Reinforcement learning for discretized Euclidean MDPs

Asger Horn Brorholt, Manfred Jaeger, Peter Gjøl Jensen, and Kim Guldstrand Larsen

Department of Computer Science, Aalborg University, Denmark

Abstract. Modern model checking tools like UPPAAL Stratego provide a rich framework for modeling cyber-physical systems involving time and continuous state descriptors. A key objective is to design controllers for such systems that optimize a given objective, e.g., minimizing energy consumption. At an abstract level, the controller design problem can be cast as optimizing a strategy in a continuous (Euclidean) Markov decision process. Discretization of the continuous state space is a simple yet effective strategy to solve this optimization problem in a flexible, non-parametric manner. In previous work we have introduced a reinforcement learning strategy on dynamically refined discretizations, and we have analyzed at the semantic level approximations of Euclidean MDPs by Imprecise MDPs. In this paper we are moving to close the gap between the theoretical analysis and the practical reinforcement learning approach. We introduce several alternative simulation strategies that on the one hand maintain approximation guarantees as the granularity of the discretization increases, and on the other hand turns our learning scenario into a standard Q-learning procedure. The known convergence guarantees for Q-learning then provide theoretical guarantees for the near-optimality of the strategies we learn for our cyber-physical system models.

1 Introduction

Modern model-checkers (e.g. PRISM [19], STORM [6, 11], MODEST [12], KeYmaeraX [23], UPPAAL Stratego [20, 5]) are increasingly having a focus on integration of state-of-the-art reinforcement learning (RL) and model checking (MC) [22]. In this effort RL is leveraged to efficiently construct near-optimal control strategies while model checking techniques are used to give absolute [4], probabilistic [2, 1] or statistical guarantees [10] of crucial safety properties.

Most importantly, the existing model checkers offer a variety of rich and mature modelling formalisms for defining Markov decision processes (MDPs) that may be used to run simulations for the off-line training of RL policies. Compared to traditional RL scenarios this allows for a number of additional capabilities, e.g. in terms of “targeted sampling” of initial states, rare configurations, etc. The modelling formalisms range from finite state MDPs (STORM, PRISM) to continuous-space (Euclidean) Markov decision processes (EMDPs) (MODEST, UPPAAL Stratego) and Simulink (KeYmaeraX). For continuous-space models abstractions are often used in order to obtain the required guarantees [14, 4].

Most of the above integrations of RL and model checking rely on external components for the RL training (e.g. Open Gym [11] and Simulinks RL Toolbox [14]). In contrast the tool UPPAAL Stratego offers its own RL method for continuous-space MDPs [16]. Here the learning method is based on a dynamic partition-refinement approach for function approximations providing high flexibility regarding the types of functions that can be approximated, but also closely aligns with continuous-time model-checking techniques, so-called zones.

The RL method of UPPAAL Stratego has already been applied for the construction of near-optimal controllers in a number of industrial applications including traffic-lights [7], water management [8, 9], floor heating systems [21], heat-pumps [13], as well as distributed fleets of autonomous mobile robots [3].

However, the proven practical usefulness of the system is not fully complemented by theoretical guarantees. The RL approach in UPPAAL Stratego is based on sampled runs in the continuous state space. This, and the interleaving with partition-refinement steps, means that classic convergence guarantees for RL [15] in finite state spaces are not directly applicable to this approach. In [18] we have started to develop theoretical underpinnings of the UPPAAL Stratego approach by using imprecise Markov decision processe (IMDPs) to formalize partition-based abstractions, and to approximate EMDPs by standard finite state MDPs that provide upper and lower bounds on the cost function of the EMDP. This analysis was at the purely semantic level and did not make an explicit link to the convergence of RL.

This paper is a step towards closing the gap between the theoretical guarantees of [18], and the algorithmic solutions implemented in UPPAAL Stratego. Our main objective here is to reduce the RL process in UPPAAL Stratego to standard finite state RL, so that all existing implementations and convergence guarantees for RL become applicable. This reduction relies on two main components: first, deviating from [18], we will consider a discounted cost criterion as our objective. As we will show by a counter-example, a conjecture stated in [18] regarding the asymptotic tightness of upper and lower bounds does not actually hold in the un-discounted case. Second, we introduce new simulation approaches, such that system runs obtained from these simulations are equivalent to simulations from a standard finite state MDP.

Related Work

We refer to [16] and [18] for a broader discussion of related approaches towards abstractions of continuous state space systems. Recent work that is most closely related to ours are [25] and [24] which also develop RL in continuous spaces via finite state space partitionings (or, more generally, finite *coverings* of the state space). Whereas [25] works with a fixed partitioning, [24] develops an approach for adaptive refinement of the partitions during learning. In contrast to our work, the authors here assume that the real systems can not be simulated, and hence can not be used for generating data for offline learning. As a result, the focus is on minimizing *regret*, i.e., the difference between the expected rewards received during training, and the expected rewards under an optimal policy. The analysis is performed for processes with a fixed time horizon, and the criterion of total (non-discounted) rewards accumulated up to that time horizon. This differs from our focus on infinite horizon processes with discounted costs, and the ability to exploit system models for flexible and efficient data generation.

2 Euclidean MDPs

We start by introducing our continuous state space system model. The definitions in this and the following section mostly follow [18], but with a few simplifications that only incur a loss of non-essential generality.

Definition 1 (Euclidean Markov Decision Processes). *A Euclidean Markov decision process (EMDP) is a tuple $\mathcal{M} = (\mathcal{S}, Act, T, \mathcal{C})$ where:*

- $\mathcal{S} \subseteq \mathbb{R}^K$ *is a compact subset of the* K*-dimensional Euclidean space equipped with the Borel* σ*-algebra* $\mathcal{B}^K$*.*
- *Act is a finite set of actions,*
- $T : \mathcal{S} \times Act \times \mathcal{B}^K \to [0,1]$ *defines for every* $a \in Act$ *a transition kernel on* $(\mathcal{S}, \mathcal{B}^K)$*, i.e.,* $T(s,a,\cdot)$ *is a probability distribution on* $\mathcal{B}^K$ *for all* $s \in \mathcal{S}$*, and* $T(\cdot, a, B)$ *is measurable for all* $B \in \mathcal{B}^K$*.*
- $\mathcal{C} : \mathcal{S} \times Act \to [0, c_{\max}]$ *is a cost-function for state-action pairs, such that for all* $a \in Act$*:* $\mathcal{C}(\cdot, a)$ *is measurable, and* $c_{\max}$ *is a global upper bound on costs.*

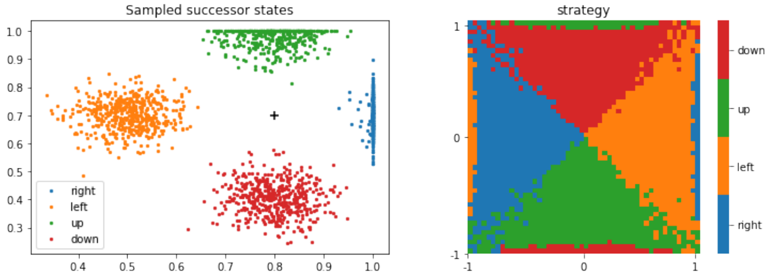

Fig. 1: Sampled successor states and strategy in Example 1 and Example 2

Example 1. The following toy example describes a moving agent in a square in the 2d-plane. Let $\mathcal{S} = [-1,1] \times [-1,1]$, $Act = \{\mathit{right}, \mathit{left}, \mathit{up}, \mathit{down}\}$. For $(x,y) \in \mathcal{S}$, let $T((x,y), \mathit{right}, \cdot)$ be $\mathit{clip}((x,y) + (0.3, 0) + N(\mathbf{0}, 0.003))$, where $N(\mathbf{0}, 0.003))$ is a Gaussian noise vector with zero mean and diagonal covariance matrix with values 0.003, and $\mathit{clip}(x) := \max\{-1, \min\{1, x\}\}$ ensures that the result stays in $\mathcal{S}$. The transition probabilities given the other three actions are defined analogously. Figure 1 on the left shows for the state $s = (0.8, 0.7)$ (marked by a black +) samples of 500 successor states according to $T(s, a, \cdot)$ for each $a \in Act$. The cost function

$$\mathcal{C}((x, y), \mathit{right}) = 2 * (x^2 + y^2) + 0.6(1 - x) \tag{1}$$

consists of two elements: a cost proportional to the squared Euclidean distance to the origin, and a cost that is inversely proportional to the effectivenes of the action *right*: at $x = 1$ it is impossible to move to the right, and the cost component $0.6(1-x)$ here is zero. The cost then linearly increases for decreasing x. The costs for actions *left*,*up*,*down* are defined analogously with the same $2 * (x^2 + y^2)$ term, but with $(1 - x)$ replaced by $(1 + x), (1 - y)$ and $(1 + y)$, respectively.

We are mostly concerned with EMDPs that satisfy the following continuity conditions. In this definition we denote with d_{tv} the total variation distance between distributions.

Definition 2 (Continuous EMDP). *A Euclidean MDP $\mathcal{M}$ is* continuous *if*

- *For each $\epsilon > 0$ there exists $\delta > 0$, such that for all $s, s' \in \mathcal{S}$, $a \in Act$: $\parallel s - s' \parallel < \delta \Rightarrow d_{\mathrm{tv}}(T(s, a, \cdot), T(s', a, \cdot)) \leq \epsilon$.*
- *$\mathcal{C}(\cdot, a)$ is continuous on $\mathcal{S}$ for all $a \in Act$.*

The EMDP of Example 1 is continuous. A *run* π of an MDP is a sequence of alternating states and actions $s_0 a_0 s_1 a_1 s_2 a_2 \ldots$. Let $\lambda \in [0, 1]$ be a *discount factor*. The *discounted cost* of a run is

$$\mathcal{C}_\lambda(\pi) := \sum_{i \geq 0} \lambda^i \mathcal{C}(s_i, a_i) \leq \frac{1}{1 - \lambda} c_{max}. \tag{2}$$

We here allow undiscounted cost as the borderline case $\lambda = 1$, in which case the right-hand side of (2) is ∞.

Definition 3 (Strategy). *A (memoryless,stationary) strategy for an MDP $\mathcal{M}$ is a function $\sigma : \mathcal{S} \to Act$, mapping states to actions, such that for every $a \in Act$ the set $\{s \in \mathcal{S} | \sigma(s) = a\}$ is measurable.*

Example 2. (Example 1 continued). Figure 1 on the right shows a strategy for the EMDP of Example 1. This strategy was learned using the partitioning approach described in Section 3, and is constant on small (measurable) squares that partition the state space $\mathcal{S}$ (see Example 5 below for more details). The strategy is for a cost with discount factor $\lambda = 0.6$, and consists of trying to move into the middle of the state space, except for positions very close to the boundaries, where it is preferable to minimize the short-term action-related costs over the long-term benefit of minimizing the state-related cost in the future.

If a run π is generated according to a given strategy σ, then it is fully characterized by the underlying path $\bar{\pi}$. An initial state $s_0 = s$ together with a strategy σ defines a probability distribution $P_{s,\sigma}$ over runs (see [18] for details).

Definition 4 (Expected Cost). *Let $s \in \mathcal{S}$. The expected cost at s under strategy σ is the expectation of $\mathcal{C}_\lambda$ under the distribution $P_{s,\sigma}$, denoted $\mathbb{E}_\sigma(\mathcal{C}_\lambda, s)$. The expected cost at state s then is defined as*

$$\mathbb{E}(\mathcal{C}_\lambda, s) := \inf_\sigma \mathbb{E}_\sigma(\mathcal{C}_\lambda, s) \leq \frac{1}{1-\lambda} c_{\max}. \tag{3}$$

3 Approximations Induced by Partitions

Let $\mathcal{A} = \{\nu_1, \ldots, \nu_{|\mathcal{A}|}\} \subset 2^{\mathcal{S}}$ be a finite partition of $\mathcal{S}$. We call an element $\nu \in \mathcal{A}$ a *region* and shall assume that each such ν is Borel measurable. For $s \in \mathcal{S}$ we denote by $[s]_\mathcal{A}$ the unique region $\nu \in \mathcal{A}$ such that $s \in \nu$. The *diameter* of a region is $\delta(\nu) := \sup_{s,s' \in \nu} \parallel s - s' \parallel$, and the diameter of a partition $\mathcal{A}$ is defined as $\delta(\mathcal{A}) := \max_{\nu \in \mathcal{A}} \delta(\nu)$. We say that a partition $\mathcal{B}$ refines a partition $\mathcal{A}$ if for every $\nu \in \mathcal{B}$ there exist $\mu \in \mathcal{A}$ with $\nu \subseteq \mu$. We write $\mathcal{A} \sqsubseteq \mathcal{B}$ in this case. The set of probability distributions on $\mathcal{A}$ is denoted $\Delta\mathcal{A}$.

3.1 Induced Imprecise Markov Decision Process

An EMDP $\mathcal{M}$ together with a partition $\mathcal{A}$ of $\mathcal{S}$ defines an *Imprecise Markov Decision Processes (IMDPs)* in the following sense:

Definition 5. *Let $\mathcal{M}(\mathcal{S}, Act, T, \mathcal{C})$ be an EMDP, and $\mathcal{A}$ a partition of $\mathcal{S}$. For $(\nu, a) \in \mathcal{A} \times Act$ let*

$$T\mathcal{C}_\mathcal{A}(\nu, a) := \{((T(s, a, \nu'))_{\nu' \in \mathcal{A}}, \mathcal{C}(s, a)) | s \in \nu\} \subseteq \Delta\mathcal{A} \times [0, c_{\max}], \tag{4}$$

and let $T\mathcal{C}^*_\mathcal{A}(\nu, a)$ be the convex closure of $T\mathcal{C}_\mathcal{A}(\nu, a)$ (by convex closure we here mean closure both under convex combinations, and in the topological sense). The Imprecise Markov Decision Processes (IMDP) *induced by $\mathcal{M}$ and $\mathcal{A}$ then is the tuple $\mathcal{M}_\mathcal{A} = (\mathcal{A}, Act, (T\mathcal{C}^*_\mathcal{A}(\nu, a))_{(\nu,a) \in \mathcal{A} \times Act})$*

The set $T\mathcal{C}_\mathcal{A}(\nu, a)$ contains all the pairs of successor state distributions (over the regions in $\mathcal{A}$), and cost values for the $s \in \nu$. Thus, given that the true state s of the EMDP lies in region ν, we know that the next state distribution and cost will be given by a pair in this set. In order to also be able to model random selections of a state $s \in \nu$ by an *adversary* in the sense of the following definition, we also allow probability/cost combinations that only lie in the convex closure of this set. Finally, we take the topological closure in order to ensure that the optimization problems defined in (6) and (7) below have a solution.

An IMDP is turned into a conventional MDP by an adversary that resolves the non-determinism in the definition of $T\mathcal{C}^*$:

Definition 6 (Adversary, Expected cost). *An* adversary *α for an IMDP consists of a function*

$$\alpha : \ (\nu, a) \mapsto (\alpha_T(\nu, a), \alpha_C(\nu, a)) \in T\mathcal{C}^*_\mathcal{A}(\nu, a) \tag{5}$$

where $\alpha_T(\nu, a) \in \Delta\mathcal{A}$ are transition probabilities, and $\alpha_C \in [0, c_{\max}]$ is a cost value. The adversary α turns the IMDP $\mathcal{M}_{\mathcal{A}}$ into a standard finite state MDP, which we denote $\mathcal{M}_{\mathcal{A},\alpha}$.

A strategy *σ for an IMDP is, as usual, a mapping $\sigma : \mathcal{A} \rightarrow Act$. A strategy σ, an adversary α, and an* initial state *ν_0 define a probability distribution $P_{\nu_0,\sigma,\alpha}$ over runs $\pi = \nu_0, a_0, \nu_1, a_1, \ldots,$ and hence the expectation over the discounted cost $\sum_{i \geq 0} \lambda^i \alpha_C(\nu_i, a_i)$, which we denote by $\mathbb{E}_{\sigma,\alpha}(\mathcal{C}_\lambda, \nu_0)$.*

Two special adversaries are those that lead to minimal and maximal expected costs under optimal strategies.

Definition 7. *A* min-adversary $\alpha^{\min}$ *is any adversary that for all $\nu \in \mathcal{A}$ satisfies*

$$\min_\sigma \min_\alpha \mathbb{E}_{\sigma,\alpha}(\mathcal{C}_\lambda, \nu) = \min_\sigma \mathbb{E}_{\sigma,\alpha^{\min}}(\mathcal{C}_\lambda, \nu). \tag{6}$$

Similarly, a max-adversary $\alpha^{\max}$ *is any adversary for which*

$$\min_\sigma \max_\alpha \mathbb{E}_{\sigma,\alpha}(\mathcal{C}_\lambda, \nu) = \min_\sigma \mathbb{E}_{\sigma,\alpha^{\max}}(\mathcal{C}_\lambda, \nu). \tag{7}$$

Due to our closure requirements for $T\mathcal{C}^*_{\mathcal{A}}$, min- and max-adversaries will always exist. A min-adversary can be seen as a cooperative agent that helps to minimize the cost (and thus is not really an adversary). A max-adversary, on the other hand, has an objective that is opposite to that of the agent controlling σ. Note, too, that according to (7) the max-adversary is conditional on the strategy σ, which represents the worst case scenario, because

$$\min_\sigma \max_\alpha \mathbb{E}_{\sigma,\alpha}(\mathcal{C}_\lambda, \nu) \geq \max_\alpha \min_\sigma \mathbb{E}_{\sigma,\alpha}(\mathcal{C}_\lambda, \nu),$$

where the right-hand side represents a scenario where σ can be chosen conditioned on a given adversary α.

For any adversary α we define in analogy with (3)

$$\mathbb{E}_\alpha(\mathcal{C}_\lambda, \nu) := \min_\sigma \mathbb{E}_{\sigma,\alpha}(\mathcal{C}_\lambda, \nu). \tag{8}$$

By definition, then for all $\nu \in \mathcal{A}$ and α:

$$\mathbb{E}_{\alpha^{min}}(\mathcal{C}_\lambda, \nu) \leq \mathbb{E}_\alpha(\mathcal{C}_\lambda, \nu) \leq \mathbb{E}_{\alpha^{max}}(\mathcal{C}_\lambda, \nu) \tag{9}$$

Furthermore, Theorem 3 in [18] showed that $\mathbb{E}_{\alpha^{min}}, \mathbb{E}_{\alpha^{max}}$ also bound the cost of the underlying EMDP: for all $s \in \mathcal{S}$:

$$\mathbb{E}_{\alpha^{min}}(\mathcal{C}_\lambda, [s]_{\mathcal{A}}) \leq \mathbb{E}(\mathcal{C}_\lambda, s) \leq \mathbb{E}_{\alpha^{max}}(\mathcal{C}_\lambda, [s]_{\mathcal{A}}) \tag{10}$$

(the theorem and proof in [18] were for un-discounted case $\lambda = 1$; the result for the discounted case follows by a simplified modification of the same arguments).

A key question now is whether the $\mathbb{E}_{\alpha^{min}}, \mathbb{E}_{\alpha^{max}}$ bounds also become tight when the granularity of the partition is increased. For the following discussion we need to make the partition defining the IMDP and the resulting expected costs explicit in the notation by writing $\mathbb{E}_{\mathcal{A},\alpha}$ for $\mathbb{E}_\alpha$ in the IMDP induced by $\mathcal{A}$. We then consider a sequence of partition refinements $\mathcal{A}_0 \sqsubseteq \mathcal{A}_1 \sqsubseteq \cdots \sqsubseteq \mathcal{A}_i \sqsubseteq \cdots$ with $\lim_{i\to\infty} \delta(\mathcal{A}_i) = 0$ and ask whether for all $s \in \mathcal{S}$

$$\lim_{i\to\infty} (\mathbb{E}_{\mathcal{A}_i,\alpha^{max}}(\mathcal{C}_\lambda, [s]_{\mathcal{A}_i}) - \mathbb{E}_{\mathcal{A}_i,\alpha^{min}}(\mathcal{C}_\lambda, [s]_{\mathcal{A}_i})) = 0. \tag{11}$$

In [18] it was conjectured that (11) holds for un-discounted costs. This conjecture turns out to be false, however, as the following example shows.

Example 3. Let $\mathcal{S} = [0,1]$, and Act only contain a single action a. For $s \in \mathcal{S}$ let $T(s,a) = \frac{s}{2}\mathcal{U}_{[0,s/2]} + (1-\frac{s}{2})\mathbf{1}_0$, where $\mathcal{U}_{[0.s/2]}$ stands for the uniform distribution on $[0, s/2]$, and $\mathbf{1}_0$ is the point-mass on 0. Let $\mathcal{C}(s,a) = s$. The resulting EMDP is continuous in the sense of Definition 2, and the expected un-discounted cost $\mathbb{E}(\mathcal{C}_1, s)$ is finite for all s, because with probability 1 the cost of the $t+1$st step is at most $1/2$ the cost of the tth step.

Now consider partitions $\mathcal{A}_i = \{[0, 1/i[, [1/i, 2/i[, \ldots, [(i-1)/i, 1]\}$ $(i \geq 1)$. In this case it is immediate that the min-adversary α^{min} is defined by $\alpha_T^{min}([h/i, (h+1)/i[, a) = T(h/i, a, \cdot)$ and $\alpha_C^{min}([h/i, (h+1)/i[, a) = \mathcal{C}(h/i, a)$, whereas α^{max} is defined by $\alpha_T^{max}([h/i, (h+1)/i[, a) = T((h+1)/i, a, \cdot)$, $\alpha_C^{max}([h/i, (h+1)/i[, a) = \mathcal{C}((h+1)/i, a)$. Then, under α^{max}, the cost at each transition step is at least $1/i > 0$, and, thus, $\mathbb{E}_{\mathcal{A}_i,\alpha^{max}}(\mathcal{C}_1, s) = \infty$ for all i and all s.

This counter-example relies crucially on the possibility of infinite cost for un-discounted costs. As we now show, (11) is guaranteed to hold for discounted costs $\mathcal{C}_\lambda$ with $\lambda < 1$.

Theorem 1. *Let $\mathcal{M}$ be a continuous EMDP, and $\mathcal{A}_0 \sqsubseteq \mathcal{A}_1 \sqsubseteq \cdots \sqsubseteq \mathcal{A}_i \sqsubseteq \cdots$ with $\lim_{i\to\infty} \delta(\mathcal{A}_i) = 0$. When $\lambda < 1$, then (11) holds.*

Proof. For $N \geq 0$ we define the *truncated expected cost* $\mathbb{E}^N$ by taking the sum in (2) only over $i = 0, \ldots, N$. For each $\lambda < 1$ and each $\epsilon > 0$ there then exists an $N \geq 0$ such that

$$0 \leq \mathbb{E}_\cdot(\mathcal{C}_\lambda, \cdot) - \mathbb{E}_\cdot^N(\mathcal{C}_\lambda, \cdot) < \epsilon, \tag{12}$$

where these bounds apply uniformly to all expectations both in the EMDP $\mathcal{M}$, the induced IMPDPs $\mathcal{M}_{\mathcal{A}_i}$, for all strategies and adversaries, and for all states, respectively regions. According to Theorem 4 of [18] there exists a $\delta > 0$, such that for all partitions $\mathcal{A}$ with $\delta(\mathcal{A}) \leq \delta$, all strategies σ defined on $\mathcal{M}_\mathcal{A}$, all pairs of adversaries α_0, α_1, and all $\nu \in \mathcal{A}$:

$$|\mathbb{E}^N_{\mathcal{A},\sigma,\alpha_0}(\mathcal{C}_\lambda, \nu) - \mathbb{E}^N_{\mathcal{A},\sigma,\alpha_1}(\mathcal{C}_\lambda, \nu)| < \epsilon \tag{13}$$

(the theorem and proof in [18] are for the case $\lambda = 1$, but the case $\lambda < 1$ is directly implied by this). Now consider $\mathcal{A}_i$ with $\delta(\mathcal{A}_i) < \delta$. Let $\alpha^{max}, \alpha^{min}$ be the max and min adversaries for $\mathcal{M}_{\mathcal{A}_i}$, and σ^+, σ^- be the strategies minimizing (8) for α^{max}, respectively α^{min}. Then

$$\begin{aligned}\mathbb{E}^N_{\mathcal{A}_i,\sigma^-,\alpha^{min}}(\mathcal{C}_\lambda, \nu) + \epsilon \geq \mathbb{E}^N_{\mathcal{A}_i,\sigma^-,\alpha^{max}}(\mathcal{C}_\lambda, \nu) \geq \\ \mathbb{E}^N_{\mathcal{A}_i,\sigma^+,\alpha^{max}}(\mathcal{C}_\lambda, \nu) \geq \mathbb{E}^N_{\mathcal{A}_i,\sigma^-,\alpha^{min}}(\mathcal{C}_\lambda, \nu),\end{aligned} \tag{14}$$

where the first inequality is due to (13) and the following ones are according to the definitions of $\alpha^{max}, \alpha^{min}, \sigma^+, \sigma^-$. We thus obtain

$$\mathbb{E}^N_{\mathcal{A}_i,\sigma^+,\alpha^{max}}(\mathcal{C}_\lambda,\nu) - \mathbb{E}^N_{\mathcal{A}_i,\sigma^-,\alpha^{min}}(\mathcal{C}_\lambda,\nu) \le \epsilon. \tag{15}$$

Combining (12) and (15) we obtain

$$\mathbb{E}_{\mathcal{A}_i,\sigma^+,\alpha^{max}}(\mathcal{C}_\lambda,\nu) - \mathbb{E}_{\mathcal{A}_i,\sigma^-,\alpha^{min}}(\mathcal{C}_\lambda,\nu) \le 2\epsilon, \tag{16}$$

which implies (11).

Example 4. Consider again the EMDP of Example 3, and the IMDPs defined by partitions $\mathcal{A}_i$. In this simple example one can compute both for the min and the max adversary the transition probabilities $\alpha_T(\nu)(\nu')$ and the costs $\alpha_C(\nu)$ (omitting the action argument, because there is only one action). This results in the full specification of a standard finite state MDP that can be solved by value iteration. Figure 2 shows for $\lambda = 0.9$ the resulting $\mathbb{E}_{\mathcal{A}_i,\alpha^{min}}(\mathcal{C}_\lambda,\cdot)$ and $\mathbb{E}_{\mathcal{A}_i,\alpha^{max}}(\mathcal{C}_\lambda,\cdot)$ functions, with the expected narrowing of the gap between the α^{min} and α^{max} values as the partitions are refined.

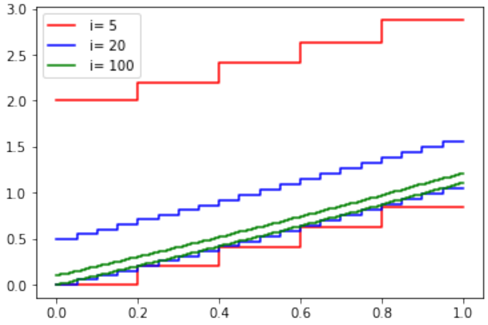

Fig. 2: Cost functions for min and max adversaries at different partition granularities

4 Q-learning

In most cases the MDP $\mathcal{M}_{\mathcal{A},\alpha}$ will not be solvable analytically. Even for quite simple examples of IMDPs and adversaries the transition probabilities $\alpha_T(\nu, a)$ are intractable. However, if we can simulate the underlying EMDP, we can learn cost functions and optimal strategies from observed runs. We here use Q-learning that for a given MDP $\mathcal{M}_{\mathcal{A},\alpha}$ aims to learn for each region-action pair (ν, a) the expected cost of performing action a in state ν, and following an optimal strategy thereafter. The Q-values thus defined are initialized as $Q_0(\nu, a) = 0$ for all ν, a. Based on an observed run $\pi = \nu_0 a_0 \nu_1 a_1 \ldots \nu_t a_t \nu_{t+1} \ldots$ the Q-values are iteratively updated as

$$Q_{t+1}(\nu_t, a_t) = (1 - \beta_t) Q_t(\nu_t, a_t) + \beta_t(\mathcal{C}(\nu_t, a_t) + \lambda \min_{a \in Act} Q_t(\nu_{t+1}, a)), \tag{17}$$

where $\beta_t \in (0, 1)$ is the *learning rate* at iteration t. In order to ensure convergence with high probability, β_t is defined as a decreasing function in the number of times the pair (ν_t, a_t) has been updated at previous time points $s < t$[15]. Thus, β_t is a function of $\pi_{0:t}$. Moreover, during learning, actions a_t are typically not selected according to a fixed, stationary strategy, but according to a strategy that also is designed to collect data from previously under-explored actions. This means that actions are selected according to a histoy-dependent strategy $a_t = \sigma_t(\pi_{0:t})$. We write $\boldsymbol{\sigma} = (\sigma_t)_t$ for such a history dependent strategy. The data may also consist of multiple runs starting at the same or at different initial state. For notational simplicity we take the data to consist of a single long sequence indexed by t.

Our goal is to approximate the true cost function $\mathbb{E}(\mathcal{C}_\lambda, s)$. In view of (10) it would be desirable to learn $\mathbb{E}_{\alpha^{min}}$ and $\mathbb{E}_{\alpha^{max}}$ as strict lower and upper bounds from simulation runs of $\mathcal{M}_{\mathcal{A},\alpha^{min}}$ and $\mathcal{M}_{\mathcal{A},\alpha^{max}}$, and to calibrate the diameter of $\mathcal{A}$ such that these bounds are sufficiently close. However, not only are the transition probabilities defined by α^{min} and α^{max} intractable to compute, but, since $\alpha^{min}, \alpha^{max}$ are only implicitly defined via (6) and (7), we often lack explicit representations of these adversaries that could be used in simulations. We therefore use more tractable adversaries that can be used efficiently in simulations.

The following definition defines adversaries in the form of random state selections.

Definition 8. *For each $(\nu, a) \in \mathcal{A} \times Act$ let $\rho_{\nu,a}$ be a probability distribution on ν. The ρ-*adversary *α^ρ is defined by*

$$\begin{aligned} \alpha^\rho_T(\nu, a)(\nu') &:= \textstyle\int_\nu T(s, a, \nu') d\rho_{\nu,a}(s) \\ \alpha^\rho_C(\nu, a) &:= \textstyle\int_\nu \mathcal{C}(s, a) d\rho_{\nu,a}(s) \end{aligned} \tag{18}$$

The class of ρ-adversaries does not narrow down the class of adversaries significantly: almost every adversary in the sense of Definition 6 is actually a ρ-adversary, with the possible exception of adversaries that pick $(\alpha_T(\nu, a), \alpha_C(\nu, a))$ on the boundary of $T\mathcal{C}^*_{\mathcal{A}}(\nu, a)$. However, the representation in form of a distribution ρ allows us to characterized critical computational properties of adversaries in the form of the following hierarchy of properties:

P1 For all ν, a, one can sample states $s \in \nu$ according to $\rho_{\nu,a}$
P2 In addition to **P1**, the cost values $\alpha^\rho_C(\nu, a)$ can be computed for all ν, a
P3 In addition to **P2**, the transition probabilities $\alpha^\rho_T(\nu, a)(\nu')$ can be computed for all ν, a, ν'.

In all cases we assume that in the underlying EMDP we can sample successor states s' according to the distribution $T(s, a, \cdot)$, and that we can compute the cost $\mathcal{C}(s,a)$. This means that whenever ρ is given by pointmass $\mathbf{1}_s$ for some $s \in \nu$, and the mapping $\nu \mapsto s \in \nu$ is explicitly given, then α^ρ satisfies at least **P2**. This was the case in Example 4 for the min and max adversaries, where furthermore also **P3** was true. When **P3** holds, then the resulting MDP can, in principle, be solved by value iteration.

P2 is important because it is sufficient to support Q-learning: we can sample runs according to the distribution defined by $\mathcal{M}_{\boldsymbol{\sigma},\alpha^\rho}$ for any (possibly non-stationary) strategy $\boldsymbol{\sigma}$: given a current state-action pair (ν, a), we sample a successor state ν' by randomly sampling $s \in \nu$ according to ρ, then sampling $s' \in \mathcal{S}$ according to $T(s, a, \cdot)$, and finally setting $\nu' := [s']_{\mathcal{A}}$. If, according to **P2**, we can at each step also compute the cost value $\alpha^\rho_C(\nu, a)$, then we obtain exactly the data needed for Q-learning.

In many cases the implicit definition of α^{min} or α^{max} will not allow for their explicit representation as an α^ρ, so that even **P1** does not hold. We therefore consider the following more tractable explicit adversaries:

- $\rho_{\nu,a}$ is the uniform distribution on ν, i.e., the Lebesgue measure normalized to a probability distribution. Thus, $\rho_{\nu,a}$ does not depend on a. We then denote α^ρ as α^{mean}, and refer to it as the *mean-adversary*.
- $\rho_{\nu,a}$ is the point mass that puts probability 1 on some state $s^{i\text{-}min} \in \nu$ that has minimal cost $\mathcal{C}(s,a)$ among all states of ν. We then denote α^ρ as $\alpha^{i\text{-}min}$, and refer to it as the *immediate min-adversary*. $\alpha^{i\text{-}min}$ is a heuristic approximations of α^{min} that is based on considering the immediate cost of the next transition only.
- In analogy to $\alpha^{i\text{-}min}$, the *immediate max-adversary* $\alpha^{i\text{-}max}$ is defined.

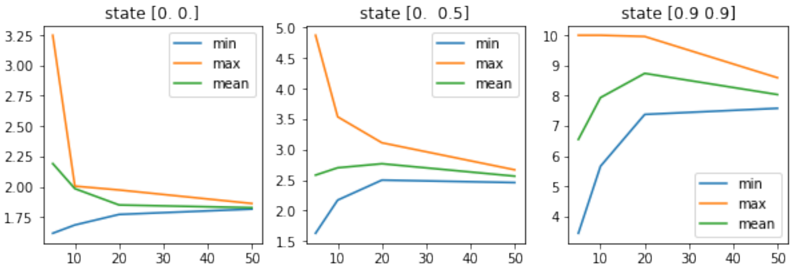

Fig. 3: Cost values (y-axis) at selected states learned from partitions of granularities 5,10,20,50, (x-axis) and the $\alpha^{i\text{-}min}, \alpha^{i\text{-}max}, \alpha^{mean}$ adversaries

Example 5. (Example 2 continued). Similar to Example 3 we consider partitions $\mathcal{A}_i$ in the form of uniform grids with region dimensions $1/i \times 1/i$. For any region ν and all $a \in \mathit{Act}$ it is then easy to identify the states $s \in \nu$ that minimize or maximize the cost $\mathcal{C}(s,a)$, so that $\alpha^{i\text{-}min}$ and $\alpha^{i\text{-}max}$ are given by explicit mappings $\nu \mapsto s \in \nu$, and **P2** holds. However, **P3** does not hold for $\alpha^{i\text{-}min}, \alpha^{i\text{-}max}$ as the integrals defining $\alpha_T^{i\text{-}min}, \alpha_T^{i\text{-}max}$ are over the cumulative distribution function of the Gaussian distribution, for which no closed-form expression exists. Thus, this model with $\alpha_T^{i\text{-}min}, \alpha_T^{i\text{-}max}$ adversaries is amenable to Q-learning, but not to analytic solutions.

Considering the α^{mean} adversary, we find that this, too, satisfies **P2**: clearly we can sample states uniformly in a grid cell ν. Also, due to the simple polynomial cost function (1), the integrals defining $\alpha_C^{mean}(\nu, a)$ can be easily computed. Figure 3 shows for three selected states in $\mathcal{S}$ the learned cost values for $\mathcal{C}_{0.6}$, of the regions $[s]_{\mathcal{A}_i}$ for $i = 5, 10, 20, 50$ under the $\alpha^{i\text{-}min}, \alpha^{i\text{-}max}, \alpha^{mean}$ adversaries. For learning, we use a strategy $\boldsymbol{\sigma}$ that always selects the next action uniformly at random, and where the learning rate β_t is $1/\sqrt{n}$, with n the number of times (ν_t, a_t) has already been updated.

Note that according Theorem 1 all costfunctions converge to the true cost $\mathbb{E}(\mathcal{C}_\lambda, s)$ as $i \to \infty$. However, since $\alpha^{i\text{-}min}, \alpha^{i\text{-}max}$ only are approximations to the true min and max adversaries, there is no strict guarantee that the $\alpha^{i\text{-}min}$ and $\alpha^{i\text{-}max}$ costs always provide a lower and an upper bound on the actual cost. Still, as expected, the cost function for α^{mean} lies in between $\alpha^{i\text{-}min}$ and $\alpha^{i\text{-}max}$, and provides better approximations to the limit for the coarser partitions $\mathcal{A}_i$.

Figure 1 on the right shows the strategy learned for $\mathcal{M}_{\mathcal{A}_{50}, \alpha^{i\text{-}min}}$ (the strategies learned for the $\alpha^{i\text{-}max}$ and α^{mean} adversaries look very similar). According to the learned strategies one should try to move to the middle of the state space, except when the current position is very close to the boundary, in which case it is preferable to perform the very cheap (but otherwise pointless) actions of moving into the boundary.

While satisfied in the preceding example, even **P2** can easily be out of reach. We now consider an approach to approximating Q-learning for $\mathcal{M}_{\mathcal{A},\alpha^\rho}$ when only **P1** is true. For any adversary α^ρ with **P1**, we can approximate simulations of $\mathcal{M}_{\mathcal{A},\alpha^\rho}$ in a Q-learning scenario with a non-stationary strategy $\boldsymbol{\sigma}$ as follows:

- Given the current history $\pi_{0:t}$ and selected action $a_t = \sigma_t(\pi_{0:t})$:
  - sample a state $s \in \nu_t$ according to ρ_{ν_t, a_t}
  - return the cost value $\mathcal{C}_t = \mathcal{C}(s, a_t)$, and sample the next state ν' according to $T(s, a, \nu')_{\nu' \in \mathcal{A}}$.

This simulation generates runs $\nu_0 a_0 \nu_1 a_1 \ldots$ according to the distribution $P_{\nu_0, \boldsymbol{\sigma}}$ defined by $\mathcal{M}_{\mathcal{A},\alpha^\rho}$, $\boldsymbol{\sigma}$, and initial state ν_0. The simulations differ from exact simulations of $\mathcal{M}_{\mathcal{A},\alpha^\rho}$ (or any MDP) in that the observed cost $\mathcal{C}_t$ at step t no longer is a function of the state-action pair (ν_t, a_t). However, one can still perform the Q-learning updates (17) with $\mathcal{C}_t$ instead of $\mathcal{C}(\nu_t, a_t)$. We denote the function defined by these updates at time t as $\tilde{Q}_t$. We now show that in expectation, we obtain the same results as with standard Q-learning from proper simulations of $\mathcal{M}_{\mathcal{A},\alpha^\rho}$.

Theorem 2. *For all $(\nu, a) \in \mathcal{A} \times Act$, $\nu_0 \in \mathcal{A}$, strategies $\boldsymbol{\sigma}$, and $t \geq 0$:*

$$\mathbb{E}_{\nu_0, \boldsymbol{\sigma}}(\tilde{Q}_t(\nu, a)) = \mathbb{E}_{\nu_0, \boldsymbol{\sigma}}(Q_t(\nu, a)) \tag{19}$$

Proof. By induction on t. For $t = 0$ we have $\tilde{Q}_0 \equiv Q_0 \equiv 0$. Assume (19) holds for t. Then we first write

$$\mathbb{E}_{\nu_0,\sigma}(\tilde{Q}_{t+1}(\nu, a)) = \sum_{\bar{\pi}_{0:t} \in \mathcal{A}^t} P_{\nu_0,\boldsymbol{\sigma}}(\bar{\pi}_{0:t}) \mathbb{E}_{\nu_0,\sigma}(\tilde{Q}_{t+1}(\nu, a) | \bar{\pi}_{0:t}), \tag{20}$$

and similarly for $\mathbb{E}_{\nu_0,\sigma}(Q_{t+1}(\nu, a))$. Since the probabilities $P_{\nu_0,\boldsymbol{\sigma}}(\bar{\pi}_{0:t})$ are the same for $\tilde{Q}_t$ and Q_t, it is sufficient to show that for all $\bar{\pi}_{0:t}$

$$\mathbb{E}_{\nu_0,\sigma}(\tilde{Q}_{t+1}(\nu, a) | \bar{\pi}_{0:t}) = \mathbb{E}_{\nu_0,\sigma}(Q_{t+1}(\nu, a) | \bar{\pi}_{0:t}). \tag{21}$$

If $\nu_t \neq \nu$ in $\bar{\pi}_{0:t}$, or $a \neq \sigma_t(\bar{\pi}_{0:t})$, then the Q and $\tilde{Q}$ values of (ν, a) are not updated in the $t + 1$'st iteration, and (19) holds by induction hypothesis.

Assume, then, that $(\nu_t a_t) = (\nu, a)$. We obtain:

$$\begin{aligned} \mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(\tilde{Q}_{t+1}(\nu, a) | \bar{\pi}_{0:t}) = (1 - \beta_t(\bar{\pi}_{0:t})) \mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(\tilde{Q}_t(\nu, a) | \bar{\pi}_{0:t}) + \\ \beta_t(\bar{\pi}_{0:t})(\mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(\mathcal{C}_t | \bar{\pi}_{0:t}) + \lambda \mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(\min_{a' \in Act} \tilde{Q}_t(\nu_{t+1}, a) | \bar{\pi}_{0:t})), \end{aligned} \tag{22}$$

where in the rightmost term the $\tilde{Q}_t(\nu_{t+1}, a)$ now are to be understood as random variables defined by the random next state ν_{t+1}. The distribution of $\mathcal{C}_t$ conditional on $\bar{\pi}_{0:t}$ only depends on $\nu_t = \nu$, and the expectation is

$$\mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(\mathcal{C}_t | \bar{\pi}_{0:t}) = \int_\nu \mathcal{C}(s, a) d\rho_{\nu,a}(s) = \alpha^\rho_C(\nu, a). \tag{23}$$

The distribution for the random ν_{t+1} given $\bar{\pi}_{0:t}$ only depends on ν_t and $a = a_t = \sigma_t(\bar{\pi}_{0:t})$. We can therefore write:

$$\begin{aligned} \mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(\min_{a' \in Act} \tilde{Q}_t(\nu_{t+1}, a) | \bar{\pi}_{0:t}) = \int_\nu \sum_{\nu' \in \mathcal{A}} T(s, a, \nu') \min_{a' \in Act} \tilde{Q}_t(\nu', a') d\rho(s) = \\ \sum_{\nu' \in \mathcal{A}} \min_{a' \in Act} \tilde{Q}_t(\nu', a') \int_\nu T(s, a, \nu') d\rho(s) = \sum_{\nu' \in \mathcal{A}} \alpha^\rho_T(\nu, a)(\nu') \min_{a' \in Act} \tilde{Q}_t(\nu', a'). \end{aligned} \tag{24}$$

Substituting the right-hand sides of (23) and (23) into the right-hand side of (22), and replacing by induction hypothesis $\tilde{Q}_t$ with Q_t everywhere, we obtain

$$\begin{aligned} (1 - \beta_t(\bar{\pi}_{0:t})) \mathbb{E}_{\nu_0,\boldsymbol{\sigma}}(Q_t(\nu, a) | \bar{\pi}_{0:t}) + \\ \beta_t(\bar{\pi}_{0:t})(\alpha^\rho_C(\nu, a) + \lambda \sum_{\nu' \in \mathcal{A}} \alpha^\rho_T(\nu, a)(\nu') \min_{a' \in Act} \tilde{Q}_t(\nu_{t+1}, a) = \\ \mathbb{E}_{\nu_0,\sigma}(Q_{t+1}(\nu, a) | \bar{\pi}_{0:t}). \end{aligned} \tag{25}$$

Example 6. (Example 5 continued). We consider the α^{mean} adversary, and compare the Q-values learned during proper Q-learning with the $\tilde{Q}$ values obtained during our approximation of the Q learning process.

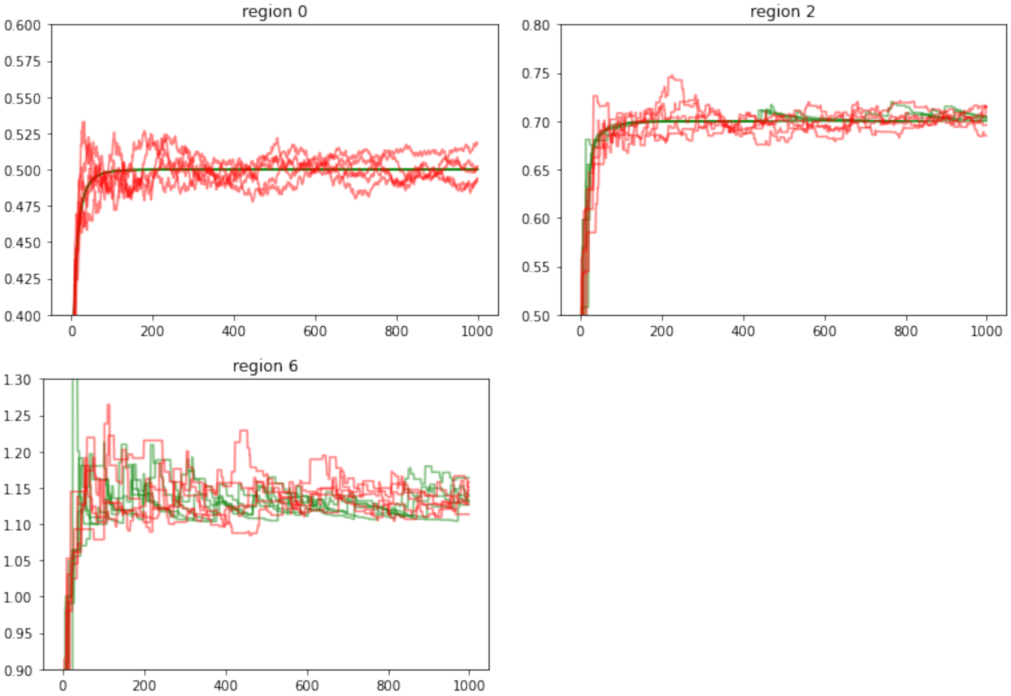

Fig. 4: Learned Q vs. $\tilde{Q}$-values for selected regions

For both exact and approximate Q-learning we perform 5 learning runs. In each learning run we sample 1000 times a random initial state, and then simulate system runs of length 10 (since in this model runs quickly get absorbed in the leftmost region, many short runs are needed to collect informative data). We record the Q and $\tilde{Q}$ values at the end of each short run.

Figure 4 shows for regions $[0, 0.1[$, $[0.2, 0.3[$ and $[0.6, 0.7[$ the developments of the Q (green) and $\tilde{Q}$ (red) values over the course of the 1000 episodes. One can see that in expectation the learned Q and $\tilde{Q}$ coincide, but that the $\tilde{Q}$ values exhibit a larger variance. For the region $[0.6, 0.7[$ further to the right also the Q-values show significant variance. This is because the values here are supported by much fewer datapoints.

5 Bouncing ball

6 Conclusion

Open problem: interleaving with refinement steps

Bibliography

[1] E. Bacci and D. Parker. Probabilistic guarantees for safe deep reinforcement learning. In N. Bertrand and N. Jansen, editors, *Formal Modeling and Analysis of Timed Systems - 18th International Conference, FORMATS 2020, Vienna, Austria, September 1-3, 2020, Proceedings*, volume 12288 of *Lecture Notes in Computer Science*, pages 231–248. Springer, 2020. doi: 10.1007/978-3-030-57628-8_14. URL `https://doi.org/10.1007/978-3-030-57628-8_14`.

[2] E. Bacci and D. Parker. Verified probabilistic policies for deep reinforcement learning. In J. V. Deshmukh, K. Havelund, and I. Perez, editors, *NASA Formal Methods - 14th International Symposium, NFM 2022, Pasadena, CA, USA, May 24-27, 2022, Proceedings*, volume 13260 of *Lecture Notes in Computer Science*, pages 193–212. Springer, 2022. doi: 10.1007/978-3-031-06773-0_10. URL `https://doi.org/10.1007/978-3-031-06773-0_10`.

[3] S. Bøgh, P. G. Jensen, M. Kristjansen, K. G. Larsen, and U. Nyman. Distributed fleet management in noisy environments via model-predictive control. In A. Kumar, S. Thiébaux, P. Varakantham, and W. Yeoh, editors, *Proceedings of the Thirty-Second International Conference on Automated Planning and Scheduling, ICAPS 2022, Singapore (virtual), June 13-24, 2022*, pages 565–573. AAAI Press, 2022. URL `https://ojs.aaai.org/index.php/ICAPS/article/view/19843`.

[4] A. David, P. G. Jensen, K. G. Larsen, A. Legay, D. Lime, M. G. Sørensen, and J. H. Taankvist. On time with minimal expected cost!

In F. Cassez and J. Raskin, editors, *Automated Technology for Verification and Analysis - 12th International Symposium, ATVA 2014, Sydney, NSW, Australia, November 3-7, 2014, Proceedings*, volume 8837 of *Lecture Notes in Computer Science*, pages 129–145. Springer, 2014. doi: 10.1007/978-3-319-11936-6_10. URL `https://doi.org/10.1007/978-3-319-11936-6_10`.

[5] A. David, P. G. Jensen, K. G. Larsen, M. Mikucionis, and J. H. Taankvist. Uppaal stratego. In C. Baier and C. Tinelli, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 21st International Conference, TACAS 2015, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2015, London, UK, April 11-18, 2015. Proceedings*, volume 9035 of *Lecture Notes in Computer Science*, pages 206–211. Springer, 2015. doi: 10.1007/978-3-662-46681-0_16. URL `https://doi.org/10.1007/978-3-662-46681-0_16`.

[6] C. Dehnert, S. Junges, J. Katoen, and M. Volk. A storm is coming: A modern probabilistic model checker. In R. Majumdar and V. Kuncak, editors, *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part II*, volume 10427 of *Lecture Notes in Computer Science*, pages 592–600. Springer, 2017. doi: 10.1007/978-3-319-63390-9_31. URL `https://doi.org/10.1007/978-3-319-63390-9_31`.

[7] A. B. Eriksen, H. Lahrmann, K. G. Larsen, and J. H. Taankvist. Controlling signalized intersections using machine learning. In *World Conference on Transport Research*, volume 48 of *Transportation Research Procedia*, pages 987–997. Science Direct, 2020.

[8] M. A. Goorden, K. G. Larsen, J. E. Nielsen, T. D. Nielsen, M. R. Rasmussen, and J. Srba. Learning safe and optimal control strategies for storm water detention ponds. In R. M. Jungers, N. Ozay, and A. Abate, editors, *7th IFAC Conference on Analysis and Design of Hybrid Systems, ADHS 2021, Brussels, Belgium, July 7-9, 2021*, volume 54 of *IFAC-PapersOnLine*, pages 13–18. Elsevier, 2021. doi: 10.1016/j.ifacol.2021.08.467. URL `https://doi.org/10.1016/j.ifacol.2021.08.467`.

[9] M. A. Goorden, P. G. Jensen, K. G. Larsen, M. Samusev, J. Srba, and G. Zhao. STOMPC: stochastic model-predictive control with uppaal stratego. In A. Bouajjani, L. Holík, and Z. Wu, editors, *Automated Technology for Verification and Analysis - 20th International Symposium, ATVA 2022, Virtual Event, October 25-28, 2022, Proceedings*, volume 13505 of *Lecture Notes in Computer Science*, pages 327–333. Springer, 2022. doi: 10.1007/978-3-031-19992-9_21. URL `https://doi.org/10.1007/978-3-031-19992-9_21`.

[10] T. P. Gros, H. Hermanns, J. Hoffmann, M. Klauck, and M. Steinmetz. Deep statistical model checking. In A. Gotsman and A. Sokolova, editors, *Formal Techniques for Distributed Objects, Components, and Systems - 40th IFIP WG 6.1 International Conference, FORTE 2020, Held as Part of the 15th International Federated Conference on Distributed Computing Techniques, DisCoTec 2020, Valletta, Malta, June 15-19, 2020, Pro-*

ceedings, volume 12136 of *Lecture Notes in Computer Science*, pages 96–114. Springer, 2020. doi: 10.1007/978-3-030-50086-3_6. URL `https://doi.org/10.1007/978-3-030-50086-3_6`.

[11] D. Gross, N. Jansen, S. Junges, and G. A. Pérez. COOL-MC: A comprehensive tool for reinforcement learning and model checking. In W. Dong and J. Talpin, editors, *Dependable Software Engineering. Theories, Tools, and Applications - 8th International Symposium, SETTA 2022, Beijing, China, October 27-29, 2022, Proceedings*, volume 13649 of *Lecture Notes in Computer Science*, pages 41–49. Springer, 2022. doi: 10.1007/978-3-031-21213-0_3. URL `https://doi.org/10.1007/978-3-031-21213-0_3`.

[12] A. Hartmanns and H. Hermanns. The modest toolset: An integrated environment for quantitative modelling and verification. In E. Ábrahám and K. Havelund, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 20th International Conference, TACAS 2014, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2014, Grenoble, France, April 5-13, 2014. Proceedings*, volume 8413 of *Lecture Notes in Computer Science*, pages 593–598. Springer, 2014. doi: 10.1007/978-3-642-54862-8_51. URL `https://doi.org/10.1007/978-3-642-54862-8_51`.

[13] I. R. Hasrat, P. G. Jensen, K. G. Larsen, and J. Srba. End-to-end heat-pump control using continuous time stochastic modelling and uppaal stratego. In Y. A. Ameur and F. Craciun, editors, *Theoretical Aspects of Software Engineering - 16th International Symposium, TASE 2022, Cluj-Napoca, Romania, July 8-10, 2022, Proceedings*, volume 13299 of *Lecture Notes in Computer Science*, pages 363–380. Springer, 2022. doi: 10.1007/978-3-031-10363-6_24. URL `https://doi.org/10.1007/978-3-031-10363-6_24`.

[14] P. Herber, J. Adelt, and T. Liebrenz. Formal verification of intelligent cyber-physical systems with the interactive theorem prover keymaera X. In S. Götz, L. Linsbauer, I. Schaefer, and A. Wortmann, editors, *Proceedings of the Software Engineering 2021 Satellite Events, Braunschweig/Virtual, Germany, February 22 - 26, 2021*, volume 2814 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021. URL `https://ceur-ws.org/Vol-2814/short-A3-2.pdf`.

[15] T. Jaakkola, M. Jordan, and S. Singh. Convergence of stochastic iterative dynamic programming algorithms. *Advances in neural information processing systems*, 6, 1993.

[16] M. Jaeger, P. G. Jensen, K. G. Larsen, A. Legay, S. Sedwards, and J. H. Taankvist. Teaching stratego to play ball: Optimal synthesis for continuous space mdps. In Y. Chen, C. Cheng, and J. Esparza, editors, *Automated Technology for Verification and Analysis - 17th International Symposium, ATVA 2019, Taipei, Taiwan, October 28-31, 2019, Proceedings*, volume 11781 of *Lecture Notes in Computer Science*, pages 81–97. Springer, 2019. doi: 10.1007/978-3-030-31784-3_5. URL `https://doi.org/10.1007/978-3-030-31784-3_5`.

[17] M. Jaeger, P. G. Jensen, K. G. Larsen, A. Legay, S. Sedwards, and J. H. Taankvist. Teaching stratego to play ball: Optimal synthesis for continuous space mdps. In *International Symposium on Automated Technology for Verification and Analysis*, pages 81–97. Springer, 2019.

[18] M. Jaeger, G. Bacci, G. Bacci, K. G. Larsen, and P. G. Jensen. Approximating euclidean by imprecise markov decision processes. In *Leveraging Applications of Formal Methods, Verification and Validation: Verification Principles: 9th International Symposium on Leveraging Applications of Formal Methods, ISoLA 2020, Rhodes, Greece, October 20–30, 2020, Proceedings, Part I 9*, pages 275–289. Springer, 2020.

[19] M. Z. Kwiatkowska, G. Norman, and D. Parker. PRISM: probabilistic symbolic model checker. In T. Field, P. G. Harrison, J. T. Bradley, and U. Harder, editors, *Computer Performance Evaluation, Modelling Techniques and Tools 12th International Conference, TOOLS 2002, London, UK, April 14-17, 2002, Proceedings*, volume 2324 of *Lecture Notes in Computer Science*, pages 200–204. Springer, 2002. doi: 10.1007/3-540-46029-2\_13. URL `https://doi.org/10.1007/3-540-46029-2_13`.

[20] K. G. Larsen, P. Pettersson, and W. Yi. UPPAAL in a nutshell. *Int. J. Softw. Tools Technol. Transf.*, 1(1-2):134–152, 1997. doi: 10.1007/s100090050010. URL `https://doi.org/10.1007/s100090050010`.

[21] K. G. Larsen, M. Mikucionis, M. Muñiz, J. Srba, and J. H. Taankvist. Online and compositional learning of controllers with application to floor heating. In M. Chechik and J. Raskin, editors, *Tools and Algorithms for the Construction and Analysis of Systems - 22nd International Conference, TACAS 2016, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2016, Eindhoven, The Netherlands, April 2-8, 2016, Proceedings*, volume 9636 of *Lecture Notes in Computer Science*, pages 244–259. Springer, 2016. doi: 10.1007/978-3-662-49674-9\_14. URL `https://doi.org/10.1007/978-3-662-49674-9_14`.

[22] K. G. Larsen, A. Legay, G. Nolte, M. Schlüter, M. Stoelinga, and B. Steffen. Formal methods meet machine learning (F3ML). In T. Margaria and B. Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation. Adaptation and Learning - 11th International Symposium, ISoLA 2022, Rhodes, Greece, October 22-30, 2022, Proceedings, Part III*, volume 13703 of *Lecture Notes in Computer Science*, pages 393–405. Springer, 2022. doi: 10.1007/978-3-031-19759-8\_24. URL `https://doi.org/10.1007/978-3-031-19759-8_24`.

[23] A. Platzer and J. Quesel. Keymaera: A hybrid theorem prover for hybrid systems (system description). In A. Armando, P. Baumgartner, and G. Dowek, editors, *Automated Reasoning, 4th International Joint Conference, IJCAR 2008, Sydney, Australia, August 12-15, 2008, Proceedings*, volume 5195 of *Lecture Notes in Computer Science*, pages 171–178. Springer, 2008. doi: 10.1007/978-3-540-71070-7\_15. URL `https://doi.org/10.1007/978-3-540-71070-7_15`.

[24] S. R. Sinclair, S. Banerjee, and C. L. Yu. Adaptive discretization for episodic reinforcement learning in metric spaces. *Proceedings of the ACM on Mea-*

surement and Analysis of Computing Systems, 3(3):1–44, 2019.

[25] Z. Song and W. Sun. Efficient model-free reinforcement learning in metric spaces. *arXiv preprint arXiv:1905.00475*, 2019.