Policy learning in Continuous-Time Markov Decision Processes using Gaussian Processes

Policy learning in Continuous-Time Markov Decision Processes using Gaussian Processes

Ezio Bartocci[a], Luca Bortolussi[b,c,d], Tomas Brázdil[e], Dimitrios Milios[f,*], Guido Sanguinetti[f,1]

[a]*Faculty of Informatics, Vienna University of Technology, Austria*
[b]*Dept. of Maths and Geosciences, University of Trieste, Italy*
[c]*CNR/ISTI, Pisa, Italy*
[d]*Modelling and Simulation Group, Saarland University, Germany*
[e]*Faculty of Informatics, Masaryk University, Czech Republic*
[f]*School of Informatics, University of Edinburgh, UK*
[g]*SynthSys, Centre for Synthetic and Systems Biology, University of Edinburgh, UK*

Abstract

Continuous-time Markov decision processes provide a very powerful mathematical framework to solve policy-making problems in a wide range of applications, ranging from the control of populations to cyber-physical systems. The key problem to solve for these models is to efficiently compute an optimal policy to control the system in order to maximise the probability of satisfying a set of temporal logic specifications. Here we introduce a novel method based on statistical model checking and an unbiased estimation of a functional gradient in the space of possible policies. Our approach presents several advantages over the classical methods based on discretisation techniques, as it does not assume the a-priori knowledge of a model that can be replaced by a black-box, and does not suffer from state-space explosion. The use of a stochastic moment-based gradient ascent algorithm to guide our search considerably improves the efficiency of learning policies and accelerates the convergence using the momentum term. We demonstrate the strong performance of our approach on two examples of non-linear population models: an epidemiology model with no permanent recovery and a queuing system with non-deterministic choice.

Keywords: Continuous-Time Markov Decision Processes, Gaussian Processes, Machine Learning, Policy learning

☆Fully documented templates are available in the elsarticle package on CTAN.

*Principal corresponding author

Email addresses: `ezio.bartocci@tuwien.ac.at` (Ezio Bartocci), `luca@dmi.units.it` (Luca Bortolussi), `xbrazdil@fi.muni.cz` (Tomas Brázdil), `dmilios@staffmail.ed.ac.uk` (Dimitrios Milios), `gsanguin@inf.ed.ac.uk` (Guido Sanguinetti)

1. Introduction

Continuous-time Markov Decision Processes (CTMDPs) [1] are a very powerful modelling tool employed in many decision-making problems that arise in systems featuring both probabilistic and nondeterministic behaviours.

CTMDPs are generally used to solve control and dependability problems in a wide range of applications, including the control of populations (i.e., epidemics [2, 3] and bird flocking [4]), power management [5], queueing systems [6], and cyber-physical systems [7].

A CTMDP [1] extends memoryless, state-based, continuous-time Markov chains (CTMC) with nondeterministic input transitions associated with the *actions* that a *decision maker* (also called *scheduler* or *policy*) can perform in each state. Each action has usually an associated *cost* or *reward.* Once an action is chosen in each state, the modelled system behaves as a CTMC.

CTMDPs provide a valuable mathematical and algorithmic framework to address two important problems such as *model checking* [8] (also called *verification*) and *policy-making* (also called *scheduling* or *planning*).

The *model checking* [9] problem consists in computing the *min/max* probability for a CTMDP to satisfy a temporal requirement of interest for a given class or for all possible schedulers. In model checking the requirement is generally formally expressed as a temporal logic formula [8]. The main target of the current quantitative model checking technology for CTMDPs is the *time-bounded reachability* [9, 10, 11, 12, 13], a property that requires a CTMDP to reach a particular set of states within a prescribed time bound.

A problem related to model checking is *policy-making.* In this case the goal is to devise the optimal sequence of actions (or *policy*) to control the system in order to maximise/minimise the expected cumulative reward/cost or the probability to satisfy a temporal logic specification such as the aforementioned *time-bounded reachability* property.

The optimal scheduling for CTMDP can be either *timed* or *untimed* depending on whether or not the scheduler is aware of the passing of time. Timed optimal scheduling can be further classified in *late* or *early* depending on whether the decision of choosing an action can change while the time passes in a state or it remains unchanged.

Here we consider *randomised time-dependent early schedulers*, and we focus on population models, where the state-space of the CTMDP is represented by a set of integer-valued variables counting how many entities of each kind are in the system. Examples of this class of models include queueing and performance models [13], epidemic scenarios [2, 3], biological systems. Despite their popularity, these models suffer severely from state-space explosion, with the number of states growing exponentially with the number of variables. This reflects on the size of the schedulers: in principle, we would need to store a function of time for each state of the CTMDP, which is unfeasible.

This paper is an extended version of [14]. There we introduced a novel statistical method to compute the lower bounds on the maximum probability for a CTMDP to satisfy a time-bounded reachability property. Our approach uses basis-function regression to compactly encode schedulers and effectively search for an optimal one. First, we leverage the structure of the state space, which can be embedded as a discrete grid in real space, to obtain a continuous relaxation of the problem and consider schedulers defined on such a continuous space. The advantage now is that we can treat time and space uniformly, representing schedulers as continuous functions. This opens up the use of machine learning methods to represent continuous functions as combinations of basis functions, and allows us to define the optimisation problem as a search in such a continuous function space. The second main contribution of [14] was to set up an efficient stochastic gradient ascent search algorithm, which considerably speeds up the search in the space of functions. This is based on a novel algorithm using Gaussian Processes (GPs) and statistical model checking to sample in an unbiased manner the gradient of the functional associating a reachability probability with a randomised scheduler. This method allowed us to effectively learn schedulers that maximise (locally) the reachability probability. In this paper we extend our preliminary work [14] by the following results:

- we have improved the gradient ascent algorithm by including a momentum term in the update of the algorithm to accelerate its convergence;
- we have validated and tested the new approach both on the *epidemiology model with no permanent recovery* presented in previous paper [14] and on a new case study on *queuing system with non-deterministic choice* from [15] and not considered in our previous paper;
- we provide extensive proofs of the main theoretical results of the paper.

Organisation of the paper. In Section 2 we present the related work and in Section 3 we provide the necessary formal background on CTMDPs. In Section 4 we present our algorithm to learn optimal policies using stochastic functional gradient ascent techniques. In Section 5 we demonstrate our algorithm on the two case studies. Finally, we draw our conclusion in Section 6.

2. Related work

Probabilistic model checking algorithms for discrete-time MDPs [16, 17] have been extensively investigated and applied in several case studies. Popular tools such as PRISM [18] provide a user-friendly front-end where the user needs to focus only on the modelling aspects (through the support of very expressive formal specification languages) and a back-end implementing very efficient model checking algorithms enabling the user to perform the automatic analysis of these models by just pressing a button.

However, these tools do not yet support the analysis of CTMDPs, largely because algorithms for CTMDPs are still in their infancy and suffer badly from high computational complexity. CTMDPs are a very popular framework in the control theory and operation research communities [19, 20, 3, 21, 7]. Here the research focuses on finding optimal policies that would maximise rewards or minimise costs in order to reach a certain goal. Theoretical analyses of CTMDPs have a long history in the control community, started by the seminal paper of Miller [20] nearly fifty years ago. However, such analyses were not in general constructive, i.e. they did not provide explicit algorithms to solve computationally the problem.

More recently computer scientists have introduced more practical approaches to handle the analysis for this class of models. In the last decade, there has been considerable effort [22, 9, 10, 11, 12, 13] to provide novel algorithmic techniques for optimal control and model checking of CTMDPs with respect to a time-bounded reachability requirement.

Baier et al. in [9] first introduce a model checking algorithm for time-bounded reachability properties in uniform CTMDPs, a special class of CT-MDPs where the delay time distribution per state visit is the same for all states. The approach in [9] considers only *time-abstract schedulers* a very restricted class of schedulers that have access to the sequence of the visited states but not to the time in which these states have been visited. Later in [23, 24, 12] other authors have also studied the existence of optimal time-abstract schedulers in arbitrary CTMDPs. In our work we consider the more powerful class of *time-dependent schedulers* studied in [25, 26].

This class of schedulers can be further classified in *early* and *late time-dependent schedulers* depending on whether the decision of performing an action is taken upon entering in a state or while the time passes in a state. The standard analysis techniques developed for time-dependent schedulers rely on efficient approximation algorithms [27, 21] that discretise the time bound in small intervals. The number of the intervals is generally quadratic with respect to the time bound divided by the required precision; as such, it can grow very rapidly to an intractable size. All the aforementioned techniques assume the a-priori knowledge of the CTMDP model under investigation and they suffer the state-explosion problem commonly encountered in model checking problems.

Statistical model checking techniques are particularly attractive when the CTMDP model is not a-priori available, even though they may suffer when the property to be verified is a rare-event. A statistical model checking algorithm for discrete-time MDP is presented in [28]. Their approach was based on random search combined with a greedy selection criterion, which is difficult to analyse in terms of convergence properties, and may be practically difficult to tune. While the work of Henriques and colleagues [28] indubitably represents a considerable step forward, random search algorithms become impractical even in spaces of moderate dimension, and their usefulness in an intrinsically infinite-dimensional problem is questionable. Our earlier paper [14] provided a principled solution to this problem by defining a strategy to compute an optimal search direction through an unbiased approximation to the (infinite-dimensional) gradient of the objective function. The availability of an unbiased estimate of the (functional) gradient allows us to improve on the efficiency, and to leverage a rich theory on the convergence of stochastic gradient ascent algorithms. Our approach relies on using Gaussian Processes (GPs), a probability distribution over the space of functions which universally approximates continuous functions. This ability of GPs to provide efficient approximations to intractable functions has been recently exploited in a formal modelling context in a number of publications [29, 30, 31, 32, 33].

Our work is closely related to research in the area of machine learning, where much research has gone on defining good local search methods to learn effective randomised schedulers, for different criteria like time bounded reward and time unbounded discounted reward. These approaches combine simulation with efficient exploration schemes, like gradient ascent [34, 35], path integral policy improvement [36], or the cross entropy method [37], see [38] for a survey. In a more recent work, statistical model checking and learning-based methods were used to verify unbounded properties [39]. Our approach differs in two main directions: firstly, we are interested in complex rewards associated with trajectories of the system, i.e., reachability probabilities. Secondly, we work directly in continuous time, which prevents the use of simple finite-dimensional gradient ascent methods. In particular, the GP-based method of defining a stochastic gradient ascent algorithm is novel, to the best of our knowledge.

3. Preliminaries

In this section we present continuous-time Markov decision processes (CTMDPs) . We will start by introducing the general CTMDPs, and the reachability probability problem we tackle in this paper. Then, we introduce Population CTMDPs (PCTMDPs), a dialect of CTMDPs which describe systems of interacting populations of agents. Focussing on PCTMDPs in the paper permits us to introduce compact representations for schedulers and leverage them in the search algorithms.

Definition 1. A CTMDP is a tuple $\mathcal{M} = (S, \mathcal{A}, R, s_0)$, where S is a finite set of *states*, $\mathcal{A}$ is a finite set of *actions*, $R : S \times \mathcal{A} \times S \to \mathbb{R}_{\geq 0}$ is the *rate function*, and $s_0 \in S$ is the initial state.

An action $a \in \mathcal{A}$ is *enabled* in a state $s \in S$ if there is a state $s' \in S$ such that $R(s, a, s') > 0$. We call $\mathcal{A}(s)$ the set of enabled actions in s. A *continuous-time Markov chain (CTMC)* is a CTMDP where every $\mathcal{A}(s)$ is a singleton.

We define $E(s, a) = \sum_{s'} R(s, a, s')$ the *exit* rate from a state s when an action a is chosen. We also let $P(s, a, s') = R(s, a, s')/E(s, a)$ be the probability of jumping from s to s' if a is selected.

Intuitively, a run of CTMDP starts in a state s_0 and proceeds as follows: Assume that the CTMDP is currently in a state s_i. First, an action a_i is selected, then the CTMDP waits for a delay t_i randomly chosen according to an exponential distribution with the exit rate $E(s_i, a_i)$, and then a next state s_{i+1} is chosen randomly with the probability $P(s_i, a_i, s_{i+1})$. This produces a run $s_0 a_0 t_0 s_1 a_1 t_1 \cdots$.

In order to obtain a complete semantics, we need to specify how the actions are selected at every step. Obviously, in CTMC, only a single action is enabled in each state. In CTMDP, actions need to be chosen by a scheduler defined as follows.

Definition 2. An *scheduler* is a function $\sigma : \mathbb{R}_{\geq 0} \times S \times \mathcal{A} \to [0,1]$ which to every $t \in \mathbb{R}_{\geq 0}$, $s \in S$ and $a \in \mathcal{A}(s)$ assigns a probability $\sigma(t,s,a)$ that the action a is chosen in s at time t. We demand that

- $\sigma(t,s,a)$ is measurable as a function of t over $\mathbb{R}_{\geq 0}$ for all $s \in S$ and $a \in \mathcal{A}(s)$,
- $\sum_{a \in \mathcal{A}(s)} \sigma(t,s,a) = 1$ for all $t \in \mathbb{R}_{\geq 0}$ and $s \in S$.

A scheduler σ is *deterministic* if for every $t \in \mathbb{R}_{\geq 0}$, $s \in S$ and $a \in \mathcal{A}$ we have that $\sigma(t,s,a) \in \{0,1\}$. We denote by Σ and Σ_D the sets of all schedulers and all deterministic schedulers, respectively.

Remark 1. Notice that the intuitive CTMDP semantics define above correspond to a restricted class of schedulers known as *early schedulers*. An early scheduler has the following property: whenever an execution of the CTMDP enters into a state s at time t, the scheduler chooses an action and commits to it. It cannot be changed while the system remains in state s, in contrast with late schedulers, that can change action while in a state.

Once a scheduler σ and an initial state s are fixed, we obtain the unique probability measure $\mathbb{P}^{\mathcal{M},s}_{\sigma}$ over the space of all runs initiated in s using standard definitions [40].

Time-Bounded Reachability. Let us first introduce some notation. Given a run $w = s_0 a_0 t_0 s_1 a_1 t_1 \ldots$ and $t \in \mathbb{R}_{\geq 0}$, we denote by $w_{\downarrow t}$ the current state of w at the time instant t, that is $w_{\downarrow t} = s_k$ for some $k \geq 0$ satisfying $\sum_{i=0}^{k-1} t_i \leq t$ and $\sum_{i=0}^{k} t_i \geq t$.

Let $G \subset S$ be a set of goal states and let $I = [b_1, b_2] \subseteq [0, \infty)$ be a closed interval. Denote by $\diamond_I G$ the set of all runs $w = s_0 a_0 t_0 s_1 a_1 t_1 \ldots$ that visit G within the time interval I, i.e. there is $t \in I$ satisfying $w_{\downarrow t} \in G$. Denote by $\mathbb{P}^{\mathcal{M},s}_{\sigma}(\diamond_I G)$ the corresponding probability. Our goal is to maximize $\mathbb{P}^{\mathcal{M},s}_{\sigma}(\diamond_I G)$, i.e. compute a scheduler σ^* satisfying

$$\mathbb{P}^{\mathcal{M},s}_{\sigma^*}(\diamond_I G) \quad = \quad \sup_{\sigma \in \Sigma} \mathbb{P}^{\mathcal{M},s}_{\sigma}(\diamond_I G)$$

We say that such a scheduler σ^* is *optimal.*

Proposition 1 ([40]). *There always exists an optimal scheduler. The following proposition has been proved in [40]; we include a sketch of the argument* for completeness.

Proof (Sketch). In general, a *history-dependent randomized (HR)* scheduler π is a (measurable) function which takes a path (a history) $h = s_0 a_0 t_0 s_1 a_1 t_1 \cdots s_n$ and returns a probability distribution on actions of $\mathcal{A}(s_n)$. We write $\pi(h,a)$ to denote the probability that a is taken after the history h. Our schedulers, as defined in Defintion 2, are called *total time-positional randomized (TTPR)* schedulers. If the scheduler always assigns the probability one to exactly one action, we say that it is *deterministic*, which gives us the classes of *history-dependent deterministic* (HD) and *total time-positional deterministic* (TTPD) schedulers.

The argument can be (roughly) summarized as follows: Let us add a counter to the state-space, i.e., states are now of the form (s,k) where s is a state of the original CTMDP $\mathcal{M}$ and k is the number of steps the process made from the beginning. The CTMDP $\mathcal{M}$ is simulated in the first component and the number of steps counted in the other one, up to the moment when a threshold $n+1$ is reached and from this moment on the counter stays at value $n+1$ forever. The new goal states are the pairs (s,k) where s is a goal state in $\mathcal{M}$ and $k \leq n$. This gives us a new CTMDP $\mathcal{M}_n$. Note that every HR scheduler in $\mathcal{M}_n$ can be easily transformed into a HR scheduler in $\mathcal{M}$ by taking a projection on the first component.

Denote by $V^n((s,k),t)$ the maximum probability of reaching a goal state in $\mathcal{M}_n$ from (s,k) within the time interval $I - t = [\max(0, b_1 - t), \max(0, b_2 - t)]$ where $I = [b_1, b_2]$. Values $V^n((s,k),t)$ in the CTMDP $\mathcal{M}_n$ can be computed using backward induction (on k) as follows: Clearly, $V^n((s,n{+}1),t)$ is 0 for all t. We now use this condition as the starting step of a backward recursion. Assume that we already have $V^n((s,k+1),t)$. Now $V^n((s,k),t)$ is the supremum of the following expression over all probability distributions d over the set of actions $\mathcal{A}$:

$$\sum_a d(a) \cdot \sum_{s'} \int_0^\infty R(s,a,s')e^{-R(s,a,s')t'} V^n((s',k+1),t+t')dt'$$

(Intuitively, first a is chosen with probability $d(a)$, then time delay t' is chosen from the exponential distribution together with the next state $(s',k+1)$, finally we proceed optimally from $(s',k+1)$ after time $t+t'$, which means that we reach a goal state with probability $V^n((s',k+1),t+t')$.) Clearly,

$$V^n((s,k),t) = \max_{a \in \mathcal{A}(s)} \sum_{s'} \int_0^\infty R(s,a,s')e^{-R(s,a,s')t'} V^n((s',k+1),t+t')dt'$$

(The maximum exists because $\mathcal{A}(s)$ is a finite set.) Denote by π^n a TTPD scheduler in $\mathcal{M}_n$ which in every state (s,k) and time t chooses an action a maximizing

$$\sum_{s'} \int_0^\infty R(s,a,s')e^{-R(s,a,s')t'} V^n((s',k+1),t+t')dt'$$

Such π^n is optimal in $\mathcal{M}_n$.

Observe that for every k and every t we have $\lim_{n\to\infty} V^n(s,k,t) = V(s,t)$ where $V(s,t) = \sup_{\sigma\in\Sigma} \mathbb{P}_\sigma^{\mathcal{M},s}(\diamond_{I-t} G)$.

Now let m be large enough so that the probability of making more than m steps in at most b_2 time units is less than ε.

It follows that for every $m' > 2m$, the scheduler $\pi^{m'}$, which is optimal in $\mathcal{M}_{m'}$, is ε-optimal in $\mathcal{M}$ (which means that it satisfies $\diamond_I G$ with probability ε-close to the maximum value).

Let $m' > 2m$ be large enough so that for all $k \leq 2m$ and all $t \leq b_2$ we have that

$$argmax_a \sum_{s'} \int_0^\infty R(s,a,s') e^{-R(s,a,s')t'} V^{m'}((s',k+1),t+t')dt' =$$

$$argmax_a \sum_{s'} \int_0^\infty R(s,a,s') e^{-R(s,a,s')t'} V(s',t+t')dt'$$

It follows that a scheduler which always chooses an action from

$$argmax_a \sum_{s'} \int_0^\infty R(s,a,s') e^{-R(s,a,s')t'} V(s',t+t')dt'$$

behaves similarly to $\pi^{m'}$ and hence is ε-optimal. As $\varepsilon > 0$ was chosen arbitrarily and the above choice depends only on s and t, we obtain the desired optimal TTPD scheduler. □

Remark 2. In this paper we concentrate on the problem of maximizing $\mathbb{P}_\sigma^{\mathcal{M},s}(\diamond_I G)$, i.e., computing a scheduler σ^* satisfying

$$\mathbb{P}_{\sigma^*}^{\mathcal{M},s}(\diamond_I G) \quad = \quad \sup_{\sigma \in \Sigma} \mathbb{P}_\sigma^{\mathcal{M},s}(\diamond_I G)$$

All our results and methods can be easily extended (by substituting suprema with infima) to *minimization* of $\mathbb{P}_\sigma^{\mathcal{M},s}(\diamond_I G)$. This is because we will adopt a gradient-based approach to the optimisation which is clearly symmetric (see below).

An equivalent problem to the minimization of $\mathbb{P}_\sigma^{\mathcal{M},s}(\diamond_I G)$ is to maximise a time-bounded safety property $\square_I G$, requiring the CTMDP to remain in a region G during the time-interval I. In this case, we have that $\mathbb{P}_{\sigma^*}^{\mathcal{M},s}(\square_I G) = \mathbb{P}_{\sigma^*}^{\mathcal{M},s}(\neg \diamond_I (S \setminus G)) = \inf_{\sigma \in \Sigma} \mathbb{P}_\sigma^{\mathcal{M},s}(\diamond_I (S \setminus G))$.

3.1. Population CTMDPs

In this work, we will consider CTMDPs modelled in a special way, reminiscent of population processes which are very common in performance modelling, epidemiology, systems biology. The basic idea is that we will have populations of agents, belonging to one or more classes, that can interact together and thus evolve in time. Individual agents are typically indistinguishable, hence the state of the system can be described by a set of variables counting the amount of agents of each kind in the system. A non-deterministic action in this context typically represents an action of a global controller, enforcing a policy controlling the system, or effects on the environment. Examples of population processes are presented in Section 5. In particular, we discuss an epidemic scenario in which agents can get infected and spread the contagion, and where the global action represents public health policy to treat infected people, speeding up their recovery though introducing a small risk of death. We also discuss a resource sharing scenario, in which a server chooses non-deterministically jobs from different queues.

In this paper, we focus on population CTMDPs for two reasons:

- they are an expressive modelling framework encompassing many practical interesting scenarios;
- they allow us to naturally approximate in a continuous way the state space, leading to efficient policy representation and an efficient algorithm to search in the policy space.

More formally, we will describe a Population CTMDP (PCTMDP), extending population processes [41, 42], as a tuple $(\vec{X}, \mathcal{T}, \mathcal{A}, \vec{s}_0)$, where:

- $\vec{X} = X_1, \ldots, X_n$ is a vector of population variables, $X_i \in \mathbb{N}$, which we assume take values on $S = \mathbb{N}^n \cap K$, where K is a compact subset of $\mathbb{R}^n$ (hence S is finite);
- $\vec{s_0} \in S$ is the initial state;
- $\tau \in \mathcal{T}$ is the set of transitions, of the form $(a, \vec{v}_\tau, f_\tau(\vec{X}))$, where a is an action from the set $\mathcal{A}$, $\vec{v}_\tau$ is an update vector, specifying that the state after the execution of a transition in state $\vec{s}$ is $\vec{s} + \vec{v}_\tau$, and $f_\tau(\vec{s})$ is the state-dependent rate function, assigning a rate with each state $\vec{s} \in S$.

The idea of this model is that in each state an action a is chosen, and then the model evolves by a race condition between transitions guarded by the action a. If a transition is enabled by all possible actions, we can either specify a copy of it guarded by each model action a, or use the notation $(*, \vec{v}, f(\vec{X}))$. The CTMDP $\mathcal{M} = (S, \mathcal{A}, R)$ associated with a PCTMDP $(\vec{X}, \mathcal{T}, \mathcal{A}, \vec{x}_0)$ is defined by specifying the state space $S = \mathbb{N}^n \cap K$ and the rate function R as

$$R(\vec{s}, a, \vec{s'}) = \sum \{ f_\tau(\vec{s}) \mid \tau = (a, \vec{v}_\tau, f_\tau(\vec{s})) \wedge \vec{s'} = \vec{s} + \vec{v} \}.$$

It is easy to observe, modulo the introduction of enough variables and actions, that the expressive power of PCTMDPs is the same as that of CTMDPs introduced earlier: just introduce one variable per state and use a boolean one-hot encoding of states of a CTMDP into a PCTMDP. However, population processes become interesting when there are populations (like in queues or epidemic models), as in these cases the dimensionality of the population process (number of variables) is much smaller than the size of the state space.

4. Learning optimal policies via stochastic functional gradient ascent

In this section we give a variational formulation of the control problem of determining the optimal scheduler for time-bounded reachability in a CTMDP. We show how to approximate statistically in an unbiased way the functional gradient of the time-bounded reachability probability, and give a convergent algorithm to achieve this.

4.1. Reachability probability as a functional

As defined in Section 3, a scheduler is a way of resolving non-determinism by associating a (time-dependent) probability to each action/ state pair. We will realise a scheduler as a vector $\mathbf{f}$ of functions $f_a : K \times [0,T] \to \mathbb{R}$, one for each action $a \in \mathcal{A}$, where K is the compact subset of $\mathbb{R}^n$ used to define S for the PCTMDP formalism. The corresponding probability of an enabled action $a \in \mathcal{A}(\vec{X})$ at state $\vec{X}$ can be retrieved using the soft-max (logistic) transform as follows:

$$\sigma(t, \vec{X}, a) = \frac{\exp(f_a(\vec{X}, t))}{\sum_{a' \in \mathcal{A}(\vec{X})} \exp(f_{a'}(\vec{X}, t))}, \qquad \vec{X} \in S, t \in [0,T] \tag{1}$$

Notice that if some actions are not enabled in some states these are automatically excluded from the soft-max formula above, due to the restriction to the set $\mathcal{A}(\vec{X})$ of actions enabled in $\vec{X}$. Given a scheduler σ, a CTMDP is reduced to a CTMC $\mathcal{M}_\sigma$, and the problem of estimating the probability of a reachability property $\phi = \diamond_I G$ can be reduced to the computation of a transient probability for $\mathcal{M}_\sigma$ by standard techniques [8]. The satisfaction probability can be therefore viewed as a *functional*

$$Q \colon \mathcal{F} \to \mathbb{R}$$

where $\mathcal{F}$ is the set of all possible scheduler functions.

The functional is defined explicitly as follows: consider a sample run w of the form $s_0 a_0 t_0 s_1 a_1 t_1 \ldots$ from the CTMC $\mathcal{M}_\sigma$ (obtained from the CTMDP by selecting a scheduler σ). Recall that $\diamond_I G$, $I = [b_1, b_2]$, is the set of all runs that visit G within the time bound specified by the interval I. We can encode it in the following indicator function:

$$\mathcal{I}_{\diamond_I G}(w) = \begin{cases} 1, & w \in \diamond_I G \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

Then the expected reachability value associated with the scheduler σ, represented by the vector of functions $\mathbf{f} = \{f_a\}_{a \in \mathcal{A}}$, is defined as follows:

$$Q\left[\mathbf{f}(\vec{X}, t)\right] = E_{\mathcal{M}_\sigma}\left[\mathcal{I}_{\diamond_I G}\right], \tag{3}$$

where expectation is taken with respect to the distribution on runs of $\mathcal{M}_\sigma$. Notice that in general it is computationally very hard to analytically compute the r.h.s. in the above equation, as it amounts to transient analysis for a time-inhomogeneous CTMC; we therefore need to resort to statistical model checking methods [43, 44] to obtain a Monte Carlo estimate of the expectation in (3).

To formulate the continuous time control problem of determining the optimal scheduler, we need to define the concept of functional derivative.

Definition 3. Let $Q \colon \mathcal{F} \to \mathbb{R}$ be a functional defined on a space of functions $\mathcal{F}$. The *functional derivative* of Q at $f \in \mathcal{F}$ along a function $g \in \mathcal{F}$, denoted by

$\frac{\delta Q}{\delta f}$, is defined by

$$\int \frac{\delta Q}{\delta f}(\vec{X},t)\, g(\vec{X},t)\, d\vec{X}dt = \lim_{\epsilon\to 0} \frac{Q[f+\epsilon g]-Q[f]}{\epsilon} \tag{4}$$

whenever the limit on the r.h.s. exists.

Notice that if we restrict ourselves to piecewise constant functions on a grid, the definition above returns the standard definition of gradient of a finite-dimensional function. We can now give a variational definition of optimal scheduler.

Lemma 2. *An optimal scheduler σ is associated with a function f such that*

$$max_{g\in\mathcal{F}} \left\| \int \frac{\delta Q}{\delta f}(\vec{X},t)\, g(\vec{X},t)\, d\vec{X}dt \right\|_2 = 0 \tag{5}$$

where $\|\cdot\|_2$ denotes the L^2 norm on functions.

The variational formulation above allows us to attack the problem via direct optimisation through a gradient ascent algorithm, as we will see below.

4.2. Stochastic Estimation of the Functional Gradient

It is well-known that a gradient ascent approach is guaranteed to find the global optimum of a convex objective function. Gradient ascent starts from an initial solution which is updated iteratively towards the direction that induces the steepest change in the objective function; that direction is given by the gradient of the function. For a functional $Q[f]$ the concept of gradient is captured by the functional derivative $\frac{\delta Q}{\delta f}$, which is a function of $\vec{X}, t$ that dictates the rate of change of the functional Q when f is perturbed at the point $(\vec{X}, t)$. In the case of functional optimisation, the gradient ascent update will have the form:

$$f' = f + \gamma \frac{\delta Q}{\delta f} \tag{6}$$

where γ is the learning rate which controls the effect of each update, and $\frac{\delta Q}{\delta f}$ is the functional derivative of Q. Unfortunately, an analytic expression for the functional derivative of the functional defined in (3) is usually not available.

We can however obtain an unbiased estimate of the functional derivative. We demonstrate that by first considering the simple case of a finite dimensional vector space; for its gradient we have the following lemma:

Lemma 3. *Let $q\colon \mathbb{R}^n \to \mathbb{R}$ be a smooth function, and let $\nabla q(\mathbf{v})$ be its gradient at a point $\mathbf{v}$. Let $\mathbf{w}$ be a random vector from an isotropic, zero mean distribution $p(\mathbf{w})$. For $\epsilon \ll 1$, define*

$$\hat{\mathbf{w}} = \begin{cases} \mathbf{w}, & \text{if } q(\mathbf{v}+\epsilon\mathbf{w}) - q(\mathbf{v}) > 0 \\ -\mathbf{w}, & \text{otherwise.} \end{cases} \tag{7}$$

Then

$$E_p\left[\epsilon\hat{\mathbf{w}}\right] \propto \nabla q(\mathbf{v}) + O(\epsilon^2).$$

Proof. The tangent space of $\mathbb{R}^n$ at the point $\mathbf{v}$ is naturally decomposed in the orthogonal direct sum of a subspace of dimension 1 parallel to the gradient, and a subspace of dimension $n-1$ tangent to the level surfaces of the function q. For small ϵ, any change in the value of the function q will be due to movement in the gradient direction. As the distribution p is isotropic, every direction is equally likely in $\mathbf{w}$; however, the flipping operation in the definition of $\hat{\mathbf{w}}$ in (7) ensures that the component of $\hat{\mathbf{w}}$ along the gradient $\nabla q(\mathbf{v})$ is always positive, while it does not affect the orthogonal components. Therefore, in expectation, $\hat{\mathbf{w}}$ returns the direction of the functional gradient.

In the section that follows we consider the infinite-dimensional generalisation of Lemma 3.

4.3. Scheduler representation in terms of basis functions

In order to obtain an unbiased estimate of a functional gradient, we need to define a zero-mean isotropic distribution on a suitable space of functions. To do so, we introduce the concept of Gaussian Process, a generalisation of the multivariate Gaussian distribution to infinite-dimensional spaces of functions (see, e.g. [45]).

Definition 4. A Gaussian Process (GP) over an input space $\mathcal{X}$ is an infinite-dimensional family of real-valued random variables indexed by $x \in \mathcal{X}$ such that, for every finite subset $X \subset \mathcal{X}$, the finite dimensional marginal obtained by restricting the GP to X follows a multi-variate normal distribution.

Thus, a GP can be thought as a distribution over functions $f\colon \mathcal{X} \to \mathbb{R}$ such that, whenever the function is evaluated at a finite number of points, the resulting random vector is normally distributed. In the following, we will only consider $\mathcal{X} = \mathbb{R}^d$ for some integer d.

Just as the Gaussian distribution is characterised by two parameters, a GP is characterised by two functions, the *mean* and *covariance* function. The mean function plays a relatively minor role, as one can always add a deterministic mean function, without loss of generality; in our case, since we are interested in obtaining small perturbations, we will set it to zero. The covariance function, which captures the correlations between function values at different inputs, instead plays a vital role, as it defines the type of functions which can be sampled from a GP. We will use the *Radial Basis Function* (RBF) covariance, defined as follows:

$$\mathrm{cov}(f(x_1), f(x_2)) = k(x_1, x_2) = \alpha^2 \exp\left[-\frac{\|x_1 - x_2\|^2}{\lambda^2}\right]. \tag{8}$$

where α and λ are the amplitude and length-scale parameters of the covariance function. RBF covariance function enjoys the universality property [46], meaning that the set of functions which are linear combinations of RBF kernels is dense on the set of continuous functions over any compact subspace of $\mathbb{R}^n$. This means that the samples of the GP with RBF covariance function can approximate any smooth function arbitrarily well. To gain insight into the geometry of the space of functions associated with a GP with RBF covariance, we report without proof the following lemma (see e.g. Rasmussen & Williams, Ch 4.2.1 [45]).

Lemma 4. *Let $\mathcal{F}_N$ be the space of random functions $f = \sum_{j=1}^{N} w_j \phi_j(x)$ generated by taking linear combinations of basis functions $\phi_j(x) = \exp\left[-\frac{\|x-\mu_j\|^2}{\lambda^2}\right]$, with $\mu_j \in \mathbb{R}$ and independent Gaussian coefficients $w_j \sim \mathcal{N}(0, \alpha^2/N)$. The sample space of a GP with RBF covariance defined by* (8) *is the infinite union of the the spaces $\mathcal{F}_N$.*

We refer to the basis functions entering in the constructive definition of GPs given in Lemma 4 as *kernel functions*. Two immediate consequences of the previous Lemma are important for us:

- A GP with RBF covariance defines an *isotropic* distribution in its sample space (this follows immediately from the i.i.d. definition of the weights in Lemma 4);
- The sample space of a GP with RBF covariance is a dense subset of the space of all continuous functions (see also [29] and references therein).

GPs therefore provide us with a convenient way of extending the procedure described in Lemma 3 to the infinite-dimensional setting. In particular, Lemma 4 implies that any scheduler function $f \in \mathcal{F}$ that is a sample from a GP (with RBF covariance) can be approximated to arbitrary accuracy in terms of basis functions as follows:

$$f(\vec{X}, t) = \sum_{j=1}^{N} w_j \exp\left[-0.5([\vec{X}, t]^\top - \mu_j)^\top \Lambda^{-1} ([\vec{X}, t]^\top - \mu_j)\right] \tag{9}$$

where $\mu_j \in \mathbb{R}^n \times [0, T]$ is the centre of a Gaussian kernel function, Λ is a diagonal matrix that contains $n+1$ squared length-scale parameters of the kernel functions, and n is the dimensionality of the state-space. This formulation allows describing functions (aka points in an infinitely-dimensional Hilbert space) as points in the finite vector space spanned by the weights $\mathbf{w}$. Note that the proposed basis function representation implies relaxation of the population variables to the continuous domain, though in practice we are only interested in evaluating $f(\vec{X}, t)$ for integer-valued $\vec{X}$. We can now extend Lemma 3 to the infinite-dimensional case:

Lemma 5. *Let $Q \colon \mathcal{F} \to \mathbb{R}$ be a smooth functional, and let $\nabla Q[f]$ be its gradient at f. Let g be a random function from a zero mean Gaussian process $p(g)$ with RBF covariance function. For $\epsilon \ll 1$, define*

$$\hat{g} = \begin{cases} g, & \text{if } Q[f + \epsilon g] - Q[f] > 0 \\ -g, & \text{otherwise.} \end{cases} \tag{10}$$

Then

$$E_p\left[\epsilon \hat{g}(\vec{X}, t)\right] \propto \nabla Q[f](\vec{X}, t) + O(\epsilon^2). \tag{11}$$

Proof. We can simply mirror the proof of Lemma 3, since the space of functions $\mathcal{F}$ is an infinitely-dimensional Hilbert space. That means that $\mathcal{F}$ is equipped with an inner product operation, implying that the concept of projection along a direction is well defined. As in the finite-dimensional case, every direction of $\mathcal{F}$ is equally probable when sampling a function g, and the flipping operation in (10) ensures that the component of $\hat{g}$ parallel to the gradient $\nabla Q[f]$ is always positive (i.e. the projection along the gradient $\nabla Q[f]$ is always positive). Moreover, the expectation of $\hat{g}$ is well-defined, as it is a sample from a Gaussian process. Finally, the use of RBF covariance guarantees that $\hat{g}$ is always integrable, since $\hat{g}$ will be (with probability 1) a finite sum of Gaussian basis functions, which permits to prove equation (11). Note that the equation holds pointwise, at each point $(\vec{X}, t)$.

The advantage of the kernel representation is that we do not need to account for all states $\vec{X} \in S$, but only for N Gaussian kernels with centres μ_j for $1 \leq j \leq N$. Therefore, the value of the scheduler at a particular state $\vec{X}$ will be determined as a linear combination of the kernel functions, with proximal kernels contributing more due to the exponential decay of the kernel functions. This method offers a compact representation of the scheduler, and essentially does not suffer from state-space explosion, as we treat states as continuous. Moreover, we do not lose accuracy, as every function on S can be extended to a continuous function on E by interpolation. On the practical side, we consider that the kernel functions are spread evenly across the joint space (state space & time), and the length-scale for each dimension is considered to be equal to the distance of two successive kernels.[1]

4.4. A Stochastic Gradient Ascent Algorithm

Given a scheduler σ, we first evaluate the reachability probability via statistical model checking. We then perturb the corresponding functions f_α by adding a draw from a zero-mean GP with marginal variance scaled by $\epsilon \ll 1$, and evaluate again by statistical model checking the probability of the perturbed scheduler. If this is increased, we take a step in the perturbed direction, otherwise we take a step in the opposite direction. Notice that this procedure can be repeated for multiple independent perturbation functions to obtain a more robust estimate. The whole procedure is described in Algorithm 1, which produces an estimate for the gradient of the functional Q at a vector $\mathbf{f}$ of functions f_α by considering the average of k random directions. We are now ready to state our main result:

[1]Kernel functions typically also have an amplitude parameter, which we consider to be equal to 1.

```
Algorithm 1 Estimate the functional gradient of $Q[\mathbf{f}]$
Require: Vector $\mathbf{f}$ of functions $f_\alpha$, scaling factor $\epsilon$, batch size $k$
Ensure: An estimate of the functional derivative (gradient) $\nabla Q \equiv \frac{\delta Q}{\delta \mathbf{f}}$
  Set gradient $\nabla Q = 0$
  Evaluate $Q[\mathbf{f}]$ via statistical model checking
  for $i = 1$ to $k$ do
    Consider random direction $\mathbf{g}$ such that $\forall \alpha \in \mathcal{A}$, we have:
        $$g_a \sim \mathcal{N}(0, 1)$$
    Evaluate $Q[\mathbf{f} + \epsilon \mathbf{g}]$
    Estimate the directional derivative:
        $$\nabla_{\mathbf{g}} Q = \frac{Q[\mathbf{f} + \epsilon \mathbf{g}] - Q[\mathbf{f}]}{\epsilon}$$
    if $\nabla_{\mathbf{g}} Q > 0$ then
      $\nabla Q \leftarrow \nabla Q + \frac{1}{k}\mathbf{g}$
    else
      $\nabla Q \leftarrow \nabla Q - \frac{1}{k}\mathbf{g}$
    end if
  end for
```

Theorem 6. *Algorithm 1 gives an unbiased estimate of the functional gradient of the functional $Q[f_\alpha]$.*

Proof. Since both the statistical model checking estimation and the gradient estimation are unbiased and independent of each other, this follows.

```
Algorithm 2 Stochastic gradient ascent for $Q[\mathbf{f}]$
Require: Initial function vector $\mathbf{f}_0$, learning rate $\gamma_0$, $n_{\max}$ iterations
Ensure: A function vector $\mathbf{f}$ that approximates a local optimum of $Q$
  for $n \leftarrow 1$ to $n_{\max}$ do
    Estimate the functional gradient $\nabla Q$ by using Algorithm 1
    Update: $\mathbf{f}_n \leftarrow \mathbf{f}_{n-1} + \gamma_{n-1} \nabla Q$
  end for
```

Therefore, we can use this stochastic estimate of the functional gradient to devise a stochastic gradient ascent algorithm which directly solves the variational problem in equation (5). This is summarised in Algorithm 2, which requires as input an initial vector of functions $\mathbf{f}_0$, and a learning rate γ_0. The effects of the learning rate on the convergence properties of the method have been extensively studied in the literature. In particular, for a decreasing learning rate convergence is guaranteed in the strictly convex scenario, if the following conditions are satisfied: $\sum_n \gamma_n = \infty$ and $\sum_n \gamma_n^2 < \infty$ [47, 48], suggesting a $\Theta(n^{-1})$ decrease for the learning rate. In non-convex problems, such as the ones considered in this work, the $\Theta(n^{-1})$ decrease is generally too aggressive, leading to vulnerability to local optima. Following the recommendations of [49], we adopt a more conservative strategy:

$$\gamma_n = \gamma_0 \, n^{-1/2} \tag{12}$$

where γ_0 is an initial value for the learning rate, which is problem dependent.

4.4.1. Momentum-based Gradient Ascent

A usual practice to accelerate the convergence of gradient ascent is to include a momentum term in the update of the algorithm [50, 51]. Regarding Algorithm 2, the introduction of momentum requires that the update step is replaced by the following assignments:

$$\Delta_n \leftarrow \eta\Delta_{n-1} + \gamma_{n-1}\nabla Q \tag{13}$$

$$\mathbf{f}_n \leftarrow \mathbf{f}_{n-1} + \Delta_n \tag{14}$$

where the momentum parameter η determines how much the current difference Δ_n will be affected by the previous update. The fact that the change in direction at each step is also affected by the history makes the search more robust to sudden changes of the gradient. This behaviour is reported to speed up convergence in cases when there is a long and narrow ridge in the surface of the objective function [51]. Normally, the standard ascent approach tends to oscillate in the direction of the short axis of the hill, while the actual maximum is approached very slowly. The inclusion effectively averages out such oscillations, resulting a faster convergence [50, 51].

In the literature the momentum parameter η is often chosen to be close to 1, typically 0.9 [51]. Of course, the original gradient ascent algorithm is retrieved for $\eta = 0$. In the examples of section 5, we experiment with a range of possible values for η.

5. Examples

5.1. Epidemiology model with no permanent recovery

As a first case study, we demonstrate the stochastic gradient ascent algorithm on a simple epidemiological model of disease spread that features no permanent recovery, also known as the SIS model (susceptible-infected-susceptible). We start by describing the results of the algorithm proposed in [14], and then demonstrate the practical effects of the algorithmic improvements proposed in this paper. The system is modelled as a PCTMDP, in which the state is described by two variables denoting the population of susceptible (X_S) and infected individuals (X_I). We assume that no immunity to the infection is gained upon recovery. The objective is to monitor how infection progresses over time, given that there is a non-deterministic choice at each step among actions in $\mathcal{A} = \{no\ treatment, treatment\}$, indicating whether an external action is taken to deal with the infection.

This non-deterministic choice will affect the dynamics of the system, which are represented by a list of transitions together with their rate functions, in the biochemical notation style (see e.g. [52]):

infection (*): $S + I \xrightarrow{k_i} I + I$, with rate function $k_i\, X_S\, X_I$;

slow recovery (*no treatment*): $I \xrightarrow{k_r} S$, with rate function $k_r\, X_I$;

self-infection (*no treatment*): $S \xrightarrow{k_i} I$, with rate function $k_i\, X_S/2$;

fast recovery (*treatment*): $I \xrightarrow{k_r} S$, with rate function $\alpha\, k_r\, X_I$;

death (*treatment*): $I \xrightarrow{k_r} \emptyset$, with rate function $k_d\, X_I$;

death (*treatment*): $S \xrightarrow{k_r} \emptyset$, with rate function $k_d\, X_S$;

Among the transitions above, only *infection* has the same rate regardless of any non-deterministic choice. If the *no treatment* action is chosen, infected individuals recover slowly as prescribed by the *slow recovery* transition, while there is a small chance of self-infection. If treatment is applied, the recovery rate is increased by a factor $\alpha > 1$, and the chance of spontaneous infection is eliminated. We assume however that the treatment is associated with some very negative side-effects that result in a small probability of death, either for healthy or infected individuals.

In this example, we seek to construct a scheduler that maximises the probability of having no deaths and no infected individuals during the time interval $[b_1, b_2]$, i.e., maximising the safety property

$$\Box_{[b_1,b_2]} G \qquad\qquad G = \{S = N\} \tag{15}$$

The application of treatment contributes in accelerating the extinction of the infected population, but it also introduces a possibility of death. Therefore a policy of constantly applying treatment cannot be optimal with respect to the satisfiability of the property considered. Moreover, maximising the satisfaction probability requires a time-dependent scheduler, as the treatment application has to be appropriately timed so that it has effect in the time-interval $[b_1, b_2]$.

In the experiments that follow, we illustrate how the stochastic gradient ascent algorithm converges to solutions that maximise this probability. We consider a system with total population $N = 100$, and initial populations $X_{S_0} = 90$ and $X_{I_0} = 10$. The rate constants are $k_i = 0.0012$ for infection, $k_r = 0.1$ for recovery, $k_d = 0.0002$ for the death event, while the increase in the recovery rate due to treatment is fixed to $\alpha = 10$. The time bounds for the safety property considered are $b_1 = 50$ and $b_2 = 60$. Regarding the stochastic gradient ascent parameters, the learning rate at the n-th step is $\gamma_n = \gamma_0/\sqrt{n}$, where

$\gamma_0 = 5$. For the numerical estimation of the directional derivatives, we consider $\epsilon = 0.1$ and the batch size for the gradient estimation was fixed to $k = 5$. For each estimation of the Q function, we have used 1000 simulation runs. In all cases, the algorithm was run for 100 iterations, meaning that a total of 600000 simulation runs were used for each experiment. This translated into roughly 3 hours of computing time on a single thread on an regular PC running Linux. The computing time is drastically decreased by using more threads, since the generation of stochastic simulation trajectories can be trivially parallelised.

We first present an example that illustrates the importance of time in the satisfaction of the time-bounded property in (15). Figure 1 reports a scheduler which is given as a solution by the stochastic gradient ascent approach. The scheduler is presented as a multivariate function that takes values in $[0, 1]$, indicating the probability of selecting the *no treatment* action for different values of state and time. In particular, we have a series of surface plots, each of which summarises the probability of *no treatment* as function of the 2-dimensional state-space for a different time-point. The white colour denotes that *no treatment* is selected with probability 1, while the black colour implies that *treatment* is used instead. We can see that *treatment* is only preferable for a particular time window and for certain parts of the state-space, that is $X_S > 80$ and $X_I < 20$. This makes sense, as the probability of achieving full recovery from a state with more than 20 infected is too small to justify the risks connected with treatment. More specifically, *treatment* is selected with high probability for $t \in [33.75, 52.5]$, which precedes with a very small overlap the time interval if interest, which is $[50, 60]$. Intuitively, to maximise the probability that all of the population is recovered over the course of a particular interval, the *treatment* action should be engaged just before. In a different case, there is an increased risk of death, as a consequence of the negative effects of prolonged treatment.

We next investigate how the algorithm responds to different initial schedulers. In Figure 2, we monitor how the value of the functional Q as function of the scheduler evolves during the course of the algorithm, starting from different initial solutions. More specifically, Figure 2(a) depicts the evolution of Q values starting from a scheduler where *no treatment* is globally selected as an action. The initial satisfaction probability is very small, but after a number of iterations it converges to values above 0.6. Figure 2(b) summarises the results where the initial solution selects *treatment* everywhere; apparently this initial solution has been closer to the local optimum and the convergence rate had been significantly faster in this case. Convergence is even faster in Figure 2(c), where a uniform initial solution was used, i.e., each of the two possible actions had equal initial probability $\forall s \in S$ and $\forall t \in T$. Finally, in Figure 2(d) we report the Q values for a run starting from a randomly initialised scheduler. In the last two instances, the starting point has had Q values at around 0.4, which is closer to the maximum; therefore the algorithm naturally required fewer iterations to converge to a good solution. Although the convergence rate is apparently dependent on the initial solution, the experiments considered resulted in solutions of similar value, which obtain satisfaction probabilities at around 0.65. It is important to note however that there is no guarantee that the algorithm will converge to

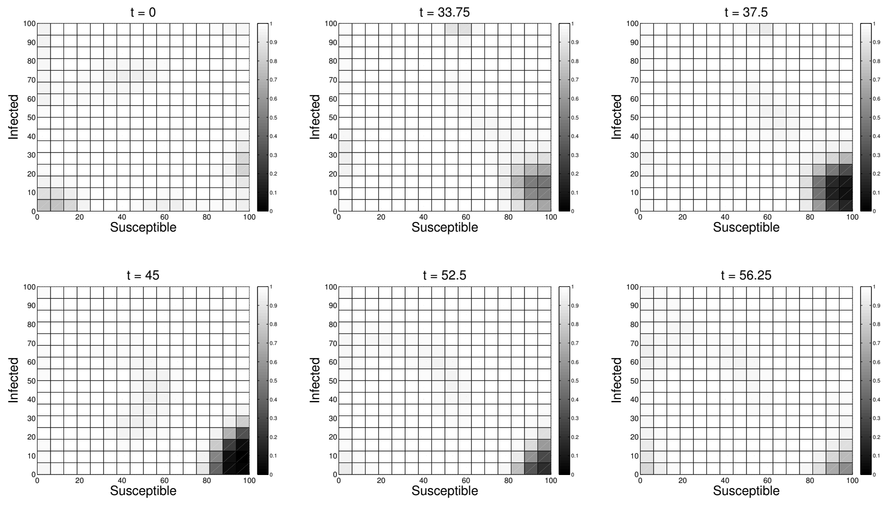


Figure 1: Example of scheduler that (locally) maximises the probability of $\mathbf{G}_{[b_1,b_2]}S = N$. The white area indicates high probability of choosing the *no treatment* action; the dark area indicates high probability of choosing *treatment*.

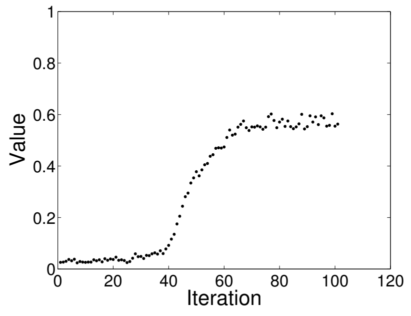
the global maximum, since the problem considered in not convex in the general case.

We finally investigate the effect that the use of momentum has on gradient ascend. Figure 3 demonstrates the evolution of Q values for three different momentum parameters: $\eta = 0.2$, $\eta = 0.5$ and $\eta = 0.9$. In all three cases the initial solution has been a scheduler that globally select *no treatment*, similar to Figure 2(a). We have chosen this initial solution since that has been the slowest converging among the examples of Figure 2. We see that increasing the momentum parameter significantly accelerates convergence; this observation is in agreement with the empirical results reported in the relevant literature [50, 51].

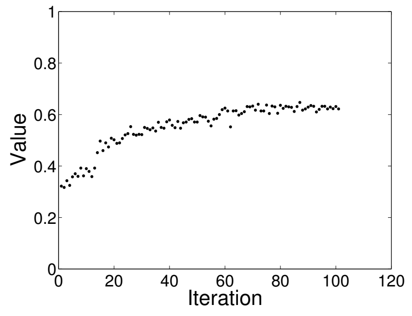
5.2. Queuing system with non-deterministic choice

In this section, we illustrate the effect of our approach on the trajectories of a queuing system. More specifically, we consider a queuing model consisting of a server and two stations, which has been adapted from Figure 2 of [15]. The two stations have incoming job requests, which are stored in their local queues. The server fetches and processes jobs from any station that has non-empty queue; if both station queues are non-empty, then the sever makes a non-deterministic choice among actions in $\mathcal{A} = \{\textit{prefer 1}, \textit{prefer 2}\}$, which indicate the preferred station. The model us summarised in the diagram of Figure 4.

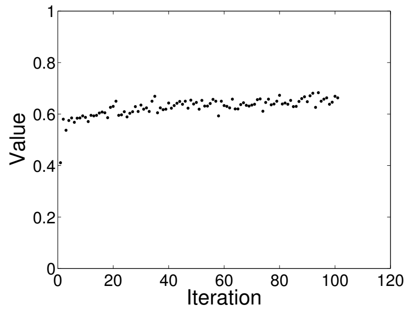
In our adaptation, the system state is described by three variables denoting the queue size in the server (X_{server}) and in the two stations (X_{S_1} and X_{S_2}). The changes of state are represented by a list of transitions in the biochemical



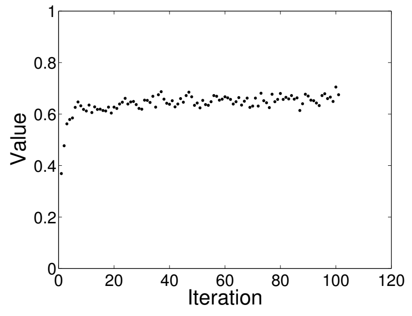
(a) *no treatment* only initial scheduler



(b) *treatment* only initial scheduler



(c) Uniform initial scheduler



(d) Random initial scheduler

Figure 2: Stochastic gradient ascent starting from different initial schedulers

notation style. We first present the following transitions that are not affected by the non-deterministic choice:

processing (*): $\mathit{Server} \xrightarrow{\mu} \emptyset, \quad \text{with rate} \quad \mu\, H[X_{sever}];$

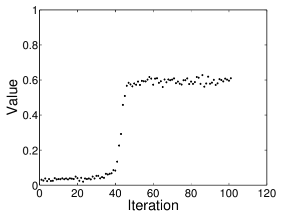
arrival 1 (*): $\emptyset \xrightarrow{\lambda_1} S_1, \quad \text{with rate} \quad \lambda_1;$

arrival 2 (*): $\emptyset \xrightarrow{\lambda_2} S_2, \quad \text{with rate} \quad \lambda_2;$

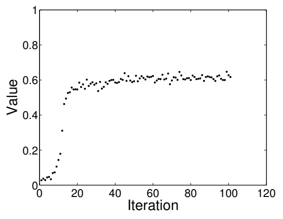
where H is the Heavyside function: $H[x] = 0$ if $x = 0$ and $H[x] = 1$ if $x > 0$. The first transition describes the processing of a job that is already in the server queue. The transitions *arrival 1* and *arrival 2* describe the arrival of new jobs that populate the station queues. We consider constant rates for the job processing by the server $\mu = 5$, and the incoming requests $\lambda_1 = 2$ and $\lambda_2 = 2$ for stations 1 and 2 correspondingly.

Regarding the non-deterministic choice, the *prefer 1* action implies that a job will be fetched from station 1 if the latter has a non-empty queue; in a different case station 2 will be served instead:

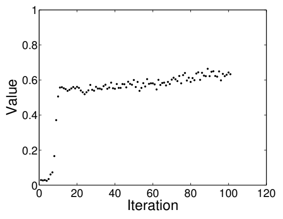
fetch 1 (*prefer 1*): $S_1 \xrightarrow{r} \mathit{Server}, \quad \text{with rate} \quad r(1-\tau)\, H[X_{S_1}];$



(a) $\eta = 0.2$



(b) $\eta = 0.5$



(c) $\eta = 0.9$

Figure 3: Evolution of Q values for stochastic gradient ascent with different momentum parameter η. The initial scheduler selects *no treatment* only, as in Figure 2(a).

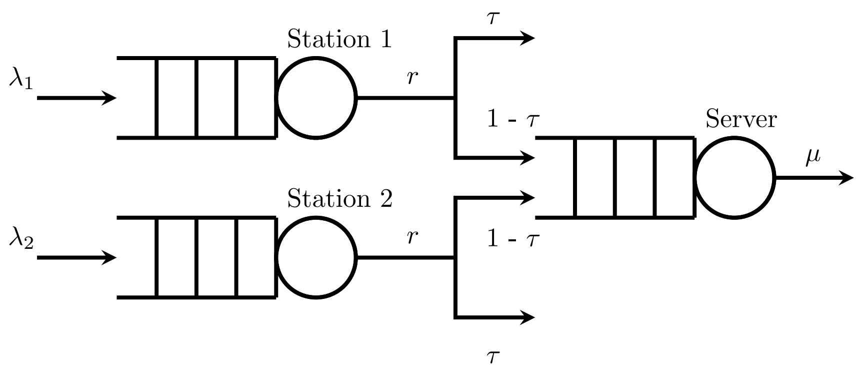


Figure 4: Queuing system with non-deterministic choice. Stations 1 and 2 accept job requests at rates λ_1 and λ_2 correspondingly. The server fetches requests from the stations by making a non-deterministic choice among actions in $\mathcal{A} = \{\textit{prefer 1}, \textit{prefer 2}\}$ indicating preference for a particular station. The requests are fetched at a total rate r and with a probability of failure τ. The jobs are eventually processed by the server at rate μ.

fetch 2 (*prefer 1*): $S_2 \xrightarrow{r} \textit{Server}$, with rate $r(1-\tau)\,H[X_{S_2}](1 - H[X_{S_1}])$;

Similarly, for the *prefer 2* action we have the following version for the fetch
650 transitions:

fetch 1 (*prefer 2*): $S_1 \xrightarrow{r} \textit{Server}$, with rate $r(1-\tau)\,H[X_{S_1}](1 - H[X_{S_2}])$;

fetch 2 (*prefer 2*): $S_2 \xrightarrow{r} \textit{Server}$, with rate $r(1-\tau)\,H[X_{S_2}]$;

For the rate of fetching a request from a station to the server we consider $r = 4$. It is also assumed that fetching a request may fail with probability $\tau = 0.1$.

655 We now demonstrate that our approach is able to synthesise such a scheduler so that the system trajectories conform with a desired reachability property. We aim to synthesise a scheduler that maximises the probability of the following time-bounded reachability property:

$$\diamond_{[b_1,b_2]} G \qquad\qquad G = \{X_{S_1} < c_1 \wedge X_{S_2} < c_2\} \tag{16}$$

The property dictates that during the time-interval $[b_1, b_2]$ the system eventually reaches a state where both queues S_1 and S_2 are below thresholds c_1 and c_2 correspondingly. For the experiments of this section, the threshold parameters of the reachability property in (16) are $c_1 = 100$ and $c_2 = 200$, while the time bounds considered are $b_1 = 100$ and $b_2 = 120$. For the stochastic gradient ascent parameters, including the leaning rate γ_n, the gradient estimation batch size k, and the scaling factor ϵ, we have used the same values as in the example of section 5.1. The initial populations for the station and the server queues are $X_{S_1} = 200$, $X_{S_2} = 0$ and $X_{server} = 0$.

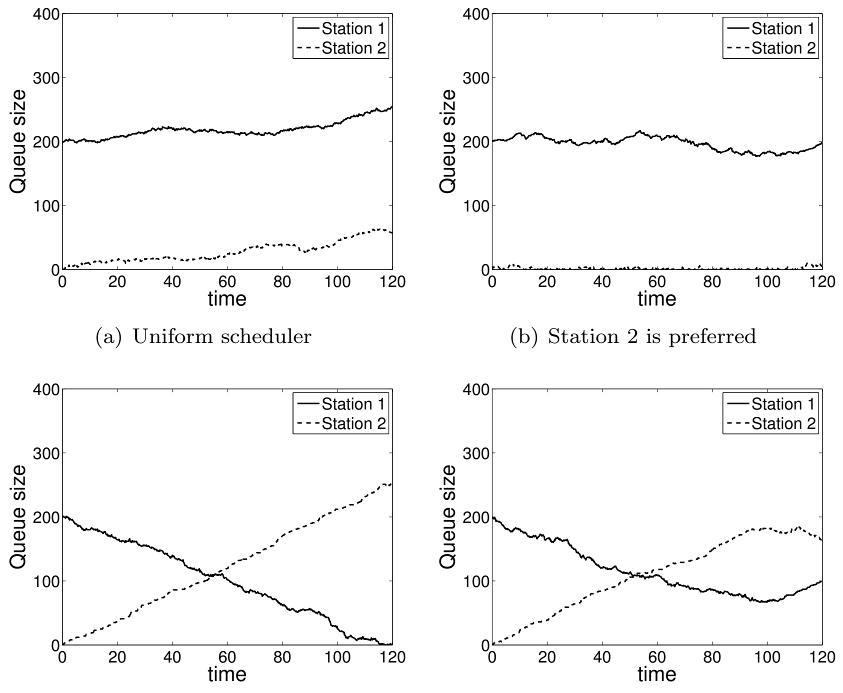
Figure 5 illustrates the effect of different schedulers on the trajectories of the queuing system. In the case of a uniform probabilistic scheduler as in Figure 5(a), both queues are slowly increasing, indicating that the server is not able to service the existing workload at a sufficient rate so that both queues are decreased. If station 2 is preferred, as in Figure 5(b), the total number of jobs waiting in the stations might be lower, but the desired property in (16) is very unlikely to be satisfied. Figure 5(c) shows a trajectory where station 1 is preferred; in this case it is very likely that the station 2 queue exceeds the desired threshold before the specified time window. Finally, in Figure 5(d) we see a sample trajectory for an optimised scheduler, for which there is an apparent change of policy at around $t = 100$. The *prefer 1* action is chosen for the most part so that the station 1 queue drops below c_1; however, the policy change maximises the probability that the station 2 queue also remains under the corresponding threshold c_2. The satisfaction probability of the property in (16) approaches 1 for the optimised scheduler.

6. Conclusions

Continuous time Markov Decision processes are a powerful mathematical tool to address control and dependability problems in many real-time applications. However, the development of practical approaches to solve policy-making problems on these models is still in its infancy and suffers badly from high computational complexity. This is particularly problematic in the case of time-dependent schedulers, which are notoriously difficult to characterise effectively.

Recent analysis techniques developed in [13, 22] for time-dependent schedulers rely on approximation techniques discretising the time bound in small intervals. Although these methods are able to compute (up to numerical precision) the reachability probability, they do not scale well to large systems due to the state-space explosion problem. In addition they require to know a-priori the mathematical description of the system of interest, and are therefore not applicable to control black-box systems where a reliable model is not available.

On the contrary our method is suitable in the case the model of the system to control is not available a-priori. Our approach is based on GPs, a probability distribution over the space of functions which universally approximates continuous functions. Since we effectively work with functions over continuous spaces, this allows us to largely circumvent the problem of state space explosion.



(a) Uniform scheduler

(b) Station 2 is preferred

(c) Station 1 is preferred

(d) Optimised scheduler

Figure 5: Example trajectories of the queuing system with different schedulers.

Our approach is vulnerable to locally optimal choices, a common problem in optimisation, where global convergence in the non-convex case is well known to be hard. From a theoretic perspective this implies that our approach can only compute a lower-bound on the reachability probability; nevertheless this can still provide a very useful result in many practical scenarios. Furthermore, we observed empirically that the algorithm had excellent performance in challenging test sets; its computational efficiency also means that practical strategies to avoid local optima, such as multiple restarts, can be feasibly employed.

Acknowledgements. L.B. acknowledges partial support from the EU-FET project QUANTICOL (nr. 600708) and by FRA-UniTS. G.S. and D.M. acknowledge support from the European Reasearch Council under grant MLCS306999. T.B. is supported by the Czech Science Foundation, grant No. 15-17564S. E.B. acknowledges the partial support of the Austrian National Research Network S 11405-N23 (RiSE/SHiNE) of the Austrian Science Fund (FWF), the ICT COST Action IC1402 Runtime Verification beyond Monitoring (ARVI) and the IKT der Zukunft of Austrian FFG project HARMONIA (nr. 845631).

References

[1] X. Guo, O. Hernandez-Lerma, Continuous-Time Markov Decision Processes, Vol. 62 of Stochastic Modelling and Applied Probability, Springer, 2009.

[2] C. Lefévre, Optimal control of a birth and death epidemic process, Oper. Res. 29 (5) (1981) 971–982. doi:10.1287/opre.29.5.971.

[3] X. Guo, O. Hernández-Lerma, T. Prieto-Rumeau, X.-R. Cao, J. Zhang, Q. Hu, M. E. Lewis, R. Vélez, A survey of recent results on continuous-time Markov decision processes, TOP 14 (2) (2006) 177–261. doi:10.1007/BF02837562.

[4] A. Lukina, L. Esterle, C. Hirsch, E. Bartocci, J. Yang, A. Tiwari, S. A. Smolka, R. Grosu, ARES: Adaptive receding-horizon synthesis of optimal plans, in: Proc. of TACAS 2017: the 23rd International Conference on Tools and Algorithms for the Construction and Analysis of Systems, Vol. 10206 of LNCS, Springer Berlin Heidelberg, 2017, pp. 286–302. doi:10.1007/978-3-662-54580-5.

[5] Q. Qiu, Q. Wu, M. Pedram, Stochastic modeling of a power-managed system-construction and optimization, IEEE T. Comput. Aid. D. 20 (10) (2001) 1200–1217. doi:10.1145/313817.313923.

[6] L. I. Sennott, Stochastic Dynamic Programming and the Control of Queueing Systems, John Wiley & Sons, Inc., 1998.

[7] A. I. Medina Ayala, S. B. Andersson, C. Belta, Probabilistic control from time-bounded temporal logic specifications in dynamic environments, in: Proc. of ICRA 2012: the 2012 IEEE International Conference on Robotics and Automation, IEEE, 2012, pp. 4705–4710. doi:10.1109/ICRA.2012.6224963.

[8] C. Baier, B. Haverkort, H. Hermanns, J.-P. Katoen, Model-checking algorithms for continuous-time Markov chains, IEEE Trans. Software Eng. 29 (6) (2003) 524–541. doi:10.1109/TSE.2003.1205180.

[9] C. Baier, H. Hermanns, J.-P. Katoen, B. R. Haverkort, Efficient computation of time-bounded reachability probabilities in uniform continuous-time Markov decision processes, Theor. Comput. Sci. 345 (1) (2005) 2–26. doi:10.1016/j.tcs.2005.07.022.

[10] M. R. Neuhäußer, L. Zhang, Time-bounded reachability probabilities in continuous-time Markov decision processes, in: Proc. of QEST: the Seventh International Conference on the Quantitative Evaluation of Systems, IEEE, 2010, pp. 209–218. doi:10.1109/QEST.2010.47.

[11] M. N. Rabe, S. Schewe, Finite optimal control for time-bounded reachability in CTMDPs and continuous-time Markov games, Acta Inform. 48 (2011) 291–315. doi:10.1007/s00236-011-0140-0.

[12] M. N. Rabe, S. Schewe, Optimal time-abstract schedulers for CTMDPs and continuous-time Markov games, Theor. Comput. Sci. 467 (2013) 53–67. doi:10.1016/j.tcs.2012.10.001.

[13] Y. Butkova, H. Hatefi, H. Hermanns, J. Krcal, Optimal continuous time Markov decisions, in: Proc. of ATVA 2015: the 13th International Symposium on Automated Technology for Verification and Analysis, Vol. 9364 of LNCS, Springer, 2015, pp. 166–182. doi:10.1007/978-3-319-24953-7_12.

[14] E. Bartocci, L. Bortolussi, T. Brázdil, D. Milios, G. Sanguinetti, Policy learning for time-bounded reachability in continuous-time markov decision processes via doubly-stochastic gradient ascent, in: Proc. of QEST 2016: the 13th International Conference on Quantitative Evaluation of Systems, Vol. 9826 of LNCS, Springer, 2016, pp. 244–259. doi:10.1007/978-3-319-43425-4_17.

[15] D. Guck, H. Hatefi, H. Hermanns, J. Katoen, M. Timmer, Modelling, reduction and analysis of Markov automata, in: Proc. of QEST 2013: the 10th International Conference on Quantitative Evaluation of Systems, Vol. 8054, Springer, 2013, pp. 55–71. doi:10.1007/978-3-642-40196-1_5.

[16] C. Baier, M. Z. Kwiatkowska, Model checking for a probabilistic branching time logic with fairness, Distributed Computing 11 (1998) 125–155. doi:10.1007/s004460050046.

[17] A. Bianco, L. de Alfaro, Model checking of probabilistic and nondeterministic systems, in: Proc. of FSTTCS: the 15th Conference on Foundations of Software Technology and Theoretical Computer Science, Vol. 1026 of LNCS, Springer, 1995, pp. 499–513. doi:10.1007/3-540-60692-0.

[18] M. Kwiatkowska, G. Norman, D. Parker, PRISM 4.0: Verification of probabilistic real-time systems, in: Proc. of CAV 2011: the 23rd International Conference on Computer Aided Verification, Vol. 6806 of LNCS, 2011, pp. 585–591. doi:10.1007/978-3-642-22110-1.

[19] A. Martin-Löf, Optimal control of a continuous-time Markov chain with periodic transition probabilities, Oper. Res. 15 (5) (1967) 872–881. doi:10.1287/opre.15.5.872.

[20] B. Miller, Finite state continuous time Markov decision processes with an infinite planning horizon, J. Math. Anal. Appl. 22 (3) (1968) 552–569. doi:10.1016/0022-247X(68)90194-7.

[21] P. Buchholz, I. Schulz, Numerical analysis of continuous-time Markov decision processes over finite horizons, Comput. Oper. Res. 38 (3) (2011) 651–659. `doi:10.1016/j.cor.2010.08.011`.

[22] J. Fearnley, M. N. Rabe, S. Schewe, L. Zhang, Efficient approximation of optimal control for continuous-time Markov games, Information and Computation 247 (2016) 106–129. `doi:10.1016/j.ic.2015.12.002`.

[23] T. Brázdil, V. Forejt, J. Krcál, J. Kretínský, A. Kucera, Continuous-time stochastic games with time-bounded reachability, in: Proc. of FSTTCS 2009: the IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science, Vol. 4 of LIPIcs, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, 2009, pp. 61–72. `doi:10.4230/LIPIcs.FSTTCS.2009.2307`.

[24] T. Brázdil, V. Forejt, J. Krcál, J. Kretínský, A. Kucera, Continuous-time stochastic games with time-bounded reachability, Inf. Comput. 224 (2013) 46–70. `doi:10.1016/j.ic.2013.01.001`.

[25] M. R. Neuhäußer, M. Stoelinga, J. Katoen, Delayed nondeterminism in continuous-time Markov decision processes, in: Proc. of FOSSACS 2009: the 12th International Conference on Foundations of Software Science and Computational Structures, Vol. 5504 of LNCS, Springer, 2009, pp. 364–379. `doi:10.1007/978-3-642-00596-1_26`.

[26] N. Wolovick, S. Johr, A characterization of meaningful schedulers for continuous-time Markov decision processes, in: Proc. of FORMATS 2006: the 4th International Conference on Formal Modeling and Analysis of Timed Systems, 2006, pp. 352–367. `doi:10.1007/11867340_25`.

[27] T. Chen, T. Han, J. Katoen, A. Mereacre, Reachability probabilities in Markovian timed automata, in: Proc. of CDC-ECC 2011: the 50th IEEE Conference on Decision and Control and European Control Conference, IEEE, 2011, pp. 7075–7080. `doi:10.1109/CDC.2011.6160992`.

[28] D. Henriques, J. Martins, P. Zuliani, A. Platzer, E. M. Clarke, Statistical model checking for Markov decision processes, in: Proc. of QEST 2012: the Ninth International Conference on Quantitative Evaluation of Systems, IEEE Computer Society, 2012, pp. 84–93. `doi:10.1109/QEST.2012.19`.

[29] L. Bortolussi, D. Milios, G. Sanguinetti, Smoothed model checking for uncertain continuous time Markov chains, Inform. Comput. 247 (2016) 235–253. `doi:10.1016/j.ic.2016.01.004`.

[30] E. Bartocci, L. Bortolussi, L. Nenzi, G. Sanguinetti, System design of stochastic models using robustness of temporal properties, in: Theor. Comput. Sci., Vol. 587, 2015, pp. 3–25. `doi:10.1016/j.tcs.2015.02.046`.

[31] L. Bortolussi, G. Sanguinetti, Learning and designing stochastic processes from logical constraints, in: Proc. of QEST 2013: the 10th international conference on Quantitative Evaluation of Systems, Vol. 8054 of LNCS, Springer-Verlag, 2013, pp. 89–105. doi:10.1007/978-3-642-40196-1_7.

[32] E. Bartocci, L. Bortolussi, G. Sanguinetti, Data-driven statistical learning of temporal logic properties, in: Proc. of FORMATS 2014: the 12th International Conference on Formal Modeling and Analysis of Timed Systems, Vol. 8711 of Lecture Notes in Computer Science, Springer, 2014, pp. 23–37. doi:10.1007/978-3-319-10512-3_3.

[33] E. Bartocci, L. Bortolussi, D. Milios, L. Nenzi, G. Sanguinetti, Studying emergent behaviours in morphogenesis using signal spatio-temporal logic, in: Proc. of HSB 2015: the Fourth International Workshop on Hybrid Systems Biology, Vol. 9271 of LNCS, Springer, 2015, pp. 156–172. doi:10.1007/978-3-319-26916-0_9.

[34] M. Rosenstein, A. G. Barto, Robot weightlifting by direct policy search, in: Proc. of IJCAI'01: the 17th International Joint Conference on Artificial Intelligence, Vol. 17, Morgan Kaufmann, 2001, pp. 839–846.
URL http://ijcai.org/proceedings/2001-1

[35] J. Baxter, P. L. Bartlett, L. Weaver, Experiments with infinite-horizon, policy-gradient estimation, J. Artif. Int. Res. 15 (1) (2011) 351–381. doi:10.1613/jair.807.

[36] F. Stulp, O. Sigaud, Path integral policy improvement with covariance matrix adaptation, arXiv preprint arXiv:1206.4621.
URL http://arxiv.org/abs/1206.4621

[37] S. Mannor, R. Y. Rubinstein, Y. Gat, The cross entropy method for fast policy search, in: ICML, 2003, pp. 512–519.
URL http://www.aaai.org/Papers/ICML/2003/ICML03-068.pdf

[38] F. Stulp, O. Sigaud, Policy improvement methods: Between black-box optimization and episodic reinforcement learning (2012).
URL http://hal.upmc.fr/hal-00738463/

[39] T. Brázdil, K. Chatterjee, M. Chmelik, V. Forejt, J. Kretínský, M. Z. Kwiatkowska, D. Parker, M. Ujma, Verification of Markov decision processes using learning algorithms, in: Proc. of ATVA 2014: the 12th International Symposium on Automated Technology for Verification and Analysis, Vol. 8837 of LNCS, Springer, 2014, pp. 98–114. doi:10.1007/978-3-319-11936-6_8.

[40] M. R. Neuhäußer, Model checking nondeterministic and randomly timed systems, Ph.D. thesis, RWTH Aachen University (2010).
URL http://darwin.bth.rwth-aachen.de/opus3/volltexte/2010/3136/

[41] L. Bortolussi, J. Hillston, D. Latella, M. Massink, Continuous approximation of collective systems behaviour: A tutorial, Perform. Evaluation 70 (5) (2013) 317–349. `doi:10.1016/j.peva.2013.01.001`.

[42] T. Henzinger, B. Jobstmann, V. Wolf, Formalisms for specifying Markovian population models, International Journal of Foundations of Computer Science 22 (04) (2011) 823–841. `doi:10.1142/S0129054111008441`.

[43] S. K. Jha, E. M. Clarke, C. J. Langmead, A. Legay, A. Platzer, P. Zuliani, A Bayesian approach to model checking biological systems, in: Proc. of CMSB 2009: the 7th International Conference on Computational Methods in Systems Biology, Vol. 5688 of LNCS, Springer, 2009, pp. 218–234.

[44] H. L. S. Younes, R. G. Simmons, Statistical probabilistic model checking with a focus on time-bounded properties, Inform. Comput. 204 (9) (2006) 1368–1409. `doi:10.1016/j.ic.2006.05.002`.

[45] C. E. Rasmussen, C. K. I. Williams, Gaussian processes for machine learning, MIT Press, Cambridge, Mass., 2006.

[46] I. Steinwart, On the influence of the kernel on the consistency of support vector machines, J. Mach. Learn. Res. 2 (2002) 67–93. `doi:10.1162/153244302760185252`.

[47] N. Murata, On-line Learning in Neural Networks, Cambridge University Press, 1998, Ch. A Statistical Study of On-line Learning, pp. 63–92.

[48] L. Bottou, Large-scale machine learning with stochastic gradient descent, in: Proc. of COMPSTAT'2010: the 19th International Conference on Computational Statistics, Physica-Verlag HD, 2010, pp. 177–186. `doi:10.1007/978-3-7908-2604-3`.

[49] L. Bottou, Neural Networks: Tricks of the Trade: Second Edition, Vol. 7700 of LNCS, Springer Berlin Heidelberg, Berlin, Heidelberg, 2012, Ch. "Stochastic Gradient Descent Tricks", pp. 421–436.

[50] D. E. Rumelhart, G. E. Hinton, R. J. Williams, Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1, MIT Press, 1986, Ch. Learning Internal Representations by Error Propagation, pp. 318–362.

[51] N. Qian, On the momentum term in gradient descent learning algorithms, Neural Networks 12 (1) (1999) 145–151. `doi:10.1016/S0893-6080(98)00116-6`.

[52] D. T. Gillespie, Exact stochastic simulation of coupled chemical reactions, J. of Physical Chemistry 81 (25). `doi:10.1021/j100540a008`.