Maximizing the Conditional Expected Reward for Reaching the Goal

Christel Baier(✉), Joachim Klein(✉), Sascha Klüppelholz(✉), and Sascha Wunderlich(✉)

Institute for Theoretical Computer Science, Technische Universität Dresden, Dresden, Germany
{christel.baier,joachim.klein,sascha.klueppelholz, sascha.wunderlich}@tu-dresden.de

Abstract. The paper addresses the problem of computing maximal conditional expected accumulated rewards until reaching a target state (briefly called *maximal conditional expectations*) in finite-state Markov decision processes where the condition is given as a reachability constraint. Conditional expectations of this type can, e.g., stand for the maximal expected termination time of probabilistic programs with nondeterminism, under the condition that the program eventually terminates, or for the worst-case expected penalty to be paid, assuming that at least three deadlines are missed. The main results of the paper are (i) a polynomial-time algorithm to check the finiteness of maximal conditional expectations, (ii) PSPACE-completeness for the threshold problem in acyclic Markov decision processes where the task is to check whether the maximal conditional expectation exceeds a given threshold, (iii) a pseudo-polynomial-time algorithm for the threshold problem in the general (cyclic) case, and (iv) an exponential-time algorithm for computing the maximal conditional expectation and an optimal scheduler.

1 Introduction

Stochastic shortest (or longest) path problems are a prominent class of optimization problems where the task is to find a policy for traversing a probabilistic graph structure such that the expected value of the generated paths satisfying a certain objective is minimal (or maximal). In the classical setting (see e.g. [15,22,25,29]), the underlying graph structure is given by a finite-state Markov decision process (MDP), i.e., a state-transition graph with nondeterministic choices between several actions for each of its non-terminal states, probability distributions specifying the probabilities for the successor states for each state-action pair and a reward function that assigns rational values to the state-action pairs. The stochastic shortest (longest) path problem asks to find a scheduler, i.e., a function that resolves the nondeterministic choices, possibly in a

The authors are supported by the DFG through the collaborative research centre HAEC (SFB 912), the Excellence Initiative by the German Federal and State Governments (cluster of excellence cfAED), the Research Training Group QuantLA (GRK 1763), and the DFG-project BA-1679/11-1.

A. Legay and T. Margaria (Eds.): TACAS 2017, Part II, LNCS 10206, pp. 269–285, 2017.
DOI: 10.1007/978-3-662-54580-5_16

history-dependent way, which minimizes (maximizes) the expected accumulated reward until reaching a goal state. To ensure the existence of the expectation for given schedulers, one often assumes that the given MDP is contracting, i.e., the goal is reached almost surely under all schedulers, in which case the optimal expected accumulated reward is achieved by a memoryless deterministic scheduler that optimizes the expectation from each state and is computable using a linear program with one variable per state (see e.g. [25]). The contraction assumption can be relaxed by requiring the existence of at least one scheduler that reaches the goal almost surely and taking the extremum over all those schedulers [15,16,22]. These algorithms and corresponding value or policy iteration approaches have been implemented in various tools and used in many application areas.

The restriction to schedulers that reach the goal almost surely, however, limits the applicability and significance of the results. First, the known algorithms for computing extremal expected accumulated rewards are not applicable for models where the probability for never visiting a goal state is positive under each scheduler. Second, statements about the expected rewards for schedulers that reach the goal with probability 1 are not sufficient to draw any conclusion for the best- or worst-case behavior, if there exist schedulers that miss the goal with positive probability. This motivates the consideration of *conditional stochastic path problems* where the task is to compute the optimal expected accumulated reward until reaching a goal state, under the condition that a goal state will indeed be reached and where the extrema are taken over all schedulers that reach the goal with positive probability. More precisely, we address here a slightly more general problem where we are given two sets F and G of states in an MDP $\mathcal{M}$ with non-negative integer rewards and ask for the maximal expected accumulated reward until reaching F, under the condition that G will be visited (denoted $\mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond F | \Diamond G)$ where s_{init} is the initial state of $\mathcal{M}$). Computation schemes for conditional expectations of this type can, e.g., be used to answer the following questions (assuming the underlying model is a finite-state MDP):

(Q1) What is the maximal termination time of a probabilistic and nondeterministic program, under the condition that the program indeed terminates?
(Q2) What are the maximal expected costs of the repair mechanisms that are triggered in cases where a specific failure scenario occurs, under the condition that the failure scenario indeed occurs?
(Q3) What is the maximal energy consumption, under the condition that all jobs of a given list will be successfully executed within one hour?

The relevance of question (Q1) and related problems becomes clear from the work [14,20,23,24,26] on the semantics of probabilistic programs where no guarantees for almost-sure termination can be given. Question (Q2) is natural for a worst-case analysis of resilient systems or other types of systems where conditional probabilities serve to provide performance guarantees on the protocols triggered in exceptional cases that appear with positive, but low probability. Question (Q3) is typical when the task is to study the trade-off between cost and utility functions (see e.g. [9]). Given the work on anonymity and related notions for information leakage using conditional probabilities in MDP-like models [7,21] or

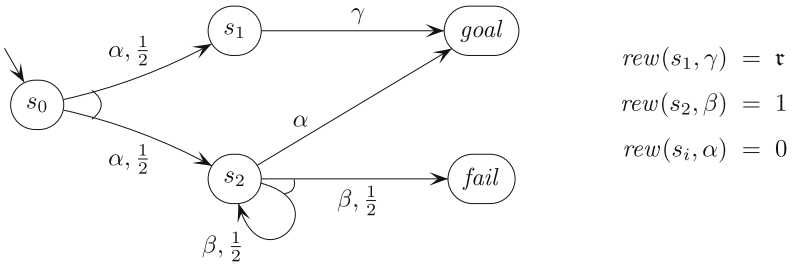

Fig. 1. MDP $\mathcal{M}[\mathfrak{r}]$ for Example 1.1

the formalization of posterior vulnerability as an expectation [4], the concept of conditional accumulated excepted rewards might also be useful to specify the degree of protection of secret data or to study the trade-off between privacy and utility, e.g., using gain functions [3,5]. Other areas where conditional expectations play a crucial role are risk management where the conditional value-at-risk is used to formalize the expected loss under the assumption that very large losses occur [2,32] or regression analysis where conditional expectations serve to predict the relation between random variables [31].

Example 1.1. To illustrate the challenges for designing algorithms to compute maximal conditional expectations we regard the MDP $\mathcal{M}[\mathfrak{r}]$ shown in Fig. 1. The reward of the state-action pair (s_1, γ) is given by a reward parameter $\mathfrak{r} \in \mathbb{N}$. Let $s_{init} = s_0$ be the initial state and $F = G = \{goal\}$. The only nondeterministic choice is in state s_2, while states s_0 and s_1 behave purely probabilistic and *goal* and *fail* are trap states. Given a scheduler $\mathfrak{S}$, we write $\mathbb{CE}^{\mathfrak{S}}$ for the conditional expectation $\mathbb{E}^{\mathfrak{S}}_{\mathcal{M}[\mathfrak{r}],s_0}(\Diamond goal | \Diamond goal)$. (See also Sect. 2 for our notations.) For the two memoryless schedulers that choose α resp. β in state s_2 we have:

$$\mathbb{CE}^{\alpha} = \frac{\frac{1}{2}\cdot\mathfrak{r} + \frac{1}{2}\cdot 0}{\frac{1}{2}+\frac{1}{2}} = \frac{\mathfrak{r}}{2} \quad \text{and} \quad \mathbb{CE}^{\beta} = \frac{\frac{1}{2}\cdot\mathfrak{r} + 0}{\frac{1}{2}+0} = \mathfrak{r}$$

We now regard the schedulers $\mathfrak{S}_n$ for $n = 1, 2, \ldots$ that choose β for the first n visits of s_2 and action α for the $(n{+}1)$-st visit of s_2. Then:

$$\mathbb{CE}^{\mathfrak{S}_n} = \frac{\frac{1}{2}\cdot\mathfrak{r} + \frac{1}{2}\cdot\frac{1}{2^n}\cdot n}{\frac{1}{2}+\frac{1}{2}\cdot\frac{1}{2^n}} = \mathfrak{r} \;+\; \frac{n-\mathfrak{r}}{2^n{+}1}$$

Thus, $\mathbb{CE}^{\mathfrak{S}_n} > \mathbb{CE}^{\beta}$ iff $n > \mathfrak{r}$, and the maximum is achieved for $n = \mathfrak{r}{+}2$.

This example illustrates three phenomena that distinguish conditional and unconditional expected accumulated rewards and make reasoning about maximal conditional expectations harder than about unconditional ones. First, optimal schedulers for $\mathcal{M}[\mathfrak{r}]$ need a counter for the number of visits in state s_2. Hence, memoryless schedulers are not powerful enough to maximize the conditional expectation. Second, while the maximal conditional expectation for $\mathcal{M}[\mathfrak{r}]$ with

initial state $s_{init} = s_0$ is finite, the maximal conditional expectation for $\mathcal{M}[\mathfrak{r}]$ with starting state s_2 is infinite as:

$$\sup_{n\in\mathbb{N}} \mathbb{E}^{\mathfrak{S}_n}_{\mathcal{M}[\mathfrak{r}],s_2}(\Diamond\!\!\!\!-\, goal \,|\, \Diamond goal) = \sup_{n\in\mathbb{N}} \frac{\frac{n}{2^n}}{\frac{1}{2^n}} = \infty$$

Third, as $\mathfrak{S}_2$ maximizes the conditional expected accumulated reward for $\mathfrak{r} = 0$, while $\mathfrak{S}_3$ is optimal for $\mathfrak{r} = 1$, optimal decisions for paths ending in state s_2 depend on the reward value r of the γ-transition from state s_1, although state s_1 is not reachable from s_2. Thus, optimal decisions for a path π do not only depend on the past (given by π) and possible future (given by the sub-MDP that is reachable from π's last state), but require global reasoning. ∎

The main results of this paper are the following theorems. We write $\mathbb{CE}^{\max}$ for the maximal conditional expectation, i.e., the supremum of the conditional expectations $\mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!-\, F|\Diamond G)$, when ranging over all schedulers $\mathfrak{S}$ where $\Pr^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\Diamond G)$ is positive and $\Pr^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\Diamond F|\Diamond G) = 1$. (See also Sect. 2 for our notations.)

Theorem 1 (Checking finiteness and upper bound). *There is a polynomial-time algorithm that checks if $\mathbb{CE}^{\max}$ is finite. If so, an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\max}$ is computable in pseudo-polynomial time for the general case and in polynomial time if $F = G$ and $\Pr^{\min}_{\mathcal{M},s}(\Diamond G) > 0$ for all states s with $s \models \exists\Diamond G$.*

The threshold problem asks whether the maximal conditional expectation exceeds or misses a given rational threshold ϑ.

Theorem 2 (Threshold problem). *The problem "does $\mathbb{CE}^{\max} \bowtie \vartheta$ hold?" (where $\bowtie \in \{>, \geqslant, <, \leqslant\}$) is PSPACE-hard and solvable in exponential (even pseudo-polynomial) time. It is PSPACE-complete for acyclic MDPs.*

For the computation of an optimal scheduler, we suggest an iterative scheduler-improvement algorithm that interleaves calls of the threshold algorithm with linear programming techniques to handle zero-reward actions. This yields:

Theorem 3 (Computing optimal schedulers). *The value $\mathbb{CE}^{\max}$ and an optimal scheduler $\mathfrak{S}$ are computable in exponential time.*

Algorithms for checking finiteness and computing an upper bound (Theorem 1) will be sketched in Sect. 3. Section 4 presents a pseudo-polynomial threshold algorithm and a polynomially space-bounded algorithm for acyclic MDPs (Theorem 2) as well as an exponential-time computation scheme for the construction of an optimal scheduler (Theorem 3). Further details, soundness proofs and a proof for the PSPACE-hardness as stated in Theorem 2 can be found in [13]. The general feasibility of the algorithms will be shown by experimental studies with a prototypical implementation (for details, see Appendix K of [13]).

Related Work. Although conditional expectations appear rather naturally in many applications and despite the large amount of publications on variants

of stochastic path problems and other forms of expectations in MDPs (see e.g. [18,30]), we are not aware that they have been addressed in the context of MDPs. Computation schemes for extremal conditional probabilities $\Pr^{\max}(\varphi|\psi)$ or $\Pr^{\min}(\varphi|\psi)$ where both the objective φ and the assumption ψ are path properties specified in some temporal logic have been studied in [6,8,11]. For reachability properties φ and ψ, the algorithm of [6,8] has exponential time complexity, while the algorithm of [11] runs in polynomial time. Although the approach of [11] is not applicable for calculating maximal conditional expectations (see Appendix B of [13]), it can be used to compute an upper bound for $\mathbb{CE}^{\max}$ (see Sect. 3). Conditional expected rewards in Markov chains can be computed using the rescaling technique of [11] for finite Markov chains or the approximation techniques of [1,19] for certain classes of infinite-state Markov chains. The conditional weakest precondition operator of [26] yields a technique to compute conditional expected rewards for purely probabilistic programs (without non-determinism).

2 Preliminaries

We briefly summarize our notations used for Markov decision processes. Further details can be found in textbooks, see e.g. [25,29] or Chapter 10 in [10].

A *Markov decision process* (MDP) is a tuple $\mathcal{M} = (S, Act, P, s_{init}, rew)$ where S is a finite set of states, Act a finite set of actions, $s_{init} \in S$ the initial state, $P : S \times Act \times S \to [0,1] \cap \mathbb{Q}$ is the transition probability function and $rew : S \times Act \to \mathbb{N}$ the reward function. We require that $\sum_{s' \in S} P(s,\alpha,s') \in \{0,1\}$ for all $(s,\alpha) \in S \times Act$. We write $Act(s)$ for the set of actions that are enabled in s, i.e., $\alpha \in Act(s)$ iff $P(s,\alpha,\cdot)$ is not the null function. State s is called a *trap* if $Act(s) = \varnothing$. The paths of $\mathcal{M}$ are finite or infinite sequences $s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2\ldots$ where states and actions alternate such that $P(s_i,\alpha_i,s_{i+1}) > 0$ for all $i \geqslant 0$. A path π is called *maximal* if it is either infinite or finite and its last state is a trap. If $\pi = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2\ldots\alpha_{k-1}\,s_k$ is finite then $rew(\pi) = rew(s_0,\alpha_0) + rew(s_1,\alpha_1) + \ldots + rew(s_{k-1},\alpha_{k-1})$ denotes the accumulated reward and $first(\pi) = s_0$, $last(\pi) = s_k$ its first resp. last state. The *size* of $\mathcal{M}$, denoted $size(\mathcal{M})$, is the sum of the number of states plus the total sum of the logarithmic lengths of the non-zero probability values $P(s,\alpha,s')$ and the reward values $rew(s,\alpha)$.[1]

An *end component* of $\mathcal{M}$ is a strongly connected sub-MDP. End components can be formalized as pairs $\mathcal{E} = (E, \mathfrak{A})$ where E is a nonempty subset of S and $\mathfrak{A}$ a function that assigns to each state $s \in E$ a nonempty subset of $Act(s)$ such that the graph induced by $\mathcal{E}$ is strongly connected.

A *(randomized) scheduler* for $\mathcal{M}$, often also called policy or adversary, is a function $\mathfrak{S}$ that assigns to each finite path π where $last(\pi)$ is not a trap a

[1] The logarithmic length of an integer n is the number of bits required for a representation of n as a binary number. The logarithmic length of a rational number a/b is defined as the sum of the logarithmic lengths of its numerator a and its denominator b, assuming that a and b are coprime integers and b is positive.

probability distribution over $Act(last(\pi))$. $\mathfrak{S}$ is called memoryless if $\mathfrak{S}(\pi) = \mathfrak{S}(\pi')$ for all finite paths π, π' with $last(\pi) = last(\pi')$, in which case $\mathfrak{S}$ can be viewed as a function that assigns to each non-trap state s a distribution over $Act(s)$. $\mathfrak{S}$ is called deterministic if $\mathfrak{S}(\pi)$ is a Dirac distribution for each path π, in which case $\mathfrak{S}$ can be viewed as a function that assigns an action to each finite path π where $last(\pi)$ is not a trap. We write $\Pr^{\mathfrak{S}}_{\mathcal{M},s}$ or briefly $\Pr^{\mathfrak{S}}_{s}$ to denote the probability measure induced by $\mathfrak{S}$ and s. Given a measurable set ψ of maximal paths, then $\Pr^{\min}_{\mathcal{M},s}(\psi) = \inf_{\mathfrak{S}} \Pr^{\mathfrak{S}}_{\mathcal{M},s}(\psi)$ and $\Pr^{\max}_{\mathcal{M},s}(\psi) = \sup_{\mathfrak{S}} \Pr^{\mathfrak{S}}_{\mathcal{M},s}(\psi)$. We will use LTL-like notations to specify measurable sets of maximal paths. For these it is well-known that optimal deterministic schedulers exists. If ψ is a reachability condition then even optimal deterministic memoryless schedulers exist.

Let $\varnothing \neq F \subseteq S$. For a comparison operator $\bowtie \in \{=, >, \geqslant, <, \leqslant\}$ and $r \in \mathbb{N}$, $\Diamond^{\bowtie r} F$ denotes the event "reaching F along some finite path π with $rew(\pi) \bowtie r$". The notation $\Diamond\!\!\!\!+ F$ will be used for the random variable that assigns to each maximal path ς in $\mathcal{M}$ the reward $rew(\pi)$ of the shortest prefix π of ς where $last(\pi) \in F$. If $\varsigma \not\models \Diamond F$ then $(\Diamond\!\!\!\!+ F)(\varsigma) = \infty$. If $s \in S$ then $\mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F)$ denotes the expectation of $\Diamond\!\!\!\!+ F$ in $\mathcal{M}$ with starting state s under $\mathfrak{S}$, which is infinite if $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond F) < 1$. $\mathbb{E}^{\max}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F) \in \mathbb{R} \cup \{\pm\infty\}$ stands for $\sup_{\mathfrak{S}} \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F)$ where the supremum is taken over all schedulers $\mathfrak{S}$ with $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond F) = 1$. Let ψ be a measurable set of maximal paths. $\mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F|\psi)$ stands for the expectation of $\Diamond\!\!\!\!+ F$ w.r.t. the conditional probability measure $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\,\cdot\,|\psi)$ given by $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\varphi|\psi) = \Pr^{\mathfrak{S}}_{\mathcal{M},s}(\varphi \wedge \psi)/\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\psi)$. $\mathbb{E}^{\max}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F|\psi)$ is the supremum of $\mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F|\psi)$ where $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\psi) > 0$ and $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond F|\psi) = 1$, and $\Pr^{\max}_{\mathcal{M},s}(\varphi|\psi) = \sup_{\mathfrak{S}} \Pr^{\mathfrak{S}}_{\mathcal{M},s}(\varphi|\psi)$ where $\mathfrak{S}$ ranges over all schedulers with $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\psi) > 0$ and $\sup \varnothing = -\infty$.

For the remainder of this paper, we suppose that two nonempty subsets F and G of S are given such that $\Pr^{\max}_{\mathcal{M},s}(\Diamond F|\Diamond G) = 1$. The task addressed in this paper is to compute the maximal conditional expectation given by:

$$\mathbb{CE}^{\max}_{\mathcal{M},s} \stackrel{\text{def}}{=} \sup_{\mathfrak{S}} \mathbb{CE}^{\mathfrak{S}}_{\mathcal{M},s} \in \mathbb{R} \cup \{\infty\} \quad \text{where} \quad \mathbb{CE}^{\mathfrak{S}}_{\mathcal{M},s} = \mathbb{E}^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond\!\!\!\!+ F|\Diamond G)$$

Here, $\mathfrak{S}$ ranges over all schedulers $\mathfrak{S}$ with $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond G) > 0$ and $\Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond F|\Diamond G) = 1$. If $\mathcal{M}$ and its initial state are clear from the context, we often simply write $\mathbb{CE}^{\max}$ resp. $\mathbb{CE}^{\mathfrak{S}}$. We assume that all states in $\mathcal{M}$ are reachable from s_{init} and $s_{init} \notin F \cup G$ (as $\mathbb{CE}^{\max} = 0$ if $s \in F$ and $\mathbb{CE}^{\max} = \mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!+ F)$ if $s \in G \setminus F$).

3 Finiteness and Upper Bound

Checking Finiteness. We sketch a polynomially time-bounded algorithm that takes as input an MDP $\mathcal{M} = (S, Act, P, s_{init}, rew)$ with two distinguished subsets F and G of S such that $\Pr^{\max}_{\mathcal{M},s_{init}}(\Diamond F|\Diamond G) = 1$. If $\mathbb{CE}^{\max} = \mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!+ F|\Diamond G) = \infty$ then the output is "no". Otherwise, the output is an MDP $\hat{\mathcal{M}} = (\hat{S}, \hat{Act}, \hat{P}, \hat{s}_{init}, \hat{rew})$ with two trap states *goal* and *fail* such that:

(1) $\mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!-\, F|\Diamond G) = \mathbb{E}^{\max}_{\hat{\mathcal{M}},\hat{s}_{init}}(\Diamond\!\!\!\!-\, goal|\Diamond goal)$,

(2) $\hat{s} \models \exists\Diamond goal$ and $\Pr^{\min}_{\hat{\mathcal{M}},\hat{s}}\big(\Diamond(goal \vee fail)\big) = 1$ for all states $\hat{s} \in \hat{S} \setminus \{fail\}$, and

(3) $\hat{\mathcal{M}}$ does not have critical schedulers where a scheduler $\mathfrak{U}$ for $\hat{\mathcal{M}}$ is said to be critical iff $\Pr^{\mathfrak{U}}_{\hat{\mathcal{M}},\hat{s}_{init}}(\Diamond fail) = 1$ and there is a reachable positive $\mathfrak{U}$-cycle.[2]

We provide here the main ideas of the algorithms and refer to Appendix C of [13] for the details. The algorithm first transforms $\mathcal{M}$ into an MDP $\tilde{\mathcal{M}}$ that permits to assume $F = G = goal$. Intuitively, $\tilde{\mathcal{M}}$ simulates $\mathcal{M}$, while operating in four modes: "normal mode", "after G", "after F" and "goal". $\tilde{\mathcal{M}}$ starts in normal mode where it behaves as $\mathcal{M}$ as long as neither F nor G have been visited. If a $G \setminus F$-state has been reached in normal mode then $\tilde{\mathcal{M}}$ switches to the mode "after G". Likewise, as soon as an $F \setminus G$-state has been reached in normal mode then $\tilde{\mathcal{M}}$ switches to the mode "after F". $\tilde{\mathcal{M}}$ enters the goal mode (consisting of a single trap state *goal*) as soon as a path fragment containing a state in F and a state in G has been generated. This is the case if $\mathcal{M}$ visits an F-state in mode "after G" or a G-state in mode "after F", or a state in $F \cap G$ in the normal mode. The rewards in the normal mode and in mode "after G" are precisely as in $\mathcal{M}$, while the rewards are 0 in all other cases. We then remove all states $\tilde{s}$ in the "after G" mode with $\Pr^{\max}_{\tilde{\mathcal{M}},\tilde{s}}(\Diamond goal) < 1$, collapse all states $\tilde{s}$ in $\tilde{\mathcal{M}}$ with $\tilde{s} \not\models \exists\Diamond goal$ into a single trap state called *fail* and add zero-reward transitions to *fail* from all states $\tilde{s}$ that are not in the "after G" mode and $\Pr^{\max}_{\tilde{\mathcal{M}},\tilde{s}}(\Diamond goal) = 0$. Using techniques as in the unconditional case [22] we can check whether $\tilde{\mathcal{M}}$ has positive end components, i.e., end components with at least one state-action pair (s,α) with $rew(s,\alpha) > 0$. If so, then $\mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!-\, F|\Diamond G) = \infty$. Otherwise, we collapse each maximal end component of $\tilde{\mathcal{M}}$ into a single state.

Let $\hat{\mathcal{M}}$ denote the resulting MDP. It satisfies (1) and (2). Property (3) holds iff $\mathbb{E}^{\max}_{\hat{\mathcal{M}},\hat{s}_{init}}(\Diamond\!\!\!\!-\, goal|\Diamond goal) < \infty$. This condition can be checked in polynomial time using a graph analysis in the sub-MDP of $\hat{\mathcal{M}}$ consisting of the states $\hat{s}$ with $\Pr^{\min}_{\hat{\mathcal{M}},\hat{s}}(\Diamond goal) = 0$ (see Appendix C of [13]).

Computing an Upper Bound. Due to the transformation used for checking finiteness of the maximal conditional expectation, we can now suppose that $\mathcal{M} = \hat{\mathcal{M}}$, $F = G = \{goal\}$ and that (2) and (3) hold. We now present a technique to compute an upper bound $\mathbb{CE}^{\mathrm{ub}}$ for $\mathbb{CE}^{\max}$. The upper bound will be used later to determine a saturation point from which on optimal schedulers behave memoryless (see Sect. 4).

We consider the MDP $\mathcal{M}'$ simulating $\mathcal{M}$, while operating in two modes. In its first mode, $\mathcal{M}'$ attaches the reward accumulated so far to the states. More precisely, the states of $\mathcal{M}'$ in its first mode have the form $\langle s,r\rangle \in S \times \mathbb{N}$ where $0 \leqslant r \leqslant R$ and $R = \sum_{s\in S'} \max\{rew_{\mathcal{M}'}(s,\alpha) : \alpha \in Act_{\mathcal{M}'}(s)\}$. The initial state of $\mathcal{M}'$ is $s'_{init} = \langle s_{init}, 0\rangle$. The reward for the state-action pairs $(\langle s,r\rangle,\alpha)$ where $r + rew(s,\alpha) \leqslant R$ is 0. If $\mathcal{M}'$ fires an action α in state $\langle s,r\rangle$ where

[2] The latter means a $\mathfrak{U}$-path $\pi = s_0\,\alpha_0\,s_1\,\alpha_1 \ldots \alpha_{k-1}\,s_k$ where $s_0 = \hat{s}_{init}$ and $s_i = s_k$ for some $i \in \{0,1,\ldots,k{-}1\}$ such that $\hat{rew}(s_j,\alpha_j) > 0$ for some $j \in \{i,\ldots,k{-}1\}$.

$r' \stackrel{\text{def}}{=} r + rew(s,\alpha) > R$ then it switches to the second mode, while earning reward r'. In its second mode $\mathcal{M}'$ behaves as $\mathcal{M}$ without additional annotations of the states and earning the same rewards as $\mathcal{M}$. From the states $\langle goal, r\rangle$, $\mathcal{M}'$ moves to *goal* with probability 1 and reward r. There is a one-to-one correspondence between the schedulers for $\mathcal{M}$ and $\mathcal{M}'$ and the switch from $\mathcal{M}$ to $\mathcal{M}'$ does not affect the probabilities and the accumulated rewards until reaching *goal*.

Let $\mathcal{N}$ denote the MDP resulting from $\mathcal{M}'$ by adding reset-transitions from *fail* (as a state of the second mode) and the copies $\langle fail, r\rangle$ in the first mode to the initial state s'_{init}. The reward of all reset transitions is 0. The reset-mechanism has been taken from [11] where it has been introduced as a technique to compute maximal conditional probabilities for reachability properties. Intuitively, $\mathcal{N}$ "discards" all paths of $\mathcal{M}'$ that eventually enter *fail* and "redistributes" their probabilities to the paths that eventually enter the goal state. In this way, $\mathcal{N}$ mimics the conditional probability measures $\Pr^{\mathfrak{S}}_{\mathcal{M}',s'_{init}}(\,\cdot\,|\Diamond goal) = \Pr^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\,\cdot\,|\Diamond goal)$ for prefix-independent path properties. Paths π from s_{init} to *goal* in $\mathcal{M}$ are simulated in $\mathcal{N}$ by paths of the form $\varrho = \xi_1; \ldots \xi_k; \pi$ where ξ_i is a cycle in $\mathcal{N}$ with $first(\xi_i) = s'_{init}$ and ξ_i's last transition is a reset-transition from some fail-state to s'_{init}. Thus, $rew(\pi) \leqslant rew_{\mathcal{N}}(\varrho)$. The distinction between the first and second mode together with property (3) ensure that the new reset-transitions do not generate positive end components in $\mathcal{N}$. By the results of [22], the maximal unconditional expected accumulated reward in $\mathcal{N}$ is finite and we have:

$$\mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!\cdot\, goal | \Diamond goal) = \mathbb{E}^{\max}_{\mathcal{M}',s'_{init}}(\Diamond\!\!\!\!\cdot\, goal | \Diamond goal) \leqslant \mathbb{E}^{\max}_{\mathcal{N},s'_{init}}(\Diamond\!\!\!\!\cdot\, goal)$$

Hence, we can deal with $\mathbb{CE}^{\text{ub}} = \mathrm{E}^{\max}_{\mathcal{N},s'_{init}}(\Diamond\!\!\!\!\cdot\, goal)$, which is computable in time polynomial in the size of $\mathcal{N}$ by the algorithm proposed in [22]. As $size(\mathcal{N}) = \Theta(R \cdot size(\mathcal{M}))$ we obtain a pseudo-polynomial time bound for the general case. If, however, $\Pr^{\min}_{\mathcal{M},s}(\Diamond goal) > 0$ for all states $s \in S \setminus \{fail\}$ then there is no need for the detour via $\mathcal{M}'$ and we can apply the reset-transformation $\mathcal{M} \rightsquigarrow \mathcal{N}$ by adding a reset-transition from *fail* to s_{init} with reward 0, in which case the upper bound $\mathbb{CE}^{\text{ub}} = \mathbb{E}^{\max}_{\mathcal{N},s_{init}}(\Diamond\!\!\!\!\cdot\, goal)$ is obtained in time polynomial in the size of $\mathcal{M}$. For details we refer to Appendix C of [13].

4 Threshold Algorithm and Computing Optimal Schedulers

In what follows, we suppose that $\mathcal{M} = (S, Act, P, s_{init}, rew)$ is an MDP with two trap states *goal* and *fail* such that $s \models \exists\Diamond goal$ for all states $s \in S \setminus \{fail\}$ and $\min_{s\in S} \Pr^{\min}_{\mathcal{M},s}(\Diamond(goal \vee fail)) = 1$ and $\mathbb{CE}^{\max} = \mathbb{E}^{\max}_{\mathcal{M},s_{init}}(\Diamond\!\!\!\!\cdot\, goal | \Diamond goal) < \infty$.

A scheduler $\mathfrak{S}$ is said to be *reward-based* if $\mathfrak{S}(\pi) = \mathfrak{S}(\pi')$ for all finite paths π, π' with $(last(\pi), rew(\pi)) = (last(\pi'), rew(\pi'))$. Thus, deterministic reward-based schedulers can be seen as functions $\mathfrak{S} : S \times \mathbb{N} \to Act$. We show in Appendix D of [13] that $\mathbb{CE}^{\max}$ equals the supremum of the values $\mathbb{CE}^{\mathfrak{S}}$, when ranging over all deterministic reward-based schedulers $\mathfrak{S}$ with $\Pr^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\Diamond goal) > 0$.

The basis of our algorithms are the following two observations. First, there exists a saturation point $\wp \in \mathbb{N}$ such that the optimal decision for all paths π with $rew(\pi) \geqslant \wp$ is to maximize the probability for reaching the goal state (see Proposition 4.1 below). The second observation is a technical statement that will be used at several places. Let $\rho, \theta, \zeta, r, x, y, z, p \in \mathbb{R}$ with $0 \leqslant p, x, y, z \leqslant 1$, $p > 0$, $y > z$ and $x + z > 0$ and let

$$\mathsf{A} = \frac{\rho + p(ry + \theta)}{x + py}, \quad \mathsf{B} = \frac{\rho + p(rz + \zeta)}{x + pz} \quad \text{and} \quad \mathsf{C} = \max\{\mathsf{A}, \mathsf{B}\}$$

Then:

$$\mathsf{A} \geqslant \mathsf{B} \quad \text{iff} \quad r + \frac{\theta - \zeta}{y - z} \geqslant \mathsf{C} \quad \text{iff} \quad \theta - (\mathsf{C} - r)y \;\geqslant\; \zeta - (\mathsf{C} - r)z \qquad (\dagger)$$

and the analogous statement for $>$ rather than $\geqslant$. For details, see Appendix G of [13]. We will apply this observation in different nuances. To give an idea how to apply statement (†), suppose $\mathsf{A} = \mathbb{CE}^{\mathfrak{T}}$ and $\mathsf{B} = \mathbb{CE}^{\mathfrak{U}}$ where $\mathfrak{T}$ and $\mathfrak{U}$ are reward-based schedulers that agree for all paths ϱ that do not have a prefix π with $rew(\pi) = r$ where $last(\pi)$ is a non-trap state, in which case x denotes the probability for reaching *goal* from s_{init} along such a path ϱ and ρ stands for the corresponding partial expectation, while p denotes the probability of the paths π from s_{init} to some non-trap state with $rew(\pi) = r$. The crucial observation is that $r + (\theta - \zeta)/(y - z)$ does not depend on x, ρ, p. Thus, if $r + (\theta - \zeta)/(y - z) \geqslant \mathbb{CE}^{\text{ub}}$ for some upper bound $\mathbb{CE}^{\text{ub}}$ of $\mathbb{CE}^{\max}$ then (†) allows to conclude that $\mathfrak{T}$'s decisions for the state-reward pairs (s, r) are better than $\mathfrak{U}$, independent of x, ρ and p.

Let $R \in \mathbb{N}$ and $\mathfrak{S}, \mathfrak{T}$ be reward-based schedulers. The *residual* scheduler $\mathfrak{S}{\restriction}R$ is given by $(\mathfrak{S}{\restriction}R)(s, r) = \mathfrak{S}(s, R{+}r)$. $\mathfrak{S} \lhd_R \mathfrak{T}$ denotes the unique scheduler that agrees with $\mathfrak{S}$ for all state-reward pairs (s, r) where $r < R$ and $(\mathfrak{S} \lhd_R \mathfrak{T}){\restriction}R = \mathfrak{T}$. We write $\mathrm{E}^{\mathfrak{S}}_{\mathcal{M},s}$ for the *partial expectation*

$$\mathrm{E}^{\mathfrak{S}}_{\mathcal{M},s} = \sum_{r=0}^{\infty} \Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond^{=r} goal) \cdot r$$

Thus, $\mathrm{E}^{\mathfrak{T}}_{\mathcal{M},s} = \mathbb{E}^{\mathfrak{T}}_{\mathcal{M},s}(\Diamond\!\!\!\!-\, goal)$ if $\Pr^{\mathfrak{T}}_{\mathcal{M},s}(\Diamond goal) = 1$, while $\mathrm{E}^{\mathfrak{T}}_{\mathcal{M},s} < \infty = \mathbb{E}^{\mathfrak{T}}_{\mathcal{M},s}(\Diamond\!\!\!\!-\, goal)$ if $\Pr^{\mathfrak{T}}_{\mathcal{M},s}(\Diamond goal) < 1$.

Proposition 4.1. *There exists a natural number $\wp$ (called* saturation point *of $\mathcal{M}$) and a deterministic memoryless scheduler $\mathfrak{M}$ such that:*

(a) $\mathbb{CE}^{\mathfrak{T}} \;\leqslant\; \mathbb{CE}^{\mathfrak{T} \lhd_\wp \mathfrak{M}}$ *for each scheduler $\mathfrak{T}$ with* $\Pr^{\mathfrak{T}}_{\mathcal{M},s_{init}}(\Diamond goal) > 0$, *and*
(b) $\mathbb{CE}^{\mathfrak{S}} \;=\; \mathbb{CE}^{\max}$ *for some deterministic reward-based scheduler $\mathfrak{S}$ such that* $\Pr^{\mathfrak{S}}_{\mathcal{M},s_{init}}(\Diamond goal) > 0$ *and* $\mathfrak{S}{\restriction}\wp = \mathfrak{M}$.

The proof of Proposition 4.1 (see Appendices E and F of [13]) is constructive and yields a polynomial-time algorithm for generating a scheduler $\mathfrak{M}$ as in Proposition 4.1 and a pseudo-polynomial algorithm for the computation of a saturation point $\wp$.

Scheduler $\mathfrak{M}$ maximizes the probability to reach *goal* from each state. If there are two or more such schedulers, then $\mathfrak{M}$ is one where the conditional expected accumulated reward until reaching goal is maximal under all schedulers $\mathfrak{U}$ with $\Pr^{\mathfrak{U}}_{\mathcal{M},s}(\Diamond goal) = \Pr^{\max}_{\mathcal{M},s}(\Diamond goal)$ for all states s. Such a scheduler $\mathfrak{M}$ is computable in polynomial time using linear programming techniques. (See Appendix E of [13].)

The idea for the computation of the saturation point is to compute the threshold $\wp$ above which the scheduler $\mathfrak{M}$ becomes optimal. For this we rely on statement (†) where θ/y stands for the conditional expectation under $\mathfrak{M}$, ζ/z for the conditional expectation under an arbitrary scheduler $\mathfrak{S}$ and $\mathsf{C} = \mathbb{CE}^{\mathrm{ub}}$ is an upper bound of $\mathbb{CE}^{\max}$ (see Theorem 1), while $r = \wp$ is the wanted value. More precisely, for $s \in S$, let $\theta_s = \mathrm{E}^{\mathfrak{M}}_{\mathcal{M},s}$, $y_s = \Pr^{\mathfrak{M}}_{\mathcal{M},s}(\Diamond goal) = \Pr^{\max}_{\mathcal{M},s}(\Diamond goal)$. To compute a saturation point we determine the smallest value $\wp \in \mathbb{N}$ such that

$$\theta_s - (\mathbb{CE}^{\mathrm{ub}} - \wp) \cdot y_s = \max_{\mathfrak{S}} \left(\mathrm{E}^{\mathfrak{S}}_{\mathcal{M},s} - (\mathbb{CE}^{\mathrm{ub}} - \wp) \cdot \Pr^{\mathfrak{S}}_{\mathcal{M},s}(\Diamond goal) \right)$$

for all states s where $\mathfrak{S}$ ranges over all schedulers for $\mathcal{M}$. In Appendix F of [13] we show that instead of the maximum over all schedulers $\mathfrak{S}$ it suffices to take the local maximum over all "one-step-variants" of $\mathfrak{M}$. That is, a saturation point is obtained by $\wp = \max\{\lceil \mathbb{CE}^{\mathrm{ub}} - D \rceil, 0\}$ where

$$D \;=\; \min \left\{ (\theta_s - \theta_{s,\alpha})/(y_s - y_{s,\alpha}) \;:\; s \in S, \alpha \in Act(s), y_{s,\alpha} < y_s \right\}$$

and $y_{s,\alpha} = \sum_{t \in S} P(s,\alpha,t) \cdot y_t$ and $\theta_{s,\alpha} = rew(s,\alpha) \cdot y_{s,\alpha} + \sum_{t \in S} P(s,\alpha,t) \cdot \theta_t$.

Example 4.2. The so obtained saturation point for the MDP $\mathcal{M}[\mathfrak{r}]$ in Fig. 1 is $\wp = \lceil \mathbb{CE}^{\mathrm{ub}} + 1 \rceil$. Note that only state $s = s_2$ behaves nondeterministically, and $\mathfrak{M}(s) = \alpha$, $y_s = y_{s,\alpha} = 1$, $\theta_s = \theta_{s,\alpha} = 0$, while $y_{s,\beta} = \theta_{s,\beta} = \frac{1}{2}$. This yields $D = (0 - \frac{1}{2})/(1 - \frac{1}{2}) = -1$. Thus, $\wp \geqslant \mathfrak{r}+2$ as $\mathbb{CE}^{\mathrm{ub}} \geqslant \mathbb{CE}^{\max} > \mathfrak{r}$. ∎

The logarithmic length of $\wp$ is polynomial in the size of $\mathcal{M}$. Thus, the value (i.e., the length of an unary encoding) of $\wp$ can be exponential in $size(\mathcal{M})$. This is unavoidable as there are families $(\mathcal{M}_k)_{k \in \mathbb{N}}$ of MDPs where the size of $\mathcal{M}_k$ is in $\mathcal{O}(k)$, while 2^k is a lower bound for the smallest saturation point of $\mathcal{M}_k$. This, for instance, applies to the MDPs $\mathcal{M}_k = \mathcal{M}[2^k]$ where $\mathcal{M}[\mathfrak{r}]$ is as in Fig. 1. Recall from Example 1.1 that the scheduler $\mathfrak{S}_{\mathfrak{r}+2}$ that selects β by the first $\mathfrak{r}+2$ visits of s and α for the $(\mathfrak{r}+3)$-rd visit of s is optimal for $\mathcal{M}[\mathfrak{r}]$. Hence, the smallest saturation point for $\mathcal{M}[2^k]$ is 2^k+2.

Threshold Algorithm. The input of the threshold algorithm is an MDP $\mathcal{M}$ as above and a non-negative rational number ϑ. The task is to generate a deterministic reward-based scheduler $\mathfrak{S}$ with $\mathfrak{S}{\uparrow}\wp = \mathfrak{M}$ (where $\mathfrak{M}$ and $\wp$ are as in Proposition 4.1) such that $\mathbb{CE}^{\mathfrak{S}} > \vartheta$ if $\mathbb{CE}^{\max} > \vartheta$, and $\mathbb{CE}^{\mathfrak{S}} = \vartheta$ if $\mathbb{CE}^{\max} = \vartheta$. If $\mathbb{CE}^{\max} < \vartheta$ then the output of the threshold algorithm is "no".[3]

[3] The threshold algorithm solves all four variants of the threshold problem. E.g., $\mathbb{CE}^{\max} \leqslant \vartheta$ iff $\mathbb{CE}^{\mathfrak{S}} = \vartheta$, while $\mathbb{CE}^{\max} < \vartheta$ iff the threshold algorithm returns "no".

The algorithm operates level-wise and determines *feasible* actions $action(s,r)$ for all non-trap states s and $r = \wp{-}1, \wp{-}2, \ldots, 0$, using the decisions $action(\cdot,i)$ for the levels $i \in \{r{+}1, \ldots, \wp\}$ that have been treated before and linear programming techniques to treat zero-reward loops. In this context, feasibility is understood with respect to the following condition: If $\mathbb{CE}^{\max} \trianglerighteq \vartheta$ where $\trianglerighteq \in \{>, \geqslant\}$ then there exists a reward-based scheduler $\mathfrak{S}$ with $\mathbb{CE}^{\mathfrak{S}} \trianglerighteq \vartheta$ and $\mathfrak{S}(s,R) = action(s, \min\{\wp, R\})$ for all $R \geqslant r$.

The algorithm stores for each state-reward pair (s,r) the probabilities $y_{s,r}$ to reach *goal* from s and the corresponding partial expectation $\theta_{s,r}$ for the scheduler given by the decisions in the action table. The values for $r = \wp$ are given by $action(s,\wp) = \mathfrak{M}(s)$, $y_{s,\wp} = \Pr^{\mathfrak{M}}_{\mathcal{M},s}(\Diamond goal)$ and $\theta_{s,\wp} = \mathrm{E}^{\mathfrak{M}}_{\mathcal{M},s}$. The candidates for the decisions at level $r < \wp$ are given by the deterministic memoryless schedulers $\mathfrak{P}$ for $\mathcal{M}$. We write $\mathfrak{P}_+$ for the reward-based scheduler given by $\mathfrak{P}_+(s,0) = \mathfrak{P}(s)$ and $\mathfrak{P}_+(s,i) = action(s, \min\{\wp, r{+}i\})$ for $i \geqslant 1$. Let $y_{s,r,\mathfrak{P}} = \Pr^{\mathfrak{P}_+}_{\mathcal{M},s}(\Diamond goal)$ and $\theta_{s,r,\mathfrak{P}} = \mathrm{E}^{\mathfrak{P}_+}_{\mathcal{M},s}$ be the corresponding partial expectation.

To determine feasible actions for level r, the threshold algorithm makes use of a variant of (†) stating that if $\theta - (\vartheta{-}r)y \geqslant \zeta - (\vartheta{-}r)z$ and $\mathsf{B} \trianglerighteq \vartheta$ then $\mathsf{A} \trianglerighteq \vartheta$, where A and B are as in (†) and the requirement $y > z$ is dropped. Thus, the aim of the threshold algorithm is to compute a deterministic memoryless scheduler $\mathfrak{P}^*$ for $\mathcal{M}$ such that the following condition (∗) holds:

$$\theta_{s,r,\mathfrak{P}^*} - (\vartheta{-}r) \cdot y_{s,r,\mathfrak{P}^*} = \max_{\mathfrak{P}} \left(\theta_{s,r,\mathfrak{P}} - (\vartheta{-}r) \cdot y_{s,r,\mathfrak{P}} \right) \qquad (*)$$

Such a scheduler $\mathfrak{P}^*$ is computable in time polynomial in the size of $\mathcal{M}$ (without the explicit consideration of all schedulers $\mathfrak{P}$ and their extensions $\mathfrak{P}_+$) using the following linear program with one variable x_s for each state. The objective is to minimize $\sum_{s \in S} x_s$ subject to the following conditions:

(1) If $s \in S \setminus \{goal, fail\}$ then for each action $\alpha \in Act(s)$ with $rew(s,\alpha) = 0$:

$$x_s \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_t$$

(2) If $s \in S \setminus \{goal, fail\}$ then for each action $\alpha \in Act(s)$ with $rew(s,\alpha) > 0$:

$$x_s \geqslant \sum_{t \in S} P(s,\alpha,t) \cdot \left(\theta_{t,R} + rew(s,\alpha) \cdot y_{t,R} \; - \; (\vartheta{-}r) \cdot y_{t,R} \right)$$

where $R = \min\{\wp, r{+}rew(s,\alpha)\}$

(3) For the trap states: $x_{goal} = r - \vartheta \quad$ and $\quad x_{fail} = 0$.

This linear program has a unique solution $(x^*_s)_{s \in S}$. Let $Act^*(s)$ denote the set of actions $\alpha \in Act(s)$ such that the following constraints (E1) and (E2) hold:

(E1) If $rew(s,\alpha) = 0$ then: $x^*_s = \sum_{t \in S} P(s,\alpha,t) \cdot x^*_t$

(E2) If $rew(s,\alpha) > 0$ and $R = \min\{\, \wp, \, r{+}rew(s,\alpha) \,\}$ then:

$$x^*_s \; = \; \sum_{t \in S} P(s,\alpha,t) \cdot \left(\theta_{t,R} + rew(s,\alpha) \cdot y_{t,R} \; - \; (\vartheta{-}r) \cdot y_{t,R} \right)$$

Let $\mathcal{M}^* = \mathcal{M}^*_{r,\vartheta}$ denote the MDP with state space S induced by the state-action pairs (s,α) with $\alpha \in Act^*(s)$ where the positive-reward actions are redirected to the trap states. Formally, for $s,t \in S$, $\alpha \in Act^*(s)$ we let $P_{\mathcal{M}^*}(s,\alpha,t) = P(s,\alpha,t)$ if $rew(s,\alpha) = 0$ and $P_{\mathcal{M}^*}(s,\alpha,goal) = \sum_{t\in S} P(s,\alpha,t)\cdot y_{t,R}$ and $P_{\mathcal{M}^*}(s,\alpha,fail) = 1 - P_{\mathcal{M}^*}(s,\alpha,goal)$ if $rew(s,\alpha) > 0$ and $R = \min\{\wp, r+rew(s,\alpha)\}$. The reward structure of $\mathcal{M}^*$ is irrelevant for our purposes.

A scheduler $\mathfrak{P}^*$ satisfying $(*)$ is obtained by computing a memoryless deterministic scheduler for $\mathcal{M}^*$ with $\Pr^{\mathfrak{P}^*}_{\mathcal{M}^*,s}(\Diamond goal) = \Pr^{\max}_{\mathcal{M}^*,s}(\Diamond goal)$ for all states s. This scheduler $\mathfrak{P}^*$ indeed provides feasible decisions for level r, i.e., if $\mathbb{CE}^{\max} \trianglerighteq \vartheta$ where $\trianglerighteq \in \{>,\geqslant\}$ then there exists a reward-based scheduler $\mathfrak{S}$ with $\mathbb{CE}^{\mathfrak{S}} \trianglerighteq \vartheta$, $\mathfrak{S}(s,r) = \mathfrak{P}^*(s)$ and $\mathfrak{S}(s,R) = action(s,\min\{\wp,R\})$ for all $R > r$.

The threshold algorithm then puts $action(s,r) = \mathfrak{P}^*(s)$ and computes the values $y_{s,r}$ and $\theta_{s,r}$ as follows. Let T denote the set of states $s \in S \setminus \{goal, fail\}$ where $rew(s,\mathfrak{P}^*(s)) > 0$. For $s \in T$, the values $y_{s,r} = y_{s,r,\mathfrak{P}^*}$ and $\theta_{s,r} = \theta_{s,r,\mathfrak{P}^*}$ can be derived directly from the results obtained for the previously treated levels $r{+}1,\ldots,\wp$ as we have:

$$y_{s,r} = \sum_{t\in S} P(s,\alpha,t)\cdot y_{t,R} \quad \text{and} \quad \theta_{s,r} = rew(s,\alpha)\cdot y_{s,r} + \sum_{t\in S} P(s,\alpha,t)\cdot \theta_{t,R}$$

where $\alpha = \mathfrak{P}^*(s)$ and $R = \min\{\wp, r{+}rew(s,\alpha)\}$. For the states $s \in S \setminus T$:

$$y_{s,r} = \sum_{t\in T} \Pr^{\mathfrak{P}^*}_{\mathcal{M},s}(\neg T \,\mathrm{U}\, t)\cdot y_{t,r} \quad \text{and} \quad \theta_{s,r} = \sum_{t\in T} \Pr^{\mathfrak{P}^*}_{\mathcal{M},s}(\neg T \,\mathrm{U}\, t)\cdot \theta_{t,r}$$

Having treated the last level $r = 0$, the output of the algorithm is as follows. Let $\mathfrak{S}$ be the scheduler given by the action table $action(\cdot)$. For the conditional expectation we have $\mathbb{CE}^{\mathfrak{S}} = \theta_{s_{init},0}/y_{s_{init},0}$ if $y_{s_{init},0} > 0$. If $y_{s_{init},0} = 0$ or $\theta_{s_{init},0}/y_{s_{init},0} < \vartheta$ then the algorithm returns the answer "no". Otherwise, the algorithm returns $\mathfrak{S}$, in which case $\mathbb{CE}^{\mathfrak{S}} > \vartheta$ or $\mathbb{CE}^{\mathfrak{S}} = \vartheta = \mathbb{CE}^{\max}$. Proofs for the soundness and the pseudo-polynomial time complexity are provided in Appendix G of [13].

Example 4.3. For the MDP $\mathcal{M}[\mathfrak{r}]$ in Example 1.1, scheduler $\mathfrak{M}$ selects action α for state $s = s_2$. Thus, $action(s,\wp) = \alpha$ for the computed saturation point $\wp \geqslant \mathfrak{r}+2$ (see Example 4.2). The threshold algorithm for each positive rational threshold ϑ computes for each level $r = \wp{-}1, \wp{-}2, \ldots, 1, 0$ where $action(s, r+1) = \alpha$, the value $x^*_s = \max\{r{-}\vartheta, \frac{1}{2} + \frac{1}{2}(r{-}\vartheta)\}$ and the action set $Act^*(s) = \{\alpha\}$ if $r > \vartheta{+}1$, $Act^*(s) = \{\alpha,\beta\}$ if $r = \vartheta{+}1$ and $Act^*(s) = \{\beta\}$ if $r < \vartheta{+}1$. Thus, if $n = \min\{\wp, \lceil\vartheta{+}1\rceil\}$ then $action(s,r) = \alpha$, $y_{s,r} = 1$, $\theta_{s,r} = 0$ for $r \in \{n,\ldots,\wp\}$, while $action(s,n{-}k) = \beta$, $y_{s,n-k} = 1/2^k$, $\theta_{s,n-k} = k/2^k$ for $k = 1,\ldots,n$. That is, the threshold algorithm computes the scheduler $\mathfrak{S}_n$ that selects β for the first n visits of s and α for the $(n{+}1)$-st visit of s. Thus, if $\mathfrak{r} \leqslant \vartheta < \mathfrak{r}{+}1$ then $n = \mathfrak{r}{+}2$, in which case the computed scheduler $\mathfrak{S}_n$ is optimal (see Example 1.1). The returned answer depends on whether $\vartheta \leqslant \mathbb{CE}^{\max}$. If, for instance, $\vartheta = \frac{\mathfrak{r}}{2}$ and $\mathfrak{r} > 0$ is even then the threshold algorithm returns the scheduler $\mathfrak{S}_n$ where $n = \frac{\mathfrak{r}}{2}{+}1$, whose conditional expectation is $\mathfrak{r} - (\frac{\mathfrak{r}}{2}{-}1)/(2^{\frac{\mathfrak{r}}{2}+1}{+}1) > \frac{\mathfrak{r}}{2} = \vartheta$. ∎

MDPs Without Zero-Reward Cycles and Acyclic MDPs. If $\mathcal{M}$ does not contain zero-reward cycles then there is no need for the linear program. Instead we can use a topological sorting of the states in the graph of the sub-MDP consisting of zero-reward actions and determine a scheduler $\mathfrak{P}^*$ satisfying $(*)$ directly. For acyclic MDPs, there is even no need for a saturation point. We can explore $\mathcal{M}$ using a recursive procedure and determine feasible decisions for each reachable state-reward pair (s, r) on the basis of $(*)$. This yields a polynomially space-bounded algorithm to decide whether $\mathbb{CE}^{\max} \trianglerighteq \vartheta$ in acyclic MDPs. (See Appendix I of [13].)

Construction of an Optimal Scheduler. Let *ThresAlgo*$[\vartheta]$ denote the scheduler that is generated by calling the threshold algorithm for the threshold value ϑ. A simple approach is to apply the threshold algorithm iteratively:

let $\mathfrak{S}$ be the scheduler $\mathfrak{M}$ as in Proposition 4.1;
`REPEAT` $\vartheta := \mathbb{CE}^{\mathfrak{S}}$; $\mathfrak{S} :=$ *ThresAlgo*$[\vartheta]$ `UNTIL` $\vartheta = \mathbb{CE}^{\mathfrak{S}}$;
return ϑ and $\mathfrak{S}$

The above algorithm generates a sequence of deterministic reward-based schedulers that are memoryless from $\wp$ on with strictly increasing conditional expectations. The number of such schedulers is bounded by $md^{\wp}$ where md denotes the number of memoryless deterministic schedulers for $\mathcal{M}$. Hence, the algorithm terminates and correctly returns $\mathbb{CE}^{\max}$ and an optimal scheduler. As md can be exponential in the number of states, this simple algorithm has double-exponential time complexity.

To obtain a (single) exponential-time algorithm, we seek for better (larger, but still promising) threshold values than the conditional expectation of the current scheduler. We propose an algorithm that operates level-wise and freezes optimal decisions for levels $r = \wp, \wp{-}1, \wp{-}2, \ldots, 1, 0$. The algorithm maintains and successively improves a left-closed and right-open interval $I = [A, B[$ with $\mathbb{CE}^{\max} \in I$ and $\mathbb{CE}^{\mathfrak{S}} \in I$ for the current scheduler $\mathfrak{S}$.

Initialization. The algorithm starts with the scheduler $\mathfrak{S} =$ *ThresAlgo*$[\mathbb{CE}^{\mathfrak{M}}]$ where $\mathfrak{M}$ is as above. If $\mathbb{CE}^{\mathfrak{S}} = \mathbb{CE}^{\mathfrak{M}}$ then the algorithm immediately terminates. Suppose now that $\mathbb{CE}^{\mathfrak{S}} > \mathbb{CE}^{\mathfrak{M}}$. The initial interval is $I = [A, B[$ where $A = \mathbb{CE}^{\mathfrak{S}}$ and $B = \mathbb{CE}^{\mathrm{ub}}{+}1$ where $\mathbb{CE}^{\mathrm{ub}}$ is as in Theorem 1.

Level-wise Scheduler Improvement. The algorithm successively determines optimal decisions for the levels $r = \wp{-}1, \wp{-}2, \ldots, 1, 0$. The treatment of level r consists of a sequence of scheduler-improvement steps where at the same time the interval I is replaced with proper sub-intervals. The current scheduler $\mathfrak{S}$ has been obtained by the last successful run of the threshold algorithm, i.e., it has the form $\mathfrak{S} =$ *ThresAlgo*$[\vartheta]$ where $\mathbb{CE}^{\mathfrak{S}} > \vartheta$. Besides the decisions of $\mathfrak{S}$ (i.e., the actions $\mathfrak{S}(s, R)$ for all state-reward pairs (s, R) where $s \in S \setminus \{goal, fail\}$ and $R \in \{0, 1, \ldots, \wp\}$), the algorithm also stores the values $y_{s,R}$ and $\theta_{s,R}$ that have been computed in the threshold algorithm.[4] For the

[4] As the decisions of the already treated levels are optimal, the values $y_{s,R}$ and $\theta_{s,R}$ for $R \in \{r{+}1, \ldots, \wp\}$ can be reused in the calls of the threshold algorithms. That is, the calls of the threshold algorithm that are invoked in the scheduler-improvement steps at level r can skip levels $\wp, \wp{-}1, \ldots, r{+}1$ and only need to process levels $r, r{-}1, \ldots, 1, 0$.

current level r, the algorithm also computes for each state $s \in S \setminus \{goal, fail\}$ and each action $\alpha \in Act(s)$ the values $y_{s,r,\alpha} = \sum_{t \in S} P(s,\alpha,t) \cdot y_{t,R}$ and $\theta_{s,r,\alpha} = rew(s,\alpha) \cdot y_{s,r,\alpha} + \sum_{t \in S} P(s,\alpha,t) \cdot \theta_{t,R}$ where $R = \min\{\wp, r + rew(s,\alpha)\}$.

Scheduler-improvement Step. Let r be the current level, $I = [A, B[$ the current interval and $\mathfrak{S}$ the current scheduler with $\mathbb{CE}^{\max} \in I$. At the beginning of the scheduler-improvement step we have $\mathbb{CE}^{\mathfrak{S}} = A$. Let

$$\mathcal{I}_{\mathfrak{S},r} = \left\{ r + \tfrac{\theta_{s,r} - \theta_{s,r,\alpha}}{y_{s,r} - y_{s,r,\alpha}} \;:\; s \in S \setminus \{goal, fail\},\ \alpha \in Act(s),\ y_{s,r} > y_{s,r,\alpha} \right\}$$

$$\mathcal{I}^{\uparrow}_{\mathfrak{S},r} = \left\{ d \in \mathcal{I}_{\mathfrak{S},r} \;:\; d \geqslant \mathbb{CE}^{\mathfrak{S}} \right\} \qquad \mathcal{I}^{B}_{\mathfrak{S},r} = \left\{ d \in \mathcal{I}_{\mathfrak{S},r} \;:\; d < B \right\}$$

Intuitively, the values in $d \in \mathcal{I}^{B}_{\mathfrak{S},r}$ are the "most promising" threshold values, as according to statement (†) these are the points where the decision of the current scheduler $\mathfrak{S}$ for some state-reward pair (s,r) can be improved, provided that $\mathbb{CE}^{\max} > d$. (Note that the values in $\mathcal{I}_{\mathfrak{S},r} \setminus \mathcal{I}^{B}_{\mathfrak{S},r}$ can be discarded as $\mathbb{CE}^{\max} < B$.)

The algorithm proceeds as follows. If $\mathcal{I}^{B}_{\mathfrak{S},r} = \varnothing$ then no further improvements at level r are possible as the function $\mathfrak{P}^* = \mathfrak{S}(\cdot, r)$ satisfies $(*)$ for the (still unknown) value $\vartheta = \mathbb{CE}^{\max}$. See Appendix H of [13]. In this case:

- If $r = 0$ then the algorithm terminates with the answer $\mathbb{CE}^{\max} = \mathbb{CE}^{\mathfrak{S}}$ and $\mathfrak{S}$ as an optimal scheduler.
- If $r > 0$ then the algorithm goes to the next level $r-1$ and performs the scheduler-improvement step for $\mathfrak{S}$ at level $r-1$.

Suppose now that $\mathcal{I}^{B}_{\mathfrak{S},r}$ is nonempty. Let $\mathcal{K} = \mathcal{I}^{\uparrow}_{\mathfrak{S},r} \cup \{\mathbb{CE}^{\mathfrak{S}}\}$. The algorithm seeks for the largest value $\vartheta' \in \mathcal{K} \cap I$ such that $\mathbb{CE}^{\max} \geqslant \vartheta'$. More precisely, it successively calls the threshold algorithm for the threshold value $\vartheta' = \max(\mathcal{K} \cap I)$ and performs the following steps for the generated scheduler $\mathfrak{S}' = ThresAlgo[\vartheta']$:

- If the result of the threshold algorithm is "no" and $\Pr^{\mathfrak{S}'}_{\mathcal{M},s_{init}}(\Diamond goal)$ is positive (in which case $\mathbb{CE}^{\mathfrak{S}'} \leqslant \mathbb{CE}^{\max} < \vartheta'$), then:
  - If $\mathbb{CE}^{\mathfrak{S}'} \leqslant A$ then the algorithm refines I by putting $B := \vartheta'$.
  - If $\mathbb{CE}^{\mathfrak{S}'} > A$ then the algorithm refines I by putting $A := \mathbb{CE}^{\mathfrak{S}'}$, $B := \vartheta'$ and adds $\mathbb{CE}^{\mathfrak{S}'}$ to $\mathcal{K}$ (Note that then $\mathbb{CE}^{\mathfrak{S}'} \in \mathcal{K} \cap I$, while $\mathbb{CE}^{\mathfrak{S}} \in \mathcal{K} \setminus I$.)
- Suppose now that $\mathbb{CE}^{\mathfrak{S}'} \geqslant \vartheta'$. The algorithm terminates if $\mathbb{CE}^{\mathfrak{S}'} = \vartheta'$, in which case $\mathfrak{S}'$ is optimal. Otherwise, i.e., if $\mathbb{CE}^{\mathfrak{S}'} > \vartheta'$, then the algorithm aborts the loop by putting $\mathcal{K} := \varnothing$, refines the interval I by putting $A := \mathbb{CE}^{\mathfrak{S}'}$, updates the current scheduler by setting $\mathfrak{S} := \mathfrak{S}'$ and performs the next scheduler-improvement step.

The soundness proof and complexity analysis can be found in Appendix H of [13], where (among others) we show that the scheduler-improvement step for schedulers $\mathfrak{S}$ with $\mathbb{CE}^{\mathfrak{S}} < \mathbb{CE}^{\max}$ terminates with some scheduler $\mathfrak{S}'$ such that $\mathbb{CE}^{\mathfrak{S}} < \mathbb{CE}^{\mathfrak{S}'}$. The total number of calls of the threshold algorithm is in $\mathcal{O}(\wp \cdot md \cdot |S| \cdot |Act|)$. This yields an exponential time bound as stated in Theorem 3.

Example 4.4. We regard again the MDP $\mathcal{M}[\mathfrak{r}]$ of Example 1.1 where we suppose $\mathfrak{r}$ is positive and even. The algorithm first computes $\mathbb{CE}^{\mathrm{ub}}$ (see Sect. 3), a saturation point $\wp \geqslant \mathfrak{r}{+}2$ (see Example 4.2), the scheduler $\mathfrak{M}$, its conditional expectation $\mathbb{CE}^{\mathfrak{M}} = \frac{\mathfrak{r}}{2}$ and the scheduler $\mathfrak{S} = \mathit{ThresAlgo}[\frac{\mathfrak{r}}{2}]$. The initial interval is $I = [A, B[$ where $A = \mathbb{CE}^{\mathfrak{S}} = \mathfrak{r} - (\frac{\mathfrak{r}}{2}{-}1)/(2^{\frac{\mathfrak{r}}{2}+1}{+}1)$ (see Example 4.3) and $B = \mathbb{CE}^{\mathrm{ub}}{+}1$. The scheduler improvement step for $\mathfrak{S}$ at levels $r = \wp{-}1, \ldots, \mathfrak{r}{+}1$ determines the set $\mathcal{I}_{\mathfrak{S},r} = \{r{-}1\}$ and calls the threshold algorithm for $\vartheta' = r{-}1$. These calls are not successful for $r = \wp{-}1, \ldots, \mathfrak{r}{+}2$. That is, the scheduler $\mathfrak{S}$ remains unchanged and the upper bound B is successively improved to $r{-}1$. At level $r = \mathfrak{r}{+}1$, the threshold algorithm is called for $\vartheta' = \mathfrak{r}$, which yields the optimal scheduler $\mathfrak{S}' = \mathit{ThresAlgo}[\vartheta']$ (see Example 4.3). ∎

Implementation and Experiments. We have implemented the algorithms presented in this paper as a prototypical extension of the model checker PRISM [27,28] and carried out initial experiments to demonstrate the general feasibility of our approach (see https://wwwtcs.inf.tu-dresden.de/ALGI/PUB/TACAS17/ and Appendix K of [13] for details).

5 Conclusion

Although the switch to conditional expectations appears rather natural to escape from the limitations of known solutions for unconditional extremal expected accumulated rewards, to the best of our knowledge computation schemes for conditional expected accumulated rewards have not been addressed before. Our results show that new techniques are needed to compute maximal conditional expectations, as optimal schedulers might need memory and local reasoning in terms of the past and possible future is not sufficient (Example 1.1). The key observations for our algorithms are the existence of a saturation point $\wp$ for the reward that has been accumulated so far, from which on optimal schedulers can behave memoryless, and a linear correlation between optimal decisions for all state-reward pairs (s, r) of the same reward level r (see $(*)$ and the linear program used in the threshold algorithm). The difficulty to reason about conditional expectations is also reflected in the achieved complexity-theoretic results stating that all variants of the threshold problem lie between PSPACE and EXPTIME. While PSPACE-completeness has been established for acyclic MDPs (Appendix I of [13]), the precise complexity for cyclic MDPs is still open. In contrast, optimal schedulers for unconditional expected accumulated rewards as well as for conditional reachability probabilities are computable in polynomial time [11,22].

Using standard automata-based approaches, our method can easily be generalized to compute maximal conditional expected rewards for regular co-safety conditions (rather than reachability conditions $\Diamond G$) and/or where the accumulation of rewards is "controlled" by a deterministic finite automaton as in the logics considered in [12,17] (rather than $\Diamond\!\!\!\!+ F$). In this paper, we restricted to MDPs with non-negative integer rewards. Non-negative rational rewards can be treated by multiplying all reward values with their least common multiple (Appendix J.1 of [13]). In the case of acyclic MDPs, our methods are even applicable if the MDP

has negative and positive rational rewards (Appendix J.2 of [13]). By swapping the sign of all rewards, this yields a technique to compute minimal conditional expectations in acyclic MDPs. We expect that minimal conditional expectations in cyclic MDPs with non-negative rewards can be computed using similar algorithms as we suggested for maximal conditional expectations. This as well as MDPs with negative and positive rewards will be addressed in future work.

References

1. Abdulla, P.A., Henda, N.B., Mayr, R.: Decisive Markov chains. Logical Methods Comput. Sci. **3**(4) (2007)
2. Acerbi, C., Tasche, D.: Expected shortfall: a natural coherent alternative to value at risk. Econ. notes **31**(2), 379–388 (2002)
3. Alvim, M.S., Andrés, M.E., Chatzikokolakis, K., Degano, P., Palamidessi, C.: On the information leakage of differentially-private mechanisms. J. Comput. Secur. **23**(4), 427–469 (2015)
4. Alvim, M.S., Chatzikokolakis, K., McIver, A., Morgan, C., Palamidessi, C., Smith, G.: Axioms for information leakage. In: Proceedings of Computer Security Foundations Symposium (CSF), pp. 77–92. IEEE Computer Society (2016)
5. Alvim, M.S., Chatzikokolakis, K., Palamidessi, C., Smith, G.: Measuring information leakage using generalized gain functions. In: Proceedings of Computer Security Foundations Symposium (CSF), pp. 265–279. IEEE Computer Society (2012)
6. Andrés, M.E.: Quantitative Analysis of Information Leakage in Probabilistic and Nondeterministic Systems. Ph.D. thesis, UB Nijmegen (2011)
7. Andrés, M.E., Palamidessi, C., van Rossum, P., Sokolova, A.: Information hiding in probabilistic concurrent systems. Theoret. Comput. Sci. **412**(28), 3072–3089 (2011)
8. Andrés, M.E., van Rossum, P.: Conditional probabilities over probabilistic and nondeterministic systems. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 157–172. Springer, Heidelberg (2008). doi:10.1007/978-3-540-78800-3_12
9. Baier, C., Dubslaff, C., Klein, J., Klüppelholz, S., Wunderlich, S.: Probabilistic model checking for energy-utility analysis. In: Breugel, F., Kashefi, E., Palamidessi, C., Rutten, J. (eds.) Horizons of the Mind. A Tribute to Prakash Panangaden. LNCS, vol. 8464, pp. 96–123. Springer, Heidelberg (2014). doi:10.1007/978-3-319-06880-0_5
10. Baier, C., Katoen, J.-P.: Principles of Model Checking. MIT Press, Cambridge (2008)
11. Baier, C., Klein, J., Klüppelholz, S., Märcker, S.: Computing conditional probabilities in Markovian models efficiently. In: Ábrahám, E., Havelund, K. (eds.) TACAS 2014. LNCS, vol. 8413, pp. 515–530. Springer, Heidelberg (2014). doi:10.1007/978-3-642-54862-8_43
12. Baier, C., Klein, J., Klüppelholz, S., Wunderlich, S.: Weight monitoring with linear temporal logic: complexity and decidability. In: Proceedings of Computer Science Logic/Logic in Computer Science (CSL-LICS), pp. 11:1–11:10. ACM (2014)
13. Baier, C., Klein, J., Klüppelholz, S. Wunderlich, S.: Maximizing the conditional expected reward for reaching the goal (extended version). arXiv:1701.05389 (2017)
14. Barthe, G., Espitau, T., Ferrer Fioriti, L.M., Hsu, J.: Synthesizing probabilistic invariants via Doob's decomposition. In: Chaudhuri, S., Farzan, A. (eds.) CAV 2016. LNCS, vol. 9779, pp. 43–61. Springer, Heidelberg (2016). doi:10.1007/978-3-319-41528-4_3

15. Bertsekas, D.P., Tsitsiklis, J.N.: An analysis of stochastic shortest path problems. Math. Oper. Res. **16**(3), 580–595 (1991)
16. Bertsekas, D.P., Yu, H.: Stochastic path problems under weak conditions. Technical report, M.I.T. Cambridge, Report LIDS 2909 (2016)
17. Boker, U., Chatterjee, K., Henzinger, T.A., Kupferman, O.: Temporal specifications with accumulative values. In: Proceedings of Logic in Computer Science (LICS), pp. 43–52. IEEE Computer Society (2011)
18. Brázdil, T., Brozek, V., Chatterjee, K., Forejt, V., Kucera, A.: Two views on multiple mean-payoff objectives in Markov decision processes. Logical Methods Comput. Sci. **10**(1) (2014)
19. Brázdil, T., Kučera, A.: Computing the expected accumulated reward and gain for a subclass of infinite Markov Chains. In: Sarukkai, S., Sen, S. (eds.) FSTTCS 2005. LNCS, vol. 3821, pp. 372–383. Springer, Heidelberg (2005). doi:10.1007/11590156_30
20. Chatterjee, K., Fu, H., Goharshady, A.K.: Termination analysis of probabilistic programs through Positivstellensatz's. In: Chaudhuri, S., Farzan, A. (eds.) CAV 2016. LNCS, vol. 9779, pp. 3–22. Springer, Heidelberg (2016). doi:10.1007/978-3-319-41528-4_1
21. Chatzikokolakis, K., Palamidessi, C., Braun, C.: Compositional methods for information-hiding. Math. Struct. Comput. Sci. **26**(6), 908–932 (2016)
22. Alfaro, L.: Computing minimum and maximum reachability times in probabilistic systems. In: Baeten, J.C.M., Mauw, S. (eds.) CONCUR 1999. LNCS, vol. 1664, pp. 66–81. Springer, Heidelberg (1999). doi:10.1007/3-540-48320-9_7
23. Gretz, F., Katoen, J., McIver, A.: Operational versus weakest pre-expectation semantics for the probabilistic guarded command language. Perform. Eval. **73**, 110–132 (2014)
24. Jansen, N., Kaminski, B.L., Katoen, J., Olmedo, F., Gretz, F., McIver, A.: Conditioning in probabilistic programming. In: Proceedings of Mathematical Foundations of Programming Semantics (MFPS), Electronic Notes Theoretical Computer Science, vol. 319, pp. 199–216 (2015)
25. Kallenberg, L.: Markov Decision Processes. Lecture Notes. University of Leiden, Leiden (2011)
26. Katoen, J.-P., Gretz, F., Jansen, N., Kaminski, B.L., Olmedo, F.: Understanding probabilistic programs. In: Meyer, R., Platzer, A., Wehrheim, H. (eds.) Correct System Design. LNCS, vol. 9360, pp. 15–32. Springer, Heidelberg (2015). doi:10.1007/978-3-319-23506-6_4
27. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV 2011. LNCS, vol. 6806, pp. 585–591. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22110-1_47
28. PRISM model checker. http://www.prismmodelchecker.org/
29. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley, New York (1994)
30. Randour, M., Raskin, J.-F., Sankur, O.: Variations on the stochastic shortest path problem. In: D'Souza, D., Lal, A., Larsen, K.G. (eds.) VMCAI 2015. LNCS, vol. 8931, pp. 1–18. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46081-8_1
31. Seber, G., Lee, A.: Linear Regression Analysis. Wiley Series in Probability and Statistics. Wiley, New York (2003)
32. Uryasev, S.: Conditional value-at-risk: optimization algorithms and applications. In Proceedings of Computational Intelligence and Financial Engineering (CIFEr), pp. 49–57. IEEE (2000)