (Lyons, 2013). Trust in AI systems is assumed to result from the interplay between person-related and technology-related factors, e.g., the level of understandability and predictability of a systems behavior (Akula et al., 2019; Rutinowski et al., 2024).

Trust in AI-assisted systems occurs when users believe an AI system will work as intended, understand its strengths and limitations, and rely on or intend to rely on it (deVisser et al., 2020). Aspects of trust include the human's belief that an AI system will accomplish the task it was designed for, as well as an understanding of its errors and limitations (McDermott and Brink, 2019).

In interactions with newly introduced and unfamiliar (AI-)technology, trust is based on: a) characteristics of the individuals (e.g., trust propensity) who are supposed to trust a technology, and b) dynamic interaction, where trust evolves over time based on direct experiences with the system. Hoff and Bashir (2015), Hancock et al. (2023), and Miller et al. (2021) distinguish (see Fig. 1):

- Dispositional trust, as a personal trait, influenced by factors such as cultural background, age, gender, and personality, which individuals bring into a shared human - technology work task.
- Situational trust, which differentiates between internal person-related variability (e.g., self-efficacy, expertise, mood, and attention span) and external variability (e.g., type of system, task difficulty, perceived risks, workload).
- Learned trust, which includes initial trust at the beginning of collaboration with the system and dynamically learned trust during interaction with the system (trust trajectory, i.e., the increase or decrease in trust through system use).

Trust in technology is also affected by system design features, e.g., appearance, ease of use, communication style, perceived reliability, predictability, perceived errors, and usefulness to fullfil a task, and task technology fit respectively.

In the present study, we understand trust as an "emergent state" rather than a fixed characteristic of the human-technology interaction. Trust is not only relevant in the context of technology disuse but also in cases of inappropriate (blind) trust (overreliance & complacency (Parasuraman and Manzey, 2010) or misuse (Parasuraman and Riley, 1997; Gupta and Woolley, 2021). Too little trust in highly capable technology leads to non-use and high costs in terms of lost time and work efficiency (the technology could perform better if it were trusted). In contrast, excessive trust in incompetent technology results in overreliance and misuse, such as safety violations (Lee and See, 2004; Hoff and Bashir, 2015).

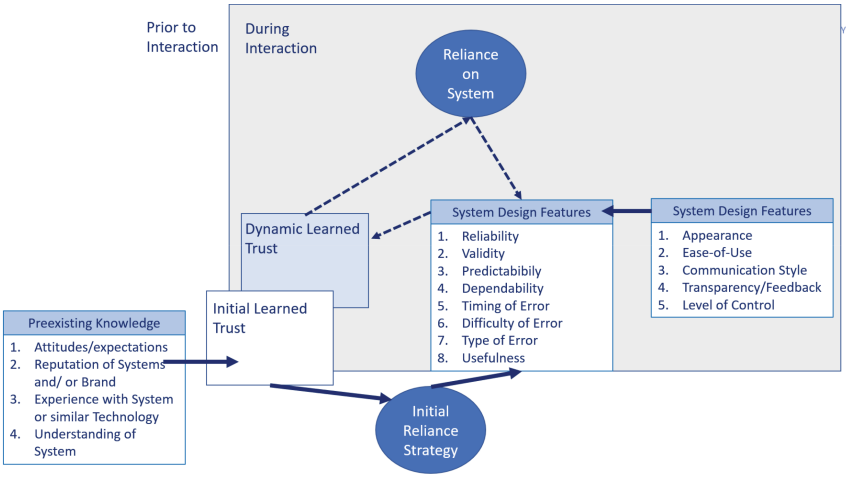


Fig. 1. Characteristics of human and technology in the development of trust in human-technology interaction (based on Hoff & Bashir, 2015).

2 The Development of Trust in AI -Systems and the Expectancy Confirmation/Disconfirmation Theory

In the past years, user expectations have changed based on prior experiences with intelligent system, e.g., due to the introduction of ChatGPT (Kocielnik et al., 2019; Lew and Walther, 2023). In current research, the expectation disconfirmation/confirmation theory (Oliver, 1980) has been used to explain and predict the continuance in using these kind of technologies (Oliver, 1980; Mi Alnaser et al., 2023; Buschmeyer et al., 2023; Choi et al., 2019; Bhatnagr and Rajesh, 2024).

Expectation confirmation/disconfirmation (C/D) theory has remained popular in measuring user satisfaction when interacting with new technologies. Originally established by Oliver (1980), this theory evaluates the confirmation or disconfirmation of users' pre-adoption expectations, as well as their perceived performance and satisfaction. User expectations refer to what individuals anticipate from a technology and what they ultimately receive in return. Consequently, perceived performance reflects users' perceptions of a technology's attributes, benefits, and outcomes. This suggests that users initially experience expectation confirmation or disconfirmation when engaging with a new technological system. If a technology meets user expectations and enhances satisfaction, it fosters a positive attitude toward adoption. Conversely, expectation disconfirmation can negative perceptions and reluctance toward using the technology (Mi Alnaser et al., 2023). On the other hand, positive expectation confirmation enhances perceived performance, resulting in higher user satisfaction and greater acceptance of the technology. Previous studies have employed the C/D theory as a theoretical framework to examine expectation confirmation/disconfirmation and perceived usefulness in the context of various technological services.

Combining both concepts, trust development by Hoff and Bashir (2015) and the expectancy C/D theory, proposed building blocks of the process of trust development are a) the initial trust level held by the person including expectations about the technology's performance, b) a concrete interaction with the technology and the building of situational trust based on the reflection of c) confirmation/disconfirmation experiences (Schön, 1987) (see Fig. 2). In a work context, the reflection of the confirmation /disconfirmation of expectations relates to an individual mental model about the work tasks to be fulfilled (Mathieu et al., 2000). The mental model we propose includes a task mental model about what the interaction task affords and what the human and the AI-system or autonomous technology is capable of.

By means of experiential learning, which includes reflection processes in iterative human-technology interactions, trust will emerge through processes called reflection before action (e.g., thinking ahead, remembering prior experiences, thinking about goals and goal achievement, and a first draft of an interpretation of the situation), reflection in action (including estimating, anticipating, experiencing (McDermott et al., 20185; McDermott and Brink, 2019; Dubey et al., 2020), and recognizing patterns), and reflection on action (thinking about past experiences/incidents, interpreting and understanding surprises, unexpected outcomes, and updating the mental model) (see Schön (1987)).

As illustrated in Fig. 2, we propose an ongoing process that involves expectations (reflection before action), interaction (reflection in action), reflection on action and updated expectation (reflection before action), in which met and unmet expectations

during concrete interactions serve as incidents that trigger updating and adjust trust levels (McDermott et al., 20185; McDermott and Brink, 2019; Dubey et al., 2020). With subsequent iterative interactions, trust is assumed to develop in relation to an individual point of reference (see Fig. 2).

The proposition is based on the hypothesis that trust is the result of a complex process of comparison between the perceived level of performance and an expected standard of comparison (expectation) prior to the interaction with the technology (Buschmeyer et al., 2023). When transferred to the process of trust development, it can be assumed that participants enter a situation with unfamiliar technology with predefined performance-specific expectations, and updates on the trust level and the trust development would only occur in cases of strong positive or negative expectancy disconfirmation.

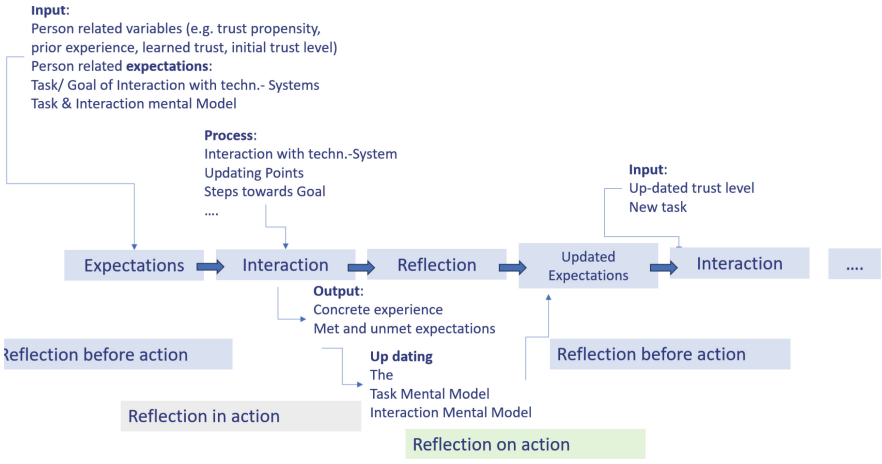


Fig. 2. Proposed process of trust development.

In the present study, that was planned as a pre-study, we focus on the question of how far trust develops or changes over time, when humans are unfamiliar with an AI-assisted technology in a warehouse setting.

Several person- and human-drone-interaction-related variables were measured, as presented at https://osf.io/pw2bs/. In the present paper, we focus on the description and analysis of three variables, as the data analysis of all measured variables is in progress:

1. the affective level of subjectively measured trust (1 item measure in %), and
2. the cognitive level of changes in the individual mental model over time (7 items, Likert Scale 1–5 (very high), developed by Tausch, 2024), and
3. the behavioral level defined as the human-drone distance perceived as comfortable before and after human-drone interaction (measured in cm).

Human-drone distance and the individual mental model were measured before and after the human-drone interaction. Additionally, the individual mental model was assessed again via an online questionnaire two weeks after the experiment. Subjective trust (expressed in precent) was measured during the human-drone interaction.

3 Study Design

To empirically explore the assumption regarding trust development we designed the use case "Warehouse Drones". Technological advancements of aerial drones for such indoor applications receive a lot of attention from the robotics community (Awasthi et al., 2023). As of now, they are not common in industrial settings. This is because challenges like indoor localization, collision avoidance, and battery runtime are not yet solved to a degree that would allow for their large-scale roll-out.

21 participants took part in the study (approved by the local ethics committee and after given informed consent, No 812) in August 2024 in the research hall at the chair of material handling and warehousing of TU Dortmund University. Two participants (participant 17 and 20) were excluded from our analysis due to incorrect execution of the experiment, leaving us with measurements from 19 participants (Mage = 28.89, SDage = 9.71; 8 females, 11 male). The study took approx. 15 min. per person for the five stations (described below). Participants received a compensation of 15 EUR.

The research hall is a testbed for swarms of unmanned aerial vehicles (UAVs) (Gramse et al., 2022). It is equipped with a marker-based motion capturing system. It allows tracking of all moving entities (humans, drones, boxes, etc.) with up to 300 Hz and an accuracy of less than 1 mm. Beyond that, RGBD cameras are deployed for video recording. The aerial drones are self-built, in cooperation with the Fraunhofer Institute for Material Flow and Logistics. Common warehouse equipment such as boxes, racks, and forklift trucks are available to create logistical scenarios that closely resemble real-world systems.

The use case scenario included the story of a recently hired warehouse worker who is learning on the job how to navigate through the warehouse – assisted by a drone. The drone shows the worker the way to the first picking station so that they get to know the warehouse. We designed a parcourse with five stations (see Fig. 3 and 4):

- Station 0 (pre-post measure): The drone flies towards the worker; the worker stops the drone via button press on a remote control when the comfort distance was reached (pre and post experiment)
- Station I: Arrival
- Station II: Flying around a barrier
- Station III: Loss of sight
- Station IV: Destination

The stations were designed to let participants make concrete experiences with the drone's behavior and to give them opportunities to update their individual trust levels.

As described in https://osf.io/pw2bs/, participants were asked through open-ended questions (via voice recordings) to indicate their mental model, e.g., 1) what do you think the drone will do on arrival?, 2) what characteristics must the drone have to fulfil the task?, 3) how close will the drone come to you?, 4) what do you think the drone will do next?, 5) how must the drone behave so that you want to continue following it despite the barrier?

The drone flew at an approximate height of 102 cm. The average height of the participants was $\bar{x} = 174.89$ cm ($SD = 9.05$).

Currently, AI is not integrated into the creation of the scenario. However, several areas where AI could contribute include indoor UAV localization without relying on motion capturing systems, planning the shortest path from the source to the destination, and obstacle avoidance along the route. Implementing these AI-driven approaches would enhance the autonomy of the drone.

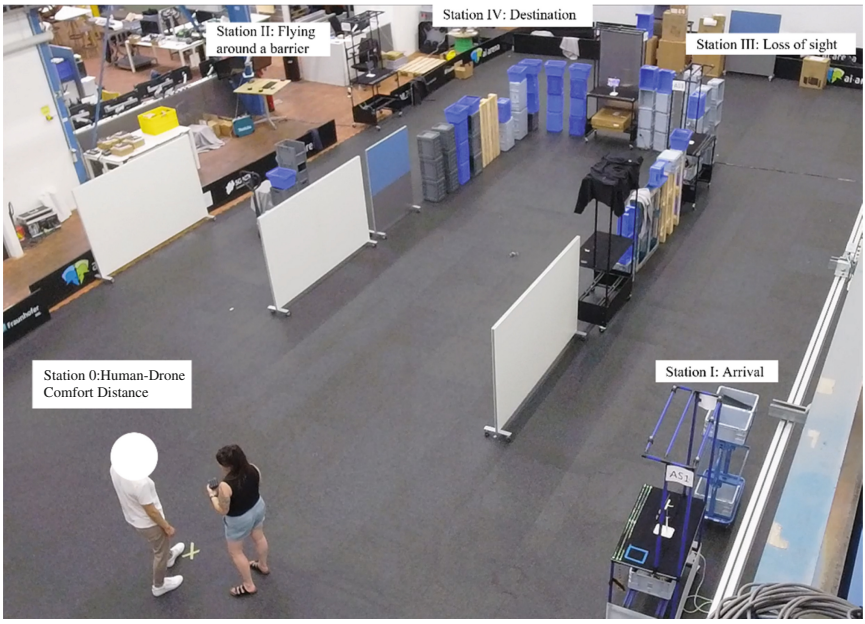


Fig. 3. Experimental set-up in the research hall. Top view of the scenario that was created in the research hall. Four picking stations are placed in different locations (station I – IV).

4 Preliminary Descriptive Results on Trust and Mental Model Over Time

We present a selection of preliminary results, focusing on a quantitative analysis of mental model development over time and subjectively measured trust. The accuracy of the mental model in relation to the drone was measured by a questionnaire post and pre-experimental, as well as two weeks after the experiment (follow-up) (Table 1). The subjectively preferred human-drone distance was measured immediately before and after the experiment/ human-drone interaction (Table 2). Trust was assessed during the human-drone interaction at picking stations I, II, and III (cf. Figure 3; Table 3).

Fig. 4. Picking stations used in the experimental set-up.

Table 1. Descriptive results of mental model measures (pre, post, and follow-up).

	t0		t1		t2	
	M	*SD*	*M*	*SD*	*M*	*SD*
Overall mean (Scale: 1;5)	3.29	1.00	4.51	0.56	3.96	0.74
I know what the drone is capable of	3.11	1.15	4.32	0.67	4.19	0.83
I have no clue how the drone will behave during our collaboration	3.11	1.37	3.95	1.31	3.56	1.41
I know the limits of the drone	3.11	1.33	3.63	0.96	3.38	1.15
I have an idea how the drone functions	3.68	1.29	4.42	0.77	4.50	0.63
I am able to estimate that the drone is useful for	3.63	1.01	4.37	0.83	4.12	0.81
It is difficult for me to say what the drone is useful for	2.21	1.32	4.05	0.97	3.88	1.09
I understand how the drone and I can collaborate	3.21	1.03	4.32	0.75	4.06	0.85

Note. t0 = pre-experimental, t1 = post experimental, t2 = follow-up (2 weeks after the experiment). *M* = Mean, *SD* = Standard Deviation

Table 2. Descriptive results of measuring human-drone distance (pre and post).

	Pre experimental		Post experimental	
	M	*SD*	***M***	***SD***
Drone distance towards the participant (in cm)	**136.95**	72.42	**112.79**	37.08

Note. M = Mean, *SD* = Standard Deviation

On the individual level, participants did change their trust level marginally in relation to their initial trust level (see Fig. 5).

Table 3. Descriptive results of measuring reported trust (during the experiment).

	Picking Station I		Picking Station II		Picking Station III	
	M	*SD*	***M***	*SD*	***M***	*SD*
Reported trust in %	**78.95**	22.04	**78.45**	21.84	**78.82**	21.74

Note. Trust was verbally reported at picking stations I-III. *M* = Mean, *SD* = Standard Deviation

Results of a repeated measure ANOVA (Friedrich et al., 2023) of the reported trust between picking stations I to III showed no statistical significance ($ATS(1.18, \infty) = 0.034$, $p = .889$). These results indicate that participants, despite their interactions with the drone, remained attached to their initial evaluations of trust. This suggests that trust

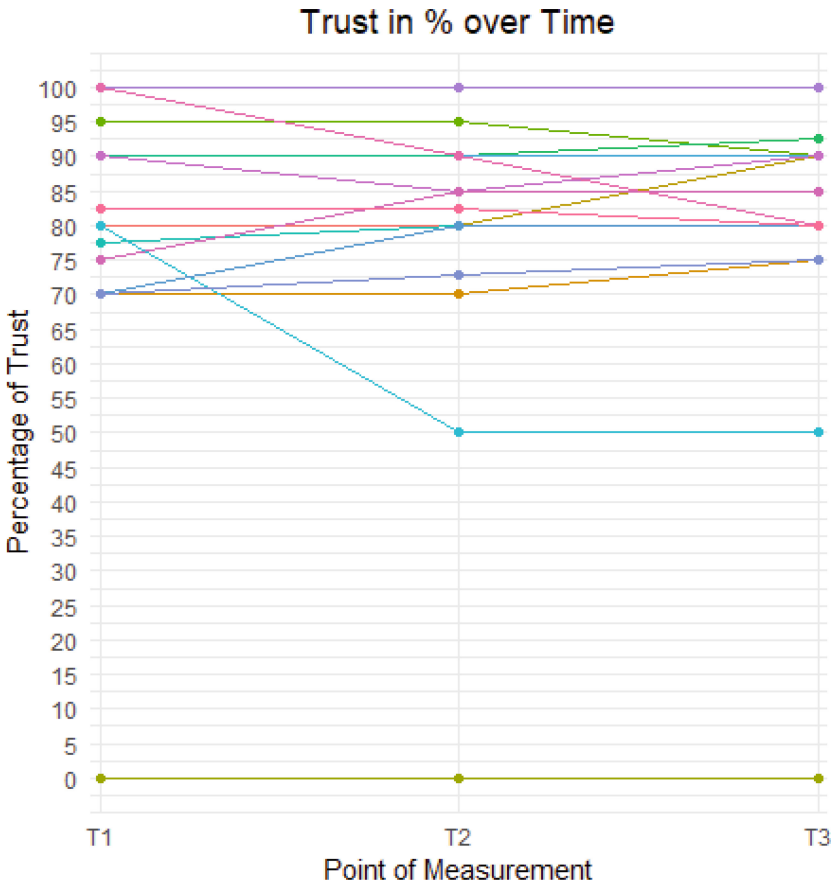


Fig. 5. Changes in trust level over time (T1-T3) for each participant. T1 = Reported trust at picking station I, T2 = reported trust at picking station II, T3 = reported trust at picking station III.

in technologies, particularly in the context of drones, is influenced by pre-existing psychological anchors, and that new interactions alone may not be sufficient to induce significant changes in trust levels.

Analyses of participants' mental models regarding the drone via a questionnaire (Tausch, 2024) showed significant changes across the three measurement time points ($ATS(1.746, \infty) = 16.409, p < .001$; distributional and heteroscedasticity robust repeated measure ANOVA (Friedrich et al., 2023). Results from Welch post-hoc tests indicated that the mental model becomes significantly clearer between the pre-experimental and post-experimental measurements (diff. = 1.044, $p < .001$). Although there was a decrease in clarity of the mental model between the post-experimental and follow-up measures (two weeks after the experiment), this change was statistically not significant (diff. = -0.205, $p = .228$). Nevertheless, the difference between the pre-experimental and follow-up measures remained statistically significant (diff. = -0.839, $p = .002$).

To provide an overview of the relationships between key variables, correlation matrices of the key variables mental model, drone distance, and reported trust are presented in the Appendix (Table A1 and A2).

5 Preliminary Conclusion and Outlook

Based on the results of this pilot study (with all its limitations regarding sampling and sample size), we can summarize that on a cognitive level of the mental model, human-drone interaction leads to positive subjectively perceived changes towards an enhanced understanding of the capabilities (and limits) of the drone and comprehension of collaborative tasks and interaction types. Participants became more confident in judging the capabilities of the drone, but this did not mean that their trust level changed strongly.

At the affective level of trust, we observed that participants, in most cases, do not significantly diverge from the initial trust level. It seems as if there is a personal trust expectancy level around which changes slightly, either positively or negatively, or remains the same. Strong changes and large jumps were not observed in this context.

From a method-related perspective, it can be argued that the interaction period was either too short (around 15 min.) or did not include experiences that were able to disconfirm the individual's expectation in a positive or negative way, or both. Further studies should consider these points, e.g., by including interaction experiences that are suitable for disconfirming expectations.

From a theoretical perspective, based on the results so far and acknowledging the sample size in the study and its implications, the theoretical assumptions of the C/D theory (Oliver, 1980; Buschmeyer et al., 2023) may serve as a useful explanation of processes of trust development and trust level maintenance, respectively. The C/D paradigm, that was originally developed to explain user and/or customer satisfaction, may be suitable to explain trust development and maintenance as a consequence of a conscious or unconscious comparison of the perceived performance of the drone with the expected performance. The result of this comparison is a specific level of trust or distrust. Due to the concrete experiences over time, trust levels may be updated - but not necessarily. Experiences made during the interaction were in most cases perceived as a confirmation of the initial expectation, independent of high or low expectations. We assume that

a kind of confirmation bias occurs, meaning that the initial assessment is more likely to be confirmed rather than fundamentally changed as interaction continues. It appears that assimilation of new experiences occurs rather than accommodation and complete re-evaluation. The behavior of the drone must deviate significantly from the initial trust level to change the trust level substantially. If drone behavior remains within a certain zone around the expectancy level, the trust level does not change.

On the behavioral level, the reduced distance as an indicator of comfort after completing the course, shows an advancement.

All results can be regarded as preliminary and should be replicated in further studies. Nevertheless, the results show the importance of a comprehensive look at cognitive, affective, and behavioral measures of human reactions in trust development in human-technology interaction with new technologies.

Acknowledgements. The project was funded as "Incubator Project" by the Research Center Trustworthy Data Science and Security, one of four research centers within the UA Ruhr that are funded by the Ministry of Culture and Science of the State of North Rhine-Westphalia.

Disclosure of Interests. The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix

Table A1. Kendall's Tau correlation matrix of the key variables mental model, drone distance, and reported trust with corresponding *p*-values of pairwise correlation tests.

	MM | t0	MM | t1	MM | t2	DD | t0	DD | t1	RT | PS1	RT | PS2	RT | PS3
MM | t0	1							
MM | t1	$\tau =$ .406, $p =$ .019*	1						
MM | t2	$\tau =$ .350, $p =$ .073	$\tau =$ .462, $p =$ .017*	1					

(*continued*)

Table A1. (*continued*)

	MM | t0	MM | t1	MM | t2	DD | t0	DD | t1	RT | PS1	RT | PS2	RT | PS3
DD | t0	τ = .038, p = .828	τ = .167, p = .346	τ = .191, p = .330	1				
DD | t1	τ = -.355, p = .048*	τ = -.264, p = .142	τ = -.028, p = .888	τ = .219, p = .230	1			
RT | PS1	τ = .153, p = .390	τ = .262, p = .141	τ = .372, p = .067	τ = .053, p = .770	τ = .055, p = .766	1		
RT | PS2	τ = .192, p = .281	τ = .411, p = .021**	τ = 395, p = .049*	τ = .227, p = .212	τ = ,144, p = .434	τ = .751, p < .001***	1	
RT | PS3	τ = .165, p = .362	τ = .558, p = .002**	τ = .433, p = .033*	τ = .290, p = .118	τ = .206, p = .273	τ = .543, p = .004**	τ = .800, p < .001***	1

Note. Kendall's Tau correlations and corresponding p-values were computed using cor.test() in R with method = 'kendall' and complete observations only (use = 'complete.obs'). No correction for multiple testing was applied. α-level (unadjusted) was set at .05; p < .05*, p < .01**, p < .001***. MM = mental model towards the drone, DD = drone distance, RT = reported trust. t0 = pre-experimental, t1 = post-experimental, t2 = follow-up. PS1 = picking station 1, PS2 = picking station 2, PS3 = picking station 3. τ = Kendalls Tau. Correlations for measures taken on the same scale are given in the grey boxes

Table A2. Kendall's Tau correlation matrix of the key variables mental model, drone distance, and reported trust with corresponding p-values of pairwise correlation tests ordered by measurement time points.

	MM | t0	DD | t0	RT | PS1	RT | PS2	RT | PS3	MM | t1	DD | t1	MM | t2
MM | t0	1							

(*continued*)

Table A2. (*continued*)

	MM | t0	DD | t0	RT | PS1	RT | PS2	RT | PS3	MM | t1	DD | t1	MM | t2
DD | t0	τ = .038, p = .828	1						
RT | PS1	τ = .153, p = .390	τ = .053, p = .770	1					
RT | PS2	τ = .192, p = .281	τ = .227, p = .212	τ = .751, p < .001***	1				
RT | PS3	τ = .165, p = .362	τ = .290, p = .118	τ = .543, p = .004**	τ = .800, p < .001***	1			
MM | t1	τ = .406, p = .019*	τ = .167, p = .346	τ = .262, p = .141	τ = .411, p = .021**	τ = .558, p = .002**	1		
DD | t1	τ = -.355, p = .048*	τ = .219, p = .230	τ = .055, p = .766	τ =,144, p = .434	τ = .206, p = .273	τ = -.264, p = .142	1	
MM | t2	τ = .350, p = .073	τ = .191, p = .330	τ = .372, p = .067	τ = 395, p = .049*	τ = .433, p = .033*	τ = .462, p = .017*	τ = -.028, p = .888	1

Note. Kendall's Tau correlations and corresponding p-values were computed using cor.test() in R with method = 'kendall' and complete observations only (use = 'complete.obs'). No correction for multiple testing was applied. α-level (unadjusted) was set at .05; p < .05*, p < .01**, p < .001***. MM = mental model towards the drone, DD = drone distance, RT = reported trust. t0 = pre-experimental, t1 = post-experimental, t2 = follow-up. PS1 = picking station 1, PS2 = picking station 2, PS3 = picking station 3. τ = Kendalls Tau. Correlations are given in temporal (non-equidistant) order

References

Akula, A.R., Liu, C., Saba-Sadiya, S., Lu, H., Todorovic, S., Chai, J.Y., Zhu, S.-C.: X-ToM: Explaining with Theory-of-Mind for Gaining Justified Human Trust (2019)

Alonso, V., de La Puente, P.: System Transparency in Shared Autonomy: A Mini Review Frontiers in neurorobotics **12**, 83 (2018). https://doi.org/10.3389/fnbot.2018.00083

Awasthi, S., et al.: Micro UAV Swarm for industrial applications in indoor environment: A systematic literature review. Logist. Res. **16**(1), 1–43 (2023)

Bhatnagr, P., Rajesh, A.: Artificial intelligence features and expectation confirmation theory in digital banking apps: Gen Y and Z perspective MD (2024). https://doi.org/10.1108/MD-07-2023-1145

Buschmeyer, K., Hatfield, S., Heine, I., Jahn, S., Markus, A.L.: Expectation management in AI implementation projects: a case study EMJB **18**(3), 441–451 (2023). https://doi.org/10.1108/EMJB-10-2021-0161

Choi, I.Y., Moon, H.S., Kim, J.K.: Assessing Personalized Recommendation Services Using Expectancy Disconfirmation Theory APJIS **29**(2), 203–216 (2019). https://doi.org/10.14329/apjis.2019.29.2.203

de Visser, E.J., et al.: Towards a Theory of Longitudinal Trust Calibration in Human–Robot Teams. Int. J. Soc. Robot. **12**(2), 459–478 (2019). https://doi.org/10.1007/s12369-019-00596-x

Dubey, A., Abhinav, K., Jain, S., Arora, V., Puttaveerana, A.: HACO: A Framework for Developing Human-AI Teaming. In: Jain, S., Gupta, A., Lo, D., Saha, D., Sharma, R. (eds) Proceedings of the 13th Innovations in Software Engineering Conference (formerly known as India Software Engineering Conference). ISEC 2020: 13th Innovations in Software Engineering Conference, Jabalpur India, 27 02 2020 29 02 2020, pp. 1–9. ACM, New York, NY, USA (2020). https://doi.org/10.1145/3385032.3385044

Friedrich, S., Konietschke, F., Pauly, M.: MANOVA.RM: Resampling-Based Analysis of Multivariate Data and (2023)

Gramse, N., Roidl, M., Awasthi, S., Reining, C.: 3.7 Micro-UAV Swarm Testbed for Indoor Applications. In: Morik, K., Rahnenführer, J., Wietfeld, C. (eds.) Applications, pp. 212–224. De Gruyter (2022)

Gupta, P., Woolley, A.W.: Articulating the role of artificial intelligence in collective intelligence: a transactive systems framework. Proc. Hum. Factors Ergon. Soc. Ann. Meet. **65**(1), 670–674 (2021). https://doi.org/10.1177/1071181321651354c

Hancock, P.A., et al.: How and why humans trust: a meta-analysis and elaborated model. Front. Psychol. **14**, 1081086 (2023). https://doi.org/10.3389/fpsyg.2023.1081086

Hoff, K.A., Bashir, M.: Trust in automation: integrating empirical evidence on factors that influence trust. Hum. Factors **57**(3), 407–434 (2015). https://doi.org/10.1177/0018720814547570

Human Factors and Ergonomics Society (HFES): What Is Human Factors and Ergonomics? (2025). https://www.hfes.org/About/What-Is-Human-Factors-and-Ergonomics. Accessed 26 Feb 2025

Kocielnik, R., Amershi, S., Bennett, P.N.: Will you accept an imperfect AI? In: Brewster, S., Fitzpatrick, G., Cox, A., Kostakos, V. (eds.) Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. CHI '19: CHI Conference on Human Factors in Computing Systems, Glasgow Scotland Uk, 04 05 2019 09 05 2019, pp. 1–14. ACM, New York (2019). https://doi.org/10.1145/3290605.3300641

Lee, J.D., See, K.A.: Trust in automation: designing for appropriate reliance. Hum. Factors **46**(1), 50–80 (2004). https://doi.org/10.1518/hfes.46.1.50_30392

Lew, Z., Walther, J.B.: Social scripts and expectancy violations: evaluating communication with human or AI Chatbot interactants. Media Psychol. **26**(1), 1–16 (2023). https://doi.org/10.1080/15213269.2022.2084111

Lyons, J.B.: Being transparent about transparency: a model for human-robot interaction. In: Association for the Advancement of Artificial Intelligence (ed) AAAI spring symposium series. AAAI Spring Symposium 2013 (2013)

Mathieu, J.E., Heffner, T.S., Goodwin, G.F., Salas, E., Cannon-Bowers, J.A.: The influence of shared mental models on team process and performance. J. Appl. Psychol. **85**(2), 273 (2000)

McDermott, P., Dominguez, C., Kasdaglis, N., Ryan, M., Trahan, I., Nelson, A.: Human-machine teaming: systems engineering guide (2018). https://www.mitre.org/sites/default/files/2021-11/prs-17-4208-human-machine-teaming-systems-engineering-guide.pdf

McDermott, P.L., ten Brink, R.N.: Practical guidance for evaluating calibrated trust. Proc. Hum. Factors Ergon. Soc. Ann. Meet. **63**(1), 362–366 (2019). https://doi.org/10.1177/1071181319631379

Mi Alnaser, F., Rahi, S., Alghizzawi, M., Ngah, A.H.: Does artificial intelligence (AI) boost digital banking user satisfaction? Integr. Expectat. Confirmation Model Antecedents Artif. Intell. Enabled Dig. Bank. Heliyon **9**(8), e18930 (2023). https://doi.org/10.1016/j.heliyon.2023.e18930

Miller, L., Kraus, J., Babel, F., Baumann, M.: More than a feeling-interrelation of trust layers in human-robot interaction and the role of user dispositions and state anxiety. Front. Psychol. **12**, 592711 (2021). https://doi.org/10.3389/fpsyg.2021.592711

Müller, L.S., Hertel, G.: Trusting information systems in everyday work events - effects on cognitive resources, performance, and well-being. Ergonomics **68**(1), 19–36 (2025). https://doi.org/10.1080/00140139.2023.2286910

Oliver, R.L.: A cognitive model of the antecedents and consequences of satisfaction decisions. J. Mark. Res. **17**(4), 460–469 (1980)

O'Neill, T., McNeese, N., Barron, A., Schelble, B.: Human-autonomy teaming: a review and analysis of the empirical literature. Hum. Factors **64**(5), 904–938 (2022). https://doi.org/10.1177/0018720820960865

Parasuraman, R., Riley, V.: Humans and automation: use, misuse, disuse, abuse. Urban Stud. **39**(2), 3044–3059 (1997)

Parasuraman, R., Manzey, D.H.: Complacency and bias in human use of automation: an attentional integration. Hum. Factors **52**(3), 381–410 (2010). https://doi.org/10.1177/0018720810376055

Rutinowski, J., Klüttermann, S., Endendyk, J., Reining, C., Müller, E.: Benchmarking trust: a metric for trustworthy machine learning. In: World Conference on Explainable Artificial Intelligence, pp. 287–307 (2024)

Schön, D.A.: Educating the reflective practitioner: toward a new design for teaching and leanring in the professions. Jossey-Bass (1987)

Tausch, A.: Instrument for measuring mental models in human drone interaction (2024)

Feedback from AI Team Members: Implications on Self-image and Trust

Eleni Georganta[1](✉), Anna-Sophie Ulfert[2], and Zhen-Cong Ng[1]

[1] University of Amsterdam, Nieuwe Achtergracht 129, 1018 WS Amsterdam, The Netherlands
e.georganta@uva.nl

[2] Eindhoven University of Technology, Atlas Building, 5600 MB Eindhoven, The Netherlands

Abstract. Establishing trust between humans and intelligent technologies is crucial for a Human-AI Team (HAIT) in the workplace. Trust in a HAIT involves evaluating both the trustworthiness of team members and receiving feedback on one's own trustworthiness. However, this evaluation and feedback process can be affected by inaccuracies due to subjective assumptions or errors leading to negative outcomes. The present study investigated how inaccurate trustworthiness feedback from human or AI team members affects individuals' self-perceived trustworthiness and trust towards the feedback providor. An online scenario study with 511 participants showed that inaccurate trustworthiness feedback led to lower self-perceived trustworthiness compared to accurate trustworthiness feedback. However, no differences in trust in the team member giving the feedback were observed based on the source of the feedback (human vs. AI team member). These findings highlight the importance of inaccurate trustworthiness feedback in HAIT and suggest avenues for future research on trust dynamics in these new types of teams.

Keywords: human-AI teams · AI team member · trust · trustworthiness · feedback

1 Introduction

Numerous organizations are incorporating AI systems into their workplace settings [1], with some even adopting AI technologies as active team members working alongside human employees [2]. This integration leads to the formation of a Human-AI Team (HAIT) [3]. A HAIT consists of both human and AI members working collaboratively and interdependently on complex tasks to achieve team objectives. AI team members, characterized as autonomous systems, collaborate with humans to fulfill shared team goals, requiring a certain level of agency [4]. Establishing trust, which refers to the intention to accept vulnerability to a trustee based on positive expectations of their actions [5], between humans and AI agents is essential for the success of a HAIT in the workplace [6].

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 218–230, 2026.
https://doi.org/10.1007/978-3-032-01377-4_12

Similar to human-only teams, HAIT members evaluate the trustworthiness of their peers based on three criteria: their (perceived) ability, benevolence, and integrity [7]. Ability refers to an individual's skills, competencies, and characteristics that are relevant to the task at hand. Benevolence pertains to an individual's intent—specifically, the extent to which a team member believes that a colleague wishes to act in the team members' best interests and pursues the collective goal in team settings. Integrity is the perception of how well an individual adheres to the set of principles deemed acceptable by the team. At the same time, others evaluate one's own trustworthiness and respond to it [8]. Thus, trust in a HAIT involves not only perceiving the trustworthiness of the other team members based on their actions and behaviors but also receiving feedback and input based on how others evaluate one's own trustworthiness [6]. Hence, within a HAIT, all team members, including AI team members, act both as evaluators of the other team member's trustworthiness and as recipients of trustworthiness evaluations from their team members [8]. However, this evaluation process can be prone to inaccuracies due to subjective assumptions and occasional errors by both human and AI team members. When such misalignment occurs, it creates uncertainty and vulnerability not only towards the other team members and how they are perceived, but also how one perceives oneself. In turn, this can hinder the development of trust and potentially leading to increased tension, distractions, and reduced team performance [9, 10]. Although previous studies have shown that developing appropriate trust between humans and AI team members remains a key challenge [11], research on this topic within HAITs is quite limited.

The goal of the present study is twofold. The first aim is to investigate how accuracy of trustworthiness feedback from a team member—whether human or AI—affects an individual's self-perceived trustworthiness as well as the trust in that team member. Specifically, we examine whether the accuracy of trustworthiness feedback from human and AI team members—defined as either a match (accurate) or a mismatch (inaccurate) between a team member's actual trustworthiness and how it is perceived by another team member—affects an individual's self-perceived trustworthiness, in other words their own subjective assessment of their trustworthiness. Receiving inaccurate feedback that undervalues one's trustworthiness can negatively impact how individuals assess their own reliability. Specifically, according to attribution theory [12, 13], we propose that individuals are more likely to attribute inaccurate trustworthiness feedback to personal shortcomings rather than external factors. This tendency causes them to question their abilities and judgment, leading to reduced confidence and diminished self-perceived trustworthiness than when receiving accurate trustworthiness feedback. Simultaneously, we explore whether the accuracy of trustworthiness feedback from human and AI team members impacts trust toward these team members. When individuals have high expectations regarding their own trustworthiness, they assume that their actions and decisions will be recognized as reliable. However, when they receive feedback that undervalues their trustworthiness (inaccurate trustworthiness feedback), this expectation is violated. According to the violated expectations model [14, 15], such a discrepancy—particularly when perceived as unjustified or misaligned with one's self-view—can lead to negative evaluations of the feedback source (e.g. the human or AI team member) and a reduction in trust.

The second aim is to explore whether the source of the inaccurate trustworthiness feedback—human versus AI team member—leads to different outcomes. Given the differences in perceptions of trust among team members between human-only teams and HAITs [16] and that humans tend to assess more negatively AI's compared to human's decisions [17] often assuming an AI's inconsistent performance [18], we expect differences when receiving inaccurate trustworthiness feedback from a human compared to an AI team member. We specifically examine whether inaccurate trustworthiness feedback has a less negative impact on one's self-image when it comes from an AI team member rather than a human team member. We hypothesize that inaccurate feedback from an AI team member is more likely to be dismissed or discounted by devaluing the discrepant information, compared to feedback from a human team member. Conversely, we investigate whether inaccurate trustworthiness feedback has a stronger negative impact on trust toward the team member when it comes from an AI rather than a human team member. Drawing on the idea that people attribute motives to behaviors—distinguishing between internal personal motives and external circumstantial ones [12]—we argue that inaccurate feedback from an AI team member is more likely to be attributed to an error of judgment on the part of the AI itself. This attribution may lead to reduced trust in the team member being assessed, compared to when the same feedback is received from a human team member.

Overall, we propose the following:

Hypothesis 1: Self-perceived trustworthiness will be lower when receiving inaccurate rather than accurate trustworthiness feedback.

Hypothesis 2: Trust in the team member (human or AI) providing the feedback will be lower when receiving inaccurate rather than accurate trustworthiness feedback.

Hypothesis 3: The negative impact of inaccurate trustworthiness feedback on self-perceived trustworthiness will be weaker when the feedback is given by an AI rather than a human team member.

Hypothesis 4: The negative impact of inaccurate trustworthiness feedback on trust in the team member providing the feedback will be stronger when the feedback is given by an AI rather than a human team member.

This study aims to deepen our understanding of trust in HAITs by examining whether perceptions of oneself, others, and trust relationships vary based on the accuracy of trustworthiness feedback and whether the feedback provider is a human or an AI team member.

2 Method

2.1 Sample and Study Design

To test our hypotheses, we used a between-subjects design. Specifically, we manipulated the type of team member (human or AI) and the accuracy of trustworthiness feedback (accurate of inaccurate). Thus, participants were randomly assigned to four conditions

(type of team member: [human/AI]) × accuracy of trustworthiness feedback: [accurate/inaccurate]) and completed an online scenario-study using the software Qualtrics. The participants did not work in a real team nor did they execute tasks during the experiment; all the information received was part of a hypothetical scenario. The inclusion criteria were being aged 18 or above and having a sufficient English-proficiency level, as the scenarios and questionnaires were written in English. Participants were recruited through the website Prolific.

In total, 881 participants completed the study. During the data processing stage, 358 participants were excluded[1]. The final sample consisted of 511 participants (50.1% female, 48.3% male, 1.4% other; *M*age = 30.32, *SD*age = 9.48), with 240 participants from the AI team member condition and 271 participants from the human team member condition. In the AI team member condition, 119 participants were assigned to the accurate-feedback condition (condition AI-Accurate) and 121 participants to inaccurate-feedback condition (condition AI-Inaccurate). In the human team member condition, 137 participants were assigned to the accurate-feedback condition (condition Human-Accurate) and 134 participants to the inaccurate-feedback condition (condition Human-Inaccurate).

2.2 Study Procedure

After receiving ethical approval from the first and third author´s institution to run the present study, participants were recruited via Prolific. After being informed about the purpose of the study, voluntariness, data anonymity and storage, all the participants completed four study parts. First, they were presented with a fictional scenario in which they were part of a company combining the skills of humans and AI technologies. The participants were informed that they were part of a HAIT, which consisted of different experts and which was developing a fitness app. They were also informed that depending on the task during the app development, they often worked in pairs with one other member of their team. Second, participants received their profile as members of this HAIT, having the role of the designer. In their profile, they were described as highly trustworthy, with high levels of ability (e.g. highly skilled), integrity (e.g. just and fair), and benevolence (e.g. caring for co-workers). After reading their profiles, we assessed their self-perceived trustworthiness.

Third, participants were informed that they received an email from a colleague, with whom they are having a meeting the next day. This email included feedback points to be included in their upcoming project. Depending on the participant´s condition, participants received feedback either from an AI or a human team member. This feedback

[1] Participants were excluded due the following reasons: 24 participants spent less than seven minutes on the study; one participant from the human teammate condition did not provide their consent; 126 participants failed the attention checks (wrong answer when asked to select "strongly disagree"); 196 remembered their type of team member incorrectly; 12 more participants were excluded as they rated the scenario realism to be 1; no additional participants were excluded after checking for outliers.

was either accurate (conditions AI-accurate and human-accurate) or inaccurate (conditions AI-inaccurate and human-inaccurate) in reflecting the high level of trustworthiness described in the participant's profile, which they have read just before receiving the feedback. After reading this email, we assessed self-perceived trustworthiness, trust in the team member that provided the feedback, and asked participants to complete two attention checks (i.e., "Who is the App Developer Expert in your team?" and "What is the hierarchical level of the App Developer Expert in the team?"). Fourth, participants were informed that the study scenario was over. Then, we assessed demographics and some additional control variables and at the end, we thanked participants for their time.

The study lasted an average of 15 min, and participants were compensated at a rate of £9.11 per hour via Prolific.

For study material (profile, manipulation of accuracy of trustworthiness feedback, and manipulation of type of team member) see Appendix.

2.3 Measures and Materials

Accuracy of Trustworthiness Feedback. The trustworthiness feedback was manipulated via the information provided in the email that the participant received from their team member. This information was either accurately aligned (i.e., accurate trustworthiness feedback) or inaccurately aligned (i.e., inaccurate trustworthiness feedback) with the high level of trustworthiness described in the participant's profile as part of the study scenario. In line with the literature showing that trustworthiness is a multifaceted construct comprising three dimensions – ability, integrity and benevolence [7] – the email, consisted of one passage per dimension (i.e. skills, values and working with others, with each described as either high (i.e. accurate trustworthiness feedback) or low (i.e. inaccurate trustworthiness feedback).

Self-perceived Trustworthiness. Self-perceived trustworthiness was assessed with the 15-item trustworthiness scale from Jarvenpaa et al. [19]. The items were modified so that the self-perceived trustworthiness was assessed replacing "my team member" with "I" (e.g., "I have much knowledge about the work that needs to be done."), showing good scale reliability ($\alpha = .85–.90$). Responses were given using a 5-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree).

Trust in the Team Member. Trust in the team member providing feedback was assessed using twelve items, measuring both cognitive and affective trust. These components were included due to their established relevance in HAIT research [16]. Six items assessed the cognitive (e.g., "Given my team members' track records, I see no reason to doubt their competence and preparation for the project"), and six items assessed the affective component of trust (e.g., "I can talk freely to my team members about the difficulties I am having with the project and know that they will want to listen"). The scale including all items demonstrated good reliability ($\alpha = .94$), with responses given using a 5-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree).

Type of Teammate. The type of teammate was manipulated based on whether the email participants received was from a human or an AI team member.

With regards to participant demographics, age, gender, nationality, education, and employment status were assessed. Finally, we also assessed participants general trust in technology using three items (e.g., "My typical approach is to trust new information technologies until they prove to me that I shouldn't trust them") drawn from Disposition to trust technology scale [20], using using a 7-point Likert scale ranging from 1 (strongly disagree) to 5 (strongly agree), and participant's experience with AI using a self-developed item ("To what extent do you have experience with using AI in general? Examples of AI are Chat GPT, Siri, and Grammarly." using a 5-point Likert scale ranging from 1 (no experience) to 7 (lots of experience).

3 Results

All the analyses were performed with IBM's SPSS Statistics 29.

3.1 Preliminary Analysis

Before testing our hypotheses, we wanted to check whether participants perceived themselves as highly trustworthy after reading their profiles and before receiving feedback from their team members. Across conditions, self-perceived trustworthiness was very high after participants read their profiles, with no significant differences between the conditions ($F(3,510) = .17$, $p = .912$; $M = 4.71$, $SD = 0.35$ for AI-Accurate, $M = 4.70$, $SD = 0.36$ for AI-Inaccurate, $M = 4.70$, $SD = 0.31$ for Human-Accurate, $M = 4.68$, $SD = 0.33$ for Human-Inaccurate). Furthermore, we found no significant differences between conditions with regards to gender ($F(3,510) = 1.03$, $p = .377$), age ($F(3,510) = 1.09$, $p = .350$), as well as with regards to the variables general trust in technology ($F(3,510) = .53$, $p = .656$) and experience with AI ($F(3,510) = .82$, $p = .483$).

Means, standard deviations and correlations between the study variables are presented in Table 1.

3.2 Hypothesis Testing

To test the impact of the accuracy of trustworthiness feedback on self-perceived trustworthiness (Hypothesis 1) and trust in the team member (Hypothesis 2), we run independent sample t-tests. As expected, receiving inaccurate compared to accurate trustworthiness feedback led to participants indicating lower values of self-perceived trustworthiness, $t(509) = 14.89$, $p < .001$, and lower values of trust in the team member giving the feedback, $t(509) = 15.09$, $p < .001$. Thus, Hypothesis 1 and 2 were supported.

To test whether the source of the inaccurate trustworthiness feedback—human versus AI team member— changed the impact of inaccurate trustworthiness feedback on self-perceived trustworthiness (Hypothesis 3) and trust in the team member (Hypothesis 4), we run two moderation analysis using Model 1 from the PROCESS SPSS macro (Hayes, 2013). In contrast to our expectations, the negative impact of inaccurate trustworthiness

Table 1. Descriptive statistics and intercorrelations of the study variables.

Variables	*M*	*SD*	1	2	3	4	5	6	7	8	9
1. Accuracy of Trustworthiness Feedback	1.50	0.50	-								
2. Type of Team Member	1.53	0.50	-.01	-							
3. Self-perceived trustworthiness (before feedback)	4.70	0.33	-.02	.00	(.84)						
4. Self-perceived trustworthiness (after feedback)	4.22	0.83	-.55**	.00	.32**	(.95)					
5. Trust in the Team Member	3.84	0.86	-.55**	.11*	.16**	.51**	(.94)				
6. General Trust in Technology	5.21	1.16	.05	.02	.18**	.05	.18**	(.74)			
7. Experience with AI	3.68	0.93	-.01	.06	.04	.03	.11*	.26**	-		
8. Gender	1.54	0.54	-.01	-.02	.10*	.01	.01	-.04	-.11*	-	
9. Age	30.32	9.48	.06	-.04	.17**	.04	-.09*	.00	-.28**	.00	-

Notes. * $p < .05$; ** $p < .001$; sample size is 511 individuals; accurate trustworthiness feedback is coded as 1 and inaccurate trustworthiness feedback is coded as 2; human team member is coded as 1 and AI team member is coded as 2; the Cronbach's alpha reliability values are displayed on the diagonals of the correlation matrix.

feedback on self-perceived trustworthiness did not differ significantly when the feedback was given by an AI than a human team member, $b = 0.06$, 95% CI [−0.173, 0.309]. Similarly, the negative impact of inaccurate trustworthiness feedback on trust in the team member did not significantly differ when the feedback was given by an AI than a human team member, $b = -0.08$, 95% CI [−0.328, 0.168]. Thus, Hypothesis 3 and 4 were not supported.

Means, standard deviations for self-perceived trustworthiness and trust in the team member for each condition are presented in Table 2.

Table 2. Means and standard deviations for self-perceived trustworthiness and trust in the team member for each condition.

	Conditions											
	Accurate Trustworthiness Feedback						Inaccurate Trustworthiness Feedback					
	AI- and human-Accurate		AI-Accurate		Human-Accurate		AI- and human-Inaccurate		AI-Inaccurate		Human-Inaccurate	
Variables	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*	*M*	*SD*
Self-Perceived Trustworthiness	4.68	0.39	4.64	0.46	4.71	0.31	3.76	0.89	3.69	0.87	3.83	0.91
Trust in the Team Member	4.32	0.52	4.19	0.59	4.42	0.42	3.36	0.87	3.28	0.86	3.43	0.87

Notes. The sample size is 256 participants in the conditions AI- and Human-Accurate, with 119 participants in the condition AI-Accurate and 137 participants in the condition Human-Accurate, and 255 participants in the conditions AI- and Human-Inaccurate, with 121 participants in the condition AI-Inaccurate and 134 participants in the condition Human-Inaccurate

4 Discussion

4.1 Implications for Research

To provide empirical insights into trust in HAITs, this study examined whether the accuracy of trustworthiness feedback—given by either a human or an AI team member—affects self-perceived trustworthiness and trust in the feedback provider. Additionally, it explored whether the impact of inaccurate feedback differs depending on whether it comes from a human or an AI team member.

In line with our expectations, we found that trustworthiness feedback resulted in less self-perceived trustworthiness when it was inaccurate compared to accurate. This result is in line with previous research stating that discrepancies between self-perception and external evaluations can harm an individual's confidence in their trustworthiness [21]. This means that when individuals receive incorrect assessments about their trustworthiness from both human and AI team members, they seem to doubt their own abilities and judgment. Furthermore, we showed that trustworthiness feedback led to less trust in the team members providing the feedback when this feedback was inaccurate compared to accurate. This aligns with previous theoretical work suggesting that trust in other team members also depends on how much you perceive they trust you as a member [8].

Contradicting our assumptions, the type of team member did not moderate the impact of inaccurate trustworthiness feedback on self-perceived trustworthiness and trust in the team member giving the feedback. Discrepancies between external evaluations and self-perception seemed to harm one's confidence in their trustworthiness as well as their interpersonal relationships with the team member providing the feedback, no matter whether this was an AI or a human, potentially perceiving it as unfair or unkind [22]. These results challenge previous studies suggesting that trust development differs between human-only teams and HAITs [16]. Prior empirical evidence may have shown that humans as trustors trust more their human than their AI team members [23]. However, in the present study, no trust differences were found when humans as trustees were

evaluated by human versus AI team members. Future research should explore the extent to which existing trustor, trustee and trust dynamics literature can be directly applied to HAITs or requires refinement and adjustment.

4.2 Limitations and Next Steps

While this study provides valuable initial insights, several limitations should be acknowledged when interpreting the findings. One of the most important limitations is that it was a scenario study. Imagining to have an AI team member might have been too unfamiliar for the participants, making it difficult for them to perceive this scenario as even possible in the future. The present study should be replicated with stronger manipulations (e.g. Wizard of Oz approach) or interactions with simplistic AI team members (e.g. adjusted ChatGPT). Furthermore, the variables were captured only via self-reports, which are susceptible to bias, and may not be fully representative of an individual's beliefs. In the future, similar studies should incorporate tasks, where behaviors and reactions towards team members can be observed and captured. Finally, many important variables that impact the trust relationships between humans as well as human and AI agents, such as AI agent and team characteristics were neglected. Previous meta-analyses of trust in AI agents and automated systems have suggested that the characteristics of the system (embodiment, anthropomorphism, etc.) strongly impact trust [8, 24, 25]. Other studies have also indicated that trust is much more important and impactful in contexts where trust behaviors can cause greater consequences [26] and that incorporating context is necessary to differentiate the consequences of trust from behavioral intentions. These and additional factors should be considered when conducting similar studies in the future.

5 Conclusion

Building trust between humans and intelligent technologies is essential for the effective functioning and success of a HAIT within the organizational setting. Trust within HAITs is shaped not only by perceptions of the trustworthiness of team members but also by the feedback and evaluations of one's own trustworthiness. The accuracy of trustworthiness feedback, whether from human or AI teammates, can influence the development of trust and potentially the subsequent outcomes. Indeed, our study revealed that inaccurate compared to accurate trustworthiness feedback resulted in lower self-perceived trustworthiness, but the source of the feedback—human or AI—did not significantly impact those relationships. These findings emphasize the importance of trustworthiness feedback in HAIT relationships. To enhance feedback quality, future work should explore systems for real-time performance-based feedback, personalized AI calibration, and training for human team members to interpret AI feedback effectively. Further research is also needed to assess whether existing trust dynamics and feedback literature from human interactions can be applied to HAITs or requires refinement.

Acknowledgments. The present study was funded by the Society for Industrial and Organizational Psychology (SIOP) Foundation as part of the Visionary Circle.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

Appendix

Profile for Designer Team Member

Your role and expertise as an App Design Expert

You are one of the best App Design Experts in your field. You are a team member who participates in decision-making and fulfils a distinct role that contributes to team performance. Your role and related expertise serve as a crucial asset in the team, which aids the further development of the fitness app. Thanks to your expertise, the milestones overcome in the app development highlight the potential benefits towards the company if the team's ideas get CEO endorsement and funding.

There are three main components that are reviewed in the feedback forms of your company. From your own perception and previous feedback you have gotten the following insights about yourself:

Skills
You have been a designer in demand and have worked on multiple successful projects. With the help of additional training, your skillset keeps expanding and you feel confident in your capabilities. Previous feedback rounds from your peers have strengthened your confidence in your capabilities. Co-workers often ask you for your feedback on their work or advice regarding new design projects. Often, you are given high-responsibility roles for important projects.

Values
Justice and fairness are fundamental to you. You believe it is important to stay honest when working with your colleagues and work in a reliable way where you keep the promises, e.g., about deadlines, you have made. From your previous team members, you have received feedback that you are very consistent with your work and that they feel they can rely on you to fulfil your responsibilities.

Working With Others
You are caring for your co-workers and team members. When collaborating, you tend to take all team members´ interests and goals into account and try to find the best solution for everyone involved. You are a real team player who likes to help others to see them grow. If you see someone struggling with a task, you tend to take time out of your own schedule to help them out.

From your colleagues, you have received feedback of them feeling seen and heard by you. They believe you don't only take your own interests into account but also theirs when making a decision and a mutually beneficial relationship has oftentimes been created.

Manipulation for Type of Team Member (AI/Human)

Tomorrow your project continues. For the next project step, you collaborate with the *AI/human* team member who functions as an App Developer Expert. Your team is focused

on building a new feature requiring both your design skills and the other *AI/human* team member's technical expertise. Before tomorrow's meeting, the *AI/Human* App Developer Expert sent you an E-mail summarizing ideas and thoughts about your collaboration in the next steps of the project.

Manipulation for Accuracy of Trustworthiness Feedback (accurate/inaccurate)

Dear App Design Expert,

I hope this E-mail finds you well.

I have reflected on our previous collaboration and would like to suggest the following for our next project.

From now on, I believe it would be important that:

We keep consulting each other on a regular basis to ensure that I can provide the right technical requirements according to your design choices for the new feature. I will give feedback on your designs to further improve the performance of the functionalities./ We should integrate more feedback loops to ensure that your design choices align with the technical requirements of the feature, as I am not sure about the performance of the functionalities that you have designed so far. Whenever I have time, I will suggest changes to your designs to ensure that the new feature will align with the rest of the application and function well.

Furthermore, I appreciate your high level of transparency and communication with me in the previous parts of the project. You have kept me well-informed about your design decisions and asked me for my input when needed. This is why I would like our collaboration to remain the same./ Furthermore, I have noticed that in the past, your design choices were not always consistent and you were not always able to keep the discussed deadlines. For the next part of the project, I will set deadlines for your final decisions to ensure we can work effectively.

Finally, I have noticed that you make design decisions that prioritize the user's and the team's needs and preferences, therefore I will commit to developing the app in the most user-friendly and accessible way possible. I will try to anticipate your preferences as a designer and suggest changes that align well with your design style./ Finally, regarding your design choices, I felt you tend to prioritize commercial interests over the user's and the team's needs. I would suggest focusing more on usability and making the app more accessible to everyone.

I am looking forward to our meeting tomorrow.

References

1. Latto, C., Richter, A., Tate, M.: Human-AI teams' impact on organizations – a review. In: Proceedings of the 58th Hawaii International Conference on System Sciences, pp. 152–161. HICSS, Hawaii (2025)
2. Larson, L., DeChurch, L.: Leading teams in the digital age: four perspectives on technology and what they mean for leading teams. Leadersh. Q. **31**(1), 1–18 (2020)
3. Schmutz, J.B., Outland, N., Kerstan, S., Georganta, E., Ulfert, A.-S.: AI-teaming: redefining collaboration in the digital era. Curr. Opin. Psychol. **58**, 101837 (2024)

4. O'Neill, T., McNeese, N., Barron, A., Schelble, B.: Human-autonomy teaming: a review and analysis of the empirical literature. Hum. Factors **64**(5), 904–938 (2022). https://doi.org/10.1177/0018720820960865
5. Breuer, C., Hüffmeier, J., Hertel, G.: Does trust matter more in virtual teams? A meta-analysis of trust and team effectiveness considering virtuality and documentation as moderators. J. Appl. Psychol. **101**(8), 1151 (2016)
6. McNeese, N.J., Demir, M., Chiou, E.K., Cooke, N.J.: Trust and team performance in human-autonomy teaming. Int. J. Electron. Commer. **25**(1), 51–72 (2021)
7. Mayer, R.C., Davis, J.H., Schoorman, F.D.: An integrative model of organizational trust. Acad. Manag. Rev. **20**(3), 709–734 (1995)
8. Ulfert, A.-S., Georganta, E., Centeio Jorge, C., Mehrotra, S., Tielman, M.: Shaping a multidisciplinary understanding of team trust in Human-AI teams: a theoretical framework. Eur. J. Work Organ. Psychol., 1–14 (2024)
9. De Dreu, C.K.W., Weingart, L.R.: Task versus relationship conflict, team performance, and team member satisfaction: a meta-analysis. J. Appl. Psychol. **88**(4), 741–749 (2003)
10. Jehn, K.A., Mannix, E.A.: The dynamic nature of conflict: a longitudinal study of intragroup conflict and group performance. Acad. Manag. J. **44**(2), 238–251 (2001)
11. Ulfert, A.-S., Georganta, E.: A model of team trust in human-agent teams. In: ICMI Workshop on Insights into Group and Team Dynamics Proceedings, pp. 171–176. ACM (2021)
12. Dirks, K.T., de Jong, B.: Trust within the workplace: a review of two waves of research and a glimpse of the third. Ann. Rev. Organ. Psychol. Organ. Behav. **9** (2022)
13. Hewett, R., Shantz, A., Mundy, J., Alfes, K.: Attribution theories in human resource management research: a review and Research Agenda. Int. J. Hum. Resour. Manage. **29**(1), 87–126 (2018)
14. Gollwitzer, M., Thorwart, A., Meissner, K.: Psychological responses to violations of expectations. Front. Psychol. **8**, 335164 (2018)
15. Rief, W., Glombiewski, J.A., Gollwitzer, M., Schubö, A., Schwarting, R., Thorwart, A.: Expectancies as core features of mental disorders. Curr. Opin. Psychiatry **28**(5), 378–385 (2015)
16. Georganta, E., Ulfert, A.S.: My colleague is an AI! Trust differences between AI and human teammates. Team Perform. Manage.: Int. J. **30**(1/2), 23–37 (2024)
17. De Visser, E.J., et al.: A little anthropomorphism goes a long way: effects of oxytocin on trust, compliance, and team performance with automated agents. Hum. Factors **59**(1), 116–133 (2017)
18. Jones-Jang, S.M., Park, Y.J.: How do people react to AI failure? Automation bias, algorithmic aversion, and perceived controllability. J. Comput.-Mediated Commun. **28**(1), zmac029 (2023)
19. Jarvenpaa, S.L., Knoll, K., Leidner, D.E.: Is anybody out there? Antecedents of trust in global virtual teams. J. Manag. Inf. Syst. **14**(4), 29–64 (1998)
20. McKnight, D.H.: Trust in information technology. Blackwell Encycl. Manage. **7**, 329–331 (2005)
21. Campagna, R.L., Dirks, K.T., Knight, A.P., Crossley, C., Robinson, S.L.: On the relation between felt trust and actual trust: examining pathways to and implications of leader trust meta-accuracy. J. Appl. Psychol. **105**(9), 994 (2020)
22. Baer, M.D., Frank, E.L., Matta, F.K., Luciano, M.M., Wellman, N.: Undertrusted, overtrusted, or just right? The fairness of (in)congruence between trust wanted and trust received. Acad. Manag. J. **64**(1), 180–206 (2021)
23. Georganta, E., Ulfert, A.-S.: Would you trust an AI team member? Team trust in human–AI teams. J. Occup. Organ. Psychol. **97**(3), 1212–1241 (2024)

24. Kaplan, A.: Artificial Intelligence (AI): when humans and machines might have to coexist. In: Verdegem, P. (ed.) AI for Everyone?: Critical Perspectives, pp. 21–32. Westminster University Press, London (2022)
25. Schaefer, K.E., Chen, J.Y.C., Szalma, J.L., Hancock, P.A.: A meta-analysis of factors influencing the development of trust in automation: implications for understanding autonomy in future systems. Hum. Factors **58**(3), 377–400 (2016)
26. Lyons, J.B., et al.: Comparing trust in auto-GCAS between experienced and novice air force pilots. Ergon. Des.: Quart. Hum. Factors Appl. **25**(4), 4–9 (2017)

Statistical Model Checking

Using Statistical Model Checker for Schedulability Analysis of Real-Time Systems Under Uncertainty

Josef Strnadel(✉)

Brno University of Technology, Bozetechova 2, 61200 Brno, Czech Republic
strnadel@fit.vut.cz
https://www.fit.vut.cz/person/strnadel/

Abstract. Schedulability analysis aims to decide whether it is possible to meet the timing constraints of the given subset of real-time tasks that will be scheduled by the given policy and executed on the given platform. For some situations (classes of systems and conditions), a guaranteed/proven schedulability analysis method exists. As such a method may be absent for other situations and there may be a practical need to cope with the schedulability analysis in such problematic (but realistic, burdened with uncertainty) situations, the analysis must rely on alternative means. This paper presents some of the situations (related, e.g., to the drift of a clock and OSTime, event/interrupt management, task design patterns or OS components such as a scheduler and task queues) that represent a problem for guaranteed approaches first. Then, it presents our approach for interruptible CPU-based systems; it builds on a simulation model over a network of stochastic timed automata, an instrument able to cope with such situations. To analyze schedulability, we apply the statistical model-checking technique to our model. The technique has already shown to easily scale to complex dynamic systems as well as to efficiently solve various problems. Our results indicate that the model and the technique are appropriate for schedulability analysis in realistic situations, where the computational complexity and result of the analysis are driven by predefined parameters such as the degree of confidence.

Keywords: Real-time · Task · Uncertainty · Schedulability · Model · Analysis · Simulation · Timed automaton · Statistical model checking · Processor · Exception · Operating System

1 Introduction

To meet expectations placed upon a *real-time* (RT) system, the system must be specified, modeled, and analyzed precisely for various conditions; such a need represents a problem to be solved in the early stages of development. Basically,

This work was supported by the project Application-Specific HW/SW Architectures and Their Applications – FIT-S-23-8141.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 233–256, 2026.
https://doi.org/10.1007/978-3-032-01377-4_13

the solution is facilitated by introducing a set of parameterized RT *tasks* and doing a *schedulability analysis* (SAN) over the set to evaluate the level of meeting the expectations (e.g., timing constraints) imposed on the system, ranging from guaranteed through best-effort to unfeasible. On top of the set of RT tasks, more complex solutions may involve information about preemptivity/migrability of tasks, dependence among tasks, policies used to schedule tasks and assign priorities to tasks, overheads of managing/executing tasks using a particular *operating system* (OS) etc. For some situations, a proven SAN method exists. As such a method may be unavailable for many other (often realistic) situations, this paper approaches the SAN problem from a practical perspective.

1.1 Scope and Structure of Paper

Especially, this paper focuses on situations that, if measuring their characteristic data (such as event occurrence/handling times, stack/queue utilization, or CPU load), involve some kind of uncertainty, i.e., each data can be, if identified, seen as a random variable the measurement of which is loaded by an error given by a confidence interval. Among the others, uncertainty may originate from facts such as harsh environment, stress/degradation of material, faults/errors/anomalies, signal propagation latency, drift of a digital clock, digitization effects, lack of energy, cache misses, pipeline flushes and interrupt requests. For example, *interrupt requests* (IRQs) may occur at arbitrary times, suffer from arrival/servicing jitters, be subject to priorities, some can be un/masked, and IRQ handlers can be nested at run-time. Mostly, such uncertainty-induced behavior can only be described stochastically and results in the explosion of situations to be considered during SAN. To cope with that, we propose a simulation model (of an RT system, RTS) that is built on a network [22] of *stochastic timed automata* (STA), an overclass of *timed automata* (TA), capable of expressing various aspects with regard to uncertainty. To analyze the schedulability of a RT task set, we use the *statistical model checking* (SMC) technique [1].

The paper is organized as follows. Section 2 outlines a closer specification and a background of our research with regard to this paper; it covers topics such as an RT system/task, sources of uncertainty in realistic situations, conventional approaches to modeling/analysis of RT systems, state-of-the-art in the SAN area, and reasoning/instruments of our research. Section 3 presents our (realistic) STA based model as well as SMC based approach to the SAN problem. Section 4 presents our representative results and Sect. 5 concludes the paper.

2 Research Background

2.1 RT Systems and Tasks

Simply put, a RTS must operate correctly from both the logical ("what") and temporal ("when") ways of control. While the former concerns the control flow and data management used to realize desired actions, the latter concerns determining when the actions take place.

Traditionally, a *action* is performed by a computational unit called an RT *task* (τ); based on Fig. 1, we distinguish the following classes of tasks:

AL tasks released just after the startup of a (RT)OS,
OF tasks released after the startup of a (RT)OS, e.g., from an IRQ handler,
1S tasks that run a sequence of commands and end afterwards, e.g., by deletion
CY tasks that enter a cycle (loop) whose iteration contains a blocking call,
PE tasks released periodically,
SP tasks released at the earliest after a predefined time has elapsed,
AP tasks released anytime,
PR tasks suspendable externally when a higher-priority execution becomes ready,
NPR tasks that run until they allow suspension,
DE tasks synchronized with other tasks,
IDE tasks independent of other tasks.

We prepared Fig. 1 both to give a reader an idea about the classes of tasks used in practice and to emphasize what classes we would like to incorporate into our approach (details to which can be found, e.g., in Sect. 3.1, especially in its Listing 1.2, Fig. 8). To the best of our knowledge, existing approaches are mostly academical/theoretical and do not reflect various practical facts. For example, a task can be configured/implemented specifically under an RT(OS), e.g., from the AL, 1S, CY classes), data structures OS uses to manage ready tasks (e.g., *tskQ* in Listing 1.2, Fig. 6), a way in which hardware works (see, e.g., Fig. 4, 7) and an arbitrary digital clock drifts [27], etc. Instead, the approaches try to abstract from some known mechanisms, trying to replace them by a simple model (for an overview, see, e.g., Sect. 2.3) used to express their effect in a computationally simple way, e.g., by means of a random variable.

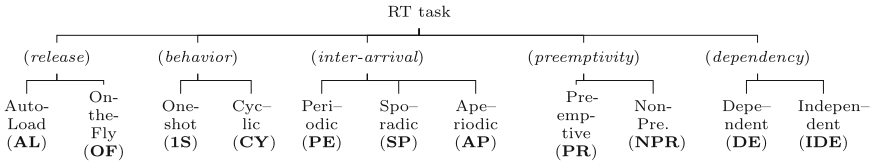


Fig. 1. Classification of RT tasks based on selected criteria.

To cope with timeliness, an RT task is characterized by *static* and *dynamic* parameters [9,14]. Static ones involve, e.g., the *worst-case execution time* (WCET, C), *priority* (P), *timing constraints* such as a relative deadline (D) and inter-arrival *period* (T), if applicable, and (partial, task's) *CPU utilization factor* ($u = \frac{C}{T}$). Dynamic parameters involve, at least, the *release time* (r) – needed, e.g., to evaluate the *absolute deadline* ($d = r + D$) –, execution *start* (s) and *end* (e) times, *response time* ($R = e - r$) or, *time to deadline* ($D(t) = d - t$) and *time to complete* a task ($C(t) = e - t$) at t. For an illustration, see Fig. 2a. At

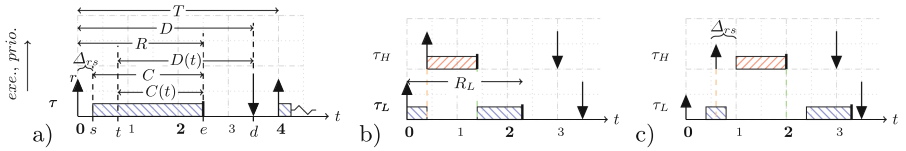


Fig. 2. An illustration to a) parameters of a task τ, b) prolongation of the response time of a low-priority task τ_L due to the execution of a high-priority task τ_H, c) delayed start of τ_H due to time needed to detect the release of τ_H, then suspend τ_L and finally, switch the context to τ_H.

a time, multiple tasks may be released and thus *ready* (to be executed). If tasks share an execution unit, they must be serialized by an arbitrator, called a *scheduler*. A scheduler makes its decision based on *scheduling policy* [9,14], where the policy "works well" for a predefined class of tasks/conditions such as a class of independent preemptive periodic tasks. Its decision results into a *schedule* (e.g., Fig. 2b, c). For a more formal, but simplified definition, see Definition 1.

Definition 1 (Basic notation). *Let $\Gamma = \{\tau_1, \tau_2, \ldots, \tau_n\}$ denote a set of n RT tasks, $\Phi = \{\phi_1, \phi_2, \ldots, \phi_m\}$, denote a set of m computational resources for executing the tasks and $\xi_{\Gamma,\Phi} \in \xi$, where $\xi = \{RR, RM, DM, EDF, \ldots\}$, denote a scheduling policy used to schedule Γ using Φ.*

The process of checking the timing correctness of an RTS typically comprises two kinds of analysis: *timing analysis* (TAN) and consequent *schedulability analysis* (SAN). While TAN tries to characterize the amount of time, e.g., WCET, a task needs to finish on a given platform [40], this paper relates to SAN [9,14,36]; based on TAN results, SAN is used to check whether timing constraints can be met under given conditions. (see Definition 2).

Definition 2 (SAN problem). *Given Γ and Φ, a schedule is called feasible $\Leftrightarrow$ $(\forall\tau)[\tau \in \Gamma \wedge (\tau$ is executed using Φ and timing constraints of τ are met$)]$. Γ is called schedulable $\Leftrightarrow$ $(\exists\xi_{\Gamma,\Phi})[\xi_{\Gamma,\Phi} \in \xi \wedge (a$ feasible schedule results from $\xi_{\Gamma,\Phi})]$. The scheduling problem is defined as the problem of finding a feasible schedule for predefined Γ, Φ and ξ. A method (called a schedulability test too) able to decide whether Γ is schedulable (using ξ, Φ) is called schedulability analysis.*

As the *scheduling problem* is NP complete, finding its solution by a scheduler at runtime can take much longer than it is acceptable in a particular application. If so, the solution must be found before rather than at runtime.

2.2 Uncertainty

This paper deals with the SAN problem in realistic situations, i.e. reflecting various *sources of uncertainty* [41]; due to very limited space in this paper, we skip an introduction to facts such as the *components/types* of uncertainty or the so-called *randomness classes* detailed, e.g., in [23]. A source of uncertainty can

contribute in a specific way to various effects at particular abstraction levels of an RT system. For example, at the task level uncertainty can cause undesired effects such as jitter of release/response times due to drift of a digital clock, signal propagation jitter, processing irregular interrupt requests, material stress, faults/errors, or lack of energy. Apparently, it seems almost impossible to consider all of the sources, so one must limit SAN, e.g., as we did in this paper.

For example, the delay Δ_{rs} from Fig. 2a,c may result from both the latency that can be measured precisely, such as the duration of context load/store operations on a particular platform, and the latency specific for a particular context. Figure 3a, b illustrate how the execution of a low-priority task (τ_L), can be affected by a high-priority task (τ_H), and an IRQ handler used to service the *tick* of an (RT)OS. The illustration (non-realistically) assumes that the digital clock does not drift and $tick_{IRQ}$ is the only IRQ, but it (realistically) illustrates that the time of its handling may vary, e.g., due to the state of pipeline/caches at the time of arising the request.

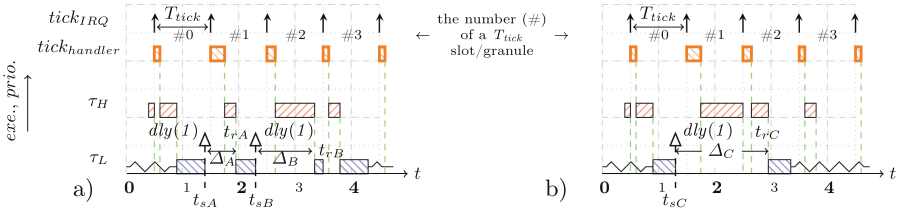


Fig. 3. An illustration to the task resume jitter at the level of τ_L after calling $dly(1)$ used to delay (i.e., block) its caller for one T_{tick}, ideally: a) τ_L may resume sooner/later – at t_{rA}, t_{rB}, resp. – than in T_{tick} units of time, but still in the T_{tick} slot/granule that follows the call – made at t_{sA}, t_{sB}, resp. –, b) due to an excessive activity at higher priority levels, τ_L may resume much later (at t_{rC}) or never after the call (made at t_{sC}); a zigzag line represents an uninteresting fragment of code execution.

Figure 4 illustrates the uncertainty that arises from processing IRQs and key components of latency with respect to IRQs. IRQs may occur at arbitrary times (e.g., asynchronously to a digital clock used by a CPU), suffer from arrival/servicing jitters, be subject to priorities, maskable IRQs can be un/masked and IRQ handlers can be nested at runtime; moreover, their management needs some resources (such as the CPU, stack) used by tasks too. Mostly, similar behavior can only be described stochastically and results in the explosion of situations to be considered during SAN.

2.3 State of the Art

Various approaches to SAN under uncertainty have already been published. Many of them proposed an ad hoc (RT task) model to substitute the lack of information about sources of uncertainty, their effects, etc. We distinguish the following classes of approaches:

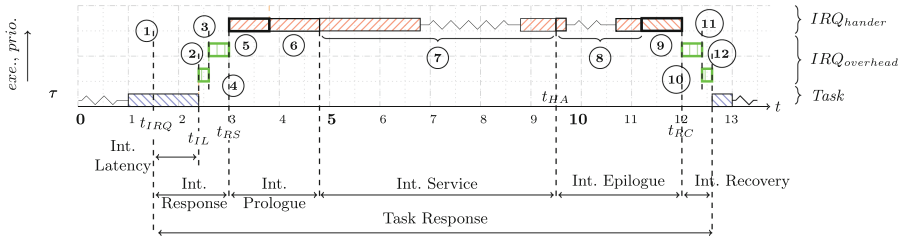


Fig. 4. An illustration to uncertainty arising from processing an IRQ. 1: at t_{IRQ} an IRQ occurs, is recognized and CPU is signalized to suspend the recent task-level execution; 2: at t_{IL} implicit CPU context saving starts, followed by the IRQ arbitration and vector fetching at 3; 4: at t_{RS} the IRQ handler starts; 5: explicit CPU context saving starts; 6: OS gets informed about entering the IRQ handler level; 7: handling of pending IRQs; 8: OS gets informed about exiting the IRQ handler level in order to decide about the consequent control-flow; 9: explicit CPU context restoring starts; 10: at t_{RC} the IRQ handler ends; 11: explicit CPU context is restored; 12: IRQ level ends, task-level execution resumes; zigzag line represents an uninteresting fragment of code execution.

- *Specific*: the class comprises approaches such as *multi-frame model* supporting various execution times for various instances of a task [33,34], *elastic model* able to temporarily prolong the period of some tasks [8,35] and *adaptive model* able to update parameters of a task based on its execution history
- *Imprecise*: it includes approaches usable in applications able to tolerate errors [24,29], e.g., video processing; typically, they temporarily decrease their demand by switching to an approximate (CPU saving) mode of computation.
- *Probabilistic*: it encompasses a multitude of approaches such as [6,7,10,11,20,26,31,32,39], applicable to systems where at least one parameter of a task can be expressed as a random variable; e.g., the approaches build on the *probabilistic worst-case execution time* (pWCET, pC) and the maximum acceptable probability of missing a deadline (the so-called *probabilistic deadline*, *pD*); for a comprehensive overview, please consult, e.g., [19].

Some UPPAAL-based solutions exist as well, e.g. RTLib [37], a model-based framework for scheduling analysis [18] and a multi-core resource model for hierarchical scheduling systems [6]. But each of them models a different subset of reality (e.g. mixed-criticality systems, resource sharing/utilization, hierarchical scheduling in a multicore environment) and concentrates on specific sets of problems and analysis. Alternatively to UPPAAL, one can model and analyze various phenomena using the CHEDDAR tool [38]. Sadly, we must conclude this part by stating that solutions we know abstract from modeling uncertainty sources; instead, they model uncertainty effects as usual, e.g., using the (*min_period*, *max_period*) to parameterize the period of a task. In our approach, we would like to take a step forward by showing that it makes sense to model uncertainty sources in the SAN context.

From the SAN point of view, there are various *schedulability tests* for various situations [9,14,21]; many of them are simple as they are based on computing

parameters such as the (static, overall) CPU utilization factor ($U = \sum_{\tau_i \in \Gamma} u_i$). Such tests, however, usually represents a *sufficient* or *necessary* condition for checking the schedulability, but not both: if a sufficient condition, e.g., Liu and Layland's [30], holds then Γ is schedulable; if a necessary condition, e.g., $U > \#$ of available CPUs, does not hold then Γ is not schedulable. Some tests, however, such as *response time analysis* (RTA) based tests, represent both conditions [9,14]; after such a test finishes, the schedulability of Γ can be checked by testing $R_i \leq D_i$ for each $\tau_i \in \Gamma$. To get more realistic results, the test can be extended [13] – by components such as *blocking time* (B_i), *release time jitter* (J_i), *context-switch time* (C_{sw}), *period of a scheduler tick* (T_{tick}), *time to handle a scheduler tick* (C_{tick}) or *time to make a task ready* (C_{queue}) – to produce more realistic, but pessimistic, results. If uncertainty (e.g., clock drift or IRQs) is considered, the tests become complicated or do not exist for the given conditions.

2.4 Instruments of Our Research

Our research tries to overcome the disadvantages of the approaches summarized in Sect. 2.3 by covering uncertainty aspects such as the drift of a digital clock, non-periodicity of events, behavior of hardware and change of its parameters in time, an OS and its scheduler, task design patterns. Below, let us present key means we used with regard to our research; for more info, consult [2,3,5,12].

Statistical Model Checking. Various means can be used to model the behavior ($\mathcal{M}$) of a RTS, formalize its expected properties (φ) and then check whether such properties can imply ($\models$) from such behavior, that is, whether $\mathcal{M} \models \varphi$. We use a *model checker* (MC) to produce such a decision: if $\mathcal{M} \models \varphi$ holds, the checking finishes; otherwise, the checker can produce a *counter-example* (CE), i.e., a sequence of timed actions leading to the violation of a property – see Fig. 5. As a "classical" MC technique [3,12] is prone to the state space explosion and its decision capability is limited and binary (i.e., either $\mathcal{M} \models \varphi$ or $\mathcal{M} \not\models \varphi$), one may use a technique such as *statistical model checking* (SMC) to overcome disadvantages of MC [1]. SMC typically involves parameters such as N (amount of simulation runs), ε (error of the *confidence interval* (CI)), α (risk of making an error, i.e., the true probability being outside the CI). Simply put, SMC conducts N simulations on a stochastic model and processes them statistically to infer, with a predefined ε, whether they provide statistical evidence for the satisfaction of a property with *confidence level* $1 - \alpha$. Apparently, SMC can produce more than the "satisfied/violated" answer about a property [17]; e.g., it can produce *probability density distribution* used to express how the probability of satisfying a property evolves in time. SMC easily scale [25] and has already been applied to solve various problems [28].

Stochastic Timed Automata. A particular behavior, described by $\mathcal{M}$ can be seen as a (potentially infinite) *path* through a digraph $G_{\mathcal{M}}$ that corresponds to $\mathcal{M}$. Our paper builds on the so-called *timed automata* ($\mathcal{TA}$) formalism [2] that

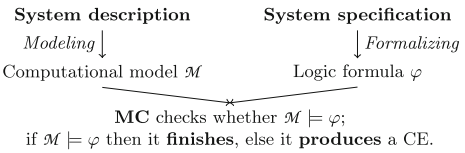


Fig. 5. Classical ("binary") model checking.

extends non-timed formalisms by temporal aspects. The "classical" definition of $\mathcal{TA}$ can be further extended (e.g., into STA [22]) by further concepts such as variables, communication channels, stochastic decisions, ability to reflect costs/rewards [4,17] etc. our approach may profit from. One of the biggest advantages of the $\mathcal{TA}$ formalism is that it allows one to model uncertainty, e.g., using random variables following a distribution of probability. Also, one can use a certainly known (deterministic, fixed) delay, unknown (non-deterministic) delay or to combine the approaches. Note that we prefer to create a usable model based on $\mathcal{TA}$ rather than exhausting all the means that $\mathcal{TA}$ offers, so some means (such as non-deterministic delays) remain unused in our approach at this time.

Specification of Properties. For reasoning purposes, the computational model of a system must be accompanied with the specification of properties to be reasoned about (see Fig. 5). While the *"classical" logic* [15] helps one to check whether a formula φ (ψ) is true or false for all/some realization(s) of logic and evaluation(s) of logic variables, the so-called *temporal logic*, a special kind of the *modal logic* [3,12], distinguishes between various *modes of truth* with respect to the given context, e.g., time or uncertainty. Properties to be checked include general (reachability, safety, liveness), modal (possibility, invariance, potentiality, eventuality), SMC (probability estimation/comparison, hypothesis testing).

3 Our Approach

This section presents our approach to the SAN problem by means of STA and shows the principle/advantage of solving such a problem using SMC. The practical applicability of our approach was tested using the publicly available Uppaal SMC toolset [17], which allows anyone to try whether our approach fits one's needs, to adapt our approach to such a need, etc.

3.1 Uncertainty Modeling

Due to a cyber-physical nature of many RT systems, SAN must ideally deal with all components/types of uncertainty and *classes of randomness*, e.g., *apparent* (may imply from the post-reset flow of instructions), *chaotic* (may relate to the communication/synchronization in an event-driven distributed system), *inherent* (implies from the occurrence of a fault due to aging of material or harsh

environment). As the problem of modeling an RTS w.r.t. uncertainty is complex in general, it tends to result in realistic but non-trivial models. The good news is that, using STA, such a model can be intuitively decomposed into a set of smaller ones (e.g., those from Fig. 6), each representing a particular fragment of reality; models can be independent or interact to correctly express reality.

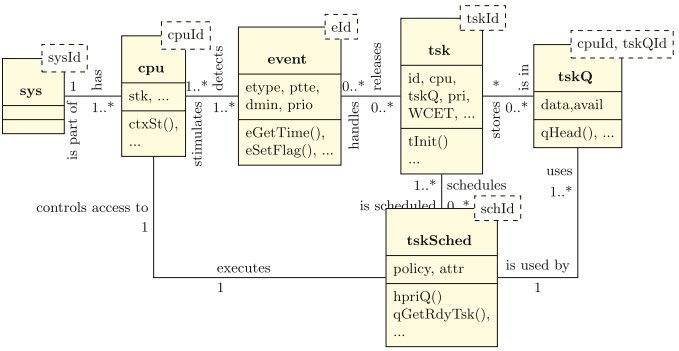


Fig. 6. Simplified UML diagram of key components our approach builds on.

We outline key aspects of our approach in the following text. In our models, common locations/states are colored ●, initial and waiting/stopped ones are colored green ● and orange ●/ yellow ●, resp.; labels are colored as usual in UPPAAL, i.e., the label of a location: magenta-red, invariant: magenta, action: blue, guard: green, channel synchronization: cyan, select construct: olive.

First, our model involves a part that allows one to characterize *events* expected to occur during the simulation time. For that purpose, we are able to characterize non-/determinism or stochasticity of occurrence times to assign them to events then (Listing 1.1). Implicitly, we use events to trigger IRQs.

Listing 1.1. Characterization of probability distributions and events

```
const int PFIX=0, PUNI=1, PEXP=2,PNOR=3,PWEI=4,PPOI=5,PBET=6,PGAM=7,PTRI=8,PARC=9;
typedef int[0, MAX_PDIST-1] t_pdist; // # of supported probability distributions
typedef int[0, MAX_PATTR-1] t_pattr; // # of attributes of probability distributions
// list of supported probability distributions (their attributes are specified tvia pattr[])
const t_pdist pdist[t_pdist] = {  PNOR, PUNI, PNOR, PEXP, PFIX, ... };
// attributes of probability distributions from pdist[]
const double pattr[t_pdist][t_pattr] = {
//  scale, atr0, atr1, atrt2, ...
    {1.0, 3000.0, 50.0, 0.0}, {1.0, 5000.0, 0.0, 0.0}, ... };
// structure used do specify an event
typedef struct {
   t_event etype;  // event type: 0 (periodic), 1 (sporadic), 2 (aperiodic)
   int ptte;       // time-to-event: -1 (unused), 0+ (given by pdist/pattr)
   int dmin;       // min. delay: >0 for sp., period for per.; ignored for aper.
   int prio;       // event priority (0 = HPRI, ...)
} t_sEvent;
t_sEvent edef[t_nedef] =  {{0, 2, 1, 0}, ... }; // definition of supported events
t_nedef egid[t_negen] = { 0, 1, 2, 3 }; // event generator is defined by an index to edef
```

For each event, we use an STA to introduce the event (by setting its binary flag) into the simulation model at the appropriate time. Figure 7a shows an illustrative model of a generator able to introduce events after a predefined

deterministic (fixed) or stochastic delay (*dly*). For an event e, *dly* is produced by *eGetTime*(e). Based on Listing 1.1, the function produces a pseudorandom number that follows the probability distribution given for e. To generate a fixed *dly* for e, we can, for example, add *PFIX* to *pdist*[] and put *dly* in the *atr0* part of *pattr*[][]. To define a nondeterministic (aperiodic) *dly* for e, we must update the STA from Fig. 7a to be able to execute either the deterministic/stochastic or non-deterministic part of the STA. The decision about which part will be executed can be made based on *ptte* (≥ 0: the former part, -1: the latter part) stored in *edef*[] for e.

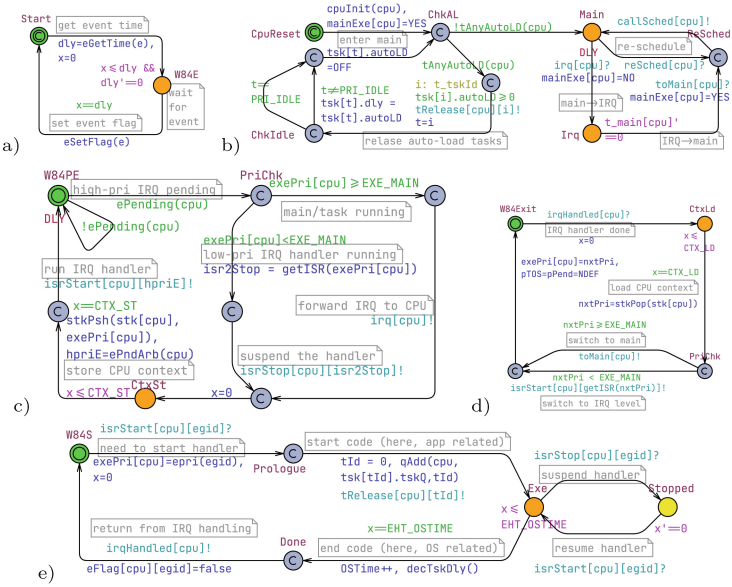


Fig. 7. Low-level components of our model: a) deterministic/stochastic event generator, b) CPU, c) IRQ level entry, d) IRQ level exit, e) IRQ handler (an example for SYSTICK; among the others, it increments *OSTime* and decrements positive values of tasks' *dly*; low priority of SYSTICK can become a source of uncertainty).

Next, our approach involves a model of a CPU-based *hardware* (Fig. 7b–e) capable of detecting an event by polling and/or interrupts. If a CPU (see Fig. 7b) is in *Main* and receives such an IRQ, it enters *Irq* (to handle the corresponding IRQ, for example, the one from Fig. 7e) and waits there while an IRQ is pending. Figure 7 c illustrates an STA that we use to produce an IRQ based on an event, if applicable. If no IRQ is pending at the time of exiting the actual IRQ handler (see Fig. 7d), the CPU enters *ReSched* to decide about the next execution. Note that a task can be released (i.e., added to the corresponding queue of ready tasks) from an IRQ handler (Fig. 7e).

In addition, our approach involves a model of (RT) *tasks* (see Fig. 8). A one-shot task consists of an acyclic sequence that typically ends by removing the task

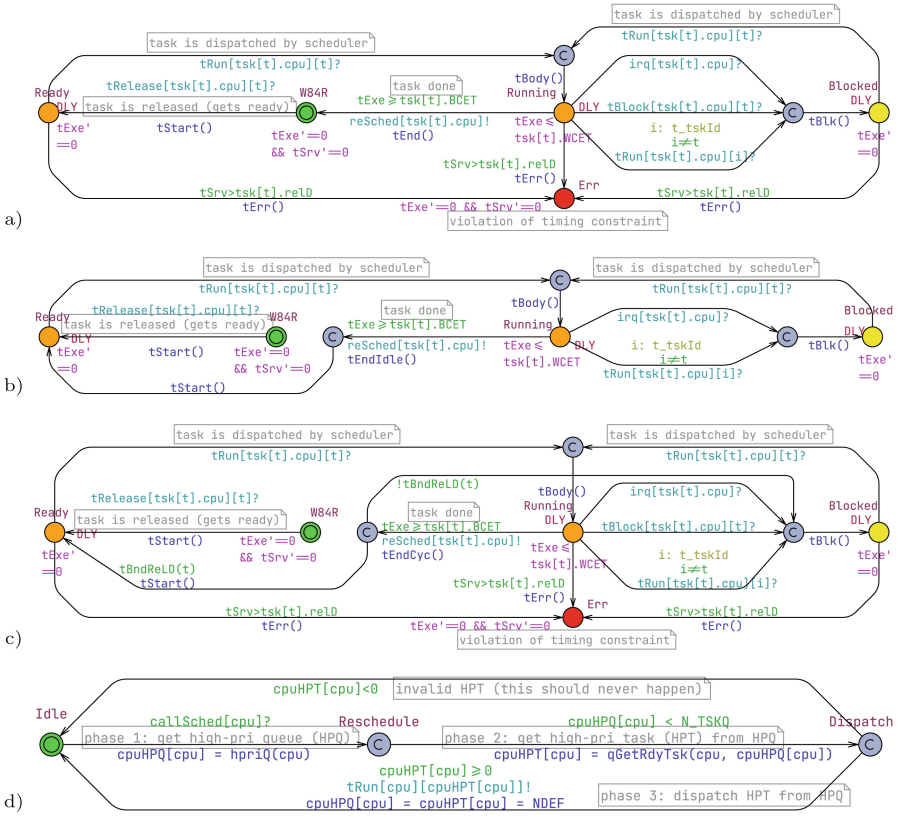


Fig. 8. High-level components of our model: a) one-shot task, b) Idle task, c) cyclic task, d) task scheduler.

from OS; it is suitable to service rare events. The idle task typically runs at the lowest priority level. and, as it can never be blocked, it facilitates the construction of an OS. A cyclic task involves a loop an iteration of which typically starts/ends by calling a blocking function. Like an event, a task (recall Fig. 1, Fig. 6) can be characterized by a structure composed of the following integers; see the list. 1.2 too: *id* (unique ID), *ena* (task enable flag), *pri* (priority), *BCET*, *WCET*, *relD* (D), *st* (state: wait for release, ready etc.), *dly* (# OSTime ticks to unblock), *cpu*, *tskQ* (queue ID), *autoLD* (auto-loading at the OS startup; −1: off, 0: at startup, > 0: delayed by # OSTime ticks), *reLD* (reloading – a delay before starting the next iteration of a cyclic task; −1: off/unbounded, 0: no delay (imm.), > 0: delayed by # OSTime ticks).

Inspired by practice, we distinguish the following kinds of tasks (recall Fig. 1 and see Fig. 8a–c): *one-shot*, *idle* and *cyclic*. Let us note that while an idle task uses immediate auto-loading and re-loading inherently, other tasks can be configured in this sense; however, reloading makes no sense for one-shot tasks.

In this paper, our approach deals with the auto-loading before the CPU enters *Main* (see the *ChkAL* cycle in Fig. 7b) and with re-loading after the iteration of a cyclic task completes (see Fig. 8c) – one can change this if needed.

Listing 1.2. Task structure and a task set definition

```
typedef struct {
    int id;     // id
    int ena;    // task enabled flag (0: task involved in simulations, 1: task excluded from simulations)
    int pri;    // priority
    int BCET;   // best-case exe. time
    int WCET;   // worst-case exe. time
    int relD;   // relative deadline
    int st;     // 0-ready, 1-running, 2-blocked, 3-irq, ...
    int dly;    // delay counter
    int cpu;    // id of a CPU used to run the task
    int tskQ;   // id of a task-queue used to store the task
    int autoLD;// auto-loading (-1: off, 0: immediate/at-startup, >0: delayed by OSTime ticks)
    int reLD;   // reload, i.e., delay before starting the next iteration of its body (makes sense
                //for cyclic, i.e., non-1shot, tasks); (-1: disabled = unbounded delay), 0: immediate,
                // >0: delayed by OSTime ticks)
} t_tsk;
//
t_tsk tsk[N_TASKS] = {
//  id, ena, pri, BCET, WCET, relD, st, dly, cpu, tskQ, autoLD, reLD
    {0, NO, 0, 50, 100, 25000, TW4R, 0, 0, 0, OFF, OFF},              // 1shot, IRQ0
    {1, NO, 1, 50, 100, 25000, TW4R, 0, 0, 0, OFF, OFF},              // 1shot, IRQ1
    {2, YES, 2, 50, 100, 25000, TW4R, 0, 0, 0, 2, 5},                 // cyclic bndDly
    {3, YES, 3, 1000, 1000, 25000, TW4R, 0,  0, 1, 0, 10},            // cyclic, RR policy, queue=1
    {4, YES, 3, 1000, 1000, 25000, TW4R, 0,  0, 1, 0, 10},            // cyclic, RR policy, queue=1
    {5, NO, 5, 0, 0, 0, TW4R, 0, 0, 0, OFF, OFF},
    {6, NO, 6, 0, 0, 0, TW4R, 0, 0, 0, OFF, OFF},
    {ID_IDLE, YES, PRI_IDLE, 5, 20, 25000, TW4R, 0, 0, Q_IDLE, 0, 0} // LPRI reserved for the IDLE task
};
```

Last, but not least, our approach involves a pool of *task queues* (*tskQ*[][]) and a *scheduler* per a queue (*tskQSch*[]) – consult Fig. 6, Listing 1.3, Fig. 8d, too. After calling a scheduler, it moves from *Idle* to *Reschedule*. During the transition, the scheduler identifies the highest-priority queue (HPQ) with a ready task inside. In the second phase, it identifies the highest-priority task (HPT) in the HPQ. Finally, it dispatches the HPT from HPQ and returns to *Idle*.

Listing 1.3. Task queue and scheduler structures and definition of schedulers

```
typedef struct {        // structure used to specify a task queue
    t_tskQData data[SIZE_TSKQ]; // data stored in the queue
    int[0, SIZE_TSKQ] avail;    // # of available elements in Q
    bool prioQ;                 // used as a priority queue: false (0) / true (1)
} t_tskQ;
//
typedef struct {        // structure used to specify a task scheduler
    t_tskSPol policy;   // scheduling policy
    int attr;           // e.g., timeslice/quantum for RR (in OSTime ticks); valid when >= 1
} t_tskSched;
//
t_tskQ tskQ[N_CPU][N_TSKQ];     // pool of task queues
//
t_tskSched tskQSch[N_TSKQ] = {  // task queue -> scheduler mapping
/* queue 0 */    { SCH_PRE|SCH_FIXP, NDEF },    // preemptive, fixed-priority, quantum-free scheduler
/* queue 1 */    { SCH_PRE|SCH_RR, 4 }          // preemptive, round-robin, quantum 4 OSTime ticks long
};
```

Finally, Table 1 illustrates how we manage *priorities* in our approach. In the table, priorities are situated in the three regions: green (reserved for low priorities, i.e., software/tasks), yellow (reserved for the MAIN priorities, i.e., bare-metal or OS kernel/scheduler), and red (reserved for high priorities, i.e., hardware, IRQs, etc.). The importance of priority increases with its decreasing numerical value, that is, the highest priority (*HPRI*) has the value 0 and the lowest priority (*LPRI*) has the maximum available value ($N\_PRI - 1$).

Table 1. An illustration to priority levels in our approach

Execution priority			Example
Level	**Name**	**Value**	(for N_PRI=16, N_IRQ_PRI=4, N_TASKS=8)
lowest (EXE_LPRI)	*TSK_LPRI*	*N_PRI*–1	Idle task, pri = 12, auto-load = ON, reload = OFF
			...
	...		Cyclic task, pri = 7, auto-load = ON(2), reload = ON(5)
			One-shot (IRQA) task, pri = 6, auto-load = OFF, reload = OFF
	TSK_HPRI	⋮	SysTick task, pri = 5, auto-load = OFF, reload = OFF
	MAIN_PRI		main (post-reset execution), pri = 4
	IRQ_LPRI		...
	...	1	IRQ_A, maskable, rand. (uni. dist. in 2500)
highest (EXE_HPRI)	*IRQ_HPRI*	0	IRQ_SysTick, maskable, periodic (T=200)

3.2 Schedulability Analysis

In our approach, the SAN problem maps to the problem of checking the possibility of entering *Err* in a (RT) task model. In Fig. 8, the checking makes sense for (an instance of) one-shot (Fig. 8a) and cyclic (Fig. 8c) tasks being limited by timing constraints, but does not make sense for an idle task (Fig. 8b) that uses no timing constraints. The ability of a task τ_i to meet – under a given uncertainty model and other conditions – its timing constraints (such as its deadline) can be checked by symbolic queries such as

- $\exists\Diamond\tau_i.Err$ ... we used the query type to check whether there exists ($\exists$) a path (in the state space of a model) such that τ_i may eventually ($\Diamond$) enter *Err*; the satisfaction of this query means that, under examined conditions, τ_i cannot meet its timing constraints.

or SMC queries such as

- $Pr[<= time\_bound](\Diamond\tau_i.Err)$ ... we used the query type to check the probability that τ_i may eventually enter *Err* within *time_bound* units of time; after checking, we can get, e.g., the probability density distribution of satisfying the property within *time_bound*. This kind of result may be more valuable than the symbolic one as it gives us information about when and how likely the timing constraints are violated.

In addition to the query types mentioned above, we also used the following.

- $E[<= time\_bound; N](max : Expr)$ ... we used the query type to estimate the maximum value of a given expression (*Expr*) based on N simulation runs (experiments) within *time_bound*; particularly, for example, we may be interested what is the maximum time CPU_i spends by executing the main program within *time_bound* (here, $Expr = t\_main[i]$),
- $simulate[<= time\_bound; N]\{Expr\}$ ... we used the query type to start N simulation runs and for each of the runs observe how the value of a given expression (*Expr*) evolves within *time_bound*; particularly, for example, we may be interested how the value of the execution priority (*exePri*) or OS time (*OSTime*) evolves ($Expr = exePri$ or $Expr = OSTime$, resp.).

4 Evaluation

We have evaluated our approach in different scenarios, the representative results of which are summarized below.

Clock Drift/OSTime Jitter
First of all, let us show the ability of our approach to model one of the most important sources of uncertainty in digital systems, i.e., jitter/drift of a clock. Figure 9a shows the result of one simulation run (produced by query *simulate* $[2500; 1]\{OSTime, 10 + exePri[0], 15 + ePending(0), 20 + eFlag[0][0], 25 + eFlag[0][1], 5 + irqExe(0)\}$) where

- $OSTime$ is the (logical) time an OS uses to realize the temporal control,
- $10 + exePri[0]$ is the execution priority (see Table 1) at the CPU #0, with its value vertically shifted by +10 (to minimize overlaps in the figure),
- $15 + ePending(0)$] is the boolean (0/1) value used to signalize that an IRQ is pending on the CPU #0, with the +15 vertical shift,
- $20 + eFlag[0][0]$ is the boolean flag used to signalize that on the CPU #0, the highest-priority IRQ#0 (from SysTick) is pending, with the +20 vertical shift; alike, we observe the flag for a lower-priority IRQ#1, vertically shifted by +25,
- $5 + irqExe(0)$ is the boolean flag used to signalize that the CPU #0 spends its time at the IRQ level, with the +5 vertical shift.

Figure 9a shows that IRQ#0 from SysTick occurs regularly every 200 units of time. As there is no higher priority IRQ, its handler starts with this period, never interrupted, to (perfectly periodically) increment the value of $OSTime$. However, IRQ#1 handling can be interrupted each time IRQ#0 occurs, so IRQ handlers may nest, prolonging the response time at lower priority levels (see what happens in time between 700 and 1300, or after 2100).

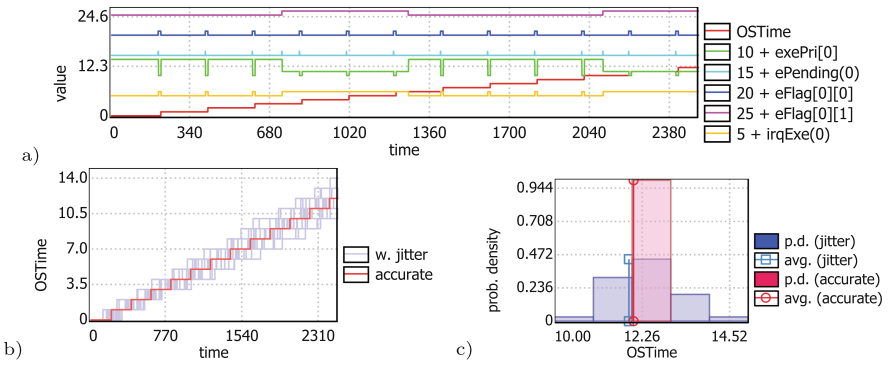


Fig. 9. An illustration to the evolution a) of jitter/drift-free increase of OSTime and nesting of IRQ handlers, and b), c) clock jitter/drift enabled.

Figure 9b, the result of $simulate[<= 2500; 1]\{OSTime\}$, shows the effect of switching the IRQ#0 occurrence time from periodic (period 200, the red line) to the value that follows the Normal distribution with mean = 200 and std. deviation = 50 (see the blue line). Apparently, at 2500 the OSTime (imperfect/blue) value differs by about ± 2 from its (perfect/red) value, which is 12. Any nonzero difference negatively affects the timing at the OS level. As this may happen in practice, we can use a model checker to analyze the consequences. Figure 9c shows the same result, but as a result of $E[<= 2500; 10]\{max : OSTime\}$, with $\alpha = \varepsilon = 0.05$, 95% CI.

In addition to clock jitter/drift (recall Fig. 9), OSTime may be affected by another source of uncertainty, i.e., by the occurrence of an IRQ having its priority above SysTick. Figure 10 shows the complexity of the latter situation using a simulation result (of the query $simulate[<= 5000; 1]\{exePri[0], \ldots, 50 + OSTime\}$) for the IRQ priority of SysTick set to the second highest (IRQ#3), multiple sources of uncertainty, masking IRQs at runtime, *late arrival* (LA) and *tail chaining* (TC) of IRQs etc.

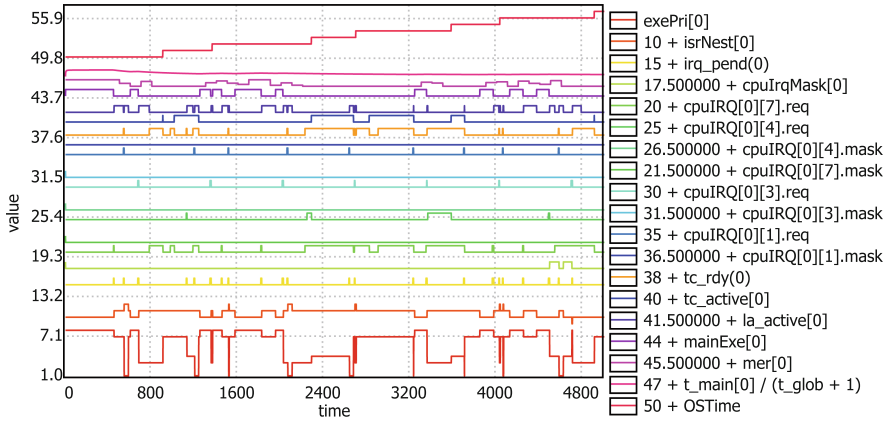


Fig. 10. Detailed simulation results. The variables being observed are (all at CPU#0): the priority of execution ($exePri[0]$), level of nesting IRQ handlers ($isrNest[0]$), state of the CPU-level IRQ mask ($cpuIrqMask[0]$), state of the IRQ mask and flag for IRQ#i ($cpuIrq[0][i].mask$ and $cpuIrq[0][i].req$, resp.), where $i \in \{1, 3, 4, 7\}$, TC readiness and activity flags ($tc\_rdy(0)$ and $tc\_active[0]$, resp.), LA activity flag ($la\_active[0]$), Main execution flag ($mainExe[0]$), fraction of time CPU spends by executing Main ($mer[0]$), ratio of time CPU spend in Main to total time ($t\_main[0]/(t\_glob + 1)$), logical time of an OS ($OSTime$).

Among others, Fig. 10 shows uncertainty effects such as irregularity in the increase of OSTime – apparently, OSTime gets very unprecise despite requests at the level of IRQ#3 occur regularly, that is, without jitter/drift of the SysTick clock. Further effects of uncertainty can be seen in Fig. 10, e.g. execution priority ($exePri[0]$) and time spent by execution of Main ($mainExe[0]$).

Effect of Uncertainty on Execution of Main

Figure 11 shows the effects of uncertainty on the time spent by executing Main within 5000 units of time. While Fig. 11a shows the result of the query $simulate[<= 5000; 1]\{MER(lLoad), \ldots\}$, Fig. 11b shows the result of the estimation query $E[<= 5000; 10](max : t\_main[0])$, with $\alpha = \varepsilon = 0.05$, 95% CI. Figure 11a shows that in the high-load situation (red lines), the CPU spends less time by executing Main than in the low-load situations (blue lines). This kind of analysis helps one to identify, e.g., the smallest (worst-case) amount of CPU time that is available to execute OS kernel, tasks, etc. If time drops below a certain level, some tasks could become unschedulable. Figure 11b shows the probability distribution of the time the CPU spends in Main during 5000 units of time; on average, the CPU spends about 3500 units of time in Main, where it is highly probable that the CPU spends in Main about 3200 or 3800 units of time.

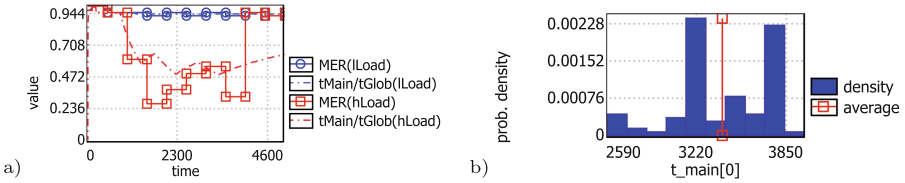


Fig. 11. Uncertainty effect on the execution of Main for low/high (lLoad/hLoad) IRQ activity. Note 1: MER (Main Execution Ratio) represents the fraction of time the CPU has spent by executing Main during last 100 units of time. Note 2: tMain/tGlob represents the ratio of time CPU spent by executing Main to the total time passed.

Behavior at the Task Level

This section presents representative results of our approach to show its readiness for various situations at the task level. For example, Fig. 12 shows the result of the query $simulate[<= 2500; 1]\{OSTime, \ldots\}$, i.e., the ability of our model to express the behavior of a task that is automatically loaded two OSTime cycles after OS starts (recall Fig. 1 and line 22 of the Listing 1.2 and no higher priority task is running. In Fig. 12, you can see that $tsk[2].dly$ is loaded with two OSTime units at t=0 to decrement then with each OSTime update until it reaches zero; if its priority (#7, consult the Table 1) is sufficiently high, the task is run (in the interval when $tsk[2].st = 2$, after excluding offset 40). After the task finishes, $tsk[2].dly$ is loaded with five OSTime units to decrement then with each OSTime update until it reaches zero at about t = 1300. So, the next instance of the task is reloaded again, five OSTime cycles after its previous instance completes, to run from about t = 1450 to about t = 1600 etc.

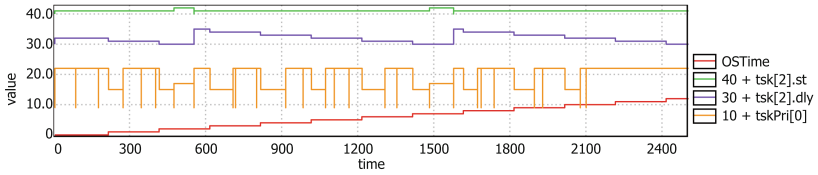


Fig. 12. An illustration to a task (id = 2, pri = 7) with the auto-load and reload capability (autoLD = 2, reLD = 5) – see Table 1; OSTime increments each 200 units of time.

Figure 13 shows that the average waiting time for a low-priority task (id = 4 in Listing 1.2) is, intuitively and as an effect of using priorities, higher than the average waiting time for a high-priority task (id = 2 in Listing 1.2) – the figure was produced based on the query $E[<= 250; 10](max : avg\_wt[task\_pri])$. Generally, such a behavior of tasks decreases the response time of the highest-priority task and, consequently, prolongs the response time of tasks below the highest priority.

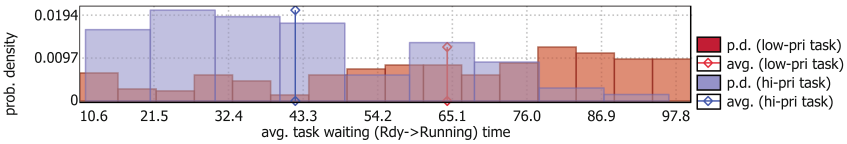


Fig. 13. An illustration to avg. waiting time of a low-pri and high-pri tasks, resp.

For the next part, we prepared [42] representative sets of tasks to be scheduled using various policies (FPS, RM, DM, EDF, RR, etc.) with the goal of checking the schedulability of the sets under different conditions. A set involves a mixture of tasks from various classes (recall Fig. 1), each defined by the list of carefully selected parameters such as r, C etc. (see the bottom of p. 3):

Set 1: $\tau_1(r = 0, C = 2, T = D = 5)$, $\tau_2(r = 0, C = 4, T = D = 7)$,
Set 2: $\tau_1(r = 0, C = 2, T = D = 5)$, $\tau_2(r = 1, C = 1, T = D = 4)$, $\tau_3(r = 2, C = 2, T = D = 20)$,
Set 3: $\tau_1(r = 0, C = 4, D = 15)$, $\tau_2(r = 0, C = 3, D = 12)$, $\tau_3(r = 2, C = 5, D = 7)$, $\tau_4(r = 5, C = 2, D = 3)$,
Set 4: $\tau_1(r = 0, C = 3, T = 20, D = 5)$, $\tau_2(r = 0, C = 3, T = 12, D = 7)$, $\tau_3(r = 0, C = 4, T = D = 10)$, $\tau_4(r = 0, C = 3, T = D = 20)$,
Set 5: $\tau_1(r = 1, BCET = 2, WCET = 3, T = 50, D = 10)$, $\tau_2(r = 4, BCET = 1, WCET = 3, T = 60, D = 7)$, $\tau_3(r = 1, BCET = 1, WCET = 2, T = 45, D = 6)$,
Set 6: $\tau_1(r = 2, , BCET = 1, WCET = 3, D = 8)$, $\tau_2(r = 4, C = 1, T_{MIN} = 17, T_{MAX} = 19, D = 2)$, $\tau_3(r = 0, BCET = 2, WCET = 3, D = 7)$, $\tau_4(r = 10, C = 4, D = 8)$.

We used the sets to prepare a set of SAN experiments, each based on checking the queries $\exists\Diamond\tau_i.Err$ and $Pr[<= 2500](\Diamond\tau_i.Err)$ under $\alpha = \varepsilon = 0.01$, 95% CI. First, we checked the former query to get a qualitative result; then, we checked the latter query to get quantitative results. If not explicitly said, interrupts were

disabled to allow us to validate our results with the help of the CHEDDAR tool, if possible [38]; e.g. CHEDDAR cannot model interrupts.

First, we used Set 1 (two periodic tasks released at t = 0) for initial validation. The results of $\exists\Diamond\tau_i.Err$ have shown that the set cannot be schedulable by RM and DM. As checking for EDF has not ended for an excessively long time (couple of hours), we used $Pr[<= 2500](\Diamond\tau_i.Err)$ to obtain a faster result (Table 2); apparently, the probability that a task enters *Err* is close to zero, so we may conclude that it is highly probable that the set is schedulable by EDF.

Table 2. Results of $Pr[<= 2500](\Diamond\tau_i.Err)$ for Set 1 under $\alpha = \varepsilon = 0.01$, 95% CI

Scheduling policy	RM	DM	EDF
Probability of timing error	[0.990015, 1]		[0, 0.00998451]

Set 2 consists of three periodic tasks, each released at a different time. The set schedulable by each of the policies (Table 3).

Table 3. Results of $Pr[<= 2500](\Diamond\tau_i.Err)$ for Set 2 under $\alpha = \varepsilon = 0.01$, 95% CI; the quantum ($q$) of RR is given in the form RR(q)

Scheduling policy	RM	DM	EDF	FIFO	RR(2)	RR(1)
Probability of timing error	[0, 0.00998451]					

Set 3 consists of three aperiodic tasks released at different times. We identified (Table 4) that the set is not schedulable by the FIFO policy but is schedulable by DM, EDF. As tasks are not periodic, it makes no sense to use RM.

Table 4. Results of $Pr[<= 2500](\Diamond\tau_i.Err)$ for Set 3 under $\alpha = \varepsilon = 0.01$, 95% CI

Scheduling policy	DM	EDF	FIFO
Probability of timing error	[0.00998451, 1]		[0.990015, 1]

Set 4 consists of four periodic tasks released at t = 0. As RM is optimal for $D = T$ (for example, DM is optimal for $D \leq T$), we can expect that RM could be unable (but DM capable) to schedule the set. Our results show (Table 5) that the set is schedulable by DM, EDF, but is not schedulable by the remaining policies.

Table 5. Results of $Pr[<= 2500](\Diamond \tau_i.Err)$ for Set 4 under $\alpha = \varepsilon = 0.01$, 95% CI; the quantum ($q$) of RR is given in the form RR(q)

Scheduling policy	DM	EDF	FIFO	RR(1)	RR(3)	RM
Probability of timing error	[0, 0.00998451]		[0.990015, 1]			

Set 5 consists of three periodic tasks with various release times. In the set, we use tasks with variable execution time (e.g., BCET differs from BCET). Our results show (Table 6) that the set is schedulable by RM, DM, EDF, but it is not schedulable by the remaining policies (for FIFO, RR(2), FPS there is about 50% chance for schedulability of the set, which is not guarantee of schedulability).

Table 6. Results of $Pr[<= 2500](\Diamond \tau_i.Err)$ for Set 5 under $\alpha = \varepsilon = 0.01$, 95% CI; the quantum ($q$) of RR is given in the form RR(q)

Sched. policy	DM	EDF	FIFO	RR(2)	FPS	RM
Prob. of timing err.	[0, 0.00998451]		[0.489192, 0.509191]		[0.495663, 0.51566]	[0, 00998451]

In addition, we prepared a set of five experiments for Set 5, each of which was carried out under different interrupt conditions (see Table 7): in Experiments 1–3, we increased the IRQ arrival rate during the constant IRQ handling time; in Experiments 4 and 5, we used the same (slowest) IRQ arrival rate, but increased the IRQ handling time in Experiment 5. Apparently, the increase of IRQ arrival rate has the most negative impact on schedulability of the set.

Table 7. Results of $Pr[<= 2500](\Diamond \tau_i.Err)$ for Set 5 w. IRQs, $\alpha = \varepsilon = 0.01$, 95% CI

Scheduling policy		EDF	RM	DM
Probability of timing error	**Exp. 1**	[0.000788202, 0.0146731]	[0.000788202, 0.0146731]	[0.000788202, 0.0146731]
	Exp. 2	[0.0644374, 0.0836153]	[0.0572148, 0.0763039]	[0.0640095, 0.0831845]
	Exp. 3	[0.23909, 0.258901]	[0.232959, 0.252762]	[0.238421, 0.25823]
	Exp. 4	[0.0148453, 0.0324876]	[0.0205155, 0.0385701]	[0.0120223, 0.0293759]
	Exp. 5	[0.0579237, 0.0770239]	[0.0475622, 0.0664997]	[0.0518603, 0.0708735]

Set 6 consists of three periodic tasks and one aperiodic task, where τ_1 and τ_3 have variable execution time and τ_2 variable release time. Our results show (Table 8) that the set is schedulable just by RM, EDF; FIFO and RR violate (more likely) the deadline of τ_2, FPS violates (less likely) the deadline of τ_3.

Table 8. Results of $Pr[<= 2500](\Diamond \tau_i.Err)$ for Set 6 under $\alpha = \varepsilon = 0.01$, 95% CI; the quantum ($q$) of RR is given in the form RR(q)

Sched. policy	DM	EDF	FIFO	RR(4)	FPS
Prob. of timing err.	[0, 0.00998451]		[0.990015, 1]		[0.737917, 0.757732]

In addition, we prepared a set of five experiments for Set 6, each of which was carried out under different interrupt conditions (see Table 9): in Experiments 1–3, we increased the IRQ arrival rate during the constant IRQ handling time; in Experiments 4 and 5, we used the same (fastest) IRQ arrival rate, but decreased the IRQ handling time in Experiment 5. Apparently, the increase in the arrival rate of IRQ and the reduction in the handling time of IRQ have a negative impact on the schedulability of the set.

Table 9. Results of $Pr[<= 2500](\Diamond \tau_i.Err)$ for Set 6 w. IRQs, $\alpha = \varepsilon = 0.01$, 95% CI

Scheduling policy		EDF	FPS	DM
Probability of timing error	**Exp. 1**	[0, 0.00998451]	[0.735683, 0.7555]	[0, 0.00998451]
	Exp. 2	[0, 0.00998451]	[0.794266, 0.813996]	[0, 0.00998451]
	Exp. 3	[0.0137475, 0.0312851]	[0.949832, 0.968381]	[0.017677, 0.0355469]
	Exp. 4	[0.822438, 0.842107]	[0.988449, 0.999992]	[0.830232, 0.849882]
	Exp. 5	[0.122489, 0.142049]	[0.984495, 0.998811]	[0.118912, 0.138458]

Based on the query $E[<= 2500; 10](max : sch\_ratio)$, we checked the percentage of schedulable tasks in Sets 1 to 6. Figure 14 shows that the ratio of schedulable sets can be about 25% higher if uncertainty sources are modeled by means of STA, replacing common approaches, based on modeling uncertainty effects, mentioned at the beginning of Sect. 2.3. Our results show that the proposed approach is capable of dealing with facts that are often modeled as random, abstracting from their realistic nature. This ability aims to better (less pessimistically, more realistically) evaluate parameters such as CPU utilization factor (e.g., to be used by a schedulability test) or WCRT (e.g., to be used then by an RTA) and consequently make an, originally pessimistic, STA result realistically optimistic.

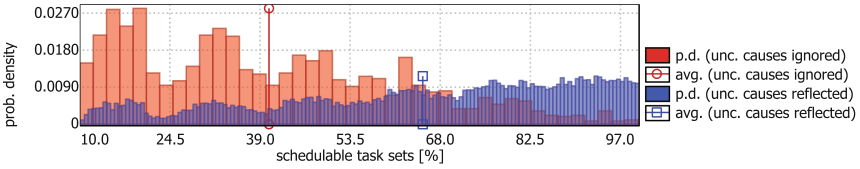


Fig. 14. Comparing ratios of schedulable task sets for uncertainty ignored (avg. ≈ 40%) and reflected (avg. ≈ 65%) in models, resp.

5 Conclusion

This paper presents our approach to the schedulability analysis (SAN) of real-time (RT) systems in realistic situations, burdened by uncertainty. We propose a simulation model that builds on a network of stochastic timed automata (STA), an instrument able to cope with such situations. For analyzing the schedulability, we process our model using the statistical model checking (SMC) technique. However, the usage of such means implies some inherent problems, such as dealing with rare events or getting a counterexample to study causes of violating a property.

The novelty of our approach can be seen in modeling causes of uncertainty (e.g., the drift of a clock, management of CPU context, OSTime and OS structures such as task queue(s), un/masking of IRQs at runtime nesting of IRQ handlers) rather than their effects (e.g., the jitter of release/execution times of tasks, servicing time of IRQ handlers, or prolonging the response time of tasks/handlers by higher-priority executions), as usual in existing approaches.

Our experiments show that an effort to incorporate causes of uncertainty into models can enhance the accuracy of SAN inputs and consequently, can revise SAN decisions about the schedulability of task sets to potentially move such sets from the (originally pessimistically classified) non-schedulable class to the (realistically classified) schedulable class. This is possible as existing approaches are pessimistic in their nature, which may result in a wrong conclusion for given conditions, however.

Actually, our model can be seen as a proof-of-concept that works but is not perfect and needs some enhancement. For example, we need to complete the support for modeling dependable tasks (this is ready, but not finished), power issues (we plan to profit from stochastic hybrid automata there) or migration of tasks in a distributed environment. Also, we plan to extend our model by further sources of uncertainty, such as pipelines and caches as is done, e.g., in [16].

In the near future, we plan to prepare a more sophisticated set of experiments to better highlight the capabilities of our model. We plan to start by preparing experiments that show an impact of the clock drift to behavior of tasks, their schedulability, etc. – to the best of our knowledge, such a relation has been ignored and not studied enough or at all, but may show surprising facts. Next, we would like to use a real platform, running a real OS, as a validation base for our experiments. Last but not least, we would like to deal with the problem of setting parameters of our model to ideally match a particular platform and OS.

References

1. Agha, G., Palmskog, K.: A survey of statistical model checking. ACM Trans. Model. Comput. Simul. (TOMACS) **28**(1), 6:1–6:39 (2018). https://doi.org/10.1145/3158668
2. Alur, R., Dill, D.L.: A theory of timed automata. Theoret. Comput. Sci. **126**(2), 183–235 (1994). https://doi.org/10.1016/0304-3975(94)90010-8

3. Baier, C., Katoen, J.P.: Principles of Model Checking. Representation and Mind, MIT Press, Cambridge (2008). https://mitpress.mit.edu/books/principles-model-checking
4. Behrmann, G., David, A., Larsen, K.G.: A tutorial on Uppaal. In: Bernardo, M., Corradini, F. (eds.) SFM-RT 2004. LNCS, vol. 3185, pp. 200–236. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30080-9_7
5. Bertrand, N., et al.: Stochastic timed automata. Logical Methods Comput. Sci. **10**(4) (2014). https://doi.org/10.2168/LMCS-10(4:6)2014
6. Boudjadar, A., et al.: Statistical and exact schedulability analysis of hierarchical scheduling systems. Sci. Comput. Prog. **127**, 103–130 (2016). https://doi.org/10.1016/j.scico.2016.05.008. special issue of the 11th International Symposium on Formal Aspects of Component Software
7. von der Brüggen, G., Piatkowski, N., Chen, K.H., Chen, J.J., Morik, K.: Efficiently approximating the probability of deadline misses in real-time systems. In: 30th Euromicro Conference on Real-Time Systems (ECRTS 2018). Leibniz International Proceedings in Informatics (LIPIcs), vol. 106, pp. 6:1–6:22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2018). https://doi.org/10.4230/LIPIcs.ECRTS.2018.6
8. Buttazzo, G.C., Lipari, G., Abeni, L.: Elastic task model for adaptive rate control. In: Proceedings 19th IEEE Real-Time Systems Symposium (Cat. No.98CB36279), pp. 286–295 (1998). https://doi.org/10.1109/REAL.1998.739754
9. Buttazzo, G.: Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications - Second Edition, Springer 2005, vol. 24. 3rd edn. (2011). https://doi.org/10.1007/978-1-4614-0676-1
10. Chen, K., Chen, J.: Probabilistic schedulability tests for uniprocessor fixed-priority scheduling under soft errors. In: 2017 12th IEEE International Symposium on Industrial Embedded Systems (SIES), pp. 1–8 (2017). https://doi.org/10.1109/SIES.2017.7993392
11. Chen, K., Von Der Brüggen, G., Chen, J.: Analysis of deadline miss rates for uniprocessor fixed-priority scheduling. In: 2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), pp. 168–178 (2018). https://doi.org/10.1109/RTCSA.2018.00028
12. Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R.: Handbook of Model Checking. Springer International Publishing, Cham, 1st edn. (2018). https://doi.org/10.1007/978-3-319-10575-8
13. Fidge, C.J.: Real-time schedulability tests for preemptive multitasking. Real Time Syst. **14**(1), 61–93 (1998). https://doi.org/10.1023/A:1007993819750
14. Cottet, F., Delacroix, J., Kaiser, C., Mammeri, Z.: Scheduling in Real-Time Systems. Wiley, Hoboken (2002)
15. van Dalen, D.: Logic and Structure. Universitext, Springer, London, 5th edn. (2013). https://doi.org/10.1007/978-1-4471-4558-5
16. Dalsgaard, A.E., Olesen, M.C., Toft, M., Hansen, R.R., Larsen, K.G.: METAMOC: modular execution time analysis using model checking. In: 10th International Workshop on Worst-Case Execution Time Analysis (WCET 2010). OASIcs, vol. 15, pp. 113–123. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2010). https://doi.org/10.4230/OASIcs.WCET.2010.113
17. David, A., Larsen, K.G., Legay, A., Mikučionis, M., Poulsen, D.B.: Uppaal SMC tutorial. Int. J. Softw. Tools Technol. Transfer **17**(4), 397–415 (2015). https://doi.org/10.1007/s10009-014-0361-y

18. David, A., Rasmussen, J., Larsen, K., Skou, A.: Model-based framework for schedulability analysis using uppaal 4.1 (2009). https://doi.org/10.1201/9781420067859-c4
19. Davis, R., Cucu-Grosjean, L.: A survey of probabilistic schedulability analysis techniques for real-time systems. Leibniz Trans. Embed. Syst. **6**(1), 04–1–04:53 (2019). https://doi.org/10.4230/LITES-v006-i001-a004
20. Diaz, J.L., Lopez, J.M., Garcia, M., Campos, A.M., Kanghee Kim, Bello, L.L.: Pessimism in the stochastic analysis of real-time systems: concept and applications. In: 25th IEEE Real-Time Systems Symposium, pp. 197–207 (2004). https://doi.org/10.1109/REAL.2004.41
21. Feld, T., Biondi, A., Davis, R.I., Buttazzo, G., Slomka, F.: A survey of schedulability analysis techniques for rate-dependent tasks. J. Syst. Softw. **138**, 100–107 (2018). https://doi.org/10.1016/j.jss.2017.12.033
22. Hartmanns, A., Hermanns, H.: In the Quantitative automata ZOO. Sci. Comput. Program. **112**, 3–23 (2015). https://doi.org/10.1016/j.scico.2015.08.009, fundamentals of Software Engineering (selected papers of FSEN 2013)
23. Helton, J., Johnson, J., Oberkampf, W.: An exploration of alternative approaches to the representation of uncertainty in model predictions. Reliab. Eng. Syst. Saf. **85**(1), 39–71 (2004). https://doi.org/10.1016/j.ress.2004.03.025
24. Baruah, S.K., Hickey, M.E.: Competitive on-line scheduling of imprecise computations. IEEE Trans. Comput. **47**(9), 1027–1032 (1998). https://doi.org/10.1109/12.713322
25. Kim, J.H., Boudjadar, A., Nyman, U., Mikucionis, M., Larsen, K.G., Lee, I.: Quantitative schedulability analysis of continuous probability tasks in a hierarchical context. In: 2015 18th International ACM SIGSOFT Symposium on Component-Based Software Engineering (CBSE), pp. 91–100 (2015). https://doi.org/10.1145/2737166.2737170
26. Kim, K., Lo Bello, L., Min, S.L., Mirabella, O.: On relaxing task isolation in overrun handling to provide probabilistic guarantees to soft real-time tasks with varying execution times. In: Proceedings of the 14th Euromicro Conference on Real-Time Systems. ECRTS 2002, p. 193. IEEE Computer Society. USA (2002)
27. Kopetz, H.: Real-Time Systems - Design Principles for Distributed Embedded Applications. Real-Time Systems Series, Springer (2011). https://doi.org/10.1007/978-1-4419-8237-7
28. Larsen, K.G., Legay, A.: Statistical model checking past, present, and future. In: Margaria, T., Steffen, B. (eds.) ISoLA 2014. LNCS, vol. 8803, pp. 135–142. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-662-45231-8_10
29. Liu, J.W.S., et al.: Algorithms for Scheduling Imprecise Computations, pp. 203–249. Springer, Boston (1991). https://doi.org/10.1007/978-1-4615-3956-8_8
30. Liu, C.L., Layland, J.W.: Scheduling algorithms for multiprogramming in a hard-real-time environment. J. ACM **20**(1), 46–61 (1973). https://doi.org/10.1145/321738.321743
31. Markovic, F., Carlson, J., Dobrin, R., Lisper, B., Thekkilakattil, A.: Probabilistic response time analysis for fixed preemption point selection. In: 2018 IEEE 13th International Symposium on Industrial Embedded Systems (SIES), pp. 1–10 (2018). https://doi.org/10.1109/SIES.2018.8442099
32. Maxim, D., Cucu-Grosjean, L.: Response time analysis for fixed-priority tasks with multiple probabilistic parameters. In: 2013 IEEE 34th Real-Time Systems Symposium, pp. 224–235 (2013). https://doi.org/10.1109/RTSS.2013.30
33. Mok, A.K., Chen, D.: A multiframe model for real-time tasks. IEEE Trans. Software Eng. **23**(10), 635–645 (1997). https://doi.org/10.1109/32.637146

34. Peng, B., Fisher, N.: Parameter adaptation for generalized multiframe tasks: schedulability analysis, case study, and applications to self-suspending tasks. Real-Time Syst. **53**(6), 957–986 (2017). https://doi.org/10.1007/s11241-017-9279-2
35. Peng, B., Fisher, N., Chantem, T.: Fast and effective multiframe-task parameter assignment via concave approximations of demand. In: 31st Euromicro Conference on Real-Time Systems (ECRTS 2019). Leibniz International Proceedings in Informatics (LIPIcs), vol. 133, pp. 20:1–20:22. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany (2019). https://doi.org/10.4230/LIPIcs.ECRTS.2019.20
36. Sha, L., et al.: Real-time scheduling theory: a historical perspective. Real-Time Syst. **28**(2–3), 101–155 (2004). https://doi.org/10.1023/B:TIME.0000045315.61234.1e
37. Shan, L., Graf, S., Quinton, S.: RTLib: a library of timed automata for modeling real-time systems. Research report, Grenoble 1 UGA - Université Grenoble Alpe ; INRIA Grenoble - Rhone-Alpes (2016). https://hal.science/hal-01393888
38. Singhoff, F., Legrand, J., Nana, L., Marcé, L.: Cheddar: a flexible real time scheduling framework. Ada Lett. **XXIV**(4), 1–8 (2004). https://doi.org/10.1145/1046191.1032298
39. Tanasa, B., Bordoloi, U.D., Eles, P., Peng, Z.: Probabilistic response time and joint analysis of periodic tasks. In: 2015 27th Euromicro Conference on Real-Time Systems, pp. 235–246 (2015). https://doi.org/10.1109/ECRTS.2015.28
40. Wilhelm, R., et al.: The worst-case execution-time problem – overview of methods and survey of tools. ACM Trans. Embedded Comput. Syst. **7**(3), 36:1–36:53 (2008). https://doi.org/10.1145/1347375.1347389
41. Zio, E., Pedroni, N.: Uncertainty characterization in risk analysis for decision-making practice [on line]. Cahiers de la Sécurité Industrielle **1**(07) (2012). https://www.foncsi.org/en/publications/collections/industrial-safety-cahiers/uncertainty-QRA
42. Čus, S.: Schedulability Analysis of Real-Time Tasks under Uncertainty. Bachelor's thesis, Brno University of Technology, Faculty of Information Technology (2024). https://www.vut.cz/en/students/final-thesis/detail/150944, bc. thesis (in Czech only)

Verification for Neuro-Symbolic Artificial Intelligence

A Parametric Model for Near-Optimal Online Synthesis with Robust Reach-Avoid Guarantees

Mario Gleirscher(✉) and Philip Hönnecke

University of Bremen, Bibliothekstraße 5, Bremen, Germany
{gleirsch,hoennecke}@uni-bremen.de

Abstract. To obtain explainable guarantees in the online synthesis of optimal controllers for high-integrity cyber-physical systems, we re-investigate the use of exhaustive search as a classical alternative to reinforcement learning. We model an application scenario as a hybrid game automaton, enabling the synthesis of robustly correct and near-optimal controllers online without prior training. We allow model uncertainty through disturbed dynamics yielding a robust reference signal for lower-level trajectory tracking. For modal synthesis, we employ discretised games solved via scope-adaptive and step-pre-shielded discrete dynamic programming. In a simulation-based experiment, we apply our approach to an autonomous aerial vehicle scenario. We propose a parametric system model and a parametric online synthesis.

Keywords: Cyber-physical systems · low-level planning · assurance · autonomous aerial vehicles · VTOL aircraft

1 Introduction

To achieve complex and critical tasks, systems with a high grade of autonomy perform decision making and control at strategic and tactical levels under sparse human-machine interaction. Consider, for example, operation of autonomous aerial vehicles (AAVs) as illustrated in Fig. 1. With guidance from the strategic level (e.g., navigation, planning, or route finding), at the tactical level (e.g., obstacle-aware route segment tracking), often control problems of non-linear, disturbed dynamical systems have to be solved by constructing provably robust (i.e., for safety) and near-optimal (i.e., for minimum-cost reachability) controllers. To accommodate uncertainties (e.g., changing environments) and perform at scale, this type of controller synthesis is preferably done online, that is, during operation and right before the actual use of the synthesised controllers.

Reinforcement learning (RL) [25], a versatile and efficient adaptive [20] type of approximate dynamic programming (ADP) frequently used in online synthesis, has shown to have assurance-related drawbacks:

Relevant parts were developed when both authors were affiliated with U Bremen.

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 259–280, 2026.
https://doi.org/10.1007/978-3-032-01377-4_14

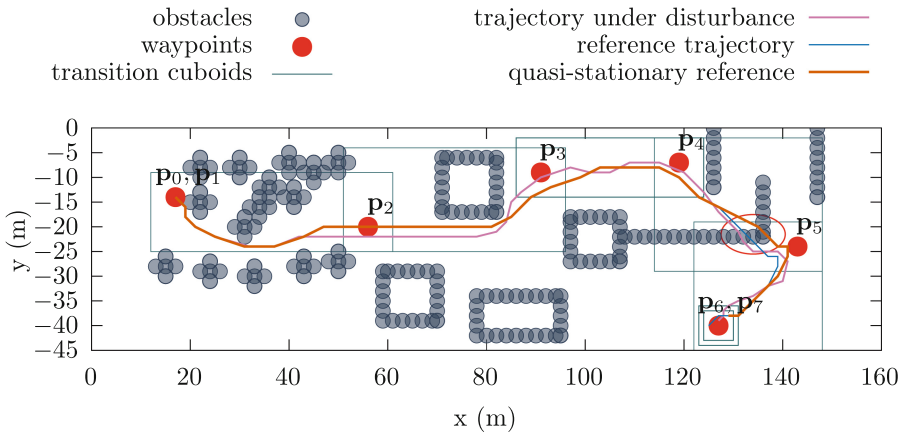


Fig. 1. 2D view of an AAV following a route $\bar{r}$ from waypoint $\mathbf{p}_0$ to $\mathbf{p}_7$. Controlled reference trajectory (solid line), trajectory under wind disturbance (dashed line), collision-freedom during shortcuts (ellipse, see Fig. 2)

- partial, imprecise, or even missing guarantees (due to a lack of behavioural coverage, inappropriate initialisation [5, Sec. 6] or reward signal [25, p. 470]),
- post-hoc explanations (i.e., identifying failure causes may require training even with transparent algorithms [13]),
- lack of robustness to changing operational profiles (e.g., obstacle scenarios not covered or differing from training setups [24, p. 30] [6]), and
- limited or costly training (off-line/simulation or on-line/real interaction; requires active supervision, e.g., real-time safety shielding [17,27]).

These limitations together with a need for more precise and complete guarantees revive exhaustive, hence, less efficient forms of synthesis. However, to keep computation within bounds while guaranteeing critical properties, online synthesis, whether or not based on RL or heuristic search, implies restrictive trade-offs (e.g., predictive imprecision, latencies in segment tracking).

Research Question. Not focusing on real-time performance, which kinds of robust reach-avoid guarantees can exhaustive numerical algorithms for near-optimal online synthesis provide, in which architectural settings and under which assumptions?

Approach and Application. For our running example and case study, we focus on an AAV-based delivery setting (Figs. 3 and 5) formalised as a

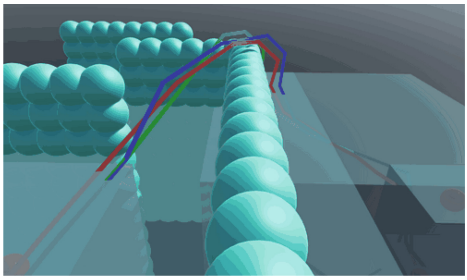


Fig. 2. 3D view of tracking the route segment from waypoint $\mathbf{p}_4$ to $\mathbf{p}_5$ in Fig. 1

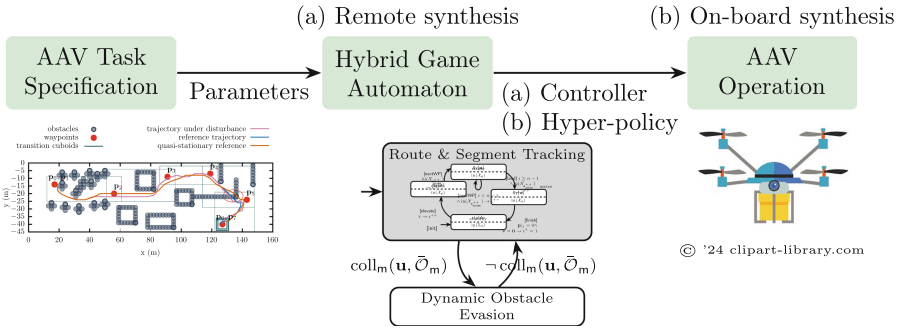


Fig. 3. Assured online synthesis with (a) remote or (b) on-board computation

jump-controlled hybrid game automaton (HGA) [12]. The HGA enables tactical controller synthesis via game solving done online (i.e., during operation) and using discrete dynamic programming (DDP) [4]. For each mode transition—called *jump*—in the HGA (e.g., at $\mathbf{p}_2$ in Fig. 1), our DDP algorithm (pre-) computes a solution—called *policy*—for the modal reach-avoid game following this jump. That policy is then used as a modal controller until another jump is triggered by reaching the *goal region* of that game (e.g., the rectangle enclosing $\mathbf{p}_3$ and $\mathbf{p}_4$ in Fig. 1). The *unsafe region* to be avoided by the policy is defined via a perforated,[1] fixed obstacle cloud. Our game setting permits bounded-uncertain wind disturbance. Moreover, unsafe actions possibly enabled by the discretisation are *filtered* during DDP before finishing a search stage. This filter corresponds to pre-shielding, which improves optimality over post-shielding because of being dynamically less restrictive [19]. A *supervisor* [8], a higher-level controller (Fig. 3 middle), coordinates the *interruption* of the tactical controller by an obstacle evasion unit if moving obstacles approach the AAV trajectory predicted based on the current policy.

Related Work. Online synthesis of controllers for autonomous systems has been a research subject for decades, with a steady activity around robustly correct, near-optimal control. Below, we highlight some more recent works.

Real-time-capable online synthesis has been proposed for special-purpose control of single- and multi-agent systems. For example, Dolgov et al. [6] apply heuristic search (using A*) for discretised 2D vehicle path planning with quadratic programming for trajectory optimisation, real-time obstacle detection, and re-planning. Li et al. [21] apply a fast model-predictive control (MPC) scheme focusing on stabilisation around a given continuous reference trajectory. They use a quadratic cost term enabling initialised, sequential quadratic programming. For AAV collision avoidance, Bertram et al. [3] and Taye et al. [26] realise multi-agent reachability-based state space reduction to accelerate backward Markov decision process (MDP) policy search. Because of the scenario-

[1] Start and end of a given route must be connected with a wide enough tube.

sensitive parameterisation, our technique can currently not be used in fast real-time settings, such as collision avoidance in autonomous driving [21]. In comparison, our 7D approach is significantly slower but more widely applicable, allowing non-convex cost structures, sparse reference trajectories, near-optimality, and guaranteed reach-avoidance under bounded disturbance, and it is integrated into a flexible multi-tier hybrid control scheme (Fig. 3 middle).

Scalable offline synthesis has been developed for discrete and hybrid single- and multi-agent systems. For example, Ivanov et al. [15] apply deep-RL to compute contract-compliant controllers for each mode of a hybrid automaton of an autonomous vehicle. Gu et al. [11] use RL to scale synthesis for timed stochastic multi-agent systems. In [23], we refine a stochastic Petri-net abstraction with partial state observability to generate optimal schedules for tasked robot collectives. By approximating fixpoints and sacrificing stochasticity in the weighted reach-avoid game setting, we bypass an even higher complexity of exact MDP synthesis. This enables us to pre-process up to 200 Mio. states, going beyond the state space in [23] in a time scale similar to backward reachability (e.g., [28]), forward reachability (e.g., [15]), and RL-based statistical MDP [11] synthesis.

Regarding *safe RL*, the efficiency of our approach is currently more on the side of offline learning techniques in this area, such as [1,16]. However, our approach provides time-bounded reach-avoid guarantees [9] without the training needed for RL, which can be intense in model-free settings. Not relying on training can be beneficial in changing environments where (a) approximate fixed-obstacle data is available only for some route segments. (b) Approximate knowledge of the system dynamics (not required by model-free RL) is compensated for by robustness to process disturbance and imprecise state observations. (a) is frequently available in the settings we consider [6] and (b) is an assumption usually made by approaches, such as refined heuristic search [6] and MPC [21].

Contributions. Our approach enhances previous work as follows:

(i) *Step-shielded online synthesis:* We provide an algorithm for solving discretised modal games with reach-avoid winning conditions. The algorithm uses step-shielded DDP and supports non- and quasi-stationary policies (increasing robustness to disturbances at the cost of performance [20]). We slightly reduce DDP's curse-of-dimensionality problem by scope adaptation[2] and fixpoint approximation for the winning region.

(ii) *Parametric hybrid game model:* We construct a parametric weighted HGA covering a range of typical AAV scenarios. Concerning neuro-symbolic architectures [18], the model enables a three-tier separation of control to allow a trading off of operational cost and flexibility: higher-level supervision (e.g., moving obstacle evasion), near-optimal trajectory tracking, and sub-tactical control based on locally learned policies (e.g., multi-rotor control via deep-RL). We embed our algorithm into a player for the HGA.

[2] The relevant fraction of the state space is selected according to the current system state and extended on-the-fly to increase solvability of the online synthesis problem.

A preliminary variant of (i) and (ii) was implemented and evaluated in [14]. Key assumptions and proofs for the correctness of our approach (incl. algorithmic convergence) are provided in a compagnion working paper [9].

Outline. After the preliminaries (Sect. 2), we describe our AAV application and control problem (Sect. 3). Our contributions to online synthesis with guarantees follow in the Sect. 4. We evaluate our approach experimentally (Sect. 6) and close with a discussion and remarks (Sections 7 and 8).

2 Preliminaries

Notation. For a set of variables $Var = X \uplus U \uplus D$, let $\mathbb{X} \subset \mathbb{Z}^X$ be an X-typed, finite, m-dimensional Euclidean *state space*, and let $\mathcal{U}$ and $\mathcal{D}$ be U- and D-typed *control* and *disturbance* ranges, each including $\mathbf{0}$. For a 3D coordinate $\mathbf{p}$, we use the typical naming convention $\mathbf{p} = (x, y, z)^\intercal$.

Moreover, let the classes of terms $\mathcal{T}(Var)$ and constraints $\mathcal{C}(Var)$ range over Var. $\|\cdot\|$ is the corresponding 2-norm, $A \oplus B = \{a + b \mid a \in A \wedge b \in B\}$ the Minkowski sum of $A, B \subseteq \mathbb{X}$, and $A|_{Var'}$ the projection of tuples in A to variables in $Var' \subseteq Var$, and $\dashrightarrow$ indicates a partial map. $\underline{I}$ and $\bar{I}$ denote the lower and upper bounds of an interval $I \subseteq \mathbb{R}$ and $\underline{I}..\bar{I}$ signifies that $I \subseteq \mathbb{Z}$. We omit set parentheses when referring to singleton sets in subscripts. Pointwise operators are lifted as usual. For example, $\min, \max$ on vectors of sets are evaluated element-wise and return a vector of scalars. Negation of a set returns the set of negated elements.

The map $[\![\varphi]\!]_{\mathfrak{M}}$ denotes φ's models in domain $\mathfrak{M}$, formally, $[\![\varphi]\!]_{\mathfrak{M}} = \{M \in \mathfrak{M} \mid M \models \varphi\}$. $\mathfrak{M}$ can, for example, be the class of states, state pairs, sequences, or transition systems. Given the finite sequences $\mathbb{X}^\star$ over $\mathbb{X}$, the length $|\bar{x}|$ of a sequence $\bar{x} \in \mathbb{X}^\star$, and its value $\bar{x}_k \in \mathbb{X}$ at position $k \in 1..|\bar{x}|$, we use $\bar{\mathbb{X}}(\Delta_\delta), \tilde{\mathbb{X}} \subset \mathbb{X}^\star$ to denote the classes of *trajectories* with any two subsequent states inside some $\Delta_\delta \subseteq \mathbb{Z}^m$ and with $\tilde{\mathbb{X}} = \bar{\mathbb{X}}([\pm 1]^m)$. Note that we use * for Kleene closures and * to indicate fixpoints or optimal solutions.

Weighted Hybrid Game Automata. Given a set of modes Q, action labels A, events $E \subseteq Q \times A \times Q$, and the hybrid state space $\mathcal{S} = Q \times \mathbb{X}$, a weighted HGA $G = (Gra, Var, Ini, Inv, f, Jmp, F)$ [7,12] comprises a mode transition graph $Gra = (Q, A, E)$, initial conditions $Ini\colon Q \to \mathcal{C}(X)$, invariants $Inv\colon Q \to \mathcal{C}(X)$, flow conditions $f\colon \mathcal{S} \to \mathcal{C}(Var \cup \dot{X})$, jump conditions $Jmp\colon E \to \mathcal{C}(Var \cup X^+)$ comprised of guards and updates, and cost structures $F\colon \mathcal{S} \to 2^{\mathcal{T}(Var)}$ as weights generalising final conditions $Fin\colon Q \to \mathcal{C}(X)$.

Flow and jump conditions [7,12] provide a flexible way to specify (non-deterministic) hybrid dynamics interleaving continuous evolutions and discrete jumps. The copies $\dot{X}$ and X^+ refer to the time derivatives and discrete updates of X. $\mathrm{grd}(e)$ and $\mathrm{upd}(e)$ denote the guard and update conditions of $Jmp(e)$. In the remainder, we frequently use $\mathbf{s} = (q, \mathbf{x}) \in \mathcal{S}$ for a state of G.

Integer Difference Games. With inspiration from [28], we associate a state $\mathbf{s}$ of G with a discretised two-player game $G_{\mathbf{s}} = (\mathcal{X}, \hat{f}, F)$ where $\hat{f}$ is the discretised Taylor expansion of the flow condition f corresponding to $\mathbf{s}$. A *policy* describes how a player controls (i.e., resolves their choices in) that game at each state and time. Players share the current state and time but do not know each others' next actions. Let $\Pi_{\mathcal{Y}} = \mathbb{X} \times \mathbb{N}_+ \dashrightarrow \mathcal{Y}$ be a policy space for a player with control range $\mathcal{Y}$. Given non-empty control and disturbance ranges $\mathcal{U}$ and $\mathcal{D}$, $\mathbf{u}\colon \Pi_{\mathcal{U}}$ and $\mathbf{d}\colon \Pi_{\mathcal{D}}$ denote the policies of the two players, the controller and the environment.

Playing a weighted game using some $(\mathbf{u}, \mathbf{d})$ for a bounded period incurs *costs* at each *stage* and on *termination*. Given a horizon N and a cost structure $F = (L, \Phi)$ with stage and terminal costs L and Φ, we are interested in a finite-horizon, discrete optimal policy $\mathbf{u}^*\colon \Pi_{\mathcal{U}}$, which can be obtained from solving a constrained, discrete dynamic optimisation problem [4]. Such a problem can be solved with a forward-Euler DDP (Algorithm 1) for the period $I = 1..N$.

Algorithm 1 Discrete dynamic programming

1: **procedure** dDP(in $G_{\mathbf{s}}, I$; out $V, \mathbf{u}^*$)
2: $(V, \mathbf{u}^*) \leftarrow \bot^{|\mathcal{X}| \times I}$ ▷ $\bot$ … *undefined or maximal, e.g.,* $\top$
3: $V(\mathcal{X}, \bar{I}) \leftarrow \Phi(\mathcal{X}, \bar{I})$ ▷ *initialise terminal cost*
4: **for** $k \in [\underline{I}, \bar{I} - 1]$ **do** ▷ *implies backward iteration*
5: **for** $\mathbf{x} \in \mathcal{X}$ **do**
6: $\mathbf{x}^{\mathcal{U}\mathcal{D}} \leftarrow \mathbf{x} + \hat{f}(\mathbf{x}, \mathcal{U}, \mathcal{D}, k)$ ▷ *forward unit Euler set*
7: $V(\mathbf{x}, k) \leftarrow \min_{u \in \mathcal{U}} \max_{d \in \mathcal{D}} \{L(\mathbf{x}, u, d, k) + V(\mathbf{x}^{ud}, k+1)\}$ ▷ *value*
8: $\mathbf{u}^*(\mathbf{x}, k) \leftarrow \arg\min_{u \in \mathcal{U}} \max_{d \in \mathcal{D}} \{L(\mathbf{x}, u, d, k) + V(\mathbf{x}^{ud}, k+1)\}$
9: **if** $\widehat{fp}_{\mathrm{dDP}}(k)$ **then break** ▷ *approximate fixpoint found?*

The single-step successor $\mathbf{x}^{ud} = \mathbf{x} + \hat{f}(\mathbf{x}, u, d, k)$ is lifted in Line 6, such that $\mathbf{x}^{ud} \in \mathbf{x}^{\mathcal{U}\mathcal{D}} \subseteq \mathbb{X}$. Moreover, let $\mathbf{x}^{\mathbf{u}\mathbf{d}} \in \bar{\mathbb{X}}$ be the N-step successor (trajectory) of G emanating from $\mathbf{x} \in \mathbb{X}$ under influence of $(\mathbf{u}, \mathbf{d})$, and $\mathbf{x}^{\mathbf{u}\mathcal{D}} \subseteq \bar{\mathbb{X}}$ be the family of such trajectories under $\mathcal{D}$. $V(\mathbf{x}, k) \in [0, \top]$ with $\top < \infty$ is the finite-horizon *value* reflecting the minimal cost incurred over the course of a play under optimal policies $(\mathbf{u}^*, \mathbf{d}^*)$. Line 9 allows a check $\widehat{fp}_{\mathrm{dDP}}$ for whether a fixpoint is approximated prior to reaching N. While the actual fixpoint yields stationary optimal policies, a fixpoint approximation still guarantees bounded correct (quasi-stationary and near-optimal) policies under reduced memory consumption and computational effort [9, Lemma 2]. Let $\mathcal{G}$ be the class of integer difference games.

3 Aerial Delivery as a Hybrid Game

Modelling Assumptions. Let $X = \{\mathbf{p}, \mathbf{v}, i\}$ be the set of state variables with the AAV *position* $\mathbf{p}\colon \mathbb{Z}^3$ and *velocity* $\mathbf{v}\colon [\pm v_{\max}]^3$ (with an absolute maximum at $v_{\max}$) and a *next-waypoint* index $i\colon \mathbb{N}$. $\mathbf{p}|_z$ measures distance above local ground outside buildings. We use convex sets $\mathcal{X}_{\mathbf{s}} \subseteq \mathbb{X}$ called *scopes* and the space $\mathcal{P} = \mathbb{X}|_{\mathbf{p}}$ to hold the position grid of the *scenery*. $\mathcal{X}_{\mathbf{s}}$ is not included as a

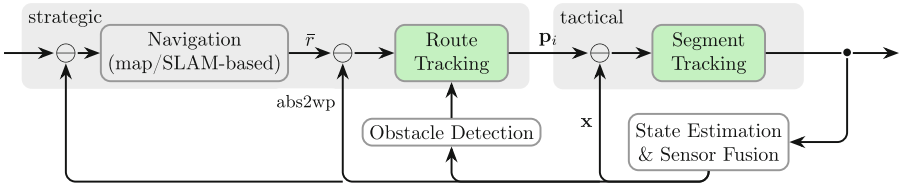


Fig. 4. Overall AAV control block structure

state variable in X because it can be derived from $\mathbf{s}$ as described in (2) below. In the AAV scenarios discussed in this work, the scopes $\mathcal{X}_\mathbf{s}$ are cuboids.

The language (i.e., the set of accepted traces) of the HGA used for the AAV (see Fig. 5a) is given by the regular expression $\mathsf{init}\ (\mathsf{elevate}\ \mathsf{nextWP}\ \mathsf{nextWP}^\star\ \mathsf{land}\ \mathsf{finish})^\star$. To produce a word of this language, a minimum of four (not necessarily different) waypoints are needed as well as a "sparse enough" or perforated obstacle cloud. Below, we will make these conditions more precise.

Given $\mathbb{X}$, an AAV *task* $T = (\mathbb{X}, \bar{r}, \mathcal{O})$ comprises a *fixed obstacle cloud* $\mathcal{O} \subset \mathcal{P}$ and a *route* $\bar{r} = \{\mathbf{p}_i\}_{i=1}^n$ where the predicate valid$(\bar{r})$ requires $\bar{r}$ to have $n \geq 4$ *waypoints* $\mathbf{p}_i \in \mathcal{P}$ with $\mathbf{p}_1, \mathbf{p}_n$ on the ground, $\mathbf{p}_2, \mathbf{p}_{n-1}$ (x, y)-superimposed on $\mathbf{p}_1, \mathbf{p}_n$, and $\mathbf{p}_{2..n-1}$ residing above a minimum height $z_{\min}$. This scheme permits elevation and landing without a horizontal move.

Given a cube $\Delta_\delta = [\pm\delta]^3$ for a small $\delta \in \mathbb{N}_0$, we require the obstacle cloud $\mathcal{O}$ to be δ-*perforated*, such that

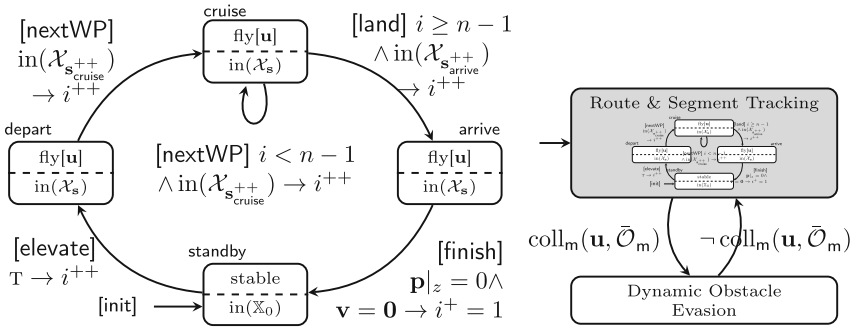
$$\exists \bar{x} \in \tilde{\mathbb{X}} \colon \bar{r} \subseteq \underbrace{\bar{x}|_\mathbf{p} \oplus \Delta_\delta}_{\text{cont. } \delta\text{-tube}} \wedge \bar{x}|_\mathbf{p} \oplus \Delta_\delta \cap \mathcal{O} = \varnothing\,. \tag{1}$$

This condition requires the route to be enclosed in a tube with radius δ not colliding with fixed obstacles. We assume $(\bar{r}, \mathcal{O})$ to be delivered by map- and SLAM-based navigation and sensor fusion on-board the AAV (Fig. 4) and allow $(\bar{r}, \mathcal{O})$ to be updated at every $\mathbf{p}_i$ invariant under (1) and valid$(\bar{r})$.

In Fig. 5 and below, we use predicates over $\mathcal{O}, \mathcal{O}_\mathsf{m}, \bar{r}$, and $\mathbf{u}$. $\mathcal{O}$ and $\bar{r}$ are game parameters, $\mathbf{u}$ is a parameter of the tactical controller's actions (Sect. 4), and $\mathcal{O}_\mathsf{m}$ and $\mathbf{u}$ are parameters of the supervisor's actions.

Tactical Control Scope. We consider a HGA G (Sect. 2) combining the logic of the AAV route and segment tracking units, where *Gra*, *Inv*, f, and *Jmp* are illustrated in Fig. 5a and $\mathit{Ini}(q) = \mathrm{T}$ for $q \in Q$. For brevity, (1) and valid$(\bar{r})$ as global invariants remain implicit in *Ini* and *Inv*. To compute the current scope, we use the function

$$\mathcal{X}_\mathbf{s} = \begin{cases} (\mathbf{p}_i, \mathbf{0}) \oplus \left([\pm\delta_{p,q}]^2 \times [0, \max\{\mathbf{p}, \mathbf{p}_i\} + \delta_{p,q}] \times [\pm\delta_{v,q}]^3\right) \times \{i\}, \\ \qquad \text{if } q \in \{\mathsf{depart}, \mathsf{arrive}, \mathsf{standby}\}\,, \\ \left([\min\{\mathbf{p}, \mathbf{p}_i\}, \max\{\mathbf{p}, \mathbf{p}_i\}] \times \{\mathbf{0}\}\right) \oplus \left([\pm\delta_{p,q}]^3 \times [\pm\delta_{v,q}]^3\right) \times \{i\}\,, \\ \qquad \text{if } q \in \{\mathsf{cruise}\}\,. \end{cases} \tag{2}$$



a: Parametric tactical control logic $G[\mathbf{U}, T]$ b: Supervisor logic

Fig. 5. Tactical AAV control (f/*Inv* are above/below the dashed lines in the modes, and *Jmp* is shown in the transition labels) and supervisor logic

$\mathcal{X}_\mathbf{s}|_\mathbf{p}$, the $\mathbf{p}$-projection of $\mathcal{X}_\mathbf{s}$, describes *extensible transition cuboids* (Fig. 6, see the examples in Fig. 1 left and right and Fig. 9 below) around route segments for moments when the AAV is near one waypoint and proceeds to the next. Likewise, $\mathcal{X}_\mathbf{s}|_\mathbf{v}$, the $\mathbf{v}$-projection of $\mathcal{X}_\mathbf{s}$, describes the space of feasible velocities in mode q. We use $\delta_{p,q}$ and $\delta_{v,q}$ for position and velocity padding in q, as illustrated for positions in Fig. 6. Furthermore, we use parametric abbreviations for an invariant $\text{in}(\mathcal{X}) \equiv \mathbf{x} \in \mathcal{X}$, an initial region $\mathbb{X}_0 \equiv \{\mathbf{x} \in \mathbb{X} \mid \mathbf{p}|_z = 0 \wedge \mathbf{v} = \mathbf{0} \wedge i = 1\}$, a successor scope $\mathbf{s}_q^{++} = (q, (\mathbf{p}_i, \mathbf{0}, i+1)^\intercal)$, and a next waypoint index $i^{++} \equiv i^+ = i+1$. Note how $\mathbf{s}_q^{++}$ refers to the next waypoint $\mathbf{p}_i$ and the one after that, $\mathbf{p}_{i+1}$. $\mathbf{s}_q^{++}$ will be used to compute the goal regions for the modal games.

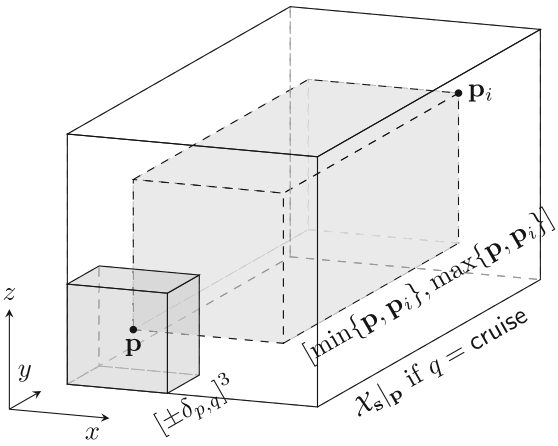


Fig. 6. $\mathbf{p}$-projection of scope $\mathcal{X}_\mathbf{s}$ from (2) as a result of position padding with $\delta_{p,q}$

Tactical Control Dynamics. We employ a simplification of the AAV dynamics to a point-mass dynamics. Using such an abstraction, the policy $\mathbf{u}^*$ induces a reference trajectory (Fig. 1) for *segment tracking* (Fig. 4) and a reference acceleration signal $\bar{u} \in \mathcal{U}^\star$ that lower-level multi-rotor control can track. We use a constant unit mass $m = 1 = m_{\text{AAV}} + m_{\text{payload}}$, such that m can stay implicit and

$\bar{u}$ becomes a force signal. Concerning robustness, our abstraction can be seen as a trade-off between relying on known dynamics and model-free RL.

Hence, the dynamics in all modes, except for standby, is given by

$$\mathit{fly} = \dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{p}} \\ \dot{\mathbf{v}} \\ \dot{i} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{u} + \mathbf{d} + \mathbf{g} \\ 0 \end{bmatrix} \tag{3}$$

with an approximate gravitational acceleration $\mathbf{g} = (0, 0, -10)^\intercal$ and $\mathbf{u} \in \mathcal{U}$, $\mathbf{d} \in \mathcal{D}$ for $\mathcal{U}, \mathcal{D} \colon 2^{\mathbb{Z}^3}$. In standby mode, we use the flow condition $\mathit{stable} \equiv \dot{\mathbf{x}} = \mathbf{0}$. Below, we omit $\mathbf{g}$ in the isolated dynamics. Numerical imprecision is assumed to be compensated by lower-level stability control and system identification.

In our experiments, we use small $\mathcal{U}, \mathcal{D} \subseteq [\pm 2]^3$ to allow the AAV to accelerate in 26 directions and wind disturbance to occur only in 4 directions, as well as $v_{\max} = 10$ ranging down to 5 for efficiently handling smaller scopes.

Supervisory Control. Strategic AAV control comprises a supervisor (Fig. 5b), which separates route and segment tracking from moving-obstacle evasion. The supervisor interrupts tactical control and resumes it after an evasion manoeuvre. Encoded in F, for all modes but standby, the unsafe set encompasses (inevitable) collisions with fixed obstacles as

$$\mathrm{coll}_{\mathsf{f}}(\mathcal{O}) \equiv \exists \mathbf{o} \in \mathcal{O} \colon \|\mathbf{o} - \mathbf{p}\| \leq \delta, \quad \text{if } q \neq \mathsf{standby}. \tag{4}$$

With $\|\mathbf{o} - \mathbf{p}\| \leq \delta$ instead of $\mathbf{p} \in \mathbf{o} \oplus \Delta_\delta$, we can work with $\delta = 1.5$ in experiments.

For a set $\tilde{\mathcal{O}}_{\mathsf{m}}$ of trajectories of moving obstacles tracked by the supervisor,

$$\mathrm{coll}_{\mathsf{m}}(\mathbf{u}, \tilde{\mathcal{O}}_{\mathsf{m}}) \equiv \exists k \in I, \bar{o} \in \tilde{\mathcal{O}}_{\mathsf{m}} \colon \mathbf{x}_k^{\mathbf{u}\mathcal{D}}|_{\mathbf{p}} \cap \bar{o}_k \oplus \Delta_\delta \cap \mathbf{p} \oplus \Delta_{sbd}(\mathbf{v}) \neq \varnothing$$

indicates that the predicted trajectory $\bar{o}$ of some moving obstacle crosses the AAV trajectory $\mathbf{x}^{\mathbf{u}\mathcal{D}}|_{\mathbf{p}}$ within safe braking distance Δ_{sbd}. Evasion manoeuvres (i.e., replacing $\mathbf{u}$ by an evasive $\mathbf{u}_e$) could be computed by Algorithm 1. However, there are other ways of computing $\mathbf{x}^{\mathbf{u}\mathcal{D}}|_{\mathbf{p}}$ efficiently (e.g., [2,27]) and deriving $\mathbf{u}_e$ with guarantees (e.g., [22]). To simplify our setting, we assume $\tilde{\mathcal{O}}_{\mathsf{m}} = \varnothing$ and refer to page 17 as well as broader treatments of supervision (e.g., [8]).

Let us now formulate the synthesis problem focused on in this work.

Definition 1 (Online Synthesis Problem). *Given a hybrid game automaton G and a task T, continuously find tactical controllers $\mathbf{u}$ steering the system along route $\bar{r}$ while safely circumventing unsafe regions. For correctness, we need $\mathbf{u}$ to be (i) δ-robust (i.e., safe under bounded disturbance $\mathcal{D}$), (ii) near-optimal (i.e., follow the minimum-cost path inside padded segment scopes, approximating all waypoints), and (iii) reaching the endpoint of $\bar{r}$.*

4 Online Synthesis via Parametric Games

Parametric Modal Games. We assume that *Gra* is controllable (i.e., all jumps controlled by the system) such that G reduces to a parametric reach-avoid integer difference game $G_{\mathbf{s}} = (\mathcal{X}_{\mathbf{s}}, \hat{f}, F)$ to be solved for and played after a jump to $\mathbf{s}$. The forward-Euler scheme in Line 6 of Algorithm 1 employs an isolated variant of the right-hand side of $\hat{f}$. In particular, the AAV setting uses (3) without $\mathbf{g}$ and changes its variables via $\mathbf{x}_{\text{iso}} = \mathbf{x}_{\text{orig}} - (\mathbf{p}_i, \mathbf{0}, 0)^\intercal$.

To specify reach-avoid winning conditions, we use a parametric global cost structure $F = (L, \Phi)$ defined as

$$L(\mathbf{u}, \mathbf{d}; \mathbf{s}, k) = \begin{cases} 0, \text{ if } \rho \wedge \neg\alpha \\ \top, \text{ if } \alpha \text{ (shielded)} \\ \lambda(\mathbf{u}, \mathbf{d}; \mathbf{x}, k), \text{ otherwise} \end{cases} \qquad \Phi(\mathbf{s}) = \begin{cases} 0, \text{ if } \rho \wedge \neg\alpha \\ \top, \text{ otherwise} \end{cases} \tag{5}$$

where ρ and α specify the goal and unsafe regions to be reached and avoided, respectively. We assume that $[\![\rho]\!] \subseteq \mathcal{X}_{\mathbf{s}}$ and $[\![\alpha]\!] \subseteq \mathbb{X}$ with $[\![\rho]\!] \setminus [\![\alpha]\!] \neq \varnothing$. Furthermore, $\lambda(\mathbf{u}, \mathbf{d}; \mathbf{x}, k) = \mathbf{x}^\intercal P \mathbf{x} + \mathbf{u}^\intercal Q \mathbf{u} + \mathbf{d}^\intercal R \mathbf{d}$ is a weighting term with correspondingly dimensioned square matrices P, Q, and R.

L penalises α and rewards ρ, maximally. Given (1), (5) implies $V(\mathbf{0}, \cdot) = 0$. Not shown here, k can be used in λ to penalise run-time. Overall, a call $\text{dDP}(G_{\mathbf{s}}, I)$ to Algorithm 1 provides a near-optimal controller $\mathbf{u}^*_{\mathbf{s}}$. Deviating from Sect. 2, we pass q to dDP by instantiating L and Φ with $\mathbf{s} = (q, \mathbf{x})$.

Winning Games via Approximate Value Fixpoints. Let

$$W(\mathcal{X}_{\mathbf{s}}, k) = \{\mathbf{x} \in \mathcal{X}_{\mathbf{s}} \mid V(\mathbf{x}, k) < \top\}$$

be $G_{\mathbf{s}}$'s *k-winning region*, that is, an under-approximation of the region of states from where the system can win the game at time k when playing an optimal policy $\mathbf{u}^*$ for $N - k$ steps. In particular, $W(\mathcal{X}_{\mathbf{s}}, N) = \{\mathbf{x} \in \mathcal{X}_{\mathbf{s}} \mid \Phi(\mathbf{x}) < \top\}$. To reduce computational effort in Algorithm 1, we employ a condition $\widehat{fp}_{\text{dDP}}$ of premature termination in Line 9. This condition is defined as

$$\widehat{fp}_{\text{dDP}}(k) \equiv |W(\mathcal{X}_{\mathbf{s}}, k)| = |W(\mathcal{X}_{\mathbf{s}}, k+1)| \vee \widehat{fp}_{\mathbf{U}}(k)\,, \tag{6}$$

where the conjunct

$$\widehat{fp}_{\mathbf{U}}(k) \equiv \exists k' \geq k \colon \top \notin V(\mathbf{x} \oplus \Delta_\delta, k')$$

ensures that, latest at step k, $\mathbf{x}$ is δ-robustly located inside $W(\mathcal{X}_{\mathbf{s}}, k)$. This approximation allows us to check whether the number of states with a *robustly safe and goal-reaching trajectory* stabilises at k and, hence, within the horizon N.

Scope Adaptation. To further reduce computational effort, we use Algorithm 1 with a spatio-temporal *scope extension*. The result is Algorithm 2, with the aim to increase solvability of $G_{\mathbf{s}}$ by checking $\widehat{fp}_{\mathbf{U}}(1)$ in Line 7.

Algorithm 2 DDP with scope extension (in Line 4, note that $\mathbf{s} = (q, \mathbf{x})$)

1: **procedure** $\mathbf{U}_\delta$(in $G_\mathbf{s}, I$; out $V^{\bar{I}'}, \mathbf{u}^*, \mathcal{X}', I'$)
2: $(\mathcal{X}', I', \Delta'_\mathcal{X}, {\delta_I}') \leftarrow (\mathcal{X}_\mathbf{s}, I, 0, 0)$ ▷ *initialise scope and prediction interval*
3: **repeat**
4: $(\mathcal{X}', I') \leftarrow (\mathcal{X}' \oplus \Delta'_\mathcal{X}, \underline{I}'..\bar{I}' + {\delta_I}')$ ▷ *extend scope (not initially)*
5: $(V^{\bar{I}'}, \mathbf{u}^*) \leftarrow \mathrm{dDP}((\mathcal{X}', \hat{f}, F), I')$ ▷ *compute controller, Algorithm 1*
6: $(\Delta'_\mathcal{X}, {\delta_I}') \leftarrow (\Delta_\mathcal{X}, \delta_I)$ ▷ *use hyper-parameters* $\Delta_\mathcal{X}, \delta_I$
7: **until** $\widehat{fp}_\mathbf{U}(1)$ ▷ *is* $\mathbf{x}$ *robustly located in the* 1*-winning region?*
8: **return** $(V^{\bar{I}'}, \mathbf{u}^*, \mathcal{X}', I')$

Hyper-Policies and Pre-computation. Algorithm 2 realises a *hyper-policy* $\mathbf{U}_\delta \colon \mathcal{G} \times 2^{\mathbb{N}_+} \to \Pi_\mathcal{U}$ for the parametric $G[\mathbf{U}, T]$ in Fig. 5a. $\mathbf{u}^*_\mathbf{s} = \mathbf{U}_\delta(G_\mathbf{s}, I)$ is defined for $\mathcal{X}' \times I' \subset \mathbb{X} \times \mathbb{N}_+$ and solves $G_\mathbf{s}$ by determinising the $(\mathcal{U}, \mathcal{D})$-underspecified modal dynamics of G. $\mathbf{u}^*_\mathbf{s}$ is intended to be computed online, remotely or on-board (Fig. 3), if $\mathbf{x} \in \mathit{Inv}(q)$. Any suffix of $\bar{r}$ after $\mathbf{p}_i$ can be updated together with the corresponding update of $\mathbf{u}^*_\mathbf{s}$.

A Hybrid Game Player. Algorithm 3 combines Algorithm 2 with an execution routine for HGAs. A play of $G_\mathbf{s}$ is a co-execution of both players successively applying their policies, presumably $(\mathbf{u}^*, \mathbf{d})$, to the dynamics $\hat{f}$.

The loop (Lines 3 to 11) plays $G_\mathbf{s}$ starting from $\mathbf{x}_0 = \mathbf{s}|_\mathbf{x}$. For the possible jump in Line 5, the $\exists$ of the **switch** is to be read as "check and pick e". G as a specification is agnostic to when and how guards are used. We use guards as triggers because Algorithm 3 interprets G as a world emulator as close to real-time as possible. Consequently, updates will be performed as soon as a modal play enters the corresponding guard region.

The numerical integrator in Line 10 describes the simultaneous inputs $(\mathbf{u}^*, \mathbf{d})$ of both players being used in the dynamics and adds the corresponding increment to the current state $\mathbf{x}$, resulting in a hybrid trajectory $\bar{\mathbf{s}} \in \bar{\mathcal{S}}$.

Algorithm 3 Hybrid game player (with restricted task updates)

1: **procedure** HYBRIDPLAY(in $(G, \mathbf{U}_\delta, T)$; inout $\bar{\mathbf{s}}$)
2: $(\mathbf{s}, \rho, k) \leftarrow ((\mathsf{standby}, \mathbf{x}_0), \mathbf{p} \in \mathcal{X}_0|_\mathbf{p}, 1)$ ▷ *initialise state and goal region*
3: **repeat** $k{+}{+}$ ▷ *modal play for at most* N *steps*
4: **switch** $\exists e = (q, a, q') \in E \colon \mathbf{x} \in [\![\mathrm{grd}(e)]\!]$ **do** ▷ *non-deterministic mode update*
5: $(\mathbf{s}, \rho, k) \leftarrow ((q', \mathrm{upd}(e)(\mathbf{x})), \mathbf{p} \in \mathcal{X}^{++}_{\mathbf{s}_{q'}}|_\mathbf{p}, 1)$ ▷ *jump (e.g.,* $\mathbf{p}$*-invariant)*
6: $T \leftarrow$ UPDTASK(T) ▷ *e.g., update obstacles* $\mathcal{O}$ *and unsafe region* α
7: $G_\mathbf{s} \leftarrow (\mathcal{X}_\mathbf{s}, \hat{f}, F[\rho])$ ▷ *set scope and cost structure*
8: $\mathbf{u}^*_\mathbf{s} \leftarrow \mathbf{U}_{\delta,N}(G_\mathbf{s})$ ▷ *solve bounded modal game*
9: $\mathbf{d} \leftarrow$ GENDIST$(\bar{\mathbf{s}})$ ▷ *opponent's turn (e.g., random wind)*
10: $\mathbf{x} \leftarrow \mathbf{x} + \hat{f}(\mathbf{u}^*_\mathbf{s}, \mathbf{d}; \mathbf{x}, k)$ ▷ *move and feedback*
11: **until** $\Omega \vee \mathbf{x} \notin W(\mathcal{X}_\mathbf{s}, k) \vee k \geq N \vee \neg \mathit{Inv}(q)$ ▷ *termination condition*

Execution (Line 11) is constrained by several conditions: The first, Ω, specifies *termination*. The other three, $\mathbf{x} \notin W(\mathcal{X}_\mathbf{s}, k)$, $k \geq N$, and $\neg \mathit{Inv}(q)$, specify

failure (i.e., unsafe behaviour, $\mathbf{x} \notin [\![\rho \wedge \neg\alpha]\!]$ if $k = N$), *timeout* (beyond $\bar{I} = N$, control choices can no more rely on $V(\mathbf{x}, k) < \top$), and *invariant violation* (e.g., behaviour or an application of the controller outside $Inv(q)$ is not specified). The latter three events, by definition, only occur if the current modal game $G_{\mathbf{s}}$ could not be solved. In a simulator, these events can be used for model debugging, while during operational tests, such events may serve, for example, system identification and the adjustment of observers.

5 Algorithmic Correctness and Convergence

This section summarises the results from a companion working paper [9] about the correctness and convergence of the algorithms developed in this paper. Below, we provide the sketch of the proof structure.

In [9], we first show, by induction over a discrete Hamilton-Jacobi-Isaacs equation, that $W(\mathcal{X}_{\mathbf{s}}, k)$ is an inductive invariant of $\mathbf{u}^*$ [9, Lemma 2 and Corollary 1]. This invariant helps us to show the *total correctness* of Algorithm 1 and, in turn, Algorithm 2 [9, Corollary 1], provided several well-formedness conditions of the considered synthesis problem. These are, in particular, perforation (1), local controllability by the system (e.g., AAV manoeuvrability), a bound on observational delays (e.g., cycle time of sensor fusion), and a safe initialisation of each modal game. A discussion of these conditions is available in [9, Sec. 3].

Partial correctness of the Algorithms 1, 2, and 3 is verified with respect to two Hoare triples that deliver the guarantees (i), (ii), and (iii) of Definition 1 as post-conditions if the aforementioned well-formedness conditions placed as preconditions hold [9, Lemma 3, Theorems 1 and 2].

Provided the well-formedness conditions, we then verify *termination* of the Algorithms 1, 2, and 3 by establishing the existence of a hybrid trajectory $\bar{\mathbf{s}}$ as an execution fixpoint of the HGA player [9, Theorem 2].

Sound Value-Fixpoint Approximations. In the Algorithms 1 and 2, the fixpoint V^*, required for $\mathbf{u}^*$ to be stationary (i.e., applicable independent of time, $N \approx \infty$, [28]) and optimal, is approximated. This approximation safes time but limits our approach to bounded correctness [9, Corollary 1]. In particular, $\widehat{fp}_{\mathrm{dDP}}$ replaces the ideal but more expensive, non-simultaneous fixpoint check $V(\mathcal{X}_{\mathbf{s}}, k) = V(\mathcal{X}_{\mathbf{s}}, k+1)$ for each k in Line 9 of Algorithm 1. On the one hand, the ideal check might, due to numerical imprecision and instability or a too small N, never be successful. On the other hand, the faster check $\top \notin V(\mathbf{x} \oplus \Delta_\delta, k)$ in Line 7 of Algorithm 2 poorly approximates V^* and would turn our synthesis into plain shortest-path finding. The latter might suffice in some applications.

Approximation increases the deviation of $\mathbf{u}^*$ from optimality. Accepting some deviation, we use $|W(\mathcal{X}_{\mathbf{s}}, k)| = |W(\mathcal{X}_{\mathbf{s}}, k+1)|$ to check in Algorithm 1 whether the number of states stabilises, where a robustly safe and goal-reaching trajectory for at least k steps exists. Then, we perform $\top \notin V(\mathbf{x} \oplus \Delta_\delta, k)$ in Algorithm 2. Although $W(\mathcal{X}_{\mathbf{s}}, k)$ may still evolve while preserving its cardinality, Algorithm 3

then either ensures $\mathbf{x} \in W(\mathcal{X}_\mathbf{s}, k)$ before leaving the solver in Line 8 or it terminates with a failure because of $W(\mathcal{X}_\mathbf{s}, k) = \varnothing$ in Line 11. However, provided the mentioned well-formedness conditions, our proof [9, Theorem 2] guarantees the former and avoids the latter.

Liveness Considerations. Through UPDTASK, Algorithm 3 allows restricted updates of T during operation. To avoid unrealistic moving-target situations, we simplify our setting to a fixed global state space $\mathbb{X}$ and route $\bar{r}$ for the entire operation and to a fixed unsafe region α for each modal play. This simplification provides a conservative assumption for preserving strategic liveness. A proof relying on updates of $\mathcal{O}$ preserving δ-perforation (1) and the weaker assumption of $|\bar{r}| < \infty$ is provided in [9, Theorem 2].

6 Experiments

We implemented the Algorithms 1, 2, and 3 in C++ and evaluated their accuracy and performance in four AAV scenarios (i.e., navigating a small domestic *yard*, a $200{\times}250\,m^2$ *industrial* area, a $400{\times}450\,m^2$ neighbourhood with several *streets*, and a *random* obstacle cloud) covering a range of realistic situations. With our HGA player (Algorithm 3), we illustrate plays of the hybrid game (Fig. 5a) for the *yard* and *industrial* scenarios from [14] as well as the more complicated *streets* scenario. We also highlight some data (Table 1).

Aerial Delivery Game Parameters. For the mentioned scenarios, we use

$$\rho \equiv \mathbf{p} \in \mathcal{X}_{\mathbf{s}^{++}_{q'}}|_\mathbf{p} \qquad \alpha \equiv \mathrm{coll}_\mathsf{f}(\mathcal{O}) \qquad \text{and} \qquad \lambda(\mathbf{u}, \mathbf{d}; \mathbf{s}, k) = \mathbf{x}^2 + \mathbf{u}^2. \tag{7}$$

Informally, ρ enforces the AAV to reach the next waypoint segment $(\mathbf{p}_i, \mathbf{p}_{i+1})$, except for the elevate segment during departure, and α enforces the vehicle to avoid static obstacles. A corresponding collision check (i.e., safety pre-shielding) with $\mathcal{O}$ is performed for each state and pair of control and disturbance inputs in Line 7 of Algorithm 1 and cached for all time steps when α is computed for L. In further experiments not shown here, we worked with a non-linear variant of λ reciprocally weighing-in the distance to the nearest obstacle.

In Algorithm 2, we initialised our settings with a fixed interval I defining N to be the maximum horizon and $\delta_I = 0$ to disable temporal scope extension. For $\Delta_\mathcal{X}$, we employ a fixed small symmetric padding of 2 units across all scenarios. Overall, $\mathbf{U}$ is only used for $q \in \{\mathsf{depart}, \mathsf{cruise}, \mathsf{arrive}\}$ where $\hat{f} = \mathit{fly}$. In Algorithm 3, we use $\Omega \equiv q = \mathsf{standby}$.

Note how Algorithm 3 leaves standby, which fulfils Ω, by immediately performing the globally enabled elevate event. In the AAV example, we keep jumps deterministic. Our implementation would resolve non-deterministic jumps by taking the first available. However, in complex applications, the discrete part of the hybrid game will require an informed policy as well.

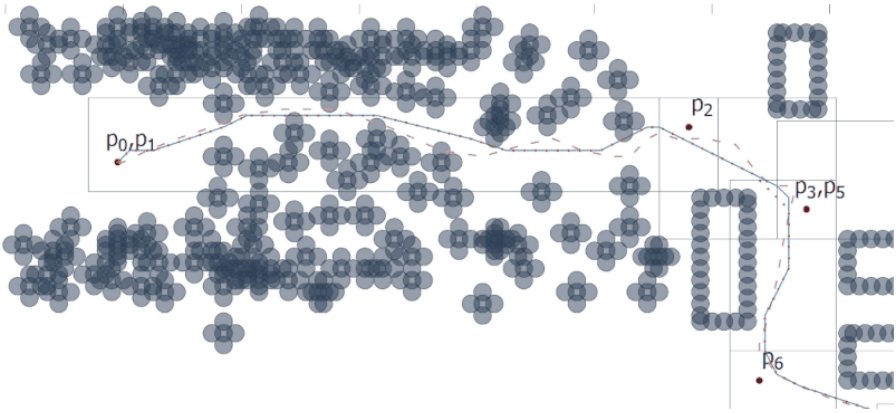


Fig. 7. Play of G with $\bar{r} = \{\mathbf{p}_i\}_{i=0}^{n}$ for the industrial scenario ($n = 13$). The solid reference trajectory results from undisturbed control ($\mathbf{d} = \mathbf{0}$) and the dashed trajectory is the result of applying random disturbance ($\|\mathbf{d}\| \leq \|\mathbf{d}^*\|$)

To focus on the more relevant cases, GENDIST in Line 9 only generates horizontal disturbances. In particular, wind (N, W, S, E) is simulated randomly, following a semi-Markov scheme with finite memory. In the scenarios, wind is more likely to change gradually than spontaneously.

Hybrid Game Plays. For validating our notion of robustness, the Figs. 7 and 8 illustrate the plays of G in the industrial and streets scenarios and Fig. 9 provides a 3D walkthrough of a play in the streets scenario. A play for the yard scenario is shown in Fig. 1. The solid AAV trajectory marks the center of a reference tube (i.e., $\mathbf{d} = \mathbf{0}$) and, for comparison, the dashed trajectory is the result of random disturbance being applied (i.e., randomised $\mathbf{d}$ no worse than $\mathbf{d}^*$). We apply the weighting term from (7).

A transition cuboid around the consecutive waypoints of segment $(\mathbf{p}_{i+1}, \mathbf{p}_{i+2})$ (cf. rectangles in Fig. 8, semi-transparent cuboids in Fig. 9) indicates the goal region ρ of the route tracking task for segment $(\mathbf{p}_i, \mathbf{p}_{i+1})$. ρ circumscribes the next pair of waypoints and, thus, allows edge cases (cf. Figure 1) where certain waypoints (e.g., $\mathbf{p}_5$) are neglected for the benefit of shortcuts (e.g., $(\mathbf{p}_4, \mathbf{p}_6)$).

3D visualisation can provide valuable insights into the vertical scene topology and the trajectory resulting from a play. During model validation and game design, it can help one to identify parameters, such as beneficial placement of waypoints in the context of scope shaping (Fig. 6) or the robustness margin δ. The 3D scenery can increase the confidence in the robustness guarantee provided

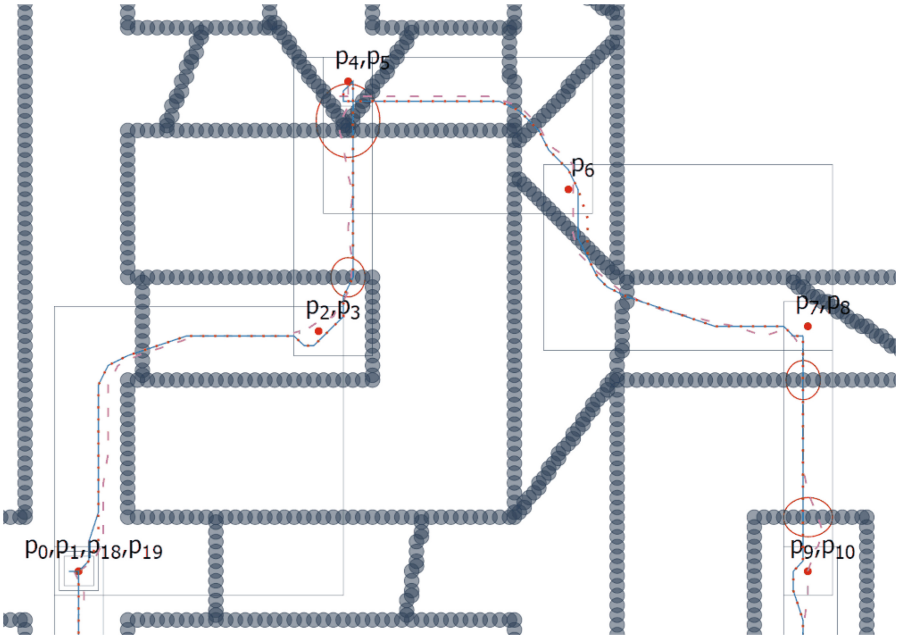


Fig. 8. Play of G with $\bar{r} = \{\mathbf{p}_i\}_{i=0}^{n}$ for the *streets* scenario ($n = 19$)

by the synthesised controllers through a comparison of the trajectories resulting from a worst-case play and a play with random disturbance. Such a validation can be useful for checking and choosing the parameters (e.g., δ) feeding into the well-formedness conditions for algorithmic correctness [9].

Data from the Plays. Table 1 summarises key indicators of the scenarios, such as the total run-time $t(\bar{r})$ of the play for route $\bar{r}$, time $t(\mathbf{p}_i)$ to compute $\mathbf{u}^*$ for segment $(\mathbf{p}_i, \mathbf{p}_{i+1})$, and the maximum number $(\#)$ of states in $\mathcal{X}$. Moreover, the expected savings in peak memory usage of the quasi-stationary controller are summarised in Fig. 10 for comparison.

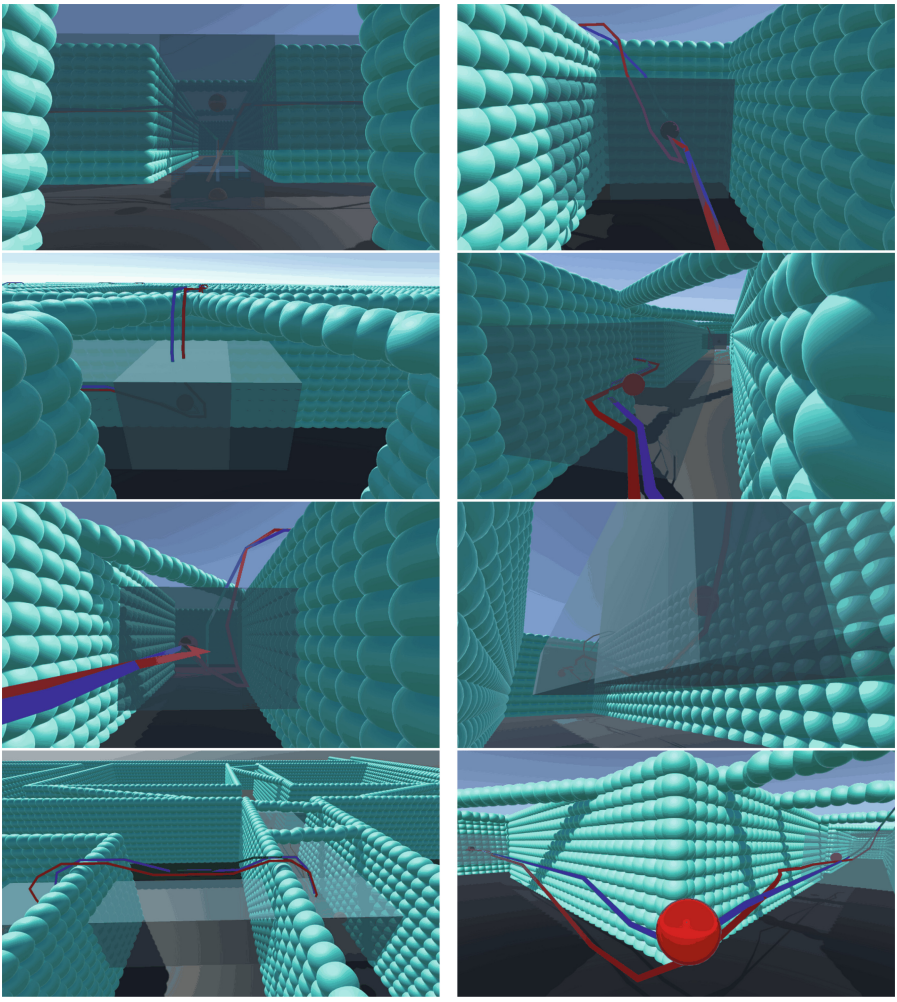


Fig. 9. Snapshots of a robust play in the *streets* scenario. In blue solid lines, the near-optimal reference trajectory enforced by $\mathbf{u}^*$ without disturbance being applied. In red solid lines, a simulation of $\mathbf{u}^*$ with random $\mathbf{d}$ applied. The quasi-stationary reference $\mathbf{u}^\infty$ (green) is hardly visible (bottom left) as it largely overlaps with $\mathbf{u}^*$. Semi-transparent transition cuboids indicate the scopes computed according to Fig. 6 and extended by Algorithm 2 while following the route $\bar{r}$. Recall that successor cuboids $\mathbf{s}_q^{++}|_{\mathbf{p}}$ are used as goal regions

7 Evaluation and Discussion

We assess our approach regarding the near-optimality of the policies, possible computation schemes, an integration with other components, feasible performance improvements, and potential generalisations.

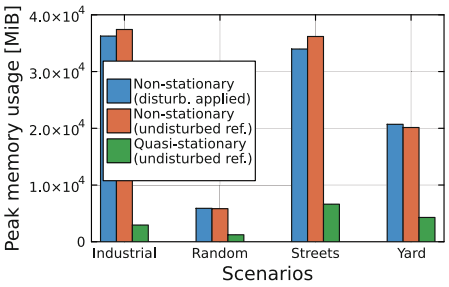


Fig. 10. Peak memory usage of Algorithm 3 by scenario and controller type

Considering Time-Invariant Dynamics. A comparison of the non-stationary $\mathbf{u}^*$, obtained for $[k, N]$ by the termination rule in Line 8 of Algorithm 3, and the quasi-stationary $\mathbf{u}^\infty \colon \mathbb{X} \to \mathcal{U}$, derived via $\mathbf{u}^\infty(\mathbf{x}) = \mathbf{u}^*(\mathbf{x}, k)$, indicates (expected) improvements of $\mathbf{u}^\infty$ (dotted red lines in 2D views, solid green lines in 3D views) over $\mathbf{u}^*$ (solid blue lines) in the undisturbed case (Fig. 11). Moreover, as the environment follows the controller (i.e., $\mathbf{u}^*$ cannot form its current choice based on $\mathbf{d}$'s current choice), applying disturbance (dashed red lines) will have the same effect on $\mathbf{u}^\infty$. Figure 11 (top right) illustrates how $\mathbf{u}^\infty$ deviates from $\mathbf{u}^*$ to get closer to optimality according to (5).

Table 1. δ-robust synthesis compared by metric across the scenarios

Scenario	Play time $t(\bar{r})$ [h:m]	Max. # stages (N)	Max. mem. usage [GiB]	Max. $\|\|X$	Avg. $\|\|X$	Max. t per dDP	Avg. t per 10^6 states	Avg. $\frac{t(\mathbf{p}_0)}{t(\mathbf{p}_{>0})}$
Yard	01:39	30	20.1	105	36.4	25:38	0.597	1.94
Industrial	01:51	60	37.4	108	25.8	47:58	0.798	3.52
Random	01:03	30	5.8	65	25.7	06:54	0.626	1.56
Streets	06:43	50	36.2	172	48.7	51:11	1.110	4.50

Operationalisation of the Hyper-Policy. Lines 9 to 10 of Algorithm 3 represent the environment part of the control loop: Based on an observation (i.e., $\mathbf{x}$ carries an estimate provided by an observer, Fig. 4), the control input and disturbance are applied in parallel, followed by the next observation. Because of this real-time dependency, Lines 7 to 8 need to be pre-computed to be available after jumps.

Ideally, scenario parameters allow enough time to compute $\mathbf{u}$ for the next segment $(\mathbf{p}_i, \mathbf{p}_{i+1})$ based on data from local environmental perception before reaching $\mathbf{p}_i$. Alternatively, there might be time to compute $\mathbf{u}$ for $(\mathbf{p}, \mathbf{p}_i)$ before being required to give up $\mathbf{p}$ as a potential waiting position (e.g., where $\mathbf{u}$ could be a PID controller to robustly hold the AAV at $\mathbf{p}$). However, if the scenario

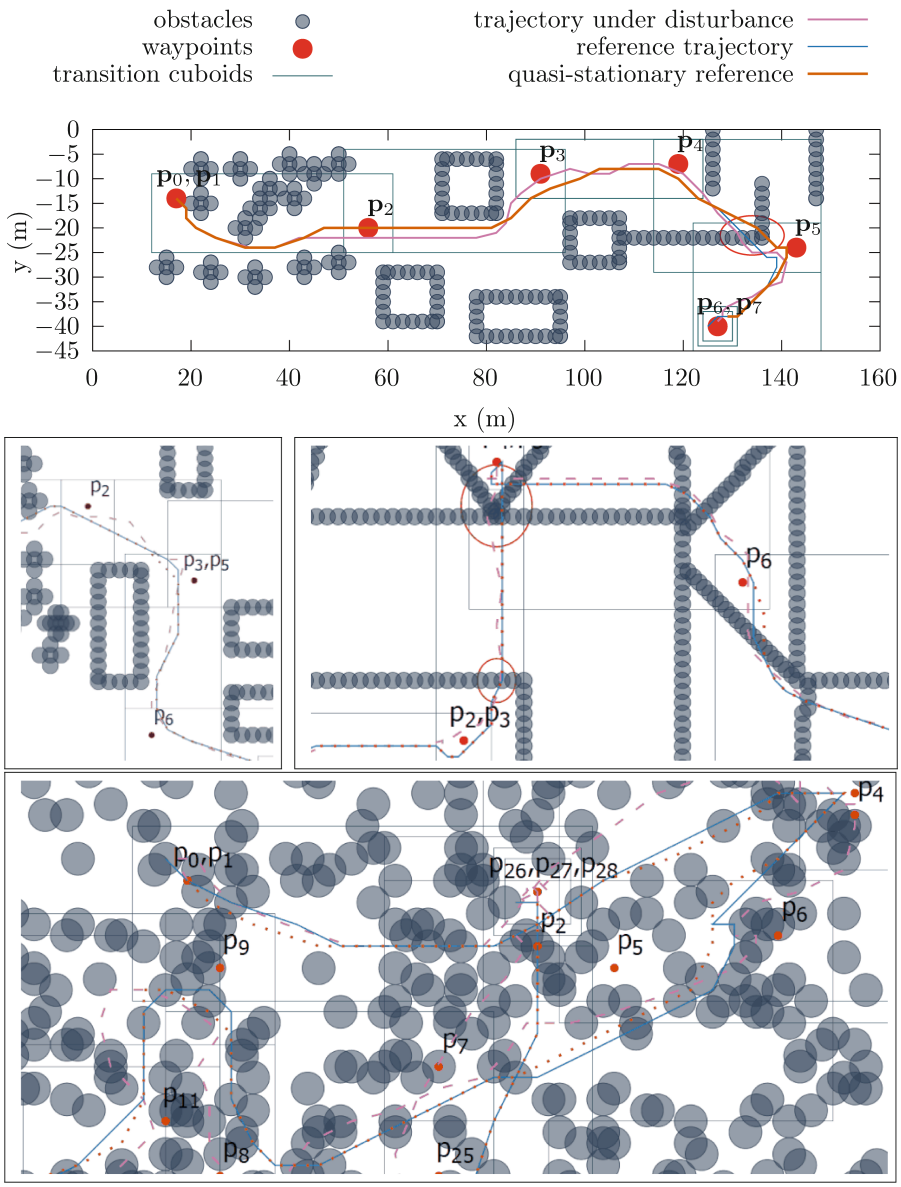


Fig. 11. Partial trajectory triples (non-stationary reference as a solid line, random disturbance applied as a dashed line, quasi-stationary reference as a dotted line) from the yard (top), industrial (middle left), streets (middle right), and random (bottom) scenarios. Indicated with red ellipses, 2D-projected trajectories never touch 3D obstacle structures (Fig. 9). The random scenario (bottom) indicates several differences between non- and quasi-stationary references

at hand does not permit a strong enough approximation, $\mathbf{U}$ will have to be pre-computed (prior to or during flight) for a sufficiently long prefix of $\bar{r}$ given corresponding stability assumptions about $\mathcal{O}$. Pre-computation might be feasible for scopes enclosing segments $(\mathbf{p}_i, \mathbf{p}_{i+1})$ but not for arbitrary states $\mathbf{s}$.

Integration with Dynamic Obstacle Evasion via the Supervisor. A feature of our approach is that recomputation of $\mathbf{u}$ upon return from some potentially complex and lengthy evasion manoeuvre controlled by $\mathbf{u}_e$ (see page 9) is only needed if this manoeuvre results in moving the AAV outside of $\mathcal{X}_\mathbf{s}$. Moreover, $\mathbf{u}_e$ can use $\mathbf{u}$ for circumventing fixed obstacles during evasion manoeuvres.

Performance Improvements. Our AAV example uses a linear approximation and a quadratic weighting term. It would, thus, be amenable to a solution by quadratic programming. However, keeping our approach more widely applicable requires significantly more time and memory (Table 1) than (non-linear) MPC schemes (e.g., [6,21]). Clearly, the C++ prototype is not ready for use in real-time settings. Nevertheless, we believe that selective state discretisation and omission (e.g., interpolating across state-time dimensions, improved parameter settings), decomposition and high-performance parallelisation as well as controller caching (see last paragraph) can reduce average DDP computation-time and space requirements by at least two orders of magnitude. These optimisations would imply losses of optimality and precision but keep much of the exhaustiveness of the scenario coverage. A computation-time reduction would not only increase segment tracking speed and flexibility in route planning but also make our approach faster than some recent safe-RL offline techniques (e.g., [1,11,15,16]).

Generalisations of the Approach. Horizon N, stage k, and controls $\mathcal{U}$ and $\mathcal{D}$ are largely left implicit in J, V, W, $\hat{f}$, and $\mathbf{x}^{ud}$. Also, we employ time resolution 1. However, making some of these parameters explicit leads to non-essential extensions and our approach can be parameterised by N and enhanced to time-varying dynamics and controls $\mathcal{U}_k$ and $\mathcal{D}_k$ as well as non-unit time resolutions. Likewise, more than just 4 wind directions can be used with the corresponding sacrifices in performance. Moreover, our approach can be extended from linear approximation (Algorithm 1 Line 6) to non-linear approximation of more complex (non-point-mass) dynamics and to using non-linear weighting terms.

8 Conclusion

In this work, we specialise a conventional controller synthesis algorithm for a parametric discretised variant of a weighted hybrid game G. We provide a model for the integrated assurance of robust safety, liveness, and near-optimality of controllers for G that are synthesised online, that is, during G's execution.

Our application focuses on the modelling of AAVs, the off- or online synthesis of tactical controllers operating under simplified dynamics, and integrating

these controllers with lower-level actuator and stability control as well as higher-level strategic and supervisory control. Our current approach can be an online-, though not real-time-, capable alternative if a combination of RL with a shielding scheme is not desirable (e.g., lack of tranferrability to other environments, controllability, or explainability of RL's value function approximation).

The abstraction, we chose, combines hybrid games (e.g., step-shielded reach-avoid reasoning), performance optimisation, and numerical aspects (e.g., data sampling, quantisation) of digital control in a way amenable to formal reasoning. This combination enables us to reason about robust safety and liveness guarantees of controllers and how these guarantees impact the assurance of an overall system. In a companion working paper [9], we provide details to the proof results summarised in Sect. 5 (e.g., Algorithm 3 ensures that $\mathbf{x} \in W(\mathcal{X}_\mathbf{s}, k)$; necessary and sufficient conditions for solvability of the weighted hybrid game).

In future work, we will extend our approach to synthesise more complex policies and refine the integration with our work [10] on supervisory control. We will improve the performance of our algorithms as suggested and outlined in Sect. 7 and extend our model to be able to deal with multiple AAVs that can form an intelligent aerial transport collective.

References

1. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. In: AAAI Conf. Artificial Intelligence, vol. 32, pp. 2669–2678. AAAI, Washington, DC (2018). https://doi.org/10.1609/aaai.v32i1.11797, https://ojs.aaai.org/index.php/AAAI/article/view/11797
2. Althoff, M., Dolan, J.M.: Online verification of automated road vehicles using reachability analysis. IEEE Trans. Rob. **30**(4), 903–918 (2014). https://doi.org/10.1109/TRO.2014.2312453
3. Bertram, J., Wei, P., Zambreno, J.: A fast Markov decision process-based algorithm for collision avoidance in urban air mobility. IEEE Trans. Intell. Transp. Syst. **23**(9), 15420–15433 (2022). https://doi.org/10.1109/tits.2022.3140724
4. Bertsekas, D.P.: Dynamic Programming and Optimal Control, vol. 1. Athena Scientific, 4th edn. (2017)
5. Clement, E., Perrin-Gilbert, N., Schlehuber-Caissier, P.: Layered controller synthesis for dynamic multi-agent systems, pp. 50–68. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-42626-1_4
6. Dolgov, D., Thrun, S., Montemerlo, M., Diebel, J.: Path planning for autonomous vehicles in unknown semi-structured environments. Int. J. Rob. Res. **29**(5), 485–501 (2010). https://doi.org/10.1177/0278364909359210
7. Doyen, L., Frehse, G., Pappas, G.J., Platzer, A.: Verification of hybrid systems. In: Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, pp. 1047–1110. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-10575-8_30
8. Gleirscher, M.: Supervision of intelligent systems: an overview. In: Applicable Formal Methods for Safe Industrial Products – Essays Dedicated to Jan Peleska on the Occasion of His 65th Birthday, LNCS, vol. 14165, pp. 202–221. Springer, Cham (2023). https://doi.org/10.1007/978-3-031-40132-9_13

9. Gleirscher, M.: Solvability of approximate reach-avoid games. CoRR (2025). https://doi.org/10.48550/arXiv.2502.04544
10. Gleirscher, M., et al.: Verified synthesis of optimal safety controllers for human-robot collaboration. Sci. Comput. Program. **218**, 102809 (2022). https://doi.org/10.1016/j.scico.2022.102809
11. Gu, R., Jensen, P.G., Poulsen, D.B., Seceleanu, C., Enoiu, E., Lundqvist, K.: Verifiable strategy synthesis for multiple autonomous agents: a scalable approach. Int. J. Softw. Tools Technol. Trans. (2022). https://doi.org/10.1007/s10009-022-00657-z
12. Henzinger, T.A.: The theory of hybrid automata. In: Verification of Digital and Hybrid Systems, NATO ASI Series F: Computer and Systems Sciences, vol. 170, pp. 265–92. Springer, Cham (2000). https://doi.org/10.1007/978-3-642-59615-5_13
13. Heuillet, A., Couthouis, F., Díaz-Rodríguez, N.: Explainability in deep reinforcement learning. Knowledge-Based Syst. **214**, 106685 (2021). https://doi.org/10.1016/j.knosys.2020.106685
14. Hönnecke, P.: Constrained Hybrid Optimal Control of Aerial Transport Systems. Master thesis, under sup. of M. Gleirscher, U Bremen (2024)
15. Ivanov, R., Jothimurugan, K., Hsu, S., Vaidya, S., Alur, R., Bastani, O.: Compositional learning and verification of neural network controllers. ACM Trans. Embed. Comput. Syst. **20**(5s), 1–26 (2021). https://doi.org/10.1145/3477023
16. Jansen, N., Könighofer, B., Junges, S., Serban, A.C., Bloem, R.: Safe reinforcement learning via probabilistic shields. In: Konnov, I., Kovacs, L. (eds.) CONCUR. pp. 31–316. Schloss Dagstuhl - LZI (2020). https://doi.org/10.4230/LIPIcs.CONCUR.2020.3
17. Kanashima, K., Ushio, T.: Finite-horizon shield for path planning ensuring safety/co-safety specifications and security policies. IEEE Access **11**, 11766–11780 (2023). https://doi.org/10.1109/access.2023.3241946
18. Kautz, H.A.: The third AI summer. AI Mag. **43**(1), 105–125 (2022). https://doi.org/10.1002/aaai.12036
19. Könighofer, B., Lorber, F., Jansen, N., Bloem, R.: Shield synthesis for reinforcement learning. In: Margaria, T., Steffen, B. (eds.) ISoLA 2020. LNCS, vol. 12476, pp. 290–306. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-61362-4_16
20. Lewis, F.L., Vrabie, D., Vamvoudakis, K.G.: Reinforcement learning and feedback control. IEEE Control. Syst. **32**(6), 76–105 (2012). https://doi.org/10.1109/mcs.2012.2214134
21. Li, N., Goubault, E., Pautet, L., Putot, S.: A real-time NMPC controller for autonomous vehicle racing. In: ICACR, pp. 148–155. IEEE (2022). https://doi.org/10.1109/icacr55854.2022.9935523
22. Mitsch, S., Ghorbal, K., Vogelbacher, D., Platzer, A.: Formal verification of obstacle avoidance and navigation of ground robots. Int. J. Rob. Res. **36**(12), 1312–1340 (2017). https://doi.org/10.1177/0278364917733549
23. Schnittka, T., Gleirscher, M.: Synthesising robust strategies for robot collectives with recurrent tasks: a case study. In: Luckcuck, M., Xu, M. (eds.) FM Auton. Sys. (FMAS), 6th Workshop. EPTCS, vol. 411, pp. 109–125. OPA (2024). https://doi.org/10.4204/EPTCS.411.7
24. Shalev-Shwartz, S., Shammah, S., Shashua, A.: Safe, multi-agent, reinforcement learning for autonomous driving. Technical report Mobileye (2016)
25. Sutton, R.S., Barto, A.G.: Reinforcement Learning, 2nd edn. MIT Press, Cambridge (2018)

26. Taye, A.G., Valenti, R., Rajhans, A., Mavrommati, A., Mosterman, P.J., Wei, P.: Safe and scalable real-time trajectory planning framework for urban air mobility. J. Aero. Inf. Sys. 1–10 (2024). https://doi.org/10.2514/1.i011381
27. Thumm, J., Althoff, M.: Provably safe deep reinforcement learning for robotic manipulation in human environments. In: ICRA. IEEE (2022). https://doi.org/10.1109/icra46639.2022.9811698
28. Tomlin, C.J., Lygeros, J., Sastry, S.S.: A game theoretic approach to controller design for hybrid systems. Proc. IEEE **88**(7), 949–970 (2000). https://doi.org/10.1109/5.871303

Author Index

B. Steffen (Ed.): AISoLA 2024, LNCS 16032, pp. 281–282, 2026.
https://doi.org/10.1007/978-3-032-01377-4