Probabilistic Model Checking and Non-standard Multi-objective Reasoning*

Christel Baier, Clemens Dubslaff, Sascha Klüppelholz, Marcus Daum, Joachim Klein, Steffen Märcker, and Sascha Wunderlich

Institute for Theoretical Computer Science
Technische Universität Dresden, Germany

Abstract. Probabilistic model checking is a well-established method for the automated quantitative system analysis. It has been used in various application areas such as coordination algorithms for distributed systems, communication and multimedia protocols, biological systems, resilient systems or security. In this paper, we report on the experiences we made in inter-disciplinary research projects where we contribute with formal methods for the analysis of hardware and software systems. Many performance measures that have been identified as highly relevant by the respective domain experts refer to multiple objectives and require a good balance between two or more cost or reward functions, such as energy and utility. The formalization of these performance measures requires several concepts like quantiles, conditional probabilities and expectations and ratios of cost or reward functions that are not supported by state-of-the-art probabilistic model checkers. We report on our current work in this direction, including applications in the field of software product line verification.

1 Introduction

Probabilistic phenomena occur rather naturally for many types of hardware- and software systems. Typical examples are randomized algorithms for the coordination of distributed systems, such as mutual exclusion, leader election or consensus protocols where coin-tossing actions are used to break symmetry, or systems with unreliable or only partially known components where stochastic distributions can be used to model the system load or the frequency of failures (e.g. message losses, bit flips in hardware components). Various models and formal methods for the analysis of probabilistic systems have been proposed in the literature. We focus here on *probabilistic model-checking* (PMC) on Markovian models, which can be seen as automata annotated with probabilistic distributions and cost or reward functions modeling resource requirements. The Markovian property that the future system behavior only depends on the current state

* The authors are supported by the DFG through the collaborative research centre HAEC (SFB 912), the cluster of excellence cfAED, Deutsche Telekom Stiftung, the ESF young researcher group IMData (100098198), the Graduiertenkolleg QuantLA (1763) the DFG/NWO-project ROCKS, and the EU-FP-7 grant MEALS (295261).

S. Gnesi and A. Rensink (Eds.): FASE 2014, LNCS 8411, pp. 1–16, 2014.

but not on the history makes these models best-suited for algorithmic quantitative analysis. Whereas Markov chains (MCs) are purely probabilistic and can be seen as transition systems where the probabilities are attached to the outgoing transitions of each state, Markov decision processes (MDPs) support both nondeterministic and probabilistic choices. This is useful for modeling randomized distributed protocols where the nondeterminism models the interleaving of independent (possibly randomized) actions executed in parallel. The nondeterminism in MDPs can be resolved by schedulers, allowing to reason about extremal probabilities. The typical task of PMC on a given MDP is to compute the maximal or minimal probabilities of path properties specified by some formula of linear temporal logic (LTL) [37,18], the path-formula fragment of probabilistic computation tree logic (PCTL) or its variant PRCTL with reward-bounded temporal modalities [27,9,19,2]. Algorithms for Markovian models and LTL- or PRCTL-specifications were implemented in various model checkers, such as Prism [29] and MRMC [31]. They provide several engines with sophisticated techniques to tackle the state-explosion problem and have been continuously extended by new features and were successfully applied in various areas, such as randomized distributed systems, multimedia, security protocols and systems biology.

In current inter-disciplinary research projects, where we apply (among others) PMC for the analysis of low-level resource-management algorithms, we made a series of interesting observations concerning the strengths and limitations of state-of-the-art PMC-techniques. Within these projects, the PMC-based approach is complementary to the measurement- and simulation-based analysis conducted by project partners to provide insights in the energy-utility, reliability and other performance characteristics from a global and long-run perspective. The evaluation results obtained by a probabilistic model checker guide the optimization of resource-management algorithms. They can be useful to predict the performance of management algorithms on future hardware or low-level protocols that have not been implemented yet, making measurements are impossible. We successfully applied PMC, e.g., for the analysis of a spinlock protocol [5], a lock-free synchronization protocol for read-write problems [6], a bonding network device [22] and an energy-aware job scheduling scenario [4].

The application of PMC was, however, not straightforward. Besides the expected state-explosion problem, difficulties arose to find appropriate probabilistic distributions to model cache-effects and other hardware details. These problems have been addressed in [5] by means of a simple spinlock example. To our surprise, our case studies revealed the lack of performance measures that have been identified as most significant by our cooperation partners, but were not supported by existing probabilistic model checkers. This mainly concerns the calculation of measures that provide insights in the tradeoff between multiple cost and reward functions, such as energy and utility. Usually, the gained utility increases (within certain bounds) with the price to be payed. For example, the performance of a CPU (measured, e.g., in the maximum number of instructions per second) crucially depends on its frequency, but so does its energy consumption. The tradeoff is now to maximize the performance, while at the same time minimizing the

energy consumption. In the described setting, one has the freedom to select the frequency the CPU is operating with. The maximal number of instructions per second then appears as a consequence of the choice for the frequency. The goal is now to find "optimal" solutions for the above tradeoff. In requirement specification it is very natural to introduce lower bounds for the gained utility and upper bounds for the costs. One could either ask for the minimal amount of energy to be spend given a lower bound on the performance, or for the maximal performance given a certain upper bound on the energy consumption. Other examples for low-cost objectives and high quality-of-service objectives are constraints on the maximal time to recover from failures in a resilient system or the penalty to be paid for missed deadlines, the average amount of time that a process has to wait for a requested resource, or the frame rate of a video platform. Assuming such a given cost function for the energy consumption and a reward function for the achieved degree of utility, we consider the design of algorithms answering multi-objective problems for MDPs exemplified by the following tasks:

(M) Given an energy budget e and a utility threshold u, find a scheduler for completing a task such that the expected achieved utility is at least u and there is a 80% chance that the energy consumption is at most e.

(Q) Given an energy budget e, maximize the utility value that can be guaranteed for the completion of a task when consuming e or less energy with probability 0.8.

(Qe) Find a scheduler requiring a minimal energy budget to ensure the expected utility for completing a task to be larger than a given utility threshold u.

(C) Suppose the total energy consumption for completing a task is at most e. Find a scheduler maximizing the probability for the property stating that the utility value achieved by completing a task is at least u.

(Cq) Find a scheduler maximizing utility in terms of completed tasks, such that the probability for utility being at least u under the assumption that the energy consumed does not exceed a given threshold e is greater than 0.8.

(R) Find a scheduler for completing infinitely many tasks such that almost surely the ratio between the achieved utility and the consumed energy is always greater than a given quality threshold.

(Rc) Under the assumption that the consumed energy is always smaller than a given threshold e, find a scheduler which guarantees almost surely the energy-utility ratio being greater than a quality threshold.

The first task **(M)** can be seen as a standard multi-objective query [14,23], where the task is to check the existence of a scheduler satisfying a probability condition and an expectation condition. This type of multi-objectives is not in the scope of this paper, which addresses non-standard multi-objectives exemplified by all the other tasks. **(Q)** and **(Qe)** can be formalized as *quantiles*, where the latter stands for an *expectation quantile*. **(C)** is an instance of *conditional probabilities* and **(Cq)** describes a *conditional quantile*. Although quantiles and conditional probabilities and expectations are standard concepts in mathematics and statistics, they have drawn very few attention in the context of PMC. Our recent work [36,4] shows how quantiles can be derived from computation

schemes for the probabilities or expectations of reward-bounded path properties. A brief summary will be provided in Section 3. Explanations on the computation of conditional probabilities are provided in Section 4, based on our recent work [8]. Tasks **(R)** and **(Rc)** refer to the quotient of two cost or reward functions in an MDP (e.g., one for the energy and one for the utility). There has been some work on such ratios, e.g., by [1,38] for expected ratios or [20] for long-run ratios when the denominator has the purpose of a counter. This work does not seem to be adequate for solving the tasks stated above. Instead, we show in Section 5 that instances of **(R)** are reducible to problems that have been studied for probabilistic energy games [12] or probabilistic push-down systems [11]. Combining these results with methods presented in Section 4 for conditional probabilities, yields solutions for **(Rc)**. We illustrate how the exemplified tasks stated above can be used in the field of software product line verification. Based on our recent work [22], we detail its application to the energy-aware server system eBond+.

2 Theoretical Foundations

The reader is supposed to be familiar with ω-automata and temporal logics. See, e.g., [15,25,7]. At several places, we will use notations of linear temporal logics (LTL) and computation tree logic (CTL) where $\Diamond$, $\Box$, $\bigcirc$, U and R stand for the temporal modalities "eventually", "always", "next", "until" and "release", while $\exists$ and $\forall$ are used as CTL-like path quantifiers. The notion *path property* will be used for any language consisting of infinite words over 2^{AP}, where AP is the underlying set of atomic propositions. LTL-formulas are often identified with such path properties that are models for the formulas. Having in mind temporal logical specifications, we use the logical operators $\vee$, $\wedge$, $\neg$ for union, intersection and complementation of path properties.

In the remainder of this section, we provide a brief summary of our notations for Markov decision processes and related concepts. For further details we refer to textbooks on model checking [15,7] and on probability theory and Markovian models [33,32,28]. Let S be a nonempty, countable set. A distribution on S is a function $\mu : S \to [0,1]$ with $\sum_{s \in S} \mu(s) = 1$.

Markov Decision Processes (MDPs). MDPs can be seen as a generalization of transition systems where the operational behavior in a state s consists of a nondeterministic selection of an enabled action α, followed by a probabilistic choice of the successor state, given s and α. Formally, an MDP is a tuple $\mathcal{M} = (S, Act, P, AP, L)$ where S is a finite set of states, Act a finite set of actions, AP a finite set of atomic propositions and $L : S \to 2^{AP}$ a labeling function. The enabled actions and transition probabilities are specified by a function $P : S \times Act \times S \to [0,1] \cap \mathbb{Q}$ with $\sum_{s' \in S} P(s, \alpha, s') \in \{0, 1\}$ for all $s \in S$ and $\alpha \in Act$.

The triples (s, α, s') where $P(s, \alpha, s') > 0$ are called transitions. We write $Act(s)$ for the set of actions that are enabled in s, i.e., $P(s, \alpha, s') > 0$ for some $s' \in S$. For technical reasons, we require that $Act(s) \neq \varnothing$ for all states s. If the sets $Act(s)$ are singletons for all states s, $\mathcal{M}$ is called a Markov chain.

Paths in $\mathcal{M}$ are finite or infinite alternating sequences $\zeta = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2\ldots$ of states and actions built by consecutive transitions, i.e., $P(s_{i-1},\alpha_{i-1},s_i)>0$ for all $i \geqslant 1$. The length $|\zeta|$ denotes the number of transitions in π. If $k \leqslant |\zeta|$ then $\zeta[k] = s_k$ denotes the $(k{+}1)$-st state in ζ. $\mathit{FinPaths}(s)$ and $\mathit{InfPaths}(s)$ stand for the sets of all finite resp. infinite paths starting in s. If $\zeta = s_0\,\alpha_0\,s_1\,\alpha_1\,s_2\,\alpha_2\ldots$ is an infinite path then its trace $\mathit{trace}(\zeta) = L(s_0)\,L(s_1)\,L(s_2)\ldots \in (2^{AP})^\omega$ is obtained by the projection to the state labels.

Schedulers and Probability Measure. Reasoning about probabilities for path properties in an MDP $\mathcal{M}$ requires the selection of an initial state and the resolution of the nondeterministic choices between the possible transitions. This is formalized via *schedulers*, which take as input a finite path and select an action to be executed. For the purposes of this paper, we define schedulers as functions $\mathfrak{S} : \mathit{FinPaths} \to \mathit{Act}$ such that $\mathfrak{S}(\pi) \in \mathit{Act}(\mathit{last}(\pi))$ for all finite paths π, where $\mathit{last}(\pi)$ denotes the last state of π. For a pointed MDP $(\mathcal{M},s)$, i.e., an MDP as before with some distinguished initial state $s = s_{init} \in S$, the behavior of $(\mathcal{M},s)$ under $\mathfrak{S}$ is purely probabilistic. The probability measure $\Pr_s^{\mathfrak{S}}$ for measurable sets of the infinite $\mathfrak{S}$-paths starting in s is defined in the standard way (see, e.g, [7]) and yields the probability for a path property φ under $\mathfrak{S}$ starting in s:

$$\Pr_s^{\mathfrak{S}}(\varphi) \;\stackrel{\text{def}}{=}\; \Pr_s^{\mathfrak{S}}\big\{\,\zeta \in \mathit{InfPaths}(s) \;:\; \zeta \models \varphi\,\big\}$$

For a worst-case analysis of a system modeled by a pointed MDP $(\mathcal{M},s)$, one ranges over all schedulers (i.e., all possible resolutions of the nondeterminism) and considers the maximal or minimal probabilities for φ:

$$\Pr_s^{\min}(\varphi) \;\stackrel{\text{def}}{=}\; \min_{\mathfrak{S}} \Pr_s^{\mathfrak{S}}(\varphi) \qquad\qquad \Pr_s^{\max}(\varphi) \;\stackrel{\text{def}}{=}\; \max_{\mathfrak{S}} \Pr_s^{\mathfrak{S}}(\varphi)$$

Automata-Based PMC for LTL-Specifications. Figure 1 sketches the main steps of the automata-based PMC-approach for MDP against LTL-specifications. Maxima and minima are taken over all potential resolutions of the nondeterminism, formalized by schedulers. We suppose here that the formula φ describes the undesired behaviors where the requirement does not hold, in which case the maximal probability for φ and a corresponding scheduler that maximizes the probabilities for φ provides insights in the worst-case scenarios.

The idea is to apply at first known algorithms that transform the given LTL-formula into a deterministic automaton $\mathcal{A}$ over infinite words (see [25]) and then to compute the maximimal probabilities for the paths satisfying $\mathcal{A}$'s acceptance condition in the product-MDP $\mathcal{M} \otimes \mathcal{A}$. The latter reduces to a probabilistic reachability problem and is solvable by linear-programming techniques [9,7].

A worst-case analysis as in Fig. 1 is adequate if the choices between the nondeterministic alternatives in the given MDP are uncontrollable (e.g. if they represent the possible interactions with an unknown or unpredictable environment). Likewise, if the given LTL-formula φ formalizes the desired behaviors, the computation of the maximal probability for φ can be seen as a best-case analysis. Then, a scheduler maximizing the probability for φ serves as an optimal controller for the objective φ.

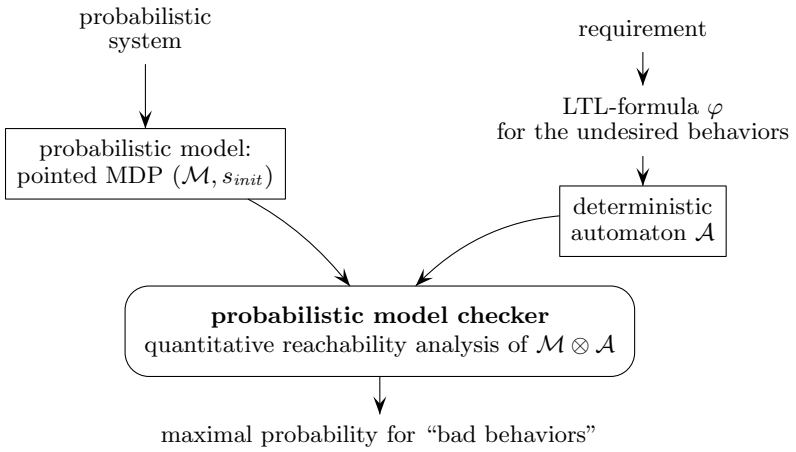


Fig. 1. Automata-based PMC-approach for MDP and LTL-specifications

Weight and Reward Functions. A weight function for $\mathcal{M}$ is a function of the form $wgt : S \times Act \to \mathbb{Z}$ that assigns an integer for all state-action pairs where $wgt(s, \alpha) = 0$ if $\alpha \notin Act(s)$. If wgt is non-negative, i.e., $wgt(s, s') \geqslant 0$ for all states s, s', then we refer to wgt as a *reward function*. We say wgt is positive if $wgt(s, \alpha) > 0$ for all state-action pairs (s, α) where α is enabled in s. Occasionally, we also consider weight functions with rational values and refer to them as *rational-valued* weight functions. The *accumulated weight* of finite paths is defined as $wgt(s_0\, \alpha_0\, s_1\, \alpha_1 \ldots \alpha_{n-1}\, s_n) = \sum_{0 \leqslant i < n} wgt(s_i, \alpha_i)$.

Expected Accumulated Reward. Let χ be a predicate for finite paths and *rew* a reward function. The random variable $rew_\chi : InfPaths \to \mathbb{Z} \cup \{\infty\}$ assigns to each infinite path ζ the accumulated reward of the longest prefix of ζ where χ does not hold. That is:

$$rew_\chi(\zeta) \stackrel{\text{def}}{=} \sup\big\{\, rew(\zeta[0 \ldots k]) \,:\, k \in \mathbb{N},\, \zeta[0 \ldots k]) \not\models \chi \,\big\}$$

We will use two types of predicates for finite paths: reachability constraints $\chi = \text{Reach}(goal)$ where *goal* is a state predicate (i.e., Boolean combination of atomic propositions) and predicates χ of the form $rew > r$ or $rew \geqslant r$ imposing a lower bound on the accumulated reward (where $r \in \mathbb{N}$). If π is a finite path of length n then $\pi \models \text{Reach}(goal)$ if $\pi[k] \models goal$ for some $k \in \{0, 1, \ldots, n\}$ and $\pi \models rew > r$ if $rew(\pi) > r$. For these types of predicates, the set of finite paths π with $\pi \models \chi$ is prefix-closed and the set $\Diamond\chi$ of infinite paths ζ with $\zeta[0 \ldots k] \models \chi$ for some position $k \in \mathbb{N}$ is measurable. Obviously, reachability predicates Reach(*goal*) can be mimicked by the predicate $rew \geqslant 1$ where *rew* is a fresh reward function with $rew(s, \alpha) = 1$ if $s \models goal$ and $\alpha \in Act(s)$ and $rew(s, \alpha) = 0$ in all other cases.

Given state s of $\mathcal{M}$ and a scheduler $\mathfrak{S}$ such that $\Pr_s^{\mathfrak{S}}(\Diamond\chi) = 1$, the *expected accumulated reward* until χ, denoted $\mathbb{E}[rew]_s^{\mathfrak{S}}(\Diamond\chi) = \sum_\pi \text{prob}^{\mathfrak{S}}(\pi)$ is the

expectation of the random variable rew_χ. Here, π ranges over all finite $\mathfrak{S}$-paths with $\pi \not\models \chi$. If $\Pr_s^{\min}(\Diamond\chi) = 1$, then

$$\mathbb{E}[rew]_s^{\max}(\Diamond\chi) = \max_{\mathfrak{S}} \mathbb{E}[rew]_s^{\mathfrak{S}}(\Diamond\chi), \quad \mathbb{E}[rew]_s^{\min}(\Diamond\chi) = \min_{\mathfrak{S}} \mathbb{E}[rew]_s^{\mathfrak{S}}(\Diamond\chi)$$

are computable using linear-programming techniques [33,21].

3 Quantiles

Quantiles are well-established in statistics (see, e.g., [34]), where they are used to reason about the cumulative distribution function of a random variable R. If $p \in]0,1[$, then the p-quantile is the maximal value r such that the probability for the event $R > r$ is at least p. 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 model-checking community. We provide here a brief summary of the concepts presented in [36,4]. The formula

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

states that eventually a goal state will be reached along some finite path where the accumulated energy is at most e and the accumulated utility value is at least u. Path properties $\varphi[e]$ (and $\psi[u]$) parametrizing only over the energy costs (utility reward, respectively) are obtained from $\phi_{e,u}$ by fixing the maximal energy costs e (minimal utility u, respectively). Whereas $\varphi[e]$ is *increasing* with the available energy budged e, $\psi[u]$ is *decreasing* with the requested utility u.

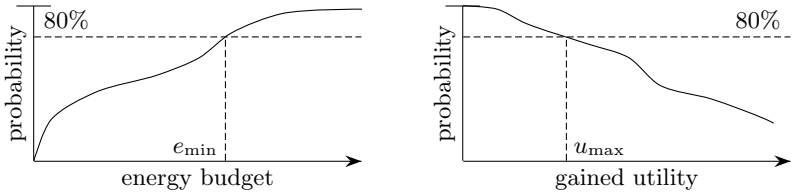


Fig. 2. Quantiles for increasing (left) and decreasing (right) properties

Quantiles now ask for the minimal e (maximal u) such that the probability of all paths starting in a designated state and fulfilling $\varphi[e]$ ($\psi[u]$, respectively) exceed a given probability bound p. The arising quantile values are illustrated in Figure 2 for $p = 0.8$. In order to formally define the quantiles used throughout this paper, let us fix an MDP $\mathcal{M}$ and a reward function $rew : S \times Act \to \mathbb{N}$ as in Section 2. Given an increasing path property $\varphi[r]$, where parameter $r \in \mathbb{N}$ stands for some bound on the accumulated reward, we can define the following types of *existential quantiles*, where $\psi[r] = \neg\varphi[r]$, $\trianglerighteq \in \{\geqslant, >\}$ and $p \in [0,1] \cap \mathbb{Q}$:

$$\min\big\{\, r \in \mathbb{N} \,:\, \Pr_s^{\max}\big(\varphi[r]\big) \trianglerighteq p \,\big\} \quad \text{and} \quad \max\big\{\, r \in \mathbb{N} \,:\, \Pr_s^{\max}\big(\psi[r]\big) \trianglerighteq p \,\big\}$$

Analogously, *universal quantiles* are defined by considering the probability of a path property which can be guaranteed under every scheduler, i.e., replacing $\Pr_s^{\max}$ above by $\Pr_s^{\min}$. If the extrema are taken over the empty set, they are defined to be ∞ in the case of minima and as undefined in the case of maxima. One example is reward-bounded path properties. For instance, $\varphi[r] = \Diamond^{\leqslant r} a$ and $\varphi[r] = \Box^{\geqslant r} a$ are increasing, while their duals $\psi[r] = \Diamond^{\geqslant r} a$ and $\psi[r] = \Box^{\leqslant r} a$ are decreasing. Here, a is an atomic proposition or a Boolean combination thereof.[1] Within these notations, the formula $\phi_{e,u}$ above can be reformulated as $\Diamond^{\leqslant e}(\mathit{goal} \wedge (\mathit{utility} \geqslant u))$ when the consumed energy is modeled by a reward function and the accumulated utility is assumed to be encoded in the states, or as $\Diamond^{\geqslant u}(\mathit{goal} \wedge (\mathit{energy} \leqslant e))$ when utility is represented by a reward function and the consumed energy is augmented to states. Thus, query **(Q)** of the introduction corresponds to the task of computing an optimal scheduler $\mathfrak{S}$ for the quantile:

$$\max \left\{ u \in \mathbb{N} : \Pr_{s_{init}}^{\max}\big(\Diamond^{\geqslant u}(\mathit{goal} \wedge (\mathit{energy} \leqslant e))\big) \geqslant 0.8 \right\}$$

Similar to the already mentioned quantiles, we can define expectation quantiles, where the probability bound is replaced by a bound on an expected accumulated reward. For example, given two reward functions *energy* and *utility*, we may ask for $\min \{e \in \mathbb{N} : \mathbb{E}[\mathit{utility}]_{s_{init}}^{\max}(\Diamond(\mathit{energy} > e)) > u\}$ given a fixed utility threshold u, which corresponds to task **(Qe)** from the introduction.

Computation of Quantiles. For qualitative quantiles with upper-bounded eventually properties, i.e., quantiles with the probability bounds $= 1$, < 1, $= 0$ or > 0, the quantile values can be computed in polynomial time using a greedy method that shares some ideas of Dijkstra's shortest-path algorithm [36]. For other probability bounds, the schema for computing the quantile is as follows. We explain here the case $q_s = \min\{r \in \mathbb{N} : \Pr_s^{\min}(\Diamond^{\leqslant r} a) > p\}$. The treatment of other probability quantiles of the above type is analogous.

1. Compute $p_s = \Pr_s^{\min}(\Diamond a)$ for all states s. Then, with $X = \{s \in S : p_s \leqslant p\}$ we have $q_s = \infty$ iff $s \in X$.
2. If $S \neq X$ then for $r = 0, 1, 2, \ldots$, compute the values $p_{s,r} = \Pr_s^{\min}(\Diamond^{\leqslant r} a)$ for all $s \in S$. Proceed with step 3, as soon as $p_{s,r} > p$ for all states $s \in S \setminus X$.
3. For each $s \in S \setminus X$, return $q_s = \min\{r \in \mathbb{N} : p_{s,r} > p\}$.

The computation of the values $p_{s,r}$ in step 2 can be carried out using linear-programming techniques and reusing the values $p_{t,i}$ for $i < r$ computed in previous iterations. The computation of expectation quantiles can follow an analogous approach as for probability quantiles. The idea is first to identify the states where the expectation quantile is infinite. We then iteratively compute the values $u_{s,e} = \mathbb{E}[\mathit{utility}]_s^{\max}(\Diamond(\mathit{energy} > e))$ for $e = 0, 1, \ldots$ solving linear programs until $u_{s,e} > u$. Detailed explanations for computing quantiles can be found in [4].

[1] The semantics of the reward-bounded eventually and always operator is as follows. If ζ is an infinite path, then $\zeta \models \Diamond^{\bowtie r} a$ where $\bowtie \in \{\leqslant, \geqslant\}$ if there exists a position $k \in \mathbb{N}$ with $\mathit{rew}(\mathit{pref}(\zeta, k)) \bowtie r$ and $\zeta[k] \models a$. Similarly, $\Box^{\bowtie r} a \equiv \neg(\neg\Diamond^{\bowtie r}\neg a)$.

4 Conditional Probabilities and Expectations

Probabilities and expectations under the assumption that some additional temporal condition holds are often needed within the quantitative analysis of protocols. Constraints in conditional probabilities or expectations can be seen as a non-standard type of multi-objective properties. For example, in the context of energy-utility analysis, conditional probabilities or expectations are useful to analyze the energy-efficiency, while assuming that a certain condition on the achieved utility is guaranteed. Vice versa, one might ask, e.g., for the expected utility, while not exceeding a given energy budget.

Conditional probabilities in Markov chains can be computed simply by the definition of conditional probabilities as the quotient of ordinary probabilities:

$$\Pr_s^{\mathcal{M}}(\,\varphi \mid \psi\,) \;=\; \frac{\Pr_s^{\mathcal{M}}(\varphi \wedge \psi)}{\Pr_s^{\mathcal{M}}(\psi)}$$

where $\Pr_s^{\mathcal{M}}(\psi) > 0$. In what follows we refer to φ as the *objective* and to ψ as the *condition*. This approach has been taken in [3], where the condition and the objective are specified as PCTL path properties. The quotient method has been extended recently [24,30] for discrete and continuous-time Markov chains and patterns of path properties with multiple time- and cost-bounds. An alternative approach relies on a transformation $\mathcal{M} \rightsquigarrow \mathcal{M}_\psi$ such that for all measurable path properties φ the conditional probability for φ of $\mathcal{M}$ under condition ψ agrees with the standard (unconditional) probability for φ in $\mathcal{M}_\psi$ [8].

More challenging is the task to reason about conditional probabilities in MDPs. The crux is that for the computation of, e.g.,

$$\Pr_s^{\max}\big(\,\varphi \,|\, \psi\,\big) \;=\; \max_{\mathfrak{S}} \Pr_s^{\mathfrak{S}}\big(\,\varphi \,|\, \psi\,\big) \;=\; \max_{\mathfrak{S}} \frac{\Pr_s^{\mathfrak{S}}\big(\,\varphi \wedge \psi\,\big)}{\Pr_s^{\mathfrak{S}}\big(\,\psi\,\big)}$$

we cannot simply maximize the nominator and denominator independently. This problem has been addressed first in [3], where an extension of PCTL over MDPs [9] by a conditional probability operator has been introduced. The presented model-checking algorithm relies on an exhaustive search (with heuristic bounding techniques) in some finite, but potentially exponentially large class of finite-memory schedulers. In [8] we improved this result by presenting a polynomial transformation $\mathcal{M} \rightsquigarrow \mathcal{M}_{\varphi|\psi}$ for reachability objectives and conditions, which has been shown to be the core problem for reasoning about ω-regular objectives and conditions by using automata representations of the objective and the condition. In this approach, we fix an initial state s_{init} of $\mathcal{M}$. The idea for the construction of the transformed MDP $\mathcal{M}_{\varphi|\psi}$ is to redistribute the probabilities of paths where the condition ψ does not hold by adding reset-transitions to s_{init}. In the case of prefix-independent conditions (e.g. a reachability or a fairness condition) reset-transition $t \longrightarrow s_{init}$ are introduced for all states t where $\Pr_t^{\max}(\psi) = 0$.

The transformation explained in [8] relies on a preprocessing depending on both the condition ψ and the objective φ and generates a normal form for conditional probabilities. This preprocessing can, however, be dropped resulting in a transformation $\mathcal{M} \rightsquigarrow \mathcal{M}_\psi$ that only depends on the condition such that $\Pr^{\max}_{\mathcal{M},s_{init}}(\varphi \,|\, \psi)$ equals $\Pr^{\max}_{\mathcal{M}_\psi,s_{init}}(\varphi \wedge \psi)$ for all objectives φ, e.g., formalized as LTL or PRCTL path formulas. With this approach we can, for instance, compute *conditional quantiles* as in **(Cq)**, which are of the form

$$\min\big\{r \in \mathbb{N} : \Pr^{\max}_{\mathcal{M},s_{init}}\big(\varphi[r] \,|\, \psi\big) \geqslant p\big\} = \min\big\{r \in \mathbb{N} : \Pr^{\max}_{\mathcal{M}_\psi,s_{init}}\big(\varphi[r] \wedge \psi\big) \geqslant p\big\}$$

where $\varphi[r]$ is an increasing path property such as $\Diamond^{\leqslant r} a$ (see also Section 3).

5 Reasoning about the Energy-Utility Ratio

As a third type of non-standard multi-objective reasoning in MDPs we consider conditions on the quotient of two reward functions. The problem to compute expected ratios of accumulated rewards in MDPs was already addressed, e.g., in [1,38]. Probabilistic constraints on special types of long-run limits of accumulated rewards were studied in [20]. The results by [10,35] indicate that reasoning about long-run limits of (ratios of) accumulated values is algorithmically simpler than reasoning about the accumulated values along the prefixes of infinite paths. Indeed, [10] proves the undecidability of the model-checking problem for temporal logics extended by assertions on the accumulated values along the prefixes of infinite paths. Nevertheless, as the work on energy games [12,13] shows, there are several interesting patterns of formulas with prefix-accumulation assertions.

We consider here a pointed MDP $(\mathcal{M}, s_{init})$ with two weight functions, say *energy* and *utility* where the energy reward function is supposed to be positive. Let $\mathit{ratio} = \frac{\mathit{utility}}{\mathit{energy}} : \mathit{FinPaths} \to \mathbb{Q}$ given by $\mathit{ratio}(\pi) = \mathit{utility}(\pi)/\mathit{energy}(\pi)$ if $|\pi| \geqslant 1$ and ϑ a rational number specifying a *quality threshold*. Using an LTL-like syntax, we define the path property $\psi_\vartheta = \Box(\mathit{ratio} > \vartheta)$ where $\zeta \models \psi_\vartheta$ iff $\mathit{ratio}(\, \mathit{pref}(\zeta, k)\,) > \vartheta$ for all positions $k \in \mathbb{N}$ with $k \geqslant 1$. We now discuss the following questions assuming a given LTL-formula φ:

(E1) $\Pr^{\max}_{s_{init}}(\psi_\vartheta \wedge \varphi) = 1$	**(A1)** $\Pr^{\min}_{s_{init}}(\psi_\vartheta \wedge \varphi) = 1$
(E0) $\Pr^{\max}_{s_{init}}(\psi_\vartheta \wedge \varphi) > 0$	**(A0)** $\Pr^{\min}_{s_{init}}(\psi_\vartheta \wedge \varphi) > 0$

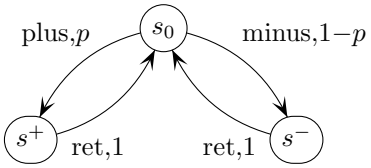
Problems (A1) and (E1). Obviously, the almost-sure problems **(A1)** and **(E1)** relate to the problems of deciding whether $\Pr^{\mathfrak{S}}_{s_{init}}(\psi_\vartheta \wedge \varphi) = 1$ for all schedulers $\mathfrak{S}$ (problem **(A1)**) respectively some scheduler $\mathfrak{S}$ (problem **(E1)**). We can rely on a simple transformation $(\mathit{energy}, \mathit{utility}) \mapsto \mathit{wgt}$ that permits to replace the ratio-constraint ψ_ϑ with a constraint $\Box(\mathit{wgt} > 0)$ for a single rational-valued weight function $\mathit{wgt} : S \times \mathit{Act} \to \mathbb{Q}$ defined by $\mathit{wgt}(s,\alpha) = \mathit{utility}(s,\alpha) - \vartheta \cdot \mathit{energy}(s,\alpha)$. It is clear that for all finite paths π we have $\mathit{ratio}(\pi) > \vartheta$ iff $\mathit{wgt}(\pi) > 0$, such that for all infinite paths ζ it holds that $\zeta \models \psi_\vartheta \wedge \varphi$ iff $\zeta \models \Box(\mathit{wgt} > 0) \wedge \varphi$. Since the weight functions for the energy and the utility are supposed to be integer-valued, we multiply *wgt* with the

denominator of ϑ to obtain an integer-valued weight function. This permits to assume that $wgt(s,\alpha) \in \mathbb{Z}$ for all states s, $s' \in S$.

We now address **(A1)**. Clearly, $\Pr^{\min}_{s_{init}}(\Box(wgt > 0) \wedge \varphi) = 1$ iff $\Pr^{\min}_{s_{init}}(\Box(wgt > 0)) = 1$ and $\Pr^{\min}_{s_{init}}(\varphi) = 1$. The condition $\Pr^{\min}_{s_{init}}(\varphi) = 1$ can be checked in time polynomial in the size of $\mathcal{M}$ using standard techniques. The condition $\Pr^{\min}_{s_{init}}(\Box(wgt > 0)) = 1$ is equivalent to $s \not\models \exists\Diamond(wgt < 0)$ and can be checked in polynomial time using standard shortest-path algorithms.

By the results established in [12] for energy games with an MDP game arena, problem **(E1)** is in NP∩coNP, when φ is a reachability, Büchi or parity condition. If φ is an LTL formula, we can rely on standard techniques to generate a deterministic parity automaton $\mathcal{A}$ for φ and then switch from $(\mathcal{M},\varphi)$ to $(\mathcal{M}\otimes\mathcal{A},\varphi_{\mathcal{A}})$, where $\varphi_{\mathcal{A}}$ is the acceptance (parity) condition of $\mathcal{A}$.

Problems (A0) and (E0). Let us now turn to **(A0)** and **(E0)**, where the task is to check whether $\Pr^{\mathfrak{S}}_{s_{init}}(\psi_\theta \wedge \varphi) > 0$ for all schedulers $\mathfrak{S}$ (problem **(A0)**) or for some scheduler $\mathfrak{S}$ (problem **(E0)**). The challenge in providing algorithms for these two problems becomes clear as they depend on the concrete transition probabilities of $\mathcal{M}$. This even holds for the case of $\mathcal{M}$ being a Markov chain. Consider the Markov chain $\mathcal{M} = \mathcal{M}_p$ in the following picture where $p \in]0,1[$ is a probability parameter.



$$\begin{aligned} wgt(s_0,\text{plus}) &= 1 \\ wgt(s_0,\text{minus}) &= -1 \\ wgt(s^+,\text{ret}) &= 0 \\ wgt(s^-,\text{ret}) &= 0 \end{aligned}$$

Finite paths in $\mathcal{M}$ starting and ending in s_0 constitute a biased random walk, for which it is well-known that for $p > \frac{1}{2}$, the random walk drifts to the right and never reaches position -1 with positive probability, whereas for $0 < p \leqslant \frac{1}{2}$, position -1 will be visited almost surely. Thus, $\Pr^{\mathcal{M}}_{s_0}\big(\Box(wgt \geqslant 0)\big) > 0$ iff $p > \frac{1}{2}$. As a consequence, the answers for questions **(A0)** and **(E0)** may depend on the concrete transition probabilities. This observation rules out simple algorithms relying on shortest-path arguments as for **(A1)**. Nevertheless, problems **(A0)** and **(E0)** and even the task to check whether $\Pr^{\mathcal{M}}_{s_{init}}(\psi_\vartheta \wedge \varphi) \bowtie p$ for some given $p \in [0,1] \cap \mathbb{Q}$ and $\bowtie \in \{\leqslant, <, \geqslant, >\}$ is decidable when $\mathcal{M}$ is a Markov chain. For this, we can rely on a reduction to the probabilistic model-checking problem for probabilistic push-down automata (pPDA) and apply the techniques presented, e.g., in [11]. The idea is simply to consider the states of $\mathcal{M}$ as control states of a pPDA and to mimic each transition $s \to s'$ in $\mathcal{M}$ of weight $k > 0$ by k push operations and each transition $s \to s'$ with weight $-k$ by k pop operations. This reduction is exponential, but shows the decidability of the problems mentioned above for Markov chains. To the best of our knowledge, the decidability of problems **(E0)** and **(A0)** for MDPs is an open problem.

6 Multi-objective Analysis of Software Product Lines

The previously presented methods are also suitable for feature-oriented software engineering areas such as *software product line (SPL)* engineering. An SPL specifies a collection of software products by their commonalities in terms of features rather than specifying all software products separately [17]. Formal analysis of SPLs has to tackle the combinatorial blow-up in the number of features arising from composing features and their dynamical changes during runtime of the system (e.g., by software updates or in-app purchases). Family-based analysis approaches [39], where all possible software products are checked at once instead of checking each feature combination one-by-one, turned out to be very successful [16]. Recently, we proposed a compositional framework for probabilistic SPLs which supports dynamic feature changes and has an MDP-based semantics, allowing for quantitative analysis using standard PMC-methods [22]. We demonstrated feasibility of our approach by analyzing an extended version of the energy-aware bonding network device eBond [26]. The so-called eBond+ describes how energy can be saved by switching to various network-card combinations in a server depending on the requested traffic load. Already the analysis of single objectives in terms of performance measures, energy consumption and monetary costs revealed interesting insights of the influence of feature combinations. In this section we detail how the methods for analyzing multiple objectives in terms of quantiles, conditionals and ratio reasoning can be applied to SPLs such as eBond+.

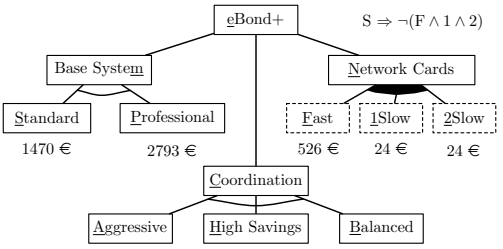


Fig. 3. Feature diagram of eBond+ SPL

For a set of features in an SPL, feature combinations which yield a valid product are usually defined through a *feature diagram*, providing a hierarchical structure of features. As shown in Figure 3, the eBond+ SPL consists of a base feature containing one base standard or professional system, one coordination feature which defines how network traffic is distributed on the network cards (aggressive, high or balanced energy savings) and one fast or two slow network cards which can be plugged into the system in several combinations. The latter is illustrated in the feature diagram by dashed *optional features* that can be activated or deactivated during runtime of the system. Feature diagrams may be further annotated with propositional formulas over features, e.g., $S \Rightarrow \neg(F \wedge 1 \wedge 2)$ states that within a standard version of the base system, at most two network cards can be plugged simultaneously.

Various measures in terms of cost and rewards can now be investigated. We considered the different measures within a fixed time horizon of t minutes the system is observed. The amount of time a service-level agreement (SLA) is fulfilled, i.e., the number of minutes within this time horizon the requested bandwidth is

served, can be used as a measure for utility that is decreasing over time. The opposite, an SLA violation, occurs, e.g., due to a probabilistically modeled system failure or due to aggressively putting network cards into sleeping mode by the energy savings coordination feature. Costs for keeping SLA violations as rare as possible may be formalized as increasing measures in terms of energy (the fast network card consumes more energy than the slower ones) or money (monetary costs as annotated in Figure 3 are higher for better equipped eBond+ product).

The eBond+ model is an MDP encoding all the cost and utility measures described above, allowing for PMC-based analysis. In order to minimize SLA violations, money investments or energy consumption we are thus able to construct optimal schedulers for activating and deactivating network cards during runtime according to rules fixed by an operational model called *feature controller* (solving the *strategy synthesis problem*). All of the exemplified tasks described in the introduction and formalized in the last sections are also useful in the eBond+ setting, employing the utility and energy measures described above and choosing the goal set T of states representing the states where the time horizon t is reached. For instance, the expectation quantile task **(Qe)** with $u = 0.99 \cdot t$ amounts to minimizing the energy required to ensure that at least 99% of the time no SLA violation can be expected. A similar task can also be stated minimizing the money which needs to be invested for expecting a quality of service of at least 99%, replacing *energy* by *money* in the formula stated in Section 3.

To reason about feature combinations, conditional probabilities turned out to be very useful: The rules for feature changes defined by the feature controller can be further restricted according to assumed behavior of the environment, e.g., by fairness assumptions. For instance, the maximal probability that no SLA violation occurs under the condition that infinitely often a slow network card (1 or 2) is plugged in can be expressed by $\Pr^{\max}_{s_{init}}\big(\Diamond \mathit{NoSla} \mid \Box\Diamond(1 \vee 2) \big)$, where $\mathit{NoSla} \subseteq T$ denotes the set of states where $\mathit{utility} \geqslant t$, i.e., the time horizon has been reached without an SLA violation in the past. Conditional expectations also play an important role when expected costs and rewards do not yield proper values, e.g., if $\Pr^{\min}_{s_{init}}(\Diamond\chi) < 1$ and hence the expectation $\mathbb{E}[\mathit{energy}]^{\min}_{s_{init}}(\Diamond\chi)$ is infinite. While $\Diamond \mathit{NoSla}$ does not hold almost surely without any further assumptions, this can be guaranteed for a plugged fast network card (F) and assuming that no failures occur. When the set of states *Fail* represents that a failure of network cards occurred, $\mathbb{E}[\mathit{energy}]^{\min}_{s_{init}}(\Diamond \mathit{NoSla} \mid \Box(\neg \mathit{Fail} \wedge F))$ thus yields a proper energy expectation. Hence, this expectation can be used for best-case analysis which may help to schedule further development steps of the eBond+ SPL or estimate overall system quality.

Departing from the finite time horizon perspective and having in mind that the fast network card consumes most energy, one could also ask for a scheduler which guarantees the ratio between energy and utility to be almost surely above a certain threshold and infinitely often switches from the fast network card to a slower one. This task corresponds to **(R)**, which can be furthermore combined with requiring the total accumulated energy to be always below a given energy threshold e if the system intents to use the fast network card **(Rc)**.

That extension, which requires the energy consumption to be encoded into the states of the model, can be formalized by asking whether

$$\Pr^{\mathfrak{S}}_{s_{init}}\big(\ \Box(\mathit{ratio} > \vartheta) \wedge \neg\Diamond\Box F \,|\, \Box(F \Rightarrow (\mathit{energy} \leqslant e))\ \big) \ = \ 1.$$

It is obvious that the approaches of computing quantiles, conditionals and ratios can be further combined, which enables to provide even more insights on feature-oriented quantitative properties of eBond+ than already demonstrated in [22] within single objectives.

Detailed case studies for multi-objective reasoning and theoretical considerations of nesting such objectives are subject of our current work.

References

1. Aggarwal, V., Chandrasekaran, R., Nair, K.: Markov ratio decision processes. Journal of Optimization Theory and Application 21(1) (1977)
2. Andova, S., Hermanns, H., Katoen, J.-P.: Discrete-time rewards model-checked. In: Larsen, K.G., Niebert, P. (eds.) FORMATS 2003. LNCS, vol. 2791, pp. 88–104. Springer, Heidelberg (2004)
3. Andrés, M., van Rossum, P.: Conditional probabilities over probabilistic and nondeterministic systems. In: Ramakrishnan, C.R., Rehof, J. (eds.) TACAS 2008. LNCS, vol. 4963, pp. 157–172. Springer, Heidelberg (2008)
4. Baier, C., Daum, M., Dubslaff, C., Klein, J., Klüppelholz, S.: Energy-utility quantiles. In: NFM 2014. LNCS (to appear, 2014)
5. Baier, C., Daum, M., Engel, B., Härtig, H., Klein, J., Klüppelholz, S., Märcker, S., Tews, H., Völp, M.: Locks: Picking key methods for a scalable quantitative analysis. Journal of Computer and System Sciences (to appear, 2014)
6. Baier, C., Engel, B., Klüppelholz, S., Märcker, S., Tews, H., Völp, M.: A probabilistic quantitative analysis of probabilistic-write/copy-select. In: Brat, G., Rungta, N., Venet, A. (eds.) NFM 2013. LNCS, vol. 7871, pp. 307–321. Springer, Heidelberg (2013)
7. Baier, C., Katoen, J.-P.: Principles of Model Checking. MIT Press (2008)
8. Baier, C., Klein, J., Klüppelholz, S., Märcker, S.: Computing conditional probabilities in Markovian models efficiently. In: TACAS 2014. LNCS (to appear, 2014)
9. Bianco, A., de Alfaro, L.: Model checking of probabilistic and non-deterministic systems. In: Thiagarajan, P.S. (ed.) FSTTCS 1995. LNCS, vol. 1026, pp. 499–513. Springer, Heidelberg (1995)
10. Boker, U., Chatterjee, K., Henzinger, T.A., Kupferman, O.: Temporal specifications with accumulative values. In: LICS 2011, pp. 43–52. IEEE Computer Society (2011)
11. Brázdil, T., Kučera, A., Stražovský, O.: On the decidability of temporal properties of probabilistic pushdown automata. In: Diekert, V., Durand, B. (eds.) STACS 2005. LNCS, vol. 3404, pp. 145–157. Springer, Heidelberg (2005)
12. Chatterjee, K., Doyen, L.: Energy and mean-payoff parity Markov decision processes. In: Murlak, F., Sankowski, P. (eds.) MFCS 2011. LNCS, vol. 6907, pp. 206–218. Springer, Heidelberg (2011)

13. Chatterjee, K., Doyen, L.: Energy parity games. Theoretical Computer Science 458, 49–60 (2012)
14. Chatterjee, K., Majumdar, R., Henzinger, T.: Markov decision processes with multiple objectives. In: Durand, B., Thomas, W. (eds.) STACS 2006. LNCS, vol. 3884, pp. 325–336. Springer, Heidelberg (2006)
15. Clarke, E., Grumberg, O., Peled, D.: Model Checking. MIT Press (2000)
16. Classen, A., Cordy, M., Schobbens, P.-Y., Heymans, P., Legay, A., Raskin, J.-F.: Featured transition systems: Foundations for verifying variability-intensive systems and their application to ltl model checking. In: IEEE TSE (2012)
17. Clements, P., Northrop, L.: Software Product Lines: Practices and Patterns. Addison-Wesley Professional (2001)
18. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. Journal of the ACM 42(4), 857–907 (1995)
19. de Alfaro, L.: Formal Verification of Probabilistic Systems. PhD thesis, Stanford University, Department of Computer Science (1997)
20. de Alfaro, L.: How to specify and verify the long-run average behavior of probabilistic systems. In: LICS 1998, pp. 454–465. IEEE Computer Society (1998)
21. de Alfaro, L.: Computing minimum and maximum reachability times in probabilistic systems. In: Baeten, J.C.M., Mauw, S. (eds.) CONCUR 1999. LNCS, vol. 1664, pp. 66–81. Springer, Heidelberg (1999)
22. Dubslaff, C., Klüppelholz, S., Baier, C.: Probabilistic model checking for energy analysis in software product lines. In: MODULARITY 2014 (to appear, 2014)
23. Etessami, K., Kwiatkowska, M., Vardi, M., Yannakakis, M.: Multi-objective model checking of Markov decision processes. Logical Methods in Computer Science 4(4) (2008)
24. Gao, Y., Xu, M., Zhan, N., Zhang, L.: Model checking conditional CSL for continuous-time Markov chains. IPL 113(1-2), 44–50 (2013)
25. Grädel, E., Thomas, W., Wilke, T. (eds.): Automata, Logics, and Infinite Games. LNCS, vol. 2500. Springer, Heidelberg (2002)
26. Hähnel, M., Döbel, B., Völp, M., Härtig, H.: ebond: Energy saving in heterogeneous R.A.I.N. In: Proc. of the Fourth International Conference on Future Energy Systems, e-Energy 2013, pp. 193–202. ACM (2013)
27. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing 6, 512–535 (1994)
28. Haverkort, B.: Performance of Computer Communication Systems: A Model-Based Approach. Wiley (1998)
29. Hinton, A., Kwiatkowska, M., Norman, G., Parker, D.: PRISM: A tool for automatic verification of probabilistic systems. In: Hermanns, H., Palsberg, J. (eds.) TACAS 2006. LNCS, vol. 3920, pp. 441–444. Springer, Heidelberg (2006)
30. Ji, M., Wu, D., Chen, Z.: Verification method of conditional probability based on automaton. Journal of Networks 8(6), 1329–1335 (2013)
31. Katoen, J.-P., Zapreev, I., Hahn, E., Hermanns, H., Jansen, D.: The ins and outs of the probabilistic model checker MRMC. Performance Evaluation 68(2) (2011)
32. Kulkarni, V.: Modeling and Analysis of Stochastic Systems. Chapman & Hall (1995)
33. Puterman, M.: Markov Decision Processes: Discrete Stochastic Dynamic Programming. Wiley (1994)
34. Serfling, R.J.: Approximation Theorems of Mathematical Statistics. Wiley (1980)

35. Tomita, T., Hiura, S., Hagihara, S., Yonezaki, N.: A temporal logic with mean-payoff constraints. In: Aoki, T., Taguchi, K. (eds.) ICFEM 2012. LNCS, vol. 7635, pp. 249–265. Springer, Heidelberg (2012)
36. Ummels, M., Baier, C.: Computing quantiles in Markov reward models. In: Pfenning, F. (ed.) FOSSACS 2013. LNCS, vol. 7794, pp. 353–368. Springer, Heidelberg (2013)
37. Vardi, M.: Automatic verification of probabilistic concurrent finite-state programs. In: FOCS 1985, pp. 327–338. IEEE Computer Society (1985)
38. von Essen, C., Jobstmann, B.: Synthesizing systems with optimal average-case behavior for ratio objectives. In: iWIGP 2011. EPTCS, vol. 50, pp. 17–32 (2011)
39. von Rhein, A., Apel, S., Kästner, C., Thüm, T., Schaefer, I.: The PLA model: On the combination of product-line analyses. In: Proc. of VaMoS 2013. ACM (2013)