Energy-Utility Quantiles[⋆]
Extended Version (2015-10-26)

Christel Baier, Marcus Daum, Clemens Dubslaff,
Joachim Klein, Sascha Klüppelholz

Institute for Theoretical Computer Science
Technische Universität Dresden, Germany

Abstract. The concept of quantiles is well-known in statistics, but its benefits for the formal quantitative analysis of probabilistic systems have been noticed only recently. To compute quantiles in Markov decision processes where the objective is a probability constraint for an until (i.e., constrained reachability) property with an upper reward bound, an iterative linear-programming (LP) approach has been proposed in a recent paper. We consider here a more general class of quantiles with probability or expectation objectives, allowing to reason about the trade-off between costs in terms of energy and some utility measure. We show how the iterative LP approach can be adapted for these types of quantiles and propose another iterative approach that decomposes the LP to be solved into smaller ones. This algorithm has been implemented and evaluated in case studies for quantiles where the objective is a probability constraint for until properties with upper reward bounds.

1 Introduction

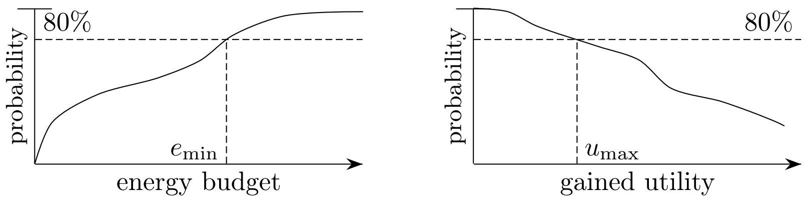
The concept of quantiles is well-known in statistics (see, e.g., [21]) and used there to reason about the cumulative distribution function of a random variable R. Quantiles are defined as maximal values r such that the probability for the event $R > r$ is beyond a given threshold. Although quantiles can provide very useful insights in the interplay of various cost functions and other system properties, they have barely obtained attention in the context of formal algorithmic system analysis. Quantiles for probabilistic operational models, such as Markov chains or Markov decision processes, can be defined using parameterized state properties $\Phi[r]$ or $\Psi[r]$, where r is a parameter for some cost or reward function and $\Phi[r]$ is increasing in r, whereas $\Psi[r]$ is decreasing in r. The notion "increasing" means that $s \models \Phi[r]$ implies $s \models \Phi[i]$ for all $i > r$ ("decreasing" has an analogous meaning). Quantiles for objectives $\Phi[r]$ and $\Psi[r]$ in state s of the given model are defined as $\min\{\, r : s \models \Phi[r] \,\}$ resp. $\max\{\, r : s \models \Psi[r] \,\}$. We formalize $\Phi[r]$ and $\Psi[r]$ by PRCTL-like constraints that assert lower or upper bounds either for the probabilities for reward-bounded path formulas or for the expected accumulated

⋆ This work was partly funded by the DFG through the CRC 912 HAEC, the cluster of excellence cfAED, the project QuaOS, and the DFG/NWO-project ROCKS and partially by Deutsche Telekom Stiftung, the ESF young researcher groups IMData 100098198 and SREX 100111037, and the EU-FP-7 grant 295261 (MEALS).

rewards until reaching a certain target. Typical examples are formulas of the form $\varPhi_u[e]$ for fixed u and $\varPsi_e[u]$ for fixed e asserting that the probability for

$$\lambda_{e,u} \;=\; \Diamond\big(\,(\mathit{energy} \leqslant e) \wedge (\mathit{utility} \geqslant u)\,\big)$$

is, e.g., at least 0.8. (We use LTL notations where the temporal operator $\Diamond$ stands for "eventually".) The quantile $e_{\min} = \min\{e \in \mathbb{N} : s \models \varPhi_u[e]\}$ is the minimal initial energy budget required to achieve the utility value u with probability at least 0.8, while $u_{\max} = \max\{u \in \mathbb{N} : s \models \varPsi_e[u]\}$ is the maximal utility that can be achieved with probability at least 0.8, when the energy budget is e. The curve for $\lambda_{e,u}$ on the left of the figure below illustrates how the probability increases when the utility value u is fixed and the energy budget e tends to ∞. The curve on the right shows how the probability for $\lambda_{e,u}$ decreases when the energy budget e is fixed and the demanded degree of utility tends to ∞.



State properties $\varPhi[r]$ or $\varPsi[r]$ can also impose a constraint on the expected value of a random variable. For example, one might ask for the minimal initial energy budget e that is needed to ensure that the expected degree of utility is at least some predefined utility threshold u. Vice versa, an expectation quantile might specify the maximal degree of utility that can be achieved when the expected energy consumption is required to be less or equal some fixed value e.

In probabilistic models with nondeterminism (e.g., for modeling concurrency by interleaving) such as Markov decision processes (MDPs), quantiles can be defined either in an existential or in a universal version, depending on whether the quantile is used in a worst-case analysis (where all possible resolutions of the nondeterminism are taken into account) or whether the task is to synthesize a control mechanism that schedules actions in an optimal way.

As the above examples suggest, quantiles can be seen as a concept to reason about the trade-off between different quantitative aspects, such as energy and utility. Thus, they yield an alternative to *multi-objective reasoning* for MDPs by means of Pareto optimal schedulers for multiple objectives given as Boolean combinations of constraints on the probabilities for certain events and/or expected accumulated costs [11,12]. The demand for algorithms to compute quantiles in Markovian models occurred to us during case studies with resource management protocols [3]. However, in various case studies with probabilistic model checkers carried out by other researchers, quantiles have been used implicitly in diagrams illustrating the evaluation results of the experimental studies.

Model-checking algorithms for various types of properties with *fixed* reward bounds have been proposed for discrete Markovian models and implemented in tools, see, e.g., [1,18,15]. The task to compute quantiles is, however, more challenging since it requires to compute an *optimal* reward bound for parameterized objectives. Our recent paper [4] briefly considers quantiles for discrete and continuous-time Markov chains, as an example for nonstandard multi-objective reasoning. To the best of our knowledge, [22] is the only paper where the computation of quantiles has been addressed for MDPs. It considers quantiles in MDPs with a nonnegative reward function for the states where the objective is a probability constraint for a reachability property with an upper reward bound r, formalized using the temporal reward-bounded until operator $\mathrm{U}^{\leqslant r}$. The above mentioned quantile $\min\{e : s \models \Phi_u[e]\}$ appears as a special case since $\Phi_u[e]$ can be seen as a probability constraint for the path property $\lambda_{e,u} = \Diamond^{\leqslant e}(\mathit{utility} \geqslant u) = \mathit{true}\, \mathrm{U}^{\leqslant e}(\mathit{utility} \geqslant u)$. In [22], polynomial-time algorithms for qualitative constraints where the probability bounds are 0 or 1 and an iterative linear-programming (LP) approach for probability bounds p with $0 < p < 1$ has been presented. The minimal or maximal probabilities for a path property $A\, \mathrm{U}^{\leqslant r}\, B$ for $r = 0, 1, 2, \ldots$ is calculated until the probability bound p is reached, where the extrema are taken over all resolutions of nondeterminism. This approach appears to be naïve, but the computation of quantiles is known to be computationally hard (at least NP-hard already for Markov chains by the results of [19]). This is reflected in the exponential upper bound in [22] for the number of iterations and the size of the LPs to be solved.

Contribution. First, we generalize the approach of [22] by introducing general notions of quantiles in MDPs where the objective can either be a probability constraint or a constraint on an expectation (Sec. 3). Second, we revisit the iterative LP approach suggested by [22] and discuss refinements that make the approach feasible in practice. The core idea is an iterative method that propagates intermediate results as much as possible and follows the dynamic-programming scheme with embedded LPs to deal with zero-reward cycles (Sec. 4.2). We implemented this approach into Prism [14] and study its performance by means of an energy-aware job-scheduling system (Sec. 6). Third, we present new algorithms for the computation of quantiles in MDPs where the objective is (a) either a probability constraint for reachability conditions with lower reward bounds (Sec. 4.3), or (b) a constraint on the expected accumulated reward (Sec. 5). These algorithms also rely on an iterative LP approach and the propagation principle is applicable as well (Sec. 4). Although we are not aware that expectation quantiles in MDPs have been addressed before, the presented algorithm for (b) shares some similarities with algorithms that have been proposed for stochastic shortest path problems [6] and to maximize/minimize the expected cost to reach a target [10].

2 Preliminaries

We provide a brief summary of the relevant concepts of MDPs and specifications given as formulas in probabilistic computation tree logic with reward-bounded modalities (PRCTL). Further details can be found, e.g., in [20,9,5].

Markov decision processes (MDPs). An MDP is a tuple $\mathcal{M} = (S, \mathit{Act}, P)$, where S is a finite set of states, Act a finite set of actions, $P : S \times \mathit{Act} \times S \to [0,1]$ such that $\sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$ for all states $s \in S$ and actions $\alpha \in \mathit{Act}$. The tuples $(s, \alpha, s') \in S \times \mathit{Act} \times S$ with $P(s, \alpha, s') > 0$ are called *steps* and we then say that state s' is an α-successor of s. We write $Act(s)$ for the set of actions α that have an α-successor from state $s \in S$ and require that $Act(s) \neq \varnothing$ for all states s. Intuitively, if the current state of $\mathcal{M}$ is s, then first there is a nondeterministic choice to select one of the enabled actions α. Then, $\mathcal{M}$ behaves probabilistically and moves with probability $P(s, \alpha, s')$ to some state s'. *Markov chains* are purely probabilistic instances of MDPs, i.e., where the action set is a singleton.

Paths in an MDP $\mathcal{M}$ can be seen as sample runs with resolved nondeterminism. Formally, paths are finite or infinite sequences $\pi = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2 \ldots \in (S\times Act)^*S \cup (S\times Act)^\omega$ that are built by consecutive steps, i.e., $\alpha_i \in Act(s_i)$ and $P(s_i, \alpha_i, s_{i+1}) > 0$ for all i. $\pi[k]$ denotes the $(k{+}1)$-st state in π and $pref(\pi, k)$ the prefix of π consisting of the first k steps, ending in state $\pi[k] = s_k$. We write $FPaths(s)$ for the set of finite paths and $IPaths(s)$ for the set of infinite paths starting in s.

Reward structure. A reward structure $\mathcal{R}$ for $\mathcal{M}$ consists of finitely many reward functions $rew : S \times Act \to \mathbb{N}$. If $\pi = s_0\,\alpha_0\,s_1\,\alpha_1 \ldots \alpha_{n-1}\,s_n$ is a finite path, then the accumulated reward $rew(\pi)$ is the sum of the rewards for the state-action pairs, i.e., $rew(\pi) = \sum_{0\leqslant i<n} rew(s_i, \alpha_i)$.

Schedulers and induced probability space. Reasoning about probabilities for path properties in MDPs requires the selection of an initial state and the resolution of the nondeterministic choices between the possible transitions. The latter is formalized via *schedulers*, often also called policies or adversaries, which take as input a finite path and select an action to be executed. A (deterministic) scheduler is a function $\mathfrak{S} : FPaths \to Act$ such that $\mathfrak{S}(\pi) \in Act\big(s_n\big)$ for all finite paths $\pi = s_0\,\alpha_0 \ldots \alpha_{n-1}\,s_n$. An $\mathfrak{S}$-*path* is any path that arises when the nondeterministic choices in $\mathcal{M}$ are resolved using $\mathfrak{S}$, i.e., $\mathfrak{S}\big(pref(\pi, k)\big) = \alpha_k$ for all $0 \leqslant k < n$. Infinite $\mathfrak{S}$-paths are defined accordingly. Given some scheduler $\mathfrak{S}$ and state s (viewed as the initial state), the behavior of $\mathcal{M}$ under $\mathfrak{S}$ is purely probabilistic and can be formalized by a tree-like (infinite-state) Markov chain $\mathcal{M}_s^{\mathfrak{S}}$. One can think of the states in $\mathcal{M}_s^{\mathfrak{S}}$ as finite $\mathfrak{S}$-paths $\pi = s_0\alpha_0 \ldots \alpha_{n-1}s_n$ starting in state s, where the probability to move from π to $\pi\,\alpha\,s'$ is simply $P(s_n, \alpha, s')$. Using standard concepts of measure and probability theory, a sigma-algebra and a probability measure $\mathsf{Pr}_s^{\mathfrak{S}}$ for measurable sets of the infinite paths in the Markov chain $\mathcal{M}_s^{\mathfrak{S}}$, also called *(path) events* or *path properties*, is defined and can be transferred to maximal $\mathfrak{S}$-paths in $\mathcal{M}$ starting in s. For further details, we refer to standard text books such as [13,16,20].

For a worst-case analysis of a system modeled by an MDP $\mathcal{M}$, one ranges over all initial states and all schedulers (i.e., all possible resolutions of the nondeterminism) and considers the minimal or maximal probabilities for φ. If φ represents a desired path property, then $\mathsf{Pr}_s^{\min}(\varphi) = \inf_{\mathfrak{S}} \mathsf{Pr}_s^{\mathfrak{S}}(\varphi)$ is the probability for $\mathcal{M}$ satisfying φ that can be guaranteed even for worst-case scenarios, i.e., when ranging over all schedulers. Similarly, if φ stands for a bad (undesired) path property, then $\mathsf{Pr}_s^{\max}(\varphi) = \sup_{\mathfrak{S}} \mathsf{Pr}_s^{\mathfrak{S}}(\varphi)$ is the least upper bound that can be guaranteed for the bad behaviors.

State and path properties. Let s be a state, $p \in [0, 1]$ a probability bound, $\bowtie \in \{<, \leqslant, \geqslant, >\}$ and φ a path property. We write $s \models \exists\, \mathrm{P}_{\bowtie p}(\varphi)$ if there exists a scheduler $\mathfrak{S}$ with $\mathsf{Pr}_s^{\mathfrak{S}}(\varphi) \bowtie p$. Similarly, $s \models \forall \mathrm{P}_{\bowtie p}(\varphi)$ if $\mathsf{Pr}_s^{\mathfrak{S}}(\varphi) \bowtie p$ for all schedulers $\mathfrak{S}$. Given a reward structure $\mathcal{R}$ with reward function *rew*, sets A, $B \subseteq S$, and $r \in \mathbb{N}$, then $A \,\mathrm{U}(\,(\mathit{rew} \bowtie r) \wedge B\,)$ stands for the set of infinite paths $\tilde{\pi}$ such that there is some $k \in \mathbb{N}$ with $\mathit{rew}(\,\mathit{pref}(\tilde{\pi}, k)\,) \bowtie r$ and $\tilde{\pi}[k] \in B$, $\tilde{\pi}[i] \in A$ for $0 \leqslant i < k$. If *rew* is clear from the context (e.g., if the reward structure $\mathcal{R}$ is a singleton), we briefly write $A \,\mathrm{U}^{\bowtie r}\, B$ rather than $A \,\mathrm{U}\big((\mathit{rew} \bowtie r) \wedge B\big)$. We often use the notation $\pi \models A \,\mathrm{U}^{\bowtie r}\, B$ instead of $\pi \in A \,\mathrm{U}^{\bowtie r}\, B$. As usual, we derive the release operator R by $A \,\mathrm{R}^{\bowtie r}\, B = \neg(\neg A \,\mathrm{U}^{\bowtie r}\, \neg B)$, where $\neg B$ denotes the complement of B. The temporal modalities $\Diamond$ (eventually) and $\Box$ (always) with or without reward-bounds are derived as usual, e.g., $\Diamond^{\bowtie r} B = \mathit{true} \,\mathrm{U}^{\bowtie r}\, B$ and $\Box^{\bowtie r} B = \neg \Diamond^{\bowtie r} \neg B$, where *true* stands for the full state space.

Reward-bounded path properties such as $\varphi[r] = A \,\mathrm{U}^{\leqslant r}\, B$ are called *increasing* as $\tilde{\pi} \models \varphi[r]$ implies $\tilde{\pi} \models \varphi[r{+}1]$. The dual path properties $\psi[r] = \neg\varphi[r]$ are called *decreasing* as $\tilde{\pi} \models \psi[r{+}1]$ implies $\tilde{\pi} \models \psi[r]$. Analogously, a state property $\Phi[r]$ is called increasing if $s \models \Phi[r]$ implies $s \models \Phi[r{+}1]$. Examples for increasing state properties are $\exists \mathrm{P}_{>p}(\varphi[r])$, $\forall \mathrm{P}_{>p}(\varphi[r])$, $\exists \mathrm{P}_{<p}(\psi[r])$ and $\forall \mathrm{P}_{<p}(\psi[r])$. Decreasing state properties are defined accordingly.

Sub-MDPs, end components. We use the notion *sub-MDP* of $\mathcal{M}$ for any pair $(T, \mathfrak{A})$ where $T \subseteq S$ and $\mathfrak{A} : T \to 2^{\mathit{Act}}$ such that for all $t \in T$: (1) $\mathfrak{A}(t) \subseteq \mathit{Act}(t)$ and (2) if $\alpha \in \mathfrak{A}(t)$ and $P(t, \alpha, t') > 0$ then $t' \in T$. An *end component* of $\mathcal{M}$ is a sub-MDP $(T, \mathfrak{A})$ of $\mathcal{M}$ where $\mathfrak{A}(t)$ is nonempty for all $t \in T$ and the underlying directed graph with node set T and the edge relation $t \to t'$ iff $P(t, \alpha, t') > 0$ for some $\alpha \in \mathfrak{A}(t)$ is strongly connected. An end component is said to be *maximal* if it is not contained in any other end component.

3 Quantiles

As stated in the introduction, quantiles in MDPs can be defined for arbitrary objectives given by increasing or decreasing parameterized state properties. We now provide general definitions for quantiles in MDPs where the state properties impose either a probability or an expectation constraint, and identify the instances for which we present algorithms in the next two sections.

Quantiles for probability objectives. Let $\mathcal{M} = (S, \mathit{Act}, P)$ be an MDP as in Sec. 2 and $\mathit{rew} : S \times \mathit{Act} \to \mathbb{N}$ a distinguished reward function in its reward structure. Given an increasing path property $\varphi[r]$ where parameter $r \in \mathbb{N}$ stands for some bound on the accumulated reward, we define the following types of *existential quantiles*, where $\psi[r] = \neg\varphi[r]$, $\trianglerighteq \in \{\geqslant, >\}$ and $p \in [0,1] \cap \mathbb{Q}$:

$$\begin{aligned}
\mathsf{Qu}_s\big(\,\exists \mathrm{P}_{\trianglerighteq p}(\varphi[?])\,\big) \;&=\; \min\,\big\{\, r \in \mathbb{N} \,:\, s \models \exists \mathrm{P}_{\trianglerighteq p}\big(\varphi[r]\big)\,\big\} \\
&=\; \min\,\big\{\, r \in \mathbb{N} \,:\, \mathsf{Pr}_s^{\max}\big(\varphi[r]\big) \trianglerighteq p\,\big\} \\
\mathsf{Qu}_s\big(\,\exists \mathrm{P}_{\trianglerighteq p}(\psi[?])\big) \;&=\; \max\,\big\{\, r \in \mathbb{N} \,:\, s \models \exists \mathrm{P}_{\trianglerighteq p}\big(\psi[r]\big)\,\big\} \\
&=\; \max\,\big\{\, r \in \mathbb{N} \,:\, \mathsf{Pr}_s^{\max}\big(\psi[r]\big) \trianglerighteq p\,\big\}
\end{aligned}$$

Similarly, we can define the corresponding types of *universal quantiles*:

$$\begin{aligned}\mathsf{Qu}_s\big(\forall\,\mathrm{P}_{\trianglerighteq p}(\varphi[?])\big) &= \min\big\{\,r\in\mathbb{N} : \mathsf{Pr}_s^{\min}\big(\varphi[r]\big)\trianglerighteq p\,\big\}\\ \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{\trianglerighteq p}(\psi[?])\big) &= \max\big\{\,r\in\mathbb{N} : \mathsf{Pr}_s^{\min}\big(\psi[r]\big)\trianglerighteq p\,\big\}\end{aligned}$$

From each of these quantiles we can derive three more quantiles by applying duality arguments, e.g., $\mathsf{Pr}_s^{\max}(\varphi[r]) = 1-\mathsf{Pr}_s^{\min}(\psi[r])$, and the fact that $\min\{r\in\mathbb{N} : s\models\Phi[r]\}$ equals $\max\{r\in\mathbb{N} : s\not\models\Phi[r-1]\}$ when $\Phi[r]$ is an increasing state property. For example:

$$\begin{aligned}\min\big\{r\in\mathbb{N} : \mathsf{Pr}_s^{\max}\big(\varphi[r]\big)>p\,\big\} &= \min\big\{\,r\in\mathbb{N} : \mathsf{Pr}_s^{\min}\big(\psi[r]\big)<1-p\,\big\}\\ &= \max\big\{\,r\in\mathbb{N} : \mathsf{Pr}_s^{\min}\big(\psi[r-1]\big)\geqslant 1-p\,\big\}\\ &= \max\big\{\,r\in\mathbb{N} : \mathsf{Pr}_s^{\max}\big(\varphi[r-1]\big)\leqslant p\,\big\}\end{aligned}$$

This observation yields groups of four quantiles that are derivable from each other. See [2] for the list of quantile dualities. For the above example we have:

$$\begin{aligned}\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{>p}(\varphi[?])\big) &= \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{<1-p}(\psi[?])\big)\\ &= \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{\geqslant 1-p}(\psi[?])\big)+1 \;=\; \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{\leqslant p}(\varphi[?])\big)+1\end{aligned}$$

The quantiles studied in [22] are obtained by considering $\varphi[r] = A\,\mathrm{U}^{\leqslant r}\,B$ and $\psi[r] = (\neg A)\,\mathrm{R}^{\leqslant r}(\neg B)$. Additionally, we address until-properties with lower reward bounds, i.e., $\varphi[r] = A\,\mathrm{U}^{\geqslant r}\,B$ and $\psi[r] = (\neg A)\,\mathrm{R}^{\geqslant r}(\neg B)$. To investigate the interplay of two reward functions (such as one for energy and one for utility) we also address path formulas where instead of the sets A and B, constraints for some other reward function are imposed. For instance:

$$\lambda_{e,u} \;=\; \Diamond\big(\,(\mathit{energy}\leqslant e)\wedge(\mathit{utility}\geqslant u)\,\big),$$

where $e,u\in\mathbb{N}$ and *energy* and *utility* stand for the accumulated reward along finite paths of reward functions $\mathit{erew} : S\times \mathit{Act}\to\mathbb{N}$ (for the energy) and $\mathit{urew} : S\times\mathit{Act}\to\mathbb{N}$ (for the utility). For an infinite path $\tilde{\pi}$, we have $\tilde{\pi}\models\lambda_{e,u}$ iff $\tilde{\pi}$ has a finite prefix π with $\mathit{erew}(\pi)\leqslant e$ and $\mathit{urew}(\pi)\geqslant u$. Likewise, $\lambda_{e,u}$ can be interpreted as an instance of an until-property with an upper or a lower reward bound. For fixed utility threshold u, the path property $\varphi[e] = \lambda_{e,u} = \Diamond^{\leqslant e}(\mathit{utility}\geqslant u)$ is increasing, while $\psi[u] = \lambda_{e,u} = \Diamond^{\geqslant u}(\mathit{energy}\leqslant e)$ is decreasing for fixed energy budget e. The task to compute the existential quantiles

$$\begin{aligned}\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{>p}(\lambda_{?,u})\big) &= \min\;\big\{\;e\in\mathbb{N} : \mathsf{Pr}_s^{\max}(\lambda_{e,u})>p\;\big\}\\ \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{>p}(\lambda_{e,?})\big) &= \max\;\big\{\;u\in\mathbb{N} : \mathsf{Pr}_s^{\max}(\lambda_{e,u})>p\;\big\}\end{aligned}$$

corresponds to the problem of constructing a scheduler that minimizes the energy ensuring that the achieved utility is at least u with probability $>p$ or to maximize the achieved degree of utility for a given energy budget e. Analogously, universal quantiles provide the corresponding information on the energy-utility characteristics in worst-case scenarios.

Quantiles for expectation objectives. We also consider quantiles where the objective is the minimal or maximal expected value of a random variable

$f[r] : \mathit{IPaths} \to \mathbb{N} \cup \{\infty\}$. For instance, if $f[r]$ is increasing in r and θ some rational threshold, then an expectation quantile can be defined as the least $r \in \mathbb{N}$ such that the expected value of $f[r]$ is larger than θ for all or some scheduler(s). As an example for quantiles with expectation objectives, we consider a Boolean condition *cond* for finite paths and the random variable $f[e] = \mathit{utility}|_{\mathit{cond}} : \mathit{IPaths} \to \mathbb{N} \cup \{\infty\}$ that returns the utility value that is earned along finite paths where *cond* holds. Formally:

$$\mathit{utility}|_{\mathit{cond}}(\tilde{\pi}) \;=\; \sup\big\{\, \mathit{urew}\big(\,\mathit{pref}(\tilde{\pi},k)\big) \;:\; k \in \mathbb{N},\, \mathit{pref}(\tilde{\pi},k) \models \mathit{cond} \,\big\}$$

That is, if $\tilde{\pi}$ is an infinite path with $\tilde{\pi} \models \Diamond \mathit{cond}$ (i.e., $\mathit{pref}(\tilde{\pi},k) \models \mathit{cond}$ for some $k \in \mathbb{N}$) then $\mathit{utility}|_{\mathit{cond}}(\tilde{\pi}) = \mathit{urew}(\pi)$, where π is the longest prefix of $\tilde{\pi}$ with $\pi \models \mathit{cond}$. If $\tilde{\pi} \models \Box \mathit{cond}$ (i.e., $\mathit{pref}(\tilde{\pi},k) \models \mathit{cond}$ for all $k \in \mathbb{N}$) then $\mathit{utility}|_{\mathit{cond}}(\tilde{\pi})$ can be finite or infinite, depending on whether there are infinitely many positions i with $\mathit{urew}(s_i,\alpha_i) > 0$. Given a scheduler $\mathfrak{S}$ and a state s in $\mathcal{M}$, the *expected utility* for condition *cond* is the expected value of the random variable $\mathit{utility}|_{\mathit{cond}}$ under the probability measure induced by $\mathfrak{S}$ and s:

$$\mathsf{ExpUtil}_s^{\mathfrak{S}}\big(\,\mathit{cond}\,\big) \;\;=\;\; \sum_{r\in\mathbb{N}} r \cdot \mathsf{Pr}_s^{\mathfrak{S}}\big\{\, \tilde{\pi} \in \mathit{IPaths} \;:\; \mathit{utility}|_{\mathit{cond}}(\tilde{\pi}) = r \,\big\}$$

Note that $\mathsf{ExpUtil}_s^{\mathfrak{S}}\big(\mathit{cond}\big) = \infty$ is possible if $\mathsf{Pr}_s^{\mathfrak{S}}\big(\,\Diamond\Box(\mathit{cond})\,\big) > 0$. We define

$$\mathsf{ExpUtil}_s^{\max}\big(\,\mathit{cond}\,\big) \;=\; \sup_{\mathfrak{S}} \mathsf{ExpUtil}_s^{\mathfrak{S}}\big(\,\mathit{cond}\,\big).$$

$\mathsf{ExpUtil}_s^{\min}(\mathit{cond})$ is defined accordingly, taking the infimum over all schedulers rather than the supremum. Expectation energy-utility quantiles can be formalized by dealing with conditions $\mathit{cond}[e]$ that are parameterized by some energy value $e \in \mathbb{N}$. Examples are the following quantiles that fix a lower bound u for the extremal expected degree of utility and ask to minimize the required energy:

$$\mathsf{Qu}_s\big(\,\exists\,\mathrm{ExpU}_{>u}(\,\mathit{energy} \leqslant ?\,)\,\big) \;=\; \min\big\{\, e \in \mathbb{N} \;:\; \mathsf{ExpUtil}_s^{\max}\big(\,\mathit{energy} \leqslant e\,\big) > u \,\big\}$$

$$\mathsf{Qu}_s\big(\,\forall\,\mathrm{ExpU}_{>u}(\,\mathit{energy} \leqslant ?\,)\,\big) \;=\; \min\big\{\, e \in \mathbb{N} \;:\; \mathsf{ExpUtil}_s^{\min}\big(\,\mathit{energy} \leqslant e\,\big) > u \,\big\}$$

where $\pi \models (\mathit{energy} \leqslant e)$ iff $\mathit{erew}(\pi) \leqslant e$. Analogous definitions can be provided for quantiles that ask to maximize the achieved utility if an upper bound e for the expected consumed energy is given.

4 Computing probability quantiles

We now present algorithms for the computation of the quantitative quantiles introduced in Sec. 3. We start in this section with quantiles where the objective is a constraint on the extremal probability for a reward-bounded until formula. As stated before, quantiles that refer to reward-bounded release formulas are dual and can be computed using the same techniques.

Recently, a linear-programming (LP) approach for computing quantiles for (constrained) reachability properties with upper reward bounds (briefly called

$$\begin{array}{ll}
\multicolumn{2}{l}{\text{minimize} \sum\limits_{(s,i)\in S[r]} x_{s,i} \text{ where } S[r] = S \times \{0,1,\ldots,r\}, \text{ subject to}} \\
x_{s,i} = 0 & \text{if } s \not\models \exists(A \,\mathrm{U}\, B) \text{ and } 0 \leqslant i \leqslant r \\
x_{s,i} = 1 & \text{if } s \in B \text{ and } 0 \leqslant i \leqslant r \\
x_{s,i} \geqslant \sum\limits_{t\in S} P(s,\alpha,t)\cdot x_{t,i-rew(s,\alpha)} & \text{if } s \notin B,\ s \models \exists(A \,\mathrm{U}\, B) \text{ and } \alpha \in Act(s) \\
 & \text{such that } rew(s,\alpha) \leqslant i \leqslant r
\end{array}$$

Fig. 1. Linear program $\mathbb{LP}_r$ with the unique solution $p_{s,i} = \mathsf{Pr}_s^{\max}\big(A \,\mathrm{U}^{\leqslant i}\, B\big)$

$\mathrm{U}^{\leqslant ?}$-quantiles) in MDPs with state rewards has been suggested [22]. We first recall this approach for quantitative $\mathrm{U}^{\leqslant ?}$-quantiles (Sec. 4.1) and then provide an efficient computation scheme that relies on an iterative back-propagation procedure including several heuristics (Sec. 4.2). In Sec. 4.3, we briefly show how to adapt these methods for reachability properties with lower reward bounds.

4.1 Iterative linear-programming based approach

We recall the approach of [22], focusing on existential $\mathrm{U}^{\leqslant ?}$-quantiles with strict probability bounds. Other $\mathrm{U}^{\leqslant ?}$-quantiles can be treated similarly (see [22]).

The idea for computing $\mathsf{Qu}_s(\,\exists\, \mathrm{P}_{>p}(A \,\mathrm{U}^{\leqslant ?}\, B)\,)$ is to first apply standard methods for computing the maximum probability $p_s = \mathsf{Pr}_s^{\max}(A \,\mathrm{U}\, B)$ for the unbounded until formula $A \,\mathrm{U}\, B$. If p_s does not meet the probability bound p, i.e., $p_s \leqslant p$, the quantile is infinite for state s. For $p_s > p$, the idea of [22] is to compute the maximal probabilities $p_{s,r} = \mathsf{Pr}_s^{\max}(A \,\mathrm{U}^{\leqslant r}\, B)$ for increasing reward bound r, until $p_{s,r} > p$. For this purpose, [22] provides an LP with variables $x_{s,i}$ for $(s,i) \in S[r] = S \times \{0,1,\ldots,r\}$ and the unique solution $(p_{s,i})_{(s,i)\in S[r]}$, where $p_{s,i} = \mathsf{Pr}_s^{\max}(\,A \,\mathrm{U}^{\leqslant i}\, B\,)$. Fig. 1 shows the LP of [22], adapted for the case of state-action rewards (rather than state rewards). This LP-based computation scheme can be solved in exponential time, as shown in [22] by establishing an upper bound r_{max} for the smallest (finite) quantile. A naïve approach thus could first compute r_{max}, generate the LP with variables $x_{s,i}$ for $(s,i) \in S[r_{\mathsf{max}}]$ and then use general-purpose linear- or dynamic-programming techniques to solve the constructed LP (e.g., the Simplex algorithm, ellipsoid methods or value or policy iteration). However, since the upper bound r_{max} is exponential in the size of $\mathcal{M}$ and depends on the number of states in $\mathcal{M}$, the transition probabilities and rewards in $\mathcal{M}$ and the probability bound p, this approach turns out to be intractable when $\mathcal{M}$ or the reward values are large.

4.2 Back-propagation approach

The main bottleneck of the LP approach for computing quantitative quantiles is the possibly exponential size of the LP. We propose an iterative approach that computes the values $p_{s,i} = \mathsf{Pr}_s^{\max}(A \,\mathrm{U}^{\leqslant i}\, B)$ successively for $i = 0,1,2,\ldots$ by decomposing the LP in Fig. 1 into smaller ones and propagating already computed values as much as possible. Due to the reuse of already computed values, we call this approach *back-propagation (BP) approach*.

Given that the solution $(p_{s,j})_{0 \leqslant j < i}$ for $\mathbb{LP}_{i-1}$ is known when considering $\mathbb{LP}_i$, the constraints for variable $x_{s,i}$ in the third case of Fig. 1 (i.e., if $s \notin B$, $s \models \exists(A \,\mathrm{U}\, B)$ and $\alpha \in Act(s)$) can be rewritten as follows:

$$x_{s,i} \geqslant c_{s,i} \overset{\text{def}}{=} \max \Big\{ \sum_{t \in S} P(s,\alpha,t) \cdot p_{t,i-rew(s,\alpha)} \;:\; \alpha \in Act(s), rew(s,\alpha) > 0 \Big\}$$
$$x_{s,i} \geqslant \textstyle\sum_{t \in S} P(s,\alpha,t) \cdot x_{t,i} \quad \text{ if } rew(s,\alpha) = 0$$

We can now use standard methods to solve $\mathbb{LP}'_i$ with variables $(x_{s,i})_{s \in S}$ consisting of the above linear constraints together with the terminal cases $x_{s,i} = 0$ if $s \not\models \exists(A \,\mathrm{U}\, B)$ and $x_{s,i} = 1$ if $s \in B$, where the objective is to "minimize $\sum_{s \in S} x_{s,i}$". $\mathbb{LP}'_i$ has indeed a unique solution which agrees with the (unique) solution $(p_{s,i})_{s \in S}$ of $\mathbb{LP}_i$ for the variables $x_{s,i}$.

Suppose the task is to compute $q_s = \mathsf{Qu}_s(\,\exists\, \mathrm{P}_{>p}(A \,\mathrm{U}^{\leqslant ?}\, B)\,)$ for all states s. Let $n = |S|$, $m = \sum_{s \in S} |Act(s)|$ and z be the number of state-action pairs (s,α) for which $s \in S$, $\alpha \in Act(s)$ and $rew(s,\alpha) = 0$. Then, with the proposed back-propagation approach, $(q_s)_{s \in S}$ is obtained by first computing $\mathsf{Pr}^{\max}_s(A \,\mathrm{U}\, B)$ for all states s (which can be done in time polynomial in the size of $\mathcal{M}$ [7,5] and serves to identify the states $s \in S$ where $q_s = \infty$) and then solving the LPs $\mathbb{LP}'_0, \mathbb{LP}'_1, \ldots, \mathbb{LP}'_r$ (where $r \in \max\{q_s : \mathsf{Pr}^{\max}_s(A \,\mathrm{U}\, B) > p\}$) with n variables and $z + |S|$ linear constraints each.

Reward window. To reduce the memory requirements, we can use the observation that the constants $c_{s,i}$ in $\mathbb{LP}'_i$ are obtained from the values $p_{t,i-rew(s,\alpha)}$ where $\alpha \in Act(s)$ and $rew(s,\alpha) > 0$. As a consequence, the solution $\big(p_{t,i}\big)_{t \in S}$ for $\mathbb{LP}'_i$ can be discarded as soon as $\mathbb{LP}'_{i+w}$ has been solved for the maximal reward value $w = \max\big\{ rew(s,\alpha) \;:\; s \in S, \alpha \in Act(s) \big\}$ in $\mathcal{M}$. A further improvement considers the maximum reward of all incoming transitions per state. That is, the value of $p_{t,i}$ is not needed any more as soon as $\mathbb{LP}'_{i+w}$ has been solved where w equals the maximal reward of the state-action pairs (s,α) with $P(s,\alpha,t) > 0$.

Linear programs for zero-reward sub-MDP. The back-propagation approach can yield a major speed-up compared to the naïve approach with a single LP. However, if the number of state-action pairs with zero reward is large compared to the full set of actions in S, $\mathbb{LP}'_i$ needs still to be solved for several i. The idea then is to decompose $\mathbb{LP}'_i$ and treat the sub-LPs in a specific order. Let $\mathcal{G}$ be the directed graph with node set S and the edge relation $\rightarrow \subseteq S \times S$ given by $s \rightarrow t$ iff $P(s,\alpha,t) > 0$ for some action $\alpha \in Act(s)$ with $rew(s,\alpha) = 0$. Applying standard graph algorithms, we compute the strongly connected components in $\mathcal{G}$ and a topological sorting $C_1, \ldots, C_k$ for them. Then the SCCs $C_1, \ldots, C_k$ are the finest partition of S such that: if $s \in C_h$, $t \in C_j$, $P(s,\alpha,t) > 0$ and $rew(s,\alpha) = 0$, then $h \leqslant j$. Thus, we can decompose $\mathbb{LP}'_i$ into LPs $\mathbb{LP}'_{i,1}, \ldots, \mathbb{LP}'_{i,k}$, where $\mathbb{LP}'_{i,h}$ consists of the linear constraints $x_{s,i} \geqslant c_{s,i}$ and

$$x_{s,i} \;\;\geqslant\;\; \sum_{t \in C_h} P(s,\alpha,t) \cdot x_{t,i} \;+\; \sum_{u \in C_{>h}} P(s,\alpha,u) \cdot p_{u,i}$$

for $s \in C_h$, $\alpha \in Act(s)$, $rew(s,\alpha) = 0$. Here, $C_{>h} = C_{h+1} \cup \ldots \cup C_k$ and $(p_{u,i})_{u \in C_j}$ denotes the solutions of $\mathbb{LP}'_{i,j}$. The objective of $\mathbb{LP}'_{i,h}$ is to minimize the sum $\sum_{s \in C_h} x_{s,i}$.

Assuming that the sub-MDP $\mathcal{M}|_{rew=0}$ of $\mathcal{M}$ resulting by removing all actions α from $Act(s)$ with $rew(s,\alpha) > 0$ is acyclic, no LP has to be solved within our approach. In this case, the sets $C_1, \ldots, C_k$ are singletons, say $C_h = \{s_h\}$, and the solution $\big(p_{s,i}\big)_{s \in S}$ is obtained directly when processing the states in reversed topological order $s_k, s_{k-1}, \ldots, s_1$.

Other improvements. Several other heuristics can be integrated to speed up the computation time or to decrease the memory requirements. For instance, zero-reward self-loops can be removed by a quantile-preserving transformation $\mathcal{M} \rightsquigarrow \mathcal{M}'$. The MDP $\mathcal{M}'$ has the same state space S as $\mathcal{M}$ and the same rewards for all state-action pairs. The transition probability function P' of $\mathcal{M}'$ is given by $P'(s,\alpha,t) = P(s,\alpha,t)/(1-P(s,\alpha,s))$ if $rew(s,\alpha) = 0$, $t \neq s$ and $0 < P(s,\alpha,s) < 1$ and $P'(s,\alpha,t) = P(s,\alpha,t)$ in all other cases (see [2]). Another heuristic, which is however not yet realized in our implementation, is the aggregation method proposed in [8]. This approach permits to collapse all states belonging to the same maximal end components in the sub-MDP $\mathcal{M}|_{rew=0}$ into a single state.

4.3 Lower reward bounds

The approach for computing $\mathrm{U}^{\leqslant ?}$-quantiles can be adapted to compute quantiles for (constrained) reachability formulas with lower reward bounds, i.e., $A\,\mathrm{U}^{\geqslant ?}\,B$. For simplicity, we sketch only the treatment of reachability ($\Diamond^{\geqslant ?}B$) with a lower reward bound. More details and proofs can be found in [2]. We start with the universal quantile:

$$\mathsf{Qu}_s\big(\forall\,\mathrm{P}_{<p}(\Diamond^{\geqslant ?}B)\big) \;=\; \min\big\{\, r \in \mathbb{N} \,:\, \mathsf{Pr}^{\max}_s\big(\Diamond^{\geqslant r}B\big) < p \,\big\}$$

Clearly, if $\mathsf{Pr}^{\max}_s(\Diamond B) < p$ then the quantile for state s is 0. Furthermore:

$$\mathsf{Qu}_s\big(\forall\,\mathrm{P}_{<p}(\Diamond^{\geqslant ?}B)\big) = \infty \quad \text{iff} \quad \mathsf{Pr}^{\max}_s\big(\Diamond(C \wedge \Diamond B)\big) \geqslant p,$$

where C consists of all states t that are contained in a maximal end component $(T, \mathfrak{A})$ with $rew(t',\alpha) > 0$ for some state $t' \in T$ and an action $\alpha \in \mathfrak{A}(t')$. Intuitively, when entering C one can stay in C until the accumulated reward is greater or equal than r, before entering B. Otherwise, we apply the same idea as before and compute the values $p_{s,r} = \mathsf{Pr}^{\max}_s(\Diamond^{\geqslant r}B)$ for increasing r until $p_{s,r} < p$. The values $p_{s,r}$ are obtained as the unique solution of the following LP with variables $x_{s,i}$ for $(s,i) \in S[r]$ and the following constraints for $s \in S$ and $1 \leqslant i \leqslant r$:

$$\begin{aligned} x_{s,0} &= \mathsf{Pr}^{\max}_s\big(\Diamond B\big) \\ x_{s,i} &\geqslant 0 \\ x_{s,i} &\geqslant \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,\ell} \quad \text{if } \alpha \in Act(s) \text{ and } \ell = \max\{0, i - rew(s,\alpha)\} \end{aligned}$$

The objective is to minimize $\sum_{(s,i) \in S[r]} x_{s,i}$. To speed up the computation, one can add the following constraints: $x_{s,i} = 1$ if $\mathsf{Pr}^{\max}_s\big(\Diamond(C \wedge \Diamond B)\big) = 1$ for $s \in S$.

The existential quantile

$$\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{<p}(\Diamond^{\geqslant ?}B)\big) \;=\; \min\big\{\, r\in\mathbb{N} \,:\, \mathsf{Pr}_s^{\min}\big(\Diamond^{\geqslant r}B\big) < p \big\}$$

can then be computed by an analogous approach, using the fact that the values $p_{s,r} = \mathsf{Pr}_s^{\min}\big(\Diamond^{\geqslant r}B\big)$ are the greatest solutions in $[0,1]$ of the linear constraints

$$\begin{array}{lcll} x_{s,0} & = & \mathsf{Pr}_s^{\min}\big(\Diamond B\big) & \\ x_{s,i} & = & 0 & \text{if } i\geqslant 1,\ \mathsf{Pr}_s^{\min}(\Diamond B)=0 \text{ or } \mathsf{Pr}_s^{\min}(\Diamond posR)=0 \\ x_{s,i} & \leqslant & \sum\limits_{t\in S} P(s,\alpha,t)\cdot x_{t,\ell} & \text{if } i\geqslant 1,\ \mathsf{Pr}_s^{\min}(\Diamond B)>0 \text{ and } \mathsf{Pr}_s^{\min}(\Diamond posR)>0, \\ & & & \alpha\in Act(s) \text{ and } \ell=\max\{0, i-rew(s,\alpha)\} \end{array}$$

where $posR \subseteq S\times Act$ is the set of state-action pairs (s,α) with $rew(s,\alpha)>0$. Then, $\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{<p}(\Diamond^{\geqslant ?}B)\big) = \infty$ iff $\mathsf{Pr}_s^{\min}\big(\Box\Diamond B \wedge \Box\Diamond posR\big) \geqslant p$. Again, one could add the following constraints: $x_{s,i}=1$ if $\mathsf{Pr}_s^{\min}(\Box\Diamond B \wedge \Box\Diamond posR)=1$ for $s\in S$. Obviously, the back-propagation approach (cf. Sec. 4.2) is applicable for the existential and universal quantiles with lower bounds as well.

4.4 Energy-utility quantiles

The energy-utility quantile $\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{>p}(\lambda_{?,u})\big)$ as introduced in Sec. 3 can be computed using the same techniques as explained for quantiles of the form $\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{>p}(\Diamond^{\leqslant ?}B)\big)$. For this purpose, we might use an automaton $\mathcal{U}_u$ with states $q_0, q_1, \ldots, q_{u-1}, q_u$ representing the accumulated utility value. The goal state q_u represents that the achieved utility is at least u. The transitions of $\mathcal{U}_u$ are given by $q_i \to q_j$ for $j\geqslant i$. We put $\mathcal{M}$ and $\mathcal{U}_u$ in parallel to obtain an MDP $\mathcal{M}\otimes\mathcal{U}_u$ with a single reward function for the energy and synchronous transitions that capture the meaning of $\mathcal{U}_u$'s states. Formally, $\mathcal{M}\otimes\mathcal{U}_u = (S\times\{q_0,\ldots,q_u\}, Act, P')$ where

$$P'(\langle s,q_i\rangle,\alpha,\langle t,q_j\rangle) \;=\; P(s,\alpha,t) \quad \text{if } j=\min\{u, i+urew(s,\alpha)\}$$

and $P'(\cdot)=0$ in all other cases. The reward structure of $\mathcal{M}\otimes\mathcal{U}_u$ consists of the energy reward function *erew* lifted to the product. That is, we deal with the reward function $erew'$ for $\mathcal{M}\otimes\mathcal{U}_u$ given by $erew'(\langle s,q_i\rangle,\alpha) = erew(s,\alpha)$ for all $s\in S$, $0\leqslant i\leqslant u$ and $\alpha\in Act$. With $B=S\times\{q_u\}$, we then have

$$\mathsf{Pr}^{\max}_{\mathcal{M},s}\big(\Diamond(\,(energy\leqslant e)\wedge(utility\geqslant u)\,)\big) \;=\; \mathsf{Pr}^{\max}_{\mathcal{M}\otimes\mathcal{U}_u,\langle s,q_0\rangle}\big(\Diamond^{\leqslant e}B\big)$$

and therefore $\mathsf{Qu}_s^{\mathcal{M}}(\,\exists\,\mathrm{P}_{>p}(\lambda_{?,u})\,) = \mathsf{Qu}^{\mathcal{M}\otimes\mathcal{U}_u}_{\langle s,q_0\rangle}(\,\exists\,\mathrm{P}_{>p}(\,\Diamond^{\leqslant ?}B\,)\,)$.

The quantile $\mathsf{Qu}_s(\,\exists\,\mathrm{P}_{>p}(\lambda_{e,?})\,)$ is computable by an analogous automata-based approach, but now using the LP approach suggested for lower reward bounds (Sec. 4.3). Various other energy-utility quantiles can be computed using reductions to the case of reward-bounded until formulas or derived path properties. It is obvious that an analogous automata-based approach is applicable for quantiles where the objective is a probability constraint on path properties of the form $\Diamond((rew \bowtie r)\wedge\kappa)$, where κ is a Boolean combination of constraints of the form $rew_i \bowtie_i r_i$ for multiple reward functions $rew_1,\ldots,rew_k$ (other than rew).

5 Computing expectation quantiles

We now discuss how to compute the expectation quantiles in MDPs with two reward functions *erew* and *urew* for modeling the energy requirements and the achieved utility (see Sec. 3). Let us exemplify the approach computing

$$E_s^{\exists} \;=\; \mathsf{Qu}_s\big(\,\exists\,\mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\,\big) \text{ and } E_s^{\forall} \;=\; \mathsf{Qu}_s\big(\,\forall\,\mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\,\big).$$

Using known results for standard MDPs, we obtain that $\mathsf{ExpUtil}_s^{\max}(\mathit{energy} \leqslant e)$ is finite, provided that $\mathsf{Pr}_s^{\min}(\,\Diamond(\mathit{energy} > e)\,) = 1$. If, however, $\mathcal{M}$ contains end components where all the state-action pairs have zero energy reward then $\mathsf{Pr}_s^{\min}(\,\Diamond(\mathit{energy} > e)\,) < 1$ and $\mathsf{ExpUtil}_s^{\max}(\mathit{energy} \leqslant e) = \infty$ is possible.

Let us first make the simplifying assumption that all end components are both energy- and utility-divergent, i.e., whenever $(T, \mathfrak{A})$ is an end component of $\mathcal{M}$ then there exist state-action pairs (t, α) and (v, β) with $t, v \in T$ and $\alpha \in \mathfrak{A}(t)$, $\beta \in \mathfrak{A}(v)$ such that $\mathit{erew}(t, \alpha)$ and $\mathit{urew}(v, \beta)$ are positive. This assumption yields that $\mathsf{Pr}_s^{\min}(\,\Diamond(\mathit{energy} > e)\,) = 1$ and hence, $\mathsf{ExpUtil}_s^{\max}(\,\mathit{energy} \leqslant e\,)$ and $\mathsf{ExpUtil}_s^{\min}(\,\mathit{energy} \leqslant e\,)$ are finite for all states $s \in S$ and all energy bounds $e \in \mathbb{N}$. Moreover, $\lim_{e \to \infty} \mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{energy} \leqslant e) = \infty$ for each scheduler $\mathfrak{S}$. This yields the finiteness of the expectation quantiles $E_s^{\exists}$ and $E_s^{\forall}$. The computation of $E_s^{\exists}$ and $E_s^{\forall}$ can be carried out using an iterative approach as for probability quantiles. For $E_s^{\exists}$, we compute iteratively the values $u_{s,e} = \mathsf{ExpUtil}_s^{\min}(\mathit{energy} \leqslant e)$ until $u_{s,e} > u$, in which case $E_s^{\exists} = e$. It remains to explain how to compute $u_{s,e}$. Again, we can use an LP-based approach and characterize the vector $(u_{s,i})_{(s,i) \in S[e]}$ as the unique solution of the LP with variables $x_{s,i}$ for $(s,i) \in S[e] = S \times \{0, 1, \ldots, e\}$ and the objective to maximize the sum of the $x_{s,i}$'s subject to:

$$x_{s,i} \;\leqslant\; \mathit{urew}(s,\alpha) + \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,i-\mathit{erew}(s,\alpha)}$$

if $\alpha \in \mathit{Act}(s)$ and $\mathit{erew}(s,\alpha) \leqslant i \leqslant e$. For computing $E_s^{\forall}$, the values $v_{s,e} = \mathsf{ExpUtil}_s^{\max}(\mathit{energy} \leqslant e)$ can be computed by a similar schema, using the fact that the vector $(v_{s,i})_{(s,i) \in S[e]}$ is the least solution in $[0,1]^{S[e]}$ of the linear constraints

$$x_{s,i} \;\geqslant\; \mathit{urew}(s,\alpha) + \sum_{t \in S} P(s,\alpha,t) \cdot x_{t,i-\mathit{erew}(s,\alpha)}$$

if $\alpha \in \mathit{Act}(s)$ and $\mathit{erew}(s,\alpha) \leqslant i \leqslant e$. Obviously, the back-propagation approach is applicable as well.

The computation of expectation quantiles for the general case, where no assumptions on the end components are imposed, are detailed in [2]. Basically, this computation relies on an analogous LP approach, but requires a preprocessing step to identify the states where $\mathsf{ExpUtil}_s^{\max}(\mathit{energy} \leqslant e) = \infty$, respectively $\mathsf{ExpUtil}_s^{\min}(\mathit{energy} \leqslant e) = \infty$ and computing those states where the quantile is infinite. The main feature for this preprocessing is an analysis of end components, similar as in [10,12].

6 Implementation and case studies

In this section, we deal with our implementation of the algorithms for computing $U^{\leqslant ?}$-quantiles presented in Sec. 4.1 and 4.2 and demonstrate its usability within case studies. Our implementation relies on the computation of extremal probabilities for upper *reward-bounded* until properties on top of the explicit engine of the prominent probabilistic model checker Prism version 4.1 [14], which have not yet been supported within Prism so far. We compute quantiles either by solving the LP of [22] (see Fig. 1) directly using the LP-solver lpsolve[1] or with our back-propagation approach (BP). Our first case study is taken from Prism's benchmark suite [17], showing the applicability of our implementation on relatively small models and compare the performance of the LP and BP approach. Then, we turn to computing energy-utility quantiles for an energy-aware job-scheduling protocol. All calculations were carried out on a computer with two Intel E5-2680 8-core CPUs at 2.70 GHz with 384GB of RAM. More detailed information and further case studies can be found in [2].

Self-stabilization. The self-stabilizing protocol by Israeli and Jalfon is modeled[2] as an MDP for N equal processes organized in a ring, each having a token at the beginning and aiming to randomly send and receive tokens until the ring is in a stable state, i.e., only one process has a token. We used our quantile algorithms to compute the minimal number of steps required for reaching a stable state with probability of at least p for some schedulers (existential quantile) or all schedulers (universal quantile). The latter problem also has been answered in

Table 1. Results for randomized self-stabilizing (existential and universal quantile)

		model		existential quantile			universal quantile		
N	p	states	build	result	LP	BP	result	LP	BP
10	0.1	1,023	0.24s	18	118.38s	0.03s	26	403.36s	0.16s
	0.5	"	"	38	1,066.64s	0.05s	43	1,388.15s	0.09s
	0.99	"	"	117	11,552.55s	0.14s	130	19,794.61s	0.15s
15	0.1	32,767	1.56s	42	timeout	1.85s	61	timeout	3.78s
	0.5	"	"	89	timeout	3.85s	100	timeout	4.10s
	0.99	"	"	270	timeout	11.42s	305	timeout	12.18s

the referred Prism case study, but by iteratively increasing the step bound until the probability bound p was met. Our approach is more elegant by implicitly computing the probability values and answering only one (quantile) query. Table 1 shows our results for the LP and BP approach, with a timeout of 12 hours. The time for BP covers the entire computation of the quantile value r. For LP, we report the time for solving the linear program $\mathbb{LP}_r$. As it can be seen, the LP approach turns out to be infeasible already for relatively small models, whereas the BP implementation performs well. Table 1 also reveals that especially within

[1] `http://lpsolve.sourceforge.net`, we used version 5.5.2, presolving deactivated
[2] `http://www.prismmodelchecker.org/casestudies/self-stabilisation.php#ij`

Table 2. Results for energy-aware job scheduling (quantiles $e_{\min}$ and $u_{\max}$)

		model		quantile $e_{\min}$				model		quantile $u_{\max}$	
N	p	states	build	result	time	N	p	states	build	result	time
4	0.1	368,521	14.67s	179	37.43s	4	0.1	872,410	14.47s	7	173.71s
	0.5	"	"	198	37.02s		0.5	"	"	7	173.22s
	0.99	"	"	225	42.69s		0.99	"	"	7	155.66s
5	0.1	6,079,533	377.95s	242	1,058.48s	5	0.1	3,049,471	65.69s	9	812.19s
	0.5	"	"	266	1,135.65s		0.5	"	"	9	812.93s
	0.99	"	"	301	1,261.89s		0.99	"	"	9	736.93s

the LP approach the time spent for evaluating the quantile increases significantly when the probability bound p is high (and hence, also the quantile value is high).

Energy-aware job scheduling. We now turn to an energy-aware job-scheduling protocol modeled as an MDP, for which we compute energy-utility quantiles. Assume a system of N processes which need to enter a critical section in order to perform tasks, each within a given deadline. Access to the critical section is exclusively granted by a scheduler, which selects processes only if they have requested to enter. When a process states such a request, a deadline counter is set and decreased over time even if the process did not enter the critical section yet. Since computing a task also requires a certain amount of time in the critical section, deadlines can be exceeded. Utility is hence provided in terms of tasks finished without exceeding their deadline. Each process consumes energy, especially if it is in the critical section, and the global energy consumption equals the sum of energy consumed by all processes. Additional dependencies between utility and energy arise as the scheduler can activate a turbo mode for the critical section, doubling the computation speed but tripling energy consumption. As motivated in the introduction, we are now interested in the following energy-utility quantiles, both illustrating the trade-off between energy and utility w.r.t. several probability bounds p. We consider the quantile for the minimal energy $e_{\min}$ required to guarantee u successfully finished tasks, and the quantile for the maximal number $u_{\max}$ of tasks successfully finished by one process requiring not more than e energy. Our experiments solving these quantiles used the BP implementation with parameters $u{=}N$, $e{=}50{\cdot}N$. The results shown in Table 2 illustrate that even for large model sizes with millions of states, our implementation of the BP algorithm is feasible. As expected, none of the quantile computations for $e_{\min}$ and $u_{\max}$ finished within 12 hours when we used the LP approach instead of our BP implementation.

7 Conclusion

We introduced a general notion of (energy-utility) quantiles for MDPs and extended the LP schema from [22] to compute quantitative quantiles with lower and upper reward bounds, where the objective can be a probability constraint or a constraint on an expectation. We implemented a BP approach for quantitative quantiles with upper reward bounds, which can significantly speed up quantile computations, and demonstrated its performance by means of case studies.

References

1. S. Andova, H. Hermanns, and J.-P. Katoen. Discrete-time rewards model-checked. In *FORMATS'03*, volume 2791 of *LNCS*, pages 88–104. Springer, 2003.
2. C. Baier, M. Daum, C. Dubslaff, J. Klein, and S. Klüppelholz. Energy-Utility Quantiles. Technical report, TU Dresden, `http://wwwtcs.inf.tu-dresden.de/ALGI/PUB/NFM14/`, 2014.
3. C. Baier, M. Daum, B. Engel, H. Härtig, J. Klein, S. Klüppelholz, S. Märcker, H. Tews, and M. Völp. Waiting for locks: How long does it usually take? In *FMICS'12*, volume 7437 of *LNCS*, pages 47–62. Springer, 2012.
4. C. Baier, C. Dubslaff, J. Klein, S. Klüppelholz, and S. Wunderlich. Probabilistic Model Checking for Energy-Utility Analysis. *Festschrift*, 2014 (to appear).
5. C. Baier and J.-P. Katoen. *Principles of Model Checking*. MIT Press, 2008.
6. D. Bertsekas and J. Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3):580–595, 1991.
7. A. Bianco and L. de Alfaro. Model checking of probabilistic and non-deterministic systems. In *FSTTCS'95*, volume 1026 of *LNCS*, pages 499–513. Springer, 1995.
8. F. Ciesinski, C. Baier, M. Größer, and J. Klein. Reduction techniques for model checking Markov decision processes. In *QEST'08*, pages 45–54. IEEE, 2008.
9. L. de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, Stanford University, Department of Computer Science, 1997.
10. L. de Alfaro. Computing minimum and maximum reachability times in probabilistic systems. In *CONCUR'99*, volume 1664 of *LNCS*, pages 66–81, 1999.
11. K. Etessami, M. Kwiatkowska, M. Vardi, and M. Yannakakis. Multi-objective model checking of Markov decision processes. *Logical Methods in Computer Science*, 4(4), 2008.
12. V. Forejt, M. Kwiatkowska, G. Norman, and D. Parker. Automated verification techniques for probabilistic systems. In *SFM'11*, volume 6659 of *LNCS*, pages 53–113. Springer, 2011.
13. B. Haverkort. *Performance of Computer Communication Systems: A Model-Based Approach*. Wiley, 1998.
14. A. Hinton, M. Kwiatkowska, G. Norman, and D. Parker. PRISM: A tool for automatic verification of probabilistic systems. In *TACAS'06*, volume 3920 of *LNCS*, pages 441–444. Springer, 2006.
15. J.-P. Katoen, I. Zapreev, E. Hahn, H. Hermanns, and D. Jansen. The ins and outs of the probabilistic model checker MRMC. *Perform. Eval.*, 68(2), 2011.
16. V. Kulkarni. *Modeling and Analysis of Stochastic Systems*. Chapman & Hall, 1995.
17. M. Kwiatkowska, G. Norman, and D. Parker. The PRISM benchmark suite. In *QEST'12*. IEEE Computer Society, 2012.
18. Y. Kwon and G. Agha. A Markov reward model for software reliability. In *IPDPS'07*, pages 1–6. IEEE, 2007.
19. F. Laroussinie and J. Sproston. Model checking durational probabilistic systems. In *FOSSACS'05*, volume 3441 of *LNCS*, pages 140–154. Springer, 2005.
20. M. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley, 1994.
21. R. J. Serfling. *Approximation Theorems of Mathematical Statistics*. Wiley, 1980.
22. M. Ummels and C. Baier. Computing quantiles in Markov reward models. In *FOSSACS'13*, volume 7794 of *LNCS*, pages 353–368. Springer, 2013.

Appendix

In this appendix, we provide details and additional information that had to be omitted from the main part of this paper due to lack of space. Sec. A provides a list of the quantile dualities. Sec. B contains additional, technical details and proofs for the computation of quantiles, i.e., for quantiles with lower reward bounds (Sec. B.1), expectation quantiles (Sec. B.2), and soundness of the zero-reward self-loop removal (Sec. B.3). Sec. C then provides further information on our implementation and case studies.

A Quantile dualities

We list dualities for probability quantiles. However, similar dualities can also be obtained for expectation quantiles. Here, we distinguish between quantiles with respect to increasing and decreasing state properties.

Concerning increasing state properties:

$$\begin{aligned}
\mathsf{Qu}_s\big(\exists\mathrm{P}_{>p}(A\ \mathrm{U}^{\leqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\exists\mathrm{P}_{<1-p}((\neg A)\ \mathrm{R}^{\leqslant ?}\,(\neg B))\big)\\
&= \mathsf{Qu}_s\big(\forall\mathrm{P}_{\geqslant 1-p}((\neg A)\ \mathrm{R}^{\leqslant ?}\,(\neg B))\big)+1\\
&= \mathsf{Qu}_s\big(\forall\mathrm{P}_{\leqslant p}(A\ \mathrm{U}^{\leqslant ?}\ B)\big)+1
\end{aligned}$$

$$\begin{aligned}
\mathsf{Qu}_s\big(\exists\mathrm{P}_{>p}(A\ \mathrm{R}^{\geqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\exists\mathrm{P}_{<1-p}((\neg A)\ \mathrm{U}^{\geqslant ?}\,(\neg B))\big)\\
&= \mathsf{Qu}_s\big(\forall\mathrm{P}_{\geqslant 1-p}((\neg A)\ \mathrm{U}^{\geqslant ?}\,(\neg B))\big)+1\\
&= \mathsf{Qu}_s\big(\forall\mathrm{P}_{\leqslant p}(A\ \mathrm{R}^{\geqslant ?}\ B)\big)+1
\end{aligned}$$

$$\begin{aligned}
\mathsf{Qu}_s\big(\forall\mathrm{P}_{>p}(A\ \mathrm{U}^{\leqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\forall\mathrm{P}_{<1-p}((\neg A)\ \mathrm{R}^{\leqslant ?}\,(\neg B))\big)\\
&= \mathsf{Qu}_s\big(\exists\mathrm{P}_{\geqslant 1-p}((\neg A)\ \mathrm{R}^{\leqslant ?}\,(\neg B))\big)+1\\
&= \mathsf{Qu}_s\big(\exists\mathrm{P}_{\leqslant p}(A\ \mathrm{U}^{\leqslant ?}\ B)\big)+1
\end{aligned}$$

$$\begin{aligned}
\mathsf{Qu}_s\big(\forall\mathrm{P}_{>p}(A\ \mathrm{R}^{\geqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\forall\mathrm{P}_{<1-p}((\neg A)\ \mathrm{U}^{\geqslant ?}\,(\neg B))\big)\\
&= \mathsf{Qu}_s\big(\exists\mathrm{P}_{\geqslant 1-p}((\neg A)\ \mathrm{U}^{\geqslant ?}\,(\neg B))\big)+1\\
&= \mathsf{Qu}_s\big(\exists\mathrm{P}_{\leqslant p}(A\ \mathrm{R}^{\geqslant ?}\ B)\big)+1
\end{aligned}$$

Concerning decreasing state properties:

$$\begin{aligned}
\mathsf{Qu}_s\big(\exists\mathrm{P}_{>p}(A\ \mathrm{U}^{\geqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\exists\mathrm{P}_{<1-p}((\neg A)\ \mathrm{R}^{\geqslant ?}\,(\neg B))\big)\\
&= \mathsf{Qu}_s\big(\forall\mathrm{P}_{\geqslant 1-p}((\neg A)\ \mathrm{R}^{\geqslant ?}\,(\neg B))\big)-1\\
&= \mathsf{Qu}_s\big(\forall\mathrm{P}_{\leqslant p}(A\ \mathrm{U}^{\geqslant ?}\ B)\big)-1
\end{aligned}$$

$$\begin{aligned}
\mathsf{Qu}_s\big(\exists\,\mathrm{P}_{>p}(A\ \mathrm{R}^{\leqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{<1-p}(\,(\neg A)\ \mathrm{U}^{\leqslant ?}\,(\neg B\,)\,)\big)\\
&= \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{\geqslant 1-p}(\,(\neg A)\ \mathrm{U}^{\leqslant ?}\,(\neg B\,)\,)\big) - 1\\
&= \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{\leqslant p}(A\ \mathrm{R}^{\leqslant ?}\ B)\big) - 1
\end{aligned}$$

$$\begin{aligned}
\mathsf{Qu}_s\big(\forall\,\mathrm{P}_{>p}(A\ \mathrm{U}^{\geqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{<1-p}(\,(\neg A)\ \mathrm{R}^{\geqslant ?}\,(\neg B\,)\,)\big)\\
&= \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{\geqslant 1-p}(\,(\neg A)\ \mathrm{R}^{\geqslant ?}\,(\neg B\,)\,)\big) - 1\\
&= \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{\leqslant p}(A\ \mathrm{U}^{\geqslant ?}\ B)\big) - 1
\end{aligned}$$

$$\begin{aligned}
\mathsf{Qu}_s\big(\forall\,\mathrm{P}_{>p}(A\ \mathrm{R}^{\leqslant ?}\ B)\big) &= \mathsf{Qu}_s\big(\forall\,\mathrm{P}_{<1-p}(\,(\neg A)\ \mathrm{U}^{\leqslant ?}\,(\neg B\,)\,)\big)\\
&= \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{\geqslant 1-p}(\,(\neg A)\ \mathrm{U}^{\leqslant ?}\,(\neg B\,)\,)\big) - 1\\
&= \mathsf{Qu}_s\big(\exists\,\mathrm{P}_{\leqslant p}(A\ \mathrm{R}^{\leqslant ?}\ B)\big) - 1
\end{aligned}$$

B Technical details on the computation of quantiles

Before we state technical proofs for statements about computing lower-bound and expectation quantiles, we first introduce some more notions for schedulers.

Finite-memory and memoryless schedulers. Let $\pi = s_0\alpha_0\ldots\alpha_{n-1}s_n$ be a finite path. Then $\mathit{last}(\pi)$ denotes the state s_n reached by π. A scheduler $\mathfrak{S}$ is called *finite-memory* scheduler if there exists a finite set $\mathfrak{M}$ of modes, a decision function $\mathfrak{Dec} : S \times \mathfrak{M} \to \mathit{Act}$, an initial mode function $\mathsf{init} : S \to \mathfrak{M}$, and a next-mode function $\mathfrak{next} : \mathfrak{M} \times S \to \mathfrak{M}$ such that for each $\mathfrak{S}$-path π we have $\mathfrak{S}(\pi) = \mathfrak{Dec}\big(\mathit{last}(\pi), \mathfrak{m}(\pi)\big)$, where $\mathfrak{m}(\pi)$ is defined inductively on the length of finite paths by $\mathfrak{m}(s_0) = \mathsf{init}(s_0)$ and $\mathfrak{m}(\pi\,\alpha\,s) = \mathfrak{next}\big(\mathfrak{m}(\pi), s\,\big)$. The size of a finite-memory scheduler is the number of modes.

The notion of a *memoryless* scheduler is used for finite-memory schedulers with a single mode.

Optimal and adversarial schedulers. Considering an existential quantile, say $R = \mathsf{Qu}_s(\exists\,\mathrm{P}_{>p}(\varphi[?]))$, scheduler $\mathfrak{S}$ is said to be *optimal* if $\mathsf{Pr}^{\mathfrak{S}}_s(\varphi[R]) > p$. Scheduler $\mathfrak{S}$ is called *adversarial* for the universal quantile $R = \mathsf{Qu}_s(\forall\,\mathrm{P}_{>p}(\varphi[?]))$ if $\mathsf{Pr}^{\mathfrak{S}}_s(\varphi[R{-}1]) \leqslant p$. The definition of optimal and adversarial schedulers for other quantiles is analogous.

A simple consequence of the representation of $\mathsf{Pr}^{\max}_s\big(\,A\,\mathrm{U}^{\leqslant r}\,B\,\big)$ as the unique solution of the linear program in Fig. 1 is the existence of a finite-memory scheduler $\mathfrak{S}$ with the modes $0, 1, \ldots, r$ that maximizes the probability for $A\,\mathrm{U}^{\leqslant r}\,B$. The same holds for $\mathsf{Pr}^{\min}_s\big(\,A\,\mathrm{U}^{\leqslant r}\,B\,\big)$ and maximal or minimal probabilities for $A\,\mathrm{U}^{\geqslant r}\,B$ by an adaption of the linear programs presented in Sec. 4.3 for the $\Diamond^{\geqslant r}B$ reachability formulas. In particular, we obtain:

Lemma 1 (Optimal schedulers for until-quantiles). *For each of the existential until-quantiles there exists an optimal scheduler $\mathfrak{S}$ such that for all finite paths π_1 and π_2:*

if $\mathit{last}(\pi_1) = \mathit{last}(\pi_2)$ *and* $\mathit{rew}(\pi_1) = \mathit{rew}(\pi_2)$ *then* $\mathfrak{S}(\pi_1) = \mathfrak{S}(\pi_2)$ (*)

For existential quantiles $R = \mathsf{Qu}_s(\exists \ldots)$ *there are optimal finite-memory schedulers whose size is bounded by* $R{+}1$*.*

Similarly, the universal until-quantiles have adversarial schedulers $\mathfrak{S}$ satisfying (*). Thus, an analogous statement holds for universal until-quantiles and adversarial schedulers.

End components and limits of paths. It is well-known [9] that for any scheduler $\mathfrak{S}$, the *limit* of almost all $\mathfrak{S}$-paths constitutes an end component. Here, the limit of an infinite path $\tilde{\pi} = s_0\, \alpha_0\, s_1\, \alpha_1 \ldots$ is given by the set $\inf(\tilde{\pi})$ of states which appear infinitely often in $\tilde{\pi}$ and the function that assigns to each state $t \in \inf(\tilde{\pi})$ the set of actions α with $(s_i, \alpha_i) = (t, \alpha)$ for infinitely many i.

B.1 Quantiles for reachability with lower-reward bounds

Let C be the set of all states $t \in S$ that are contained in some (maximal) end component $(T, \mathfrak{A})$ with $\mathit{rew}(t', \alpha) > 0$ for some state $t' \in T$ and some action $\alpha \in \mathfrak{A}(t')$.

Lemma 2. *For all states* s *in* $\mathcal{M}$*, we have:*

$$\mathsf{Qu}_s\big(\forall\, \mathrm{P}_{<p}(\Diamond^{\geqslant ?} B)\big) \;=\; \infty \quad \textit{iff} \quad \mathsf{Pr}^{\max}_s\big(\Diamond(C \wedge \Diamond B)\big) \;\geqslant\; p$$

Proof. We first observe:

$$\begin{aligned} & \min\big\{\, r \in \mathbb{N} \,:\, \mathsf{Pr}^{\max}_s\big(\Diamond^{\geqslant r} B\big) < p \,\big\} = \infty \\ \text{iff}\quad & \text{there is no } r \in \mathbb{N} \text{ such that } \mathsf{Pr}^{\max}_s\big(\Diamond^{\geqslant r} B\big) < p \\ \text{iff}\quad & \text{for all } r \in \mathbb{N} \text{ there exists a scheduler } \mathfrak{S}_r \text{ with } \mathsf{Pr}^{\mathfrak{S}_r}_s\big(\Diamond^{\geqslant r} B\big) \geqslant p \end{aligned}$$

"$\Longleftarrow$": Suppose $\mathsf{Pr}^{\max}_s\big(\Diamond(C \wedge \Diamond B)\big) \geqslant p$. Let $\mathfrak{S}_{opt}$ be a (finite-memory) scheduler that maximizes the probability for $\Diamond(C \wedge \Diamond B)$ for all states. We pick some (finite-memory) scheduler $\mathfrak{S}_C$ such that from each state $t \in C$ with probability 1 all states of the maximal end component $(T, \mathfrak{A})$ with $t \in T$ will be visited infinitely often and each of its actions will be taken infinitely often. Then, the accumulated reward for almost all infinite $\mathfrak{S}_C$-paths starting in a C-state is ∞. Furthermore, let $\mathfrak{S}_{\Diamond B}$ be a (memoryless) scheduler that maximizes the probability to reach B for all states. Given $r \in \mathbb{N}$, we now regard the scheduler $\mathfrak{S}_r$ that operates in three phases:

- Phase 1: As long as C has not been reached, $\mathfrak{S}_r$ behaves as $\mathfrak{S}_{opt}$. As soon as C has been reached, $\mathfrak{S}_r$ switches from phase 1 to phase 2.
- Phase 2: $\mathfrak{S}_r$ mimics $\mathfrak{S}_C$, provided that the total accumulated reward is less than r. If the current state belongs to C and the total accumulated reward is larger or equal r then $\mathfrak{S}_r$ switches from phase 2 to phase 3.
- Phase 3: $\mathfrak{S}_r$ behaves as $\mathfrak{S}_{\Diamond B}$.

When entering a C-state in the first phase and the total accumulated reward is $\geqslant r$ then $\mathfrak{S}_r$ can move directly from phase 1 to the third phase.

We now use the fact that all states that belong to the same maximal end component have the same maximal reachability probabilities [8]. This yields that for each $\rho \in \{0, 1, \dots, r{-}1\}$ and all states t of a maximal end component $(T, \mathfrak{A})$ of $\mathcal{M}$ that contains at least one state-action pair with positive reward:

$$\begin{aligned} & \mathsf{Pr}^{\mathfrak{S}_r}_{t|rew=\rho}\big(\lozenge^{\geqslant r} B\big) \\ = & \sum_{t' \in T} \mathsf{Pr}^{\mathfrak{S}_C}_{t|rew=\rho}\big(C \,\mathrm{U}^{\geqslant r}\, t'\big) \cdot \mathsf{Pr}^{\mathfrak{S}_{\lozenge B}}_{t'}\big(\lozenge B\big) \\ = & \sum_{t' \in T} \mathsf{Pr}^{\mathfrak{S}_C}_{t|rew=\rho}\big(C \,\mathrm{U}^{\geqslant r}\, t'\big) \cdot \mathsf{Pr}^{\max}_{t}\big(\lozenge B\big) \\ = & \mathsf{Pr}^{\max}_{t}\big(\lozenge B\big) \end{aligned}$$

Here, the notation $\mathsf{Pr}^{\mathfrak{S}_r}_{t|rew=\rho}$ indicates the probability under $\mathfrak{S}_r$ under the condition that state t has been just entered by switching from phase 1 to phase 2, while the accumulated reward is ρ. We obtain:

$$\begin{aligned} & \mathsf{Pr}^{\mathfrak{S}_r}_{s}\big(\lozenge^{\geqslant r} B\big) \\ = & \sum_{0 \leqslant \rho < r} \sum_{(T,\mathfrak{A})} \sum_{t \in T} \mathsf{Pr}^{\mathfrak{S}_r}_{s}\big(\neg C \,\mathrm{U}((rew = \rho) \wedge t)\big) \cdot \mathsf{Pr}^{\mathfrak{S}_r}_{t|rew=\rho}\big(\lozenge^{\geqslant r} B\big) \\ & + \sum_{t \in C} \mathsf{Pr}^{\mathfrak{S}_r}_{s}\big(\neg C \,\mathrm{U}((rew \geqslant r) \wedge t)\big) \cdot \mathsf{Pr}^{\mathfrak{S}_{\lozenge B}}_{t}\big(\lozenge B\big) \\ = & \sum_{t \in C} \mathsf{Pr}^{\mathfrak{S}_{opt}}_{s}\big(\neg C \,\mathrm{U}\, t\big) \cdot \mathsf{Pr}^{\max}_{t}\big(\lozenge B\big) \\ = & \mathsf{Pr}^{\max}_{s}\big(\lozenge(C \wedge \lozenge B)\big) \end{aligned}$$

where $(T, \mathfrak{A})$ ranges over all maximal end components that contain a state-action pair with positive reward.

"$\Longrightarrow$": We now suppose that the quantile for the state s and the objective $\forall\, \mathrm{P}_{<p}(\lozenge^{\geqslant ?} B)$ is ∞. Let $\big(\mathfrak{S}_r\big)_{r \in \mathbb{N}}$ be a family of schedulers such that:

$$\mathsf{Pr}^{\mathfrak{S}_r}_{s}\big(\lozenge^{\geqslant r} B\big) \geqslant p$$

The task is to show that $\mathsf{Pr}^{\max}_{s}(\lozenge(C \wedge \lozenge B)) \geqslant p$. For this we show that for each $\varepsilon > 0$ there exists a scheduler $\mathfrak{S}$ such that $\mathsf{Pr}^{\mathfrak{S}}_{s}(\lozenge(C \wedge \lozenge B)) \geqslant p - \varepsilon$.

In what follows, we fix some positive ε. There exists some $r \in \mathbb{N}$ such that for each scheduler $\mathfrak{S}$:

$$\mathsf{Pr}^{\mathfrak{S}}_{s}\big((\neg C)\,\mathrm{U}^{\geqslant r}\, B\big) \;<\; \varepsilon$$

This is due to the fact that the limit of almost all $\mathfrak{S}$-paths constitutes an end component and that the reward earned in end components not contained in C

is zero. For scheduler $\mathfrak{S} = \mathfrak{S}_r$ we obtain:

$$\begin{aligned} p \ &\leqslant \ \mathsf{Pr}^{\mathfrak{S}}_s\big(\Diamond^{\geqslant r} B\big) \\ &= \ \mathsf{Pr}^{\mathfrak{S}}_s\big((\neg C)\,\mathrm{U}^{\geqslant r}\,B\big) \ + \ \mathsf{Pr}^{\mathfrak{S}}_s\big(\Diamond(C \wedge \Diamond((\mathit{rew} \geqslant r) \wedge B))\big) \\ &\leqslant \ \varepsilon \ + \ \mathsf{Pr}^{\mathfrak{S}}_s\big(\Diamond(C \wedge \Diamond B)\big) \end{aligned}$$

Hence, $\mathsf{Pr}^{\mathfrak{S}}_s\big(\Diamond(C \wedge \Diamond B)\big)$ is at least $p - \varepsilon$. ■

Let $\widetilde{\mathcal{M}}$ be the MDP that results from $\mathcal{M}$ by adding two new states *goal* and *fail* and a fresh action symbol τ with transition probabilities:

$$\begin{aligned} P(t, \tau, \mathit{goal}) \ &= \ \mathsf{Pr}^{\max}_{\mathcal{M},t}\big(\Diamond B\big) \\ P(t, \tau, \mathit{fail}) \ &= \ 1 - \mathsf{Pr}^{\max}_{\mathcal{M},t}\big(\Diamond B\big) \end{aligned}$$

if $t \in C$ and $P(s, \tau, s') = 0$ for all states $s \in S \setminus C$, $s' \in S$. The outgoing transitions of the new states *goal* and *fail* are irrelevant for our purposes. We then have:

$$\mathsf{Pr}^{\max}_{\mathcal{M},s}\big(\Diamond(C \wedge \Diamond B)\big) \quad = \quad \mathsf{Pr}^{\max}_{\widetilde{\mathcal{M}},s}\big(\Diamond \mathit{goal}\big)$$

The generation of $\widetilde{\mathcal{M}}$ mainly requires the computation of the values $\mathsf{Pr}^{\max}_{\mathcal{M},s}(\Diamond B)$ and the computation of the maximal end components of $\mathcal{M}$. The former can be done using graph algorithms and linear-programming techniques in time polynomial in the size of $\mathcal{M}$, while the latter is possible using standard algorithms in time quadratic in the size of the underlying graph of $\mathcal{M}$.

We now consider the existential quantile for until-properties with lower reward bounds. We have:

$$\begin{aligned} &\min\big\{\, r \in \mathbb{N} \,:\, \mathsf{Pr}^{\min}_s\big(\Diamond^{\geqslant r} B\big) < p \,\big\} = \infty \\ \text{iff} \quad &\text{there is no } r \in \mathbb{N} \text{ such that } \mathsf{Pr}^{\min}_s\big(\Diamond^{\geqslant r} B\big) < p \\ \text{iff} \quad &\mathsf{Pr}^{\min}_s\big(\Diamond^{\geqslant r} B\big) \geqslant p \text{ for all } r \in \mathbb{N} \end{aligned}$$

Obviously, this is the case if under each scheduler, with probability at least p, the set B will be visited infinitely often and the accumulated reward tends to infinity. Let $\mathit{posR} \subseteq S \times \mathit{Act}$ be the set of state-action pairs (s, α) with $\mathit{rew}(s, \alpha) > 0$.

Lemma 3. *For all states s in $\mathcal{M}$, we have:*

$$\mathsf{Qu}_s\big(\exists\mathrm{P}_{<p}(\Diamond^{\geqslant ?} B)\big) \ = \ \infty \quad \textit{iff} \quad \mathsf{Pr}^{\min}_s\big(\Box\Diamond B \wedge \Box\Diamond \mathit{posR}\big) \ \geqslant \ p$$

To compute the minimal probability for the generalized Büchi condition $\Box\Diamond B \wedge \Box\Diamond \mathit{posR}$ we can rely on standard techniques. We compute the set D consisting of states that are contained in some end component $(T, \mathfrak{A})$ with $T \cap B = \varnothing$ or with $\mathit{rew}(t', \alpha) = 0$ for all actions $\alpha \in \mathfrak{A}(t')$ and states $t' \in T$. Then:

$$\mathsf{Pr}^{\min}_s\big(\Box\Diamond B \wedge \Box\Diamond \mathit{posR}\big) \quad = \quad 1 - \mathsf{Pr}^{\max}_s\big(\Diamond D\big)$$

Corollary 1. *The following two problems are in* P*:*

(a) *decide whether* $\mathsf{Qu}_s\big(\,\forall\,\mathrm{P}_{<p}(\Diamond^{\geqslant ?}B)\big) = \infty$

(b) *decide whether* $\mathsf{Qu}_s\big(\,\exists\,\mathrm{P}_{<p}(\Diamond^{\geqslant ?}B)\big) = \infty$

B.2 Expectation quantiles

We now explain how to compute expectation quantiles when no assumptions on the end components are imposed. That is, there might exist end components that contain no state-action pair with positive energy or utility reward. Essentially, we can use the same linear program as sketched in Section 5, restricted to those variables $x_{s,e}$ where $\mathsf{ExpUtil}_s^{\max}(\mathit{energy} \leqslant e) < \infty$ resp. $\mathsf{ExpUtil}_s^{\min}(\mathit{energy} \leqslant e) < \infty$. Furthermore, we have to identify those states s where the quantile is infinite.

Zero-reward and reward-divergent end components

An end component $(T,\mathfrak{A})$ of $\mathcal{M}$ with $\mathit{erew}(t,\alpha) = 0$ for all states $t \in T$ and actions $\alpha \in \mathfrak{A}(t)$ is called a *zero-energy end component.* $(T,\mathfrak{A})$ is called *energy-divergent* if there is some state-action pair (t,α) with $t \in T$, $\alpha \in \mathfrak{A}(t)$ and $\mathit{erew}(t,\alpha) > 0$.

We write ZE for the set of all states $t \in S$ that are contained in some zero-energy end component, and ED for the set of all states t that belong to some energy-divergent end component.

Similar notations are used for the utility reward function. End components $(T,\mathfrak{A})$ with $\mathit{urew}(t,\alpha) > 0$ for some state-action pair (t,α) where $t \in T$ and $\alpha \in \mathfrak{A}(t)$ are said to be *utility-divergent.* End components that are not utility-divergent are called *zero-utility* end components. UD denotes the set of states that are contained in some utility-divergent end component, while ZU stands for the set of states that are contained in some zero-utility end component.

The sets UD and ED are obtained using standard algorithms for computing maximal end components, see, e.g., [CY95] and [9,5]. Note that $t \in \mathit{UD}$ if and only if there exists a maximal end component of $\mathcal{M}$ that contains t and that is utility-divergent. The analogous statement holds for ED. This, however, does not hold for ZE and ZU since zero-energy end components might be contained in energy-divergent maximal end components. Nevertheless, the computation of ZE and ZU can also be carried out using algorithms to determine maximal end components of the sub-MDPs $\mathcal{M}|_{\mathit{erew}=0}$ and $\mathcal{M}|_{\mathit{urew}=0}$, respectively. For instance, $\mathcal{M}|_{\mathit{erew}=0}$ results from $\mathcal{M}$ by successively removing actions and states. We start dropping all actions α with $\mathit{erew}(s,\alpha) > 0$ from $\mathit{Act}(s)$. We then remove all states s where $\mathit{Act}(s)$ is empty and all actions β with $P(s',\beta,s) > 0$ from $\mathit{Act}(s')$, and so on.

Lemma 4. *The sets ZE, ED, ZU and UD can be computed in time quadratic in the size of the underlying graph of $\mathcal{M}$.*

Expected accumulated utility for fixed energy budget

For each infinite path $\tilde{\pi} = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2\ldots$ and fixed energy budget $e \in \mathbb{N}$, $\mathit{utility}|_{\mathit{energy}\leqslant e}(\tilde{\pi}) = \infty$ if and only if for each $u \in \mathbb{N}$ there exists a finite prefix π of $\tilde{\pi}$ such that $\mathit{energy}(\pi) \leqslant e$ and $\mathit{utility}(\pi) > u$. As the energy budget e is fixed, this is only possible if there exists some $k \in \mathbb{N}$ such that $\mathit{erew}(s_i,\alpha_i) = 0$ for all $i \geqslant k$ and $\mathit{urew}(s_i,\alpha_i) > 0$ for infinitely many indices i. Hence, if $\tilde{\pi} = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2\ldots$ then $\mathit{utility}|_{\mathit{energy}\leqslant e}(\tilde{\pi}) = \infty$ if and only if there exists $k \in \mathbb{N}$ such that

$$\begin{aligned}
&(1) \quad \mathit{energy}(\,\mathit{pref}(\tilde{\pi},k)\,) \leqslant e\\
&(2) \quad \forall j \geqslant k.\ \mathit{erew}(s_j,\alpha_j) = 0\\
&(3) \quad \overset{\infty}{\exists}\, j \in \mathbb{N}.\ \mathit{urew}(s_j,\alpha_j) > 0
\end{aligned}$$

Let ZEUD denote the set of all states $t \in S$ that are contained in some zero-energy utility-divergent end component.

Lemma 5. *For all states s in $\mathcal{M}$ and all $e \in \mathbb{N}$:*

(a) $\mathsf{ExpUtil}^{\max}_s(\mathit{energy} \leqslant e) < \infty$ *iff* $s \not\models \exists\Diamond((\mathit{energy} \leqslant e) \wedge \mathit{ZEUD}))$

(b) $\mathsf{ExpUtil}^{\min}_s(\mathit{energy} \leqslant e) < \infty$ *iff* $\mathsf{Pr}^{\min}_s(\varphi) = 0$

where φ denotes the path property given by $\tilde{\pi} \models \varphi$ iff there exists some $k \in \mathbb{N}$ such that conditions (1), (2) and (3) hold.

Proof. To prove part (a), we first suppose $s \models \exists\Diamond((\mathit{energy} \leqslant e) \wedge \mathit{ZEUD}))$ and show that there is a scheduler $\mathfrak{S}$ with $\mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{energy} \leqslant e) = \infty$. Let $\mathfrak{T}$ be any finite-memory scheduler with $\mathsf{Pr}^{\mathfrak{T}}_t(\Box \mathit{ZEUD}) = 1$ for all states $t \in \mathit{ZEUD}$ and such that the limit of almost all $\mathfrak{T}$-paths that start in some state $t \in \mathit{ZEUD}$ is a zero-energy utility-divergent end component. We may pick any scheduler $\mathfrak{S}$ that generates a finite path π with $\mathit{energy}(\pi) \leqslant e$ and $\mathit{last}(\pi) \in \mathit{ZEUD}$ and that behaves as $\mathfrak{T}$ as soon as a state in ZEUD has been reached. Then, $\mathit{utility}|_{\mathit{energy}\leqslant e}(\tilde{\pi}) = \infty$ for almost all infinite $\mathfrak{S}$-paths $\tilde{\pi}$ in the cylinder set of π. This yields:

$$\mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \quad = \quad \infty$$

To prove the second part of statement (a), we suppose $s \not\models \exists\Diamond((\mathit{energy} \leqslant e) \wedge \mathit{ZEUD}))$. The task is to show that $\mathsf{ExpUtil}^{\max}_{\mathcal{M},s}(\mathit{energy} \leqslant e)$ is finite. We regard the MDP $\mathcal{M}_e$ that arises from $\mathcal{M}$ by mimicking the energy function erew of $\mathcal{M}$ by a counter with values in $\{0,1,\ldots,e\}$. The enabled actions of state $\langle t,f\rangle \in S \times \{0,1,\ldots,e\}$ in $\mathcal{M}_e$ are the actions $\alpha \in \mathit{Act}_{\mathcal{M}}(t)$ where $\mathit{erew}(t,\alpha) \leqslant e{-}f$. Furthermore, $\mathcal{M}_e$ collapses all zero-energy zero-utility end components of $\mathcal{M}$ into a single state and discards $\mathcal{M}$'s actions inside zero-energy zero-utility end components.[3] Then, $\mathsf{Pr}^{\min}_{\mathcal{M}_e,\langle s,0\rangle}(\Diamond \mathsf{final}) = 1$ where final

[3] We drop here the precise definition of $\mathcal{M}_e$. A similar construction of the MDP $\widetilde{\mathcal{M}}$ that arises from $\mathcal{M}$ by identifying all states that belong to some zero-utility end component is presented in the proof of Lemma 6.

is an atomic proposition characterizing all states of $\mathcal{M}_e$ that do not have any enabled action. Thus, $\mathsf{ExpUtil}^{\max}_{\mathcal{M}_e,\langle s,0\rangle}(\Diamond \mathsf{final})$ is finite, and we have:

$$\mathsf{ExpUtil}^{\max}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \quad = \quad \mathsf{ExpUtil}^{\max}_{\mathcal{M}_e,\langle s,0\rangle}(\Diamond \mathsf{final})$$

We now turn to the proof of part (b). Let us first consider the case where $\mathsf{ExpUtil}^{\min}_{s}(\mathit{energy} \leqslant e)$ is finite, i.e., $\mathsf{ExpUtil}^{\mathfrak{S}}_{s}(\mathit{energy} \leqslant e) < \infty$ for some scheduler $\mathfrak{S}$. Then:

$$\mathsf{Pr}^{\mathfrak{S}}_{s}\big\{\, \tilde{\pi} \in \mathit{IPaths} \,:\, \mathit{utility}|_{\mathit{energy}\leqslant e}(\tilde{\pi}) = \infty \,\big\} \quad = \quad 0$$

But then, $\mathsf{Pr}^{\mathfrak{S}}_{s}(\varphi) = 0$.

Vice versa, suppose $\mathsf{Pr}^{\min}_{s}(\varphi) = 0$. We pick some finite-memory scheduler $\mathfrak{S}$ with $\mathsf{Pr}^{\mathfrak{S}}_{s}(\varphi) = 0$. Then, the limit of almost all $\mathfrak{S}$-paths from s is either an energy-divergent end component or a zero-energy zero-utility end component. Let $\mathcal{M}_e$ be the MDP as above. When regarding $\mathfrak{S}$ as a scheduler for $\mathcal{M}_e$ (as done in the proof of Lemma 6), $\mathsf{Pr}^{\mathfrak{S}}_{\mathcal{M}_e,\langle s,0\rangle}(\Diamond \mathsf{final}) = 1$ as $\mathcal{M}_e$ has no zero-energy zero-utility end components (by construction). Moreover:

$$\mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \quad = \quad \mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M}_e,\langle s,0\rangle}(\Diamond \mathsf{final}) \quad < \quad \infty$$

Hence, $\mathsf{ExpUtil}^{\min}_{\mathcal{M},s}(\mathit{energy} \leqslant e)$ is finite. ■

Existential utility-expectation quantiles

The procedure to compute the expectation quantiles $\mathsf{Qu}_s(\,\exists\, \mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\,)$ and $\mathsf{Qu}_s(\,\forall\, \mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\,)$ by an iterative search $e = 0, 1, 2, \ldots$ as explained in Section 5 requires a preprocessing that identifies all states where the quantile has value ∞. We start with explanations for the case of existential expectation quantiles. Obviously, for each utility bound $u \in \mathbb{Q}$:

$$\begin{array}{ll} & \mathsf{Qu}_s\big(\,\exists\, \mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\,\big) \;=\; \infty \\ \text{iff} & \big\{\, e \in \mathbb{N} \,:\, \mathsf{ExpUtil}^{\max}_{s}(\mathit{energy} \leqslant e) > u \big\} = \varnothing \\ \text{iff} & \sup\limits_{e \in \mathbb{N}} \mathsf{ExpUtil}^{\max}_{s}(\mathit{energy} \leqslant e) \leqslant u \\ \text{iff} & \mathsf{ExpUtil}^{\mathfrak{S}}_{s}(\mathit{energy} \leqslant e) \leqslant u \text{ for all } e \in \mathbb{N} \text{ and all schedulers } \mathfrak{S} \end{array}$$

Let $\mathit{Lim}(\mathit{UD})$ denote the event consisting of all infinite paths $\tilde{\pi}$ such that the limit of $\tilde{\pi}$ is a utility-divergent end component. Likewise, we write $\mathit{Lim}(\mathit{ZU})$ to denote the event given by $\tilde{\pi} \models \mathit{Lim}(\mathit{ZU})$ iff the limit of $\tilde{\pi}$ is a zero-utility end component. Hence, $\mathsf{Pr}^{\min}_{s}(\mathit{Lim}(\mathit{UD}) \vee \mathit{Lim}(\mathit{ZU})) = 1$.

Lemma 6. *For each state s:*

$$s \models \exists \Diamond\, \mathit{UD} \quad \textit{iff} \quad \sup_{e \in \mathbb{N}} \mathsf{ExpUtil}^{\max}_{s}(\mathit{energy} \leqslant e) \;=\; \infty$$

Proof. To prove "$\Longrightarrow$", we suppose $s \models \exists\lozenge UD$ and pick some utility bound $u \in \mathbb{Q}$. The task is to show that there exists an energy bound $e \in \mathbb{N}$ with $\mathsf{ExpUtil}^{\max}_s(\mathit{energy} \leqslant e) > u$. The latter amounts proving the existence of some scheduler $\mathfrak{U}$ with $\mathsf{ExpUtil}^{\mathfrak{U}}_s(\mathit{energy} \leqslant e) > u$.

As $s \models \exists\lozenge UD$, we may choose a shortest finite path π starting in s and ending in some state in UD. Let $\mathfrak{S}$ be a (memoryless) scheduler such that π is a $\mathfrak{S}$-path. With $e_1 = \mathit{energy}(\pi)$ and p the probability of π we get:

$$\mathsf{Pr}^{\mathfrak{S}}_s\big(\lozenge((\mathit{energy} \leqslant e_1) \wedge UD)\big) \;\geqslant\; p \;>\; 0$$

Let $k \in \mathbb{N}$ be a positive natural number with $p \geqslant 1/k$. We now consider any finite-memory scheduler $\mathfrak{T}$ that "realizes" the utility-divergent end components, i.e., the limit of almost all infinite $\mathfrak{T}$-paths starting in some state $t \in UD$ is a utility-divergent end component. Clearly, along all these infinite $\mathfrak{T}$-paths, the accumulated utility tends to ∞. Hence, there exists some $e_2 \in \mathbb{N}$ such that for all states $t \in UD$:

$$\mathsf{Pr}^{\mathfrak{T}}_t\big(\lozenge((\mathit{energy} \leqslant e_2) \wedge (\mathit{utility} > 2k \cdot u))\big) \;>\; \frac{1}{2}$$

Let $\mathfrak{U} = \mathfrak{S} \circ \mathfrak{T}$ be the scheduler that behaves like $\mathfrak{S}$ as long as no state in UD has been visited. As soon as a state in UD is visited, then $\mathfrak{U}$ mimics $\mathfrak{T}$. We get:

$$\mathsf{Pr}^{\mathfrak{U}}_s\{\, \tilde{\pi} \;:\; \tilde{\pi} \models \varphi \,\} \;>\; \frac{1}{2k}$$

where φ is the following LTL-like formula:

$$\varphi \;=\; \lozenge\big((\mathit{energy} \leqslant e_1) \wedge UD\big) \;\wedge\; \lozenge\big((\mathit{energy} \leqslant e) \wedge (\mathit{utility} > 2k \cdot u)\big)$$

and $e = e_1 + e_2$. Hence:

$$\mathsf{ExpUtil}^{\mathfrak{U}}_t\big(\mathit{energy} \leqslant e\big) \;>\; \frac{1}{2k} \cdot 2k \cdot u \;=\; u$$

This yields the claim.

We now turn to the proof of the implication "$\Longleftarrow$". The task is to show that $s \not\models \exists\lozenge UD$ implies the existence of some $u \in \mathbb{Q}$ such that

$$\mathsf{ExpUtil}^{\max}_s(\mathit{energy} \leqslant e) \leqslant u$$

for all energy bounds $e \in \mathbb{N}$. The assumption that no utility-divergent end component is reachable from s yields that all end components that are reachable from s enjoy the zero-utility property. Since under each scheduler the limit of almost all infinite paths is an end component we get:

$$\mathsf{Pr}^{\min}_s(\mathit{Lim}(ZU)) \;=\; 1$$

In what follows, we define a new MDP $\widetilde{\mathcal{M}}$ with a reward function $\widetilde{\mathit{rew}}$ and a goal state that will be reached almost surely under each scheduler for $\widetilde{\mathcal{M}}$ such that, for each $e \in \mathbb{N}$, the value $\mathsf{ExpUtil}^{\max}_{\mathcal{M},s}(\mathit{energy} \leqslant e)$ is bounded by the maximal expected reward until reaching the goal state in $\widetilde{\mathcal{M}}$.

The idea for the definition of $\widetilde{\mathcal{M}}$ is to collapse the effect of all transitions inside zero-utility end components. Intuitively, this is justified as for the accumulated utility in $\mathcal{M}$, the behavior inside zero-utility end components is irrelevant. The accumulated energy might increase inside zero-utility end components. However, we are interested only in an upper bound for the expected total accumulated utility in the new MDP $\widetilde{\mathcal{M}}$.

We first define an equivalence relation $\sim_{ZU}$ on ZU. Let $t_1, t_2 \in ZU$. Then, $t_1 \sim_{ZU} t_2$ iff there exists some zero-utility end component that contains t_1 and t_2. For $t \in ZU$, let $[t]_{ZU}$ denote the $\sim_{ZU}$-equivalence class of t and let

$$\mathit{Act}_{ZU}(t) \;=\; \bigl\{\, \beta \in \mathit{Act}(t) \;:\; \mathit{urew}(s,\beta) = 0,\; \{t' \in S : P(t,\beta,t') > 0\} \subseteq [t]_{ZU} \,\bigr\}$$

denote the set of actions of t inside some zero-utility end component. We write $\sim$ to denote the following equivalence relation on the state space S of $\mathcal{M}$:

$$s_1 \sim s_2 \qquad \text{iff} \qquad (s_1 = s_2) \;\vee\; (s_1, s_2 \in ZU \;\wedge\; s_1 \sim_{ZU} s_2)$$

Let $[s]$ denote the equivalence class of s with respect to $\sim$. Thus, $[s] = \{s\}$ if $s \in S \setminus ZU$, while $[t] = [t]_{ZU}$ for $t \in ZU$. The state space of the new MDP $\widetilde{\mathcal{M}}$ is:

$$\widetilde{S} \;=\; \bigl\{\; [s] \;:\; s \in S \;\bigr\} \;\cup\; \{\mathsf{goal}\}$$

For simplicity, let us suppose that the actions in $\mathcal{M}$ have been renamed such that $\mathit{Act}(s_1) \cap \mathit{Act}(s_2) = \varnothing$ if $s_1 \neq s_2$. Then, the action set of $\widetilde{\mathcal{M}}$ is

$$\widetilde{\mathit{Act}} \;=\; \mathit{Act} \cup \{\tau\}$$

The transition probability function $\widetilde{P}$ of $\widetilde{\mathcal{M}}$ is defined as follows.

– If $s \in S \setminus ZU$ and $\alpha \in \mathit{Act}(s)$, then $\widetilde{P}([s],\alpha,[s']) = P(s,\alpha,[s'])$ for $s' \in S$, $\widetilde{P}([s],\alpha,\mathsf{goal}) = 0$ and $\mathit{rew}(s,\alpha) = \mathit{urew}(s,\alpha)$.[4] Action τ is not enabled in the states $[s]$ where $s \in S \setminus ZU$. Thus, the set $\widetilde{\mathit{Act}}([s])$ of actions that are enabled in $[s]$ equals $\mathit{Act}(s)$.

– Let $Z \subseteq ZU$ be a $\sim_{ZU}$-equivalence class. The set of actions that are enabled in Z is:
$$\widetilde{\mathit{Act}}(Z) \;=\; \{\tau\} \;\cup\; \bigcup_{t \in Z} \mathit{Act}(t) \setminus \mathit{Act}_{ZU}(t)$$
If $t \in Z$ and $\alpha \in \mathit{Act}(t) \setminus \mathit{Act}_{ZU}(t)$ then $\widetilde{P}(Z,\alpha,[s']) = P(t,\alpha,[s'])$ for $s' \in S$, $\widetilde{P}(Z,\alpha,\mathsf{goal}) = 0$ and $\mathit{rew}(Z,\alpha) = \mathit{urew}(t,\alpha)$. Furthermore, $\widetilde{P}(Z,\tau,\mathsf{goal}) = 1$ and $\mathit{rew}(Z,\tau) = 0$.

– The fresh state goal is a trap with a single τ-labeled zero-reward self-loop, i.e., $\widetilde{P}(\mathsf{goal},\tau,\mathsf{goal}) = 1$ and and $\mathit{rew}(\mathsf{goal},\tau) = 0$.

[4] Here and in what follows, for $s \in S$, $\alpha \in \mathit{Act}$ and $R \subseteq S$, we write $P(s,\alpha,R)$ for $\sum_{s' \in R} P(s,\alpha,s')$.

As $\mathsf{Pr}^{\min}_{\mathcal{M},s}(\mathit{Lim}(ZU)) = 1$, the only end component that is accessible from state s in $\widetilde{\mathcal{M}}$ consists of the goal state. In particular:

$$\mathsf{Pr}^{\min}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal}) \; = \; 1$$

Moreover, for each scheduler $\mathfrak{S}$ for $\mathcal{M}$ there exists a corresponding (randomized) scheduler $\widetilde{\mathfrak{S}}$ for $\widetilde{\mathcal{M}}$ that mimicks $\mathfrak{S}$'s behavior inside zero-utility end components by a probabilistic choice. Speaking roughly, if the current state is neither the goal state nor a $\sim_{ZU}$-equivalence class, then $\widetilde{\mathfrak{S}}$ behaves as $\mathfrak{S}$. When entering a $\sim_{ZU}$-equivalence class Z, then $\widetilde{\mathfrak{S}}$ selects an action in $\widetilde{\mathit{Act}}(Z)$ probabilistically according to the probabilities for $\mathfrak{S}$ to generate a finite path $\pi = t_0\, \beta_0\, t_1\, \beta_1 \ldots \beta_{n-1}\, t_n\, \alpha\, v$ where $t_0, \ldots, t_n$ and their actions $\beta_0, \ldots, \beta_{n-1}$ belong to Z and $\alpha \notin \mathit{Act}_{ZU}(t_n)$ or to stay forever in ZU (in which case $\widetilde{\mathfrak{S}}$ moves to the goal state by scheduling τ). Then, the expected total accumulated utility for $\mathfrak{S}$ from state s in $\mathcal{M}$ equals the expected total utility for $\widetilde{\mathfrak{S}}$ from state $[s]$ in $\widetilde{\mathcal{M}}$, which again agrees with the expected accumulated utility until reaching goal from $[s]$ under scheduler $\widetilde{\mathfrak{S}}$. That is:

$$\mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{true}) \quad = \quad \mathsf{ExpRew}^{\widetilde{\mathfrak{S}}}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal})$$

Recall that the limit of almost all $\mathfrak{S}$-paths is a zero-utility end component in $\mathcal{M}$. Thus, the expected total accumulated utility exists in $\mathcal{M}$. Hence, for all $e \in \mathbb{N}$ we have:

$$\mathsf{ExpUtil}^{\max}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \quad \leqslant \quad \mathsf{ExpRew}^{\max}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal})$$

and therefore:

$$\sup_{e \in \mathbb{N}} \mathsf{ExpUtil}^{\max}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \quad \leqslant \quad \mathsf{ExpRew}^{\max}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal})$$

The latter is finite as $\Diamond \mathsf{goal}$ holds almost surely under each scheduler for $\widetilde{\mathcal{M}}$. ■

Corollary 2 (Infinite existential expectation quantiles). *For each $u \in \mathbb{Q}$, the following statements are equivalent:*

(a) $\mathsf{Qu}_s\big(\exists\, \mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\big) = \infty$

(b) $\sup\limits_{e \in \mathbb{N}} \mathsf{ExpUtil}^{\max}_{s}(\mathit{energy} \leqslant e) \; \leqslant \; u$

(c) $s \not\models \exists \Diamond\, \mathit{UD}$ *and* $\mathsf{ExpRew}^{\max}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal}) \leqslant u$

where $\widetilde{\mathcal{M}}$ is defined as in the proof of Lemma 6.

Proof. The equivalence of (a) and (b) is obvious. The implication "(c) $\Longrightarrow$ (b)" has been shown in the proof of Lemma 6.

It remains to prove the implication "(b) $\Longrightarrow$ (c)". The fact that (b) implies $s \not\models \exists \Diamond\, \mathit{UD}$ is a consequence of Lemma 6. To see why (b) implies $\mathsf{ExpRew}^{\max}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal}) \leqslant$

u we pick some scheduler $\widetilde{\mathfrak{S}}$ for $\widetilde{\mathcal{M}}$. It suffices to show that there exists a scheduler $\mathfrak{S}$ for $\mathcal{M}$ such that the expected total accumulated utility under $\mathfrak{S}$ agrees with the expected accumulated reward until reaching the goal state in $\widetilde{\mathcal{M}}$ under the scheduler $\widetilde{\mathfrak{S}}$. That is:

$$\mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{true}) \quad = \quad \mathsf{ExpRew}^{\widetilde{\mathfrak{S}}}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal})$$

To prove the existence of such a scheduler $\mathfrak{S}$, we pick a finite-memory scheduler $\mathfrak{T}$ for $\mathcal{M}$ realizing the zero-utility end components in the sense that $\mathsf{Pr}^{\mathfrak{T}}_{\mathcal{M},t'}(ZU \,\mathrm{U}\, t) = 1$ for all states $t, t' \in ZU$ with $t \sim_{ZU} t'$, where $ZU \,\mathrm{U}\, t$ is used as a short form notation for all infinite paths that have a finite prefix $s_0\,\alpha_0\,s_1\,\alpha_1 \ldots \alpha_{k-1}\,s_k$ with $s_k = t$ and $\mathit{urew}(s_i, \alpha_i) = 0$ for $0 \leqslant i < k$.

Let $\mathfrak{S}$ be the scheduler that behaves as $\widetilde{\mathfrak{S}}$ for the states $s \in S \setminus ZU$ (where we identify s and the singleton $[s] = \{s\}$) and mimicks $\widetilde{\mathfrak{S}}$'s decisions for the $\sim_{ZU}$-equivalence classes by simulating the actions $\alpha \in \mathit{Act}(t) \setminus \mathit{Act}_{ZU}(t)$ for $t \in ZU$ by first following $\mathfrak{T}$'s decisions until state t has been reached, and then selecting action α. If $\widetilde{\mathfrak{S}}$ moves to the goal state via the action τ from some $\sim_{ZU}$-equivalence class $Z \in \widetilde{S}$, then $\mathfrak{S}$ changes mode and behaves as $\mathfrak{T}$ from then on. We obtain:

$$\begin{aligned} \mathsf{ExpRew}^{\widetilde{\mathfrak{S}}}_{\widetilde{\mathcal{M}},[s]}(\Diamond \mathsf{goal}) \quad &= \quad \mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{true}) \\ &= \quad \sup_{e \in \mathbb{N}} \mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \end{aligned}$$

As $\sup\limits_{e \in \mathbb{N}} \mathsf{ExpUtil}^{\mathfrak{S}}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \leqslant u$ by assumption (b), this yields the claim. ■

Universal utility-expectation quantiles

Let us now consider universal expectation quantiles. Obviously, for each utility bound $u \in \mathbb{Q}$:

$$\begin{aligned} & \mathsf{Qu}_s\big(\forall\, \mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\big) \;=\; \infty \\ \text{iff} \quad & \big\{\, e \in \mathbb{N} \,:\, \mathsf{ExpUtil}^{\min}_s(\mathit{energy} \leqslant e) > u \big\} = \varnothing \\ \text{iff} \quad & \sup_{e \in \mathbb{N}} \; \mathsf{ExpUtil}^{\min}_s(\mathit{energy} \leqslant e) \leqslant u \\ \text{iff} \quad & \text{for each } e \in \mathbb{N} \text{ there is some scheduler } \mathfrak{S}_e \\ & \text{such that } \mathsf{ExpUtil}^{\mathfrak{S}_e}_s(\mathit{energy} \leqslant e) \leqslant u \end{aligned}$$

We first observe that for reasoning about $\mathsf{ExpUtil}^{\min}_s(\mathit{energy} \leqslant e)$ it suffices to consider schedulers that stay forever in ZU as soon as they have reached some state in ZU; see Lemma 7 below.

With abuse of notations, we interpret ZU as a set of states or as a set of states and actions. Thus, if $\tilde{\pi} = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2 \ldots$ is an infinite path then:

$$\tilde{\pi} \models \Diamond ZU \quad \text{iff} \quad s_k \in ZU \text{ for some } k \in \mathbb{N}$$

$$\tilde{\pi} \models \Box ZU \quad \text{iff} \quad s_k \in ZU \text{ and } \mathit{urew}(s_k, \alpha_k) = 0 \text{ for all } k \in \mathbb{N}.$$

Moreover, $\tilde{\pi} \models \Box(ZU \to \Box ZU)$ if and only if either $ZU \cap \{s_j : j \in \mathbb{N}\} = \varnothing$ or there is some $k \in \mathbb{N}$ with $ZU \cap \{s_0, \ldots, s_{k-1}\} = \varnothing$ and $\mathit{suff}(\tilde{\pi}, k) \models \Box ZU$, where $\mathit{suff}(\tilde{\pi}, k) = s_k\, \alpha_k\, s_{k+1}\, \alpha_{k+1}\, s_{k+2}\, \alpha_{k+2} \ldots$ denotes the suffix of $\tilde{\pi}$ starting at position k.

Lemma 7. *Let $\mathfrak{S}$ be a scheduler for $\mathcal{M}$. Then, there exists a scheduler $\mathfrak{S}'$ for $\mathcal{M}$ such that for each state s and each energy bound $e \in \mathbb{N}$:*

$$\mathsf{ExpUtil}_s^{\mathfrak{S}'}(\mathit{energy} \leqslant e) \quad \leqslant \quad \mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{energy} \leqslant e)$$

and

$$\mathsf{Pr}_s^{\mathfrak{S}'}\big(\Box(ZU \to \Box ZU)\big) \quad = \quad 1$$

Proof. Again, let $\mathfrak{T}$ be a finite-memory scheduler realizing the zero-utility end components, i.e., $\mathsf{Pr}_t^{\mathfrak{T}}(\Box ZU) = 1$ for each state $t \in ZU$. Given an arbitrary scheduler $\mathfrak{S}$, we define $\mathfrak{S}'$ to be a scheduler $\mathfrak{S}'$ that behaves as $\mathfrak{S}$ until some ZU-state has been reached, in which case $\mathfrak{S}'$ switches mode and behaves as $\mathfrak{T}$ from then one. ■

Lemma 8. *If $\mathfrak{S}$ is a scheduler for $\mathcal{M}$ such that $\mathsf{Pr}_s^{\mathfrak{S}}(\mathit{Lim}(UD)) > 0$, then*

$$\sup_{e \in \mathbb{N}} \mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{energy} \leqslant e) = \infty$$

Proof. We first observe that

$$\mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{true}) \quad = \quad \sup_{e \in \mathbb{N}} \mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{energy} \leqslant e)$$

is the expected total utility under scheduler $\mathfrak{S}$ from state s. The claim then follows from the fact that the accumulated utility value of the prefixes along paths whose limit is a utility-diverengent end component converges to ∞. The assumption $\mathsf{Pr}_s^{\mathfrak{S}}(\mathit{Lim}(UD)) > 0$ yields that these paths have positive measure. ■

Corollary 3. *If* $\mathsf{Pr}_s^{\min}(\mathit{Lim}(UD)) > 0$ *then* $\sup_{e \in \mathbb{N}} \mathsf{ExpUtil}_s^{\min}(\mathit{energy} \leqslant e) = \infty$.

The remainder of this section addresses the case $\mathsf{Pr}_s^{\min}(\mathit{Lim}(UD)) = 0$, in which case $\mathsf{Pr}_s^{\max}(\mathit{Lim}(ZU)) = 1$. We will rely on the following assumptions that can be ensured by some adequate preprocessing. Lemma 7 allows to suppose that $\mathcal{M}$ satisfies the following property (A1):

(A1) Whenever $t \in ZU$ and $P(t, \alpha, t') > 0$ then $t' \in ZU$ and $\mathit{urew}(t, \alpha) = 0$.

Furthermore, we suppose that $t \models \exists \Diamond ZU$ for all states t in $\mathcal{M}$. Hence, there is a scheduler $\mathfrak{S}$ such that from all states t, the limit of almost all $\mathfrak{S}$-paths is a zero-utility end component. In particular:

(A2) $\mathsf{Pr}^{\max}_{\mathcal{M},t}(\Diamond ZU) = 1$ for all states t

Let $\mathfrak{V}$ be a deterministic memoryless scheduler for $\mathcal{M}$ such that $\mathsf{Pr}^{\mathfrak{V}}_{\mathcal{M},t}(\Diamond ZU) = 1$ for all states t and

(A3) $\mathsf{ExpUtil}^{\mathfrak{V}}_{\mathcal{M},t}(\Diamond ZU) \ = \ \mathsf{ExpUtil}^{\min}_{\mathcal{M},t}(\Diamond ZU)$ for all states t

The existence of such a (deterministic and memoryless) scheduler $\mathfrak{V}$ has been shown by de Alfaro [10]. Note that $\mathsf{ExpUtil}^{\mathfrak{V}}_{\mathcal{M},t}(\Diamond ZU)$ is the expected total utility from t under scheduler $\mathfrak{V}$.

Lemma 9. *Suppose assumptions (A1) and (A2) hold. Then, for all $u \in \mathbb{Q}$ and all states s of $\mathcal{M}$:*

$$\sup_{e\in\mathbb{N}} \mathsf{ExpUtil}^{\min}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \quad = \quad \mathsf{ExpUtil}^{\min}_{\mathcal{M},s}(\Diamond ZU)$$

Proof. To prove "$\geqslant$" we consider a deterministic memoryless scheduler $\mathfrak{V}$ minimizing the expected total utility (see (A3)). Then:

$$\begin{aligned}
\mathsf{ExpUtil}^{\min}_{\mathcal{M},s}(\Diamond ZU) \quad &= \quad \mathsf{ExpUtil}^{\mathfrak{V}}_{\mathcal{M},s}(\Diamond ZU) \\
&= \quad \mathsf{ExpUtil}^{\mathfrak{V}}_{\mathcal{M},s}(\mathit{true}) \\
&= \quad \sup_{e\in\mathbb{N}} \mathsf{ExpUtil}^{\mathfrak{V}}_{\mathcal{M},s}(\mathit{energy} \leqslant e) \\
&\geqslant \quad \sup_{e\in\mathbb{N}} \mathsf{ExpUtil}^{\min}_{\mathcal{M},s}(\mathit{energy} \leqslant e)
\end{aligned}$$

The remaining task is to prove "$\leqslant$". Let

$$u^{\min} \quad = \quad \sup_{e\in\mathbb{N}} \mathsf{ExpUtil}^{\min}_{\mathcal{M},s}(\mathit{energy} \leqslant e)$$

For each $e \in \mathbb{N}$ there is a (deterministic) scheduler $\mathfrak{S}_e$ such that

$$\mathsf{ExpUtil}^{\mathfrak{S}_e}_{s}(\mathit{energy} \leqslant e) \quad \leqslant \quad u^{\min}$$

The sequence $(\mathfrak{S}_e)_{e\in\mathbb{N}}$ can be used to generate a scheduler $\mathfrak{S}$ such that for each $k \in \mathbb{N}$ there are infinitely many $e \in \mathbb{N}$ with $\mathfrak{S}(\pi) = \mathfrak{S}_e(\pi)$ for all finite paths π of length at most k. For this scheduler $\mathfrak{S}$ and each $e \in \mathbb{N}$, we have:

$$\begin{aligned}
\mathsf{ExpUtil}^{\mathfrak{S}}_{s}(\mathit{energy} \leqslant e) \quad &= \quad \sup_{k\in\mathbb{N}} \mathsf{ExpUtil}^{\mathfrak{S}}_{s}\big(\ (\mathit{energy} \leqslant e) \wedge (\mathit{steps} \leqslant k)\ \big) \\
&\leqslant \quad u^{\min}
\end{aligned}$$

where *steps* serves as a step counter. Thus:

$$\sup_{e\in\mathbb{N}} \mathsf{ExpUtil}^{\mathfrak{S}}_{s}(\mathit{energy} \leqslant e) \quad \leqslant \quad u^{\min}$$

In particular, $\mathsf{Pr}_s^{\mathfrak{S}}(\mathit{Lim}(UD)) = 0$ by Lemma 8. Therefore, $\mathsf{Pr}_s^{\mathfrak{S}}(\mathit{Lim}(ZU)) = 1$ and:

$$\begin{aligned} \mathsf{ExpUtil}_s^{\mathfrak{S}}(\Diamond ZU) &= \mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{true}) \\ &= \sup_{e \in \mathbb{N}} \mathsf{ExpUtil}_s^{\mathfrak{S}}(\mathit{energy} \leqslant e) \end{aligned}$$

Putting things together, we obtain:

$$\mathsf{ExpUtil}_s^{\min}(\Diamond ZU) \leqslant \mathsf{ExpUtil}_s^{\mathfrak{S}}(\Diamond ZU) \leqslant u^{\min}$$

This yields the claim. ■

Corollary 4 (Infinite universal expectation quantiles). *Under assumptions (A1) and (A2), for each $u \in \mathbb{Q}$, the following statements are equivalent:*

(a) $\mathsf{Qu}_s\big(\forall\,\mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\big) = \infty$

(b) $\sup_{e \in \mathbb{N}} \mathsf{ExpUtil}_s^{\min}(\mathit{energy} \leqslant e) \leqslant u$

(c) $\mathsf{Pr}_s^{\max}(\Diamond ZU) = 1$ *and* $\mathsf{ExpUtil}_s^{\min}(\Diamond ZU) \leqslant u$

Statements (c) in Corollary 2 and Corollary 4 provide criteria to check whether an expectation quantiles is infinite in polynomial time. Hence, we get:

Corollary 5. *The following two problems are in* P*:*

(a) *decide whether* $\mathsf{Qu}_s\big(\exists\,\mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\big) = \infty$

(b) *decide whether* $\mathsf{Qu}_s\big(\forall\,\mathrm{ExpU}_{>u}(\mathit{energy} \leqslant ?)\big) = \infty$

B.3 Treatment of self-loops

We now provide a formal justification for the removal of self-loops with zero reward as mentioned in Sec. 4.2. In this context we are given subsets A and B of the state space S of the MDP $\mathcal{M}$ and we deal with path properties of the form $A\,\mathrm{U}^{\leqslant r} B$. Let $\mathcal{M}' = (S, \mathit{Act}, P')$ be the MDP where the transition probability function P' is defined as follows. For all state-action pairs (t, α) with $\mathit{rew}(t, \alpha) = 0$ and $0 < P(t, \alpha, t) < 1$, let $P'(t, \alpha, t) = 0$ and

$$P'(t, \alpha, t') = \frac{P(t, \alpha, t')}{1 - P(t, \alpha, t)} \quad \text{for } t' \neq t$$

with $P'(\cdot) = P(\cdot)$ in all other cases. The reward function on $\mathcal{M}$ can also be used for $\mathcal{M}'$. The transformation $\mathcal{M} \rightsquigarrow \mathcal{M}'$ is indeed quantile-preserving as the following lemma shows:

Lemma 10 (Soundness of the transformation). *For all states s we have:*

$$\text{(a)}\quad \mathsf{Qu}_s^{\mathcal{M}}(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B)) \;=\; \mathsf{Qu}_s^{\mathcal{M}'}(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B))$$

$$\text{(b)}\quad \mathsf{Qu}_s^{\mathcal{M}}(\forall\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B)) \;=\; \mathsf{Qu}_s^{\mathcal{M}'}(\forall\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B))$$

Proof. We only prove statement (a). Suppose that $\mathfrak{S}$ is a scheduler for the original MDP $\mathcal{M}$ that is optimal for the existential quantile $\mathsf{Qu}_s^{\mathcal{M}}(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B))$ and enjoys condition (*) in Lemma 1. As all paths in $\mathcal{M}'$ are paths in $\mathcal{M}$ and the enabled actions in $\mathcal{M}$ and $\mathcal{M}'$ agree, $Act_{\mathcal{M}}(s) = Act_{\mathcal{M}'}(s)$ for all $s \in S$, the scheduler $\mathfrak{S}$ can also be viewed as a scheduler for $\mathcal{M}'$. We consider the case where $\mathcal{M}$ and $\mathcal{M}'$ differ for a single state-action pair (t, α), the general case with multiple state-action pairs then follows from repeated application of the same arguments.

Let (t, α) be the state-action pair where $\mathcal{M}$ and $\mathcal{M}'$ differ, i.e, with $0 < P(t, \alpha, t) < 1$ and $rew(t, \alpha) = 0$. If $\mathfrak{S}$ never schedules action α for paths ending in t then the probabilities for $A\,\mathrm{U}^{\leqslant r}\,B$ under $\mathfrak{S}$ viewed as scheduler in $\mathcal{M}$ and in $\mathcal{M}'$ are the same for all states s. Suppose now that $\mathfrak{S}(\pi) = \alpha$ for some finite $\mathfrak{S}$-path π with $last(\pi) = t$. Then, all finite paths of the form $\pi\,\alpha\,t\,\alpha\,t\,\alpha\,\ldots\,\alpha\,t$ have the same accumulated reward as π. By (*), they are $\mathfrak{S}$-paths too. Almost all infinite $\mathfrak{S}$-paths in $\mathcal{M}$ that start with π will eventually take a step $t \xrightarrow{\alpha} t'$ with $t \neq t'$, after having taken the self-loop at t finitely often. In $\mathcal{M}$, the probability of the set consisting of all infinite paths of the form

$$\pi\,\alpha\,t\,\alpha\,t\,\alpha\,\ldots\,\alpha\,t\,\alpha\,t'$$

under $\mathfrak{S}$ is $\mathrm{prob}(\pi) \cdot P'(t, \alpha, t')$, which is the probability of the finite path $\pi\,\alpha\,t'$ in $\mathcal{M}'$. Since no other paths are affected by the switch from $\mathcal{M}$ to $\mathcal{M}'$, this yields that for each state s, the value $\mathsf{Pr}_s^{\mathfrak{S}}(A\,\mathrm{U}^{\leqslant r}\,B)$ does not depend on whether we consider $\mathfrak{S}$ as a scheduler for $\mathcal{M}$ or $\mathcal{M}'$. This yields that the existential quantile in $\mathcal{M}$ is less or equal than in $\mathcal{M}'$:

$$\mathsf{Qu}_s^{\mathcal{M}}\big(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B)\big) \;\leqslant\; \mathsf{Qu}_s^{\mathcal{M}'}\big(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B)\big)$$

Vice versa, let us suppose that $\mathfrak{S}'$ is a scheduler for $\mathcal{M}'$. We then consider the scheduler $\mathfrak{S}$ for $\mathcal{M}$ where $\mathfrak{S}(\pi)$ is $\mathfrak{S}'(\pi')$ when π arises from π by erasing all steps $t \xrightarrow{\alpha} t$. Since $rew(t, \alpha) = 0$, the probability $\mathsf{Pr}_s^{\mathfrak{S}}(A\,\mathrm{U}^{\leqslant r}\,B)$ in $\mathcal{M}$ agrees with $\mathsf{Pr}_s^{\mathfrak{S}'}(A\,\mathrm{U}^{\leqslant r}\,B)$ in $\mathcal{M}'$ for all states s. Hence:

$$\mathsf{Qu}_s^{\mathcal{M}}\big(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B)\big) \;\geqslant\; \mathsf{Qu}_s^{\mathcal{M}'}\big(\exists\, \mathrm{P}_{\trianglerighteq p}(A\,\mathrm{U}^{\leqslant ?}\,B)\big)$$

The argument for universal quantiles (statement (b)) is analogous, except that we consider adversarial schedulers. ■

The switch from $\mathcal{M}$ to $\mathcal{M}'$ indeed can simplify the computation of quantiles by our algorithm. This is in particular of interest in combination with the SCC-based decomposition techniques of the LPs $\mathbb{LP}'_r$. If $\mathsf{Pr}^{\max}(A\,\mathrm{U}^{\leqslant r}\,B)$ for all α-successors t' with $t' \neq t$ has already been computed the inequality for state t and α in the LP reduces to $x_t \leqslant c$ for some constant c. There is even no need to consider state t in $\mathbb{LP}'_r$ when the quantiles for all successors different from s are already known.

C Details on our implementation and case studies

In this section, we provide some more details on the case studies carried out to evaluate the performance of our implementation in practice. We start with further details on the self-stabilizing protocol already briefly presented in Sec. 6. Furthermore, we detail two additional case studies from the PRISM benchmark suite [17] and provide details for the energy-aware job-scheduling protocol presented in the main part. Besides addressing the general case of computing quantiles, our implementation also supports the specialized algorithms for qualitative quantiles, i.e., with probability bounds 0 or 1, proposed in [22]. These algorithms only rely on an analysis of the underlying graph-structure of the model.

C.1 Self-stabilization

As detailed in the main part of this paper, we considered the randomized self-stabilizing protocol by Israeli and Jalfon [IJ90]. The PRISM MDP model can be obtained at the URL `http://www.prismmodelchecker.org/casestudies/self-stabilisation.php#ij`. Our main purpose is to show the applicability of our implementation using the BP approach as stated in Sec. 4.2 and compare it to the iterative LP approach of [22] recalled in Sec. 4.1.

The self-stabilizing protocol works as follows: At the beginning, the N processes all are active, i.e., have one token assigned to each of them. Then, the protocol is reassigning tokens step-wise to the processes, where at each step, one active process is selected through a scheduler to randomly pass its token either to its right or left neighbor. If a process has two tokens, the tokens are merged into one. The ring is in a stable state if exactly one process owns the last remaining token. Given N and a probability threshold p, we now considered the following quantiles: "The minimal number of steps required for reaching a stable state with probability of at least p for some schedulers (existential quantile) or all schedulers (universal quantile)." Formally, these quantiles can be stated as

$$\mathsf{Qu}_s\big(\exists\, \mathrm{P}_{\geqslant p}(\Diamond^{\leqslant ?} \mathit{Stable})\big) \text{ and } \mathsf{Qu}_s\big(\forall\, \mathrm{P}_{\geqslant p}(\Diamond^{\leqslant ?} \mathit{Stable})\big),$$

where s is the initial state in which all processes own a token and where Stable is the set of all configurations of the ring with exactly one process owning a token. Using the BP approach, we computed the existential and universal quantiles for $N \in \{3, 6, 10, 15, 20\}$ with probability bounds $p \in \{0.1, 0.2, \ldots, 0.9, 0.95, 0.99\}$. The results of the computations are presented in Fig. 2. Note that the plot almost agrees with the plot for the minimal probability of reaching a stable configuration presented in PRISM's model description[5], where, however, the parameterization is over the number of steps, which is in contrast to our parameterization over the probability bound p.

Comparison to the LP approach. After computing the quantiles, we focused on the comparison to the naïve implementation of the LP approach. In [22], a

[5] see `http://www.prismmodelchecker.org/casestudies/self-stabilisation.php#ij`

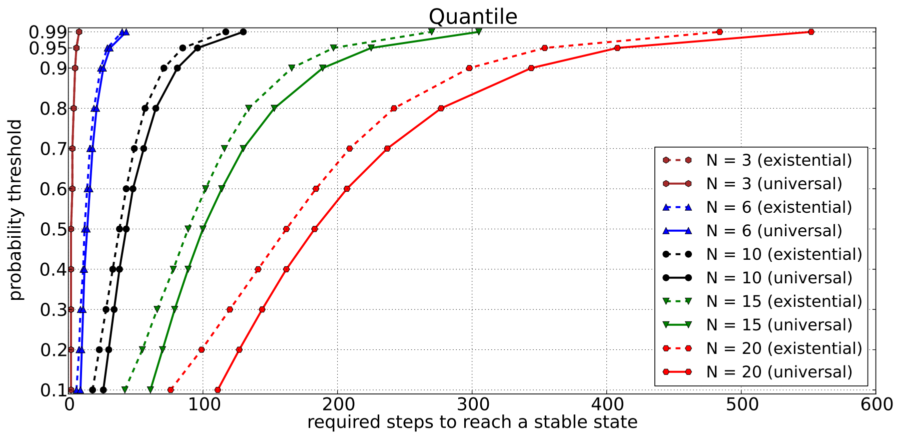


Fig. 2. Results for the self-stabilization algorithm

theoretical upper bound r_{max} for the reward was established that is exponential in the size of the model. For the given bound it is guaranteed that solving $\mathbb{LP}_{r_{\mathsf{max}}}$ yields the desired quantile. However, in our case studies we observed that this theoretical upper bound r_{max} is hardly reached and usually much greater than the quantile values r. For the evaluation of the naïve LP approach we considered $\mathbb{LP}_r$ rather than $\mathbb{LP}_{r_{\mathsf{max}}}$ for the precomputed actual quantile value r, as we know that solving $\mathbb{LP}_r$ is sufficient for computing the desired quantile. We also ignored the computational effort for solving $\mathbb{LP}_{r-1}$, $\mathbb{LP}_{r-2}$, ... that would normally occur in the iterative scheme. For solving the linear program $\mathbb{LP}_r$ we used the LP solver LPSOLVE, version 5.5.2. Although LPSOLVE provides methods for reducing the size of the LP before applying the solver, we turned off these presolving methods, since they turned out to have only marginal impact on the timings. Table 3 shows our results for the BP implementation and compares the time for computing the quantiles with the LP approach. It can easily be seen that the BP approach performs significantly better than the LP approach. Already within $N = 15$ processes, the LP approach exceeded the timeout bound of 12 hours, whereas the BP approach is still applicable with computation times of a few seconds.

C.2 Leader election

Besides reasoning over quantiles with step-bounded until properties, our implementation also allows for quantiles with reward-bounded until properties, required, e.g., for computing energy-utility quantiles. In this section, we investigate quantiles with reward-bounded until properties in two case studies concerning randomized leader-election protocols. Such protocols aim to elect a leader, i.e., a uniquely designated process of N equal processes organized in a ring structure by sending messages to the other processes. We considered a synchronous and

Table 3. Results for randomized self-stabilizing (existential and universal quantile)

		model		existential quantile			universal quantile		
N	p	states	build	result	LP	BP	result	LP	BP
6	0.1	63	0.05s	6	0.06s	<0.01s	9	0.11s	0.03s
	0.5	"	"	12	0.16s	<0.01s	14	0.22s	<0.01s
	0.99	"	"	40	1.44s	<0.01s	43	1.34s	<0.01s
8	0.1	255	0.09s	11	1.44s	<0.01s	16	4.58s	0.09s
	0.5	"	"	24	9.58s	0.01s	26	12.8s	0.01s
	0.99	"	"	74	107.38s	0.03s	81	135.09s	0.04s
10	0.1	1,023	0.24s	18	118.38s	0.03s	26	403.36s	0.16s
	0.5	"	"	38	1,066.64s	0.05s	43	1,388.15s	0.09s
	0.99	"	"	117	11,552.55s	0.14s	130	19,794.61s	0.15s
15	0.1	32,767	1.56s	42	timeout	1.85s	61	timeout	3.78s
	0.5	"	"	89	timeout	3.85s	100	timeout	4.10s
	0.99	"	"	270	timeout	11.42s	305	timeout	12.18s
20	0.1	1,048,575	53.99s	76	timeout	176.77s	111	timeout	264.08s
	0.5	"	"	162	timeout	354.24s	183	timeout	417.28s
	0.99	"	"	484	timeout	1,028.11s	552	timeout	1,219.48s

an asynchronous variant of such a protocol, both developed by Itai and Rodeh [IR90] and also described in the context of probabilistic model checking within the benchmark suite of Prism [17].

Synchronous leader election. The synchronous variant fixes a number K of probabilistic choices, where in each round every process selects an ID from $\{1, ..., K\}$ and passes it synchronously over the ring. If there is a unique maximal ID, the processor which chose this ID is the elected leader – otherwise, a new round starts, where the processes again chose randomly an ID from $\{1, ..., K\}$.

We use the Markov chain model of the synchronous leader-election protocol from `http://www.prismmodelchecker.org/casestudies/synchronous_leader.php`.

Asynchronous leader election. In the asynchronous setting the processes are located in a ring. Initially all processes are active. Inactive processes still pass along messages to their neighbors. The protocol operates in rounds consisting of three phases. In the first phase each active process probabilistically selects its preference, i.e., whether to remain active or to become inactive. Then, a process communicates its preference to the next process along the ring. A process is then allowed to become inactive only if the active process preceding it in the ring prefers to remain active. In a third phase, the processes send a counter around the ring to determine if only one active process remains, which then becomes the leader. Otherwise the protocol proceeds with a further round.

We use the MDP model of the asynchronous leader-election protocol obtained from Prism's benchmark suite, see `http://www.prismmodelchecker.org/casestudies/asynchronous_leader.php`.

Quantile experiments. One of the most significant quantitative measures for both, the synchronous and the asynchronous variant, is the minimal number of rounds r required to elect a leader with a certain probability p for some/all schedulers. Note that in a Markov chain, the probabilities to elect a leader agree for all schedulers, i.e., the minimal number of rounds r is the same for all schedulers in the synchronous protocol variant. As noticed in the PRISM case study, this is also the case for the asynchronous variant, although that model is an MDP. Hence, we only considered the following existential quantile, whose value, however, agrees with the value of the universal quantile:

$$\mathsf{Qu}_s\big(\,\exists\,\mathrm{P}_{\geqslant p}(\Diamond^{\leqslant ?}\,\mathit{LeaderElected})\,\big),$$

interpreted over step-bounded reachability and round-bounded reachability. The figures below show the results of our experiments for $N \in \{3, 4, \ldots, 11\}$ processes and probability bounds $p \in \{0.1, 0.2, \ldots, 0.9, 0.95, 0.99\}$ for the synchronous variant (Fig. 3) assuming $K = 3$ and for the asynchronous variant (Fig. 4). Our

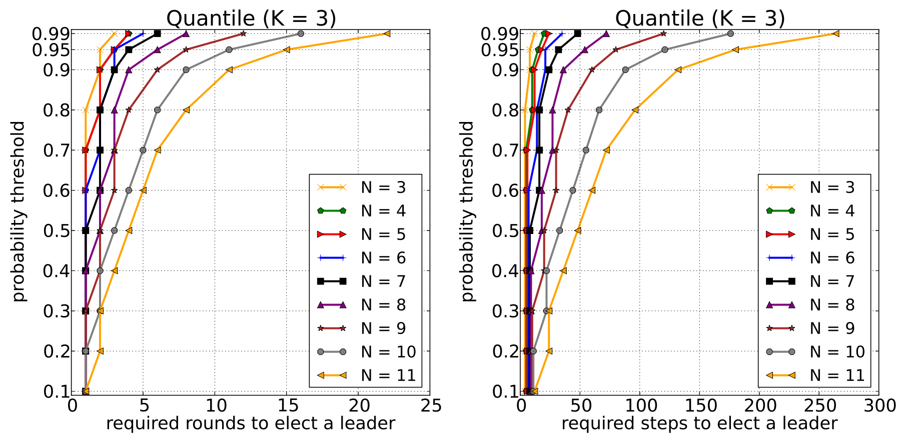


Fig. 3. Results for the synchronous leader election

results were all obtained employing the BP approach. As all cycles have a positive reward in terms of rounds, no additional LP had to be solved. We also considered round-bounded reachability using the LP approach. Table 4 documents our results in terms of rounds required to elect a leader and the times needed to compute the quantile. We fixed a timeout bound of 12 hours and restricted memory consumption to 32GB. As detailed already in the last section, it is not surprising that the LP approach leads to timeouts. In the synchronous case, this was the case for $N \geq 10$ processes. For $N = 9$ processes, the cases where $p = 0.5$ and $p = 0.1$ could be computed, whereas for $p = 0.99$ the computation exceeded the time bound. As the asynchronous model is much larger than the synchronous one, timeouts already arise for $N = 6$ processes. In the asynchronous case with

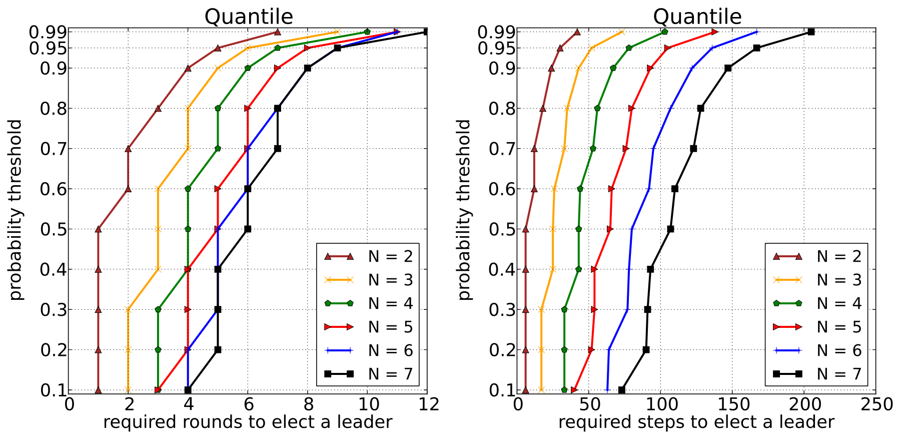


Fig. 4. Results for the asynchronous leader election

Table 4. Results for synchronous and asynchronous leader election

		synchronous					asynchronous				
N	p	states	build	rounds	LP	BP	states	build	rounds	LP	BP
3	0.1	61	0.04s	1	0.02s	0.01s	364	0.09s	2	0.11s	0.06s
	0.5	"	"	1	0.02s	0.01s	"	"	3	0.18s	0.01s
	0.99	"	"	3	0.02s	0.01s	"	"	9	1.04s	0.01s
4	0.1	256	0.08s	1	0.05s	0.03s	3,172	0.33s	3	7.24s	0.25s
	0.5	"	"	1	0.05s	0.02s	"	"	4	14.34s	0.08s
	0.99	"	"	4	0.15s	0.01s	"	"	10	117.46s	0.06s
5	0.1	990	0.16s	1	0.30s	0.08s	27,299	1.01s	3	728.77s	1.09s
	0.5	"	"	1	0.30s	0.01s	"	"	5	3,259.03s	0.80s
	0.99	"	"	4	1.69s	0.01s	"	"	11	28,389.70s	0.85s
6	0.1	3,669	0.31s	1	3.14s	0.21s	237,656	5.92s	4	timeout	11.35s
	0.5	"	"	1	3.13s	0.04s	"	"	5	timeout	10.04s
	0.99	"	"	5	32.47s	0.05s	"	"	11	timeout	11.34s
7	0.1	13,153	0.70s	1	37.84s	0.44s	2,095,783	68.28s	4	timeout	122.52s
	0.5	"	"	1	37.52s	0.13s	"	"	6	timeout	117.71s
	0.99	"	"	6	767.73s	0.15s	"	"	12	timeout	129.66s
8	0.1	45,966	1.40s	1	578.26s	0.90s					
	0.5	"	"	2	1,565.29s	0.55s	memout				
	0.99	"	"	8	18,188.80s	0.88s					

$N \geq 8$, the memory limit was reached during the construction of the explicit model representation.

Note on the implementation of step-bounded until. For quantiles using a step-bounded until, we reuse our implementation for reward-bounded untils by assigning a reward of 1 to each state of the model. As a positive reward is assigned to all states, our BP approach can then be applied without having to solve linear programs for zero-reward cycles.

C.3 Energy-aware job scheduling

The energy-aware job-scheduling protocol sketched in the main part of this paper describes rules for processes to access a shared resource. We provide the operational behavior of the model by means of control-flow graphs representing MDPs: one for the processes (see Fig. 5) and one for the resource (see Fig. 6). We assume a setting of N processes. Similar to the input language of PRISM, such graphs define MDP modules which we denote by $\mathcal{P}_i$ for each process $i = 1, ..., N$ and $\mathcal{R}$ for the resource. Using standard parallel composition [Seg95], these MDP modules yield the MDP semantics $\mathcal{M} = \mathcal{P}_1 \| \cdots \| \mathcal{P}_N \| \mathcal{R}$ of the whole protocol, which is subject of our algorithms applied in this case study.

Operational behavior of the model. Each control-flow graph $\mathcal{P}_i$ of the ith process is described by four different locations $start_i$, $ncrit_i$, $wait_i$ and $crit_i$ (see Fig. 5). Starting in the initial location $start_i$, the process moves to the noncritical location $ncrit_i$, indicating that process i has not requested access to the resource yet. At this transition, the timer t_i is randomly set according to a distribution τ, representing the time where the process stays in $ncrit_i$. Afterwards, it requests access to the shared resource by entering its waiting location $wait_i$ and randomly choosing a deadline counter d_i and a computation timer c_i according to distributions δ and γ, respectively. Intuitively, the timer c_i represents the time required to finish the desired task while process i uses the shared resource, and the deadline counter d_i formalizes the time until the task needs to be finished. Clearly, c_i is required to be smaller than d_i but greater than 0. Waiting for the access to the shared resource, d_i is decreased if it is greater than zero (this is expressed by the operation $d_i := d_i \dot{-} 1$ in Fig. 5, where $\dot{-}$ stands for the positive difference, i.e., $m \dot{-} n = \max(0, m - n)$ for natural numbers $m, n \in \mathbb{N}$). Since the access to the shared resource is exclusive, the decision which process can access the resource is performed by a scheduler, which we will detail later on. The scheduler decides to grant access to process i by setting $user = i$, allowing the process to move to its critical location $crit_i$. The process then performs the task which requires access to the shared resource. In order to meet the deadline, the scheduler may decide to activate a turbo mode ($turbo = \top$), doubling the computation speed of the task. This is modeled by subtracting 2 instead of the standard 1 from the computation timer c_i at every time step in the critical section. After the computation of the task has finished, the process leaves the critical location, enters its noncritical location and the procedure starts over again. All N processes are synchronously composed, i.e., at each step of the protocol, all processes perform exactly one transition in their control-flow graph.

The resource module $\mathcal{R}$ settles the rules to access the shared resource. The control-flow graph of $\mathcal{R}$ is depicted in Fig. 6, where k and i range over all processes, i.e., one transition in the graph involving k or i stands for multiple transitions replacing k and i by $1, \ldots, N$. Synchronously composed with the processes' behavior $\mathcal{P}_1 \| \cdots \| \mathcal{P}_N$, the resource behavior depends on the processes (guards $c_i = 0$ indicating that process i leaves the critical section and $wait_k$ standing for process k's request to enter the critical section) and the processes

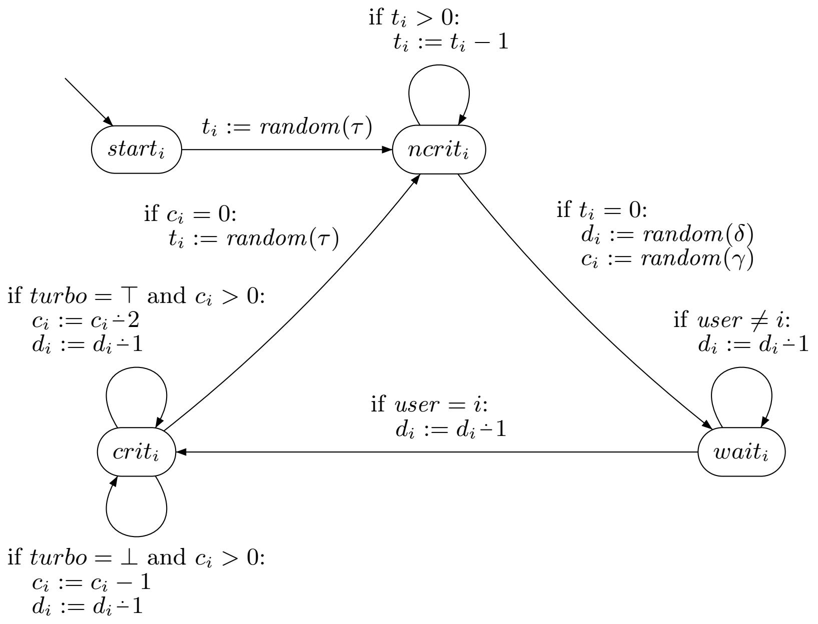
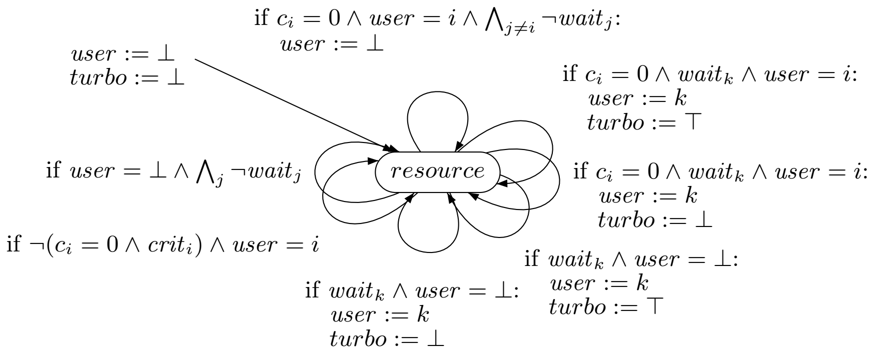


Fig. 5. Control flow for a process (energy-aware job scheduling)

depend on the resource (the guards $user = i$ if the process i has the right to be in the critical section and *turbo* standing for the case whether the turbo mode is activated (formally, $\top$) or deactivated ($\bot$)). Whereas the processes' behavior is deterministic, the model of the resource involves nondeterminism, e.g., if multiple processes are in its waiting location or for switching on the turbo mode.

At the beginning, the resource is not in use, meaning that *user* is undefined and the turbo mode is deactivated (both are set to $\bot$). Whenever there is a process k waiting (i.e., k is in location $wait_k$) and no process has access to the resource ($user = \bot$), the resource can be assigned to the waiting process by setting $user := k$. In this turn, it is also decided whether to activate the turbo mode or not. As long as the computation counter c_i of the process i in the critical section is set, i.e., the task of process i is still computed, the resource does not change its parameters. Only if the computation is finished ($c_i = 0$ and $user = i$), then either the resource is directly handed to another waiting process k, also activating or deactivating the turbo mode, or the user of the resource is set undefined if no process is in its waiting location.

Energy, utility and model parameters. For our case study, we fixed certain parameters while parameterizing over the number of processes N. We provide the probability distributions for each process in the form $(p{:}n, 1{-}p{:}m)$, where $p \in [0, 1]$ is the probability of value n and $1 - p$ the probability that value m occurs. In our setting, we fixed a deadline distribution $\delta = (\frac{1}{3}{:}7, \frac{2}{3}{:}9)$, a computation distribution $\gamma = (\frac{1}{3}{:}2, \frac{2}{3}{:}3)$ and a timer distribution $\tau = (\frac{1}{3}{:}5, \frac{2}{3}{:}4)$.



if $c_i = 0 \wedge user = i \wedge \bigwedge_{j \neq i} \neg wait_j$: $user := \bot$
$user := \bot$ $turbo := \bot$
if $c_i = 0 \wedge wait_k \wedge user = i$: $user := k$ $turbo := \top$
if $user = \bot \wedge \bigwedge_j \neg wait_j$
$resource$
if $c_i = 0 \wedge wait_k \wedge user = i$: $user := k$ $turbo := \bot$
if $\neg(c_i = 0 \wedge crit_i) \wedge user = i$
if $wait_k \wedge user = \bot$: $user := k$ $turbo := \top$
if $wait_k \wedge user = \bot$: $user := k$ $turbo := \bot$

Fig. 6. Control flow for the resource (energy-aware job scheduling)

The energy consumption and the achieved degree of utility are represented using reward structures over the MDP $\mathcal{M}$, which arises from synchronous parallel composition of the models for processes and the resource model. Utility is given as the number of successfully finished tasks, i.e., a transition reward of 1 is granted if some process i takes a transition from $crit_i$ to $ncrit_i$ and the deadline counter d_i is greater than 0. Energy is modeled by state rewards, where the global energy consumption in a state of $\mathcal{M}$ is the sum of the energy consumption of all processes in the respective local states. Depending on the local state of a process, the consumed energy differs as detailed in Table 5. In order to model

Table 5. Rewards modeling the energy consumption of a process

location	energy consumption			
process i	$i \mod 3 \neq 0$		$i \mod 3 = 0$	
	$turbo = \bot$	$turbo = \top$	$turbo = \bot$	$turbo = \top$
$ncrit_i$	2	2	4	4
$wait_i$	1	1	2	2
$crit_i$	3	9	6	18

a heterogeneous architecture, we decided to double the energy consumption of every third process. The scheduler has to take this into account, e.g., by deciding whether to allow jobs for the process that has more energy consumption to run immediately or to postpone them as long as possible. For the turbo mode, the energy consumption is tripled while the computation speed is doubled. Overall, the energy consumption accumulated in the critical section with active turbo mode is thus 50% higher than in the case without turbo mode, as the critical section is left faster than usual as well.

Reasoning about the energy-utility trade-off. We analyzed the trade-off between energy and utility by means of the following two values:

– The minimal amount of energy $e_{\min}$ required to exceed a given utility bound u with at least a probability of p.

– The maximal utility $u_{\max}$ obtained by one process such that it consumes less than e energy within a probability greater than p.

Formally, these values can be expressed as the following existential energy-utility upper-bound quantiles:

$$\begin{aligned} e_{\min} &= \mathsf{Qu}_s\big(\exists \mathrm{P}_{\geqslant p}(\Diamond^{\leqslant ?}(\mathit{utility} > u))\big) \\ &= \min\big\{\ \hat{e} \in \mathbb{N} : \mathsf{Pr}_s^{\max}(\ \Diamond(\mathit{energy} \leqslant \hat{e}\ \wedge\ \mathit{utility} > u)\ ) \geqslant p\ \big\} \end{aligned}$$

$$\begin{aligned} u_{\max} &= \mathsf{Qu}_s\big(\exists \mathrm{P}_{> p}(\Box^{\leqslant ?}(\mathit{energy}_1 < e))\big) \\ &= \max\big\{\ \hat{u} \in \mathbb{N} : \mathsf{Pr}_s^{\max}(\ \Box(\mathit{utility}_1 \leqslant \hat{u} \implies \mathit{energy}_1 < e)\ ) > p\ \big\} \end{aligned}$$

In these formulas, *utility* and *energy* are the accumulated utility and energy rewards of all processes, whereas $\mathit{utility}_1$ and energy_1 stand for the accumulated utility and energy of process 1 only. Hence, $e_{\min}$ describes a "global" quantile, whereas $u_{\max}$ stands for a "local" one. Considering the local accumulated rewards only w.r.t. process 1 and not w.r.t. another process is w.l.o.g., since process 1 always consumes minimal energy compared to the processes i with $i \mod 3 = 0$ (see Table 5). Note that $u_{\max}$ differs from the lower-bound quantile motivated in the main part of the paper, but provides a reasonable measurement for the energy-utility trade-off (in fact, concerning our model the values differ only in corner cases).

Empirical results. We modeled our energy-aware job-scheduling protocol in Prism, encoding the program graphs detailed in Fig. 5 and Fig. 6 in the guarded-command input language of Prism. Since both energy and utility are provided as reward structures in $\mathcal{M}$, we employed the automata-based approach detailed in Sec. 4.4, e.g., to encode the utility threshold u into the states. This yields a model $\mathcal{M}_u$ on which the quantile $e_{\min}$ can be computed using our implementation for upper-bound quantiles. The same approach is used for computing the quantile $u_{\max}$ in a model $\mathcal{M}_e$, which arises from encoding the energy threshold e into the states of $\mathcal{M}$.

The figures below document the results of the upper-bound quantiles detailed above and computed for the energy-aware job scheduling protocol, parameterizing over the number N of processes. In Fig. 7, $e_{\min}$ for $\mathcal{M}_u$ with $N = 2, ..., 5$ processes is shown with a fixed utility threshold of $u = N$. The gap between the curves for $N = 2$ and $N = 3$ is greater than the one between the curves for $N = 3$ and $N = 4$, which can be explained through the higher energy consumption of the third process (see Table 5).
Fig. 8 depicts $u_{\max}$ for $N = 2, ..., 7$ processes and an energy threshold of $e = 50 \cdot N$. It can easily be seen that the plots in both figures have the same shape as the figures contained in the introduction of the main paper: Fig. 7 clearly shows increasing quantiles for the energy budget and Fig. 8 represents decreasing quantiles for the gained utility.

In Table 6, we show the statistical data of our case study. As in the main part of this paper, all calculations were carried out on a computer equipped with two Intel E5-2680 8-core CPUs at 2.70 GHz with 384GB of RAM. We furthermore fixed a timeout within 12 hours and a memory bound of 32GB for each computation.

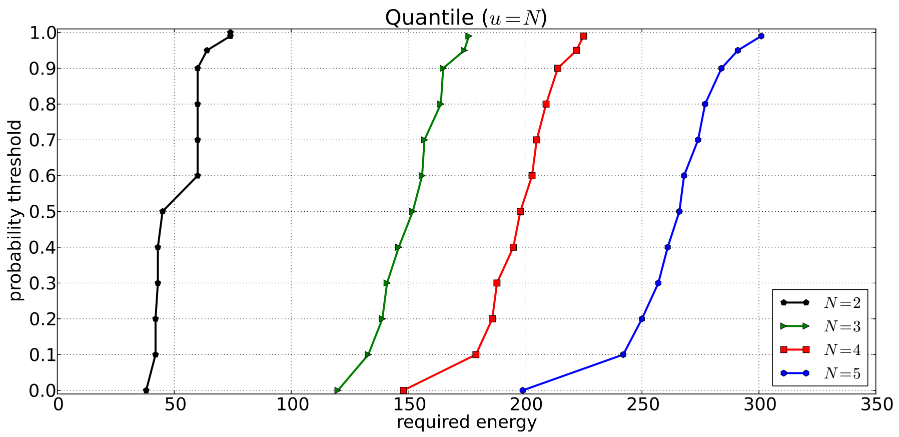


Fig. 7. Results for the energy consumption until N utility is generated

The results for quantitative quantiles (i.e., for probability bound $0 < p < 1$) were obtained using our implementation of the BP algorithms presented in this paper, whereas for qualitative ones (i.e., $p = 0$ or $p = 1$) we used our implementation of the graph-based algorithms presented in [22]. We could not get further than $N = 5$ processes for computing $e_{\min}$ and not more than $N = 7$ processes for computing $u_{\max}$, due to exceeding the memory bound. Notably, $\mathcal{M}_e$ and $\mathcal{M}_u$ scale similar depending on N, although the modeling of energy is more fine-grained than the modeling of utility. However, we only considered local energy rewards encoded into $\mathcal{M}_e$. We also checked for a global variant of $u_{\max}$, requiring

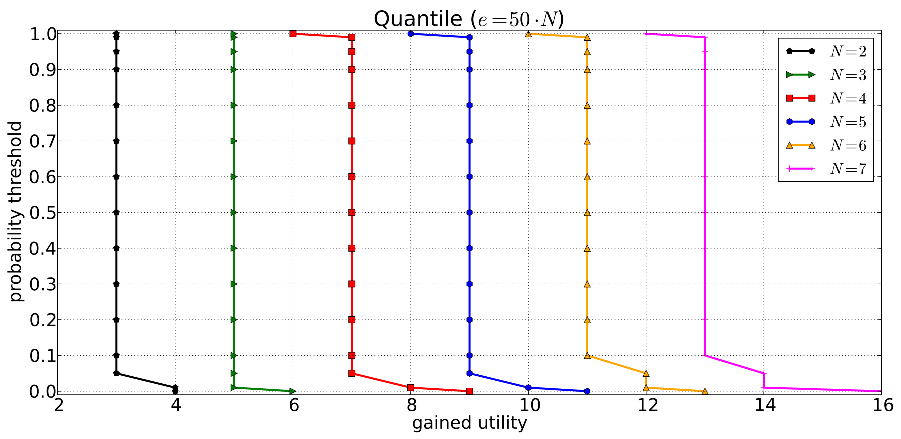


Fig. 8. Results for the gained utility until $50 \cdot N$ energy units are consumed

Table 6. Results for energy-aware job scheduling (quantiles $e_{\min}$ and $u_{\max}$)

		model $\mathcal{M}_u$		quantile $e_{\min}$				model $\mathcal{M}_e$		quantile $u_{\max}$	
N	p	states	build	result	time	N	p	states	build	result	time
2	0	917	0.21s	38	0.08s	4	0	872,410	14.47s	9	255.99s
	0.1	"	"	42	0.12s		0.1	"	"	7	173.71s
	0.5	"	"	45	0.02s		0.5	"	"	7	173.22s
	0.99	"	"	74	0.04s		0.99	"	"	7	155.66s
	1	"	"	74	0.03s		1	"	"	6	212.45s
3	0	16,341	0.91s	120	0.47s	5	0	3,049,471	65.69s	11	1,433.91s
	0.1	"	"	133	1.09s		0.1	"	"	9	812.19s
	0.5	"	"	152	1.08s		0.5	"	"	9	812.93s
	0.99	"	"	176	1.14s		0.99	"	"	9	736.93s
	1	"	"	∞	0.16s		1	"	"	8	1,048.28s
4	0	368,521	14.67s	148	8.89s	6	0	7,901,694	196.68s	13	5,235.94s
	0.1	"	"	179	37.43s		0.1	"	"	11	2,769.04s
	0.5	"	"	198	37.02s		0.5	"	"	11	2,782.04s
	0.99	"	"	225	42.69s		0.99	"	"	11	2,525.76s
	1	"	"	∞	4.57s		1	"	"	10	4,109.23s
5	0	6,079,533	377.95s	199	189.19s	7	0	17,037,097	637.19s	16	16,392.71s
	0.1	"	"	242	1,058.48s		0.1	"	"	13	6,936.23s
	0.5	"	"	266	1,135.65s		0.5	"	"	13	6,900.29s
	0.99	"	"	301	1,261.89s		0.99	"	"	13	6,247.79s
	1	"	"	∞	101.09s		1	"	"	12	12,456.47s

global energy rewards to be encoded into $\mathcal{M}$, which yield a model exceeding our memory constraints already within $N = 4$. Hence, we decided to go for the local variant as presented above and dropped the global version.

Further references for the appendix

CY95. C. Courcoubetis and M. Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42(4):857–907, 1995.

IJ90. A. Israeli and M. Jalfon. Token management schemes and random walks yield self-stabilizating mutual exclusion. In *PODC'90*, pages 119–131. ACM, 1990.

IR90. A. Itai and M. Rodeh. Symmetry breaking in distributed networks. *Information and Computation*, 88(1), 1990.

Seg95. Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, Massachusetts Institute of Technology, 1995.