Formal Methods in System Design
https://doi.org/10.1007/s10703-024-00445-2

Software doping analysis for human oversight

Sebastian Biewer[1] · Kevin Baum[1,2,3] · Sarah Sterz[1] · Holger Hermanns[1] · Sven Hetmank[4] · Markus Langer[5] · Anne Lauber-Rönsberg[4] · Franz Lehr[4]

Received: 22 December 2022 / Accepted: 11 January 2024

Abstract
This article introduces a framework that is meant to assist in mitigating societal risks that software can pose. Concretely, this encompasses facets of software doping as well as unfairness and discrimination in high-risk decision-making systems. The term *software doping* refers to software that contains surreptitiously added functionality that is against the interest of the user. A prominent example of software doping are the tampered emission cleaning systems that were found in millions of cars around the world when the diesel emissions scandal surfaced. The first part of this article combines the formal foundations of software doping analysis with established probabilistic falsification techniques to arrive at a black-box analysis technique for identifying undesired effects of software. We apply this technique to emission cleaning systems in diesel cars but also to high-risk systems that evaluate humans in a possibly unfair or discriminating way. We demonstrate how our approach can assist humans-in-the-loop to make better informed and more responsible decisions. This is to promote effective human oversight, which will be a central requirement enforced by the European Union's upcoming AI Act. We complement our technical contribution with a juridically, philosophically, and psychologically informed perspective on the potential problems caused by such systems.

Keywords Software doping · Artificial intelligence · Algorithmic fairness · Probabilistic falsification · Adequate trust · Human oversight

1 Introduction

Software is the main driver of innovation of our times. Software-defined systems are permeating our communication, perception, and storage technology as well as our personal interactions with technical systems at an unprecedented pace. *"Software-defined everything"* is among the hottest buzzwords in IT today [78, 121].

✉ Sebastian Biewer
biewer@depend.uni-saarland.de

✉ Kevin Baum
kevin.baum@dfki.de

✉ Sarah Sterz
sterz@depend.uni-saarland.de

Extended author information available on the last page of the article

Published online: 04 April 2024

At the same time, we are doomed to trust these systems, despite being unable to inspect or look inside the software we are facing: The owners of the physical hull of 'everything' are typically not the ones owning the software defining 'everything', nor will they have the right to look at what and how 'everything' is defined. This is because commercial software typically is protected by intellectual property rights of the software manufacturer. This prohibits any attempt to disassemble the software or to reconstruct its inner working, albeit it is the very software that is forecasted to be defining 'everything'. The use of machine-learnt software components amplifies the problem considerably by adding opacity of its own kind. Since commercial interests of the software manufacturers seldomly are aligned with the interest of end users, the promise of 'software-defined everything' might well become a dystopia from the perspective of individual digital sovereignty. In this article, we address two of the most pressing incarnations of problematic software behaviour.

Diesel emissions scandal

A massive example of software-defined collective damage is the diesel emissions scandal. Over a period of more than 10 years, millions of diesel-powered cars have been equipped with illegal software that altogether polluted the environment for the sake of commercial advantages of the car manufacturers. At its core, this was made possible by the fact that only a single, precisely defined test setup was put in place for checking conformance with exhaust emissions regulations. This made it a trivial software engineering task to identify the test particularities and to turn off emission cleaning outside these particular conditions. This is an archetypal instance of software doping.

Software doping can be formally characterised as a violation of a *cleanness* property of a program [10, 32]. A detailed and comparative account of meaningful cleanness definitions related to software doping is available [16, Chapter 3]. One cleanness notion that has proven suitable to detect diesel emissions doping is *robust cleanness* [16, 19]. It is based on the assumption that there is some well-defined and agreed standard input/output behaviour of the system which the definition extends to the vicinity around the inputs and outputs close to the standard behaviour. The precise specification of "vicinity" and of "standard behaviour" is assumed to be part of a *contract* between software manufacturer and user. That contract entails the standard behaviour, distance functions for input and output values, and distance thresholds to define the input and output vicinity, respectively. With this, a system behaviour is considered clean, if its output is (or stays) in the output vicinity of the standard, unless the input is (or moves) outside the standard's input vicinity (see Fig. 1).

Example 1 Every car model that is to enter the market in the European Union (and other countries) must be compliant with local regulations. As part of this homologation process, common to all of these regulations is the need for executing a test under precisely defined lab conditions, carried out on a chassis dynamometer. In this, the car has to follow a speed profile, which is called *test cycle* in regulations. At the time when the diesel scandal surfaced, the *New European Driving Cycle* (NEDC) [128] was the single test cycle used in the European Union. It has by now been replaced by the *Worldwide harmonized Light vehicles Test Cycle* (WLTC) [124] in many countries. We refer to previous work for more details [16, 19, 22]. From a perspective of fraud prevention, having only a single test cycle is a major weakness of the homologation procedure. Robust cleanness can overcome this problem. It admits the consideration of driving profiles that stay in the bounded vicinity of one of several standardised test cycle (i.e., NEDC as well as WLTC), while enforcing bounds on the deviations regarding exhaust emission.

Discrimination mitigation
Another set of exemplary scenarios we consider in this article are *high-risk* AI systems, systems empowered by AI technology whose functioning may introduce risks to health, safety, or fundamental rights of human individuals. The European Union is currently developing the *AI Act* [40, 41] that sets out to mitigate many of the risks that such systems pose. Application areas of concern include credit approval ([95]), decisions on visa applications ([84]), admissions to higher education ([27, 133]), screening of individuals in predictive policing ([58]), selection in HR ([92–94]), judicial decisions (as with COMPAS [3, 30, 34, 72]), tenant screening ([115]), and more. In many of these areas, there are legitimate interests and valid reasons for using well-understood AI technology, although the risks associated with their use to date is manifold.

It is widely recognised that discrimination by unfair classification and regression models is one particularly important risk. As a result, a colourful zoo of different operationalisations of unfairness has emerged [96, 131], which should be seen less as a set of competing approaches and more as mutually complementary [52]. At the same time, a consensus is emerging that human oversight is an important piece of the puzzle for mitigating and minimising societal risks of AI [59, 83, 129]. Accordingly, that principle made it into recent drafts of legislation including the European AI Act [40, 41] or certain US state laws [132].

The generic approach we develop for software-doping analysis turns out to be powerful enough to provide automated assistance for human overseers of high-risk AI systems. Apart from spelling out the necessary refocusing, we illustrate the challenge that our work helps to overcome by an exemplary, albeit hypothetical admission system for higher education (inspired by [27, 133]).

Example 2 A large university assigns scores to applicants aiming to enter their computer science PhD program. The scores are computed using an automated, model-based procedure P which is based on three data points: the position of the applicant's last graduate institution in an official, subject-specific ranking, the applicant's most recent grade point average (GPA), and their score in a subject-specific standardised test taken as part of the application procedure. The system then automatically computes a score for the candidate based on an estimation of how successful it expects them to be as students. A dedicated university employee, Unica is in charge of overseeing the individual outcomes of P and is supposed to detect cases where the output of P is or appears flawed. The university pays especial attention to fairness in the scoring procedure, so Unica has to watch out to any signs of potential unfairness. Unica is supposed to desk-reject candidates whose scores are below a certain, predefined threshold—unless she finds problems with P's scoring. Without any additional support, Unica, as human overseer in the loop, must manually check all cases for signs of unfairness as they are processed. This can be a tedious, complicated, and error-prone task and as such constitutes an impediment for the assumed scalability of the automated scoring process to high numbers of applicants. Therefore, she at least requires tool support that assists her in detecting when something is off about the scoring of individual applicants.

This support can be made real by exploiting the technical contributions of this article, in terms of a runtime monitor that provides automated assistance to the human oversight and itself is based on the probabilistic falsification technique we develop. As we will explain, func-cleanness, a variant of cleanness, is a suitable basis for rolling out runtime monitors for such high-risk systems, that are able to detect and flag discrimination or unfair treatment of humans.

The contributions made by this article are threefold.

Detecting software doping using probabilistic falsification. The paper starts off by developing the theory of robust cleanness and func-cleanness. We provide characterisations in the temporal logics HyperSTL and STL, that are then used for an adaptation of existing probabilistic falsification techniques [1, 49]. Altogether, this reduces the problem of software doping detection to the problem of falsifying the logical characterisation of the respective cleanness definition.

Falsification-based test input generation. Recent work [19] proposes a formal framework for robust cleanness testing, with the ambition of making it usable in practice, namely for emissions tests conducted with a real diesel car on a chassis dynamometer. However, that approach leaves open how to perform test input selection in a meaningful manner. The probabilistic falsification technique presented in this article attacks this shortcoming. It supports the testing procedure by guiding it towards test inputs that make the robust cleanness tests likely to fail.

Promoting effective human oversight. We discuss and demonstrate how the technical contributions of this paper contribute to effective human oversight of high-risk systems, as required by the current proposal of the AI act. The hypothetical university admission scenario introduced above will serve as a demonstrator for shedding light on the applicability of our approach as well as the the principles behind it. On a technical level, we provide a runtime monitor for individual fairness based on probabilistic falsification of func-cleanness. On a conceptual level, we consider it important to clarify which duties come with the usage of such a system; from a *legal* perspective, particularly considering the AI Act, substantiated by considering the *ethical* dimension from a philosophical perspective, and from a *psychological* perspective, particularly deliberating on how the overseeing can become *effective*.

This paper is based on a conference publication [17]. Relative to that paper, the development of the theory here is more complete and now includes temporal logic characterisations for func-cleanness. On the conceptual side, this article adds a principled analysis of the applicability of func-cleanness to effective human oversight, spelled out in the setting of admission to higher education. We live up to the societal complexity of this new example and provide an interdisciplinary situation analysis and an interdisciplinary assessment of our proposed solution. Accordingly, although the technical realisation is based on the probabilistic falsification approach outlined in this article, our solution is substantially more thoughtful than a naive instantiation of the falsification framework.

This article is structured as follows. Section 2 provides the preliminaries for the contributions in this article. Section 3 develops the theoretical foundations necessary to use the concept of probabilistic falsification with robust cleanness and func-cleanness. Section 4 demonstrates how the probabilistic falsification approach can be combined with the previously proposed testing approach [19] for robust cleanness, with a focus on tampered emission cleaning systems of diesel cars. Section 5 develops the technical realisation of a fairness monitor based on func-cleanness for high-risk systems. Section 6 evaluates the fairness monitor from the perspective of the disciplines philosophy, psychology, and law. Finally, Sect. 7 summarises the contributions of this article and discusses limitations of our approaches. The appendix of this article contains additional technical details, proofs, and further philosophical and juridical explanations.

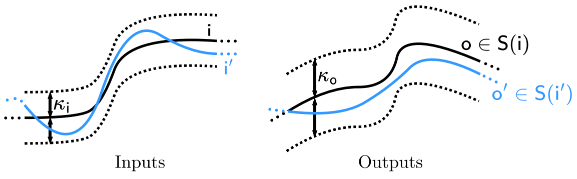

Fig. 1 Robust cleanness intuition

2 Background

2.1 Software doping

After early informal characterisations of *software doping* [10, 13], D'Argenio et al. [32] propose a collection of formal definitions that specify when a software is *clean*. The authors call a software *doped* (w.r.t. a cleanness definition) whenever it does not satisfy such cleanness definition. We focus on *robust cleanness* and *func-cleanness* in this article [32].

We define by $\mathbb{R}_{\geq 0}{:=}\{x \in \mathbb{R} \mid x \geq 0\}$ the set of non-negative real numbers, by $\overline{\mathbb{R}}{:=}\mathbb{R} \cup \{-\infty, \infty\}$ the set of *extended reals* [104], and by $\overline{\mathbb{R}}_{\geq 0}{:=}\mathbb{R}_{\geq 0} \cup \{\infty\}$ the set of the non-negative extended real numbers. We say that a function $d : X \times X \to \overline{\mathbb{R}}_{\geq 0}$ is a *distance function* if and only if it satisfies $d(x, x) = 0$ and $d(x, y) = d(y, x)$ for all $x, y \in X$. We let $\sigma[k]$ denote the kth literal of the finite or infinite word σ.

Reactive Execution Model

We can view a nondeterministic reactive program as a function $\mathsf{S} : \mathsf{In}^{\omega} \to 2^{(\mathsf{Out}^{\omega})}$ perpetually mapping inputs In to sets of outputs Out [32]. To formally model contracts that specify the concrete configuration of robust cleanness or func-cleanness, we denote by $\mathsf{StdIn} \subseteq \mathsf{In}^{\omega}$ the input space of the system designated to define the standard behaviour, and by $d_{\mathsf{In}} : (\mathsf{In} \times \mathsf{In}) \to \overline{\mathbb{R}}_{\geq 0}$ and $d_{\mathsf{Out}} : (\mathsf{Out} \times \mathsf{Out}) \to \overline{\mathbb{R}}_{\geq 0}$ distance functions on inputs, respectively outputs.

For robust cleanness, we additionally consider two constants $\kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \in \overline{\mathbb{R}}_{\geq 0}$. κ_{i} defines the maximum distance that a non-standard input must have to a standard input to be considered in the cleanness evaluation. For this evaluation, κ_{o} defines the maximum distance between two outputs such that they are still considered sufficiently close. Intuitively, the contract defines tubes around standard inputs and their outputs. For example, in Fig. 1, i is a standard input and d_{In} and κ_{i} implicitly define a $2\kappa_{\mathsf{i}}$ wide tube around i. Every input i' that is within this tube will be evaluated on its outputs. Similarly, d_{Out} and κ_{o} define a tube around each of the outputs of i. An output for i' that is within this tube satisfies the robust cleanness condition. Together, the above objects constitute a formal contract $\mathcal{C} = \langle \mathsf{StdIn}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$. Robust cleanness is composed of two separate definitions called l-robust cleanness and u-robust cleanness. Assuming a fixed standard behaviour of a system, l-robust cleanness imposes a lower bound on the non-standard outputs that a system must exhibit, while u-robust cleanness imposes an upper bound. Such lower and upper bound considerations are necessary because of the potential nondeterministic behaviour of the system; for deterministic systems the two notions coincide. We remark that in this article we are using past-forgetful distance functions and the trace integral variants of robust cleanness and func-cleanness (see Biewer [16, Chapter 3] for details).

Definition 1 A nondeterministic reactive program $\mathsf{S} : \mathsf{In}^{\omega} \to 2^{(\mathsf{Out}^{\omega})}$ is *robustly clean* w.r.t. contract $\mathcal{C} = \langle \mathsf{StdIn}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$ if for every standard input $\mathsf{i} \in \mathsf{StdIn}$ and input sequence $\mathsf{i}' \in \mathsf{In}^{\omega}$ it is the case that

1. for every $\mathsf{o} \in \mathsf{S}(\mathsf{i})$, there exists $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$, such that for every index $k \in \mathbb{N}$, if $d_{\mathsf{In}}(\mathsf{i}[j], \mathsf{i}'[j]) \leq \kappa_{\mathsf{i}}$ for all $j \leq k$, then it holds that $d_{\mathsf{Out}}(\mathsf{o}[k], \mathsf{o}'[k]) \leq \kappa_{\mathsf{o}}$, *(l-robust cleanness)*
2. for every $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$, there exists $\mathsf{o} \in \mathsf{S}(\mathsf{i})$, such that for every index $k \in \mathbb{N}$, if $d_{\mathsf{In}}(\mathsf{i}[j], \mathsf{i}'[j]) \leq \kappa_{\mathsf{i}}$ for all $j \leq k$, then it holds that $d_{\mathsf{Out}}(\mathsf{o}[k], \mathsf{o}'[k]) \leq \kappa_{\mathsf{o}}$. *(u-robust cleanness)*

We will in the following refer to Definition 1.1 for l-robust cleanness and Definition 1.2 for u-robust cleanness. Intuitively, l-robust cleanness enforces that whenever an input i' remains within κ_{i} vicinity around the standard input i, then for every standard output $\mathsf{o} \in \mathsf{S}(\mathsf{i})$, there must be a non-standard output $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$ that is in κ_{o} proximity of o. Referring to Fig. 1, every i' in the tube around i must produce for every standard output $\mathsf{o} \in \mathsf{S}(\mathsf{i})$ at least one output $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$ that resides in the κ_{o}-tube around o. In other words, for non-standard inputs the system must not lose behaviour that it can exhibit for a standard input in κ_{i} proximity.

For u-robust cleanness the standard and non-standard output switch roles. It enforces that whenever an input i' remains within κ_{i} vicinity around the standard input i, then for every output $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$ the system can exhibit for this non-standard input, there must be a standard output $\mathsf{o} \in \mathsf{S}(\mathsf{i})$ that is in κ_{o} proximity of o'. Referring to Fig. 1, every i' in the tube around i must only produce outputs $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$ that are in the κ_{o}-tube of at least one $\mathsf{o} \in \mathsf{S}(\mathsf{i})$. In other words, for non-standard inputs within κ_{i} proximity of a standard input the system must not introduce new behaviour, i.e., it must not exhibit an output that is further than κ_{o} away from the set of standard outputs.

A generalisation of robust cleanness is func-cleanness. A cleanness contract for func-cleanness replaces the constants κ_{i} and κ_{o} by a function $f : \overline{\mathbb{R}}_{\geq 0} \to \overline{\mathbb{R}}_{\geq 0}$ inducing a dynamic threshold for output distances based on the distance between the inputs producing such outputs.

Definition 2 A nondeterministic reactive system S is *func-clean* w.r.t. contract $\mathcal{C} = \langle \mathsf{StdIn}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ if for every standard input $\mathsf{i} \in \mathsf{StdIn}$ and input sequence $\mathsf{i}' \in \mathsf{In}^{\omega}$ it is the case that

1. for every $\mathsf{o} \in \mathsf{S}(\mathsf{i})$, there exists $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$, such that for every index $k \in \mathbb{N}$, $d_{\mathsf{Out}}(\mathsf{o}[k], \mathsf{o}'[k]) \leq f(d_{\mathsf{In}}(\mathsf{i}[k], \mathsf{i}'[k]))$, *(l-func-cleanness)*
2. for every $\mathsf{o}' \in \mathsf{S}(\mathsf{i}')$, there exists $\mathsf{o} \in \mathsf{S}(\mathsf{i})$, such that for every index $k \in \mathbb{N}$, $d_{\mathsf{Out}}(\mathsf{o}[k], \mathsf{o}'[k]) \leq f(d_{\mathsf{In}}(\mathsf{i}[k], \mathsf{i}'[k]))$. *(u-func-cleanness)*

We will in the following refer to Definition 2.1 for l-func-cleanness and Definition 2.2 for u-func-cleanness.

For the fairness monitor in Sect. 5 we will use a simpler variant of func-cleanness for deterministic sequential programs. Since P is deterministic, the lower and upper bound requirements coincide, yielding the following simplified definition.

Definition 3 A deterministic sequential program P is *func-clean* w.r.t. contract $\mathcal{C} = \langle \mathsf{StdIn}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ if for every standard input $\mathsf{i} \in \mathsf{StdIn}$ and input $\mathsf{i}' \in \mathsf{In}$, it holds that $d_{\mathsf{Out}}(\mathsf{P}(\mathsf{i}), \mathsf{P}(\mathsf{i}')) \leq f(d_{\mathsf{In}}(\mathsf{i}, \mathsf{i}'))$.

Mixed-IO System Model

The reactive execution model above has the strict requirement that for every input, the system produces exactly one output. Recent work [18, 19] instead considers mixed-IO models, where a program $\mathsf{L} \subseteq (\mathsf{In} \cup \mathsf{Out})^{\omega}$ is a subset of traces containing both inputs and outputs, but without any restriction on the order or frequency in which inputs and outputs appear in the trace. In particular, they are not required to strictly alternate (but they may, and in this way the reactive execution model can be considered a special case [16]). A particularity of this model is the distinct output symbol δ for quiescence, i.e., the absence of an output. For example, finite behaviour can be expressed by adding infinitely many δ symbols to a finite trace.

The new system model induces consequences regarding cleanness contracts. Every mixed-IO trace is projected into an input, respectively output domain. The set of input symbols contains one additional element $-_\mathsf{i}$, that indicates that in the respective steps an output was produced, but masking the concrete output. Similarly, the set of output symbols contains the additional element $-_\mathsf{o}$ to mask a concrete input symbol. *Projection on inputs* $\downarrow_\mathsf{i}\colon (\mathsf{In} \cup \mathsf{Out})^\omega \to (\mathsf{In} \cup \{-_\mathsf{i}\})^\omega$ and *projection on outputs* $\downarrow_\mathsf{o}\colon (\mathsf{In} \cup \mathsf{Out})^\omega \to (\mathsf{Out} \cup \{-_\mathsf{o}\})^\omega$ are defined for all traces $\sigma \in (\mathsf{In} \cup \mathsf{Out})^\omega$ and $k \in \mathbb{N}$ as follows: $\sigma\downarrow_\mathsf{i}[k] := \textbf{if } \sigma[k] \in \mathsf{In} \textbf{ then } \sigma[k] \textbf{ else } -_\mathsf{i}$ and similarly $\sigma\downarrow_\mathsf{o}[k] := \textbf{if } \sigma[k] \in \mathsf{Out} \textbf{ then } \sigma[k] \textbf{ else } -_\mathsf{o}$. The distance functions d_In and d_Out apply on input and output symbols or their respective masks, i.e., they are functions $(\mathsf{In} \cup \{-_\mathsf{i}\}) \times (\mathsf{In} \cup \{-_\mathsf{i}\}) \to \overline{\mathbb{R}}_{\geq 0}$ and, respectively, $(\mathsf{Out} \cup \{-_\mathsf{o}\}) \times (\mathsf{Out} \cup \{-_\mathsf{o}\}) \to \overline{\mathbb{R}}_{\geq 0}$. Finally, instead of a set of standard inputs StdIn, we evaluate mixed-IO system cleanness w.r.t. to a set of standard behaviour $\mathsf{Std} \subseteq \mathsf{L}$. Thus, not only inputs, but also outputs can be defined as standard behaviour and for an input, one of its outputs can be considered in Std while a different output can be excluded from Std. As a consequence, the set Std is specific for some mixed-IO system L, because Std is useful only if $\mathsf{Std} \subseteq \mathsf{L}$. To emphasise this difference we will call the tuple $\mathcal{C} = \langle \mathsf{Std}, d_\mathsf{In}, d_\mathsf{Out}, \kappa_\mathsf{i}, \kappa_\mathsf{o} \rangle$ (cleanness) *context* (instead of cleanness contract). Robust cleanness of mixed-IO systems w.r.t. such a context is defined below [19].

Definition 4 A mixed-IO system $\mathsf{L} \subseteq (\mathsf{In} \cup \mathsf{Out})^\omega$ is *robustly clean* w.r.t. context $\mathcal{C} = \langle \mathsf{Std}, d_\mathsf{In}, d_\mathsf{Out}, \kappa_\mathsf{i}, \kappa_\mathsf{o} \rangle$ if and only if $\mathsf{Std} \subseteq \mathsf{L}$ and for all $\sigma \in \mathsf{Std}$ and $\sigma' \in \mathsf{L}$,

1. there exists $\sigma'' \in \mathsf{L}$ with $\sigma'\downarrow_\mathsf{i} = \sigma''\downarrow_\mathsf{i}$, such that for every index $k \in \mathbb{N}$ it holds that whenever $d_\mathsf{In}(\sigma[j]\downarrow_\mathsf{i}, \sigma'[j]\downarrow_\mathsf{i}) \leq \kappa_\mathsf{i}$ for all $j \leq k$, then $d_\mathsf{Out}(\sigma[k]\downarrow_\mathsf{o}, \sigma''[k]\downarrow_\mathsf{o}) \leq \kappa_\mathsf{o}$, *(l-robust cleanness)*
2. there exists $\sigma'' \in \mathsf{Std}$ with $\sigma\downarrow_\mathsf{i} = \sigma''\downarrow_\mathsf{i}$, such that for every index $k \in \mathbb{N}$ it holds that whenever $d_\mathsf{In}(\sigma[j]\downarrow_\mathsf{i}, \sigma'[j]\downarrow_\mathsf{i}) \leq \kappa_\mathsf{i}$ for all $j \leq k$, then $d_\mathsf{Out}(\sigma'[k]\downarrow_\mathsf{o}, \sigma''[k]\downarrow_\mathsf{o}) \leq \kappa_\mathsf{o}$. *(u-robust cleanness)*

We will in the following refer to Definition 4.1 for l-robust cleanness and Definition 4.2 for u-robust cleanness. Definition 4 universally quantifies a standard trace σ. For l-robust cleanness, the universal quantification of σ' effectively only quantifies an input sequence; the input projection for the existentially quantified σ'' must match the projection for σ'. The remaining parts of the definition are conceptually identical to their reactive systems counterpart in Definition 1.1. For u-robust cleanness, the existentially quantified trace σ'' is obtained from set Std in contrast to l-robust cleanness, where σ'' can be any arbitrary trace of L. This is necessary, because u-robust cleanness is defined w.r.t. a cleanness context; from knowing that $\sigma \in \mathsf{Std}$ is a standard trace and by enforcing that $\sigma\downarrow_\mathsf{i} = \sigma''\downarrow_\mathsf{i}$ we cannot conclude that also $\sigma'' \in \mathsf{Std}$.

Definition 5 shows the definition func-cleanness of mixed-IO systems.

Definition 5 A mixed-IO system $\mathsf{L} \subseteq (\mathsf{In} \cup \mathsf{Out})^\omega$ is *func-clean* w.r.t. context $\mathcal{C} = \langle \mathsf{Std}, d_\mathsf{In}, d_\mathsf{Out}, f \rangle$ if and only if $\mathsf{Std} \subseteq \mathsf{L}$ and for all $\sigma \in \mathsf{Std}$ and $\sigma' \in \mathsf{L}$,

1. there exists $\sigma'' \in \mathsf{L}$ with $\sigma'\downarrow_\mathsf{i} = \sigma''\downarrow_\mathsf{i}$, such that for every index $k \in \mathbb{N}$, it holds that $d_\mathsf{Out}(\sigma[k]\downarrow_\mathsf{o}, \sigma''[k]\downarrow_\mathsf{o}) \leq f(d_\mathsf{In}(\sigma[k]\downarrow_\mathsf{i}, \sigma'[k]\downarrow_\mathsf{i}))$, *(l-func-cleanness)*

2. there exists $\sigma'' \in \mathsf{Std}$ with $\sigma{\downarrow_{\mathsf{i}}} = \sigma''{\downarrow_{\mathsf{i}}}$, such that for every index $k \in \mathbb{N}$, it holds that $d_{\mathsf{Out}}(\sigma'[k]{\downarrow_{\mathsf{o}}}, \sigma''[k]{\downarrow_{\mathsf{o}}}) \leq f(d_{\mathsf{In}}(\sigma[k]{\downarrow_{\mathsf{i}}}, \sigma'[k]{\downarrow_{\mathsf{i}}}))$. *(u-func-cleanness)*

We will in the following refer to Definition 5.1 for l-func-cleanness and Definition 5.2 for u-func-cleanness.

2.2 Temporal logics

2.2.1 HyperLTL

Linear Temporal Logic (LTL) [97] is a popular formalism to reason about properties of traces. A trace is an infinite word where each literal is a subset of AP, the set of atomic propositions. We interpret programs as circuits encoded as sets $\mathsf{C} \subseteq (2^{\mathsf{AP}})^{\omega}$ of such traces. LTL provides expressive means to characterise sets of traces, often called *trace properties*. For some set of traces T, a trace property defines a subset of T (for which the property holds), whereas a *hyperproperty* defines a *set of* subsets of T (constituting combinations of traces for which the property holds). In this way it specifies which traces are valid in combination with one another. Many temporal logics have been extended to corresponding hyperlogics supporting the specification of hyperproperties.

HyperLTL [31] is such a temporal logic for the specification of hyperproperties of reactive systems. It extends LTL with trace quantifiers and trace variables that make it possible to refer to multiple traces within a logical formula. A *HyperLTL formula* is defined by the following grammar, where π is drawn from a set $\mathcal{V}$ of *trace variables* and a from the set AP:

$$\begin{aligned} \psi &::= \exists \pi.\, \psi \mid \forall \pi.\, \psi \mid \phi \\ \phi &::= a_\pi \mid \neg \phi \mid \phi \wedge \phi \mid \mathsf{X}\phi \mid \phi\, \mathcal{U}\, \phi \end{aligned}$$

The quantifiers $\exists$ and $\forall$ quantify existentially and universally, respectively, over the set of traces. For example, the formula $\forall \pi.\, \exists \pi'.\, \phi$ means that for every trace π there exists another trace π' such that ϕ holds over the pair of traces. To account for distinct valuations of atomic propositions across distinct traces, the atomic propositions are indexed with trace variables: for some atomic proposition $a \in \mathsf{AP}$ and some trace variable $\pi \in \mathcal{V}$, a_π states that a holds in the initial position of trace π. The temporal operators and Boolean connectives are interpreted as usual for LTL. Further operators are derivable: $\Diamond \phi \equiv \mathsf{true}\, \mathcal{U}\, \phi$ enforces ϕ to eventually hold in the future, $\Box \phi \equiv \neg \Diamond \neg \phi$ enforces ϕ to always hold, and the weak-until operator $\phi\, \mathcal{W}\, \phi' \equiv \phi\, \mathcal{U}\, \phi' \vee \Box \phi$ allows ϕ to always hold as an alternative to the obligation for ϕ' to eventually hold.

HyperLTL Characterisations of Cleanness

D'Argenio et al. [32] assume distinct sets of atomic propositions to encode inputs and outputs. That is, they assume that $\mathsf{AP} = \mathsf{AP_i} \cup \mathsf{AP_o}$ of atomic propositions, where $\mathsf{AP_i}$ and $\mathsf{AP_o}$ are the atomic propositions that define the the input values and, respectively, the output values. Thus, in the context of Boolean circuit encodings of programs, we take $\mathsf{In} = 2^{\mathsf{AP_i}}$ and $\mathsf{Out} = 2^{\mathsf{AP_o}}$. We capture the following natural correspondence between reactive programs and Boolean circuits; a circuit C can be interpreted as a function $\hat{\mathsf{S}} : \mathsf{In}^{\omega} \to 2^{(\mathsf{Out}^{\omega})}$, where

$$t \in \mathsf{C} \quad \text{if and only if} \quad (t{\downarrow_{\mathsf{AP_o}}}) \in \hat{\mathsf{S}}(t{\downarrow_{\mathsf{AP_i}}}), \tag{1}$$

with $t{\downarrow_A}$ defined by $(t{\downarrow_A})[k] = t[k] \cap A$ for all $k \in \mathbb{N}$.

In the HyperLTL formulas below occur, for convenience, non-atomic propositions. Their semantics is encoded by atomic propositions and Boolean connectives according to a Boolean encoding of inputs and outputs. We refer to the original work for the details [32, Table 1]. Further, we assume that there is a quantifier-free HyperLTL formula StdIn_π that can check whether the trace represented by trace variable π is in the set of standard inputs $\mathsf{StdIn} \subseteq \mathsf{In}^\omega$. That is, StdIn_π should be defined such that for every trace $t \in \mathsf{C}$ it holds that $\{\pi{:=}t\} \models_{\mathsf{C}} \mathsf{StdIn}_\pi$ if and only if $(t{\downarrow_{\mathsf{AP_i}}}) \in \mathsf{StdIn}$.

Proposition 1 shows HyperLTL formulas for l-robust cleanness and u-robust cleanness, respectively.[1]

Proposition 1 *Let* C *be a set of infinite traces over* 2^{AP}*, let* $\hat{\mathsf{S}}$ *be the reactive system constructed from* C *according to Equation 1, and let* $\mathcal{C} = \langle \mathsf{StdIn}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$ *be a contract for robust cleanness. Then* $\hat{\mathsf{S}}$ *is l-robustly clean w.r.t.* $\mathcal{C}$ *if and only if* C *satisfies the HyperLTL formula*

$$\forall \pi_1.\, \forall \pi_2.\, \exists \pi_2'.\, \mathsf{StdIn}_{\pi_1} \rightarrow \Big(\Box(\mathsf{i}_{\pi_2} = \mathsf{i}_{\pi_2'}) \wedge \big((d_{\mathsf{Out}}(\mathsf{o}_{\pi_1}, \mathsf{o}_{\pi_2'}) \le \kappa_{\mathsf{o}})\, \mathcal{W} (d_{\mathsf{In}}(\mathsf{i}_{\pi_1}, \mathsf{i}_{\pi_2'}) > \kappa_{\mathsf{i}}) \big) \Big),$$

and $\hat{\mathsf{S}}$ *is u-robustly clean w.r.t.* $\mathcal{C}$ *if and only if* C *satisfies the HyperLTL formula*

$$\forall \pi_1.\, \forall \pi_2.\, \exists \pi_1'.\, \mathsf{StdIn}_{\pi_1} \rightarrow \Big(\Box(\mathsf{i}_{\pi_1} = \mathsf{i}_{\pi_1'}) \wedge \big((d_{\mathsf{Out}}(\mathsf{o}_{\pi_1'}, \mathsf{o}_{\pi_2}) \le \kappa_{\mathsf{o}})\, \mathcal{W} (d_{\mathsf{In}}(\mathsf{i}_{\pi_1'}, \mathsf{i}_{\pi_2}) > \kappa_{\mathsf{i}}) \big) \Big).$$

The first quantifier (for π_1) in both formulas implicitly quantifies the standard input i and the second quantifier (for π_2) implicitly quantifies the second input i'. Due to the potential nondeterminism in the behaviour of the system, the third, existential, quantifier for π_1', respectively π_2' is necessary. While the formula for l-robust cleanness has the universal quantification on the outputs of the program that takes the standard input i and the existential quantification on the output for i', the formula for u-robust cleanness works in the other way around. Thus, the formulas capture the $\forall\exists$ alternation in Definition 1. The weak until operator $\mathcal{W}$ has exactly the behaviour necessary to represent the interaction between the distances of inputs and the distances of outputs.

The HyperLTL formulas for func-cleanness are given below.

Proposition 2 *Let* C *be a set of infinite traces over* 2^{AP}*, let* $\hat{\mathsf{S}}$ *be the reactive system constructed from* C *according to Equation 1, and let* $\mathcal{C} = \langle \mathsf{StdIn}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ *be a contract for func-cleanness. Then* $\hat{\mathsf{S}}$ *is l-func-clean w.r.t.* $\mathcal{C}$ *if and only if* C *satisfies the HyperLTL formula*

$$\forall \pi_1.\, \forall \pi_2.\, \exists \pi_2'.\, \mathsf{StdIn}_{\pi_1} \rightarrow \Big(\Box(\mathsf{i}_{\pi_2} = \mathsf{i}_{\pi_2'}) \wedge \Box \Big(d_{\mathsf{Out}}(\mathsf{o}_{\pi_1}, \mathsf{o}_{\pi_2'}) \le f(d_{\mathsf{In}}(\mathsf{i}_{\pi_1}, \mathsf{i}_{\pi_2'})) \Big) \Big),$$

and $\hat{\mathsf{S}}$ *is u-func-clean w.r.t.* $\mathcal{C}$ *if and only if* C *satisfies the HyperLTL formula*

$$\forall \pi_1.\, \forall \pi_2.\, \exists \pi_1'.\, \mathsf{StdIn}_{\pi_1} \rightarrow \Big(\Box(\mathsf{i}_{\pi_1} = \mathsf{i}_{\pi_1'}) \wedge \Box \Big(d_{\mathsf{Out}}(\mathsf{o}_{\pi_1'}, \mathsf{o}_{\pi_2}) \le f(d_{\mathsf{In}}(\mathsf{i}_{\pi_1'}, \mathsf{i}_{\pi_2})) \Big) \Big).$$

2.2.2 Signal temporal logic

LTL enables reasoning over traces $\sigma \in (2^{\mathsf{AP}})^\omega$ for which it is necessary to encode values using the atomic propositions in AP. Each literal in a trace represents a discrete time step of an underlying model. Thus, σ can equivalently be viewed as a function $\mathbb{N} \rightarrow 2^{\mathsf{AP}}$. One extension of LTL is *Signal Temporal Logic* (STL) [33, 76], which instead is used for reasoning over

[1] All HyperLTL formulas from D'Argenio et al. [32] are adapted for non-parametrised systems.

real-valued signals that may change in value along an underlying continuous time domain. In this article, we generalise the original work and use *generalised timed traces* (GTTs) [53], which, for some value domain X and time domain $\mathcal{T}$ define traces as functions $\mathcal{T} \to X$. The time domain $\mathcal{T}$ can be either $\mathbb{N}$ (*discrete-time*), or $\mathbb{R}_{\geq 0}$ (*continuous-time*). For the value domain we will use vectors of real values $X = \mathbb{R}^n$ for some $n > 0$ or, to express mixed-IO traces, the set $X = \mathsf{In} \cup \mathsf{Out}$.

STL formulas can express properties of systems modelled as sets $\mathsf{M} \subseteq (\mathcal{T} \to X)$ of traces by making the atomic properties refer to booleanisations of the signal values. The syntax of the variant of STL that we use in this article is as follows, where $f \in X \to \mathbb{R}$:

$$\phi ::= \top \mid f > 0 \mid \neg\phi \mid \phi \wedge \phi \mid \phi\,\mathcal{U}\,\phi\,.$$

STL replaces atomic propositions by *threshold predicates* of the form $f > 0$, which hold if and only if function f applied to the trace value at the current time returns a positive value. The Boolean operators and the Until operator $\mathcal{U}$ are very similar to those of HyperLTL. The Next operator X is not part of STL, because "next" is without precise meaning in continuous time. The definitions of the derived operators $\Diamond$, $\Box$ and $\mathcal{W}$ are the same as for HyperLTL. Formally, the *Boolean semantics* of an STL formula ϕ at time $t \in \mathcal{T}$ for a trace $w \in \mathcal{T} \to X$ is defined inductively:

$$\begin{array}{lll}
w, t \models \top & & \\
w, t \models f > 0 & \text{iff} & f(w(t)) > 0 \\
w, t \models \neg\phi & \text{iff} & w, t \not\models \phi \\
w, t \models \phi \wedge \psi & \text{iff} & w, t \models \phi \text{ and } w, t \models \psi \\
w, t \models \phi\,\mathcal{U}\,\psi & \text{iff} & \text{exists } t' \geq t \text{ s.t. } w, t' \models \psi \text{ and} \\
 & & \qquad \text{for all } t'' \in [t, t'),\ w, t'' \models \phi
\end{array}$$

A system M satisfies a formula ϕ, denoted $\mathsf{M} \models \phi$, if and only if for every $w \in \mathsf{M}$ it holds that $w, 0 \models \phi$.

Quantitative Interpretation

STL has been extended by a *quantitative semantics* [1, 33, 49]. This semantics is designed in such a way that whenever $\rho(\phi, w, t) \neq 0$, its sign indicates whether $w, t \models \phi$ holds in the Boolean semantics. For any STL formula ϕ, trace w and time t, if $\rho(\phi, w, t) > 0$, then $w, t \models \phi$ holds, and if $\rho(\phi, w, t) < 0$, then $w, t \models \phi$ does not hold. The quantitative semantics for an STL formula ϕ, trace w, and time t the quantitative semantics is defined inductively:

$$\begin{array}{rcl}
\rho(\top, w, t) & = & \infty \\
\rho(f > 0, w, t) & = & f(w(t)) \\
\rho(\neg\phi, w, t) & = & -\rho(\phi, w, t) \\
\rho(\phi \wedge \psi, w, t) & = & \min(\rho(\phi, w, t), \rho(\psi, w, t)) \\
\rho(\phi\,\mathcal{U}\,\psi, w, t) & = & \sup\limits_{t' \geq t} \min\{\rho(\psi, w, t'), \inf\limits_{t'' \in [t, t')} \rho(\phi, w, t'')\}
\end{array}$$

Robustness and Falsification

The value of the quantitative semantics can serve as a *robustness estimate* and as such be used to search for a violation of the property at hand, i.e., to falsify it. The robustness of STL formula ϕ is its quantitative value at time 0, that is, $\mathcal{R}_\phi(w) := \rho(\phi, w, 0)$. So, falsifying a formula ϕ for a system M boils down to a search problem with the goal condition $\mathcal{R}_\phi(w) < 0$.

Algorithm 1 Monte-Carlo falsification

```
Input: $w$: Initial trace, $\mathcal{R}$: Robustness function, PS: Proposal Scheme
Output: $w \in \mathsf{M}$
1: while $\mathcal{R}(w) > 0$ do
2:   $w' \leftarrow \mathsf{PS}(w)$
3:   $\alpha \leftarrow \exp(-\beta(\mathcal{R}(w') - \mathcal{R}(w)))$
4:   $r \leftarrow \mathsf{UniformRandomReal}(0, 1)$
5:   if $r \leq \alpha$ then
6:     $w \leftarrow w'$
7:   end if
8: end while
```

Successful falsification algorithms solve this problem by understanding it as the optimisation problem $\mathsf{minimise}_{w \in \mathsf{M}} \mathcal{R}_\phi(w)$. Algorithm 1 [1, 88] sketches an algorithm for Monte-Carlo Markov Chain falsification, which is based on acceptance-rejection sampling [29].

An input to the algorithm is an initial trace w and a computable robustness function $\mathcal{R}$. Robustness computation for STL formulas has been addressed in the literature [33, 49]; we omit this discussion here. The third input PS is a proposal scheme that proposes a new trace to the algorithm based on the previous one (line 2). The parameter β (used in line 3) can be adjusted during the search and is a means to avoid being trapped in local minima, preventing to find a global minimum.

Notably, there exists prior work by Nguyen et al. [89] that discusses an extension of STL to HyperSTL though using a non-standard semantic underpinning. In this context, they present a falsification approach restricted to the fragment "t-HyperSTL" where, according to the authors, "a nesting structure of temporal logic formulas involving different traces is not allowed". Therefore, none of our cleanness definitions belongs to this fragment.

3 Logical characterisation of Mixed-IO cleanness

In this section we provide a temporal logic characterisation for robust cleanness and func-cleanness for mixed-IO systems. For this, we propose a HyperSTL semantics (different to that of [89]) and propose HyperSTL formulas for robust cleanness and func-cleanness. We explain how these formulas can be applied to mixed-IO traces and prove that the characterisation is correct. Furthermore, for the special case that Std is a finite set, we reformulate the HyperSTL formulas characterising the u-cleannesses as equivalent STL formulas.

Hyperlogics over Continuous Domains

Previous work [89] extends STL to HyperSTL echoing the extension of LTL to HyperLTL. We use a similar HyperSTL syntax in this article:

$$\begin{aligned}\psi &::= \exists \pi.\, \psi \mid \forall \pi.\, \psi \mid \phi\\ \phi &::= \top \mid f > 0 \mid \neg\phi \mid \phi \wedge \phi \mid \phi\, \mathcal{U}\, \phi\,.\end{aligned}$$

The meaning of the universal and existential quantifier is as for HyperLTL. In contrast to HyperLTL (and to the existing definition of HyperSTL), we consider it insufficient to allow propositions to refer to only a single trace. In HyperLTL atomic propositions of individual traces can be compared by means of the Boolean connectives. To formulate thresholds for real values, however, we feel the need to allow real values from multiple traces to be combined in the function f, and thus to appear as arguments of f. Hence, in our semantics of HyperSTL, $f > 0$ holds if and only if the result of f, applied to all traces quantified over, is greater than

0. For this to work formally, the arity of function f is the number m of traces quantified over at the occurrence of $f > 0$ in the formula, so $f : X^m \to \mathbb{R}$.

A trace assignment [31] $\Pi : \mathcal{V} \to \mathsf{M}$ is a partial function assigning traces of M to variables. Let $\Pi[\pi:=w]$ denote the same function as Π, except that π is mapped to trace w. The Boolean semantics of HyperSTL is defined below.

Definition 6 Let ψ be a HyperSTL formula, $t \in \mathcal{T}$ a time point, $\mathsf{M} \subseteq (\mathcal{T} \to X)$ a set of GTTs, and Π a trace assignment. Then, the Boolean semantics for $\mathsf{M}, \Pi, t \models \psi$ is defined inductively:

$$\begin{array}{lcl}
\mathsf{M}, \Pi, t \models \exists\pi.\psi & \Leftrightarrow & \exists w \in \mathsf{M}.\ \mathsf{M}, \Pi[\pi:=w], t \models \psi \\
\mathsf{M}, \Pi, t \models \forall\pi.\psi & \Leftrightarrow & \forall w \in \mathsf{M}.\ \mathsf{M}, \Pi[\pi:=w], t \models \psi \\
\mathsf{M}, \Pi, t \models \top & & \\
\mathsf{M}, \Pi, t \models f > 0 & \Leftrightarrow & f(\Pi(\pi_1)(t), \ldots, \Pi(\pi_m)(t)) > 0 \text{ for } dom(\Pi) = \{\pi_1, \ldots, \pi_m\}^2 \\
\mathsf{M}, \Pi, t \models \neg\phi & \Leftrightarrow & \mathsf{M}, \Pi, t \not\models \phi \\
\mathsf{M}, \Pi, t \models \phi_1 \wedge \phi_2 & \Leftrightarrow & \mathsf{M}, \Pi, t \models \phi_1 \text{ and } \mathsf{M}, \Pi, t \models \phi_2 \\
\mathsf{M}, \Pi, t \models \phi_1 \,\mathcal{U}\, \phi_2 & \Leftrightarrow & \exists t' \geq t.\ \mathsf{M}, \Pi, t' \models \phi_2 \text{ and } \forall t'' \in [t, t').\ \mathsf{M}, \Pi, t'' \models \phi_1
\end{array}$$

A system M satisfies a formula ψ if and only if $\mathsf{M}, \varnothing, 0 \models \psi$. The quantitative semantics for HyperSTL is defined below:

Definition 7 Let ψ be a HyperSTL formula, $t \in \mathcal{T}$ a time point, $\mathsf{M} \subseteq (\mathcal{T} \to X)$ a set of GTTs, and Π a trace assignment. Then, the quantitative semantics for $\rho(\psi, \mathsf{M}, \Pi, t)$ is defined inductively:

$$\begin{array}{rcl}
\rho(\exists\pi.\ \psi, \mathsf{M}, \Pi, t) & = & \sup\limits_{w \in \mathsf{M}} \rho(\psi, \mathsf{M}, \Pi[\pi:=w], t) \\
\rho(\forall\pi.\ \psi, \mathsf{M}, \Pi, t) & = & \inf\limits_{w \in \mathsf{M}} \rho(\psi, \mathsf{M}, \Pi[\pi:=w], t) \\
\rho(\top, \mathsf{M}, \Pi, t) & = & \infty \\
\rho(f > 0, \mathsf{M}, \Pi, t) & = & f(\Pi(\pi_1)(t), \ldots, \Pi(\pi_m)(t)) \text{ for } dom(\Pi) = \{\pi_1, \ldots, \pi_m\}^2 \\
\rho(\neg\phi, \mathsf{M}, \Pi, t) & = & -\rho(\phi, \mathsf{M}, \Pi, t) \\
\rho(\phi_1 \wedge \phi_2, \mathsf{M}, \Pi, t) & = & \min(\rho(\phi_1, \mathsf{M}, \Pi, t), \rho(\phi_2, \mathsf{M}, \Pi, t)) \\
\rho(\phi_1 \,\mathcal{U}\, \phi_2, \mathsf{M}, \Pi, t) & = & \sup\limits_{t' \geq t} \min\{\rho(\phi_2, \mathsf{M}, \Pi, t'), \inf\limits_{t'' \in [t, t')} \rho(\phi_1, \mathsf{M}, \Pi, t'')\}
\end{array}$$

HyperSTL Characterisation

The HyperLTL characterisations in Sect. 2.2.1 assume the system to be a subset of $(2^{\mathsf{AP}})^\omega$ and works with distances between traces by means of a Boolean encoding into atomic propositions. By using HyperSTL, we can characterise cleanness for systems that are representable as subsets of $(\mathcal{T} \to X)$.

We can take the HyperLTL formulas from Proposition 1 and 2 and transform them into HyperSTL formulas by applying simple syntactic changes. We get for l-robust cleanness the formula

$$\begin{aligned}
\psi_{\text{l-rob}} := {} & \forall\pi_1.\forall\pi_2.\exists\pi_2'.\,\mathsf{Std}_{\pi_1} > 0 \\
& \to \Big(\Box(\mathsf{eq}(\pi_2{\downarrow_\mathsf{i}}, \pi_2'{\downarrow_\mathsf{i}}) \leq 0) \wedge
\end{aligned}$$

2 We admit some sloppiness; the set $dom(\Pi)$ should have a fixed order.

$$\Big((d_{\mathsf{Out}}(\pi_1{\downarrow_{\mathsf{o}}}, \pi'_2{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\,\mathcal{W}(d_{\mathsf{In}}(\pi_1{\downarrow_{\mathsf{i}}}, \pi'_2{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)\Big)\Big), \tag{2}$$

u-robust cleanness is characterised by

$$\begin{aligned}\psi_{\text{u-rob}} := \forall\pi_1.\,\forall\pi_2.\,\exists\pi'_1.\,\mathsf{Std}_{\pi_1} &> 0\\ \rightarrow \Big(\mathsf{Std}_{\pi'_1} &> 0 \wedge \square(\mathsf{eq}(\pi_1{\downarrow_{\mathsf{i}}}, \pi'_1{\downarrow_{\mathsf{i}}}) \leq 0) \wedge\\ &\Big((d_{\mathsf{Out}}(\pi'_1{\downarrow_{\mathsf{o}}}, \pi_2{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\,\mathcal{W}(d_{\mathsf{In}}(\pi'_1{\downarrow_{\mathsf{i}}}, \pi_2{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)\Big)\Big),\end{aligned} \tag{3}$$

for l-func-cleanness we get the formula

$$\begin{aligned}\psi_{\text{l-fun}} := \forall\pi_1.\,\forall\pi_2.\,\exists\pi'_2.\,\mathsf{Std}_{\pi_1} &> 0\\ \rightarrow \Big(\square(\mathsf{eq}(\pi_2{\downarrow_{\mathsf{i}}}, \pi'_2{\downarrow_{\mathsf{i}}}) \leq 0) &\wedge \big(\square(d_{\mathsf{Out}}(\pi_1{\downarrow_{\mathsf{o}}}, \pi'_2{\downarrow_{\mathsf{o}}}) - f(d_{\mathsf{In}}(\pi_1{\downarrow_{\mathsf{i}}}, \pi'_2{\downarrow_{\mathsf{i}}})) \leq 0)\big)\Big),\end{aligned} \tag{4}$$

and, finally, u-func-cleanness is encoded by

$$\begin{aligned}\psi_{\text{u-fun}} := \forall\pi_1.\,\forall\pi_2.\,\exists\pi'_1.\,\mathsf{Std}_{\pi_1} &> 0\\ \rightarrow \Big(\mathsf{Std}_{\pi'_1} &> 0 \wedge \square(\mathsf{eq}(\pi_1{\downarrow_{\mathsf{i}}}, \pi'_1{\downarrow_{\mathsf{i}}}) \leq 0) \wedge\\ &\big(\square(d_{\mathsf{Out}}(\pi'_1{\downarrow_{\mathsf{o}}}, \pi_2{\downarrow_{\mathsf{o}}}) - f(d_{\mathsf{In}}(\pi'_1{\downarrow_{\mathsf{i}}}, \pi_2{\downarrow_{\mathsf{i}}})) \leq 0)\big)\Big).\end{aligned} \tag{5}$$

The quantifiers remain unchanged relative to the formulas in Propositions 1 and 2. The formulas use generic projection functions $\downarrow_{\mathsf{i}} : X \rightarrow \mathsf{In}$ and $\downarrow_{\mathsf{o}} : X \rightarrow \mathsf{Out}$ to extract the input values, respectively output values from a trace. To apply the formulas, these functions must be instantiated with functions for the concrete instantiation of the value domain X of the traces to be analysed. For example, for $\mathsf{In} = \mathbb{R}^m$, $\mathsf{Out} = \mathbb{R}^l$, and $\mathsf{M} \subseteq (\mathcal{T} \rightarrow \mathbb{R}^{m+l})$, the projections could be defined for every $w = (s_1, \dots, s_m, s_{m+1}, \dots, s_{m+l})$ as $w{\downarrow_{\mathsf{i}}} = (s_1, \dots, s_m)$ and $w{\downarrow_{\mathsf{o}}} = (s_{m+1}, \dots, s_{m+l})$. The input equality requirement for two traces π and π' is ensured by globally enforcing $\mathsf{eq}(\pi{\downarrow_{\mathsf{i}}}, \pi'{\downarrow_{\mathsf{i}}}) \leq 0$. eq is a generic function that returns zero if its arguments are identical and a positive value otherwise. It must be instantiated for concrete value domains. For example, $\mathsf{eq}((s_1, \dots, s_m), (s'_1, \dots, s'_m))$ could be defined as the sum of the component-wise distances $\sum_{1 \leq i \leq m} |s_i - s'_i|$. Finally, in the above formulas we perform simple arithmetic operations to match the syntactic requirements of HyperSTL.

Formulas (3) and (5) are prepared to express u-robust cleanness, respectively u-func-cleanness w.r.t. both cleanness *contracts* or cleanness *contexts*. That is, we assume the existence of a function Std_π that returns a positive value if and only if the trace assigned to π encodes a standard input (when considering cleanness *contracts*) or encodes an input and output that constitute a standard behaviour (when considering cleanness *contexts*). Explicitly requiring that π'_1 represents a standard behaviour echoes the setup in Definitions 4.2 and 5.2.

We remark that for encoding Std_π, due to the absence of the Next-operator in HyperSTL, it might be necessary to add a clock signal $s(t) = t$ to traces in a preprocessing step.

Example 3 Let $\mathsf{In} = \mathsf{Out} = \mathbb{R}$ be the sets representing real-valued inputs and outputs, $\mathcal{T} = \mathbb{N}$ be the discrete time domain, and $X = \mathsf{In} \times \mathsf{Out}$ the value domain that considers pairs of inputs and outputs as values. We consider the robust cleanness context $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$, where $\mathsf{Std} = \{w_0, w_1\}$ contains the two standard traces

$w_0 = (1; 0)\,(2; 0)\,(3; 0)\,(4; 0) \cdots$ and $w_1 = (1; 1)\,(2; 2)\,(3; 3)\,(4; 4) \cdots$.

For the distance functions we use the absolute differences, i.e., $d_{\mathsf{In}}(v_1, v_2) = d_{\mathsf{Out}}(v_1, v_2) = |v_1 - v_2|$. Let the value thresholds be $\kappa_{\mathsf{i}} = 1$ and $\kappa_{\mathsf{o}} = 2$, and let $\downarrow_{\mathsf{i}}, \downarrow_{\mathsf{o}}, \mathsf{eq}$ and Std_π be defined as explained above. We consider the non-standard traces $w_A = (1.3; 0)\,(2.6; 0)\,(3.9; 0)\,(5.2; 0) \cdots$, $w_B = (1.3; 1.3)\,(2.6; 2.6)\,(3.9; 3.9)\,(5.2; 5.2) \cdots$, and $w_{\times} = (1.5; 1.5)\,(2.5; 3.2)\,(3.5; 4.9)\,(4.5; 6.6) \cdots$.

The HyperSTL formulas $\psi_{\text{l-rob}}$ and $\psi_{\text{u-rob}}$ reason about sets of traces. For example, the set $\mathsf{M} = \{w_0, w_1, w_A, w_B\}$ satisfies both formulas. If both π_1 and π_2 represent standard traces, then $\pi_1{\downarrow_\mathsf{i}} = \pi_2{\downarrow_\mathsf{i}}$, because $w_0{\downarrow_\mathsf{i}} = w_1{\downarrow_\mathsf{i}}$, and the formulas hold for $\pi_2' = \pi_1$, respectively $\pi_1' = \pi_2$. Otherwise, assume that π_1 represents w_0 and π_2 represents w_B (the reasoning for other combinations of traces is similar).

First considering $\psi_{\text{l-rob}}$, we pick w_A for π_2'. We get that $\pi_2{\downarrow_\mathsf{i}} = \pi_2'{\downarrow_\mathsf{i}}$, because $w_B{\downarrow_\mathsf{i}} = w_A{\downarrow_\mathsf{i}}$. Hence, we globally have $|\pi_2{\downarrow_\mathsf{i}} - \pi_2'{\downarrow_\mathsf{i}}| = 0$ and, thus, $\mathsf{eq}(\pi_2{\downarrow_\mathsf{i}}, \pi_2'{\downarrow_\mathsf{i}}) = 0$. At time steps $0 \leq t \leq 3$, the distance between the outputs $|w_0{\downarrow_\mathsf{o}}(t) - w_A{\downarrow_\mathsf{o}}(t)|$ is at most κ_o. Hence, the left operand of $\mathcal{W}$ holds and the formula is satisfied for $t \leq 3$. At time $t = 3$ we have that $|w_0{\downarrow_\mathsf{i}}(t) - w_A{\downarrow_\mathsf{i}}(t)| = |4.0 - 5.2| > \kappa_\mathsf{i}$. Hence, the right operand of the $\mathcal{W}$ operator holds and $\psi_{\text{l-rob}}$ is satisfied also for $t \geq 3$. Notice that if we would remove w_A from M, then it would violate $\psi_{\text{l-rob}}$, because there is no possible choice for π_2' that has the same inputs as w_B and where the output distances to w_0 are below the κ_o threshold.

To satisfy $\psi_{\text{u-rob}}$, we pick w_1 for π_1'. The reasoning why the formula holds for this choice is analogue to $\psi_{\text{l-rob}}$. Notice that if we add the trace $w_{\boldsymbol{\times}}$ to M, then $\psi_{\text{u-rob}}$ is violated. Concretely, π_2 could represent $w_{\boldsymbol{\times}}$; then, whether we pick w_0 or w_1 for π_1', we eventually get outputs that violate the κ_o constraint, while the κ_i constraint is always satisfied. For example, if we compare $w_{\boldsymbol{\times}}$ and w_1, then we have for all time steps $t \leq 3$ that $|w_1{\downarrow_\mathsf{i}}(t) - w_{\boldsymbol{\times}}{\downarrow_\mathsf{i}}(t)| = 0.5 \leq \kappa_\mathsf{i}$, but at time $t = 3$ we get $|w_1{\downarrow_\mathsf{o}}(t) - w_{\boldsymbol{\times}}{\downarrow_\mathsf{o}}(t)| = 2.6 > \kappa_\mathsf{o}$. Hence, at $t = 3$ the left and right operand of $\mathcal{W}$ are false, so $\psi_{\text{u-rob}}$ is violated.

Correctness under Mixed-IO Interpretation

Mixed-IO signals are defined in the discrete time domain $\mathbb{N}$ and value domain $\mathsf{In} \cup \mathsf{Out}$. The abstract functions $\downarrow_\mathsf{i}$ and $\downarrow_\mathsf{o}$ can be defined equally to the syntactically identical projection functions for mixed-IO models defined in Sect. 2.1. The function $\mathsf{eq}(\mathsf{i}_1, \mathsf{i}_2)$ can be defined using the distance function d_In and some arbitrary small $\varepsilon > 0$:

$$\mathsf{eq}(\mathsf{i}_1, \mathsf{i}_2) := \begin{cases} 0, & \text{if } \mathsf{i}_1 = \mathsf{i}_2 \\ d_\mathsf{In}(\mathsf{i}_1, \mathsf{i}_2) + \varepsilon, & \text{if } \mathsf{i}_1 \neq \mathsf{i}_2 \wedge \mathsf{i}_1, \mathsf{i}_2 \in \mathsf{In} \\ \infty, & \text{otherwise.} \end{cases} \tag{6}$$

In the second clause of the above definition we add some positive value ε to the result of d_In, because $d_\mathsf{In}(\mathsf{i}_1, \mathsf{i}_2)$ could be 0 even if $\mathsf{i}_1 \neq \mathsf{i}_2$. For the correctness of the above HyperSTL formulas, however, it is crucial that $\mathsf{eq}(\mathsf{i}_1, \mathsf{i}_2) = 0$ if and only if $\mathsf{i}_1 = \mathsf{i}_2$. For a good performance of the falsification algorithm, we will nevertheless want to make use of d_In if $\mathsf{i}_1 \neq \mathsf{i}_2$.

Proposition 3 shows that HyperSTL formulas (2) and (3) under the mixed-IO interpretation outlined above indeed characterise l-robust cleanness and u-robust cleanness. Proposition 4 shows the same for func-cleanness.

Proposition 3 *Let* $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ *be a mixed-IO system and* $\mathcal{C} = \langle \mathsf{Std}, d_\mathsf{In}, d_\mathsf{Out}, \kappa_\mathsf{i}, \kappa_\mathsf{o} \rangle$ *a contract or context for robust cleanness with* $\mathsf{Std} \subseteq \mathsf{L}$. *Further, let* Std_π *be a quantifier-free HyperSTL subformula, such that* $\mathsf{L}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_\pi$ *if and only if* $w \in \mathsf{Std}$. *Then,* L *is l-robustly clean w.r.t.* $\mathcal{C}$ *if and only if* $\mathsf{L}, \varnothing, 0 \models \psi_{\textit{l-rob}}$, *and* L *is u-robustly clean w.r.t.* $\mathcal{C}$ *if and only if* $\mathsf{L}, \varnothing, 0 \models \psi_{\textit{u-rob}}$.

Proposition 4 *Let* $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ *be a mixed-IO system and* $\mathcal{C} = \langle \mathsf{Std}, d_\mathsf{In}, d_\mathsf{Out}, f \rangle$ *a contract or context for func-cleanness with* $\mathsf{Std} \subseteq \mathsf{L}$. *Further, let* Std_π *be a quantifier-free HyperSTL subformula, such that* $\mathsf{L}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_\pi$ *if and only if* $w \in \mathsf{Std}$. *Then,* L *is l-func-clean w.r.t.* $\mathcal{C}$ *if and only if* $\mathsf{L}, \varnothing, 0 \models \psi_{\textit{l-fun}}$, *and* L *is u-func-clean w.r.t.* $\mathcal{C}$ *if and only if* $\mathsf{L}, \varnothing, 0 \models \psi_{\textit{u-fun}}$.

STL Characterisation for Finite Standard Behaviour

In many practical settings—when the different standard behaviours are spelled out upfront explicitly, as in NEDC and WLTC—it can be assumed that the number of distinct standard behaviours Std is finite (while there are infinitely many possible behaviours in M). Finiteness of Std makes it possible to remove by enumeration the quantifiers from the u-robust cleanness and u-func-cleanness HyperSTL formulas. This opens the way to work with the STL fragment of HyperSTL, after proper adjustments. In the following, we assume that the set $\mathsf{Std} = \{w_1, \dots, w_c\}$ is an arbitrary standard set with c unique standard traces, where every $w_k : \mathcal{T} \to X$ uses the same time domain $\mathcal{T}$ and value domain X.

To encode the HyperSTL formulas (3) and (5) in STL, we use the concept of *self-composition*, which has proven useful for the analysis of hyperproperties [9, 51]. We concatenate a trace under analysis $w : \mathcal{T} \to X$ and the standard traces w_1 to w_c to the composed trace $w_+ = (w, w_1, \dots, w_c) \subseteq (\mathcal{T} \to X^{c+1})$. Given a system $\mathsf{M} \subseteq (\mathcal{T} \to X)$ and a set $\mathsf{Std} = \{w_1, \dots, w_c\} \subseteq \mathsf{M}$, we denote by $\mathsf{M} \circ \mathsf{Std} := \{(w, w_1, \dots, w_c) \mid w \in \mathsf{M}\}$ the system in which every trace in M is composed with the standard traces in Std. For every $w_+ \in \mathsf{M} \circ \mathsf{Std}$, we will in the following STL formula write w to mean the projection on w_+ to the trace w, and we write w_k, for $1 \le k \le c$, to mean the projection on w_+ to the kth standard trace.

Theorem 5 *Let* $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ *be a mixed-IO system and* $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$ *a context for robust cleanness with finite standard behaviour* $\mathsf{Std} = \{w_1, \dots, w_c\} \subseteq \mathsf{L}$. *Then,* L *is u-robustly clean w.r.t.* $\mathcal{C}$ *if and only if* $(\mathsf{L} \circ \mathsf{Std}) \models \varphi_{\textit{u-rob}}$, *where*

$$\varphi_{\textit{u-rob}} := \bigwedge_{1 \le a \le c} \bigvee_{1 \le b \le c} \Big(\Box(\mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \le 0) \wedge \\ \big((d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \le 0)\, \mathcal{W} (d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)\big)\Big).$$

The theorem for u-func-cleanness is analogue to Theorem 5.

Theorem 6 *Let* $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ *be a mixed-IO system and* $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ *a context for func-cleanness with finite standard behaviour* $\mathsf{Std} = \{w_1, \dots, w_c\} \subseteq \mathsf{L}$. *Then,* L *is u-func-clean w.r.t.* $\mathcal{C}$ *if and only if* $(\mathsf{L} \circ \mathsf{Std}) \models \varphi_{\textit{u-fun}}$, *where*

$$\varphi_{\textit{u-fun}} := \bigwedge_{1 \le a \le c} \bigvee_{1 \le b \le c} \Big(\Box(\mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \le 0) \wedge \\ \big(\Box(d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - f(d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}})) \le 0)\big)\Big).$$

Example 4 We consider the robust cleanness context $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$ where $\mathsf{Std} = \{w_1, w_2\}$ contains the two standard traces $w_1 = 1_{\mathsf{i}}\, 2_{\mathsf{i}}\, 3_{\mathsf{i}}\, 7_{\mathsf{o}}\, 0_{\mathsf{i}}\, \delta^{\omega}$ and $w_2 = 0_{\mathsf{i}}\, 1_{\mathsf{i}}\, 2_{\mathsf{i}}\, 3_{\mathsf{i}}\, 6_{\mathsf{o}}\, \delta^{\omega}$. We here decorate inputs with index i and outputs with index o, i.e., w_1 describes a system receiving the three inputs 1, 2, and 3, then producing the output 7, and finally receiving input 0 before entering quiescence. We take

$$d_{\mathsf{In}}(\mathsf{i}_1, \mathsf{i}_2) = \begin{cases} |\mathsf{i}_1 - \mathsf{i}_2|, & \text{if } \mathsf{i}_1, \mathsf{i}_2 \in \mathsf{In} \\ 0, & \text{if } \mathsf{i}_1 = \mathsf{i}_2 = -_{\mathsf{i}} \\ \infty, & \text{otherwise,} \end{cases}$$

and

$$d_{\mathsf{Out}}(\mathsf{o}_1, \mathsf{o}_2) = \begin{cases} |\mathsf{o}_1 - \mathsf{o}_2|, & \text{if } \mathsf{o}_1, \mathsf{o}_2 \in \mathsf{Out}\backslash\{\delta\} \\ 0, & \text{if } \mathsf{o}_1 = \mathsf{o}_2 = \text{–}_{\mathsf{o}} \text{ or } \mathsf{o}_1 = \mathsf{o}_2 = \delta \\ \infty, & \text{otherwise.} \end{cases}$$

The contractual value thresholds are assumed to be $\kappa_{\mathsf{i}} = 1$ and $\kappa_{\mathsf{o}} = 6$.

Assume we are observing the trace $w = 0_{\mathsf{i}}\, 1_{\mathsf{i}}\, 2_{\mathsf{i}}\, 6_{\mathsf{o}}\, 0_{\mathsf{i}}\, \delta^{\omega}$ to be monitored with STL formula $\varphi_{\text{u-rob}}$ (from Theorem 5). First notice, that for combinations of a and b in $\varphi_{\text{u-rob}}$, where $a \neq b$, the subformula $\square(\mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \leq 0)$ is always false, because w_1 and w_2 have different (input) values at time point 0. Hence, it remains to show that

$$(d_{\mathsf{Out}}(w_1{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\, \mathcal{W}(d_{\mathsf{In}}(w_1{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0) \wedge$$
$$(d_{\mathsf{Out}}(w_2{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\, \mathcal{W}(d_{\mathsf{In}}(w_2{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0).$$

For the first conjunct, the input distance between inputs in w and w_1 is always 1 at positions 1 to 3, it is 0 at position 4 (because –_{i} is compared to –_{i}), and remains 0 in position 5 and beyond. Thus, $d_{\mathsf{In}}(w_1{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}}$ is always at most 0, and the right hand-side of the $\mathcal{W}$ operator is always false. Consequently, by definition of $\mathcal{W}$, the left operand of $\mathcal{W}$ must always hold, i.e., $d_{\mathsf{Out}}(w_1{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}})$ must always be less or equal to 6. This is the case for w_1 and w: at all positions except for 4, –_{o} is compared to –_{o} (or δ to δ), so the difference is 0, and at position 4, the distance of 6 and 7 is 1.

For the second $\mathcal{W}$-formula, w is compared to w_2. These two traces are comparable only to a limited extent: the order of input and output is altered at the last two positions of the signals before quiescence. Hence, the right operand of $\mathcal{W}$ is true at position 4, and the formula holds for the remaining trace. For positions 1 to 3, the input distances are 0, because the input values are identical. At these positions, the left operand must hold. The values are input values, so –_{o} is compared to –_{o} at each position. This distance is defined to be 0, so it holds that $-6 \leq 0$, and the formula is satisfied. Since both formulas hold, the conjunction of both holds, too, and trace w is qualified as robustly clean. There could however be other system traces not considered in this example, that overall could violate robust cleanness of the system.

Restriction of input space

Robust cleanness puts semantic requirements on fragments of a system's input space, outside of which the system's behaviour remains unspecified. Typically, the fragment of the input space covered is rather small. To falsify the STL formula $\varphi_{\text{u-rob}}$ from Theorem 5, the falsifier has two challenging tasks. First, it has to find a way to stay in the relevant input space, i.e., select inputs with a distance of at most κ_{i} from the standard behaviour. Only if this is assured it can search for an output large enough to violate the κ_{o} requirement. In this, a large robustness estimate provided by the quantitative semantics of STL cannot serve as an indicator for deciding whether an input is too far off or whether an output stays too close to the standard behaviour. We can improve the efficiency of the falsification process significantly by narrowing upfront the input space the falsifier uses.

In practice, test execution traces will always be finite. In previous real-life doping tests, test execution lengths have been bounded by some constant $B \in \mathbb{N}$ [19], i.e., systems are represented as sets of finite traces $\mathsf{M} \subseteq (\mathsf{In} \cup \mathsf{Out})^B$ (which for formality reasons each can be considered suffixed with δ^{ω}). In this bounded horizon, we can provide a predicate discriminating between relevant and irrelevant input sequences. Formally, the restriction to the relevant input space fragment of a system $\mathsf{M} \subseteq (\mathsf{In} \cup \mathsf{Out})^B$ is given by the set $\mathsf{In}_{\mathsf{Std},\kappa_{\mathsf{i}}} = \{w \in \mathsf{M} \mid \exists w' \in \mathsf{Std}.\ \bigwedge_{k=0}^{B-1}(d_{\mathsf{In}}(w[k]{\downarrow_{\mathsf{i}}}, w'[k]{\downarrow_{\mathsf{i}}}) \leq \kappa_{\mathsf{i}})\}$. Since Std and B are finite, membership is computable.

There are rare cases in which this optimisation may prevent the falsifier from finding a counterexample. This is only the case if there is an input prefix leading to a violation of the formula for which there is no suffix such that the whole trace satisfies the κ_i constraint. Below is a pathological example in which this could make a difference.

Example 5 Apart from NO_x emissions, NEDC (and WLTC) tests are used to measure fuel consumption. Consider a contract similar to the contracts above, but with fuel rate as the output quantity. Assuming a "normal" fuel rate behaviour during the standard test, there might be a test within a reasonable κ_i distance, where the fuel is wasted insanely. Then, the fuel tank might run empty before the intended end of the test, which therefore could not be finished within the κ_i distance, because speed would be constantly 0 at the end. The actually driven test is not in set $\mathsf{In}_{\mathsf{Std},\kappa_\mathsf{i}}$, but there is a prefix within κ_i distance that violates the robust cleanness property.

Notably, there may be additional techniques to reduce the size of the input space. For example, if the next input symbol depends on the history of inputs, this constraint could be considered in the proposal scheme.

4 Supervision of diesel emission cleaning systems

The severity of the diesel emissions scandal showed that the regulations alone are insufficient to prevent car manufacturers from implementing tampered—or doped—emission cleaning systems. Recent works [19] shows that robust cleanness is a suitable means to extend the precisely defined behaviour of cars for the NEDC to test cycles within a κ_i range around the NEDC. To demonstrate the usefulness of robust cleanness, the essential details of the emission testing scenario were modelled: the set of inputs is the set of speed values, an output value represents the amount of emissions—in particular, the nitric oxide (NO_x) emissions—measured at the exhaust pipe of a car. The distance functions are the absolute differences of speed, respectively NO_x, values, and the standard behaviour is the singleton set that contains a trace that consists of the inputs that define the test cycle followed by the average amount of NO_x gas measured during the test. Thus, formally, we get $\mathsf{In} = \mathbb{R}$, $\mathsf{Out} = \mathbb{R}$, $\mathsf{Std} = \{\mathrm{NEDC} \cdot \mathsf{o}\}$,[3] and d_In and d_Out as defined in Example 4 [19].

The STL formulas developed in the previous section, combined with the probabilistic falsification approach, give rise to further improvements to the existing testing-based work [19] on diesel doping detection.

To use the falsification algorithm in Algorithm 1, we implement the restriction of the input space to $\mathsf{In}_{\{\mathrm{NEDC}\cdot\mathsf{o}\},\kappa_\mathsf{i}}$ as explained in Sect. 3. With this restriction the STL formula $\varphi_{\mathsf{u\text{-}rob}}$ from Theorem 5 can be simplified to

$$\square(d_\mathsf{Out}((\mathrm{NEDC} \cdot \mathsf{o})\downarrow_\mathsf{o}, w\downarrow_\mathsf{o}) - \kappa_\mathsf{o} \leq 0). \tag{7}$$

This is because the conjunction and disjunction over standard traces becomes obsolete for only a single standard trace. For the same reason, the requirement $\square(\mathsf{eq}(w_a\downarrow_\mathsf{i}, w_b\downarrow_\mathsf{i}) \leq 0)$ becomes obsolete, as the compared traces are always identical. In the $\mathcal{W}$ subformula, the right proposition is always false, because of the restricted input space. We implemented Algorithm 1 for the robustness computation according to formula (7).

[3] NEDC is the sequence of 1180 inputs with the kth input defining the speed of the car after k seconds from the beginning of the NEDC

In practice, running tests like NEDC with real cars is a time consuming and expensive endeavour. Furthermore, tests on chassis dynamometers are usually prohibited to be carried out with rented cars by the rental companies. On the other hand, car emission models for simulation are not available to the public—and models provided by the manufacturer cannot be considered trustworthy. To carry out our experiments, we instead use an approximation technique that estimates the amount of NO_x emissions of a car along a certain trajectory based on data recorded during previous trips with the same car, sampled at a frequency of 1 HZ (one sample per second). Notably, these trips do not need to have much in common with the trajectory to be approximated. A trip is represented as a finite sequence $\vartheta \in (\mathbb{R} \times \mathbb{R} \times \mathbb{R})^*$ of triples, where each such triple (v, a, n) represents the speed, the acceleration, and the (absolute) amount of NO_x emitted at a particular time instant in the sample. Speed and acceleration can be considered as the main parameters influencing the instant emission of NO_x. This is, for instance, reflected in the regulation [67, 124] where the decisive quantities to validate test routes for real-world driving emissions tests on public roads are speed and acceleration.

A recording $\mathcal{D}$ is the union of finitely many trips ϑ. We can turn such a recording into a predictor $\mathcal{P}$ of the NO_x values given pairs of speed and acceleration as follows:

$$\mathcal{P}(v, a) = \mathsf{average}[n \mid (\exists v', a'. (|v - v'| \leq 2 \ \wedge |a - a'| \leq 2 \ \wedge \ (v', a', n) \in \mathcal{D}))].$$

The amount of NO_x assigned to a pair (v, a) here is the average of all NO_x values seen in the recording $\mathcal{D}$ for $v \pm \ell$ and $a \pm \ell$, with $0 \leq \ell \leq 2$. To overcome measurement inaccuracies and to increase the robustness of the approximated emissions, the speed and acceleration may deviate up to 2 km/h, and 2 m/s^2, respectively. This tolerance is adopted from the official NEDC regulation [128], which allows up to 2km/h of deviations while driving the NEDC.

To demonstrate the practical applicability of our implementation of Algorithm 1 and our NO_x approximation, we report here on experiments with an Audi A6 Avant Diesel admitted in June 2020 as well as with its successor model admitted in 2021. We will refer to the former as car *A20* and to the latter as car *A21*. We used the app LolaDrives to perform in total six low-cost RDE tests—two with A20 and four[4] with A21—and recorded the data received from the cars' diagnosis ports. The raw data is available on Zenodo [15]. Using the emissions predictor proposed above we estimate that for an NEDC test A20 emits 86 mg/km of NO_x and that A21 emits 9 mg/km. Car A20 has previously been falsified w.r.t. the RDE specification. Neither A20 nor A21 has been falsified w.r.t. robust cleanness.

Before turning to falsification, we spell out meaningful contexts for robust cleanness. We identified suitable In, Out, Std, d_{In}, and d_{Out} at the beginning of the section. For κ_i, it turned out that $\kappa_\mathsf{i} = 15$ km/h is a reasonable choice, as it leaves enough flexibility for human-caused driving mistakes and intended deviations [19]. The threshold for NO_x emissions under lab conditions is 80mg/km. The emission limits for RDE tests depend on the admission date of the car. Cars admitted in 2020 or earlier, must emit 168 mg/km at most, and cars admitted later must adhere to the limit of 120 mg/km. For our experiments, we use $\kappa_\mathsf{o} = 88$ mg/km for A20 and $\kappa_\mathsf{o} = 40$ mg/km for A21 to have the same tolerances as for RDE tests. Effectively, the upper threshold for A20 is $84+88 = 172$ mg/km, and for A21 the limit is $9+40 = 49$ mg/km. Notice that for software doping analysis, the output observed for a certain standard behaviour and the constant κ_o define the effective threshold; this threshold is typically different from the thresholds defined by the regulation.

We modified Algorithm 1 by adding a timeout condition: if the algorithm is not able to find a falsifying counterexample within 3,000 iterations, it terminates and returns both the

[4] We do not consider test A21.3 in this article, see [22, Section 5] for details

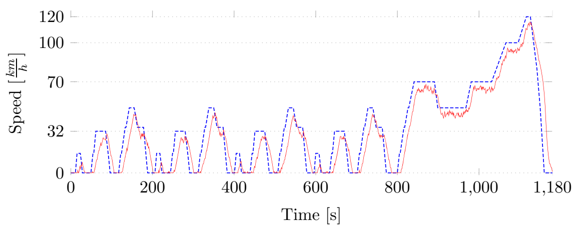

Fig. 2 NEDC speed profile (blue, dashed) and input falsifying $\mathcal{C}$ for $\kappa_{\mathsf{o}} = 88$ mg/km (red) with 182 mg/km of emitted NO$_x$

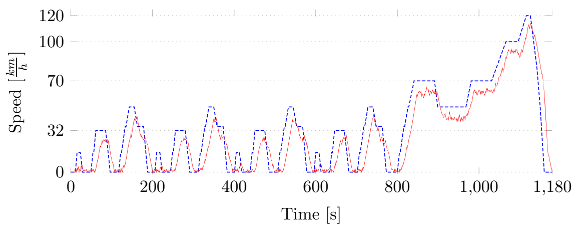

Fig. 3 NEDC speed profile (blue, dashed) and input maximising NO$_x$ emissions to 11 mg/km (red)

trace for which the smallest robustness has been observed and its corresponding robustness value. Hence, if falsification of robust cleanness for a system is not possible, the algorithm outputs an upper bound on how robust the system satisfies robust cleanness.

For the concrete case of the diesel emissions, the robustness value during the first 1180 inputs (sampled from the restricted input space $\mathsf{In}_{\mathsf{Std},\kappa_{\mathsf{i}}}$) is always κ_{o}. When the NEDC output o_{NEDC} and the non-standard output o are compared, the robustness value is $\kappa_{\mathsf{o}} - |o_{\text{NEDC}} - o|$ (cf., eq. (7), the quantitative semantics of STL, and definition of d_{Out}). Hence, for test cycles with small robustness values, we get NO$_x$ emissions o that are either very small or very large compared to o_{NEDC}. We ran the modified Algorithm 1 on A20 and A21 for the contexts defined above. For A20, it found a robustness value of -8, i.e., it was able to falsify robust cleanness relative to the assumed contract and found a test cycle for which NO$_x$ emissions of 182 mg/km are predicted. The test cycle is shown in Fig. 2. For A21, the smallest robustness estimate found—even after 100 independent executions of the algorithm—was 38, i.e., A21 is predicted to satisfy robust cleanness with a very high robustness estimate. The corresponding test cycle is shown in Fig. 3.

On Doping Tests for Cyber-physical Systems

The proposed probabilistic falsification approach to find instances of software doping needs several hundreds of iterations. This is problematic for testing real-world cyber-physical systems (CPS) to which inputs cannot be passed in an automated way. To conduct a test with a car, for example, the input to the system is a test cycle that is passed to the vehicle by driving it. Notably, we consider here the scenario that the CPS is tested by an entity that is different from the manufacturer. While the latter might have tools to overcome these technical challenges, the former typically does not have access to them.

We propose the following *integrated testing approach* for effective doping tests of cyber-physical systems. The big picture is provided in Fig. 4. In a first step, the CPS is used under real-world conditions without enforcing any specific constraints on the inputs to the system. For all executions, the inputs and outputs are recorded. So, essentially, the system can be used as it is needed by the user, but all interactions with it are recorded. From these recordings, a *model* can be learned that for arbitrary inputs (whether they were covered in the recorded data or not) predicts the output of the system. Such learning can be as simple as using statistics as we did for the emissions example above, or as complex as using deep neural nets. For the

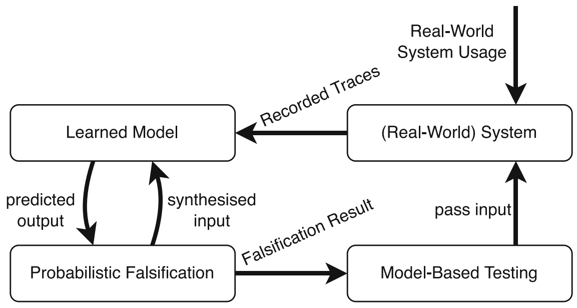

Fig. 4 Integrated testing approach

learned model, the probabilistic falsification algorithm computes a test input that falsifies it—inputs to this model can be passed automatically and an output is produced almost instantly. The resulting input serves as an input for the real CPS. If the prediction was correct, also the real system is falsified. If it was incorrect, the learned model can be refined and the process starts again.

For diesel emissions, the first part of this integrated testing approach has been carried out as part of the work reported in this article. We leave the second part—evaluating the generated test traces from Figs. 2 and 3 with a real car—for future work.

Technical Context

Software doping theory provides a formal basis for enlarging the requirements on vehicle exhaust emissions beyond too narrow lab test conditions. That conceptual limitation has by now been addressed by the official authorities responsible for car type approval [124, 127]: The old NEDC-based test procedure is replaced by the newer *Worldwide Harmonised Light Vehicles Test Procedure* (WLTP), which is deemed to be more realistic. WLTP replaces the NEDC test by a new WLTC test, but WLTC still is just a single test scenario. In addition, WLTP embraces so called *Real Driving Emissions* (RDE) tests to be conducted on public roads. A recently launched mobile phone app [20, 22], LolaDrives, harvests runtime monitoring technology for making low-cost RDE tests accessible to everyone.

Learning or approximating the behaviour of a system under test has been studied intensively. Meinke and Sindhu [82] were among the first to present a testing approach incrementally learning a Kripke structure representing a reactive system. Volpato and Tretmans [130] propose a learning approach which gradually refines an under- and over-approximation of an input-output transition system representing the system under test. The correctness of this approach needs several assumptions, e.g., an oracle indicating when, for some trace, all outputs, which extend the trace to a valid system trace, have been observed.

5 Individual fairness of systems evaluating humans

Example 2 introduces a new application domain for cleanness definitions. Unica uses an AI system that is supposed to assist her with the selection of applicants for a hypothetical university. Cleanness of such a system can be related to the fair treatment of the humans that are evaluated by it. A usable fairness analysis can happen no later than at runtime, since Unica needs to make a timely decision on whether to include the applicant in further considerations. We describe technical measures that help in mitigating this challenge by providing her with information from an individual fairness analysis in a suitable, purposeful, expedient way. To this end, we propose a formal definition for individual fairness extending the one by

[35] and based on func-cleanness. We develop a runtime monitor that analyses every output of P immediately after P's decision, which strategically searches for unfair treatment of a particular individual by comparing them to relevant hypothetical alternative individuals so as to provide a fairness assessment in a timely manner.

Much like P is to support Unica, AI systems—in the broadest sense of the word—more and more often support human decision makers. Undoubtedly, such systems should be compliant with applicable law (such as the future European AI Act [40, 41] or the Washington State facial recognition law [132]) and ought to minimise any risks to health, safety or fundamental rights. Sometimes, we cannot mitigate all these risks in advance by technical measures and also some risk-mitigation requires trade-off decisions involving features that are either impossible or difficult to operationalise and formalise. This is why it is essential that a human effectively oversees the system (which is also emphasised by several institutions such as UNESCO [129] and the European High Level Expert Group [59]). *Effective* human oversight, however, is only possible with the appropriate technical measures that allow human overseers to better understand the system at runtime [70, 71]. From a technical point of view, this raises the pressing question of what such technical measures can and ought to look like to actually enable humans to live up to these responsibilities. Our contribution is intended to bridge the gap between the normative expectations of law and society and the current reality of technological design.

5.1 Positioning within related research topics

Our contribution draws on and adds to three vibrant topics of current research, namely Explainable AI (XAI), AI fairness, and discrimination.

XAI

Many of the most successful AI systems today are some kind of black boxes [11]. Accordingly, the field of 'Explainable AI' [54] focuses on the question of how to provide users (and possibly other stakeholders) with more information via several key perspicuity properties [117] of these systems and their outputs to make them understand these systems and their outputs in ways necessary to meet various desiderata [5, 28, 69, 74, 85, 91]. The concrete expectations and promises associated with various XAI methods are manifold. Among them are enabling warranted trust in systems [12, 62, 65, 102, 111], increasing human-system decision-making performance [68] for instance through increasing human situation awareness when operating systems [109], enabling responsible decision-making and effective human oversight [14, 80, 114], as well as identifying and reducing discrimination [74]. It often remains unclear what kind of explanations are generated by the various explainability methods and how they are meant to contribute to the fulfilment of the desiderata, even though these questions have become the subject of systematic and interdisciplinary research [69, 103].

Our approach can be taxonomised along at least two different distinctions [70, 86, 101, 102, 116]: First, it is *model-agnostic* (not *model-specific*), i.e., it is not tailored to a particular class of models but operates on observable behaviour—the inputs and outputs of the model. Second, our method is a *local method* (not *global*), i.e., it is meant to shed light on certain outputs rather than the system as a whole.

(Un-)Fair Models

Fairness, discrimination, justice, equal opportunity, bias, prejudice, and many more such concepts are part of a meaningfully interrelated cluster that has been analysed and dissected for millennia [6, 7]. Many fields are traditionally concerned with the concepts of fairness and discrimination, ranging from philosophy [6, 7, 36, 52, 98–100] to legal sciences [25, 57, 125,

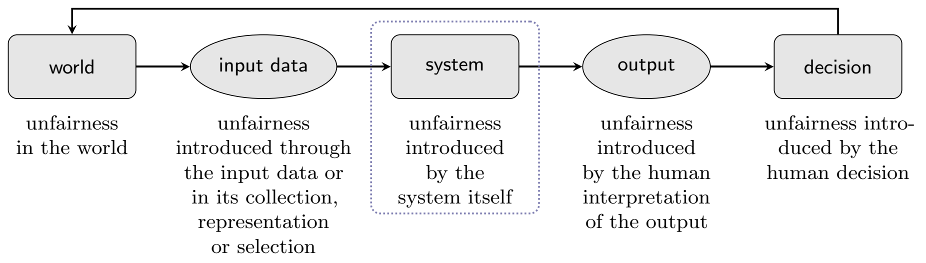
world
input data
system
output
decision
unfairness in the world
unfairness introduced through the input data or in its collection, representation or selection
unfairness introduced by the system itself
unfairness introduced by the human interpretation of the output
unfairness introduced by the human decision

Fig. 5 Sketch of different origins of unfairness in a decision process supported by a system; the dotted box indicates which unfairness our monitoring targets

131], to psychology [60, 136], to sociology [2, 63], to political theory [99], to economics [55]. Nowadays, it has also become a technological topic that calls for cross-disciplinary perspectives [50].

With regard to fairness, there are two distinctions that are especially relevant to our work. First, one distinction is made between *individual fairness*, i.e., that similar individuals are treated similarly [35], and *group fairness*, i.e., that there is adequate group parity [23]. Measures of individual fairness are often close to the Aristotelian dictum to treat like cases alike [6, 7]. In a sense, operationalisations of individual fairness are robustness measures [24, 118], but instead of requiring robustness with respect to noise or adversarial attacks, measures of individual fairness, such as the one by Dwork et al. [35], call for robustness with respect to highly context-dependent differences between representations of human individuals. Second, recent work from the field of law [131] suggests to differentiate between *bias preserving* and *bias transforming* fairness metrics. Bias preserving fairness metrics seek to avoid adding new bias. For such metrics, historic performances are the benchmarks for models, with equivalent error rates for each group being a constraint. In contrast, bias transforming metrics do not accept existing bias as a given or neutral starting point, but aim at adjustment. Therefore, they require to make a ‘positive normative choice’ [131], i.e. to actively decide which biases the system is allowed to exhibit, and which it must not exhibit.

Over the years, many concrete approaches have been suggested to foster different kinds of fairness in artificial systems, especially in AI-based ones [74, 81, 96, 131, 134]. Yet, to the best of our knowledge, an approach like ours is still missing. One of the approaches that is closest to ours, namely that by John et al. [64], is not local and therefore not suitable for runtime monitoring. Also, it is not model-agnostic. So, to the best of our knowledge, our approach provides a new contribution to the debate on unfairness detection.

It is important to note/recognise that our approach can only be understood as part of a more holistic approach to preventing or reducing unfairness. After all, there are many sources of unfairness [8] (also see Fig. 5 and Appendix B). Therefore, not every technical measure is able to detect every kind of unfairness and eliminating one source of unfairness might not be sufficient to eliminate all unfairness. Our approach tackles only unfairness introduced by the system, but not other kinds of unfairness.

Discrimination

We understand discrimination as dissimilar treatment of similar cases or similar treatment of dissimilar cases without justifying reason. This is a definition that can also be found in the law [44, §43]. Our work is exclusively focused on discrimination *qua* dissimilar treatment of similar cases. Discrimination requires a thoughtful and largely not formalisable consideration of ‘justifying reason’. However, we will exploit the relation of discrimination and fairness:

Unfairness in a system can arguably be a good proxy of discrimination—even though not every unfair treatment by a system necessarily constitutes discrimination (especially not in the legal sense). Thus, a tool that highlights cases of unfairness in a system can be highly instrumental in detecting discriminatory features of a system. It is not viable, though, to let such a tool rule out unfair treatment fully automatically without human oversight, since there could be justifying reason to treat two similar inputs in a dissimilar way.

5.2 Individual fairness

Unica from Example 2 should be able to detect individual unfairness. An operationalisation thereof by Dwork et al. [35] is based on the Lipschitz condition to enforce that similar individuals are treated similarly. To measure similarity, they assume the existence of an input distance function d_{In} and an output distance function d_{Out}. This assumption is very similar to the one that we implicitly made in the previous sections for robust cleanness and func-cleanness. However, in the case of the fair treatment of humans finding reasonable distance functions is more challenging than it was for the examples in the previous chapters. Dwork et al. assume that both distance functions perfectly measure distances between individuals[5] and between outputs of the system, respectively, but admit that in practice these distance functions are only approximations of a ground truth at best. They suggest that distance measures might be learned, but there is no one-size-fits-all approach to selecting distance measures. Indeed, obtaining such distance metrics is a topic of active research [61, 87, 135]. Additionally, the Lipschitz condition assumes a Lipschitz constant L to establish a linear constraint between input and output distances.

Definition 8 A deterministic sequential program $\mathsf{P} : \mathsf{In} \to \mathsf{Out}$ is *Lipschitz-fair* w.r.t. $d_{\mathsf{In}} : \mathsf{In} \times \mathsf{In} \to \mathbb{R}$, $d_{\mathsf{Out}} : \mathsf{Out} \times \mathsf{Out} \to \mathbb{R}$, and a Lipschitz constant L, if and only if for all $\mathsf{i}_1, \mathsf{i}_2 \in \mathsf{In}$, $d_{\mathsf{Out}}(\mathsf{P}(\mathsf{i}_1), P(\mathsf{i}_2)) \leq L \cdot d_{\mathsf{In}}(\mathsf{i}_1, \mathsf{i}_2)$.

Lipschitz-fairness comes with some restrictions that limit its suitability for practical application:

d_{In}-d_{Out}-relation: High-risk systems are typically complex systems and ask for more complex fairness constraints than the linearly bounded output distances provided by the Lipschitz condition. For example, using the Lipschitz condition prevents us from allowing small local jumps in the output and at the same time forbidding jumps of the same rate of increase over larger ranges of the input space (also see supplementary material in Section Appendix A).

Input relevance: The condition quantifies over the entire input domain of a program. This overlooks two things: first, it is questionable whether each input in such a domain is plausible as a representation for a real-world individual. But whether a system is unfair for two implausible and purely hypothetical inputs is largely irrelevant in practice. Secondly, it also ignores that mere potential unfair treatment is at most a threat, not necessarily already a harm [106]. Therefore, even with a restriction to only plausible applicants, the analysis might take into account more inputs than needed for

[5] For easier readability, we will not distinguish between *individuals* and their *representations* unless this distinction is relevant in the specific context. It is nevertheless important to note that inputs are not individuals, but only representations of individuals, since an input could inadequately represent an individual and therefore be unfair (also see Appendix B).

many real-world applications. What is important in practice is the ability to determine whether *actual* applicants are treated unfairly—and for this it is often not needed to look at the entire input domain.

Monitorability: In a monitoring scenario with the Lipschitz condition in place, a fixed input i_1 must be compared to potentially all other inputs i_2. Since the input domain of the system can be arbitrarily large, the Lipschitz condition is not yet suitable for monitoring in practice (for a related point see John et al. [64]).

We propose a notion of individual fairness that is based on Definition 3. Instead of cleanness contracts we consider here *fairness contracts*, which are tuples $\mathcal{F} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ containing input and output distance functions and the function f relating input distances and output distances. Notably, the set of standard inputs StdIn known from cleanness contracts is not part of a fairness contract; it is unknown what qualifies an input to be 'standard' in the context of fairness analyses. Still, our fairness definition evaluates fairness for a set of individuals $\mathcal{I} \subseteq \mathsf{In}$ (e.g., a set of applicants), which has conceptual similarities to the set StdIn. A fairness contract specifies certain fairness parameters for a concrete context or situation. Such parameters should generally not already include $\mathcal{I}$ to avoid introducing new unfairness through the monitor by tailoring it to specific inputs individually or by treating certain inputs differently from others. Func-fairness can thus be defined as follows:

Definition 9 A deterministic sequential program $\mathsf{P} : \mathsf{In} \to \mathsf{Out}$ is *func-fair* for a set $\mathcal{I} \subseteq \mathsf{In}$ of actual inputs w.r.t. a fairness contract $\mathcal{F} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$, if and only if for every $\mathsf{i} \in \mathcal{I}$ and $\mathsf{i}' \in \mathsf{In}$, $d_{\mathsf{Out}}(\mathsf{P}(\mathsf{i}), \mathsf{P}(\mathsf{i}')) \leq f(d_{\mathsf{In}}(\mathsf{i}, \mathsf{i}'))$.

The idea behind func-fairness is that every individual in set $\mathcal{I}$ is compared to potential other inputs in the domain of P. These other inputs do not necessarily need to be in $\mathcal{I}$, nor do these inputs need to have "physical counterparts" in the real world. Driven by the insights of the *Input relevance* restriction of Lipschitz-fairness, we explicitly distinguish inputs in the following and will call inputs that are given to P by a user *actual inputs*, denoted i_a, and call inputs to which such i_a are compared to *synthetic inputs*, denoted i_s. Actual inputs are typically[6] inputs that have a real-world counterpart, while this might or might not be true for synthetic inputs. On first glance, an alternative to using synthetic inputs is to use only actual inputs, e.g., to compare every actual input with every other actual input in $\mathcal{I}$. For example, for a university admission, all applicants could be compared to every other applicant. However, this would heavily rely on contingencies: the detection of unfair treatment of an applicant depends on whether they were lucky enough that, coincidentally, another candidate has also applied who aids in unveiling the system's unfairness towards them. Instead, func-fairness prefers to over-approximate the set of plausible inputs that actual inputs are compared to rather than under-approximating it by comparing only to other inputs in $\mathcal{I}$. This way, the attention of the human exercising oversight of the system might be drawn to cases that are actually not unfair, but as a competent human in the loop, they will most likely be able to judge that the input was compared to an implausible counterpart. This will usually enable more effective human oversight than an under-approximation that misses to alert the human to unfair cases.

Notice that func-fairness is a conservative extension of Lipschitz-fairness. With $\mathcal{I} = \mathsf{In}$ and $f(x) = L \cdot x$, func-fairness mimics Lipschitz-fairness. Wachter et al. [131] classify the Lipschitz-fairness of Dwork et al. [35] as bias-transforming. As we generalise this and

[6] A case where actual inputs might not have real-world counterparts is testing.

Algorithm 2 $\mathsf{FairnessMonitor}$, with $\xi\text{-}\mathsf{min}\ S = (\xi, \mathsf{i}_1, \mathsf{i}_2)$ only if $(\xi, \mathsf{i}_1, \mathsf{i}_2) \in S$ and for all $(\xi', \mathsf{i}'_1, \mathsf{i}'_2) \in S, \xi' \geq \xi$

```
Falsification Parameters: PS: Proposal scheme, $\beta$: Temperature parameter
Input: System $P : \mathsf{In} \to \mathsf{Out}$, Fairness contract $\mathcal{F} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$, and set of actual inputs $\mathcal{I}$
Output: A minimal fairness score triple from $\mathbb{R} \times \mathcal{I} \times \mathsf{In}$.
1: $\mathsf{i_s} \leftarrow$ any input $\mathsf{i_a} \in \mathcal{I}$
2: $(\xi, \mathsf{i_{min}}, \mathsf{i_s}) \leftarrow \xi\text{-}\mathsf{min}\{(F(\mathsf{i_a}, \mathsf{i_s}), \mathsf{i_a}, \mathsf{i_s}) \mid \mathsf{i_a} \in \mathcal{I}\}$
3: $(\xi_{\mathsf{min}}, \mathsf{i}_1, \mathsf{i}_2) \leftarrow (\xi, \mathsf{i_{min}}, \mathsf{i_s})$
4: while not timeout do
5:    $\mathsf{i}'_\mathsf{s} \leftarrow \mathsf{PS}(\mathsf{i_s}, \mathsf{P}(\mathsf{i_s}))$
6:    $(\xi', \mathsf{i}'_{\mathsf{min}}, \mathsf{i}'_\mathsf{s}) \leftarrow \xi\text{-}\mathsf{min}\{(F(\mathsf{i_a}, \mathsf{i}'_\mathsf{s}), \mathsf{i_a}, \mathsf{i}'_\mathsf{s}) \mid \mathsf{i_a} \in \mathcal{I}\}$
7:    $(\xi_{\mathsf{min}}, \mathsf{i}_1, \mathsf{i}_2) \leftarrow \xi\text{-}\mathsf{min}\{(\xi_{\mathsf{min}}, \mathsf{i}_1, \mathsf{i}_2), (\xi', \mathsf{i}'_{\mathsf{min}}, \mathsf{i}'_\mathsf{s})\}$
8:    $\alpha \leftarrow \exp(-\beta(\xi' - \xi))$
9:    $r \leftarrow \mathsf{UniformRandomReal}(0, 1)$
10:   if $r \leq \alpha$ then
11:      $\mathsf{i_s} \leftarrow \mathsf{i}'_\mathsf{s}$
12:      $\xi \leftarrow \xi'$
13:   end if
14: end while
15: return$(\xi_{\mathsf{min}}, \mathsf{i}_1, \mathsf{i}_2)$
```

introduce no element that has to be regarded as bias-preserving, our approach arguably is bias-transforming, too.

Func-fairness, with its function f, provides a powerful tool to model complex fairness constraints. How such an f is defined has profound impact on the quality of the fairness analysis. A full discussion about which types of functions make a good f go beyond the scope of this article. A suitable choice for f and the distance functions d_{In} and d_{Out} heavily depends on the context in which fairness is analysed—there is no one-fits-it-all solution. Func-fairness makes this explicit with the formal fairness contract $\mathcal{F} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$.

5.3 Fairness monitoring

We develop a probabilistic-falsification-based fairness monitor that, given a set of actual inputs, searches for a synthetic counterexample to falsify a system P w.r.t. a fairness contract $\mathcal{F}$. To this end, it is necessary to provide a quantitative description of func-fairness that satisfies the characteristics of a robustness estimate. We call this description *fairness score*. For an actual input $\mathsf{i_a}$ and a synthetic input $\mathsf{i_s}$ we define the fairness score as $F(\mathsf{i_a}, \mathsf{i_s}) := f(d_{\mathsf{In}}(\mathsf{i_a}, \mathsf{i_s})) - d_{\mathsf{Out}}(\mathsf{P}(\mathsf{i_a}), \mathsf{P}(\mathsf{i_s}))$. F is indeed a robustness estimate function: if $F(\mathsf{i_a}, \mathsf{i_s})$ is non-negative, then $d_{\mathsf{Out}}(\mathsf{P}(\mathsf{i_a}), \mathsf{P}(\mathsf{i_s})) \leq f(d_{\mathsf{In}}(\mathsf{i_a}, \mathsf{i_s}))$, and if it is negative, then $d_{\mathsf{Out}}(\mathsf{P}(\mathsf{i_a}), \mathsf{P}(\mathsf{i_s})) \not\leq f(d_{\mathsf{In}}(\mathsf{i_a}, \mathsf{i_s}))$. For a set of actual inputs $\mathcal{I}$, the definition generalises to $F(\mathcal{I}, \mathsf{i_s}) := \min\{F(\mathsf{i_a}, \mathsf{i_s}) \mid \mathsf{i_a} \in \mathcal{I}\}$, i.e., the overall fairness score is the minimum of the concrete fairness scores of the inputs in $\mathcal{I}$. Notice that $\mathcal{R}_{\mathcal{I}}(\mathsf{i_s}) := F(\mathcal{I}, \mathsf{i_s})$ is essentially the quantitative interpretation of $\varphi_{\mathsf{u\text{-}func}}$ (from Theorem 6) after simplifications attributed to the fact that P is a sequential and deterministic program (cf. Definition 2.2 vs. Definition 3).

Algorithm 2 shows $\mathsf{FairnessMonitor}$, which builds on Algorithm 1 to search for the minimal fairness score in a system P for fairness contract $\mathcal{F}$. The algorithm stores fairness scores in triples that also contain the two inputs for which the fairness score was computed. The minimum in a set of such triples is defined by the function $\xi\text{-}\mathsf{min}$ that returns the triple with the smallest fairness score of all triples in the set. The first line of $\mathsf{FairnessMonitor}$ initialises the variable $\mathsf{i_s}$ with an arbitrary actual input from $\mathcal{I}$. For this value of $\mathsf{i_s}$, the algorithm checks

Algorithm 3 FairnessAwareSystem

```
Parameters: System $\mathsf{P} : \mathsf{In} \to \mathsf{Out}$, Fairness contract $\mathcal{F} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$
Input: Input $\mathsf{i_a} \in \mathsf{In}$
Output: Tuple of the system output, normalised fairness score, and synthetic values witnessing the fairness score
1: $(\xi_{\mathsf{min}}, \mathsf{i_a}, \mathsf{i_s}) \leftarrow \mathsf{FairnessMonitor}(\mathsf{P}, \mathcal{F}, \{\mathsf{i_a}\})$
2: return$(\mathsf{P}(\mathsf{i_a}), \xi_{\mathsf{min}} \div f(d_{\mathsf{In}}(\mathsf{i_a}, \mathsf{i_s})), (\mathsf{i_s}, \mathsf{P}(\mathsf{i_s})))$
```

the corresponding fairness scores for all actual inputs $\mathsf{i_a} \in \mathcal{I}$ and stores the smallest one. In line 3, the globally smallest fairness score triple is initialised. In line 5 it uses the proposal scheme to get the next synthetic input i'_s. Line 6 is similar to line 2: for the newly proposed i'_s it finds the smallest fairness score, stores it, and updates the global minimum if it found a smaller fairness score (line 7). Lines 8-13 come from Algorithm 1. The only difference is that in addition to $\mathsf{i_s}$ we also store the fairness score ξ. Line 4 of Algorithm 2 differs from Algorithm 1 by terminating the falsification process after a timeout occurs (similar to the adaptation of Algorithm 1 in Sect. 4). Hence, the algorithm does not (exclusively) aim to falsify the fairness property, but aims at minimising the fairness score; even if the fair treatment of the inputs in $\mathcal{I}$ cannot be falsified in a reasonable amount of time, we still learn how robustly they are treated fairly, i.e., how far the least fairly treated individual in $\mathcal{I}$ is away from being treated unfairly. After the timeout occurs, the algorithm returns the triple with the overall smallest seen fairness score ξ_{min}, together with the actual input i_1 and the synthetic input i_2 for which ξ_{min} was found. In case ξ_{min} is negative, i_2 is a counterexample for P being func-fair.

FairnessMonitor implements a sound $\mathcal{F}$-unfairness detection as stated in Proposition 7. However, it is not complete, i.e., it is not generally the case that P is func-fair for $\mathcal{I}$ if ξ is positive. It may happen that there is a counterexample, but FairnessMonitor did not succeed in finding it before the timeout. This is analogue to results obtained for model-agnostic robust cleanness analysis [19].

Proposition 7 *Let* $\mathsf{P} : \mathsf{In} \to \mathsf{Out}$ *be a deterministic sequential program,* $\mathcal{F} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ *a fairness contract, and* $\mathcal{I}$ *a set of actual inputs. Further, let* $(\xi_{\mathsf{min}}, \mathsf{i}_1, \mathsf{i}_2)$ *be the result of* $\mathsf{FairnessMonitor}(\mathsf{P}, \mathcal{F}, \mathcal{I})$. *If* ξ_{min} *is negative, then* P *is not func-fair for* $\mathcal{I}$ *w.r.t.* $\mathcal{F}$.

Moreover, FairnessMonitor circumvents major restrictions of the Lipschitz-fairness:

d_{In}-d_{Out}-relation: Func-fairness defines constraints between input and output distances by means of a function f, which allows to express also complex fairness constraints. For a more elaborate discussion, see Sect. Appendix A.

Input relevance: Func-fairness explicitly distinguishes between actual and synthetic inputs. This way, func-fairness acknowledges a possible obstacle of the fairness theory when it comes to a real-world usage of the analysis, namely that only some elements of the system's input domain might be plausible and that usually only few of them become actual inputs that have to be monitored for unfairness.

Monitorability: FairnessMonitor demonstrates that func-fairness is monitorable. It resolves the quantification over In using the above concepts from probabilistic falsification using the robustness estimate function F as defined above.

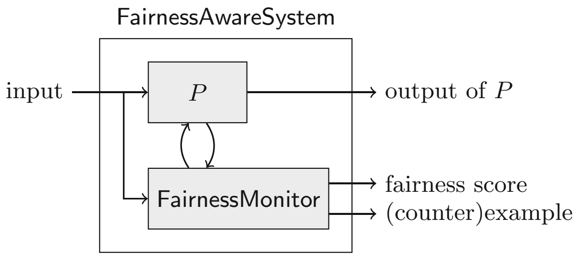

Fig. 6 Schematic visualisation of FairnessAwareSystem

Towards func-fairness in the loop
If a high-risk system is in operation, a human in the loop must oversee the correct and fair functioning of the outputs of the system. To do this, the human needs real-time fairness information. Figure 6 shows how this can be achieved by coupling the system P and the FairnessMonitor in Algorithm 2 in a new system called FairnessAwareSystem. FairnessAwareSystem is sketched in Algorithm 3. Intuitively, the FairnessAwareSystem is a higher-order program that is parameterised with the original program P and the fairness contract $\mathcal{F}$. When instantiated with these parameters, the program takes arbitrary (actual) inputs $\mathsf{i_a}$ from In. In the first step, it does a fairness analysis using FairnessMonitor with arguments P, $\mathcal{F}$, and $\{\mathsf{i_a}\}$. To make fairness scores comparable, FairnessAwareSystem normalises the fairness score ξ received from FairnessMonitor by dividing[7] it by the output distance limit $f(d_{\mathsf{In}}(\mathsf{i_a}, \mathsf{i_s}))$. For fair outputs, the score will be between 0 (almost unfair) and 1 (as fair as possible).[8] Outputs that are not func-fair are accompanied by a negative score representing how much the limit $f(d_{\mathsf{In}}(\mathsf{i_a}, \mathsf{i_s}))$ is exceeded. A fairness score of $-n$ means that the output distance of $\mathsf{P}(\mathsf{i_a})$ and $\mathsf{P}(\mathsf{i_s})$ is $n+1$ times as high as that limit. Finally, FairnessAwareSystem returns the triple with P's output for $\mathsf{i_a}$, the normalised fairness score, and the synthetic input with its output witnessing the fairness score.
Interpretation of monitoring results
Especially when FairnessAwareSystem finds a violation of func-fairness, the suitable interpretation and appropriate response to the normalised fairness score proves to be a non-trivial matter that requires expertise.

Example 6 Instead of using P from Example 2 on its own, Unica now uses FairnessAwareSystem with a suitable fairness contract. and thereby receive a fairness score along with P's verdict on each applicant. (Which fairness contracts are suitable is an open research problem, see *Limitations & Challenges* in Sect. 7.) If the fairness score is negative, she can also take into account the information on the synthetic counterpart returned by FairnessAwareSystem. Among the 4096 applicants for the PhD program, the monitoring assigns a negative fairness score to three candidates: Alexa, who received a low score, Eugene, who was scored very highly, and John, who got an average score. According to their scoring, Alexa would be desk-rejected, while Eugene and John would be considered further.

Alexa's synthetic counterpart, let's call him Syntbad, is ranked much higher than Alexa. In fact, he is ranked so high that Syntbad would not be desk-rejected. Unica compares Alexa

[7] For f that can return 0, there may be a $0 \div 0$ division. The result of this division should be defined depending on the concrete context; reasonable values range from the extreme scores 0 (to indicate that the score is on the edge to becoming 'unfair') to 1 (to indicate that more fairness is impossible).

[8] Fairness may be a vague concept that cannot be dichotomised. By its choice of the fairness contract parameters, our approach nevertheless specifies a (non-arbitrary) cut-off point at 0; but it does so for purely instrumental and non-ontological reasons.

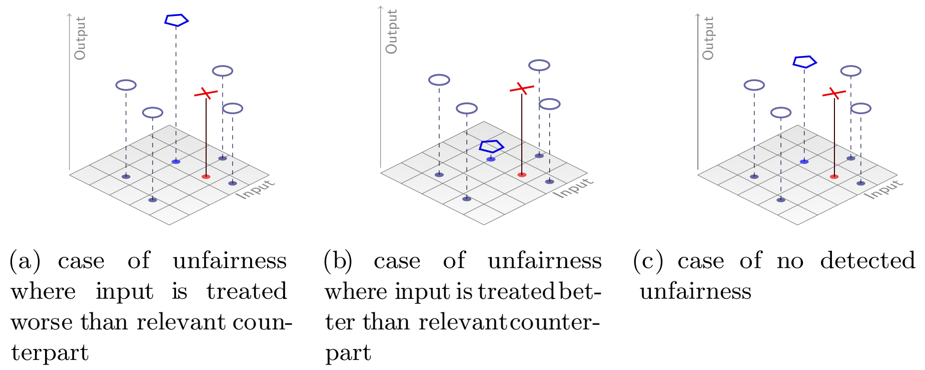
Output
Input
(a) case of unfairness where input is treated worse than relevant counterpart
(b) case of unfairness where input is treated better than relevant counterpart
(c) case of no detected unfairness

Fig. 7 Exemplary illustration of configurations of an input (red cross) and its synthetic counterparts (grey circles) and the synthetic counterpart with the minimal fairness score (blue polygon); with a two-dimensional input space (grid) and a one-dimensional output

and Syntbad and finds that they only differ in one respect: Syntbad's graduate university is the one in the official ranking that is immediately *below* the one that Alexa attended. Unica does some research and finds that Alexa's institution is predominantly attended by People of Colour, while this is not the case for Syntbad's institution. Therefore, FairnessAwareSystem helped Unica not only to find an unfair treatment of Alexa, but also to uncover a case of potential racial discrimination.

John's counterpart, Synclair, is ranked much lower than him. Unica manually inspects John's previous institution (an infamous online university), his GPA of 1.8, and his test result with only 13%. She finds that this very much suggests that John will not be a successful PhD candidate and desk-rejects him. Therefore, Unica has successfully used FairnessAwareSystem to detect a fault in scoring system P whereby John would have been treated unfairly in a way that would have been to his advantage.

Eugene received a top score, but his synthetic counterpart, Syna, received only an average one. Unica suspects that Eugene was ranked too highly given his graduate institution, GPA, and test score. However, as he would not have been desk-rejected either way, nothing changes for Eugene, and the unfairness he was subject to, is not of effect to him.

The cases of John and Eugene share similarities with the configuration in (b) in Fig. 7, the one of Alexa with (a), and the ones of all other 4093 candidates with (c).

If our monitor finds only a few problematic cases in a (sufficiently large and diverse) set of inputs, our monitoring helps Unica from our running example by drawing her attention to cases that require special attention. Thereby, individuals who are judged by the system have a better chance of being treated fairly, since even rare instances of unfair treatment are detected. If, on the other hand, the number of problematic cases found is large, or Unica finds especially concerning cases or patterns, this can point to larger issues within the system. In these cases, Unica should take appropriate steps and make sure that the system is no longer used until clarity is established why so many violations or concerning patterns are found. If the system is found to be systematically unfair, it should arguably be removed from the decision process. A possible conclusion could also be that the system is unsuitable for certain use cases, e.g., for the use on individuals from a particular group. Accordingly, it might not have to be removed altogether but only needs to be restricted such that problematic use cases are avoided. In any case, significant findings should also be fed back to developers or deployers of the potentially problematic system. A fairness monitoring such as in FairnessAwareSystem or a fairness analysis as in FairnessMonitor could also be useful to developers, regulating authorities, watchdog organisations, or forensic analysts as it helps them to check the individual fairness of a system in a controlled environment.

6 Interdisciplinary assessment of fairness monitoring

Regulations for car related emissions are in force for a considerable amount of time, thus, its legal interpretation is mostly clear. In case of human oversight of AI systems, the AI act is new and parts of it are legally ambiguous. This raises the question of whether our approach meets requirements that go beyond pre-theoretical deliberations. Even though comprehensive analyses would go far beyond the scope of this paper, we will nevertheless assess some key normative aspects in philosophical and legal terms, and also briefly turn to the related empirical aspects, especially from psychology.

6.1 Psychological assessment

Fairness monitoring promises various advantages in terms of human-system interaction in application contexts—provided it is extended by an adequate user interface—which call for empirical tests and studies. We will only discuss a possible benefit that closely aligns with the current draft of the AI Act: our approach may support effective human oversight. Two central aspects of effective oversight are situation awareness and warranted trust. Our method highlights unfairness in outputs which can be expected to increase users' situation awareness (i.e., 'the perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future' [37, p. 36]), which is a variable central for effective oversight [38]. In the minimal case, this allows users to realise that something requires their attention and that they should check the outputs for plausibility and adequacy. In the optimal case and after some experience with the monitor, it may even allow users to predict instances where a system will produce potentially unfair outputs. In any case, the monitoring should enable them to understand limitations of the system and to feed back their findings to developers who can improve the system. This leads us to warranted trust, which includes that users are able to adequately judge when to rely on system outputs and when to reject them [62, 73]. Building warranted trust strongly depends on users being able to assess system trustworthiness in the given context of use [73, 110]. According to their theoretical model on trust in automation, Lee and See [73] propose that trustworthiness relates to different facets of which performance (e.g., whether the system performs reliably with high accuracy) and process (e.g., knowing how the system operates and whether the system's decision-processes help to fulfil the trustor's goals) are especially relevant in our case. Specifically, fairness monitoring should enable users to more accurately judge system performance (e.g., by revealing possible issues with system outputs) and system processes (e.g., whether the system's decision logic was appropriate). In line with Lee and See's propositions, this should provide a foundation for users to be better able to judge system trustworthiness and should thus be a promising means to promote warranted trust. In consequence, our monitoring provides a needed addition to high-risk use contexts of AI because it offers information enabling humans to more adequately use AI-based systems in the sense of possibly better human-system decision performance and with respect to user duties as described in the AI Act.

6.2 Philosophical assessment

More effective oversight promises more informed decision-making. This, in turn, enables morally better decisions and outcomes, since humans can morally ameliorate outcomes in terms of fairness and can see to it that moral values are promoted. Also, fairness monitoring helps in safeguarding fundamental democratic values if it is applied to potentially unfair systems which are used in certain societal institutions of a high-risk character such as courts or parliaments. It could, for example, make AI-aided court decisions more transparent and promote equality before the law. However, since our approach requires finding context-appropriate and morally permissible parameters for $\mathcal{F}$, moral requirements arise to enable the finding of such parameters. This not only affects, e.g., developers of such systems, but also those who are in a position to enforce that adequate parameters are chosen, such as governmental authorities, supervising institutions or certifiers.

Apart from that, various parties have arguably a legitimate interest in adequately ascribing moral responsibility for the outcomes of certain decisions to human deciders [14]—regardless of whether the decision making process is supported by a system. Adequately ascribing moral responsibility is not always possible, though. One precondition for moral responsibility is that the agent had sufficient epistemic access to the consequences of their doing [90, 119], i.e., that they have enough and sufficiently well justified beliefs about the results of their decision. Someone overseeing a university selection process (like Unica) should, for example, have sufficiently well justified beliefs that, at the very least, their decisions do not result in more unfairness in the world. If the admission process is supported by a black-box system, though, Unica cannot be expected to have any such beliefs since she lacks insight in the fairness of the system. Therefore, adequate responsibility ascription is usually not possible in this scenario. Our monitoring alleviates this problem by providing the decider with better epistemic access to the fairness of the system.

FairnessAwareSystem helps in making Unica's role in the decision process significant and not only that of a mere button-pusher. FairnessAwareSystem makes it possible for her to fulfil some of the responsibilities and duties plausibly associated with her role. For example, she can now be realistically expected to not only detect, but resolve at least some cases of apparent unfairness competently (although she may need additional information to do so). In this respect, she should not be 'automated away' (cf. [79]).

6.3 Legal assessment

A central legislative debate of our time is how to counter the risks AI systems can pose to the health and safety or fundamental rights of natural persons. Protective measures must be taken at various levels: First, before being permitted on the market, it must be ensured *ex ante* that such high-risk AI-systems are in conformity with mandatory requirements[9] regarding safety and human rights. This means in particular that the selection of the properties which a system should exhibit requires a positive normative choice and should not simply replicate biases present in the status quo [131]. In addition, AI-systems must be designed and developed in such a way that natural persons can oversee their functioning. For this purpose, it is necessary for the provider to identify appropriate human oversight measures before its placing on the market or putting into service. In particular, such measures should guarantee that the natural

[9] The specific risks set by AI-systems may also give reason to consider an adaptation and expansion of European legal frameworks such that an even broader prohibition of discrimination (cf. "Appendix C.1") is set into place.

persons to whom human oversight has been assigned have the necessary competence, training and authority to carry out that role [40, recital 48] [41, Art. 14 (5)].

Second, during runtime, the proper functioning of high-risk AI systems, which have been placed on the market lawfully, must be ensured. To achieve this goal, a bundle of different measures is needed, ranging from legal obligations to implement and perform meaningful oversight mechanisms to user training and awareness in order to counteract 'automation bias'. Furthermore, the AI Act proposal requires deployers to inform the provider or distributor and suspend the use of the system when they have identified any serious incidents or any malfunctioning [40, 41, Art. 29(4)].

Third, and *ex post*, providers must act and take the necessary corrective actions as soon as they become aware, e.g. through information provided by the deployer, that the high-risk system does not (or no longer) meet the legal requirements [40, 41, Art. 16(g)]. To this end, they must establish and document a system of monitoring that is proportionate to the type of AI technology and the risks of the high-risk AI system [40, 41, Art. 61(1)].

Fairness monitoring can be helpful in all three of the above respects. Therefore, we argue that there is even a legal obligation to use technical measures such as the method presented in this paper if this is the only way to ensure effective human oversight.

7 Conclusion and future work

This articles brings together software doping theory and probabilistic falsification techniques. To this end, it proposes a suitable HyperSTL semantics and characterises robust cleanness and func-cleanness as HyperSTL formulas and, for the special case of finite standard behaviour, STL formulas. Software doping techniques have been extensively applied to the tampered diesel emission cleaning systems; this article continues this path of research by demonstrating how testing of real cars can become more effective. For the first time, we apply software doping techniques to high-risk (AI) systems. We propose a runtime fairness monitor to promote effective human oversight of high-risk systems. The development of this monitor is complemented by an interdisciplinary evaluation from a psychological, philosophical, and legal perspective.

Limitations & Challenges

A challenge to those employing robust cleanness or func-cleanness analysis is the selection of suitable parameters, especially d_{In}, d_{Out}, and f or κ_{i} and κ_{o}. Because of their high degree of context sensitivity, there are no paradigmatic candidates for them that one can default to. Instead, they have to be carefully selected with the concrete system, the structure of input data and the situation of use in mind.

Reasonable choices for robust cleanness analysis of diesel emissions have been proposed in recent work [19, 21]. With respect to individual fairness analysis, potential systems to which FairnessAwareSystem or FairnessMonitor can be applied to are still too diverse to give recommendations for the contract parameters. Obviously, further technical limitations include that f, d_{In}, and d_{Out} must be computable.

With a particular regard to fairness analysis, we identify also non-technical limitations. As seen in Fig. 5, our fairness monitoring aims to uncover a particular kind of unfairness, namely individual unfairness that originates from within the system. This excludes all kinds of group unfairness as well as unfairness from sources other than the system. Another limitation is the human's competence to interpret the system outputs. Even though this is not a limitation that is inherent to our approach, it nevertheless will arguably be relevant in some practical cases, and an implementation of the monitoring always has to happen with the human in mind. For example, the design of the tool should avoid creating the false impression that the system is proven to be fair for an individual if no counterexample has been found. Interpretations like this could lead to inflated judgements of system trustworthiness and eventually to overtrusting system outputs [110, 112]. Also, it might be reasonable to limit access to the monitoring results: if individuals who are processed by the system have full access to their fairness analysis, they could use this to ‘game’ the system, i.e. they could use the synthetic inputs to slightly modify their own input such that they receive a better outcome. While more transparency for the user is generally desirable, this has to be kept in mind to avoid introducing new unfairness on a meta-level.

Future Work

The probabilistic falsification technique we use in this article can be seen as a modular framework that consists of several interchangeable components. One of these components is the optimisation technique used to find the input with minimal robustness value. Algorithm 1 uses a simulated annealing technique [29, 107], but other techniques have been proposed for temporal logic falsification, too [4, 108]. We want to further look into such alternative optimisation techniques and to evaluate if they offer benefits w.r.t. cleanness falsification.

Finally, the fairness monitoring approach has been presented using a toy example. It is not claimed to be readily applicable to real-life scenarios. Besides the future work that has already been mentioned throughout the paper, we are planning on various extensions of our approach, and are working on an implementation that will allow us to integrate the monitoring into a real system. Moreover, we plan to test the possible benefits and shortcomings of the approach in user studies where decision-makers are tasked to make hiring decisions with and without the fairness monitoring approach. Further work will encompass activities such as the improvement and embedding of the algorithm FairnessAwareSystem into a proper tool that can be used by non-computer-scientists, and the extension of the monitoring technique to cover more types of unfairness. For example, logging the output of the fairness monitor could be used to identify groups that are especially likely to be treated unfairly by the system: The individual fairness verdicts provided by FairnessAwareSystem and FairnessMonitor may also be logged and considered for further fairness assessments or other means of quality assurance of system P. Statistical analysis might unveil that individuals of certain groups are treated unfairly more frequently than individuals from other groups. Depending on the distinguishing features of the evaluated group, this can uncover problems in P, especially if protected attributes, such as gender, race, age, etc, are taken into account. Thereby, system fairness can be assessed for protected attributes without including them in the input of P, which should generally be avoided, and even without disclosing them to the human in the loop. By evaluating the monitoring logs from sufficiently many diverse runs of FairnessAwareSystem, our local method can be lifted such that it resembles a global method for many practical applications, i.e. we can make statistical statements about the general fairness of P. Such an evaluation can also be used to extract prototypes and counterexamples in the spirit of Been et al. [66] illustrating the *tendency* to judge unfairly. This is an interesting combination of individual and group fairness that we want to look into further. Other insights from the research on reactive systems [19, 21, 32] can potentially be used to further enrich the monitoring. Finally, various disciplines have to join forces to resolve highly interdisciplinary questions such as what constitutes reasonable and adequate choices for f, d_{In}, and d_{Out} in given contexts of application.

Author Contributions Not applicable

Funding Open Access funding enabled and organized by Projekt DEAL. This work is partially funded by DFG grant 389792660 as part of TRR 248—CPEC (see https://perspicuous-computing.science), by VolkswagenStiftung as part of grants AZ 98514, 98513 and 98512 EIS—Explainable Intelligent Systems (see https://explainable-intelligent.systems), and by the European Regional Development Fund (ERDF) and the Saarland as part of CERTAIN—Center for European Research in Trusted AI (see https://www.certain-trust.eu/). It has received support as part of STORM_SAFE, an Interreg project supported by the North Sea Programme of the European Regional Development Fund of the European Union.

Data Availability The datasets analysed during the current study are available in a Zenodo repository [15] (https://zenodo.org/record/8058770).

Declarations

Competing interests The authors have no competing interests to declare that are relevant to the content of this article.

Ethics approval Not applicable

Consent to participate Not applicable

Consent for publication Not applicable

Code availability Not applicable

Appendix A Technical Appendix

This appendix illustrates that func-fairness is more expressive than Lipschitz-fairness and why this is useful. For this, we use as a toy example a very simple, hypothetical HR scoring system that aggregates five scores given to the candidates. We remark that the whole scenario, the implementation of the system, the choice of distance functions and f, is likely not applicable for real-life situations; everything is picked so that our explanations are understandable.

Suppose that certain qualities and characteristics of the applicants are pre-scored by other systems on a scale from 0 to 100 %, where 0 means that the candidate is utterly unsuitable for the job in a certain regard, while a scoring of 100 % means that the candidate is perfect for the job in this regard. In particular, we will assume that the following marks are given to each applicant: an *education mark* for how well they are academically suitable for the job, an *experience mark* for how well their previous work experience fits the job, a *personality mark* for their personal and social skills, a *mental ability mark* for what is colloquially referred to as an applicant's general intelligence, and, finally, a *skill mark* that tracks the special skills that applicants have which might be beneficial for the job, such as their knowledge of foreign languages.

The system P that is of interest for us in this example is the one that aggregates all of these marks and gives out an overall score of how well the candidate is suited for the job.

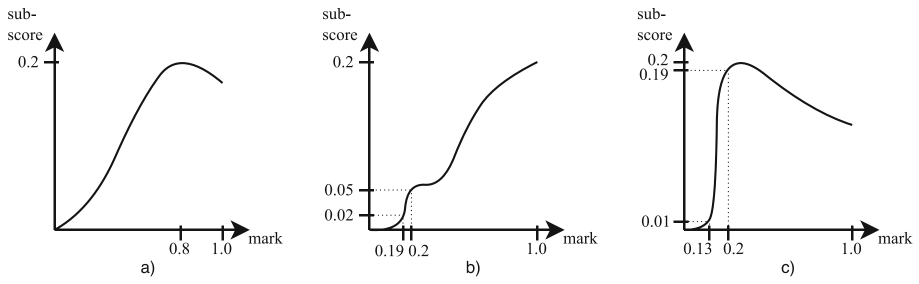

Fig. 8 Visualisation of subscoring functions mapping marks to subscores

The human responsible for the hiring process can use this in her hiring decision, e.g., she can focus on the top-scoring candidates and choose among them.

Let $\mathcal{M} = [0, 1] \subseteq \mathbb{R}$ be the reals between 0 and 1. Each of the five marks mentioned above is a real number from set $\mathcal{M}$. The input domain $\mathsf{In} = \mathcal{M}^5$ for the sketched HR system is a tuple of five marks. The output of the system is the overall suitability score of an applicant, which is also a value from $\mathcal{M}$. The distance between two inputs is defined as the euclidean distance, normalised to a value between 0 and 1, i.e.,

$$d_{\mathsf{In}}\big((\mathsf{ed}_1, \mathsf{ex}_1, \mathsf{pe}_1, \mathsf{in}_1, \mathsf{sk}_1), (\mathsf{ed}_2, \mathsf{ex}_2, \mathsf{pe}_2, \mathsf{in}_2, \mathsf{sk}_2)\big) =$$
$$\sqrt{\frac{(\mathsf{ed}_1 - \mathsf{ed}_2)^2 + (\mathsf{ex}_1 - \mathsf{ex}_2)^2 + (\mathsf{pe}_1 - \mathsf{pe}_2)^2 + (\mathsf{in}_1 - \mathsf{in}_2)^2 + (\mathsf{sk}_1 - \mathsf{sk}_2)^2}{5}},$$

where ed represents the education mark, ex the experience mark, pe the personality mark, in the mental ability mark, and sk the skill mark of an applicant. The distance between two outputs $d_{\mathsf{Out}}(\mathsf{o}_1, \mathsf{o}_2) = |\mathsf{o}_1 - \mathsf{o}_2|$ is the absolute difference between the overall scores o and o'. Note that also output distances are values between 0 and 1.

Our scoring system is a function $P : \mathcal{M}^5 \to \mathcal{M}$. We will assume here that P is defined as the sum of five subscoring systems, one for each of the five input marks, computing a value between 0 and 0.2. Then,

$$P((\mathsf{ed}, \mathsf{ex}, \mathsf{pe}, \mathsf{in}, \mathsf{sk})) := P_{\mathsf{ed}}(\mathsf{ed}) + P_{\mathsf{ex}}(\mathsf{ex}) + P_{\mathsf{pe}}(\mathsf{pe}) + P_{\mathsf{in}}(\mathsf{in}) + P_{\mathsf{sk}}(\mathsf{sk}).$$

Let P_{ed}, P_{ex}, P_{pe} and P_{in} be defined according to the plot shown in Fig. 8a. With an increasing mark, these subscores increases up to an input mark of 0.8, whereafter the applicant becomes overqualified and the subscore slowly decreases. P_{sk} is depicted in Fig. 8b: The skill mark is less important, however a minimum amount of skills is required for the job. Hence, there is a jump of the skill score at an skill mark of roughly 0.19. Let John be an applicant with $\mathsf{ed} = \mathsf{ex} = \mathsf{pe} = \mathsf{in} = 0.5$ and a skill mark of $\mathsf{sk} = 0.2$, which maps to a skill score on the plateau after the jump. The subscores for education, experience, personality and mental ability mark are 0.12 each. The skill score computed for John is 0.05. Hence, John's overall score is $P(\text{John}) = 4 \cdot 0.12 + 0.05 = 0.53$. Let Synthia be a synthetic applicant with the same marks as John, except for the skill mark, which is 0.19 in Synthia's case. As depicted in Fig. 8b, the skill subscore for skill mark 0.19 is 0.02—Synthia is at the plateau right before the jump of the skill score. Her overall score is $P(\text{Synthia}) = 4 \cdot 0.12 + 0.02 = 0.50$. The input distance between John and Synthia is $d_{\mathsf{In}}(\text{John, Synthia}) = \sqrt{\frac{0.01^2}{5}} \approx 0.0045$ and the output distance is $d_{\mathsf{Out}}(\text{John, Synthia}) = |0.53 - 0.5| = 0.03$. It is easy to see that if we use Lipschitz-fairness, the Lipschitz constant L must be at least $L = 6.7$ to allow the small jump in the skill subscoring function. We argue that small jumps like those in the skill subscore are normal behaviour and, hence, fair. Assume for the remainder of this example that we use Lipschitz-fairness with $L = 6.7$.

Consider now a slightly modified variant P' of P. P' is as P but uses a different subscoring function P'_{sk} for the skill score. Fig. 8c shows the skill subscoring function for P'. P'_{sk} has a jump at skill mark 0.13 that is significantly larger than that in P_{sk}. We assume in this example that such a big jump is unfair. This assumption is warranted since, for many applications, such a small change in technical skills which has an immense impact on the skill subscore is not reasonable. Considering applicant John, his skill mark still maps to a very high skill score of 0.19. Let Synclair be a third (potentially synthetic) applicant with $\mathsf{ed} = \mathsf{ex} = \mathsf{pe} = \mathsf{in} = 0.5$ (as for John and Synthia) and $\mathsf{sk} = 0.13$. Her skill mark maps to a very small skill score of 0.01. The overall scores are $P'(\text{John}) = 4 \cdot 0.12 + 0.19 = 0.67$ and $P'(\text{Synclair}) = 4 \cdot 0.12 + 0.01 = 0.49$. The input distance is $d_{\mathsf{In}}(\text{John, Synclair}) = 0.0313$ and the output distance is $d_{\mathsf{Out}}(\text{John, Synclair}) = 0.18$. Applying the Lipschitz condition to P' and $d_{\mathsf{In}}(\text{John, Synclair})$, it easy to see that $d_{\mathsf{Out}}(\text{John, Synclair})$ may become as large as 0.21. Hence, P' is classified as fair w.r.t. the Lipschitz condition. We see that a problem of the Lipschitz condition is that it is not possible to allow small jumps and at the same time disallow large jumps with equal increasing rate. This is because the distance of the inputs can only be used to multiply it with the Lipschitz constant.

$$f(d) = \begin{cases} 0.001 + 8d, & \text{for } d \in [0.0, 0.01] \\ 0.001 + 4d, & \text{for } d \in (0.01, 0.1] \\ 0.001 + 2d, & \text{for } d \in (0.1, 1.0] \end{cases}$$

Func-fairness is different in this regard. Function f receives the input distance and can freely define a bound on output distances based on the input distance. Indeed, the concrete f on the right overcomes the problem observed in the example. It uses the input distance for a case distinction on the magnitude of the input distance. For input distances up to 0.01, f effectively applies Lipschitz-fairness with $L = 8$ to allow small jumps. For input distances between 0.01 and 0.1, f behaves like Lipschitz-fairness for $L = 4$, and for larger input distances, it enforces $L = 2$. In all cases we add 0.001 to the result to avoid f becoming zero (see footnote 7 on page 27 in the main paper). Applying func-fairness with $\mathcal{C} = \langle d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ to P, the combination of John and Synthia (and hence the small jump of the skill score function) is not highlighted by $\mathsf{FairnessAwareSystem}$, i.e., it is correctly detected as func-fair. Applied to P', however, John and Synclair fall into the second case in the definition of f, but, as the emulated Lipschitz condition with $L = 4$ is violated, $\mathsf{FairnessAwareSystem}$ likely finds a negative fairness score, i.e., P' is not func-fair w.r.t. John. We remark that we propose this f for purely illustrative purposes. For real-world examples, f should be more sophisticated. Finding a suitable f can be a non-trivial task which hinges on various aspects that are crucial for the fairness evaluation in a given context. Clearly, the P and f provided in this illustration are toy examples that are probably inappropriate for real-world usage.

A.1 Proofs

In this section, we will provide proofs for most of the propositions and theorems in the main paper. First, we show the correctness of the HyperSTL characterisations of robust cleanness and func-cleanness.

We first provide a lemma, which destructs the globally ($\square$) and weak until ($\mathcal{W}$) operators such that the timing constraints encoded by these operators becomes explicit.

Lemma 8 *Let $\sigma : \mathcal{T} \to X$ be a trace with $\mathcal{T} = \mathbb{N}$ or $\mathcal{T} = \mathbb{R}_{\geq 0}$ and let ϕ and ψ be STL formulas. Then the following equivalences hold.*

1. *$\sigma, 0 \models \Box\phi$ if and only if $\forall t \geq 0.\ \sigma, t \models \phi$,*
2. *if $\mathcal{T} = \mathbb{N}$, then $\sigma, 0 \models \phi\, \mathcal{W}\, \psi$ if and only if $\forall t \geq 0.\ (\forall t' \leq t.\ \sigma, t' \models \neg\psi) \Rightarrow \sigma, t \models \phi$.*

Proof We prove the two statements separately.

1. Using the definition of the derived operators $\Box$ and $\Diamond$, we get that $\sigma, 0 \models \Box\phi$ holds if and only if $\sigma, 0 \models \neg(\top\, \mathcal{U}\, \neg\phi)$ holds. Using the (Boolean) semantics of STL, we get that this is equivalent to $\neg(\exists t \geq 0.\ \sigma, t \models \neg\phi \wedge \forall t' < t.\ \sigma, t' \models \top)$. After simple logical operations, we get that this is equivalent to $\forall t \geq 0.\ \sigma, t \models \phi$ as required.
2. Using , the definition of $\mathcal{W}$, the (Boolean) semantics of STL, and considering that $\mathcal{T} = \mathbb{N}$, we get that $\sigma, 0 \models \phi\, \mathcal{W}\, \psi$ if and only if $\exists t \in \mathbb{N}.\ \sigma, t \models \psi \wedge \forall t' < t.\ \sigma, t' \models \phi$ or $\forall t \in \mathbb{N}.\ \sigma, t \models \phi$. We denote this proposition as V. It is easy to see that the right operand of the equivalence to prove can be rewritten to $\forall t \in \mathbb{N}.\ (\exists t' \leq t.\ \sigma, t' \models \psi) \vee \sigma, t \models \phi$. We denote this proposition as W and must show that $V \Rightarrow W$ and $W \Rightarrow V$. To prove that V implies W, we distinguish two cases.

   - For the first case, assume that the left operand of the disjunction in V holds, i.e., there is some $t \in \mathbb{N}$, such that $\sigma, t \models \psi \wedge \forall t' < t.\ \sigma, t' \models \phi$. To show W, let $t_0 \in \mathbb{N}$ be arbitrary. If $t \leq t_0$, then there exists $t' \leq t_0$ (namely $t' = t$) such that $\sigma, t' \models \psi$; hence W holds. If $t > t_0$, then we know from $\forall t' < t.\ \sigma, t' \models \phi$ that $\sigma, t_0 \models \phi$ is true; hence, W holds.
   - For the second case, assume that the right operand of the disjunction in V holds, i.e., $\forall t \in \mathbb{N}.\ \sigma, t \models \phi$. Then, obviously W holds.

   To prove that W implies V, let $PV = \{t \in \mathbb{N} \mid \sigma, t \models \psi\}$ be the set of all time points at which ψ holds. If PV is the empty set, it follows immediately from W that $\forall t \in \mathbb{N}.\ \sigma, t \models \phi$ and that, hence, V holds. If PV is not empty, let $t = \min \mathsf{PV}$ be the smallest time in PV (the minimum always exists, because $\mathcal{T} = \mathbb{N}$). Then, obviously, $\exists t \in \mathbb{N}.\ \sigma, t \models \psi$. To show that V holds, it suffices to show that $\forall t' < t.\ \sigma, t' \models \phi$. This follows from W, because t is the smallest time at which $\sigma, t \models \psi$ holds and, therefore, for every $t' < t$ it does not hold that $\sigma, t' \models \psi$.

$\Box$

Lemma 9 is specific to the HyperSTL formula (3); it converts it into a first-order logic formula.

Lemma 9 *Let $\mathsf{M} \subseteq (\mathbb{N} \to X)$ be a discrete-time system and let $\mathsf{Std} \subseteq \mathsf{M}$ be a set of standard traces. Also, let Std_π be a quantifier-free HyperSTL subformula, such that $\mathsf{M}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_\pi$ if and only if $w \in \mathsf{Std}$. Then, $\mathsf{M}, \varnothing, 0 \models \psi_{u\text{-}rob}$ if and only if*

$$\begin{aligned}&\forall w \in \mathsf{Std}.\ \forall w' \in \mathsf{M}.\ \exists w'' \in \mathsf{Std}.\ (\forall t \geq 0.\ \mathsf{eq}(w{\downarrow_\mathsf{i}}[t], w''{\downarrow_\mathsf{i}}[t]) \leq 0) \wedge \\ &\quad \forall t \geq 0.\ (\forall t' \leq t.\ d_{\mathsf{In}}(w''{\downarrow_\mathsf{i}}[t'], w'{\downarrow_\mathsf{i}}[t']) - \kappa_\mathsf{i} \leq 0) \Rightarrow d_{\mathsf{Out}}(w''{\downarrow_\mathsf{o}}[t], w'{\downarrow_\mathsf{o}}[t]) - \kappa_\mathsf{o} \leq 0.\end{aligned}$$

Proof Using Lemma 8.1, Lemma 8.2, and Definition 6, we get that

$$\begin{aligned}&\mathsf{M}, \varnothing, 0 \models \forall\pi.\ \forall\pi'.\ \exists\pi''.\ \mathsf{Std}_\pi \\ &\quad \to \Big(\mathsf{Std}_{\pi''} \wedge \Box(\mathsf{eq}(\pi{\downarrow_\mathsf{i}}, \pi''{\downarrow_\mathsf{i}}) \leq 0) \wedge \\ &\qquad\qquad \big((d_{\mathsf{Out}}(\pi''{\downarrow_\mathsf{o}}, \pi'{\downarrow_\mathsf{o}}) - \kappa_\mathsf{o} \leq 0)\ \mathcal{W} (d_{\mathsf{In}}(\pi''{\downarrow_\mathsf{i}}, \pi'{\downarrow_\mathsf{i}}) - \kappa_\mathsf{i} > 0)\big)\Big)\end{aligned}$$

holds if and only if

$$\forall w \in \mathsf{M}.\ \forall w' \in \mathsf{M}.\ \exists w'' \in \mathsf{M}.\ (\mathsf{M}, \Pi, 0 \models \mathsf{Std}_{\pi})$$
$$\to \Big((\mathsf{M}, \Pi, 0 \models \mathsf{Std}_{\pi''}) \wedge (\forall t \geq 0.\ (\mathsf{M}, \Pi, t \models \mathsf{eq}(\pi{\downarrow_{\mathsf{i}}}, \pi''{\downarrow_{\mathsf{i}}}) \leq 0)) \wedge$$
$$(\forall t \geq 0.\ (\forall t' \leq t.\ (\mathsf{M}, \Pi, t' \models \neg d_{\mathsf{In}}(\pi''{\downarrow_{\mathsf{i}}}, \pi'{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0))$$
$$\Rightarrow (\mathsf{M}, \Pi, t \models d_{\mathsf{Out}}(\pi''{\downarrow_{\mathsf{o}}}, \pi'{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0))\Big)$$

holds for $\Pi = \{\pi{:=}w, \pi'{:=}w', \pi''{:=}w''\}$. Using the the constraint under which Std_{π} must be modelled, and by further applying Definition 6 and basic logical operations, we get that the above proposition is equivalent to

$$\forall w \in \mathsf{M}.\ \forall w' \in \mathsf{M}.\ \exists w'' \in \mathsf{M}.\ w \in \mathsf{Std}$$
$$\to \Big(w'' \in \mathsf{Std} \wedge (\forall t \geq 0.\ \mathsf{eq}(w{\downarrow_{\mathsf{i}}}[t], w''{\downarrow_{\mathsf{i}}}[t]) \leq 0) \wedge$$
$$(\forall t \geq 0.\ (\forall t' \leq t.\ d_{\mathsf{In}}(w''{\downarrow_{\mathsf{i}}}[t'], w'{\downarrow_{\mathsf{i}}}[t']) - \kappa_{\mathsf{i}} \leq 0) \Rightarrow d_{\mathsf{Out}}(w''{\downarrow_{\mathsf{o}}}, w'{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\Big).$$

Finally, after carefully reordering premises, we get that the above holds if and only if

$$\forall w \in \mathsf{Std}.\ \forall w' \in \mathsf{M}.\ \exists w'' \in \mathsf{Std}.\ (\forall t \geq 0.\ \mathsf{eq}(w{\downarrow_{\mathsf{i}}}[t], w''{\downarrow_{\mathsf{i}}}[t]) \leq 0) \wedge$$
$$\forall t \geq 0.\ (\forall t' \leq t.\ d_{\mathsf{In}}(w''{\downarrow_{\mathsf{i}}}[t'], w'{\downarrow_{\mathsf{i}}}[t']) - \kappa_{\mathsf{i}} \leq 0) \Rightarrow d_{\mathsf{Out}}(w''{\downarrow_{\mathsf{o}}}, w'{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0.$$

□

We omit the lemma analogue to Lemma 9 that reformulates formula (2) as a first-order characterisation. The proof for Proposition 3 further transforms the first-order characterisations of formulas (2) and (3) to show that they indeed match the definitions of l-robust cleanness and u-robust cleanness.

Proposition 3 Let $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ be a mixed-IO system and $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, \kappa_{\mathsf{i}}, \kappa_{\mathsf{o}} \rangle$ a contract or context for robust cleanness with $\mathsf{Std} \subseteq \mathsf{L}$. Further, let Std_{π} be a quantifier-free HyperSTL subformula, such that $\mathsf{L}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_{\pi}$ if and only if $w \in \mathsf{Std}$. Then, L is l-robustly clean w.r.t. $\mathcal{C}$ if and only if $\mathsf{L}, \varnothing, 0 \models \psi_{\text{l-rob}}$, and L is u-robustly clean w.r.t. $\mathcal{C}$ if and only if $\mathsf{L}, \varnothing, 0 \models \psi_{\text{u-rob}}$.

Proof We prove the correctness for l-robust cleanness and u-robust cleanness separately and begin with u-robust cleanness. Using Lemma 9, we get that

$$\mathsf{L}, \varnothing, 0 \models \forall \pi_1.\ \forall \pi_2.\ \exists \pi_1'.\ \mathsf{Std}_{\pi_1}$$
$$\to \Big(\mathsf{Std}_{\pi_1'} \wedge \Box(\mathsf{eq}(\pi_1{\downarrow_{\mathsf{i}}}, \pi_1'{\downarrow_{\mathsf{i}}}) \leq 0) \wedge$$
$$\big((d_{\mathsf{Out}}(\pi_1'{\downarrow_{\mathsf{o}}}, \pi_2{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\, \mathcal{W} (d_{\mathsf{In}}(\pi_1'{\downarrow_{\mathsf{i}}}, \pi_2{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)\big)\Big)$$

holds if and only if

$$\forall w_1 \in \mathsf{Std}.\ \forall w_2 \in \mathsf{L}.\ \exists w_1' \in \mathsf{Std}.\ (\forall t \geq 0.\ \mathsf{eq}(w_1{\downarrow_{\mathsf{i}}}[t], w_1'{\downarrow_{\mathsf{i}}}[t]) \leq 0) \wedge$$
$$\forall t \geq 0.\ (\forall t' \leq t.\ d_{\mathsf{In}}(w_1'{\downarrow_{\mathsf{i}}}[t'], w_2{\downarrow_{\mathsf{i}}}[t']) - \kappa_{\mathsf{i}} \leq 0) \Rightarrow d_{\mathsf{Out}}(w_1'{\downarrow_{\mathsf{o}}}, w_2{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0.$$

After applying simple logical operations and using that $\mathsf{eq}(\mathsf{i}_1, \mathsf{i}_2) = 0$ if and only if $\mathsf{i}_1 = \mathsf{i}_2$, we get that this is equivalent to

$$\forall w_1 \in \mathsf{Std}.\ \forall w_2 \in \mathsf{L}.\ \exists w_1' \in \mathsf{Std} \text{ with } w_1{\downarrow_{\mathsf{i}}} = w_1'{\downarrow_{\mathsf{i}}}.$$
$$(\forall t \geq 0.\ (\forall t' \leq t.\ d_{\mathsf{In}}(w_1'{\downarrow_{\mathsf{i}}}[t'], w_2{\downarrow_{\mathsf{i}}}[t']) \leq \kappa_{\mathsf{i}}) \Rightarrow d_{\mathsf{Out}}(w_1'{\downarrow_{\mathsf{o}}}[t], w_2{\downarrow_{\mathsf{o}}}[t]) \leq \kappa_{\mathsf{o}}),$$

which, since we assumed $\mathsf{Std} \subseteq \mathsf{L}$, is equivalent to the definition of u-robust cleanness for mixed-IO systems.

The proof for l-robust cleanness is analogue. □

We recapitulate the proposition similar to Proposition 3 for func-cleanness.

Proposition 4 Let $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ be a mixed-IO system and $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ a contract or context for func-cleanness with $\mathsf{Std} \subseteq \mathsf{L}$. Further, let Std_π be a quantifier-free HyperSTL subformula, such that $\mathsf{L}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_\pi$ if and only if $w \in \mathsf{Std}$. Then, L is l-func-clean w.r.t. $\mathcal{C}$ if and only if $\mathsf{L}, \varnothing, 0 \models \psi_{\text{l-fun}}$, and L is u-func-clean w.r.t. $\mathcal{C}$ if and only if $\mathsf{L}, \varnothing, 0 \models \psi_{\text{u-fun}}$.

The proof for Proposition 4 is conceptually similar to the one for Proposition 3. The only difference is that instead of the reasoning about the $\mathcal{W}$ construct, the globally enforced relation between output distances and the result of f must be proven equivalent in the HyperSTL formulas and func-cleanness. We omit the proofs here.

Correctness of STL characterisations

Next, we show the correctness of the STL characterisations, i.e., we will prove the correctness of Theorems 5 and 6. We do so by first establishing a connection between the HyperSTL and the STL characterisations.

Lemma 10 *Let* $\mathsf{M} \subseteq (\mathbb{N} \to X)$ *be a discrete-time system and let* $\mathsf{Std} = \{w_1, \ldots, w_c\} \subseteq \mathsf{M}$ *be a finite set of standard traces. Also, let* Std_π *be a quantifier-free HyperSTL subformula, such that* $\mathsf{M}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_\pi$ *if and only if* $w \in \mathsf{Std}$. *Then,* $\mathsf{M}, \varnothing, 0 \models \psi_{\textsf{u-rob}}$ *if and only if* $(\mathsf{M} \circ \mathsf{Std}) \models \varphi_{\textsf{u-rob}}$ *(with* $\varphi_{\textsf{u-rob}}$ *from Theorem 5).*

Proof Using Lemma 9 we get that

$$\begin{aligned}&\mathsf{M}, \varnothing, 0 \models \forall \pi'. \forall \pi''. \exists \pi'''. \mathsf{Std}_{\pi'}\\ &\quad \to \Big(\mathsf{Std}_{\pi'''} \wedge \Box(\mathsf{eq}(\pi'{\downarrow_{\mathsf{i}}}, \pi'''{\downarrow_{\mathsf{i}}}) \leq 0) \wedge \\ &\qquad\qquad ((d_{\mathsf{Out}}(\pi'''{\downarrow_{\mathsf{o}}}, \pi''{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\, \mathcal{W} (d_{\mathsf{In}}(\pi'''{\downarrow_{\mathsf{i}}}, \pi''{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0))\Big)\end{aligned}$$

holds if and only if

$$\begin{aligned}&\forall w' \in \mathsf{Std}. \forall w'' \in \mathsf{M}. \exists w''' \in \mathsf{Std}. (\forall t \geq 0. \mathsf{eq}(w'{\downarrow_{\mathsf{i}}}[t], w'''{\downarrow_{\mathsf{i}}}[t]) \leq 0) \wedge \\ &\forall t \geq 0. (\forall t' \leq t. d_{\mathsf{In}}(w'''{\downarrow_{\mathsf{i}}}[t'], w''{\downarrow_{\mathsf{i}}}[t']) - \kappa_{\mathsf{i}} \leq 0) \Rightarrow d_{\mathsf{Out}}(w'''{\downarrow_{\mathsf{o}}}, w''{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0.\end{aligned}$$

Since $\mathsf{Std} = \{w_1, \ldots, w_c\}$, we can replace the universal and existential quantifiers over Std by a conjunction, respectively disjunction, over the standard traces [105]. We instantiate the universal quantifier for w'' with w and get that

$$\begin{aligned}&\bigwedge_{1\leq a\leq c} \bigvee_{1\leq b\leq c} (\forall t \geq 0. \mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}[t], w_b{\downarrow_{\mathsf{i}}}[t]) \leq 0) \wedge \\ &\forall t \geq 0. (\forall t' \leq t. d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}[t'], w{\downarrow_{\mathsf{i}}}[t']) - \kappa_{\mathsf{i}} \leq 0) \Rightarrow d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0.\end{aligned}$$

From the Boolean semantics of STL and by replacing all traces w, respectively w_k, by the corresponding w_+-projections, we get the equivalent proposition

$$\begin{aligned}&\bigwedge_{1\leq a\leq c} \bigvee_{1\leq b\leq c} (\forall t \geq 0. (w_+, t \models \mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \leq 0)) \wedge \\ &\forall t \geq 0. (\forall t' \leq t. (w_+, t' \models \neg d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)) \Rightarrow (w_+, t \models d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0).\end{aligned}$$

With the Boolean semantics of STL and Lemma 8.1 and 8.2 we get the equivalent statement that

$$\begin{aligned}&w_+, 0 \models \bigwedge_{1\leq a\leq c} \bigvee_{1\leq b\leq c} (\Box(\mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \leq 0)) \wedge \\ &\big((d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \leq 0)\, \mathcal{W} (d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)\big).\end{aligned}$$

□

We are now able to prove Theorem 5.

Theorem 5 Let $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ be a mixed-IO system and $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ a context for func-cleanness with finite standard behaviour $\mathsf{Std} = \{w_1, \ldots, w_c\} \subseteq \mathsf{L}$. Then, L is u-func-clean w.r.t. $\mathcal{C}$ if and only if $(\mathsf{L} \circ \mathsf{Std}) \models \varphi_{\text{u-fun}}$, where

$$\varphi_{\text{u-fun}} := \bigwedge_{1 \le a \le c} \bigvee_{1 \le b \le c} \Big(\Box(\mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \le 0) \wedge \big(\Box(d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - f(d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}})) \le 0) \big) \Big).$$

Proof The theorem follows from Proposition 3 and Lemma 10. □

To prove Theorem 6, we establish the following lemma, which is analogue to Lemma 10, up to u-func-cleanness replacing u-robust cleanness.

Lemma 11 *Let* $\mathsf{M} \subseteq (\mathcal{T} \to X)$ *be a system and let* $\mathsf{Std} = \{w_1, \ldots, w_c\} \subseteq \mathsf{M}$ *be a finite set of standard traces. Also, let* Std_π *be a quantifier-free HyperSTL subformula, such that* $\mathsf{M}, \{\pi{:=}w\}, 0 \models \mathsf{Std}_\pi$ *if and only if* $w \in \mathsf{Std}$*. Then,* $\mathsf{M}, \varnothing, 0 \models \psi_{\text{u-fun}}$ *if and only if* $(\mathsf{M} \circ \mathsf{Std}) \models \varphi_{\text{u-fun}}$ *(with* $\varphi_{\text{u-fun}}$ *from Theorem 6).*

The proof for Lemma 11 is, up to the different reasoning for $\Box(d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - f(d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}})) \le 0)$ instead of $(d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - \kappa_{\mathsf{o}} \le 0) \,\mathcal{W}(d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}}) - \kappa_{\mathsf{i}} > 0)$, identical to that of Lemma 10. We omit it here.

Theorem 6 Let $\mathsf{L} \subseteq \mathbb{N} \to (\mathsf{In} \cup \mathsf{Out})$ be a mixed-IO system and $\mathcal{C} = \langle \mathsf{Std}, d_{\mathsf{In}}, d_{\mathsf{Out}}, f \rangle$ a context for func-cleanness with finite standard behaviour $\mathsf{Std} = \{w_1, \ldots, w_c\} \subseteq \mathsf{L}$. Then, L is u-func-clean w.r.t. $\mathcal{C}$ if and only if $(\mathsf{L} \circ \mathsf{Std}) \models \varphi_{\text{u-fun}}$, where

$$\varphi_{\text{u-fun}} := \bigwedge_{1 \le a \le c} \bigvee_{1 \le b \le c} \Big(\Box(\mathsf{eq}(w_a{\downarrow_{\mathsf{i}}}, w_b{\downarrow_{\mathsf{i}}}) \le 0) \wedge \big(\Box(d_{\mathsf{Out}}(w_b{\downarrow_{\mathsf{o}}}, w{\downarrow_{\mathsf{o}}}) - f(d_{\mathsf{In}}(w_b{\downarrow_{\mathsf{i}}}, w{\downarrow_{\mathsf{i}}})) \le 0) \big) \Big).$$

Proof The theorem follows from Proposition 4 and Lemma 11. □

Appendix B Fairness Pipeline

As explained in Section 2 in the main paper, it is important to recognise that there are many sources of unfairness [8]. Section Appendix B shows a more detailed version of Fig. 5 in the main paper. Not every technical measure is able to detect every kind of unfairness and eliminating one source of unfairness might not be sufficient to eliminate all unfairness.

World There can be unfairness in the world that leads to individuals already having worse (or better) starting conditions than others and subsequently have a lower (or higher) chance that the final decision is made in their favour. For example, an individual could be systematically excluded from certain societal resources (e.g., girls who are excluded from education in Afghanistan under the Taliban) which puts these individuals at a disadvantage.

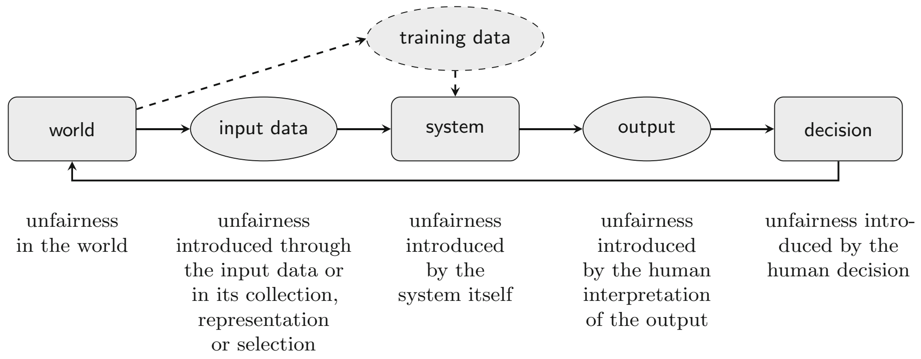
training data
world
input data
system
output
decision
unfairness in the world
unfairness introduced through the input data or in its collection, representation or selection
unfairness introduced by the system itself
unfairness introduced by the human interpretation of the output
unfairness introduced by the human decision

Fig. 9 Sketch of different origins of unfairness in a decision process supported by a system; dashed elements are inapplicable to systems that are not learning-based

Input data The input data or its collection, representation or selection could be problematic and lead to unfairness [139]. If, for example, crucial information is left out in the input data or data is aggregated in unsuitable ways, individuals could face an outcome that is unwarranted by the factual situation.

System (and training data) The system itself can introduce new unfairness. Among other things, this can come about by erroneous algorithms or (in the case of a trained model) by problematic training data, e.g., if a certain group of individuals is not properly represented [120].

Output The human decider can fail to interpret the output properly [138, 140], which can lead to further unfairness. They could, for example, lack knowledge of the limitations of the system or fail to take into account that the system output is subject to some systematic uncertainty.

Decision The human decider can make an unfair decision even in the face of a fair system output and an adequate interpretation thereof, for example if they have conscious or subconscious bias against certain groups [137].

Unfairness in any part of the chain can arguably perpetuate or reinforce unfairness in the world.

In the main paper, we propose a runtime monitoring technique that aims to uncover individual unfairness introduced by the system. By focusing on the system and its input-output relation only, we can say that the system is unfair without having to say anything about the degree of fairness with which an individual is treated in other respects in the decision process. It especially allows us to say that a system output is unfair, even though the outcome of the overall decision process is not. It may, for example, be that the system unfairness is 'cancelled out' by something else that is hidden from the system: an applicant with a stellar-looking CV might be treated unfairly by the system because of their age, but not hiring them is not unfair because they are known to have forged their diploma. Cases like this, however, do not make the unfairness introduced by the system any less problematic.

Appendix C Legal Appendix

C.1 EU anti-discrimination law

Antidiscrimination is a principle deeply rooted in EU law. It is enshrined in Art. 21 of the Charter of Fundamental Rights (CFR) [47], which prohibits '[a]ny discrimination based on any ground such as sex, race, colour, ethnic or social origin, genetic features, language, religion or belief, political or any other opinion, membership of a national minority, property, birth, disability, age or sexual orientation' as well as 'any discrimination based on grounds of nationality'. According to Art. 51 CFR, the addressees of this fundamental right are the EU and its institutions, bodies, offices and agencies as well as the Member States, insofar as they implement Union law. They are directly bound by Art. 21 above all in their legislative activities, but also in their executive and judicial measures. In contrast, private individuals are not directly bound by Art. 21 CFR, but they may be bound by regulations implementing this provision. However, according to recent European Court of Justice (ECJ) case law, Art. 21 CFR is directly applicable as a result of Directives, such as Directive 2000/78/EC [122] establishing a general framework for equal treatment in employment and occupation [45, §76]. Apart from this, while Art. 21 CFR stipulates a general prohibition of any unjustified discrimination, the more specific secondary legislation applicable to private actors only prohibits discrimination only in certain sensitive areas and only with regard to certain protected attributes. Correspondingly, private actors may not discriminate against certain persons—to name just a few—in employment relationships [122], in cases of abuse of a dominant market position [48, Art. 102] or also in so-called mass transactions, i.e., contracts that are typically concluded without regard to the person on comparable terms in a large number of cases [123]. In contrast, discriminating in other legal relationships or on other grounds such as local origin (as opposed to ethnic origin), or a person's financial situation is not generally prohibited. The rationale behind these 'discriminatory standards of anti-discrimination law' [57, 113, 126] is the principle of private (or personal) autonomy, and more specifically freedom of contract as one of its manifestations, which govern legal transactions between private individuals [75]. According to this principle, individuals are free to shape their legal relationships according to their own preferences and ideas, however irrational or socially unacceptable they may be. In essence, this also includes a right to discriminate against others. This freedom to autonomously form legal relations is only constrained where this is stipulated by anti-discrimination legislation for policy reasons.

When using an AI-system to recruit candidates, developers and deployers have to make sure that the system with its parameters comply with these legal requirements set by anti-discrimination law. This means in particular that the selection of the properties which a classifier should exhibit requires a positive normative choice and should not simply replicate biases present in the status quo [131]. However, the risks associated with deploying such systems in an HR context (such as a malfunctioning remaining undetected due to the system's opacity, a huge practical relevance of biased outputs due to the systems' scalability or the human operator's tendency of over-relying on the output produced by the AI system ('automation bias')), raise the question whether it can still be deemed normatively acceptable that the EU legal framework turns a blind eye on certain forms of discrimination. Furthermore, the principle of private autonomy as rationale for justifying the freedom to discriminate against others is only valid with regard to human's wilful actions, but not to algorithm-generated output. We are not advocating for abolishing the existing balance between private autonomy (freedom to contract) and prohibition to discriminate. So humans should still be permitted to differentiate on grounds that are not caught by anti-discrimination law. However, there is no reason to grant the 'right to discriminate' also to a non-human system that has merely "learned" this discrimination. In this respect, it seems justified to apply different standards for algorithms with regard to the prohibition of discrimination than for human decisions. With regard to an AI system's decision metrics, therefore, it should be considered to expand the secondary legal framework to include a broad prohibition of discrimination. This would not mean that all discrimination would be unlawful, since objectively justified unequal treatment is, after all, permissible, but it would shift the focus to the question of objective justification [46]. Another legal challenge that will become even more pressing with the advent of technical decision systems is how to detect and prove prohibited discrimination. This is because the prohibition of discrimination resulting from various legal regulations in certain, especially sensitive, areas, such as human resources, presupposes that a difference in treatment is recognised in the first place. The recognition of discrimination is therefore not only in the interest of the decision-maker, who is threatened with sanctions in the event of a violation of the prohibition of discrimination. Rather, it is also essential for the discriminated party to prove the discrimination. For as far as a legal claim follows from a prohibited discrimination, the principle applies that the person who invokes the legal claim must prove the facts giving rise to the claim. Especially when complex algorithms are used, however, it is likely to be extremely difficult to prove corresponding circumstantial evidence. According to the case law of the ECJ, however, the burden of proof is reversed if the party who has prima facie been discriminated against would otherwise have no effective means of enforcing the prohibition of discrimination [42, 43]. Monitoring, as described here, would therefore be a suitable means of providing the 'prima facie' evidence necessary for shifting the burden of proof.

C.2 Discrimination and the GDPR

There has recently been discussion if and to which extent data protection law contains obligations for non-discriminating data processing or whether the scope of protection of data protection law is thereby overstretched. There is no explicit prohibition of discrimination in the General Data Protection Regulation (GDPR). According to Article 1 (2), however, the GDPR is intended to protect the fundamental rights and freedoms of natural persons. This is aimed in particular at their right to protection of personal data (Article 8 CFR), but not exclusively so. Thus, the broad and non-restrictive reference to fundamental rights also encompasses all other fundamental rights, including the right to non-discrimination (Article 21 CFR) [39]. This is reflected, for example, in the higher level of protection for data with an increased potential for discrimination, the so-called special categories of personal data under Article 9 GDPR. The GDPR can also be interpreted as granting a "preventive protection against discrimination", namely when discrimination is made impossible from the outset, in that the data-processing agencies cannot gain knowledge of characteristics susceptible to discrimination in the first place, i.e., when any respective data processing is forbidden [26]. Any processing of personal data must furthermore comply with the processing principles set out in Article 5 GDPR, including the fairness principle ('personal data shall be processed fairly') set out in Article 5(1)(a). While formerly transparency obligations were read into this principle while the Data Protection Directive was into effect, the regulatory content of the fairness principle is highly disputed since it was split off into a separate processing principle. But due to the fact that discriminatory data processing can hardly be described as fair, a prohibition of discrimination can be linked to the fairness principle [56, 77]. However, the concrete scope of the fairness principle clearly goes beyond the understanding of fairness in the context of technical systems on which this paper is based.

Specifically for the HR context, there are discrimination-sensitive regulations in the GDPR. Article 9 GDPR makes the processing of special categories of data, i.e., sensitive data and data susceptible to discrimination, subject to particularly strict authorisation criteria, which should in practice rarely be present in recruitment situations. On the one hand, processing for recruitment purposes, i.e., prior to the establishment of an employment relationship, is rarely necessary in order to exercise certain rights and obligations under employment law (Art. 9(2)(b) GDPR), and on the other hand, explicit consent (Art. 9(2)(a) GDPR) will often lack the necessary voluntariness due to the specifics of job application situations and the power imbalances inherent in them. The prohibition of processing sensitive data may be problematic in cases where the link to sensitive data is strictly necessary to detect discriminatory effects. For high-risk systems, Art. 10 V AI Regulation Proposal therefore provides for a new permissive clause: 'To the extent that it is strictly necessary for the purposes of ensuring bias monitoring, detection and correction, ... the providers of such systems may process special categories of personal data' while ensuring appropriate safeguards for the fundamental rights of natural persons.

With regard to the processing of non-sensitive personal data, however, the opening clause in Art 88(1) GDPR allows Member States to adopt more specific rules for processing for recruitment purposes, whereby, according to paragraph 2, suitable and specific measures must be ensured to safeguard the fundamental rights of the data subject. These requirements can be met by state-of-the-art monitoring tools. The national regulations cannot be discussed in depth here. For Germany, for example, Section 26 of the Federal Data Protection Act (BDSG) stipulates that personal data may only be processed for recruitment purposes if this is necessary, i.e., if the data processing is required for the decision on recruitment. In any case, data processing may not be necessary if the characteristics depicted in the data may not be taken into account in the hiring decision, for example due to anti-discrimination law [103].

References

1. Abbas H, Fainekos GE, Sankaranarayanan S et al (2013) Probabilistic temporal logic falsification of cyber-physical systems. ACM Trans Embed Comput Syst 12(2):95:1-95:30. https://doi.org/10.1145/2465787.2465797
2. Alves WM, Rossi PH (1978) Who should get what? fairness judgments of the distribution of earnings. Am J Sociol 84(3):541–564
3. Angwin J, Larson J, Mattu S, et al (2016) Machine bias. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing
4. Annapureddy YSR, Fainekos GE (2010) Ant colonies for temporal logic falsification of hybrid systems. In: IECON 2010—36th annual conference on IEEE industrial electronics society, pp 91–96, https://doi.org/10.1109/IECON.2010.5675195
5. Arrieta AB, Díaz-Rodríguez N, Del Ser J et al (2020) Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI. Inf Fusion 58:82–115
6. Artistotle (1998a) The nicomachean ethics. Oxford worlds classics, Oxford University Press, Oxford, translation by W.D. Ross. Edition by John L. Ackrill, and James O. Urmson
7. Artistotle (1998b) Politics. Oxford worlds classics, Oxford University Press, Oxford, translation by Ernest Barker. Edition by R. F. Stalley
8. Barocas S, Selbst AD (2016) Big data's disparate impact. Calif L Rev 104:671
9. Barthe G, D'Argenio PR, Rezk T (2011) Secure information flow by self-composition. Math Struct Comput Sci 21(6):1207–1252. https://doi.org/10.1017/S0960129511000193
10. Barthe G, D'Argenio PR, Finkbeiner B, et al (2016) Facets of software doping. In: Margaria T, Steffen B (eds) Leveraging applications of formal methods, verification and validation: discussion, dissemination,

applications—7th international symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10-14, 2016, Proceedings, Part II, pp 601–608, https://doi.org/10.1007/978-3-319-47169-3_46
11. Bathaee Y (2017) The artificial intelligence black box and the failure of intent and causation. Harvard J Law Tech 31:889
12. Baum D, Baum K, Gros TP, et al (2023) XAI requirements in smart production processes: a case study. In: World conference on explainable artificial intelligence. Springer, pp 3–24
13. Baum K (2016) What the hack is wrong with software doping? In: Margaria T, Steffen B (eds) Leveraging applications of formal methods, verification and validation: discussion, dissemination, applications–7th international symposium, ISoLA 2016, Imperial, Corfu, Greece, October 10-14, 2016, Proceedings, Part II, pp 633–647, https://doi.org/10.1007/978-3-319-47169-3_49,
14. Baum K, Mantel S, Schmidt E et al (2022) From responsibility to reason-giving explainable artificial intelligence. Philos Tech 35(1):12. https://doi.org/10.1007/s13347-022-00510-w
15. Biewer S (2023). Real driving emissions tests records. https://doi.org/10.5281/zenodo.8058770
16. Biewer S (2023b) Software doping—theory and detection. Dissertation. https://doi.org/10.22028/D291-40364
17. Biewer S, Hermanns H (2022) On the detection of doped software by falsification. In: Johnsen EB, Wimmer M (eds) Fundamental approaches to software engineering—25th international conference, FASE 2022, Held as Part of the European joint conferences on theory and practice of software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings, Lecture Notes in Computer Science, vol 13241. Springer, pp 71–91, https://doi.org/10.1007/978-3-030-99429-7_4,
18. Biewer S, D'Argenio PR, Hermanns H (2019) Doping tests for cyber-physical systems. In: Parker D, Wolf V (eds) Quantitative evaluation of systems, 16th international conference, QEST 2019, Glasgow, UK, September 10-12, 2019, proceedings, lecture notes in computer science, vol 11785. Springer, pp 313–331, https://doi.org/10.1007/978-3-030-30281-8_18,
19. Biewer S, D'Argenio PR, Hermanns H (2021) Doping tests for cyber-physical systems. ACM Trans Model Comput Simul 31(3):161–1627. https://doi.org/10.1145/3449354
20. Biewer S, Finkbeiner B, Hermanns H, et al (2021b) RTLOLA on board: testing real driving emissions on your phone. In: Groote JF, Larsen KG (eds) Tools and algorithms for the construction and analysis of systems—27th international conference, TACAS 2021, Held as Part of the European joint conferences on theory and practice of software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings, Part II, Lecture Notes in Computer Science, vol 12652. Springer, pp 365–372, https://doi.org/10.1007/978-3-030-72013-1_20
21. Biewer S, Dimitrova R, Fries M, et al (2022) Conformance relations and hyperproperties for doping detection in time and space. Log Methods Comput Sci. https://doi.org/10.46298/lmcs-18(1:14)2022,
22. Biewer S, Finkbeiner B, Hermanns H et al (2023) On the road with rtlola. Int J Softw Tools Technol Transf 25(2):205–218. https://doi.org/10.1007/s10009-022-00689-5
23. Binns R (2020) On the apparent conflict between individual and group fairness. In: Proceedings of the 2020 conference on fairness, accountability, and transparency. Association for computing machinery, New York, FAT* '20, pp 514–524, https://doi.org/10.1145/3351095.3372864,
24. Bloem R, Chatterjee K, Greimel K et al (2014) Synthesizing robust systems. Acta Inf 51(3–4):193–220. https://doi.org/10.1007/s00236-013-0191-5
25. Borgesius FJZ (2020) Strengthening legal protection against discrimination by algorithms and artificial intelligence. Int J Human Rights 24(10):1572–1593. https://doi.org/10.1080/13642987.2020.1743976
26. Buchner B (2020) DS-GVO Art. 1 Gegenstand und Ziele Rn. 14. In: Buchner JK (ed) Datenschutz-Grundverordnung, Bundesdatenschutzgesetz. C.H. Beck, Munich
27. Burke L (2020) The death and life of an admissions algorithm. https://www.insidehighered.com/admissions/article/2020/12/14/u-texas-will-stop-using-controversial-algorithm-evaluate-phd
28. Chazette L, Brunotte W, Speith T (2021) Exploring explainability: a definition, a model, and a knowledge catalogue. In: 2021 IEEE 29th international requirements engineering conference (RE), pp 197–208, https://doi.org/10.1109/RE51729.2021.00025
29. Chib S, Greenberg E (1995) Understanding the metropolis-hastings algorithm. Am Stat 49(4):327–335. https://doi.org/10.1080/00031305.1995.10476177
30. Chouldechova A (2017) Fair prediction with disparate impact: a study of bias in recidivism prediction instruments. Big Data 5(2):153–163. https://doi.org/10.1089/big.2016.0047
31. Clarkson MR, Finkbeiner B, Koleini M, et al (2014) Temporal logics for hyperproperties. In: Principles of security and trust—third international conference, POST 2014, Held as Part of the European joint conferences on theory and practice of software, ETAPS 2014, Grenoble, France, April 5-13, 2014, Proceedings, LNCS, vol 8414. Springer, pp 265–284, https://doi.org/10.1007/978-3-642-54792-8_15
32. D'Argenio PR, Barthe G, Biewer S, et al (2017) Is your software on dope? - formal analysis of surreptitiously "enhanced" programs. In: Yang H (ed) Programming Languages and Systems - 26th European

Symposium on Programming, ESOP 2017, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2017, Uppsala, Sweden, April 22-29, 2017, Proceedings, Lecture Notes in Computer Science, vol 10201. Springer, pp 83–110, https://doi.org/10.1007/978-3-662-54434-1_4,

33. Donzé A, Ferrère T, Maler O (2013) Efficient robust monitoring for STL. In: Sharygina N, Veith H (eds) Computer aided verification—proceedings of 25th international conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Lecture Notes in Computer Science, vol 8044. Springer, pp 264–279, https://doi.org/10.1007/978-3-642-39799-8_19
34. Dressel J, Farid H (2018) The accuracy, fairness, and limits of predicting recidivism. Sci Adv 4(1):eaao5580
35. Dwork C, Hardt M, Pitassi T, et al (2012) Fairness through awareness. In: Proceedings of the 3rd innovations in theoretical computer science conference, pp 214–226
36. Dworkin R (1981) What is equality? Part 2: equality of resources. Philos Public Aff 10(4):283–345
37. Endsley MR (1995) Toward a theory of situation awareness in dynamic systems. Hum Factors 37(1):32–64. https://doi.org/10.1518/001872095779049543
38. Endsley MR (2017) From here to autonomy: lessons learned from human-automation research. Hum Factors 59(1):5–27. https://doi.org/10.1177/0018720816681350
39. European Commission (2011) Proposal for a regulation of the European parliament and of the council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (general data protection regulation) /* com/2012/011 final. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A52012PC0011
40. European Commission (2021) Laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts (proposal for a regulation) no 0106/2021. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A52021PC0206
41. European Commission (2023) Amendments adopted by the european parliament on 14 june 2023 on the proposal for a regulation of the european parliament and of the council on laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts. https://www.europarl.europa.eu/doceo/document/TA-9-2023-0236_EN.html
42. European Court of Justice (1993) C-127/92 - enderby ecli:eu:c:1993:859. https://curia.europa.eu/juris/liste.jsf?language=en&num=C-127/92
43. European Court of Justice (1995) C-400/93 - royal copenhagen ecli:eu:c:195:155. https://curia.europa.eu/juris/liste.jsf?language=en&num=C-400/93
44. European Court of Justice (2014) C-356/12 - glatzel ecli:eu:c:2014:350. https://curia.europa.eu/juris/liste.jsf?language=en&num=C-356/12
45. European Court of Justice (2018) C-414/16 - egenberger ecli:eu:c:2018:257. https://curia.europa.eu/juris/liste.jsf?language=en&num=C-414/16
46. European Parliament (2020) European parliament resolution of 20 october 2020 with recommendations to the commission on a framework of ethical aspects of artificial intelligence, robotics and related technologies. https://www.europarl.europa.eu/doceo/document/TA-9-2020-0275_EN.html
47. European Union (2016a) Charter of fundamental rights of the european union. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A12012P%2FTXT
48. European Union (2016b) Consolidated version of the treaty on the functioning of the european union. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%3A12016ME%2FTXT
49. Fainekos GE, Pappas GJ (2009) Robustness of temporal logic specifications for continuous-time signals. Theor Comput Sci 410(42):4262–4291. https://doi.org/10.1016/j.tcs.2009.06.021
50. Ferrer X, Tv N, Such JM et al (2021) Bias and discrimination in AI: a cross-disciplinary perspective. IEEE Technol Soc Mag 40(2):72–80. https://doi.org/10.1109/MTS.2021.3056293
51. Finkbeiner B, Rabe MN, Sánchez C (2015) Algorithms for model checking HyperLTL and HyperCTL*. In: CAV 2015, LNCS, vol 9206. Springer, pp 30–48, https://doi.org/10.1007/978-3-319-21690-4_3
52. Friedler SA, Scheidegger C, Venkatasubramanian S (2021) The (im)possibility of fairness: different value systems require different mechanisms for fair decision making. Commun ACM 64(4):136–143. https://doi.org/10.1145/3433949
53. Gazda M, Mousavi MR (2020) Logical characterisation of hybrid conformance. In: Czumaj A, Dawar A, Merelli E (eds) 47th international colloquium on automata, languages, and programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference), LIPIcs, vol 168. Schloss Dagstuhl—Leibniz-Zentrum für Informatik, pp 130:1–130:18, https://doi.org/10.4230/LIPIcs.ICALP.2020.130,
54. Gunning D (2016) Explainable artificial intelligence (XAI) (darpa-baa-16-53). Tech. rep, Arlington, VA, USA
55. Guryan J, Charles KK (2013) taste-based or statistical discrimination: the economics of discrimination returns to its roots. Econ J 123(572):F417–F432. http://www.jstor.org/stable/42919257

56. Hacker P (2018) Teaching fairness to artificial intelligence: existing and novel strategies against algorithmic discrimination under EU law. Common Market Law Rev (55):1143–1186. https://ssrn.com/abstract=3164973
57. Hartmann F (2006) Diskriminierung durch Antidiskriminierungsrecht? Möglichkeiten und Grenzen eines postkategorialen Diskriminierungsschutzes in der Europäischen Union. EuZA - Europäische Zeitschrift für Arbeitsrecht p 24
58. Heaven WD (2020) Predictive policing algorithms are racist. They need to be dismantled. https://www.technologyreview.com/2020/07/17/1005396/predictive-policing-algorithms-racist-dismantled-machine-learning-bias-criminal-justice/
59. High-Level Expert Group on Artificial Intelligence (2019) Ethics Guidelines for Trustworthy AI. https://digital-strategy.ec.europa.eu/en/library/ethics-guidelines-trustworthy-ai
60. Hough LM, Oswald FL, Ployhart RE (2001) Determinants, detection and amelioration of adverse impact in personnel selection procedures: issues, evidence and lessons learned. Int J Sel Assess 9(1–2):152–194
61. Ilvento C (2019) Metric learning for individual fairness. arXiv:1906.00250
62. Jacovi A, Marasović A, Miller T, et al (2021) Formalizing trust in artificial intelligence: prerequisites, causes and goals of human trust in AI. In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency, pp 624–635
63. Jewson N, Mason D (1986) Modes of discrimination in the recruitment process: formalisation, fairness and efficiency. Sociology 20(1):43–63
64. John PG, Vijaykeerthy D, Saha D (2020) Verifying individual fairness in machine learning models. In: Adams RP, Gogate V (eds) Proceedings of the thirty-sixth conference on uncertainty in artificial intelligence, UAI 2020, virtual online, August 3-6, 2020, Proceedings of machine learning research, vol 124. AUAI Press, pp 749–758, http://proceedings.mlr.press/v124/george-john20a.html
65. Kästner L, Langer M, Lazar V, et al (2021) On the relation of trust and explainability: Why to engineer for trustworthiness. In: Yue T, Mirakhorli M (eds) 29th IEEE international requirements engineering conference workshops, RE 2021 workshops, Notre Dame, IN, USA, September 20-24, 2021. IEEE, pp 169–175, https://doi.org/10.1109/REW53955.2021.00031,
66. Kim B, Khanna R, Koyejo O (2016) Examples are not enough, learn to criticize! criticism for interpretability. In: Proceedings of the 30th international conference on neural information processing systems. Curran Associates Inc., Red Hook, NIPS'16, pp 2288–2296
67. Köhl MA, Hermanns H, Biewer S (2018) Efficient monitoring of real driving emissions. In: Colombo C, Leucker M (eds) Runtime Verification—Proceedings of 18th international conference, RV 2018, Limassol, Cyprus, November 10-13, 2018, Lecture Notes in Computer Science, vol 11237. Springer, pp 299–315, https://doi.org/10.1007/978-3-030-03769-7_17
68. Lai V, Tan C (2019) On human predictions with explanations and predictions of machine learning models: a case study on deception detection. In: Proceedings of the conference on fairness, accountability, and transparency, pp 29–38
69. Langer M, Baum K, Hartmann K, et al (2021a) Explainability auditing for intelligent systems: a rationale for multi-disciplinary perspectives. In: Yue T, Mirakhorli M (eds) 29th IEEE international requirements engineering conference workshops, RE 2021 workshops, Notre Dame, IN, USA, September 20-24, 2021. IEEE, pp 164–168, https://doi.org/10.1109/REW53955.2021.00030,
70. Langer M, Oster D, Speith T et al (2021) 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(103):473. https://doi.org/10.1016/j.artint.2021.103473
71. Langer M, Baum K, Schlicker N (2023) A signal detection perspective on error and unfairness detection as a critical aspect of human oversight of ai-based systems https://doi.org/10.31234/osf.io/ke256
72. Larson J, Mattu S, Kirchner L, et al (2016) How we analyzed the COMPAS recidivism algorithm. https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm
73. Lee JD, See KA (2004) Trust in automation: designing for appropriate reliance. Hum Factors 46(1):50–80
74. Linardatos P, Papastefanopoulos V, Kotsiantis S (2021) Explainable AI: a review of machine learning interpretability methods. Entropy. https://doi.org/10.3390/e23010018
75. Looschelders D (2012) Diskriminierung und Schutz vor Diskriminierung im Privatrecht. JZ - Juristen-Zeitung p 105
76. Maler O, Nickovic D (2004) Monitoring temporal properties of continuous signals. In: Lakhnech Y, Yovine S (eds) Formal techniques, modelling and analysis of timed and fault-tolerant systems, joint international conferences on formal modelling and analysis of timed systems, FORMATS 2004 and Formal Techniques in Real-Time and Fault-Tolerant Systems, FTRTFT 2004, Grenoble, France, September 22-24, 2004, Proceedings, Lecture Notes in Computer Science, vol 3253. Springer, pp 152–166, https://doi.org/10.1007/978-3-540-30206-3_12

77. Malgieri G (2020) What “fairness” means? A linguistic and contextual interpretation from the GDPR. In: FAT* ’20: conference on fairness, accountability, and transparency, Barcelona, Spain, January 27-30, 2020. ACM, pp 154–166, https://doi.org/10.1145/3351095.3372868,
78. Mathews M (2023) Are you ready for software-defined everything? Wired, https://www.wired.com/insights/2013/05/are-you-ready-for-software-defined-everything/, Accessed 23 June 2023
79. Matthias A (2004) The responsibility gap: ascribing responsibility for the actions of learning automata. Ethics Inf Technol 6(3):175–183. https://doi.org/10.1007/s10676-004-3422-1
80. Mecacci G, de Sio FS (2020) Meaningful human control as reason-responsiveness: the case of dual-mode vehicles. Ethics Inf Technol 22(2):103–115. https://doi.org/10.1007/s10676-019-09519-w
81. Mehrabi N, Morstatter F, Saxena N et al (2021) A survey on bias and fairness in machine learning. ACM Comput Surv 54(6):1–35
82. Meinke K, Sindhu MA (2011) Incremental learning-based testing for reactive systems. In: Gogolla M, Wolff B (eds) Tests and proofs—proceedings of 5th international conference, TAP@TOOLS 2011, Zurich, Switzerland, June 30–July 1, 2011. Lecture Notes in Computer Science, vol 6706. Springer, pp 134–151, https://doi.org/10.1007/978-3-642-21768-5_11
83. Methnani L, Aler Tubella A, Dignum V et al (2021) Let me take over: variable autonomy for meaningful human control. Front Artific Intell. https://doi.org/10.3389/frai.2021.737072
84. Meurrens S (2021) The increasing role of AI in visa processing. https://canadianimmigrant.ca/immigrate/immigration-law/the-increasing-role-of-ai-in-visa-processing
85. Mittelstadt BD, Allo P, Taddeo M et al (2016) The ethics of algorithms: mapping the debate. Big Data Soc 3(2):2053951716679679. https://doi.org/10.1177/2053951716679679
86. Molnar C, Casalicchio G, Bischl B (2020) Interpretable machine learning—a brief history, state-of-the-art and challenges. In: Koprinska I, Kamp M, Appice A, et al (eds) ECML PKDD 2020 workshops—workshops of the European conference on machine learning and knowledge discovery in databases (ECML PKDD 2020): SoGood 2020, PDFL 2020, MLCS 2020, NFMCP 2020, DINA 2020, EDML 2020, XKDD 2020 and INRA 2020, Ghent, Belgium, September 14-18, 2020, Proceedings, Communications in Computer and Information Science, vol 1323. Springer, pp 417–431, https://doi.org/10.1007/978-3-030-65965-3_28,
87. Mukherjee D, Yurochkin M, Banerjee M, et al (2020) Two simple ways to learn individual fairness metrics from data. In: III HD, Singh A (eds) Proceedings of the 37th international conference on machine learning, proceedings of machine learning research, vol 119. PMLR, pp 7097–7107, https://proceedings.mlr.press/v119/mukherjee20a.html
88. Nghiem T, Sankaranarayanan S, Fainekos GE, et al (2010) Monte-carlo techniques for falsification of temporal properties of non-linear hybrid systems. In: Johansson KH, Yi W (eds) Proceedings of the 13th ACM international conference on hybrid systems: computation and control, HSCC 2010, Stockholm, Sweden, April 12-15, 2010. ACM, pp 211–220, https://doi.org/10.1145/1755952.1755983
89. Nguyen LV, Kapinski J, Jin X, et al (2017) Hyperproperties of real-valued signals. In: Talpin J, Derler P, Schneider K (eds) Proceedings of the 15th ACM-IEEE international conference on formal methods and models for system design, MEMOCODE 2017, Vienna, Austria, September 29 - October 02, 2017. ACM, pp 104–113, https://doi.org/10.1145/3127041.3127058
90. Noorman M (2020) Computing and Moral Responsibility. In: Zalta EN (ed) The stanford encyclopedia of philosophy, Spring, 2020th edn. Stanford University, Metaphysics Research Lab
91. Nunes I, Jannach D (2017) A systematic review and taxonomy of explanations in decision support and recommender systems. User Model User-Adap Inter 27(3):393–444
92. O’Neil C (2016a) How algorithms rule our working lives. https://www.theguardian.com/science/2016/sep/01/how-algorithms-rule-our-working-lives, Accessed 23 June 2023
93. O’Neil C (2016) Weapons of math destruction: how big data increases inequality and threatens democracy. Crown Publishing Group, USA
94. Orcale (2019) AI in human resources: The time is now. https://www.oracle.com/a/ocom/docs/applications/hcm/oracle-ai-in-hr-wp.pdf
95. Organisation for Economic Co-operation and Development (OECD) (2021) Artificial intelligence, machine learning and big data in finance: opportunities, challenges and implications for policy makers. https://www.oecd.org/finance/financial-markets/Artificial-intelligence-machine-learning-big-data-in-finance.pdf
96. Pessach D, Shmueli E (2022) A review on fairness in machine learning. ACM Comput Surv. https://doi.org/10.1145/3494672
97. Pnueli A (1977) The temporal logic of programs. In: 18th annual symposium on foundations of computer science, Providence, Rhode Island, USA, 31 October–1 November 1977. IEEE Computer Society, pp 46–57, https://doi.org/10.1109/SFCS.1977.32

98. Rawls J (1985) Justice as fairness: Political not metaphysical. Philos Public Affairs 14(3):223–251. http://www.jstor.org/stable/2265349
99. Rawls J (1999) A theory of justice: Revised edition. Harvard university press
100. Rawls J (2001) Justice as fairness: a restatement. Harvard University Press
101. Ribeiro MT, Singh S, Guestrin C (2016a) Model-agnostic interpretability of machine learning. abs/1606.05386. arxiv:1606.05386
102. Ribeiro MT, Singh S, Guestrin C (2016b) "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. Association for computing machinery, New York, KDD '16, pp 1135–1144, https://doi.org/10.1145/2939672.2939778,
103. Riesenhuber K (2021) BDSG §26 Datenverarbeitung für Zwecke des Beschäftigungsverhältnisses Rn. 79f. In: Wolff SBA (ed) BeckOK Datenschutzrecht. C.H. Beck, Munich
104. Rockafellar RT, Wets RJB (2009) Variational analysis, vol 317. Springer Science & Business Media
105. Rosen KH, Krithivasan K (2012) Discrete mathematics and its applications: with combinatorics and graph theory. Tata McGraw-Hill Education
106. Rowe T (2022) Can a risk of harm itself be a harm? Analysis 81(4):694–701. https://doi.org/10.1093/analys/anab033
107. Rubinstein RY (1981) Simulation and the Monte Carlo method. Wiley series in probability and mathematical statistics, Wiley https://www.worldcat.org/oclc/07275104
108. Sankaranarayanan S, Fainekos G (2012) Falsification of temporal properties of hybrid systems using the cross-entropy method. In: Dang T, Mitchell IM (eds) Hybrid systems: computation and control (part of CPS Week 2012), HSCC'12, Beijing, China, April 17-19, 2012. ACM, pp 125–134, https://doi.org/10.1145/2185632.2185653,
109. Sanneman L, Shah JA (2020) A situation awareness-based framework for design and evaluation of explainable AI. International workshop on explainable. Springer, Transparent Autonomous Agents and Multi-Agent Systems, pp 94–110
110. Schlicker N, Langer M (2021) Towards warranted trust: a model on the relation between actual and perceived system trustworthiness. Mensch Comput 2021:325–329
111. Schlicker N, Langer M, Ötting SK et al (2021) What to expect from opening up black boxes? comparing perceptions of justice between human and automated agents. Comput Hum Behav 122(106):837. https://doi.org/10.1016/j.chb.2021.106837
112. Schlicker N, Uhde A, Baum K, et al (2022) Calibrated trust as a result of accurate trustworthiness assessment—introducing the trustworthiness assessment model. https://doi.org/10.31234/osf.io/qhwvx
113. Schwab D (2006) Schranken der Vertragsfreiheit durch die Antidiskriminierungsrichtlinien und ihre Umsetzung in Deutschland. DNotZ—Deutsche Notar-Zeitschrift
114. Santoni de Sio F, van den Hoven J (2018) Meaningful human control over autonomous systems: a philosophical account. Frontiers in Robotics and AI 5. https://doi.org/10.3389/frobt.2018.00015https://www.frontiersin.org/article/10.3389/frobt.2018.00015
115. Smith E, Vogell H (2021) How your shadow credit score could decide whether you get an apartment. https://www.propublica.org/article/how-your-shadow-credit-score-could-decide-whether-you-get-an-apartment, Accessed 23 June 2023
116. Speith T (2022) A review of taxonomies of explainable artificial intelligence (XAI) methods. In: 2022 ACM conference on fairness, accountability, and transparency. Association for computing machinery, New York, FAccT '22, pp 2239–2250, https://doi.org/10.1145/3531146.3534639,
117. Sterz S, Baum K, Lauber-Rönsberg A, et al (2021) Towards perspicuity requirements. In: Yue T, Mirakhorli M (eds) 29th IEEE international requirements engineering conference workshops, RE 2021 Workshops, Notre Dame, IN, USA, September 20-24, 2021. IEEE, pp 159–163, https://doi.org/10.1109/REW53955.2021.00029,
118. Tabuada P, Balkan A, Caliskan SY, et al (2012) Input-output robustness for discrete systems. In: Proceedings of the 12th International Conference on Embedded Software, EMSOFT 2012, part of the eighth embedded systems week, ESWeek 2012, Tampere, Finland, October 7-12, 2012. ACM, pp 217–226, https://doi.org/10.1145/2380356.2380396
119. Talbert M (2019) Moral responsibility. In: Zalta EN (ed) The stanford encyclopedia of philosophy, Winter, 2019th edn. Stanford University, Metaphysics Research Lab
120. Tay L, Woo SE, Hickman L et al (2022) A conceptual framework for investigating and mitigating machine-learning measurement bias (mlmb) in psychological assessment. Adv Methods Pract Psychol Sci. https://doi.org/10.1177/25152459211061337
121. Technavio (2022) Software defined everything (SDE) market by end-user and geography—forecast and analysis 2022-2026. https://www.technavio.com/report/software-defined-everything-sde-market-industry-analysis, Accessed 23 June 2023

122. The Council of the European Union (2000) Council directive 2000/78/EC of 27 november 2000 establishing a general framework for equal treatment in employment and occupation. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:32000L0078
123. The Council of the European Union (2004) Council directive 2004/113/EC of 13 december 2004 implementing the principle of equal treatment between men and women in the access to and supply of goods and services. https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32004L0113
124. The European Parliament and the Council of the European Union (2017) Commission Regulation (EU) 2017/1151. http://data.europa.eu/eli/reg/2017/1151/oj
125. Thüsing G (2013) European Labour Law, §3 Protection against discrimination. C.H, Beck
126. Thüsing G (2019) Das künftige Anti-Diskriminierungsrecht als Herausforderung für Wissenschaft und Praxis. ZfA - Zeitschrift für Arbeitsrecht p 241
127. Tutuianu M, Bonnel P, Ciuffo B et al (2015) Development of the world-wide harmonized light duty test cycle (WLTC) and a possible pathway for its introduction in the european legislation. Trans Res Part D Trans Environ 40(Suppl C):61–75. https://doi.org/10.1016/j.trd.2015.07.011
128. United Nations (2013) UN Vehicle Regulations—1958 Agreement, Revision 2, Addendum 100, Regulation No. 101, Revision 3—E/ECE/324/Rev.2/Add.100/Rev.3. http://www.unece.org/trans/main/wp29/wp29regs101-120.html
129. United Nations Educational, Scientific and Cultural Organization (UNESCO) (2021) Recommendation on the ethics of artificial intelligence. https://unesdoc.unesco.org/ark:/48223/pf0000380455
130. Volpato M, Tretmans J (2015) Approximate active learning of nondeterministic input output transition systems. Electron Commun Eur Assoc Softw Sci Technol 72. https://doi.org/10.14279/tuj.eceasst.72.1008
131. Wachter S, Mittelstadt B, Russell C (2020) Bias preservation in machine learning: the legality of fairness metrics under eu non-discrimination law. W Va L Rev 123:735. https://doi.org/10.2139/ssrn.3792772
132. Washington State (2020) Certification of enrollment: engrossed substitute senate bill 6280 ('Washington State Facial Recognition Law'). https://lawfilesext.leg.wa.gov/biennium/2019-20/Pdf/Bills/Senate%20Passed%20Legislature/6280-S.PL.pdf?q=20210513071229
133. Waters A, Miikkulainen R (2014) Grade: machine learning support for graduate admissions. AI Mag 35(1):64. https://doi.org/10.1609/aimag.v35i1.2504
134. Zehlike M, Yang K, Stoyanovich J (2021) Fairness in ranking: a survey. CoRR abs/2103.14000. arxiv:2103.14000,
135. Zemel R, Wu Y, Swersky K, et al (2013) Learning fair representations. In: International conference on machine learning, PMLR, pp 325–333
136. Ziegert JC, Hanges PJ (2005) Employment discrimination: the role of implicit attitudes, motivation, and a climate for racial bias. J Appl Psychol 90(3):553
137. Bertrand M, Mullainathan S (2004) Are emily and greg more employable than lakisha and jamal? a field experiment on labor market discrimination. Am Econ Rev 94(4):991–1013
138. Hoff KA, Bashir M (2015) Trust in automation: Integrating empirical evidence on factors that influence trust. Hum Factors 57(3):407–434
139. Lahoti P, Gummadi KP, Weikum G (2019) ifair: Learning individually fair data representations for algorithmic decision making. In: 2019 IEEE 35th international conference on data engineering (icde), IEEE, pp 1334–1345
140. Langer M, König CJ, Back C, et al (2022) Trust in artificial intelligence: comparing trust processes between human and automated trustees in light of unfair bias. J Bus Psychol

Authors and Affiliations

Sebastian Biewer[1] · Kevin Baum[1,2,3] · Sarah Sterz[1] · Holger Hermanns[1] · Sven Hetmank[4] · Markus Langer[5] · Anne Lauber-Rönsberg[4] · Franz Lehr[4]

Holger Hermanns
hermanns@cs.uni-saarland.de

Sven Hetmank
sven.hetmank@tu-dresden.de

Markus Langer
markus.langer@uni-goettingen.de

Anne Lauber-Rönsberg
anne.lauber@tu-dresden.de

[1] Department of Computer Science, Saarland University, Saarland Informatics Campus, Saarbrücken 66123, Germany

[2] Institute of Philosophy, Saarland University, Campus, Saarbrücken 66123, Germany

[3] Neuro-Mechanistic Modeling, German Research Center for Artificial Intelligence (DFKI), Stuhlsatzenhausweg 3, Saarbrücken 66123, Germany

[4] IRGET, TU Dresden, Bergstraße 53, Dresden 01062, Germany

[5] Institute of Psychology, University of Göttingen, Goßlerstraße 14, Göttingen 37073, Germany