The Odds of Staying on Budget

Christoph Haase[1](✉) and Stefan Kiefer[2]

[1] Laboratoire Spécification et Vérification (LSV), CNRS and ENS de Cachan, Cachan Cedex, France
haase@lsv.ens-cachan.fr

[2] Department of Computer Science, University of Oxford, Oxford, UK

Abstract. Given Markov chains and Markov decision processes (MDPs) whose transitions are labelled with non-negative integer costs, we study the computational complexity of deciding whether the probability of paths whose accumulated cost satisfies a Boolean combination of inequalities exceeds a given threshold. For acyclic Markov chains, we show that this problem is PP-complete, whereas it is hard for the PosSLP problem and in PSpace for general Markov chains. Moreover, for acyclic and general MDPs, we prove PSpace- and EXP-completeness, respectively. Our results have direct implications on the complexity of computing reward quantiles in succinctly represented stochastic systems.

1 Introduction

Computing the shortest path from s to t in a directed graph is a ubiquitous problem in computer science, so shortest-path algorithms such as Dijkstra's algorithm are a staple for every computer scientist. These algorithms work in polynomial time even if the edges are weighted, so it is easy to answer questions like:

(I) *Is it possible to travel from Copenhagen to Kyoto in less than 15 hours?*

The shortest-path problem becomes more intricate as soon as uncertainties are taken into account. For example, additional information such as "*there might be congestion in Singapore, so the Singapore route will, with probability 10%, trigger a delay of 1 hour*" leads to questions of the following kind:

(II) *Is there a travel plan avoiding trips longer than 15 hours with probability at least* 0.9*?*

Markov decision processes (MDPs) are the established model to formalise problems such as (II). We consider MDPs where each transition is equipped with a non-negative "weight". The weight could be interpreted as time, distance, reward, or—as in this paper—as *cost*. For another example, imagine the plan of a research project whose workflow can be modelled by a directed weighted graph. In each project state the investigators can hire a programmer, travel to collaborators, acquire new equipment, etc., but each action costs money, and the result (i.e., the next project state) is probabilistic. The objective is to meet the goals of the project before exceeding its budget for the total accumulated cost.

C. Haase—Supported by Labex Digicosme, Univ. Paris-Saclay, project VERICONISS.

S. Kiefer—Supported by a Royal Society University Research Fellowship.

M.M. Halldórsson et al. (Eds.): ICALP 2015, Part II, LNCS 9135, pp. 234–246, 2015.
DOI: 10.1007/978-3-662-47666-6_19

(III) *Is there a strategy to stay on budget with probability* ≥ 0.85*?*

MDP problems like (II) and (III) become even more challenging when each transition is equipped with both a cost and a *utility*. Such *cost-utility trade-offs* have recently been studied in [2].

The problems (II) and (III) may become easier in *Markov chains*, which have no non-determinism, i.e., there are no actions. Referring to the project example above, the activities may be completely planned out, but their effects (i.e. cost and next state) may still be probabilistic, yielding problems of the kind:

(IV) *Will the budget be kept with probability* ≥ 0.85*?*

Closely related to the aforementioned decision problems is the following optimisation problem, referred to as the *quantile query* in [2,19]. A quantile query asked by a funding body, for instance, could be the following:

(V) *Given a probability threshold* τ*, compute the smallest budget that suffices with probability at least* τ*.*

Non-stochastic problems like (I) are well understood. The purpose of this paper is to investigate the complexity of MDP problems such as (II) and (III), of Markov-chain problems such as (IV), and of quantile queries like (V). More formally, the models we consider are Markov chains and MDPs with non-negative integer costs, and the main focus of this paper is on the *cost problem* for those models: Given a budget constraint φ represented as a Boolean combination of linear inequalities and a probability threshold τ, we study the complexity of determining whether the probability of paths reaching a designated target state with cost consistent with φ is at least τ.

In order to highlight some issues, let us briefly discuss two approaches that do not, at least not in an obvious way, resolve the core challenges. First, one approach to answer the MDP problems could be to compute a strategy that minimises the *expected* total cost, which is a classical problem in the MDP literature, solvable in polynomial time using linear programming methods [13]. However, minimising the expectation may not be optimal: if you don't want to be late, it may be better to walk than to wait for the bus, even if the bus saves you time in average. The second approach with shortcomings is to phrase problems (II), (III) and (IV) as MDP or Markov-chain *reachability* problems, which are also known to be solvable in polynomial time. This, however, ignores the fact that numbers representing cost are commonly represented in their natural succinct *binary* encoding. Augmenting each state with possible accumulated costs leads to an exponential blow-up of the state space.

Our Contribution. The goal of this paper is to comprehensively investigate under which circumstances and to what extent the complexity of the cost problem and of quantile queries may be below EXP. We also significantly strengthen the NP lower bound derivable from [12]. We distinguish between acyclic and general control graphs. In short, we show the following for the cost problem: (1) for acyclic Markov chains it is PP-complete, (2) for general Markov chains it is hard for the PosSLP problem and in PSpace, (3) for acyclic MDPs it is PSpace-complete, and (4) for general MDPs it is EXP-complete. Due to space constraints, full details have been moved to a technical report [9].

Related Work. The motivation for this paper comes from the work on quantile queries in [2,19] mentioned above and on model checking so-called durational probabilistic systems [12] with a probabilistic timed extension of CTL. While the focus of [19] is mainly on "qualitative" problems where the probability threshold is either 0 or 1, an iterative linear-programming-based approach for solving quantile queries has been suggested in [2]. The authors report satisfying experimental results, the worst-case complexity however remains exponential time. Settling the complexity of quantile queries has been identified as one of the current challenges in the conclusion of [2].

Recently, there has been considerable interest in models of stochastic systems that extend weighted graphs or counter systems, see e.g. the survey [15]. Multi-dimensional percentile queries for various payoff functions are studied in [14]. The work by Bruyère et al. [6] has also been motivated by the fact that minimising the expected total cost is not always adequate. For instance, they consider the problem of computing a scheduler in an MDP with positive integer weights that ensures that both the expected and the maximum incurred cost remain below a given values. Other recent work also investigated MDPs with a single counter ranging over the non-negative integers, see e.g. [5]. However, in that work updates to the counter can be both positive and negative. For that reason, the analysis focuses on questions about reaching the counter value *zero*.

2 Preliminaries

We write $\mathbb{N} = \{0, 1, 2, \ldots\}$. For a countable set X we write $\mathit{dist}(X)$ for the set of *probability distributions* over X; i.e., $\mathit{dist}(X)$ consists of those functions $f : X \to [0, 1]$ such that $\sum_{x \in X} f(x) = 1$.

Markov Chains. A *Markov chain* is a triple $\mathcal{M} = (S, s_0, \delta)$, where S is a countable (finite or infinite) set of states, $s_0 \in S$ is an initial state, and $\delta : S \to \mathit{dist}(S)$ is a probabilistic transition function that maps a state to a probability distribution over the successor states. Given a Markov chain we also write $s \xrightarrow{p} t$ or $s \to t$ to indicate that $p = \delta(s)(t) > 0$. A *run* is an infinite sequence $s_0 s_1 \cdots \in \{s_0\}S^\omega$ with $s_i \to s_{i+1}$ for $i \in \mathbb{N}$. We write $\mathit{Run}(s_0 \cdots s_k)$ for the set of runs that start with $s_0 \cdots s_k$. To $\mathcal{M}$ we associate the standard probability space $(\mathit{Run}(s_0), \mathcal{F}, \mathcal{P})$ where $\mathcal{F}$ is the σ-field generated by all basic cylinders $\mathit{Run}(s_0 \cdots s_k)$ with $s_0 \cdots s_k \in \{s_0\}S^*$, and $\mathcal{P} : \mathcal{F} \to [0, 1]$ is the unique probability measure such that $\mathcal{P}(\mathit{Run}(s_0 \cdots s_k)) = \prod_{i=1}^{k} \delta(s_{i-1})(s_i)$.

Markov Decision Processes. A *Markov decision process (MDP)* is a tuple $\mathcal{D} = (S, s_0, A, \mathit{En}, \delta)$, where S is a countable set of states, $s_0 \in S$ is the initial state, A is a finite set of actions, $\mathit{En} : S \to 2^A \setminus \emptyset$ is an action enabledness function that assigns to each state s the set $\mathit{En}(s)$ of actions enabled in s, and

$\delta : S \times A \to \mathit{dist}(S)$ is a probabilistic transition function that maps a state s and an action $a \in \mathit{En}(s)$ enabled in s to a probability distribution over the successor states. A (deterministic, memoryless) *scheduler* for $\mathcal{D}$ is a function $\sigma : S \to A$ with $\sigma(s) \in \mathit{En}(s)$ for all $s \in S$. A scheduler σ induces a Markov chain $\mathcal{M}_\sigma = (S, s_0, \delta_\sigma)$ with $\delta_\sigma(s) = \delta(s, \sigma(s))$ for all $s \in S$. We write $\mathcal{P}_\sigma$ for the corresponding probability measure of $\mathcal{M}_\sigma$.

Cost Processes. A *cost process* is a tuple $\mathcal{C} = (Q, q_0, t, A, \mathit{En}, \Delta)$, where Q is a finite set of control states, $q_0 \in Q$ is the initial control state, t is the target control state, A is a finite set of actions, $\mathit{En} : Q \to 2^A \setminus \emptyset$ is an action enabledness function that assigns to each control state q the set $\mathit{En}(q)$ of actions enabled in q, and $\Delta : Q \times A \to \mathit{dist}(Q \times \mathbb{N})$ is a probabilistic transition function. Here, for $q, q' \in Q$, $a \in \mathit{En}(q)$ and $k \in \mathbb{N}$, the value $\Delta(q,a)(q',k) \in [0,1]$ is the probability that, if action a is taken in control state q, the cost process transitions to control state q' and cost k is incurred. For the complexity results we define the *size* of $\mathcal{C}$ as the size of a succinct description, i.e., the costs are encoded in binary, the probabilities are encoded as fractions of integers in binary (so the probabilities are rational), and for each $q \in Q$ and $a \in \mathit{En}(q)$, the distribution $\Delta(q,a)$ is described by the list of triples (q', k, p) with $\Delta(q,a)(q',k) = p > 0$ (so we assume this list to be finite). Consider the directed graph $G = (Q, E)$ with

$$E := \{(q, q') \in (Q \setminus \{t\}) \times Q : \exists a \in \mathit{En}(q)\ \exists k \in \mathbb{N}.\ \Delta(q,a)(q',k) > 0\}\ .$$

We call $\mathcal{C}$ *acyclic* if G is acyclic (which can be determined in linear time).

A cost process $\mathcal{C}$ induces an MDP $\mathcal{D}_\mathcal{C} = (Q \times \mathbb{N}, (q_0, 0), A, \mathit{En}', \delta)$ with $\mathit{En}'(q,c) = \mathit{En}(q)$ for all $q \in Q$ and $c \in \mathbb{N}$, and $\delta((q,c),a)(q',c') = \Delta(q,a)(q', c'-c)$ for all $q, q' \in Q$ and $c, c' \in \mathbb{N}$ and $a \in A$. For a state $(q,c) \in Q \times \mathbb{N}$ in $\mathcal{D}_\mathcal{C}$ we view q as the current control state and c as the current cost, i.e., the cost accumulated thus far. We refer to $\mathcal{C}$ as a *cost chain* if $|\mathit{En}(q)| = 1$ holds for all $q \in Q$. In this case one can view $\mathcal{D}_\mathcal{C}$ as the Markov chain induced by the unique scheduler of $\mathcal{D}_\mathcal{C}$. For cost chains, actions are not relevant, so we describe cost chains just by the tuple $\mathcal{C} = (Q, q_0, t, \Delta)$.

Recall that we restrict schedulers to be deterministic and memoryless, as such schedulers will be sufficient for the objectives in this paper. Note, however, that our definition allows schedulers to depend on the current cost, i.e., we may have schedulers σ with $\sigma(q,c) \neq \sigma(q,c')$.

The Accumulated Cost K. In this paper we will be interested in the cost accumulated during a run before reaching the target state t. For this cost to be a well-defined random variable, we make two assumptions on the system: (i) We assume that $\mathit{En}(t) = \{a\}$ holds for some $a \in A$ and $\Delta(t,a)(t,0) = 1$. Hence, runs that visit t will not leave t and accumulate only a finite cost. (ii) We assume that for all schedulers the target state t is almost surely reached, i.e., for all schedulers the probability of eventually visiting a state (t,c) with $c \in \mathbb{N}$ is equal to one. The latter condition can be verified by graph algorithms in time quadratic in the input size, e.g., by computing the *maximal end components* of the MDP obtained from $\mathcal{C}$ by ignoring the cost, see e.g. [3, Alg. 47].

Given a cost process $\mathcal{C}$ we define a random variable $K_{\mathcal{C}} : \mathit{Run}((q_0, 0)) \to \mathbb{N}$ such that $K_{\mathcal{C}}((q_0, 0)\ (q_1, c_1)\ \cdots) = c$ if there exists $i \in \mathbb{N}$ with $(q_i, c_i) = (t, c)$. We often drop the subscript from $K_{\mathcal{C}}$ if the cost process $\mathcal{C}$ is clear from the context. We view $K(w)$ as the accumulated cost of a run w.

From the above-mentioned assumptions on t, it follows that for any scheduler the random variable K is almost surely defined. Dropping assumption (i) would allow the same run to visit states (t, c_1) and (t, c_2) for two different $c_1, c_2 \in \mathbb{N}$. There would still be reasonable ways to define a cost K, but no apparently best way. If assumption (ii) were dropped, we would have to deal with runs that do not visit the target state t. In that case one could study the random variable K as above *conditioned* under the event that t is visited. For Markov chains, [4, Sec. 3] describes a transformation that preserves the distribution of the conditional cost K, but t is almost surely reached in the transformed Markov chain. In this sense, our assumption (ii) is without loss of generality for cost chains. For general cost processes the transformations of [4] do not work. In fact, a scheduler that "optimises" K conditioned under reaching t might try to avoid reaching t once the accumulated cost has grown unfavourably. Hence, dropping assumption (ii) in favour of conditional costs would give our problems an aspect of multi-objective optimisation, which is not the focus of this paper.

The Cost Problem. Let x be a fixed variable. An *atomic cost formula* is an inequality of the form $x \le B$ where $B \in \mathbb{N}$ is encoded in binary. A *cost formula* is an arbitrary Boolean combination of atomic cost formulas. A number $n \in \mathbb{N}$ *satisfies* a cost formula φ, in symbols $n \models \varphi$, if φ is true when x is replaced by n.

This paper mainly deals with the following decision problem: given a cost process $\mathcal{C}$, a cost formula φ, and a probability threshold $\tau \in [0, 1]$, the *cost problem* asks whether there exists a scheduler σ with $\mathcal{P}_\sigma(K_{\mathcal{C}} \models \varphi) \ge \tau$. The case of an atomic cost formula φ is an important special case. Clearly, for cost chains $\mathcal{C}$ the cost problem simply asks whether $\mathcal{P}(K_{\mathcal{C}} \models \varphi) \ge \tau$ holds. One can assume $\tau = 1/2$ without loss of generality, thanks to a simple construction, see [9]. Moreover, with an oracle for the cost problem at hand, one can use binary search over τ to approximate $\mathcal{P}_\sigma(K \models \varphi)$: i oracle queries suffice to approximate $\mathcal{P}_\sigma(K \models \varphi)$ within an absolute error of 2^{-i}.

By our definition, the MDP $\mathcal{D}_{\mathcal{C}}$ is in general infinite as there is no upper bound on the accumulated cost. However, when solving the cost problem, there is no need to keep track of costs above B, where B is the largest number appearing in φ. So one can solve the cost problem in so-called *pseudo-polynomial time* (i.e., polynomial in B, not in the size of the encoding of B) by computing an explicit representation of a restriction, say $\widehat{\mathcal{D}}_{\mathcal{C}}$, of $\mathcal{D}_{\mathcal{C}}$ to costs up to B, and then applying classical linear-programming techniques [13] to compute the optimal scheduler for the finite MDP $\widehat{\mathcal{D}}_{\mathcal{C}}$. Since we consider reachability objectives, the optimal scheduler is deterministic and memoryless. This shows that our restriction to deterministic memoryless schedulers is without loss of generality. In terms of our succinct representation we have:

Proposition 1. *The cost problem is in* EXP.

The subject of this paper is to investigate to what extent the EXP complexity is optimal.

3 Quantile Queries

In this section we consider the following function problem, referred to as *quantile query* in [2,19]. Given a cost chain $\mathcal{C}$ and a probability threshold τ, a quantile query asks for the smallest budget B such that $\mathcal{P}_\sigma(K_\mathcal{C} \leq B) \geq \tau$. We show that polynomially many oracle queries to the cost problem for atomic cost formulas "$x \leq B$" suffice to answer a quantile query. This can be done using binary search over the budget B. The following proposition provides a suitable general upper bound on this binary search, by exhibiting a concrete sufficient budget, computable in polynomial time:

Proposition 2. *Suppose* $0 \leq \tau < 1$. *Let* p_{min} *be the smallest non-zero probability and* k_{max} *be the largest cost in the description of the cost process. Then* $\mathcal{P}_\sigma(K \leq B) \geq \tau$ *holds for all schedulers* σ*, where*

$$B := k_{max} \cdot \left\lceil |Q| \cdot \left(-\ln(1-\tau)/p_{min}^{|Q|} \ + \ 1 \right) \right\rceil \ .$$

The case $\tau = 1$ is covered by [19, Thm. 6], where it is shown that one can compute in polynomial time the smallest B with $\mathcal{P}_\sigma(K \leq B) = 1$ for all schedulers σ, if such B exists. We conclude that quantile queries are polynomial-time inter-reducible with the cost problem for atomic cost formulas.

4 Cost Chains

In this section we consider the cost problems for acyclic and general cost chains. Even in the general case we obtain PSpace membership, avoiding the EXP upper bound from Prop. 1.

Acyclic Cost Chains. The complexity class PP [8] can be defined as the class of languages L that have a probabilistic polynomial-time bounded Turing machine M_L such that for all words x one has $x \in L$ if and only if M_L accepts x with probability at least 1/2. The class PP includes NP [8], and Toda's theorem states that P^{PP} contains the polynomial-time hierarchy [17]. We show that the cost problem for acyclic cost chains is PP-complete.

Theorem 3. *The cost problem for acyclic cost chains is in PP. It is PP-hard under polynomial-time Turing reductions, even for atomic cost formulas.*

Proof (sketch). To show membership in PP, we construct a probabilistic Turing machine that simulates the acyclic cost chain, and keeps track of the currently accumulated cost on the tape. For the lower bound, it follows from [12, Prop. 4] that an instance of the *K*TH LARGEST SUBSET problem can be reduced to a cost problem for acyclic cost chains with atomic cost formulas. This problem is PP-hard under polynomial-time Turing reductions [10, Thm. 3]. □

PP-hardness strengthens the NP-hardness result from [12] substantially: by Toda's theorem it follows that any problem in the polynomial-time hierarchy can be solved by a deterministic polynomial-time bounded Turing machine that has oracle access to the cost problem for acyclic cost chains.

General Cost Chains. For the PP upper bound in Thm. 3, the absence of cycles in the control graph seems essential. Indeed, we can use cycles to show hardness for the PosSLP problem, suggesting that the acyclic and the general case have different complexity. PosSLP is a fundamental problem for numerical computation [1]. Given an arithmetic circuit with operators $+$, $-$, $*$, inputs 0 and 1, and a designated output gate, the PosSLP problem asks whether the circuit outputs a positive integer. PosSLP is in PSpace; in fact, it lies in the 4th level of the *counting hierarchy (CH)* [1], an analogue to the polynomial-time hierarchy for classes like PP. We have the following theorem:

Theorem 4. *The cost problem for cost chains is in* PSpace *and hard for* PosSLP*.*

The remainder of this section is devoted to a proof sketch of this theorem. Showing membership in PSpace requires non-trivial results. There is no agreed-upon definition of probabilistic PSpace in the literature, but we can define it in analogy to PP as follows: *Probabilistic* PSpace is the class of languages L that have a probabilistic polynomial-space bounded Turing machine M_L such that for all words x one has $x \in L$ if and only if M_L accepts x with probability at least $1/2$. The cost problem for cost chains is in this class, as can be shown by adapting the argument from the beginning of the proof sketch for Thm. 3, replacing PP with probabilistic PSpace. It was first proved in [16] that probabilistic PSpace equals PSpace, hence the cost problem for cost chains is in PSpace.

For the PosSLP-hardness proof one can assume the following normal form, see the proof of [7, Thm.5.2]: there are only $+$ and $*$ operators, the corresponding gates alternate, and all gates except those on the bottom level have exactly two incoming edges, cf. the top of Fig. 1. We write $val(g)$ for the value output by gate g. Then PosSLP asks: given an arithmetic circuit (in normal form) including gates g_1, g_2, is $val(g_1) \geq val(g_2)$?

As an intermediate step of independent interest, we show PosSLP-hardness of a problem about deterministic finite automata (DFAs). Let Σ be a finite alphabet and call a function $f : \Sigma \to \mathbb{N}$ a *Parikh function.* The *Parikh image* of a word $w \in \Sigma^*$ is the Parikh function f such that $f(a)$ is the number of occurrences of a in w. We show:

Proposition 5. *Given an arithmetic circuit including gate g, one can compute in logarithmic space a Parikh function f (in binary encoding) and a DFA $\mathcal{A}$ such that $val(g)$ equals the number of accepting computations in $\mathcal{A}$ that are labelled with words that have Parikh image f.*

The construction is illustrated in Fig. 1. It is by induction on the levels of the arithmetic circuit. A gate labelled with "+" is simulated by *branching* into the inductively constructed gadgets corresponding to the gates this gate connects

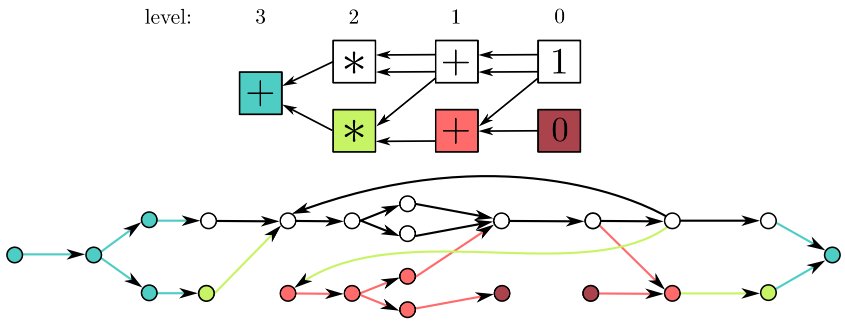

Fig. 1. Top: an arithmetic circuit in normal form. Bottom: a DFA (omitting input letters) corresponding to the construction of Prop. 5. Identical colours indicate a correspondence between gates and states.

to. Likewise, a gate labelled with "$*$" is simulated by *sequentially composing* the gadgets corresponding to the gates this gate connects to. It is the latter case that may introduce cycles in the structure of the DFA. Building on this construction, by encoding alphabet letters in natural numbers encoded in binary, we then show:

Proposition 6. *Given an arithmetic circuit including gate g on odd level ℓ, one can compute in logarithmic space a cost process $\mathcal{C}$ and $T \in \mathbb{N}$ with $\mathcal{P}(K_\mathcal{C} = T) = val(g)/m$, where $m = \exp_2(2^{(\ell-1)/2+1} - 1) \cdot \exp_d(2^{(\ell-1)/2+1} - 3)$.*

Towards the PosSLP lower bound from Thm. 4, given an arithmetic circuit including gates g_1, g_2, we use Prop. 6 to construct two cost chains $\mathcal{C}_1 = (Q, q_1, t, \Delta)$ and $\mathcal{C}_2 = (Q, q_2, t, \Delta)$ and $T_1, T_2 \in \mathbb{N}$ such that $\mathcal{P}(K_{\mathcal{C}_i} = T_i) = val(g_i)/m$ holds for $i \in \{1, 2\}$ and for $m \in \mathbb{N}$ as in Prop. 6. Then we compute a number $H \geq T_2$ such that $\mathcal{P}(K_{\mathcal{C}_2} > H) < 1/m$. The representation of m from Prop. 6 is of exponential size. However, using Prop. 2, H depends only logarithmically on $m + 1$. We combine $\mathcal{C}_1$ and $\mathcal{C}_2$ to a cost chain $\mathcal{C} = (Q \uplus \{q_0\}, q_0, t, \widetilde{\Delta})$, where $\widetilde{\Delta}$ extends Δ by $\widetilde{\Delta}(q_0)(q_1, H + 1) = 1/2$ and $\widetilde{\Delta}(q_0)(q_2, 0) = 1/2$. By this construction, the new cost chain $\mathcal{C}$ initially either incurs cost $H + 1$ and then emulates $\mathcal{C}_1$, or incurs cost 0 and then emulates $\mathcal{C}_2$. Those possibilities have probability 1/2 each.

Finally, we compute a suitable cost formula φ such that we have $val(g_1) \geq val(g_2)$ if and only if $\mathcal{P}(K_\mathcal{C} \models \varphi) \geq 1/2$, completing the logspace reduction. We remark that the structure of the formula φ, in particular the number of inequalities, is fixed. Only the involved numbers depend on the concrete instance.

5 Cost Processes

Acyclic Cost Processes. We now prove that the cost problem for acyclic cost processes is PSpace-complete. The challenging part is to show that PSpace-hardness even holds for *atomic* cost formulas. For our lower bound, we reduce from a generalisation of the classical SubsetSum problem: Given a tuple $(k_1, \ldots, k_n, T)$ of natural numbers with n even, the QSubsetSum *problem* asks whether the following formula is true:

$$\exists x_1 \in \{0,1\}\ \forall x_2 \in \{0,1\}\ \cdots\ \exists x_{n-1} \in \{0,1\}\ \forall x_n \in \{0,1\}\ :\ \sum_{1 \le i \le n} x_i k_i = T$$

Here, the quantifiers $\exists$ and $\forall$ occur in strict alternation. It is shown in [18, Lem. 4] that QSubsetSum is PSpace-complete. One can think of such a formula as a turn-based game, the QSubsetSum *game*, played between Player Odd and Player Even. If $i \in \{1, \ldots, n\}$ is odd (even), then turn i is Player Odd's (Player Even's) turn, respectively. In turn i the respective player decides to either *take* k_i by setting $x_i = 1$, or *not* to *take* k_i by setting $x_i = 0$. Player Odd's objective is to make the sum of the taken numbers equal T, and Player Even tries to prevent that. If Player Even is replaced by a random player, then Player Odd has a strategy to win with probability 1 if and only if the given instance is a "yes" instance for QSubsetSum. This gives a PSpace-hardness proof for the cost problem with non-atomic cost formulas $\varphi \equiv (x = T)$. In order to strengthen the lower bound to atomic cost formulas $\varphi \equiv (x \le B)$ we have to give Player Odd an incentive to take numbers k_i, although she is only interested in not exceeding the budget B. This challenge is addressed in our PSpace-hardness proof.

The PSpace-hardness result reflects the fact that the optimal strategy must take the current cost into account, not only the control state, even for atomic cost formulas. This may be somewhat counter-intuitive, as a good strategy should always "prefer small cost". But if there always existed a strategy depending *only* on the control state, one could guess this strategy in NP and invoke the PP-result of Sec. 4 in order to obtain an $\mathrm{NP}^{\mathrm{PP}}$ algorithm, implying $\mathrm{NP}^{\mathrm{PP}} = $ PSpace and hence a collapse of the counting hierarchy.

Indeed, for a concrete example, consider the acyclic cost process with $Q = \{q_0, q_1, t\}$, and $En(q_0) = \{a\}$ and $En(q_1) = \{a_1, a_2\}$, and $\Delta(q_0, a)(q_1, +1) = \frac{1}{2}$ and $\Delta(q_0, a)(q_1, +3) = \frac{1}{2}$ and $\Delta(q_1, a_1)(t, +3) = 1$ and $\Delta(q_1, a_2)(t, +6) = \frac{1}{2}$ and $\Delta(q_1, a_2)(t, +1) = \frac{1}{2}$. Consider the atomic cost formula $\varphi \equiv (x \le 5)$. An optimal scheduler σ plays a_1 in $(q_1, 1)$ and a_2 in $(q_1, 3)$, because additional cost 3, incurred by a_1, is fine in the former but not in the latter configuration. For this scheduler σ we have $\mathcal{P}_\sigma(K \models \varphi) = \frac{3}{4}$.

Theorem 7. *The cost problem for acyclic cost processes is in* PSpace. *It is* PSpace*-hard, even for atomic cost formulas.*

Proof (sketch). To prove membership in PSpace, we consider a procedure Opt that, given $(q, c) \in Q \times \mathbb{N}$ as input, computes the optimal (i.e., maximised over all schedulers) probability $p_{q,c}$ that starting from (q, c) one reaches (t, d) with $d \models \varphi$. The following procedure characterisation of $p_{q,c}$ for $q \ne t$ is crucial for $\mathrm{Opt}(q, c)$:

$$p_{q,c} = \max_{a \in En(q)} \sum_{q' \in Q} \sum_{k \in \mathbb{N}} \Delta(q, a)(q', k) \cdot p_{q', c+k}$$

So $\textsc{Opt}(q,c)$ loops over all $a \in En(q)$ and all $(q',k) \in Q \times \mathbb{N}$ with $\Delta(q,a)(q',k) > 0$ and recursively computes $p_{q',c+k}$. Since the cost process is acyclic, the height of the recursion stack is at most $|Q|$. The representation size of the probabilities that occur in that computation is polynomial. To see this, consider the product D of the denominators of the probabilities occurring in the description of Δ. The encoding size of D is polynomial. All probabilities occurring during the computation are integer multiples of $1/D$. Hence computing $\textsc{Opt}(q_0,0)$ and comparing the result with τ gives a PSpace procedure.

For the lower bound we reduce the QSubsetSum problem, defined above, to the cost problem for an atomic cost formula $x \le B$. Given an instance $(k_1,\ldots,k_n,T)$ with n is even of the QSubsetSum problem, we construct an acyclic cost process $\mathcal{C} = (Q, q_0, t, A, En, \Delta)$ as follows. We take $Q = \{q_0, q_2, \ldots, q_{n-2}, q_n, t\}$. Those control states reflect pairs of subsequent turns that the QSubsetSum game can be in. The transition rules Δ will be set up so that probably the control states $q_0, q_2, \ldots, q_n, t$ will be visited in that order, with the (improbable) possibility of shortcuts to t. For even i with $0 \le i \le n-2$ we set $En(q_i) = \{a_0, a_1\}$. These actions correspond to Player Odd's possible decisions of not taking, respectively taking k_{i+1}. Player Even's response is modelled by the random choice of not taking, respectively taking k_{i+2} (with probability 1/2 each). In the cost process, taking a number k_i corresponds to incurring cost k_i. We also add an additional cost ℓ in each transition.[1] Therefore we define our cost problem to have the atomic formula $x \le B$ with $B := (n/2)\cdot \ell + T$. For some large number $M \in \mathbb{N}$, formally defined in [9], we set for all even $i \le n-2$ and for $j \in \{0,1\}$:

$$\begin{aligned}
\Delta(q_i,a_j)(q_{i+2}, \ell + j\cdot k_{i+1}) &= (1/2)\cdot(1 - (\ell + j\cdot k_{i+1})/M)\\
\Delta(q_i,a_j)(t, \ell + j\cdot k_{i+1}) &= (1/2)\cdot(\ell + j\cdot k_{i+1})/M\\
\Delta(q_i,a_j)(q_{i+2}, \ell + j\cdot k_{i+1} + k_{i+2}) &= (1/2)\cdot(1 - (\ell + j\cdot k_{i+1} + k_{i+2})/M)\\
\Delta(q_i,a_j)(t, \ell + j\cdot k_{i+1} + k_{i+2}) &= (1/2)\cdot(\ell + j\cdot k_{i+1} + k_{i+2})/M
\end{aligned}$$

So with high probability the MDP transitions from q_i to q_{i+2}, and cost ℓ, $\ell + k_{i+1}$, $\ell + k_{i+2}$, $\ell + k_{i+1} + k_{i+2}$ is incurred, depending on the scheduler's (i.e., Player Odd's) actions and on the random (Player Even) outcome. But with a small probability, which is proportional to the incurred cost, the MDP transitions to t, which is a "win" for the scheduler as long as the accumulated cost is within budget B. We make sure that the scheduler loses if q_n is reached:

$$\Delta(q_n,a)(t,B+1) = 1 \qquad \text{with } En(q_n) = \{a\}$$

The MDP is designed so that the scheduler probably "loses" (i.e., exceeds the budget B); but whenever cost k is incurred, a winning opportunity with probability k/M arises. Since $1/M$ is small, the overall probability of winning is approximately C/M if total cost $C \le B$ is incurred. In order to maximise this

[1] This is for technical reasons. Roughly speaking, this prevents the possibility of reaching the full budget B before an action in control state q_{n-2} is played.

chance, the scheduler wants to maximise the total cost without exceeding B, so the optimal scheduler will target B as total cost.

The values for ℓ, M and τ need to be chosen carefully, as the overall probability of winning is not exactly the sum of the probabilities of the individual winning opportunities. By the "union bound", this sum is only an upper bound, and one needs to show that the sum approximates the real probability closely enough. $\square$

General Cost Processes. We show the following theorem:

Theorem 8. *The cost problem is* EXP*-complete.*

The EXP upper bound was stated in Prop. 1. The lower bound is based on a reduction from *countdown games* [11].

The Cost-Utility Problem. MDPs with *two* non-negative and non-decreasing integer counters, viewed as cost and utility, respectively, were considered e.g. in [2]. Specifically, those works consider problems such as computing the minimal cost C such that the probability of gaining at least a given utility U is at least τ. Possibly the most fundamental of those problems is the following: the *cost-utility problem* asks, given an MDP with both cost and utility, and numbers $C, U \in \mathbb{N}$, whether one can, with probability 1, gain utility at least U using cost at most C. Using essentially the proof of Thm. 8 we show:

Corollary 9. *The cost-utility problem is* EXP*-complete.*

The Universal Cost Problem. We defined the cost problem so that it asks whether *there exists* a scheduler σ with $\mathcal{P}_\sigma(K_\mathcal{C} \models \varphi) \geq \tau$. A variant is the *universal cost problem*, which asks whether $\mathcal{P}_\sigma(K_\mathcal{C} \models \varphi) \geq \tau$ holds *for all* schedulers σ. Here the scheduler is viewed as an adversary which tries to prevent the satisfaction of φ. For cost chains the cost problem and the universal cost problem are equivalent. Thms. 7 and 8 hold analogously in the universal case:

Theorem 10. *The universal cost problem for acyclic cost processes is in* PSPACE. *It is* PSPACE*-hard, even for atomic cost formulas. The universal cost problem is* EXP*-complete.*

6 Conclusions and Open Problems

In this paper we have studied the complexity of analysing succinctly represented stochastic systems with a single non-negative and only increasing integer counter. We have improved the known complexity bounds significantly. Among other results, we have shown that the cost problem for Markov chains is in PSPACE and both hard for PP and the PosSLP problem. Can one prove PSPACE-hardness or membership in the counting hierarchy?

Regarding acyclic and general MDPs, we have proved PSPACE-completeness and EXP-completeness, respectively. Our results leave open the possibility that the cost problem for atomic cost formulas is not EXP-hard and even in PSPACE. The technique described in the proof sketch of Thm. 7 cannot be applied to general cost processes, because there we have to deal with paths of exponential length, which, informally speaking, have double-exponentially small probabilities. Proving hardness in an analogous way would thus require probability thresholds τ of exponential representation size.

Acknowledgments. The authors would like to thank Andreas Göbel for valuable hints, Christel Baier and Sascha Klüppelholz for thoughtful feedback on an earlier version of this paper, and anonymous referees for their helpful comments.

References

1. Allender, E., Bürgisser, P., Kjeldgaard-Pedersen, J., Bro, P.: Miltersen. On the complexity of numerical analysis. SIAM J. Comput. **38**(5), 1987–2006 (2009)
2. Baier, C., Dubslaff, C., Klüppelholz, S.: Trade-off analysis meets probabilistic model checking. In: Proc. CSL-LICS, pp. 1:1–1:10. ACM (2014)
3. Baier, C., Katoen, J.-P.: Principles of Model Checking. MIT Press (2008)
4. Baier, C., Klein, J., Klüppelholz, S., Märcker, S.: Computing conditional probabilities in markovian models efficiently. In: Ábrahám, E., Havelund, K. (eds.) TACAS 2014 (ETAPS). LNCS, vol. 8413, pp. 515–530. Springer, Heidelberg (2014)
5. Brázdil, T., Brožek, V., Etessami, K., Kučera, A., Wojtczak, D.: One-counter Markov decision processes. In: Proc. SODA, pp. 863–874. SIAM (2010)
6. Bruyère, V., Filiot, E., Randour, M., Raskin, J.-F.: Meet your expectations with guarantees: beyond worst-case synthesis in quantitative games. In: Proc. STACS, LIPIcs, vol. 25, pp. 199–213 (2014)
7. Etessami, K., Yannakakis, M.: Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations. J. ACM **56**(1), 1:1–1:66 (2009)
8. Gill, J.: Computational complexity of probabilistic Turing machines. SIAM J. Comput. **6**(4), 675–695 (1977)
9. Haase, C., Kiefer, S.: The odds of staying on budget (2014). Technical Report at http://arxiv.org/abs/1409.8228
10. Haase, C., Kiefer, S.: The complexity of the Kth largest subset problem and related problems (2015). Technical Report at http://arxiv.org/abs/1501.06729
11. Jurdziński, M., Sproston, J., Laroussinie, F.: Model checking probabilistic timed automata with one or two clocks. Log. Meth. Comput. Sci. **4**(3), 12 (2008)
12. Laroussinie, F., Sproston, J.: Model checking durational probabilistic systems. In: Sassone, V. (ed.) FOSSACS 2005. LNCS, vol. 3441, pp. 140–154. Springer, Heidelberg (2005)
13. Puterman, M.L.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley and Sons (2008)
14. Randour, M., Raskin, J.-F., Sankur, O.: Percentile queries in multi-dimensional Markov decision processes. In: Proc. CAV, LNCS (2015)
15. 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)

16. Simon, J.: On the difference between one and many. In: Salomaa, A., Steinby, M. (eds.) ICALP 1977. LNCS, vol. 52, pp. 480–491. Springer, Heidelberg (1977)
17. Toda, S.: PP is as hard as the polynomial-time hierarchy. SIAM J. Comput. **20**(5), 865–877 (1991)
18. Travers, S.: The complexity of membership problems for circuits over sets of integers. Theor. Comput. Sci. **369**(1–3), 211–229 (2006)
19. Ummels, M., Baier, C.: Computing quantiles in Markov reward models. In: Pfenning, F. (ed.) FOSSACS 2013 (ETAPS 2013). LNCS, vol. 7794, pp. 353–368. Springer, Heidelberg (2013)