Towards a Formal Account on Negative Latency*

Clemens Dubslaff[1,2], Jonas Schulz[2,4], Patrick Wienhöft[2,3], Christel Baier[2,3], Frank H. P. Fitzek[2,4], Stefan J. Kiebel[2,5], and Johannes Lehmann[2,3]

[1] Eindhoven University of Technology, Eindhoven, The Netherlands
c.dubslaff@tue.nl
[2] Centre for Tactile Internet with Human-in-the-Loop (CeTI)
[3] Department of Computer Science, Dresden University of Technology, Germany
{christel.baier,johannes_alexander.lehmann,patrick.wienhoeft}@tu-dresden.de
[4] Department of Electrical Engineering, Dresden University of Technology, Germany
{frank.fitzek,jonas.schulz2}@tu-dresden.de
[5] Department of Psychology, Dresden University of Technology, Germany
stefan.kiebel@tu-dresden.de

Abstract. Low latency communication is a major challenge when humans have to be integrated into cyber physical systems with mixed realities. Recently, the concept of *negative latency* has been coined as a technique to use anticipatory computing and performing communication ahead of time. For this, behaviors of communication partners are predicted, e.g., by components trained through supervised machine learning, and used to precompute actions and reactions.
In this paper, we approach negative latency as anticipatory networking with formal guarantees. We first establish a formal framework for modeling predictions on goal-directed behaviors in Markov decision processes. Then, we present and characterize methods to synthesize predictions with formal quality criteria that can be turned into negative latency. We provide an outlook on applications of our approach in the settings of formal methods, reinforcement learning, and supervised learning.

1 Introduction

A key ingredient for modern communication is low latency, enabling a seamless integration of multimodal information transfer and hence remote robotic control and human interaction in virtual realities. It is thus not surprising that the 5G mobile standard also focused on ultra-low latency to compete with new technological demands [40,26]. For instance, towards an implementation of an *internet of skills*, where tactile skill information has to be communicated between sensors and actuators, a round-trip latency of less than one millisecond is an essential prerequisite [13,40,24,14].

* This work was partially supported by the DFG under the projects EXC 2050/1 (CeTI, project ID 390696704, as part of Germany's Excellence Strategy) and TRR 248 (see https://perspicuous-computing.science, project ID 389792660).

However, physical limits for communication in terms of speed of light directly tell that under those latency requirements there is a boundary at 25 km distances for an end-to-end communication [53]. In a global world striving towards ubiquitous computing, there is hence a need for more sophisticated techniques to reduce latency and enable greater distances of low-latency communication [36].

One promising technique is *anticipatory computing*. Following the seminal definition by Rosen [33], behaviors of anticipatory systems are characterized by not only depending on the past but also on beliefs in the future and future needs. Research on such systems has a long history. Similar concepts have already been considered, e.g., within *speculative execution* to speed up processing in parallel computing [46]. Recently, anticipatory computing gained more and more attention in the field of software and systems engineering [27]. For instance, automated driver assistance systems benefit from prediction functionalities to navigate in common traffic schemes but also reduce the control action space to react timely on incidents. Here, machine-learning predictors show great performance [44,25]. In the context of mobile devices and communication, *anticipatory networking* enables to reduce latency [28], especially in combination with *edgecloud computing*: Future behaviors of users and computing systems are predicted, consequences computed, and resources then shifted towards infrastructure that is physically close to the communication partner. Depending on the actual behaviors, the speculated results can then directly be transmitted from the edge computing devices to meet latency requirements. Most prominently, a similar technique has been introduced in 2019 for Google's cloud-based gaming service Stadia to precompute game display frames. There, latency lags were mitigated by ahead-of-time computation of most likely game-playing behaviors, leading to a smoother gaming experience. Google coined the term *negative latency* to promote their variant of anticipatory networking towards reducing latencies. One crucial aspect is the clear focus on gaming experience, where incorrect predictions do not have harmful impacts and thus, there is no need to require any guarantees on predicted outcomes. The latter however renders Google's technique not suitable for safety critical applications such as remote surgery or autonomous driving, whose functioning crucially depends on reliably low latency.

We argue that guarantees on predictions are key to actually demarcate negative latency from basic latency reductions through anticipatory networking. To this end, we propose a more strict understanding of negative latency, which essentially boils down to the simple relation

$$\text{negative latency} \;=\; \text{anticipatory networking} \;+\; \text{formal guarantees}$$

Guarantees on predictions for anticipatory networking are required to quantify the reliability of systems that depend on low latency, also to meet safety standards and classify potential failures of the system [32]. Having in mind the success of machine-learning trained predictors, our variant of negative latency also opens a new field in formal methods and artificial intelligence.

In this paper, we develop foundations of a formal framework that captures our understanding of negative latency. Here, we focus on the formal guarantees that can be provided for systems modeled as *Markov decision processes (MDPs, [31])*. MDPs are an expressive stochastic model having a rich support for formal quantitative analysis [4,10] and are also used as underlying model of *reinforcement learning* [45]. Towards reasoning about negative latency, we introduce MDPs with dedicated goal states and model goal-directed behaviors through sequences of state-action pairs towards reaching goals. For instance, goals could stand for different surgery tools a doctor intends to grab during remote surgery, where the hand movements model goal-directed behaviors [34,35]. Further, we introduce *predictions* as sets of anticipated goals, leading to a general framework of MDPs with goals and predictions. Since in practical applications many goals are theoretically possible, we are focusing on so-called k*-predictions* where only up to k goals are predicted. For formal guarantees, we impose thresholds on the quality of predictions, e.g., ensuring that the predicted set of goals is reached with high probability. We then consider cost annotations in MDPs that formalize execution and communication timings, e.g., the time of a hand movement during grabbing surgery tools. The k*-negative latency* corresponds to the maximal time with during any execution a high-quality k-prediction can be made ahead of reaching a predicted goal. Intuitively, giving the formal guarantees in form of high-quality predictions, the communication partner can then start reasoning about consequences of each of the k goals and react ahead of time, leading to negative latency. In case of the remote surgery example, the remote operator can already provide advice to the surgery tools ahead of time compensating the communication latency.

To the best of our knowledge, this view on predictions in MDPs has not yet been established in the literature. Actually, while our motivation stems from formalizing negative latency, our framework can be used in various other settings where, e.g., cost annotations stand for energy consumed or packages transmitted. We present algorithms for the general case of MDPs with goal predictions to solve the k-prediction problems, i.e., computing high-quality k-predictions that optimize costs before reaching goals. Here, we distinguish between *additive quality measures* that allow for synthesizing high-quality k-predictions in polynomial time, and the *canonical quality measure* of pessimistic goal reachability probability where already the problem of deciding whether a high-quality k-prediction exists is NP-complete.

In general, our basic framework models behavior of the communication partner through both nondeterministic and probabilistic choices, leading to overall pessimistic predictions and negative latency. With more information about underlying strategies and behaviors, e.g., obtained by reinforcement-learning techniques, we can reason about Markov chains as fully probabilistic models with uncertainty. In combination with well-known confidence measures, this enables to provide better predictions and negative latencies while maintaining formal guarantees. We also illustrate how supervised-learning techniques could be exploited to trade performance towards higher negative latencies for formal guarantees.

To summarize, we provide a starting point towards formally reasoning about negative latency and MDPs with goals and predictions and contribute

(1) a generic formalization of predictions and prediction qualities in cost-aware MDPs that specify goal-directed behaviors,
(2) algorithms to synthesize high-quality (cost-bounded) predictions,
(3) a discussion of applications of our framework with additional knowledge from strategy estimates and machine learning predictors, and
(4) instantiations of our framework to the setting of negative latency.

Related work. There is a body of research on anticipatory computing [33], anticipatory networking [28], speculative execution [46], and related concepts [27]. Differently, negative latency only recently attracted attention [36,34,35]. However, we are not aware of any attempt to provide a formal account on negative latency or to conceptionally demarcate anticipatory networking for reducing latency and negative latency.

Our framework formalizing predictions and negative latency relates to and uses concepts from cost-bounded reachability analysis in MDPs [47,6,17,18]. The cost-optimal k-prediction problem towards maximizing negative latency is closely related to the synthesis of resilient strategies in MDPs [8]. The latter addresses a somehow inverse problem by asking for a strategy to reach goals within a given cost bound while maximizing performance.

In the context of reinforcement learning, estimates of transition probability errors and confidence have been extensively studied [45]. Approaches use, e.g., Hoeffding bounds [19] with reasoning on lower [2] and upper confidence [9]. In this paper, we mainly follow the approach by Strehl and Littman [41,42] towards L_1-estimations of transition probabilities [50]. Such estimations are also employed for offline reinforcement learning with safe policy improvement [29,23].

The application of formal methods to machine learning models remains subject to ongoing research. Previous work in the field of formal methods focused on probabilistic guarantees for predictions based on training data [48]. In this paper, our aim is not to advance the field in this direction, but sketch how machine learning predictions also might provide negative latency.

2 Preliminaries

We briefly present our notations on Markovian models and temporal logics with costs. For more details, see standard textbooks on systems modeling and verification [10]. A *distribution* over a finite set X is a function $\delta\colon X \to [0,1]$ where $\sum_{x\in X} \delta(x) = 1$. The set of distributions over X is denoted by $\mathit{Distr}(X)$, the power set of X by $\wp(X)$. When clear from the context, we omit brackets of singleton sets, i.e., write x for $\{x\}$.

Markov decision processes. A *Markov decision process (MDP)* is a tuple $\mathcal{M} = (S, \mathit{Act}, P, C, \imath)$ where S and Act are finite sets of states and actions, respectively, $P\colon S \times \mathit{Act} \rightharpoonup \mathit{Distr}(S)$ is a partial transition probability function,

$C\colon S\times Act \to \mathbb{N}$ is a cost function, and $\imath \in S$ is an initial state. The size of $\mathcal{M}$ is the sum of all set sizes as well as binary encodings of costs and probabilities. We say that an action $\alpha \in Act$ is *enabled* in state $s \in S$ if $P(s,\alpha)$ is defined, and assume that the set of enabled actions $Act(s)$ is always non-empty. For $(s,\alpha,s') \in S\times Act\times S$ we define $P(s,\alpha,s') = P(s,\alpha)(s')$ if $\alpha \in Act(s)$ and $P(s,\alpha,s') = 0$ otherwise. If for all states $s \in S$ there is exactly one action enabled, i.e., $|Act(s)| = 1$, then $\mathcal{M}$ is called a *Markov chain (MC)*. In this case we may omit the actions from the definitions. A finite *path* is a sequence $\rho = s_0\alpha_0 s_1\alpha_1 \dots s_n$ where $P(s_i,\alpha_i,s_{i+1}) > 0$ for all $i = 0,1,\dots,n{-}1$. The *accumulated cost* on ρ is defined by $C(\rho) = \sum_{i=0}^{n-1} C(s_i,\alpha_i)$. The set of all paths in $\mathcal{M}$ starting in s is denoted by $Paths(\mathcal{M},s)$ and the fragment of finite paths by $Paths_{\mathrm{fin}}(\mathcal{M},s)$. We assume that all states in $\mathcal{M}$ are reachable from $\imath$, i.e., appear in some path starting in $\imath$.

The semantics of the MDP $\mathcal{M}$ is given by resolving nondeterministic choices through *strategies*, i.e., mappings $\sigma\colon Paths_{\mathrm{fin}}(\mathcal{M},\cdot) \to Distr(Act)$ where $\sigma(s_0\alpha_0\dots s_n)(\alpha) = 0$ for all $a \notin Act(s_n)$. We call a path ρ as above a *σ-path* if $\sigma(s_0\alpha_0\dots s_i)(\alpha_i) > 0$ for all $i = 0,\dots,n{-}1$. The probability of ρ w.r.t. σ and starting state $s \in S$ is defined as $\Pr_s^\sigma(\rho) = \prod_{i=0}^{n-1} \sigma(s_0\alpha_0\dots s_i)(\alpha_i)\cdot P(s_i,\alpha_i,s_{i+1})$ if ρ is a σ-path with $s_0 = s$ and $\Pr_s^\sigma(\rho) = 0$ otherwise. The probability of some set of finite σ-paths $B \subseteq Paths_{\mathrm{fin}}(\mathcal{M},s)$, where any path is not a prefix of another, is defined by $\Pr_s^\sigma(B) = \sum_{\rho\in B} \Pr_s^\sigma(\rho)$. Definitions for probabilities extend to measurable sets of infinite paths in the standard way [49,10].

Property specification. We specify properties on MDPs $\mathcal{M}$ as above by formulas in cost-aware *linear temporal logic (LTL)* [30], given as expressions of the grammar

$$\varphi \;::=\; \mathtt{true} \mid a \mid \neg\varphi \mid \varphi \vee \varphi \mid \bigcirc\varphi \mid \varphi \mathsf{U}^{\sim\tau} \varphi$$

where a ranges over a set of atomic propositions AP, $\sim \in \{\leq,\geq\}$, and $\tau \in \mathbb{N}$. Here, $\bigcirc$ stands for the *next operator* and $\mathsf{U}^{\sim\tau}$ for the cost-constrained *until operator*. Further standard operators can be derived, e.g., $\varphi \wedge \psi \equiv \neg(\neg\varphi \vee \neg\psi)$, *until* $\mathsf{U} \equiv \mathsf{U}^{\geq 0}$, *eventually* $\Diamond\varphi \equiv \mathtt{true}\mathsf{U}\varphi$, or *globally* $\Box\varphi \equiv \neg\Diamond\neg\varphi$. In this paper, we consider AP as the set of states of $\mathcal{M}$ and provide the semantics of an LTL formula φ as the set $[\![\varphi]\!]$ of infinite paths $\rho = s_0\alpha_0 s_1\alpha_1\dots$ in $\mathcal{M}$ for which $s_0 s_1 \dots$ satisfies φ, written $\rho \models \varphi$. For a state $s \in S$ we write $s \models \varphi$ if for all infinite $\rho \in Paths(\mathcal{M},s)$ we have $\rho \models \varphi$. We also use set notations to represent disjunctions of atomic propositions, e.g., $\rho \models A\mathsf{U}B$ for $A,B \subseteq S$ for all paths ρ that reach a state in B and only visit states of A on the way to B.

For any strategy σ, an LTL formula φ constitutes a measurable set $[\![\varphi]\!]_{\mathcal{M}^\sigma}$ of infinite σ-paths that satisfy φ [49]. To this end, for a state $s \in S$ we write $\Pr^\sigma_{\mathcal{M},s}(\varphi)$ for $\Pr^\sigma_{\mathcal{M},s}\left([\![\varphi]\!]_{\mathcal{M}^\sigma}\right)$. Best- and worst-case probabilities on LTL properties are captured by ranging over possible resolutions of nondeterminism:

$$\Pr^{\min}_{\mathcal{M},s}(\varphi) \;=\; \inf_\sigma \Pr^\sigma_{\mathcal{M},s}(\varphi) \qquad \text{and} \qquad \Pr^{\max}_{\mathcal{M},s}(\varphi) \;=\; \sup_\sigma \Pr^\sigma_{\mathcal{M},s}(\varphi)\ .$$

Quantiles. Quantiles are defined as optimal values q such that the probability of a random variable Q exceeding q is beyond a given threshold. In our setting,

we consider quantiles minimizing costs (such as time consumed) in MDP models w.r.t. worst- and best-case resolution of nondeterminism [47,6]. Let $\mathcal{M}$ be an MDP as above, φ and ψ LTL formulas over AP without cost constraints, $\vartheta \in [0,1]$ be a probability threshold, and $s \in S$. Then, the *lower-bound quantile* with respect to ϑ, φ, and ψ in s is defined as

$$\max \left\{ c \in \mathbb{N} \mid \Pr^{\min}_{\mathcal{M},s}(\varphi \mathsf{U}^{\geq c} \psi) \geq \vartheta \right\} \ . \tag{1}$$

The above quantile specifies the minimal costs c required for paths that start in s to guarantee a probability greater than ϑ for reaching a state where ψ holds, solely via states that satisfy φ and surely spending at least c. *Upper-bound quantiles* are defined accordingly, i.e.,

$$\min \left\{ c \in \mathbb{N} \mid \Pr^{\min}_{\mathcal{M},s}(\varphi \mathsf{U}^{\leq c} \psi) \geq \vartheta \right\} \ . \tag{2}$$

Quantiles can be computed using a back-propagation approach, leading to quantile values in all states of the given MDP [6].

3 Predictions in Markov Decision Processes

To model goal-directed behaviors of systems or humans and to reason about situations where predictions about targeted goals can be made, we establish a formal framework of MDPs with goals and cost predictions. For this, we fix an MDP $\mathcal{M} = (S, Act, P, C, \imath)$ throughout the section.

MDPs with goals. Let $G \subseteq S$ be a finite set of *goal states*. The pair $(\mathcal{M}, G)$ is called an *MDP with goals* if goal states are reached almost surely for any strategy and state, i.e., $(\mathcal{M}, G)$ is an MDP with goals iff $\Pr^{\min}_{\mathcal{M},s}(\Diamond G) = 1$. MDPs with goals describe goal-directed behavior starting in the initial state $\imath$ until reaching a goal state. After reaching such a goal state, the behavior continues towards a new (possibly different) goal. To this end, each path in $(\mathcal{M}, G)$ operates in several phases, constituting *goal-directed runs* that start and end in a state of $\mathit{Init} = \{\imath\} \cup G$. We define the set of runs by

$$\mathit{Runs}(\mathcal{M}, G) \ = \ \{\rho \in \mathit{Paths}(\mathcal{M}, \mathit{Init}) \mid \rho \models \bigcirc \Diamond G\}$$

Example 1. Let us introduce our running example of a human grabbing either a plate, an espresso cup, or a coffee cup [35]. We model the goal-directed hand movement by an MDP with goals $(S, Act, P, C, 211, G)$ over the state space $S = \{2,1,0\}^2 \times \{1,0\}$, goal states $G = \{000, 001, 021\}$, action space $Act = \{\mathrm{move}, \mathrm{close}, \mathrm{open}, \mathrm{turn}\}$, and transition probabilities and costs as depicted in Figure 1. The first state component models how close the user's hand is situated relative to the object: far (2), close (1), or touching (0). The second state component models the hand pose: wide (2), normal (1), or narrow (0). The third state component stands for the hand position vertical (1) or horizontal (0). Grabbing an object starts far from the object, with a normal vertical hand

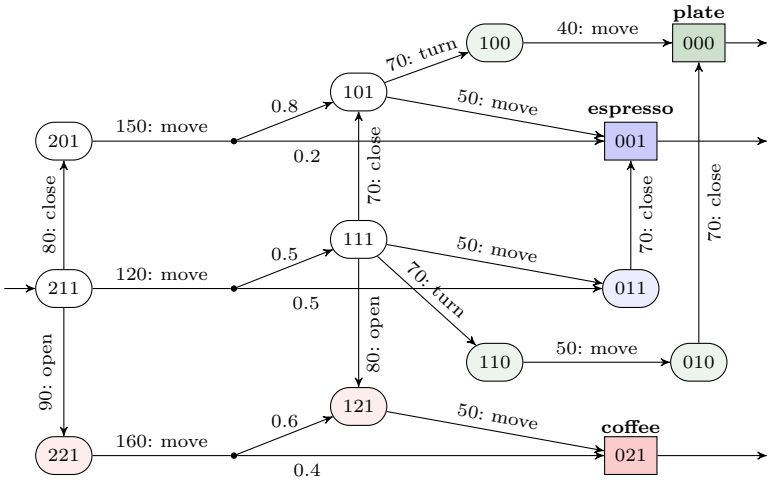

Fig. 1. Example MDP with goals of a grabbing scenario

pose (cf. initial state "211"). Costs of transitions are annotated in front of action names in Figure 1, modeling the time in milliseconds to perform the task. Moving at different speeds is modeled via probabilistic transitions towards either "close to" or "touching" the object. Essentially, the user can follow two strategies to grab the desired object: either first moving and then adjusting the hand pose towards the needs of grabbing the object (closing, opening, or turning) or the other way around. Note that turning the hand towards horizontal position directly at the beginning implies that the user targets a plate (goal "000"), since grabbing an espresso or coffee cup requires a vertical hand pose (goals "001" and "021", respectively). Further, observe that touching the object in a normal vertical pose (state "011") does not allow for opening the hand anymore, required to grab the coffee cup. After a goal is reached, i.e., one of the objects is touched and consequently grabbed, a new phase starts by returning to the initial state (indicated by the outgoing arrows of the three goals).

To reason about time-dependent properties, let us consider the LTL formula $\varphi = \Diamond^{\geq 60}\mathbf{plate}$ where **plate** stands for the state "000" (see Figure 1). Then $\Pr^{\min}_{\mathcal{M},s}(\varphi) \geq 0.9$ only holds in states "110" and "010", since from all other states, either less than 60 ms are spent to reach the goal state "000" (state "100") or there is a strategy for the human to not grab the plate (e.g., deciding to go for espresso in state "101" by moving forward).

Predictions. To formalize predictions on goal-directed behaviors in MDPs with goals, we introduce *prediction* mappings $\pi\colon S \to \wp(G)$ that assign a set of predicted goals to states. Predictions are ordered via goal inclusion, i.e., $\pi \subseteq \pi'$ iff $\pi(s) \subseteq \pi'(s)$ for all $s \in S$. We identify with $\mathit{supp}(\pi)$ the set of states where a prediction is made, i.e., $\mathit{supp}(\pi) = \{s \in S \mid \pi(s) \neq \varnothing\}$.

Intuitively, predictions are made within the goal-directed phases of $(\mathcal{M}, G)$, i.e., $\pi(s) \neq \varnothing$ in some state $s \in S$ then corresponds to predicting the reachability event $\varphi_{\pi(s)} = (\neg G)\mathsf{U}\pi(s)$, i.e., reaching the set of goal states in $\pi(s)$ without passing any other goal state. Note that with this understanding, for a meaningful prediction it should hold that $\pi(s) = \{s\}$ for all $s \in G$.

3.1 Prediction Quality Criteria

Obviously, not all prediction mappings are sensible, e.g., when a goal is not reachable but predicted. Also predicting all reachable goals or none in every state does not provide any useful information. We are mainly interested in predictions with at least one prediction in each goal-directed phase of the MDP with goals:

Definition 1 (Proper prediction). *A prediction* $\pi\colon S \to \wp(G)$ *for* $(\mathcal{M}, G)$ *is called* proper *if for all* $s_0\alpha_0 s_1 \cdots s_n \in \mathit{Runs}(\mathcal{M}, G)$ *with* $n > 1$ *there is* $i \in \{1, \ldots, n-1\}$ *such that* $\pi(s_i) \neq \varnothing$.

Within proper predictions we further would like to predict as few goals as possible but with fulfilling certain quality criteria, e.g., predicting with high confidence. For this, we introduce prediction quality measures.

Definition 2 (Prediction quality). *A* prediction quality *for* $(\mathcal{M}, G)$ *is a mapping* $\mu\colon S \times \wp(G) \to [0,1]$ *where* $\mu(s, \varnothing) = 0$ *for all* $s \in S$. *We call* μ additive *if* $\mu(s, X) + \mu(s, x) = \mu(s, X \cup \{x\})$ *for all* $s \in S$, $X \subseteq G$, *and* $x \in G \backslash X$.

Intuitively, values $\mu(s, X)$ stand for the quality of a goal prediction $X \subseteq G$ in some state $s \in S$. Note that additivity implies monotonicity, i.e., $\mu(s, X) \leq \mu(s, Y)$ for all $X \subseteq Y \subseteq G$ and $s \in G$. We relate different quality measures by pointwise comparison, i.e., for two quality measures μ and μ', we write $\mu \leq \mu'$ iff $\mu(s, X) \leq \mu'(s, X)$ for all $s \in S$ and $X \subseteq G$.

Having full knowledge about $(\mathcal{M}, G)$, the canonical candidate for prediction quality is given by the worst-case reachability probability of predicted goals. Formally, for $s \in S$ and $X \subseteq G$ this instance can be defined by

$$\mu_\varphi(s, X) \;=\; \Pr^{\min}_{\mathcal{M},s}\left(\varphi_X\right) \;=\; \Pr^{\min}_{\mathcal{M},s}\left((\neg G)\mathsf{U}X\right) \tag{3}$$

Note that this prediction quality can be computed in polynomial time using standard methods for model checking MDPs [10]. But also other quality measures could well be imagined, e.g., when uncertainty on the MDP probabilities or external influences are present. Instances we will discuss in this paper include, e.g., confidence measures in MDPs with goals learned by model-based reinforcement learning (see Section 4.1) or confidence on predictors trained with supervised machine learning (see Section 4.2).

Remark 1. The canonical prediction quality μ_φ is not additive. To see this, consider $S = \{\iota, g, g'\}$, $G = \{g, g'\}$, $\mathit{Act} = \{\alpha, \alpha'\}$ and $P(\iota, \alpha, g) = P(\iota, \alpha', g') = 1$ being the only transitions. Then

$$\Pr^{\min}_{\mathcal{M},\iota}\left(\varphi_g\right) = \Pr^{\min}_{\mathcal{M},\iota}\left(\varphi_{g'}\right) = 0 \qquad \text{but} \quad \Pr^{\min}_{\mathcal{M},\iota}\left(\varphi_{\{g,g'\}}\right) = 1.$$

Algorithm 1: SynthKMin – Synthesize $\mu\vartheta$-state k-prediction

input : MDP with goals $(\mathcal{M}, G)$, $k \in \mathbb{N}$, prediction quality μ, $\vartheta \in [0,1]$
output: A $\mu\vartheta$-state k-prediction π or **false** if none exists

forall $s \in S$ **do**
  $\pi(s) := \varnothing$ `// initialize prediction`
  while $\mu\big(s, \pi(s)\big) < \vartheta$ *and* $|\pi(s)| \leq k$ **do**
    $x_{max} := \operatorname{argmax}_{x \in G \setminus \pi(s)} \mu\big(s, x\big)$ `// add highest quality goal`
    $\pi(s) := \pi(s) \cup \{x_{max}\}$
  if $\mu\big(s, \pi(s)\big) < \vartheta$ *or* $|\pi(s)| > k$ **then**
    $\pi(s) := \varnothing$ `// no greedy prediction at` s
    if μ *is not additive* **then**
      forall $X \subseteq G$, $|X| \leq k$ **do**
        if $\mu(s, X) \geq \vartheta$ **then**
          $\pi(s) := X$ `// non-greedy prediction found at` s
          break
if *there is* $\rho \in \mathit{Runs}(\mathcal{M}, G)$ *with* $\rho \not\models \bigcirc\big((\neg G)\mathsf{U}(\mathit{supp}(\pi) \setminus G)\big)$ **then**
  return false `// there is no proper` k`-prediction`
return π

Nevertheless, μ_φ is monotonic since adding goals can only increase the probability of reaching goals independent from the chosen strategy.

State-based quality threshold criterion. A natural way of stating requirements on overall prediction quality of a system is by means of thresholds on qualities of each prediction state.

Definition 3. *Given a quality measure μ and a threshold $\vartheta \in [0,1]$, a proper prediction π for $(\mathcal{M}, G)$ is called a* $\mu\vartheta$-state prediction *if for all states $s \in S$ with $\pi(s) \neq \varnothing$ we have*

$$\mu\big(s, \pi(s)\big) \;\geq\; \vartheta \;. \tag{4}$$

π is complete *if for all $\mu\vartheta$-state predictions π' we have $\mathit{supp}(\pi') \subseteq \mathit{supp}(\pi)$ and $\pi(s) \subseteq \pi'(s)$ for all $s \in \mathit{supp}(\pi')$.*

Note that every $\mu\vartheta$-state prediction has to be proper and hence, there is at least one state $s \in S$ that has to meet the threshold criterion (4). For a prediction to be complete, it has to make a prediction in as many states as possible, with a minimal set of goals in each state (w.r.t. subset inclusion).

3.2 *k*-Predictions

In practice, the number of goals in a system might be huge, posing challenges when predictions have to be processed timely. Towards meaningful predictions and to deal with limited resources when anticipating predicted goals, we hence mainly consider predictions with a limited number of goals. Formally, given a goal bound $k \in \mathbb{N}$, we call a prediction π a *k-prediction* if $|\pi(s)| \leq k$ for all $s \in S$.

$\mu\vartheta$-state k-prediction decision problem
For an MDP with goals $(\mathcal{M}, G)$, $k \in \mathbb{N}$, prediction quality μ, and threshold $\vartheta \in [0,1]$, decide whether there is a $\mu\vartheta$-state k-prediction.

Clearly, in case of a positive answer, the goal is to synthesize complete predictions. A simple greedy scheme can compute a complete $\mu\vartheta$-state k-prediction if μ is additive. Algorithm 1 implements this scheme by iterating through all states checking whether in those states a k-prediction can be made that fulfills the state quality criterion. Note that Line 4 involves a nondeterministic choice in case at least two goals can be predicted with same quality. Further, when μ is not additive, the greedy scheme might not succeed in finding a prediction with at most k goals in some state, requiring to possibly check predictions exhaustively (see Line 8). After computing prediction sets for each state, π is a k-prediction. However, this prediction might not be proper, in which case there is no $\mu\vartheta$-state k-prediction (see Definition 1) and the algorithm returns **false** (Line 13).

Theorem 1. *Let $(\mathcal{M}, G)$ be an MDP with goals, $k \in \mathbb{N}$, μ a prediction quality that can be computed in polynomial time, and $\vartheta \in [0,1]$. If there is a $\mu\vartheta$-state k-prediction for $(\mathcal{M}, G)$, Algorithm 1 returns one taking at most exponential time. For additive μ, the prediction is complete and computed in polynomial time.*

Example 2. Let us illustrate predictions on our running example (see Example 1), where we abbreviate by **plate** the state "000", by **espresso** the state "001", and by **coffee** the state "021". Application of Algorithm 1 for $k = 2$, $\vartheta = 0.9$, and the canonical prediction quality $\mu_\varphi(s, X) = \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}X\big)$ yields a complete $\mu\vartheta$-state 2-prediction π:

$$\begin{aligned}
&\pi(211) = \pi(111) = \varnothing\\
&\pi(000) = \pi(100) = \pi(010) = \pi(110) = \{\textbf{plate}\}\\
&\pi(001) = \pi(011) = \{\textbf{espresso}\}\\
&\pi(021) = \pi(221) = \pi(121) = \{\textbf{coffee}\}\\
&\pi(201) = \pi(101) = \{\textbf{plate}, \textbf{espresso}\}
\end{aligned}$$

Note that in the initial state "211" and in "111" there are strategies that eventually open the hand, leading almost surely to goal **coffee**, eventually turn, leading almost surely to goal **plate**, or only move and close without turning, leading almost surely to **espresso**. Thus, by ranging over all possible strategies for checking our quality criteria, we see that there is no $\mu\vartheta$-state 2-prediction possible in those states (cf. first line above). Singleton predictions are trivial for goal states but also those that can reach only a single goal.

Let us more elaborate on the case where the greedy computation scheme in Algorithm 1 is not successful and one has to check possible sets with at most k goals whether their prediction quality exceeds the quality threshold. In fact, while in Example 2 this situation did not arise, this can well happen for the canonical prediction quality μ_φ (see Remark 1).

Example 3. Let $S = \{\imath, g, h, h'\}$, $G = \{g, h, h'\}$, $Act = \{\alpha, \alpha'\}$, and transitions are given by $P(\imath, \alpha, g) = P(\imath, \alpha', g) = 0.3$, $P(\imath, \alpha, h) = P(\imath, \alpha', h') = 0.6$, and $P(\imath, \alpha, h') = P(\imath, \alpha', h) = 0.1$. Then, Algorithm 1 with $k = 2$, the canonical prediction quality μ_φ, and $\vartheta = 0.5$ first adds goal g in state s, since

$$\mu_\varphi(\imath, g) = 0.3 \;\geq\; \mu_\varphi(\imath, h) = \mu_\varphi(\imath, h') = 0.1.$$

After adding goal h or h', we obtain $\pi(\imath) = \{g, h\}$ or $\pi(\imath) = \{g, h'\}$ with $|\pi(\imath)| = k = 2$, where $\mu_\varphi\big(\imath, \pi(\imath)\big) = 0.4$, not exceeding ϑ. This leads the algorithm to invoke a subset search, finally returning the complete $\mu_\varphi\vartheta$-state k-prediction π with $\pi(\imath) = \{h, h'\}$ since $\mu_\varphi\big(\imath, \{h, h'\}\big) = 0.7 \geq \vartheta$.

This example already hints at the $\mu_\varphi\vartheta$-state k-prediction decision problem to be not easily solvable. In fact, this problem is NP-complete, which can be shown by a reduction from the *minimum hitting set problem (MHS)* [15].

Theorem 2. *The $\mu_\varphi\vartheta$-state k-prediction decision problem is NP-complete.*

3.3 Cost-aware Predictions

In many practical applications it is not only relevant to reach a goal, but also to do so with meeting cost constraints. For instance, when costs are given by means of energy and we are in a state with low battery, it is important to also take the energy budget into account when predicting goals to be reached. We hence extend our notion of prediction quality (cf. Definition 2) to assess costs for reaching a predicted goal.

Definition 4 (Cost prediction quality). *A* cost prediction quality *for $(\mathcal{M}, G)$ is a mapping $\nu\colon \mathbb{N} \times S \times \wp(G) \to [0,1]$ where for all $c \in \mathbb{N}$ the mapping $\nu_c\colon S \times \wp(G) \to [0,1]$ defined by $\nu_c(s, X) = \nu(c, s, X)$ for all $s \in S$, $X \subseteq G$ is a prediction quality. If for all $c, c' \in \mathbb{N}$ we have that $c \leq c'$ implies $\nu_c \leq \nu_{c'}$, we call ν* increasing *and likewise, if $c \leq c'$ implies $\nu_c \geq \nu_{c'}$, we call ν* decreasing.

Intuitively, increasing measures shall be used when reaching goals at high costs is preferred, e.g., when charging a battery if costs model energy or if time is seen as cost and goal-directed behavior tries to extend the time until some maintenance operation. Decreasing measures are relevant when lowering costs is preferable, e.g., when draining a battery in case of energy consumption or to increase performance if costs stand for execution times.

Natural candidates for cost prediction qualities are cost-bounded reachability probabilities. Here, increasing and decreasing measures $\nu_{\leq}$ and $\nu_{\geq}$, respectively, can be specified for all $s \in S$, $c \in \mathbb{N}$, and $X \subseteq G$ by

$$\nu_{\leq c}(s, X) \;=\; \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}^{\leq c} X\big) \tag{5}$$

$$\nu_{\geq c}(s, X) \;=\; \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}^{\geq c} X\big). \tag{6}$$

Cost-optimal k-predictions. Synthesizing $\nu_c\vartheta$-state k-predictions for a fixed

Algorithm 2: Synthesize cost-maximal $\nu_c\vartheta$-state k-prediction

input : MDP with goals $(\mathcal{M}, G)$, $k \in \mathbb{N}$, decreasing cost prediction quality ν, quality threshold $\vartheta \in [0,1]$

output: A cost-maximal $\nu_c\vartheta$-state k-prediction π or **false** if there is none

1 $c_{\max} := 1$
2 **while** $\textsc{SynthKMin}(\mathcal{M}, G, k, \nu_{c_{\max}}, \vartheta)$ **do**
3 $\quad c_{\max} := 2 \cdot c_{\max}$
4 $c := c_{\min} := \lfloor c_{\max}/2 \rfloor$
5 **while** $c_{\max} - c_{\min} > 1$ **do**
6 $\quad c := \lfloor \frac{c_{\max}+c_{\min}}{2} \rfloor$
7 $\quad$ **if** $\textsc{SynthKMin}(\mathcal{M}, G, k, \nu_c, \vartheta)$ **then**
8 $\quad\quad c_{\min} := c$
9 $\quad$ **else**
10 $\quad\quad c_{\max} := c$
11 **return** c, $\textsc{SynthKMin}(\mathcal{M}, G, k, \nu_c, \vartheta)$

cost bound c could possibly be done by Algorithm 1 (see the definition of cost prediction quality). A more interesting problem is to not fix a cost bound but to determine the minimal or maximal c for increasing or decreasing quality measures, respectively, for which there is a $\nu_c\vartheta$-state k-prediction.

Definition 5. *Let $(\mathcal{M}, G)$ be an MDP with goals, $k \in \mathbb{N}$, ν a cost prediction quality, and $\vartheta \in [0,1]$. A prediction π is a* cost-maximal (cost-minimal) $\nu\vartheta$-state k-prediction *if there is a $c \in \mathbb{N}$ such that π is a $\nu_c\vartheta$-state k-prediction and for all $c' > c$ ($c' < c$) there is no $\nu_{c'}\vartheta$-state k-prediction.*

If also a synthesis of such cost-optimal k-predictions can be achieved, this then also provides predictions that (to some extent) predict costs to be spend for reaching goals. Towards a synthesis algorithm, let us first note that computing the minimal and maximal cost bounds for canonical cost qualities (5) and (6), respectively, and exceeding a quality threshold ϑ corresponds to classical quantile computations [6]. We use similar techniques by exploiting the increasing or decreasing property of cost quality measures and performing an exponential search followed by a binary search on cost bounds while stepwise invoking Algorithm 1. Algorithm 2 implements the cost-maximizing case for decreasing cost quality measures, assuming there is some cost bound $c_{\max}$ such that there is no $\nu_{c_{\max}}\vartheta$-state k-prediction. First, we perform an exponential search, i.e., the cost bound $c_{\max}$ is exponentially increased until we cannot synthesize any $\nu_{c_{\max}}\vartheta$-state k-prediction anymore (cf Line 3). At Line 4 we then have the situation where there is such a prediction for $c_{\min}$ but not for $c_{\max}$. Shrinking the interval $[c_{\min}, c_{\max}]$ then is the purpose of a binary search (see Line 6).

Proposition 1. *Let $(\mathcal{M}, G)$ be an MDP with goals, ν an additive decreasing cost prediction quality, $\vartheta \in [0,1]$, and $k \in \mathbb{N}$. Then if there is a $\nu_0\vartheta$-state k-prediction and a cost bound $c_{\max} \in \mathbb{N}$ for which there is no $\nu_{c_{\max}}\vartheta$-state k-prediction, then Algorithm 2 returns a cost-maximal $\nu\vartheta$-state k-prediction.*

A similar algorithm as Algorithm 2 and proposition can be also established for increasing cost prediction qualities in a straight-forward manner, starting with an exponential search until a prediction can be synthesized and then a binary search with flipped roles of $c_{\max}$ and $c_{\min}$.

Example 4. Continue Example 2 with the canonical decreasing cost prediction quality $\nu_{\geq}$ (6). Algorithm 2 first performs an exponential search until $c_{\max} = 64$, where no prediction on the run $211 \xrightarrow{\text{move}} 111 \xrightarrow{\text{open}} 121 \xrightarrow{\text{move}} 021$ can be made since from "121" the goal **coffee** is reached in less than 64 ms and this is the only non-trivial 2-prediction state on this path in the unconstrained case (cf. Example 2). The binary search then terminates at the maximal cost bound $c = 50$ such that there is a complete $\nu_{\geq 50}0.9$-state 2-prediction

$$\begin{aligned}
&\pi(211) = \pi(111) = \pi(000) = \pi(001) = \pi(021) = \pi(100) = \varnothing \\
&\pi(010) = \pi(110) = \{\textbf{plate}\} \\
&\pi(011) = \{\textbf{espresso}\} \\
&\pi(221) = \pi(121) = \{\textbf{coffee}\} \\
&\pi(201) = \pi(101) = \{\textbf{plate}, \textbf{espresso}\}
\end{aligned}$$

Observe that there is no prediction in goal states, since non-zero cost prediction qualities (6) require costs invested while passing through non-goal states only.

3.4 Negative Latency

Consider a communication system modeled as MDP with goals in which costs correspond to timings of system executions and transmissions. Then, anticipatory networking [36] can be implemented through predictions in system states, fitting well in our formal framework developed in the last sections. Following the principle of negative latency as anticipatory networking with guarantees as we motivated in the introduction, we illustrate in this section how to exploit prediction guarantees to turn anticipatory networking into *negative latency*.

Intuitively, we define *negative latency* as the amount of time a goal can be predicted with high confidence, i.e., predicting goals with meeting prediction quality criteria. Formally, let $(\mathcal{M}, G) = (S, Act, P, C, \imath, G)$ be an MDP with goals where the cost function C assigns (non-zero) execution times to state-action pairs. Negative latency can then be established through cost bounds of $\nu_c\vartheta$-state k-predictions (see Definition 5).

Definition 6 (Negative latency). *For $k \in \mathbb{N}$, decreasing cost prediction quality ν, and $\vartheta \in [0, 1]$ the MDP with goals $(\mathcal{M}, G)$ has $\nu\vartheta$-*state k-negative latency

$$\ell \;=\; \max\{c \in \mathbb{N} \mid \textit{there is a } \nu_c\vartheta\textit{-state } k\textit{-prediction}\}.$$

When an MDP with goals has k-negative latency ℓ, then at minimum ℓ ahead of time there is a prediction that guarantees with high confidence to reach the predicted goals. To this end, the impact of at most k predicted goals can be anticipated and precomputed ℓ time ahead, used to reduce latency by ℓ. Practically

most relevant is the case where $k = 1$, i.e., the time that one single outcome can be predicted ahead of time [34].

The possible negative latency in given MDPs and predictions as implementations could be computed by Algorithm 2. The inverse synthesis problem might be also relevant: given a target negative latency ℓ and compute the smallest k such that there is a $\nu_\ell\vartheta$-state k-prediction. This can be achieved by a simple algorithm similar to Algorithm 2 but performing a binary search on k.

4 Machine Learning for Negative Latency

Nondeterminism in the system's underlying MDP model is commonly interpreted as environmental impact, e.g., human input as in our running example, or unknown behavior from the side of the system. To account for all possible environmental and unknown influences, still providing strict quality guarantees on predictions, we hence quantified over all possible strategies in our canonical quality measures $\nu_{\leq c}$ (5) and $\nu_{\geq c}$ (6), covering all worst-case scenarios. However, this view is overly conservative as neither a human user nor the underlying system specifically "design" their strategy in such a way to defy possible predictions. Instead, another perspective is to consider user's strategies as memoryless and randomized but hidden, i.e., unknown to the system communicating with the user [5,9]. This implies that the underlying model does not include any nondeterminism anymore but instead also transitions of which we do not know the exact transition probability.

In this section, we describe how machine-learning predictors can be used to resolve nondeterminism by probabilism with uncertainty that follow the canonical quality criteria. For this, we first utilize results from reinforcement learning to estimate strategies from sample runs on the system, on which the machinery of the last section can be directly applied due to the same underlying concepts of MDPs. Second, we showcase how supervised machine learning can be used to predict goals in system states based on continuous sensor data, implementing an extension of the abstract predictions on MDPs we introduced.

4.1 Strategy Estimation

By integrating information from sample data on an underlying strategy, we cannot expect to obtain exact transition probabilities but estimations only. To this end, we define a formalism of sets of MCs to reflect this uncertainty.

Definition 7 (L_1-Markov chain). *An* L_1-Markov chain (L1MC) $\mathcal{C}^e$ *is a Markov chain* $\mathcal{C}$ *along with an* error function $e\colon S \to \mathbb{R}$.

Given an MC $\mathcal{C} = (S, P, C, \imath)$ and error function e, we call an MC $\mathcal{C}' = (S, P', C, \imath)$ an *instantiation* of $\mathcal{C}^e$ if $\|P(s) - P'(s)\|_1 \leq e(s)$ for all $s \in S$. The set of all instantiations of an L1MC $\mathcal{C}^e$ is denoted by $[\mathcal{C}^e]$. Intuitively, an L1MC $\mathcal{C}^e$ represents the set of all MCs where the L_1-distance between their transition probabilities in each state s is bounded by $e(s)$. The semantics of L1MCs arises from

first picking an instantiation $\mathcal{C}' \in [\mathcal{C}^e]$ and then considering all infinite paths on $\mathcal{C}'$ as within standard Markov chains, similar to the *uncertain Markov chain (UMC)* semantics in the setting of *interval Markov chains (IMCs)* [37,12,3]. In the framework of IMCs, another common semantics is the *interval MDP* semantics, where an adversarial player chooses the transition probabilities at each step [37,12,3]. However, considering our assumption of the existence of a fixed memoryless strategy we do not consider this variant here.

Strategy estimation. To represent past information gathered through prior knowledge or observations from an underlying MDP model $\mathcal{M} = (S, Act, P, C, \imath)$, we consider a data set $\mathcal{D} \subseteq Paths(\mathcal{M}, \imath)$ of paths observed in $\mathcal{M}$. We denote by $\#_{\mathcal{D}}(s, A)$ the number of occurrences of an action from the set $A \subseteq Act$ being executed in state $s \in S$ in paths of $\mathcal{D}$. As we assume there is a fixed and memoryless strategy σ that is employed in the system, even observations of single transitions without history are sufficient to estimate σ by a set of strategies $\Sigma_{\mathcal{D}}$. We do this by an S-rectangular set of strategies, i.e., we estimate $\sigma(s)$ by a set of distributions $\Sigma_{\mathcal{D}}(s) \subseteq Distr\big(Act(s)\big)$ for each $s \in S$ and construct $\Sigma_{\mathcal{D}}$ as the cross product of all $\Sigma_{\mathcal{D}}(s)$.

Further, we introduce an *error tolerance* $\delta \in \mathbb{R}$ to formalize uncertainty that $\sigma(s)$ lies within any (non-trivial) $\Sigma_{\mathcal{D}}(s)$. To provide guarantees on the latter we require that $\sigma \in \Sigma_{\mathcal{D}}$ with probability at least $1 - \delta$. As we aim for S-rectangular estimations, we split the error tolerance δ uniformly over all states, defining $\delta_T = \delta / |S|$. For all states $s \in S$ we then can utilize a result of Weissman et al. [50] to guarantee that $\sigma(s) \in \Sigma_{\mathcal{D}}(s)$ with probability at least $1 - \delta_T$:

Definition 8 (L_1-strategy estimation). *Let $\mathcal{M}$ be an MDP and $\mathcal{D}$ be a set of runs on $\mathcal{M}$ sampled from common fixed strategy σ. For a given error tolerance $\delta_T \in (0, 1)$ we define the* L_1-strategy interval estimate *$\Sigma_{\mathcal{D}}$ for all $s \in S$ by*

$$\Sigma_{\mathcal{D}}(s) = \left\{ \hat{\sigma}(s) \in Distr(Act(s)) \;\middle|\; \|\hat{\sigma}(s) - \hat{p}(s)\|_1 \leq \sqrt{\frac{2(\ln(2^{|S|} - 2) - \ln \delta_T)}{\#_{\mathcal{D}}(s, Act)}} \right\}$$

where $\hat{p}(s)(\alpha) = \frac{\#_{\mathcal{D}}(s,\alpha)}{\#_{\mathcal{D}}(s,Act)}$.

This strategy estimation can be included into our framework towards L1MCs estimates of MDPs: Given an MDP $\mathcal{M}$ with a fixed, but hidden, strategy σ that gives rise to the induced MC $\mathcal{M}^\sigma$. Then for an error tolerance δ we can construct an L1MC $\mathcal{M}^e_{\mathcal{D}}$ from an MC $\mathcal{M}_{\mathcal{D}}$ that resolves nondeterminism through $\mathcal{D}$ such that $\mathcal{M}^\sigma$ is an instantiation of $\mathcal{M}^e_{\mathcal{D}}$ with probability at least $1 - \delta$.

Definition 9 (MDP estimation). *Let $\mathcal{M} = (S, Act, C, P, \imath)$ be an MDP, $\mathcal{D} \subseteq Paths(\mathcal{M}, \imath)$ sampled from a fixed strategy σ, and $\delta \in (0, 1)$ an error tolerance. The $\mathcal{D}$-*estimate of $\mathcal{M}$ *is the L1MC $\mathcal{M}^e_{\mathcal{D}}$ with $\mathcal{M}_{\mathcal{D}} = (S \cup S_{Act}, \hat{C}, \hat{P}, \imath)$ where $S_{Act} = \{s_\alpha \mid s \in S, \alpha \in Act(s)\}$, and where for all $s, s' \in S \cup S_{Act}$ cost estimates*

$\hat{C}$ are defined by $\hat{C}(s) = C(s,\alpha)$ if $s = s_\alpha \in S_{Act}$ and $\hat{C}(s) = 0$ otherwise, probability estimates $\hat{P}$ and the error function e are defined by

$$\hat{P}(s,s') = \begin{cases} \frac{\#_{\mathcal{D}}(s,\alpha)}{\#_{\mathcal{D}}(s,Act(s))} & \text{if } s \in S, s' = s_\alpha \in S_{Act} \text{ and } \#_{\mathcal{D}}\big(s,Act(s)\big) > 0 \\ 1/|Act(s)| & \text{if } s \in S, s' = s_\alpha \in S_{Act} \text{ and } \#_{\mathcal{D}}\big(s,Act(s)\big) = 0 \\ P(s,\alpha,s') & \text{if } s = s_\alpha \in S_{Act} \text{ and } s' \in S \\ 0 & \text{otherwise} \end{cases}$$

$$e(s) = \begin{cases} 0 & \text{if } s \in S \\ |Act(s)| & \text{if } s = s_\alpha \in S_{Act} \text{ and } \#_{\mathcal{D}}(s,Act) = 0 \\ \sqrt{\frac{2(\ln(2^{|S|}-2)-\ln\delta_T)}{\#_{\mathcal{D}}(s,Act)}} & \text{if } s = s_\alpha \in S_{Act} \text{ and } \#_{\mathcal{D}}(s,Act) > 0 \end{cases}$$

Intuitively, we replace each transition in $\mathcal{M}$ by two transitions in $\mathcal{M}_{\mathcal{D}}$: The first one estimates the probability that an action α is taken with an admissible L_1-error as in the L_1-strategy estimation (cf. Definition 8), while the second performs the original transition of $\mathcal{M}$ without any error. For the first, we take care of the case where $\#_{\mathcal{D}}\big(s,Act(s)\big) = 0$, by setting the error function to a value such that every transition function is a valid instantiation (see second case of the $\hat{P}$ definition). The newly introduced states s_α serve as the intermediate states between these transitions such that any run in $\mathcal{M}_{\mathcal{D}}$ alternates between states from S and S_{Act}. Further, intermediate states also encode taken actions, leading to a well-defined cost function that preserves accumulated rewards.

Note that each instantiation $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ corresponds to the MDP $\mathcal{M}$ under a strategy $\sigma_{\mathcal{C}} \in \Sigma_{\mathcal{D}}$. Since the probability that the hidden strategy is within the L_1-strategy estimate, i.e., $\sigma \in \Sigma_{\mathcal{D}}$, is at least $1-\delta$, we immediately obtain that the probability that an $\mathcal{M}^\sigma$ with stutter steps has an instantiation of $\mathcal{M}^e_{\mathcal{D}}$ is also at least $1-\delta$. Here, for including stutter steps in $\mathcal{M}^\sigma$, for each transition $s \xrightarrow{\alpha} s'$ in $\mathcal{M}$ there is a finite path $s \dots s'$ in $\mathcal{C}$ that does not contain any other states from S, and that has the same probability mass and accumulated cost.

Proposition 2. *Let $\mathcal{M}$ be an MDP, $\mathcal{D}$ be a set of runs on $\mathcal{M}$ sampled from a fixed strategy σ, and $\delta \in (0,1)$ an error tolerance. Then with probability at least $1-\delta$ there is a $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ that is $\mathcal{M}^\sigma$ up to stutter steps.*

Example 5. Let us again consider our running example (see Example 1). We have three states in which nondeterministic choices can be made: states "211", "101", and "111". Assume we have observed a total of 100 runs in the environment and made the following observations:

state s	$\#_{\mathcal{D}}(s,\alpha)$ for action α:				$\#_{\mathcal{D}}(s,Act)$
	close	move	open	turn	
210	10	40	50	-	100
101	-	7	-	3	10
111	1	14	9	1	25

For a given error tolerance of δ we can then construct the MDP estimation via an L1MC. In Figure 2 we show the MC $\mathcal{C}$ of the L1MC $\mathcal{M}^e_{\mathcal{D}}$ for this specific example. Note that we omit auxiliary states here if they only had a deterministic successor anyway, e.g., state "211_{close}" as its only successor would be "201".

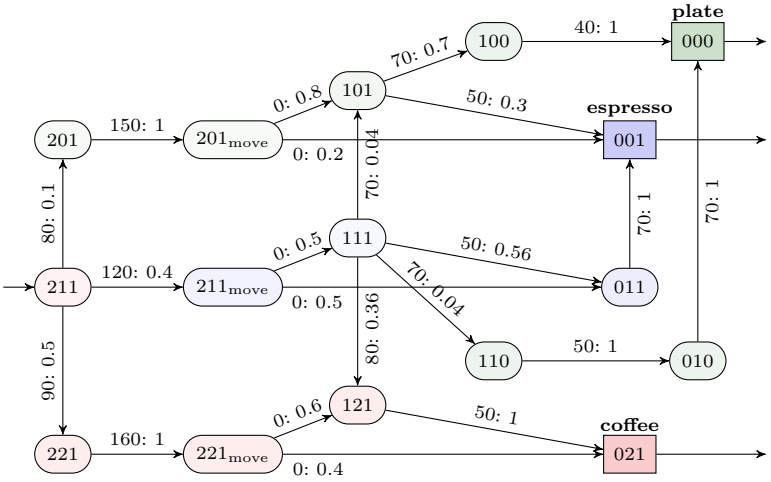

Fig. 2. Example MC of a grabbing scenario with estimated strategy

For $\delta = 0.1$ we obtain an error tolerance $\delta_T = \frac{1}{30}$ for each of the three states in which we estimate the strategy. The corresponding error function e is then computed as in Definition 9 where all the non-zero entries are

$$e(211) \approx 0.264, \qquad e(101) \approx 0.692, \quad \text{and} \quad e(111) \approx 0.589.$$

k-Predictions in L1MCs. We can consider our framework of prediction in MDPs (see Section 3) in the setting of L1MCs. For this, we take quality measures that quantify over all possible instantiations of an L1MC instead of all strategies of an MDP as done for canonical qualities (3), (5), and (6). Formally, for an L1MC with goals $(\mathcal{M}^e_{\mathcal{D}}, G)$, we define the prediction quality measure $\hat{\mu}_\varphi \colon S \times \wp(G) \to [0,1]$ for all states $s \in S$ and goals $X \subseteq G$ by

$$\hat{\mu}_\varphi(s, X) = \min_{\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]} \Pr_{\mathcal{C},s}(\varphi_X) = \min_{\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]} \Pr_{\mathcal{C},s}\left((\neg G) \mathsf{U} X\right) \ .$$

Similarly, for $\sim \in \{\leq, \geq\}$ we define cost prediction qualities as

$$\hat{\nu}_{\sim c}(s, X) = \min_{\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]} \Pr_{\mathcal{C},s}\left((\neg G) \mathsf{U}^{\sim c} X\right) \ .$$

Note that in case the data set is empty, i.e., no information about the resolution of the nondeterminism is available, the (cost) prediction quality as well as

the quality threshold agree for an MDP $\mathcal{M}$ and the corresponding L1MC $\mathcal{M}^e_{\mathcal{D}}$ estimating $\mathcal{M}$. Intuitively, this is due to the corresponding transition function corresponding to an L1MC that is an instantiation of $\mathcal{M}^e_{\mathcal{D}}$, no matter by which (possibly probabilistic) scheduler the nondeterminism is resolved.

In that sense, we see that even in general and with using canonical quality measures, taking on the L1MC view on MDPs is beneficial, since even without any data collected the qualities calculated are equivalent:

Lemma 1. *Let $(\mathcal{M}, G)$ be an MDP with goals and $\mathcal{M}^e_{\mathcal{D}}$ an L1MC that estimates $\mathcal{M}$ as in Definition 9 with the empty data set $\mathcal{D} = \varnothing$. Then, for all states $s \in S$, goals $X \subseteq G$, $\sim \in \{\leq, \geq\}$, and cost thresholds $c \in \mathbb{N}$*

$$\hat{\mu}_{\varphi}(s, X) = \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}X\big) \quad \text{and} \quad \hat{\nu}_{\sim c}(s, X) = \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}^{\sim c}X\big).$$

Observe that with using L1MCs, however, we introduce additional uncertainty as we can only guarantee with probability $1 - \delta$ that the MC $\mathcal{M}^\sigma$ induced by the underlying hidden strategy σ is actually an instantiation of the L1MC $\mathcal{M}^e_{\mathcal{D}}$. However, when lifting the definition of quality thresholds to L1MCs, we can incorporate this error term:

Definition 10. *Given an MDP with goals $(\mathcal{M}, G)$ and an error tolerance $\delta \in [0, 1]$, let $\mathcal{M}^e_{\mathcal{D}}$ be the L1MC estimating $\mathcal{M}$ through data $\mathcal{D} \subseteq Paths(\mathcal{M}, \iota)$. Further, given a quality threshold $\vartheta \in [0, 1]$, a cost bound $c \in \mathbb{N}$, and cost bound c, a proper prediction π for $(\mathcal{M}_{\mathcal{D}}, G)$ is called a $\hat{\nu}_{\sim c}\vartheta\delta$-*state prediction *where $\sim \in \{\leq, \geq\}$ if for all states $s \in S$ with $\pi(s) \neq \varnothing$ we have*

$$(1 - \delta) \cdot \hat{\nu}_{\sim c}\big(s, \pi(s)\big) \;\geq\; \vartheta \;. \tag{7}$$

Proposition 3. *Given an MDP with goals $(\mathcal{M}, G)$ and a $\hat{\nu}_{\sim c}\vartheta\delta$-state prediction $\pi(s)$ as in Definition 10, then $\Pr_{\mathcal{M}^\sigma,s}\big((\neg G)\mathsf{U}^{\sim c}X\big) \geq \vartheta$.*

Intuitively, $(1-\delta)$ is the (minimal) probability that the underlying MC $\mathcal{M}^\sigma$ is an instantiation of $\mathcal{M}^e_{\mathcal{D}}$. This is feasible when we see $\hat{\nu}$ is related to the worst-case reachability probabilities of goals, e.g., being one of the canonical cost predictions $\hat{\nu}_{\leq c}$ or $\hat{\nu}_{\geq c}$. Then, $\hat{\nu}_c\big(s, \pi(s)\big)$ is the minimal probability over all instantiations to fulfill the prediction in state s with given cost constraint depend on c.

Example 6. Let us continue Example 5 by computing a $\nu_{\leq 250}\vartheta\delta$-state 1-prediction in the initial state "211" with $\delta = 0.1$ while maximizing ϑ. To do this, we compute $\nu_{\leq 250}(211, x)$ for all $x \in \{\textbf{plate}, \textbf{espresso}, \textbf{coffee}\}$ by constructing a minimizing instantiation for each goal. Here, we take an adversary role trying to minimize the probability to reach each goal state.
(plate): Since $0.1 \leq e(211)/2$ and $0.08 \leq e(111)/2$, there is an instantiation where **plate** is unreachable, obtainable by minimizing the probability from transitions to states "201", "101", and "110". Hence, $\hat{\nu}_{\leq 250}(211, \textbf{coffee}) = 0$.
(espresso): As $0.3 \leq e(101)/2$ we have instantiations with $P(101, \textbf{espresso}) = 0$. In an effort to minimize the reachability probability of **espresso** we pick an instantiation where $P(211, 211_{\text{move}})$ is as small as possible. To ensure that the MC

is still an instantiation, we must have $P(211, 211_{\text{move}}) \geq 0.4 - e(211)/2 \approx 0.268$. Similarly, we minimize $P(111, 011) \geq 0.56 - e(111)/2 \approx 0.265$. Hence, we have an instantiation with only one path reaching **espresso** and by evaluating its probability we obtain $\hat{\nu}_{\leq 250}(211, \textbf{espresso}) = 0.268 \cdot 0.265 = 0.071$.
(coffee): The path via states "221" and "121" violates our cost constraint and is thus not relevant to our example. As before, we minimize $P(111, 121) \geq 0.36 - e(111)/2 \approx 0.065$. For state "211" we would like to minimize the transition probability to both states "111" and "221". However, as the error function only allows a fixed L_1 deviation of the transition function, we can only minimize one of the two. In this case we prefer to minimize $P(211, 221) \geq 0.5 - e(211)/2 \approx 0.368$ as the probability to reach **coffee** is greater from "221" than from "111". This leaves two non-zero paths reaching **coffee** under the given cost constraints from which we can compute $\hat{\nu}_{\leq 250}(211, \textbf{coffee}) = 0.368 \cdot 0.4 + 0.4 \cdot 0.065 \approx 0.173$.

Thus, we have that the goal **coffee** is a $\nu_{\leq 250}\vartheta\delta$-state 1-prediction in state "211" where $\delta = 0.1$ and $\vartheta \approx 0.173$ is maximal. Notice the difference to the original setting where no 1-prediction with $\vartheta > 0$ could be made in the initial state. This in fact matches our intuition: If we have observed that a person has often grabbed the coffee in the past, we might predict with some level of confidence that they will do so again.

Computing $\hat{\nu}_{\sim c}\vartheta\delta$-state k-predictions. Notice that a high error tolerance δ implies that the set of instantiations also grows, which in turn makes cost prediction quality estimates $\hat{\nu}$ smaller. Finding the optimal value of δ, i.e., the value that maximizes the left-hand side of (7) seems like a hard task for which we do not see a direct procedure. However, for a given δ it is straight forward to compute $\mathcal{M}^e_{\mathcal{D}}$ by following Definition 9. We now show that we can also compute the values of $\hat{\nu}_{\leq}$ and $\hat{\nu}_{\geq}$ in L1MCs as well as lift the algorithms for computing k-estimates towards L1MCs. Lifting Algorithm 1 and Algorithm 2 to L1MCs is relatively straight forward as we only need to replace all occurrences of μ and ν with $\hat{\mu}$ and $\hat{\nu}$. In turn, this leaves the question how to compute the canonical quality measures $\hat{\mu}_{\varphi}$, $\hat{\nu}_{\leq}$, and $\hat{\nu}_{\geq}$ (cf. Equations (3), (5) and (6)).

Without cost restrictions the canonical quality measure solely relies on computing $\hat{\mu} = \min_{\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]} \Pr_{\mathcal{C},s}\big((\neg G) \mathsf{U} X\big)$. This can be computed by an extension of the standard value iteration algorithm that roughly works as follows [41,42]: In each iteration, we construct the minimizing instantiation $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ w.r.t. the value function computed in the previous iteration. For this, we initialize all transitions as $P'(s, s') = \hat{P}(s, s')$ and then shift probability mass away from those successors that have the highest value into states that have the lowest value, continuing this until either the transition is deterministic, or we have shifted a total of $e(s)/2$ of probability mass. While we allow for different instantiations of transitions in each step of the value iteration, the algorithm converges towards a probability-minimizing instantiation of the L1MC. Hence, the value iteration computes the value iteration under the UMC semantics (cf. Definition 7). Convergence for this procedure was originally only shown for discounted rewards [42] but later also for contracting models [52], i.e., models such that a goal state is always eventually reached, which is a given assumption for MDPs with goals

(see Section 3). While [52] does not directly consider L1MCs, the convergence proof does not rely on the specific shape of the uncertainty.
In the case where we have cost constraints, we have to compute $\hat{\nu}_{\geq c}$ or $\hat{\nu}_{\leq c}$. For a given threshold ϑ this is equivalent to deciding *probabilistic computation tree logic (PCTL)* formulas to hold in all instantiations. While we are not aware of any results showing this for L1MCs, this is an active area of research for the related model of IMCs [21]. For these, the decision problem can be solved via a reduction to checking *parametric Markov reward models* [1] against step-bounded PCTL formulas [3]. Using, e.g., a binary search on ϑ, we can approximate $\hat{\nu}_{\sim c}$ to arbitrary precision. While L1MC and IMCs generally differ in their semantics, they do coincide for MCs in which every state has only two successors. Further, any MC can be transformed into this form [51]. Alternatively, one can also directly encode the PCTL decision problem on L1MCs as a quadratic program by treating the transition probabilities as variables and adding constraints according to the error function given by the L1MC.

4.2 Supervised Learning

While estimates on the strategy employed in the model can yield better predictions, it requires sampling data for every state of the environment to do so. In particular, it does not exploit the fact that the strategy may behave similarly in similar states, e.g., that only differ slightly in spatial coordinates. Under the assumption that similar states indeed employ similar strategies in practice, we can encode the components of a state as numerical values and use a neural network to estimate the strategy employed, essentially interpolating the strategy at states where no or little data is available. While we cannot give precise guarantees over the prediction quality anymore, this allows us to cover larger state spaces with less data. Additionally, with this method using a neural network, it is easily possible to handle even continuous state spaces.

Goal predictions through ML. Supervised learning for classification problems aims to predict a label y from a set of labels Y based on a system state $\mathbf{x}$. Referring to the example given in Fig. 1, $\mathbf{x}$ represents the posture of the hand, captured via a sensor glove for example, and y is the goal of a human grab. Note that we depart from the notations of our MDP framework of the last sections, since we also allow for continuous states $\mathbf{x}$ and multiple goal states labelled by single goal predictions $y \in Y$.

In general, we assume a data generating process $p_{data}(\mathbf{x}, y) = p(y \mid \mathbf{x})p(\mathbf{x})$ over states $\mathbf{x}$ and goal labels $y \in Y$. The aim is then to fit the tuneable parameters θ of a model $p_\theta(y \mid \mathbf{x})$ to increase the quality of the prediction. Since the data generating process $p_{data}(\mathbf{x}, y)$ is not known a priori, a data set $\mathcal{D} = \{(\mathbf{x}^i, y^i)\}_{i=1}^N$ consisting of N pairs $(\mathbf{x}_i, y_i)$ is used to represent the empirical distribution $\hat{p}_{data}(\mathbf{x}, y)$. The common machine learning objective is then to minimize the *empirical risk* defined as

$$\mathbb{E}_{(\mathbf{x},y)\sim\hat{p}_{data}(\mathbf{x},y)}\big[\mathcal{L}\big(p_\theta(y \mid \mathbf{x}), y\big)\big] \;=\; \frac{1}{N}\sum\nolimits_{i=1}^{N} \mathcal{L}\big(p_\theta(y \mid \mathbf{x}^i), y^i\big),$$

with $\mathcal{L}$ representing a loss function, measuring the discrepancy between the model output $p_\theta(y \mid \mathbf{x}^i)$ and corresponding target label y^i [16]. Therefore, we adjust the model parameters by optimizing for

$$\theta^* \;=\; \arg\min_\theta \sum\nolimits_{i=1}^{N} \mathcal{L}\big(p_\theta(y \mid \mathbf{x}^i), y^i\big).$$

Uncertainty in ML predictors. Neural networks are subject to *aleatoric* and *epistemic* uncertainty [20]. Similar to the predictions we discussed in our MDP prediction framework, high-quality predictions should satisfy two goals. First, the neural network should accurately predict the most likely label $\hat{y} = \arg\max_k p_\theta(\, y{=}k \mid \mathbf{x}\,)$ in any state $\mathbf{x}$. Second, it should also communicate the uncertainty a prediction is entailed with. A simple and computationally easy measure to quantify the uncertainty of a prediction for state $\mathbf{x}$ is to compute the entropy [38] of the predictive distribution, also known as *predictive entropy* [22]:

$$H(p_\theta, \mathbf{x}) \;=\; -\sum\nolimits_{y \in Y} p_\theta(y \mid \mathbf{x}) \cdot \log_2 \big(p_\theta(y \mid \mathbf{x})\big).$$

We extend the entropy to the k-prediction setting by

$$H_k(p_\theta, \mathbf{x}) \;=\; -\sum\nolimits_{\substack{Z \subseteq Y \\ |Z|=k}} p_\theta(Z \mid \mathbf{x}) \cdot \log_2 \big(p_\theta(Z \mid \mathbf{x})\big). \tag{8}$$

The prediction of the k most likely predictions can then be associated with a measure of uncertainty $H_k(p_\theta, \mathbf{x})$ in state $\mathbf{x}$. While this slightly departs from goal-dependent prediction qualities (cf. Definition 3) in our MDP setting, we can also impose a threshold ϑ on the *confidence* computed as the inverse of the normalized uncertainty $1 - \frac{H_k(p_\theta, \mathbf{x})}{\log_2 |Y|}$. The prediction can be potentially rejected if the associate confidence of that prediction does not exceed ϑ.

Trustworthy uncertainty estimates by calibration. To correctly quantify uncertainty associated with predictions, a neural network needs to be *well-calibrated*. Calibration allows to justifiably determine thresholds on the predictive uncertainty [39]. For the case of predicting a single goal, a neural network classifier is considered calibrated if for any predicted goals $\hat{y}$ and the corresponding classifier probability output $\hat{p} = p_\theta(\hat{y} = y \mid \mathbf{x})$ we have $P(\hat{y} = y \mid \hat{p} = p) = p$ for all $p \in [0, 1]$. Intuitively, this means a neural network is well-calibrated if its output $\hat{p}$ matches the probabilities p_{data} of the data-generating process for all labels $y \in Y$. While the fact that we do not know p_{data} means that we can never expect an exact calibration, we can still employ empirical methods that yield a neural network that is close to being well-calibrated.

A commonly-used scheme for neural network calibration is to generate an ensemble of diverse neural network by randomizing the weight initialization and training process of neural network and subsequently ensemble-averaging multiple individual models [22]. We then can ensure that the estimation is probably approximately correct by utilizing statistical methods such as Hoeffding's inequality [19] or Weissman's bound on the L_1 distance used in Section 4.1 [50].

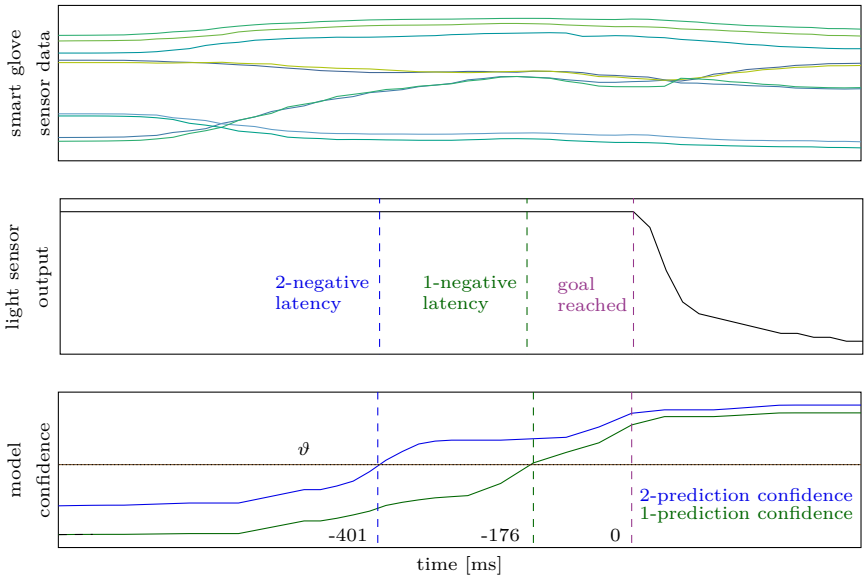

Fig. 3. Sensor data (top plot), light sensor output (center plot), and 1-prediction and 2-prediction confidence of the classifier (bottom plot) in a sample grabbing task [34].

Note that this only guarantees that the empirically estimated neural network is calibrated w.r.t. the neural network minimizing loss on the training data $\mathcal{D}$. Hence, we cannot provide hard guarantees that the gathered data $\mathcal{D}$ is representative for the actual data-generating process.

Negative latency. In the setting of negative latency, i.e., predicting goals ahead of time with guarantees, supervised learning can be used to tackle two fundamental problems that arise with formal MDP model-based and strategy estimations obtained from reinforcement learning. First, it is also applicable when the ground truth underlying process is completely unknown apart from the influencing features of goal-directed behavior. Then, supervised learning can provide at least tools to estimate the process based on observed data. Second, real-world processes often result in observations in the form of continuous random variables, which can be handled by most popular supervised learning techniques.

Given a neural network and estimates on prediction performance and calibration, negative latency can then be seen as the time difference between the moment of predicting the end state of an ongoing process before that end state is reached. While the predictive uncertainty of the model allows handling the trade-off between prediction accuracy and prediction rejection, the calibration estimated during training gives rise to the empirical approximation of the model to the underlying process.

Example 7. Our running example (see Example 1) has been inspired by a real-world experiment where we used a smart glove as source for sensor data during

grabbing objects such as a plate, an espresso cup, or a coffee cup [35,34]. In this experiment, we used a light sensor beneath the objects to determine the time where a goal is reached (i.e., the object is lifted). Training and calibrating a neural network classifier on these inputs, we were able to obtain a predictor that in combination with confidence thresholds could be used to turn predictions into negative latency. Figure 3 shows a sample grab scenario from [34] where smart glove sensor data (top), the light sensor (center), and the 1-prediction and 2-prediction confidence (8) (bottom) are depicted. Note that all sensor data acquired during gradual hand movements towards the object are continuous, i.e., cannot directly be modeled within our MDP and L1MC framework with goals and predictions. In the bottom plot, we indicated the moments in time where the 1-prediction and 2-prediction confidence first surpass the quality threshold $\vartheta = 0.5$. 1-negative and 2-negative latencies then are determined by the time from these time points to the actual goal reached, i.e., 176 ms and 401 ms, respectively.

5 Concluding Remarks

In this paper, we sharpened the concept of negative latency by including formal guarantees into anticipatory computing. Through a general formal framework, we introduced goal-oriented predictions in MDPs and how they can be synthesized under formal guarantees. Applications to negative latency in the theoretical setting, but also for strategy estimations obtained by reinforcement learning and supervised learning were showcased. To this end, our work opened a new perspective on anticipatory computing, where many further research could be conducted. Our notions could be well extended to much richer classes of models, such as continuous-time MDPs or models that account for partial observability, or incremental versions. This would also imply the need for new methods, e.g., including machine-learning methods for partially observable MDPs [11]. Towards handling settings where the strategy is not necessarily fixed, it is possible to adapt the reinforcement learning approach by utilizing linearly updating intervals [43]. Causal relationships between actions in goal-directed behavior could also be the source of formal guarantees towards negative latency. A direct application would be using *degrees of sufficiency and necessity* [7] as quality measure of predictive models.

In the presented framework of cost-optimal predictions, we took on a local view on predictions, assessing the quality of predictions in states independently from each other. This approach renders our framework more accessible to machine learning applications, but might appear strict in other cases. For instance, when many rare events could only provide small negative latencies, the overall negative latency drops as well. An approach to investigate in future work would be to provide *phase-based formal guarantees* instead of state-based ones, i.e., imposing quality thresholds on runs during phases of goal-directed behavior instead. Also other synthesis problems with varying parameters such as ϑ, k, overlapping goal assignments etc., are worth directions to investigate.

References

1. Alur, R., Henzinger, T.A., Vardi, M.Y.: Parametric real-time reasoning. In: Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing. pp. 592–601. STOC '93, ACM, New York, NY, USA (1993)
2. Ashok, P., Křetínský, J., Weininger, M.: Pac statistical model checking for markov decision processes and stochastic games. In: Dillig, I., Tasiran, S. (eds.) Computer Aided Verification. pp. 497–519. Springer International Publishing, Cham (2019)
3. Bacci, G., Delahaye, B., Larsen, K.G., Mariegaard, A.: Quantitative Analysis of Interval Markov Chains, pp. 57–77. Springer International Publishing, Cham (2021)
4. Baier, C., de Alfaro, L., Forejt, V., Kwiatkowska, M.: Model checking probabilistic systems. In: Clarke, E.M., Henzinger, T.A., Veith, H., Bloem, R. (eds.) Handbook of Model Checking, pp. 963–999. Springer (2018)
5. Baier, C., Cuevas Rivera, D., Dubslaff, C., Kiebel, S.J.: Human-inspired models for tactile computing, chap. 8, pp. 173–200. Academic Press (2021)
6. Baier, C., Daum, M., Dubslaff, C., Klein, J., Klüppelholz, S.: Energy-utility quantiles. In: Proc. of the 6th NASA Formal Methods Symposium (NFM). LNCS, vol. 8430, pp. 285–299. Springer (2014)
7. Baier, C., Dubslaff, C., Funke, F., Jantsch, S., Piribauer, J., Ziemek, R.: Operational causality - necessarily sufficient and sufficiently necessary. In: A Journey from Process Algebra via Timed Automata to Model Learning. Lecture Notes in Computer Science, vol. 13560, pp. 27–45. Springer (2022)
8. Baier, C., Dubslaff, C., Korenčiak, L., Kučera, A., Řehák, V.: Synthesis of optimal resilient control strategies. In: 15th International Symposium on Automated Technology for Verification and Analysis (ATVA). LNCS, vol. 10482, pp. 417–434. Springer (2017)
9. Baier, C., Dubslaff, C., Wienhöft, P., Kiebel, S.J.: Strategy synthesis in markov decision processes under limited sampling access. In: Rozier, K.Y., Chaudhuri, S. (eds.) NASA Formal Methods. pp. 86–103. Springer Nature Switzerland, Cham (2023)
10. Baier, C., Katoen, J.P.: Principles of Model Checking. The MIT Press (2008)
11. Carr, S., Jansen, N., Wimmer, R., Fu, J., Topcu, U.: Human-in-the-loop synthesis for partially observable markov decision processes. In: 2018 Annual American Control Conference, ACC 2018, Milwaukee, WI, USA, June 27-29, 2018. pp. 762–769. IEEE (2018)
12. Chatterjee, K., Sen, K., Henzinger, T.A.: Model-checking omega-regular properties of interval markov chains. In: "Foundations of Software Science and Computational Structures". pp. 302–317. FOSSACS'08/ETAPS'08, Springer-Verlag, Berlin, Heidelberg (2008)
13. Fettweis, G.P.: The tactile internet: Applications and challenges. IEEE Vehicular Technology Magazine **9**(1), 64–70 (2014)
14. Fitzek, F.H., Li, S.C., Speidel, S., Strufe, T.: Chapter 1 - Tactile Internet with Human-in-the-Loop: New frontiers of transdisciplinary research. Academic Press (2021)
15. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences). W. H. Freeman, first edition edn. (1979)
16. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press (2016)
17. Haase, C., Kiefer, S.: The odds of staying on budget. In: 42nd International Colloquium on Automata, Language and Programming (ICALP). LNCS, vol. 9135, pp. 234–246 (2015)

18. Hartmanns, A., Junges, S., Katoen, J.P., Quatmann, T.: Multi-cost bounded reachability in MDP. In: Beyer, D., Huisman, M. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 320–339. Springer International Publishing, Cham (2018)
19. Hoeffding, W.: Probability inequalities for sums of bounded random variables. Journal of the American Statistical Association **58**(301), 13–30 (1963)
20. Hüllermeier, E., Waegeman, W.: Aleatoric and epistemic uncertainty in machine learning: An introduction to concepts and methods. Machine Learning **110**, 457–506 (2021)
21. Jonsson, B., Larsen, K.G.: Specification and refinement of probabilistic processes. In: Proc. of the 6th Annual IEEE Symp. on Logic in Computer Science (LICS). pp. 266–277 (1991)
22. Lakshminarayanan, B., Pritzel, A., Blundell, C.: Simple and scalable predictive uncertainty estimation using deep ensembles. Advances in neural information processing systems **30** (2017)
23. Laroche, R., Trichelair, P., des Combes, R.T.: Safe policy improvement with baseline bootstrapping. In: ICML. pp. 3652–3661. PMLR (2019)
24. Lema, M.A., Antonakoglou, K., Sardis, F., Sornkarn, N., Condoluci, M., Mahmoodi, T., Dohler, M.: 5G case study of internet of skills: Slicing the human senses. 2017 European Conference on Networks and Communications (EuCNC) pp. 1–6 (2017)
25. Mozaffari, S., Al-Jarrah, O.Y., Dianati, M., Jennings, P., Mouzakitis, A.: Deep learning-based vehicle behavior prediction for autonomous driving applications: A review. IEEE Transactions on Intelligent Transportation Systems **23**(1), 33–47 (2022)
26. Parvez, I., Rahmati, A., Guvenc, I., Sarwat, A.I., Dai, H.: A survey on low latency towards 5G: Ran, core network and caching solutions. IEEE Communications Surveys & Tutorials **20**(4), 3098–3130 (2018)
27. Peischl, B., Tazl, O.A., Wotawa, F.: Testing anticipatory systems: A systematic mapping study on the state of the art. Journal of Systems and Software **192**, 111387 (2022)
28. Pejovic, V., Musolesi, M.: Anticipatory mobile computing: A survey of the state of the art and research challenges. ACM Comput. Surv. **47**(3) (apr 2015)
29. Petrik, M., Ghavamzadeh, M., Chow, Y.: Safe policy improvement by minimizing robust baseline regret. In: NIPS. pp. 2298–2306. Curran Associates, Inc. (2016)
30. Pnueli, A.: The temporal logic of programs. In: Proceedings of the 18th Symposium on Foundations of Computer Science (SFCS). pp. 46–57. IEEE (1977)
31. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons, Inc. (1994)
32. Rausand, M.: Reliability of Safety-Critical Systems: Theory and Applications. Wiley Publishing, 1st edn. (2014)
33. Rosen, R.: Anticipatory Systems: Philosophical, Mathematical, and Methodological Foundations. IFSR international series on systems science and engineering, Elsevier Science & Technology Books (1985)
34. Schulz, J., Dubslaff, C., Seeling, P., Li, S.C., Speidel, S., Fitzek, F.H.P.: Negative latency in the tactile internet as enabler for global metaverse immersion. IEEE Network (accepted for publication) (2023)
35. Schulz, J., Nguyen, V., Seeling, P., Nguyen, G.T., Fitzek, F.H.P.: Anticipatory hand glove: Understanding human actions for enhanced interaction. In: Proceedings of the ACM international joint conference on Pervasive and Ubiquitous Computing (UbiComp). Association for Computing Machinery (2023), accepted for publication

36. Seeling, P., Fitzek, F.H.: Anticipatory networking: Negative latency for ubiquitous computing. In: 2021 IEEE 18th Annual Consumer Communications & Networking Conference (CCNC). pp. 1–4 (2021)
37. Sen, K., Viswanathan, M., Agha, G.: Model-checking markov chains in the presence of uncertainties. In: Hermanns, H., Palsberg, J. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 394–410. Springer Berlin Heidelberg, Berlin, Heidelberg (2006)
38. Shannon, C.E.: A mathematical theory of communication. The Bell system technical journal **27**(3), 379–423 (1948)
39. Silva Filho, T., Song, H., Perello-Nieto, M., Santos-Rodriguez, R., Kull, M., Flach, P.: Classifier calibration: a survey on how to assess and improve predicted class probabilities. Machine Learning pp. 1–50 (2023)
40. Simsek, M., Aijaz, A., Dohler, M., Sachs, J., Fettweis, G.: The 5G-enabled tactile internet: Applications, requirements, and architecture. In: 2016 IEEE Wireless Communications and Networking Conference. pp. 1–6 (2016)
41. Strehl, A., Littman, M.: An empirical evaluation of interval estimation for markov decision processes. pp. 128– 135 (12 2004)
42. Strehl, A., Littman, M.: An analysis of model-based interval estimation for markov decision processes. Journal of Computer and System Sciences **74**, 1309–1331 (12 2008)
43. Suilen, M., Simão, T., Jansen, N., Parker, D.: Robust anytime learning of markov decision processes. Proceedings of NeurIPS (2022)
44. Sun, Q., Huang, X., Gu, J., Williams, B.C., Zhao, H.: M2i: From factored marginal trajectory prediction to interactive prediction. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6533–6542. IEEE Computer Society, Los Alamitos, CA, USA (jun 2022)
45. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. The MIT Press, second edn. (2018)
46. Tomasulo, R.M.: An efficient algorithm for exploiting multiple arithmetic units. IBM Journal of Research and Development **11**(1), 25–33 (1967)
47. Ummels, M., Baier, C.: Computing quantiles in Markov reward models. In: 16th International Conference on Foundations of Software Science and Computation Structures (FOSSACS). LNCS, vol. 7794, pp. 353–368 (2013)
48. Urban, C., Miné, A.: A review of formal methods applied to machine learning. arXiv preprint arXiv:2104.02466 (2021)
49. Vardi, M.Y.: Automatic verification of probabilistic concurrent finite-state programs. In: Proc. of the 26th IEEE Symp. on Foundations of Computer Science (FOCS). pp. 327–338. IEEE Computer Society (1985)
50. Weissman, T., Ordentlich, E., Seroussi, G., Verdú, S., Weinberger, M.J.: Inequalities for the L1 deviation of the empirical distribution. Hewlett-Packard Labs, Tech. Rep (2003)
51. Wienhöft, P., Suilen, M., Simão, T.D., Dubslaff, C., Baier, C., Jansen, N.: More for less: Safe policy improvement with stronger performance guarantees. In: IJCAI (2023)
52. Wu, D., Koutsoukos, X.: Reachability analysis of uncertain systems using bounded-parameter markov decision processes. Artificial Intelligence **172**(8), 945–954 (2008)
53. Xiang, Z., Gabriel, F., Urbano, E., Nguyen, G.T., Reisslein, M., Fitzek, F.H.P.: Reducing latency in virtual machines: Enabling tactile internet for human-machine co-working. IEEE Journal on Selected Areas in Communications **37**(5), 1098–1116 (2019)

A Appendix

In this appendix, we mainly provide the proofs of our statements that did not fit into the main paper due to space constraints.

A.1 Predictions in MDPs

Theorem 3 (Proof of Theorem 1). *Let $(\mathcal{M}, G)$ be an MDP with goals, $k \in \mathbb{N}$, μ an additive prediction quality, and $\vartheta \in [0,1]$. If there is a $\mu\vartheta$-state k-prediction for $(\mathcal{M}, G)$, then Algorithm 1 computes a complete one in time polynomial in the size of $(\mathcal{M}, G)$.*

Proof. For all $s \in S$, it is checked whether there is a k-prediction by performing a while loop in Line 3. This while loop terminates after adding exactly one element to $\pi(s)$ in each execution the latest after k steps. Hence, Algorithm 1 terminates and there are in total at most $|S| \cdot |G|$ executions of the while loop, which is also polynomial in the size of $(\mathcal{M}, G)$. By setting the prediction to $\varnothing$ in Line 6 when (4) is violated or the prediction is larger than k, we ensure that $\pi(s)$ is indeed a $\mu\vartheta$-state k-prediction. Completeness is guaranteed through probability ordering in Line 4 and additivity of the prediction quality μ. Properness is correctly ensured in Line 13, since all states that are not in the support of π violated the condition of Line 6. □

Theorem 4. *The $\mu_\varphi\vartheta$-state k-prediction decision problem is NP-complete.*

Proof. Containment in NP is clear, since nondeterministically guessing a prediction π and checking whether for all $s \in S$ with $\pi(s) \neq \varnothing$ we have $\mu_\varphi\big(s, \pi(s)\big) \geq \vartheta$ can be done in polynomial time through standard MDP reachability [10].

We show NP-hardness through a reduction from the *minimum hitting set problem (MHS)*, which is known to be NP-complete [15]. MHS takes a finite set of vertices V, hyperedges $E_0, E_1, \ldots, E_n \subseteq V$, and a hitting set parameter $k \in \mathbb{N}$ with $k \leq |V|$ as input, asking whether there is a hitting set $H \subseteq V$ with $|H| \leq k$ such that for all $i \in \{0, 1, \ldots, n\}$ the hyperedge E_i hits H, i.e., $E_i \cap H \neq \varnothing$. Assume an instance of the MHS and set $\vartheta = 1/|V|$ along with the MDP with goals $(\mathcal{M}, G)$ where $S = \{\imath\} \cup V$ where $\imath \notin V$, $G = V$, $Act = \{\alpha_0, \alpha_1, \ldots, \alpha_n\}$, and the transition probabilities are given for all $i \in \{0, 1, \ldots, n\}$ by $P(\imath, \alpha_i, s) = 1/|E_i|$ if $s \in E_i$ and $P(\imath, \alpha_i, s) = 0$ otherwise. We now show that any solution of the MHS induces a $\mu_\varphi\vartheta$-state k-prediction and vice versa.
(⇒): Let $H \subseteq V$ be a solution of the MHS. Then for all $i \in \{0, 1, \ldots, n\}$ pick an $s_i \in E_i \cap H$, which exists due to $E_i \cap H \neq \varnothing$, and observe that $P(\imath, \alpha_i, s_i) \geq 1/|E_i| \geq \vartheta$. Hence, for all distributions δ over Act we have $\sum_{i=0}^{n} \delta(\alpha_i) \cdot P(\imath, \alpha_i, s_i) \geq \vartheta$, which directly yields $\mu_\varphi(\imath, H) \geq \vartheta$ since every possible strategy can be represented as a distribution over Act. Now define the proper k-prediction π by $\pi(\imath) = H$ and $\pi(s) = \varnothing$ for $s \in V$, which is in fact a $\mu_\varphi\vartheta$-state k-prediction.
(⇐): Let π be a $\mu_\varphi\vartheta$-state k-prediction. Then, $\pi(\imath) \neq \varnothing$ since π is proper. Further, for all $i \in \{0, 1, \ldots, n\}$ we have $P\big(\imath, \alpha_i, \pi(\imath)\big) \geq \vartheta$ since otherwise there

would be a strategy σ with $\sigma(\iota, \alpha_i) = 1$ such that $\mu_\varphi\big(\iota, \pi(\iota)\big) < \vartheta$ due to μ_φ minimizing over all strategies. Thus, for all $i \in \{0, 1, \ldots, n\}$ there is $s_i \in \pi(\iota)$ with $P(\iota, \alpha_i, s_i) \geq \vartheta$, since all transition probabilities in $\mathcal{M}$ are at least ϑ. By construction $s_i \in E_i \cap \pi(\iota)$ and since $|\pi(\iota)| \leq k$, the hitting set $H = \pi(\iota)$ is a solution of the corresponding MHS. □

A.2 Strategy Estimation

Proposition 4 (Proof of Proposition 2). *Let $\mathcal{M}$ be an MDP, $\mathcal{D}$ be a set of runs on $\mathcal{M}$ sampled from a fixed strategy σ, and $\delta \in (0, 1)$ an error tolerance. Then with probability at least $1 - \delta$ there is a $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ that is $\mathcal{M}^\sigma$ up to stutter steps.*

Proof. From Definition 8 and [50] it follows that with probability at least $1 - \delta$ there is an $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ with state space $S \cup S_\alpha$ such that for all $s \in S$ and $s_\alpha \in S_{Act}$ we have $P_{\mathcal{C}}(s, s_\alpha) = \sigma(s)(\alpha)$.

By definition, in $\mathcal{M}$ we have that the path $s\alpha s'$ has probability mass $\sigma(s)(\alpha) \cdot P(s, \alpha, s') = P_{\mathcal{C}}(s, s_\alpha) \cdot P(s_\alpha, s') = \Pr(ss_\alpha s')$, i.e., the path $ss_\alpha s'$ in $\mathcal{M}'$ has the same probability mass as $s\alpha s'$ in $\mathcal{M}$. The accumulated cost is $C(s, \alpha)$ for both paths by definition. □

Lemma 2 (Proof of Lemma 1). *Let $\mathcal{M} = (S, Act, C, P, \iota, G)$ be an MDP and $\mathcal{M}^e_{\mathcal{D}}$ an L1MC that estimates $\mathcal{M}$ as in Definition 9 with the empty data set $\mathcal{D} = \varnothing$. Then, for all states $s \in S$, goals $X \subseteq G$, cost thresholds $c \in \mathbb{N}$*

$$\begin{aligned}
\hat{\mu}_\varphi(s, X) &= \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}X\big)\\
\hat{\nu}_{\leq c}(s, X) &= \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}^{\leq c}X\big)\\
\hat{\nu}_{\geq c}(s, X) &= \Pr^{\min}_{\mathcal{M},s}\big((\neg G)\mathsf{U}^{\geq c}X\big)
\end{aligned}$$

Proof. As $\mathcal{D} = \emptyset$ we have that the error function e is defined as $e(s) = |Act(s)|$ for all $s \in S$ and $e(s_\alpha) = 0$ for all $s_\alpha \in S_{Act}$. Hence, the transition function for all instantiations $\mathcal{C}' = (S \cup S_{Act}, C', P', \iota, G)$ are exactly specified by the following constraints for all $s \in S$ and $s_\alpha \in S_{Act}$:

$$\begin{aligned}
\sum_{s' \in S_{Act}} |P'(s, s') - \hat{P}(s, s')| &\leq |Act(s)|\\
\sum_{s' \in S} |P'(s_\alpha, s') - \hat{P}(s_\alpha, s')| &\leq 0
\end{aligned}$$

In case $s \in S$ by definition we have $|P'(s, s') - \hat{P}(s, s')| > 0$ for at most $|Act(s)|$ values of s', i.e., the first condition is always true. The second condition obviously only holds if $P'(s\alpha, s') = \hat{P}(s\alpha, s')$, and by definition, $\hat{P}(s\alpha, s') = P(s, \alpha, s')$, i.e., all instantiations preserve the probabilistic transitions in the MDP $\mathcal{M}$.

Now, consider an instantiation $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ of our L1MC. In the L1MC have $\Pr_{\mathcal{M}'}(ss_\alpha s') = P'(s, s_\alpha) \cdot P'(s_\alpha, s') = P'(s, s_\alpha) \cdot P(s, \alpha, s')$ for all $s \in S$. Similarly, in the MDP we have $\Pr_{\mathcal{M}}(s\alpha s') = \sigma(s)(\alpha) \cdot P(s, \alpha, s')$ for all $s \in S$. This

means that we can establish a bijection f between instantiations $\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]$ and schedulers σ over $\mathcal{M}$: We define $f(P') = \sigma$ where σ satisfies that for all $s \in S$ and $\alpha \in Act(s)$ we have

$$\sigma(s)(a) = P'(s, s_\alpha)$$

Then $\mathcal{C}$ and the MDP with associated scheduler $\sigma = f(P')$ are equivalent up to stutter steps (cf. Proposition 2), i.e., have same probability and cost for equivalent paths. From this it follows that μ and $\hat{\mu}$, as well as $\nu_{\sim c}$ and $\hat{\nu}_{\sim c}$ always agree. □

Proposition 5 (Proof of Proposition 3). *Given an MDP with goals $(\mathcal{M}, G)$ and a $\hat{\nu}_{\sim c}\vartheta\delta$-state prediction $\pi(s)$ as in Definition 10, we have $\Pr_{\mathcal{M}^\sigma,s}((\neg G)\mathsf{U}^{\sim c}X) \geq \vartheta$.*

Proof. We show this directly by a sequence of straight-forward inequalities:

$$\begin{aligned}
&\Pr_{\mathcal{M}^\sigma,s}((\neg G)\mathsf{U}^{\sim c}X) \\
\geq &\Pr_{\mathcal{M}^\sigma,s}((\neg G)\mathsf{U}^{\sim c}X \mid \mathcal{M}^\sigma \in [\mathcal{M}^e_{\mathcal{D}}]) \cdot \Pr(\mathcal{M}^\sigma \in [\mathcal{M}^e_{\mathcal{D}}]) \\
\geq &\min_{\mathcal{C} \in [\mathcal{M}^e_{\mathcal{D}}]} \Pr_{\mathcal{C},s}((\neg G)\mathsf{U}^{\sim c}X) \cdot \Pr(\mathcal{M}^\sigma \in [\mathcal{M}^e_{\mathcal{D}}]) \\
\geq &\hat{\nu}_{\sim c}(s, X) \cdot (1 - \delta) \\
\geq &\vartheta
\end{aligned}$$