In *Proc. 13th IEEE Symp. Logic in Comp. Sci., 1998.*

How to Specify and Verify the Long-Run Average Behavior of Probabilistic Systems

Luca de Alfaro*
University of California at Berkeley

Abstract

Long-run average properties of probabilistic systems refer to the average behavior of the system, measured over a period of time whose length diverges to infinity. These properties include many relevant performance and reliability indices, such as system throughput, average response time, and mean time between failures.

In this paper, we argue that current formal specification methods cannot be used to specify long-run average properties of probabilistic systems. To enable the specification of these properties, we propose an approach based on the concept of experiments. Experiments are labeled graphs that can be used to describe behavior patterns of interest, such as the request for a resource followed by either a grant or a rejection. Experiments are meant to be performed infinitely often, and it is possible to specify their long-run average outcome or duration.

We propose simple extensions of temporal logics based on experiments, and we present model-checking algorithms for the verification of properties of finite-state timed probabilistic systems in which both probabilistic and nondeterministic choice are present. The consideration of system models that include nondeterminism enables the performance and reliability analysis of partially specified systems, such as systems in their early design stages.

1 Introduction

Long-run average properties of probabilistic systems include many classical performance and reliability indices, such as system throughput, average response time, and mean time between failures. These properties refer to the average behavior of the system, measured over a period of time whose length diverges to infinity [15]. In systems modeled as Markov chains, long-run average properties are related to the steady-state distribution of the chain [20]. In this paper we argue that current approaches to formal specification are not suited for the study of long-run average properties of probabilistic systems, and we present verification and specification methods that overcome this limitation.

Current approaches to the specification and verification of probabilistic systems are based either on extensions of temporal logics, or on probabilistic process algebras, simulation and bisimulation relations, and testing preorders.

Temporal logics for the specification of quantitative properties of probabilistic systems have been presented in [18, 2, 7, 21, 13], and a probabilistic duration calculus has been studied in [24]. These logics enable the specification of bounds for the probability of satisfying temporal or duration calculus formulas, starting from given subsets of system states. Model-checking algorithms for these logics have been presented in [10, 11, 7, 21, 13]. These logics can be used to specify many properties of interest, such as bounds on the probability of meeting a deadline, or of reaching a deadlock, starting from a given set of states. These properties are related to *ensemble averages* (or probabilities) over the set of behaviors that originate from single system states.

Long-run average properties of systems are related to *time averages* along system behaviors, rather than ensemble averages. This indicates that the above logics cannot be used to specify long-run average properties of systems. In fact, even in the case of purely probabilistic systems, these logics cannot take into account the long-run average probability of being at given system states, or the long-run average outcome of system choices. This limits their ability to capture a large number of classical performance and reliability properties.

Another approach to the specification of probabilistic systems is based on the use of probabilistic pro-

*Dept. of EECS, University of California, Berkeley, CA 94720-1770. Email: dealfaro@eecs.berkeley.edu. The work was partially supported by the NSF grant CCR-95-27927, by DARPA under the NASA grant NAG2-892, by the ARO grant DAAH04-95-1-0317, by ARO under the MURI grant DAAH04-96-1-0341, by Army contract DABT63-96-C-0096 (DARPA), by the ONR YIP award N00014-95-1-0520, by the NSF CAREER award CCR-9501708, and by the NSF grant CCR-9504469.

cess algebras, simulation and bisimulation relations, and testing preordres (see for instance [23, 9, 29], or [28, 12] for more comprehensive summaries of this approach). Simulation and bisimulation relations preserve the long-run average behavior of probabilistic systems, but they offer no direct method for the specification of performance and reliability properties connected to this type of behavior. The approaches based on tests enable to quantify the probability with which the system passes the tests. Since the tests are generally executed only once, they cannot be used to measure long-run average properties.

This paper presents a method for the formal specification and verification of long-run average properties of probabilistic systems. The method is based on the concept of *experiments,* inspired by the tests of process algebra. Experiments are labeled graphs that can be used to describe behavior patterns of interest, such as the request for a resource followed by either a grant or a rejection. Experiments associate with each occurrence of these patterns an *outcome:* a real number representing the success or the duration of the pattern. For example, we can associate to the above experiment outcome 1 if the request is granted, and 0 if it is rejected. Unlike tests, experiments are meant to be performed infinitely often, and it is possible to measure their long-run average outcome. The long-run average outcome of the above experiment is equal to the long-run average fraction of requests that are granted. Experiments provide a specification method for long-run average properties that is semantically sound, and easy to understand, even when the system model includes nondeterminism.

We model the behavior of probabilistic systems in terms of probability, nondeterminism, and time: our models are based on Markov decision processes, augmented by additional information on the timing behavior of the system. The inclusion of nondeterminism in the system model enables the study of performance and reliability of partially specified systems, such as systems in their early design stages.

We propose simple extensions of branching-time temporal logics based on experiments. These extensions are obtained by introducing new operators that enable to express bounds on the long-run average outcome of experiments. These extensions enable the use of a single language for the specification of correctness, reliability, and performance properties of systems. We also discuss the relationship between the long-run average properties studied in this paper and the properties expressible in previous probabilistic extensions of temporal logic.

Finally, we present model-checking algorithms for the verification of long-run average properties of finite-state systems. The model-checking algorithms are based on new results from the theory of Markov decision processes. They have polynomial time-complexity both in the size of the system and in the size of the experiment, indicating that this proposal provides a practical approach to the formal specification and verification of long-run average properties of systems.

Formal Methods for Performance Modeling

To complete our review of related work, we mention the use of formal methods for the construction of performance models of systems. While this approach does not deal with the issue of specification languages, it nevertheless provides methods for measuring several performance and reliability indices of purely probabilistic systems (i.e. systems not containing nondeterministic choice).

A popular approach to the construction of performance models relies on probabilistic extensions of Petri nets, such as the *stochastic Petri nets* of [26, 25, 30] and the *generalized stochastic Petri nets* (GSPNs) of [1]. The transitions of a GSPN can either fire immediately, or fire with an exponential delay distribution. GSPNs can be translated into continuous-time Markov chains, thus enabling the performance analysis of the systems.

Another approach to the compositional modeling of probabilistic systems is based on extensions of process algebras. The process algebras MPA and EMPA associate delay distributions with the actions [5]. An EMPA system model can be either translated into a GSPN, or directly into a continuous-time Markov chain, thus allowing the performance evaluation of the system [4, 3]. The idea of associating delay distributions with the actions is also at the basis of the process algebras PEPA [19] and TIPP [16]. System models written in PEPA and in subsets of TIPP can again be translated into continuous-time Markov chains.

These formalisms enable the performance modeling only of purely probabilistic systems: nondeterminism can be present in system sub-components, but not in the complete model that is translated into a continuous-time Markov chain.

The performance and reliability quantities of interest can be measured by adding annotations to the models. In a *GSPN reward model,* a reward rate is associated with each place and transition of the net [8]; in PEPA, a reward rate can be associated with each action. The average reward per unit of time can then

be computed by solving the continuous-time Markov chains obtained from the systems.

The experiments we introduce here also provide a flexible way of associating rewards with a system. Additionally, using experiments we can measure not only the amount of reward per unit of time, but also the amount of reward *per experiment*. For systems that can be translated into ergodic Markov chains [20], the long-run average outcome per experiment can be computed by computing separately the rates of outcome generation and of experiment completion, and by taking the ratio between the two. However, in the case of systems with nondeterminism this approach fails, and the introduction of experiments leads to more expressive specification methods.

2 Models for Probabilistic Systems

Our models for probabilistic systems are based on *Markov decision processes* (MDPs). An MDP is a generalization of a Markov chain in which a set of possible actions is associated with each state. To each state-action pair corresponds a probability distribution on the states, which is used to select the successor state [14, 6].

Definition 1 (Markov decision process) A *Markov decision process* (MDP) (S, A, p) consists of a finite set S of states, and of two components A, p that specify the transition structure:

- For each $s \in S$, $A(s)$ is a non-empty finite set of *actions* available at s.
- For each $s, t \in S$ and $a \in A(s)$, $p_{st}(a)$ is the probability of a transition from s to t when action a is selected. For every $s, t \in S$ and $a \in A(s)$, it is $0 \leq p_{st}(a) \leq 1$ and $\sum_{t \in S} p_{st}(a) = 1$. ∎

We will often associate with an MDP additional labelings to represent quantities of interest; the labelings will be simply added to the list of components.

A *behavior* of an MDP is an infinite sequence of alternating states and actions, constructed by iterating a two-phase selection process. First, given the current state s, an action $a \in A(s)$ is selected nondeterministically; second, the successor state t of s is chosen according to the probability distribution $\Pr(t \mid s, a) = p_{st}(a)$. The formal definition of behavior is as follows.

Definition 2 (behaviors of MDP) A *behavior* of an MDP Π is an infinite sequence $\omega : s_0 a_0 s_1 a_1 \cdots$ such that $s_i \in S$, $a_i \in A(s_i)$ and $p_{s_i, s_{i+1}}(a_i) > 0$ for all $i \geq 0$. We let X_i, Y_i be the random variables representing the i-th state and the i-th action along a behavior, respectively. Formally, X_i and Y_i are variables that assume the value s_i, a_i on the behavior $\omega : s_0 a_0 s_1 a_1 \cdots$. ∎

Policies and probability of behaviors. For every state $s \in S$, we denote by Ω_s the set of behaviors starting from s, and we let $\mathcal{B}_s \subseteq 2^{\Omega_s}$ be the σ-algebra of *measurable* subsets of Ω_s, following the classical definition of [20]. To be able to talk about the probability of system behaviors, we need to specify the criteria with which the actions are chosen. To this end, we use the concept of *policy* [14], closely related to the adversaries of [29, 28] and to the schedulers of [31, 27].

A policy η is a set of conditional probabilities $Q_\eta(a \mid s_0 s_1 \cdots s_n)$, for all $n \geq 0$, $s_0, s_1, \ldots, s_n \in S$ and $a \in A(s_n)$. According to policy η, after the finite prefix $s_0 a_0 s_1 \cdots s_n$, action $a \in A(s_n)$ is chosen with probability $Q_\eta(a \mid s_0 s_1 \cdots s_n)$. Hence, under policy η the probability of following a finite behavior prefix $s_0 a_0 s_1 a_1 \cdots s_n$ is $\prod_{i=0}^{n-1} p_{s_i, s_{i+1}}(a_i)\, Q_\eta(a_i \mid s_0 \cdots s_i)$.

These probabilities for prefixes give rise to a unique probability measure on $\mathcal{B}_s$. We write $\Pr_s^\eta(\mathcal{A})$ to denote the probability of event $\mathcal{A}$ in Ω_s under policy η, and $\mathrm{E}_s^\eta\{f\}$ to denote the expectation of the random function f from state s under policy η.

Timed probabilistic systems. Our model for probabilistic systems is that of *timed probabilistic system* (TPSs). A TPS is an MDP with three additional labelings, that describe the set of initial states, the timing properties of the system, and the values of a set of state variables at all system states. TPSs are closely related to *semi-Markov decision processes* [6].

For simplicity, we assume a fixed set $\mathcal{V}$ of state variables.

Definition 3 (TPS) A TPS $(S, A, p, S_{in}, \mathit{time}, \mathcal{I})$ is an MDP (S, A, p) with three additional components:

- A subset $S_{in} \subseteq S$ of initial states.
- A labeling *time* that associates with each $s \in S$ and $a \in A(s)$ the *expected* amount of time $\mathit{time}(s, a) \in \mathbb{R}^+$ spent at s when action a is selected.
- A labeling $\mathcal{I}$ that associates with each $x \in \mathcal{V}$ and $s \in S$ the value $\mathcal{I}_s[\![x]\!]$ of x at s. ∎

The relative simplicity of this model enables us to focus our attention on the specification and verification of long-run average properties, rather than on

modeling issues. It is possible to define higher-level compositional models for probabilistic systems that can be automatically translated into TPSs; an example of such higher-level models are the *stochastic transition systems* of [12]. Several other models can be similarly translated.

We say that time *diverges* along a behavior iff $\sum_{k=0}^{\infty} time(X_k, Y_k)$ diverges. Since behaviors along which time does not diverge do not have a physical meaning, we make the following assumption about the TPSs under consideration:

Non-Zenoness Assumption: For every policy η and state s, $\Pr_s^{\eta}\left(\sum_{i=0}^{\infty} time(X_i, Y_i) = \infty\right) = 1$.

This assumption can be verified using the algorithm described in [13]. A more general approach to the problem of time divergence, inspired by [28], is presented in [12, §8].

3 A Motivational Example

To gain a better understanding of why existing formal specification methods cannot capture long-run average properties of systems, we present a simple example: the specification of the long-run average probability of gaining access to a shared resource in a multi-user system.

Specifically, we consider a system SHARED-RES, consisting of N users that can access a shared resource. The resource can be used by at most $M \leq N$ users at any single time. Each user can be in one of three states: *idle*, *requesting* and *using*. For the sake of simplicity, we assume that each step of the system has unit duration.

Initially, all users are at *idle*. At each time step, if a user is at *idle* it has probability p of going to *requesting* and $1-p$ of staying at *idle*. If the user is in *using*, it has probability q of going to *idle*, and $1-q$ of staying in *using*. The behavior of a single user is depicted in Figure 1.

Let j and k, with $k \leq M$, be the number of users in *requesting* and *using*, respectively, at the beginning of a time step. The scheduler may grant access to any number of users between $m = \min\{M_0, M-k, j\}$ and $n = \min\{M-k, j\}$, where $M_0 > 0$ is a constant. Thus, from the state there are actions $a_m, a_{m+1}, \ldots, a_n$. If action a_l is chosen, with $m \leq l \leq n$, then l among the j users at *requesting* are selected uniformly at random and go to *using*, while the remaining $j-l$ users are sent back to *idle*. From this informal description, for given N, M, M_0, p, and q, it is possible to construct a TPS $\Pi(N, M, M_0, p, q)$.

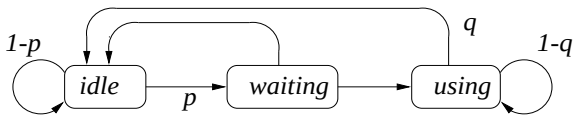

Figure 1: Behavior of a single user in system SHARED-RES. The transitions from state *waiting* depend on the scheduler and on the states of the other users in the system.

This system was originally devised as a very simple model for the behavior of people placing phone calls: N is the number of people, M is the maximum number of calls that can be active at the same time, and M_0 is the minimum number of new calls that the phone company guarantees to be able to connect in a time unit. The transition from *idle* to *requesting* corresponds to the act of lifting the handset to place a call; the transition from *using* to *idle* corresponds to hanging up at the end of the call. The transitions out of the *requesting* state model the acts of either getting the connection or of hanging up upon hearing the busy signal. Our intended specification is as follows:

Req1: *For any user, the long-run fraction of requests that are granted is at least b_0, for some specified $0 < b_0 < 1$.*

We will attempt to encode this specification in the probabilistic temporal logic pCTL, derived from CTL by introducing the probability operator P [17, 7, 18]. The operator P can be used to express bounds on probabilities, and it is syntactically similar to a path quantifier. If ϕ is a linear-time temporal logic formula, then $\mathrm{P}_{\geq b_0}\phi$ holds at a state s iff the probability that a system behavior from s satisfies ϕ is at least b_0 (the cases for $\leq$, $<$, $>$ are analogous).

As the situation is symmetrical for all users, let us concentrate on the first user. Let I_1, R_1, U_1 be atomic formulas representing the fact that user 1 is at *idle*, *requesting* or *using*, respectively. It might seem plausible at first to encode the requirement Req1 with the following pCTL formula:

$$\mathrm{A}\Box(R_1 \rightarrow \mathrm{P}_{\geq b_0}(R_1 \,\mathcal{U}\, U_1)) \,. \qquad (1)$$

Let us analyze this formula. Call any state that satisfies R_1 an *R_1-state*. Subformula $\mathrm{P}_{\geq b_0}(R_1 \,\mathcal{U}\, U_1)$ holds at an R_1-state iff the request of user 1 at that state will be granted with probability at least b_0, regardless of the policy. By definition of A and $\Box$, specification (1) holds for SHARED-RES iff every reachable R_1-state satisfies $\mathrm{P}_{\geq b_0}(R_1 \,\mathcal{U}\, U_1)$.

The problem with specification (1) is that there are many reachable R_1-states in the system: in some of them, few users are accessing the resource; in others, the resource is fully utilized. Clearly, the probability that the first user gets access to the resource from these latter R_1-states is 0. Thus, as long as there is a reachable R_1-state in which M users are already using the resource, specification (1) will not be satisfied, regardless of the long-run average probability with which the first user succeeds in accessing the resource.

More generally, the problem is that the temporal logics presented in [17, 2, 7, 13] can specify the probability with which behaviors satisfy temporal formulas from given states, but they do not take into account the long-run probability of being in those states.

A similar argument applies to the specification approaches based on process algebras, simulation relations and testing preorders. While these approaches can characterize many probabilistic properties of interest, they do not take into account the long-run probability of being in given system states. Indeed, an exam of the verification algorithms that have been proposed to check these relations in purely probabilistic systems confirms that the computation of steady-state probability distributions is not among the tasks performed by the algorithms.

4 Specification of Long-Run Average Properties

The specification method we present is based on the concept of *experiments*. An *experiment* is simply a finite deterministic automaton, with a distinguished set of initial states and some additional labelings. Experiments describe behavior patterns of interest, and are applied to a TPS by forming the synchronous composition between the experiment and the TPS. Each time an experiment is performed, it yields an outcome, related either to the success, or to the duration, of the experiment. Accordingly, we distinguish two types of experiments: *P-experiments,* to measure discrete outcomes, and *D-experiments,* to measure durations.

Definition 4 (experiment) An *experiment* $\Psi = (V, E, E_r, V_{in}, \lambda)$ is a labeled graph (V, E), with set of vertices V and set of edges $E \subseteq V \times V$, and with the following additional components:

- A set $V_{in} \subseteq V$ of *initial vertices.*
- A set $E_r \subseteq E - \{(v, v) \mid v \in V\}$ of *reset edges.*
- A labeling λ that assigns to each $u \in V$ a first-order formula $\lambda(u)$ over the state variables $\mathcal{V}$.

For all $u \in V$, we denote by $dst(u) = \{v \in V \mid (u, v) \in E\}$ the set of vertices that can be reached in one step from u, and we require $u \in dst(u)$. The labeling of the vertices must be deterministic and total. Specifically, the following formulas must be valid (i.e. true in any type-consistent variable interpretation):

1. $\bigvee_{v \in V_{in}} \lambda(v)$.
2. $\bigvee_{v \in dst(u)} \lambda(v)$, for all $u \in V$.
3. $\neg[\lambda(v_1) \wedge \lambda(v_2)]$, for all $v_1, v_2 \in V_{in}$.
4. $\neg[\lambda(v_1) \wedge \lambda(v_2)]$, for all $u \in V$, $v_1, v_2 \in dst(u)$. ∎

When the synchronous composition between an experiment Ψ and a TPS Π is formed, the vertex labels λ of Ψ are used to synchronize the transitions of Ψ and Π. The fact that $u \in dst(u)$ for all $u \in V$ ensures that, if the variable assignment does not change, the experiment remains at the same vertex. Each time a reset edge is traversed, we say that the experiment *ends,* so that the number of reset edges traversed along a behavior indicates how many experiments have been completed.

In a P-experiment, we associate with each reset edge an *outcome:* a real number representing a reward earned when the experiment is ended.

Definition 5 (P-experiment) A *P-experiment* is an experiment in which each reset edge $(u, v) \in E_r$ is labeled with an *outcome* $\gamma(u, v) \in \mathbb{R}$. ∎

Often, we are interested in experiments whose outcome is binary: they can either succeed or fail. In this case, we associate outcome 1 with the reset edges that represent a successful completion of the experiment, and outcome 0 with those representing failures.

In a D-experiment, we specify a set of *timed vertices*: the outcome of a D-experiment is equal to the total time spent at timed vertices.

Definition 6 (D-experiment) A *D-experiment* is an experiment with a distinguished nonempty subset $V_t \subseteq V$ of *timed vertices.* ∎

Example 1 (average success in SHARED-RES**)** In Figure 2 we present a P-experiment that can be used to express specification Req1. If user 1 proceeds from *requesting* to *using,* the outcome is 1; if user 1 proceeds from *requesting* to *idle*, the outcome is 0. Specification Req1 can be encoded by requiring the long-run average outcome of the experiment to be at least b_0. ∎

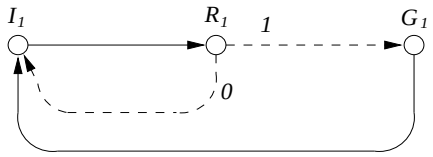

Figure 2: P-experiment Ψ_{Req1}, for the specification of Requirement Req1 of SHARED-RES. All vertices are initial; the reset edges are drawn as dashed lines. We omitted the self-loops and the edges that are never followed by system SHARED-RES.

4.1 Long-Run Average Outcome of Experiments

To use experiments for the specification of long-run average properties of systems, we need to compose them with the system and to define their *long-run average outcome*. The synchronous composition with a system is defined as follows.

Definition 7 (product of TPS and experiment) Given a TPS $\Pi = (S, A, p, S_{in}, time, \mathcal{I})$ and an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, we define their *product MDP* $\Pi_\Psi = \Pi \otimes \Psi = (\widehat{S}, \widehat{A}, \widehat{p}, r, w)$ as follows:

- $\widehat{S} = \{\langle s, u\rangle \mid s \models \lambda(u)\}$, where $s \models \lambda(u)$ holds iff the formula $\lambda(u)$ is true in the variable interpretation $\mathcal{I}(s)$ corresponding to s.
- For all $\langle s, u\rangle \in \widehat{S}$, we let $\widehat{A}(\langle s, u\rangle) = A(s)$.
- For all states $\langle s, u\rangle, \langle t, v\rangle \in \widehat{S}$ and actions $a \in A(s)$, the transition probabilities and the labelings r, w are defined as follows.

  Let $v_u^t \in V$ be the unique vertex such that $t \models \lambda(v_u^t)$ and $(u, v_u^t) \in E$. Intuitively, if Π is at s and Ψ at u, and Π takes a transition to t, Ψ will take a transition to v_u^t. We define

$$\widehat{p}_{\langle s,u\rangle\langle t,v\rangle}(a) = \begin{cases} p_{st}(a) & \text{if } v = v_u^t \\ 0 & \text{otherwise.} \end{cases}$$

$$w(\langle s, u\rangle, \langle t, v\rangle) = \begin{cases} 1 & \text{if } (u, v) \in E_r \\ 0 & \text{otherwise.} \end{cases}$$

  If Ψ is a P-experiment, we define

$$r(\langle s, u\rangle, a, \langle t, v\rangle) = \begin{cases} \gamma(u, v) & \text{if } (u, v) \in E_r \\ 0 & \text{otherwise.} \end{cases}$$

  If Ψ is a D-experiment, we define

$$r(\langle s, u\rangle, a, \langle t, v\rangle) = \begin{cases} time(s, a) & \text{if } u \in V_t \\ 0 & \text{otherwise.} \end{cases}$$

Since experiments are total and deterministic, to each $s \in S$ corresponds a *unique* vertex $u \in V_{in}$ such that $\langle s, u\rangle \in \widehat{S}$. We denote this unique vertex by $v_{in}(s)$, for $s \in S$. ∎

In the product MDP, the sum $\sum_{k=0}^{n-1} w(X_k, X_{k+1})$ indicates how many experiments have been completed in the first n steps of a behavior. Similarly, the sum $\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})$ indicates the total outcome for the first n steps. Hence, we can define the n-stage average outcome of experiments as follows.

Definition 8 (n-stage average outcome) Given a behavior ω of a product between a TPS and an experiment, we define the *n-stage average outcome* of ω by

$$\mathcal{H}_n(\omega) = \frac{\sum_{k=0}^{n-1} r(X_k, Y_k, X_{k+1})}{\sum_{k=0}^{n-1} w(X_k, X_{k+1})} .$$

The argument ω in $\mathcal{H}_n(\omega)$ is simply a reminder that $\mathcal{H}_n$ is a random variable, i.e. a measurable function of the behavior. ∎

The long-run average outcome of the experiment on a behavior ω is thus related to $\lim_{n\to\infty} \mathcal{H}_n(\omega)$, or better to $\liminf_{n\to\infty} \mathcal{H}_n(\omega)$ and $\limsup_{n\to\infty} \mathcal{H}_n(\omega)$, since under certain policies the sequence $\{\mathcal{H}_n(\omega)\}_{n\geq 0}$ may oscillate for $n \to \infty$.

4.2 Extending Temporal Logics with Experiments

To enable the specification of long-run average properties of systems, we extend probabilistic or non-probabilistic versions of the temporal logics CTL and CTL* with experiments. The extensions are obtained by introducing two new operators $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$, which are used to express bounds on the average long-run outcome of experiments from given starting states. For $\bowtie \in \{<, \leq, \geq, >\}$ we introduce the following state formulas in the logics.

- If Ψ is a P-experiment and $a \in \mathbb{R}$, then $\overline{\mathrm{P}}_{\bowtie a}(\Psi)$ is a state formula.
- If Ψ is a D-experiment and $a \in \mathbb{R}$, then $\overline{\mathrm{D}}_{\bowtie a}(\Psi)$ is a state formula.

Intuitively, $\overline{\mathrm{P}}_{\bowtie a}(\Psi)$ holds at a state s if, under all policies, a behavior that performs infinitely many experiments yields a long-run average outcome that is $\bowtie a$ with probability 1. The semantics of operator

$\overline{\mathrm{D}}$ is analogous. To define the semantics precisely, we need an auxiliary predicate I that characterizes the behaviors on which the long-run outcome of an experiment is well-defined.

Definition 9 (predicate I) For a behavior ω of a product between a TPS and an experiment, the truth value of predicate I is defined by $\omega \models I$ iff

$$\sum_{k=0}^{\infty} \left[w(X_k, X_{k+1}) + |r(X_k, Y_k, X_{k+1})| \right] = \infty \ . \quad (2)$$

Thus, I holds either if ω spends an infinite amount of time at timed vertices, or if ω performs infinitely many experiments. It is not difficult to show that the truth-set of I is measurable. ∎

The operators $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$ specify bounds on the long-run average outcome only for behaviors that satisfy predicate I, i.e. behaviors that perform infinitely many experiments, or that spend an infinite amount of time at timed vertices. The reason is that the behaviors that do not satisfy I eventually cease to be involved in the active part of the experiment, so that the limit $\lim_{n\to\infty} \mathcal{H}_n(\omega)$ represents a short-term average outcome, rather than a long-run one. This short-term average outcome can be influenced by statistical fluctuations. This will be illustrated later in Example 3.

The semantics of operators $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$ is defined in terms of $\mathcal{H}$ and I as follows.

Definition 10 (semantics of $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$) Given a TPS $\Pi = (S, A, p, S_{in}, \mathit{time}, \mathcal{I})$ and an experiment $\Psi = (V, E, E_r, V_{in}, \lambda)$, let $\Pi_\Psi = \Pi \otimes \Psi$ be the product MDP.

First, we define the semantics of $\overline{\mathrm{P}}$, $\overline{\mathrm{D}}$ on Π_Ψ. For a state $\langle s, v\rangle$ of Π_Ψ, $a \in \mathbb{R}$ and $\bowtie \in \{\geq, >\}$, we have that $\langle s, v\rangle \models \overline{\mathrm{P}}_{\bowtie a}(\Psi)$ (resp. $\langle s, v\rangle \models \overline{\mathrm{D}}_{\bowtie a}(\Psi)$) if, for all policies η,

$$\Pr^{\eta}_{\langle s,v\rangle}\big(I \ \rightarrow \ \liminf_{n\to\infty} \mathcal{H}_n(\omega) \bowtie a\big) = 1 \ . \quad (3)$$

The definition of $\langle s, v\rangle \models \overline{\mathrm{P}}_{\bowtie a}(\Psi)$ (resp. $\langle s, v\rangle \models \overline{\mathrm{D}}_{\bowtie a}(\Psi)$) for $\bowtie \in \{\leq, <\}$ is analogous.

For all states $s \in S$ of Π, the semantics of $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$ is then defined by

$$s \models \Xi_{\bowtie a}(\Psi) \quad \textit{iff} \quad \langle s, v_{in}(s)\rangle \models \Xi_{\bowtie a}(\Psi) \ ,$$

where Ξ is one of $\overline{\mathrm{P}}$, $\overline{\mathrm{D}}$ and $\bowtie \in \{<, \leq, \geq, >\}$. ∎

Example 2 (specifying the call throughput of SHARED-RES) Using the experiment Ψ of Figure 2, we can specify the call throughput requirement Req1 of system SHARED-RES using the formula $\phi_{Req1} : \overline{\mathrm{P}}_{\geq b_0}(\Psi)$. ∎

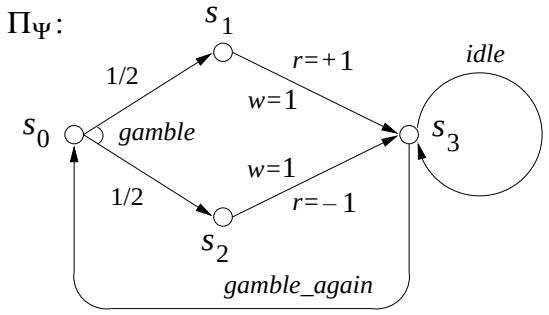

Figure 3: Product MDP $\Pi_\Psi = (S, A, p, w, r)$, corresponding to a gambling system. It is $A(s_0) = \{\mathit{gamble}\}$, $A(s_3) = \{\mathit{idle}, \mathit{gamble\_again}\}$.

The following example illustrates the necessity of excluding from the considerations the behaviors on which predicate I does not hold.

Example 3 Consider a gambling system Π which, upon each gamble, returns a gain of ± 1 with equal probability. After each gamble, the player can either be idle, or gamble again. Figure 3 depicts the MDP resulting from the product of the system with an experiment Ψ that measures the average gain per gamble. Our specification for this system is $\overline{\mathrm{P}}_{\geq -0.3}(\Psi)$. This formula specifies that the long-run average outcome of the experiment (i.e., the long-run average gain per gamble) should be at least -0.3.

Along a behavior that gambles infinitely often, the long-run average gain per gamble is 0 with probability 1. Thus, by (2) and (3), we have $s_0 \models \overline{\mathrm{P}}_{\geq -0.3}(\Psi)$, which agrees with our intuitive understanding of long-run average outcome.

On the other hand, the short-term average outcome might be different from 0. In particular, let η' be the policy that prescribes to gamble exactly 4 times starting from s_0, and then be idle forever. Clearly,

$$\Pr^{\eta'}_{s_0}\big(\liminf_{n\to\infty} \mathcal{H}_n(\omega) = -4\big) = 1/16 \ .$$

Hence, if we dropped the restriction to behaviors satisfying I in (3), we would have $s_0 \not\models \overline{\mathrm{P}}_{\geq -0.3}(\Psi)$.

This example shows that omitting predicate I from (3) would alter drastically the semantics of $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$, making them useless for the specification of long-run average properties. ∎

4.3 Specification Languages and Property Classification

The specification language we propose in this paper embeds experiments into branching-time temporal logic. This might seem a poor fit, since temporal operators and experiments cannot be nested arbitrarily. Two reasons motivated our proposal. First, if the state space of the system under analysis is not strongly connected, the long-run average outcome of experiments can have different values when measured from different states. Temporal logic enables us to specify the set of states from which to measure it.

Moreover, operators $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$ can be combined with other probabilistic extensions of CTL and CTL* to yield powerful languages for the specification of a wide range of correctness, reliability and performance properties. The logics GPTL and GPTL* (*generalized probabilistic temporal logic*) presented in [12] combine the operators $\overline{\mathrm{P}}$ and $\overline{\mathrm{D}}$ with the operators P and D. The operator P, mentioned in Section 3, can express bounds on the probability with which linear-time temporal logic formulas hold; operator D, presented in [13], can express bounds on the expected amount of time required to reach given subsets of states. We call the properties that can be expressed by operators P and D *single-event properties,* to emphasize the fact that they involve the occurrence of a single event (satisfying a linear-time temporal formula, or reaching a subset of states).

The duality between long-run average properties and single-event properties has been mentioned in the introduction: long-run average properties refer to time averages, while single-event properties refer to ensemble averages. This duality is reflected in the different types of questions that arise during verification.

The model checking of GPTL* formula $\mathrm{P}_{\bowtie a}\phi$ involves the construction of the product between the TPS and the deterministic Rabin automaton for ϕ or $\neg\phi$ [13]. Deterministic Rabin automata are strictly more expressive than deterministic Büchi automata [22]. Nonetheless, for the sake of simplicity we consider an algorithm that computes the product with a deterministic Büchi automaton instead. Such an algorithm can be used for the subclass of formulas that can be encoded as deterministic Büchi automata.

In the resulting product structure, to decide whether $s \models \mathrm{P}_{\bowtie a}\phi$, we essentially need to answer the question:

> *What is the probability of visiting the accepting states infinitely often?*

Consider now the specification $\overline{\mathrm{P}}_{\bowtie a}(\Psi)$. In the structure resulting from the product between the system and the experiment, the outcomes are associated with the reset edges of the experiment. The long-run average outcome depends on the relative frequency with which we traverse these edges. Hence, to decide whether $s \models \overline{\mathrm{P}}_{\bowtie a}(\Psi)$, we essentially need to answer the question:

> *If we traverse the reset edges infinitely often, with what relative frequency do we traverse them?*

Hence, we see that there is a direct correspondence between deterministic Büchi automata and experiments: to the accepting states of Büchi automata correspond the reset edges of experiments. P-experiments could in fact be defined as deterministic stutter-invariant edge-Büchi automata with outcomes associated to the accepting edges. The duality between the verification questions corresponding to P and $\overline{\mathrm{P}}$ illustrates the duality between the classes of properties expressed by these operators.

Given the duality between single-event and long-run average properties, it is natural to ask whether there are families of systems whose specifications are better captured by a particular class of properties. While there is no absolute answer to this question, long-run average properties seem to be better suited to the study of systems that have an interesting steady-state behavior. Examples of such systems are communication networks and distributed systems in which no irreversible failures can occur.

Single-event properties are instead suited to the study of systems that have uninteresting steady-state behavior. Several system models used for reliability analysis fall in this category. These models can often reach an irreversible "failure" state, in which case the steady-state distribution is degenerate. The properties of interest are related to the probability or expected time to reach the failure state from given sets of states.

5 Verification of Long-Run Average Properties

In this section, we present a model-checking algorithm to determine the truth value of formulas of the form $\overline{\mathrm{P}}_{\bowtie a}(\Psi)$ and $\overline{\mathrm{D}}_{\bowtie a}(\Psi)$ at all states of the product between a TPS and an experiment, where $\bowtie \in \{<, \leq, \geq, >\}$ and $a \in \mathbb{R}$. The algorithm relies on new results on the theory of Markov decision processes, and on a connection with optimization problems for semi-Markov decision processes [12]. The correctness proof for the algorithms is fairly complex, and

has been presented in [12, §6]: here, we will only provide a partial and informal justification.

We note that for purely probabilistic systems, it is possible to use a simpler algorithm, based on the computation of the steady-state distribution of Markov chains [12, §6].

To present the algorithms, we need the preliminary notions of *sub-MDP* and *end component* [12].

5.1 End Components

End components are the analogous concept in Markov decision processes of the closed recurrent classes of Markov chains [20]: they represent the set of states and actions that can be repeated infinitely often along a behavior with non-zero probability.

Definition 11 (sub-MDPs and end components) Given an MDP $\Pi = (S, A, p)$, a *sub-MDP* is a pair (C, D), where $C \subseteq S$ and D is a function that associates with each $s \in C$ a set $D(s) \subseteq A(s)$ of actions. A sub-MDP (C, D) is an *end component* if the following conditions hold:

- *Closure:* for all $s \in C$, $a \in D(s)$, and $t \in S$, if $p_{st}(a) > 0$ then $t \in C$.
- *Connectivity:* Let

$$E = \big\{(s,t) \in C \times C \;\big|\; \exists a \in D(s) \,.\, p_{st}(a) > 0\big\} \;;$$

then, the graph (C, E) is strongly connected.

We say that an end component (C, D) is contained in a sub-MDP (C', D') if

$$\{(s,a) \mid s \in C \wedge a \in D(s)\} \subseteq \{(s,a) \mid s \in C' \wedge a \in D'(s)\} \,.$$

We say that an end component (C, D) is *maximal* in a sub-MDP (C', D') if there is no other end component (C'', D'') contained in (C', D') that properly contains (C, D). We denote by $maxEC(C', D')$ the set of maximal end components of (C', D'). ∎

It is not difficult to see that, given a sub-MDP (C, D), the set $maxEC(C, D)$ can be computed in time polynomial in $|C| + \sum_{s \in C} |D(s)|$ using simple graph algorithms; an algorithm to do so is given in [12, §3].

Given a behavior ω, let

$$C_\omega = \{s \mid \overset{\infty}{\exists}\, k \,.\, X_k = s\}$$

$$D_\omega(s) = \{a \mid \overset{\infty}{\exists}\, k \,.\, X_k = s \wedge Y_k = a\} \,,$$

where $\overset{\infty}{\exists}\, k$ stands for "there are infinitely many different k's". The sub-MDP (C_ω, D_ω) corresponds to the states and actions that are repeated infinitely often along ω. The proof of the following result can be found in [12].

Theorem 1 (fundamental theorem of end components) *For all $s \in S$ and all policies η, $\Pr^\eta_s((C_\omega, D_\omega)$ is an end component$) = 1$.*

5.2 The Model-Checking Algorithm

Consider an MDP $\Pi = \Pi_0 \otimes \Psi = (S, A, p, r, w)$ resulting from the synchronous product of a TPS Π_0 with experiment Ψ.

We define the *threshold outcomes* $\bar{\mathrm{T}}^+_s$ and $\bar{\mathrm{T}}^-_s$ of Π to be, respectively, the maximum and minimum values of the long-run average outcome of Ψ that can be attained with non-zero probability under some policy starting from state s.

Definition 12 (threshold outcomes) For all $s \in S$, we define the *threshold outcome* $\bar{\mathrm{T}}^+_s$ by

$$\bar{\mathrm{T}}^+_s = \sup\big\{a \in \mathbb{R} \;\big|\; \exists \eta \,.\, \Pr^\eta_s\big(I \wedge \limsup_{n \to \infty} \mathcal{H}_n(\omega) \geq a\big) > 0\big\} \,.$$

The threshold outcome $\bar{\mathrm{T}}^-$ is defined similarly. We use the conventions $\sup \emptyset = -\infty$, $\inf \emptyset = +\infty$. ∎

The truth value of $\overline{\mathrm{P}}_{\bowtie a}(\Psi)$ and $\overline{\mathrm{D}}_{\bowtie a}(\Psi)$ at all $s \in S$ can be computed by comparing the threshold outcomes with a, as stated by the following theorem.

Theorem 2 *For $\Xi \in \{\overline{\mathrm{P}}, \overline{\mathrm{D}}\}$ and $\bowtie \in \{\leq, <\}$, we have*

$$s \models \Xi_{\bowtie a}(\Psi) \text{ iff } \bar{\mathrm{T}}^+_s \bowtie a \,.$$

A similar result holds for $\bowtie \in \{\geq, >\}$ and $\bar{\mathrm{T}}^-_s$.

The following algorithm computes the threshold outcomes. It uses Algorithms 2, 3 and 4, which will be described later.

Algorithm 1 (threshold outcomes)

Input: MDP $\Pi = (S, A, p, r, w)$.

Output: $\bar{\mathrm{T}}^+_s$ and $\bar{\mathrm{T}}^-_s$ for all $s \in S$.

Method:

1. Compute the labelings W and R, defined by

$$W(s,a) = \sum_{t \in S} p_{st}(a)\, w(s,t)$$

$$R(s,a) = \sum_{t \in S} p_{st}(a)\, r(s,a,t)$$

for all $s \in S$ and $a \in A(s)$. Denote by $\Pi' = (S, A, p, W, R)$ the MDP obtained by replacing r and w with R and W, respectively. The purpose of this step is to simplify the notation.

2. Let $\{(S_1, A_1), \ldots, (S_n, A_n)\} = maxEC(S, A)$. For each $1 \leq i \leq n$, construct an MDP $\Pi_i = (S_i, A_i, p^i, R_i, W_i)$, where p^i, R_i, W_i are the restrictions of p, R, W to (S_i, A_i). For $s \in S$, let

$$M_s = \{i \in [1..n] \mid S_i \textit{ reachable in } \Pi' \textit{ from } s\} .$$

By Theorem 1, for a behavior ω there is with probability 1 an $i \in [1..n]$ such that $(C_\omega, D_\omega) \subseteq (S_i, A_i)$, i.e. ω is eventually confined to (S_i, A_i). If a behavior ω satisfies I (the case of interest), the limit $\liminf_{n\to\infty} \mathcal{H}_n(\omega)$ does not depend on any initial prefix of ω (and similarly for $\limsup_{n\to\infty} \mathcal{H}_n(\omega)$).

Let $\bar{\mathrm{T}}^+_{t,i}$, $\bar{\mathrm{T}}^-_{t,i}$ be the threshold outcomes associated with state $t \in S_i$, computed with respect to the MDP Π_i. Since for $1 \leq i \leq n$ each Π_i is strongly connected, it is possible to prove that $\bar{\mathrm{T}}^+_{t,i}$ and $\bar{\mathrm{T}}^-_{t,i}$ do not depend on $t \in S_i$, so that we can write simply $\bar{\mathrm{T}}^+_i$ and $\bar{\mathrm{T}}^-_i$. From the above considerations it follows that

$$\bar{\mathrm{T}}^+_s = \max_{i \in M_s} \bar{\mathrm{T}}^+_i \qquad \bar{\mathrm{T}}^-_s = \min_{i \in M_s} \bar{\mathrm{T}}^-_i \ . \tag{4}$$

Hence, to solve the model-checking problem it suffices to compute $\bar{\mathrm{T}}^+_i$ and $\bar{\mathrm{T}}^-_i$ for all $1 \leq i \leq n$.

3. Compute the set

$$\mathcal{L} = \big\{ i \in [1..n] \;\big|\; \exists s \in S_i \,.\, \exists a \in A_i(s) \,.\, [R_i(s,a) > 0 \vee W_i(s,a) > 0] \big\} .$$

If $i \notin \mathcal{L}$, the behaviors that are confined to Π_i do not satisfy I. Hence, for $i \in \{1, \ldots, n\} - \mathcal{L}$ let

$$\bar{\mathrm{T}}^+_i = -\infty \qquad \bar{\mathrm{T}}^-_i = +\infty \ . \tag{5}$$

4. Transform, using Algorithm 2, each $\Pi_i = (S_i, A_i, p^i, R_i, W_i)$, $i \in \mathcal{L}$, into an MDP $\widetilde{\Pi}_i = (\widetilde{S}_i, \widetilde{A}_i, \widetilde{p}^i, \widetilde{R}_i, \widetilde{W}_i)$ such that the predicate I holds with probability 1, for all policies.

Let $\widetilde{\mathrm{T}}^+_i$, $\widetilde{\mathrm{T}}^-_i$ be the threshold outcomes computed on the MDPs $\widetilde{\Pi}_i$, for $i \in \mathcal{L}$. For $i \in \mathcal{L}$, it can be shown that:

$$\bar{\mathrm{T}}^+_i = \widetilde{\mathrm{T}}^+_i \qquad \bar{\mathrm{T}}^-_i = \widetilde{\mathrm{T}}^-_i \ . \tag{6}$$

5. Compute, using Algorithm 3, the sets

$$K^- = \{i \in \mathcal{L} \mid \widetilde{\mathrm{T}}^-_i = +\infty\} \tag{7}$$

$$K^+ = \{i \in \mathcal{L} \mid \widetilde{\mathrm{T}}^+_i = +\infty\} \ . \tag{8}$$

6. For every policy, predicate I holds with probability 1 on $\widetilde{\Pi}_i$, for $i \in \mathcal{L}$. This enables us to disregard predicate I when working on $\widetilde{\Pi}_i$, leading to a connection between the computation of $\widetilde{\mathrm{T}}^+_i$, $\widetilde{\mathrm{T}}^-_i$ and the solution of an optimization problem for semi-Markov MDPs [12].

Algorithm 4 exploits this connection to compute $\widetilde{\mathrm{T}}^+_i$ (resp. $\widetilde{\mathrm{T}}^-_i$) for all $i \notin K^+$ (resp. $i \notin K^-$). The threshold outcomes $\bar{\mathrm{T}}^+_s$ and $\bar{\mathrm{T}}^-_s$ in Π at all $s \in S$ can be computed by (4), (5), (6), (7), (8). ∎

5.3 Eliminating non-I Behaviors

Algorithm 2 transforms an MDP into a related MDP on which predicate I holds with probability 1. The idea of the algorithm is the following. From Theorem 1, predicate I can be false with positive probability iff the MDP contains a (reachable) end component all of whose state-action pairs have $R = W = 0$. By eliminating these end components, we can insure that I holds with probability 1. The offending end components are eliminated by collapsing each of them to a single state, and by removing all the actions belonging to the end component. Since the state-action pairs that are collapsed have $R = W = 0$, it is possible to prove that the transformation leaves the threshold outcomes unchanged, as stated in (6).

Algorithm 2 (I-transformation)

Input: MDP $\Pi = (S, A, p, R, W)$.

Output: MDP $\widetilde{\Pi} = (\widetilde{S}, \widetilde{A}, \widetilde{p}, \widetilde{R}, \widetilde{W})$.

Method: For each $s \in S$, let

$$D(s) = \big\{ a \in A(s) \;\big|\; R(s,a) = 0 \wedge W(s,a) = 0 \big\}$$

be the set of actions associated with s that have R and W-labels equal to 0. Also, let

$$\{(B_1, D_1), \ldots, (B_n, D_n)\} = maxEC(S, D) \ .$$

The MDP $\widetilde{\Pi}$ is obtained from Π by collapsing each EC (B_i, D_i) into a single state $\widetilde{s}_i$, for $1 \leq i \leq n$. The new set of states is given by $\widetilde{S} = S \cup \{\widetilde{s}_1, \ldots, \widetilde{s}_n\} - \bigcup_{i=1}^n B_i$. The action sets are then defined as follows.

- For $s \in S - \bigcup_{i=1}^n B_i$, $\widetilde{A}(s) = \{\langle s, a\rangle \mid a \in A(s)\}$.

- For $1 \leq i \leq n$:

$$\widetilde{A}(\widetilde{s}_i) = \left\{ \langle s, a \rangle \;\middle|\; s \in B_i \wedge a \in A(s) - D(s) \right\} .$$

For $s \in \widetilde{S}$, $t \in S - \bigcup_{i=1}^{n} B_i$, $1 \leq i \leq n$ and $\langle u, a \rangle \in \widetilde{A}(s)$, the transition probabilities and the labelings R, W are defined by

$$\widetilde{p}_{st}(\langle u, a \rangle) = p_{ut}(a) \qquad \widetilde{p}_{s,\widetilde{s}_i}(\langle u, a \rangle) = \sum_{t \in B_i} p_{ut}(a)$$

$$\widetilde{R}(s, \langle u, a \rangle) = R(u, a) \qquad \widetilde{W}(s, \langle u, a \rangle) = W(u, a) . \quad \blacksquare$$

5.4 Computation of Convergent MDPs

Algorithm 3 (convergent MDPs)

Input: Set $\mathcal{L}$.

Output: Sets K^- and K^+, defined as in (7), (8).

Method: For each $i \in \mathcal{L}$:

- For $s \in S_i$, let $B(s) = \{a \in \widetilde{A}_i(s) \mid \widetilde{W}_i(s, a) = 0\}$ be the set of actions having $\widetilde{W}_i = 0$. Let

$$\{(C_1, D_1), \ldots, (C_m, D_m)\} = maxEC(\widetilde{S}_i, B) .$$

  Then, $i \in K^+$ iff there are $j \in [1..m]$, $s \in C_j$ and $a \in D_j(s)$ such that $\widetilde{R}_i(s, a) > 0$.

- $i \in K^-$ iff both of the following conditions hold:
  - for all $s \in \widetilde{S}_i$ and $a \in \widehat{A}_i(s)$, $\widetilde{W}_i(s, a) = 0$;
  - there are $s \in \widetilde{S}_i$ and $a \in \widehat{A}_i(s)$ such that $\widetilde{R}_i(s, a) > 0$. $\blacksquare$

5.5 Computation of Threshold Outcomes

The following algorithm computes threshold outcomes on strongly connected MDPs on which predicate I holds with probability 1.

Algorithm 4 (computation of $\widetilde{\mathrm{T}}_i^+$, $\widetilde{\mathrm{T}}_i^-$) If $i \in K^-$, consider the following linear programming problem, with variables λ, $\{h_s\}_{s \in \widetilde{S}_i}$: Maximize λ subject to

$$h_s \leq \widetilde{R}_i(s, a) - \lambda \widetilde{W}_i(s, a) + \sum_{t \in \widetilde{S}_i} \widetilde{p}^i_{st}(a)\, h_t \qquad (9)$$

for all $s \in \widetilde{S}_i$ and $a \in \widetilde{A}_i(s)$. Then, all optimal solutions to the problem share the same value for λ, and this value is equal to $\widetilde{\mathrm{T}}_i^-$ [12].

For $i \in K^+$, the outcome $\widetilde{\mathrm{T}}_i^+$ can be computed by solving a similar linear programming problem, in which the direction of the inequality in (9) is reversed, and λ is minimized. $\blacksquare$

5.6 Correctness and Complexity

The following theorem provides results on the correctness and the complexity of the model-checking procedure.

Theorem 3 *Algorithm 1 correctly computes the threshold outcomes, and it has time-complexity polynomial in $l\,|S| \sum_{s \in S} |A(s)|$, where l is the length of the fixed-precision binary numbers used to encode the transition probabilities.*

If the labels of the experiment vertices are written in appropriately restricted sub-languages of first-order logic, the time-complexity of the verification process is polynomial both in the size of the TPS and in the size of the experiment.

Acknowledgments. We thank Orna Kupferman for many valuable comments on drafts of this manuscript.

References

[1] M. Ajmone Marsan, G. Balbo, and G. Conte. A class of generalized stochastic Petri nets for the performance analysis of multiprocessor systems. *ACM Trans. Comp. Sys.*, 2(2):93–122, May 1984.

[2] A. Aziz, V. Singhal, F. Balarin, R.K. Brayton, and A.L. Sangiovanni-Vincentelli. It usually works: The temporal logic of stochastic systems. In *Computer Aided Verification*, volume 939 of *LNCS*, Springer-Verlag, 1995.

[3] M. Bernardo, N. Busi, and R. Gorrieri. A distributed semantics for EMPA based on stochastic contextual nets. *Computer J.*, 38(7):492–509, 1995.

[4] M. Bernardo, L. Donatiello, and R. Gorrieri. Giving a net semantics to Markovian process algebra. In *Proc. 6th Int. Workshop on Petri Nets and Performance Models*, pages 169–178. IEEE Comput. Soc. Press, 1995.

[5] M. Bernardo and R. Gorrieri. Extended Markovian process algebra. In *CONCUR'96: Concurrency Theory. 7th Int. Conf.*, volume 1119 of *LNCS*, pages 315–330. Springer-Verlag, 1996.

[6] D.P. Bertsekas. *Dynamic Programming and Optimal Control.* Athena Scientific, 1995.

[7] A. Bianco and L. de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Found. of Software Tech. and Theor. Comp. Sci.*, volume 1026 of *LNCS*, pages 499–513. Springer-Verlag, 1995.

[8] G. Ciardo, J.K. Muppala, and K.S. Trivedi. On the solution of GSPN reward models. *Performance Evaluation*, 12:237–253, 1991.

[9] R. Cleaveland, S.A. Smolka, and A. Zwarico. Testing preorders for probabilistic processes. In *Proc. 19th Int. Colloq. Aut. Lang. Prog.*, volume 623 of *LNCS*, pages 708–719. Springer-Verlag, 1992.

[10] C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *Proc. 29th IEEE Symp. Found. of Comp. Sci.*, 1988.

[11] C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. In *Proc. 17th Int. Colloq. Aut. Lang. Prog.*, volume 443 of *LNCS*, pages 336–349. Springer-Verlag, 1990.

[12] L. de Alfaro. *Formal Verification of Probabilistic Systems.* PhD thesis, Stanford University, 1997. Technical Report STAN-CS-TR-98-1601.

[13] L. de Alfaro. Temporal logics for the specification of performance and reliability. In *Proc. of Symp. on Theor. Asp. of Comp. Sci.*, volume 1200 of *LNCS*, pages 165–176. Springer-Verlag, 1997.

[14] C. Derman. *Finite State Markovian Decision Processes.* Acedemic Press, 1970.

[15] D. Ferrari. *Computer Systems Performance Evaluation.* Prentice-Hall, 1978.

[16] H.N. Götz, U. Herzog, and M. Rettelbach. Multiprocessor and distributed system design: the integration of functional specification and performance analysis using stochastic process algebras. In *PERFORMANCE'93*, volume 729 of *LNCS*, pages 121–146, Springer-Verlag, 1993.

[17] H. Hansson and B. Jonsson. A framework for reasoning about time and reliability. In *Proc. of Real Time Systems Symposium*, pages 102–111. IEEE, 1989.

[18] H. Hansson and B. Jonsson. A logic for reasoning about time and probability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[19] J. Hillston. *A Compositional Approach to Performance Modelling.* Distinguished Dissertations Series. Cambridge University Press, 1996.

[20] J.G. Kemeny, J.L. Snell, and A.W. Knapp. *Denumerable Markov Chains.* D. Van Nostrand Company, 1966.

[21] M. Kwiatkowska and C. Baier. Model checking for a probabilistic branching time logic with fairness. Technical Report CSR-96-12, University of Birmingham, June 1996.

[22] L.H. Landweber. Decision problems for ω–automata. *Mathematical Systems Theory*, 3:376–384, 1969.

[23] K.G. Larsen and A. Skou. Bisimulation through probabilistic testing (preliminary report). In *Proc. 16th ACM Symp. Princ. of Prog. Lang.*, pages 344–352, 1989.

[24] Z. Liu, A.P. Ravn, E.V. Sorensen, and C.C. Zhou. A probabilistic duration calculus. In *Proc. of the Second Intl. Workshop on Responsive Computer Systems*, pages 14–27. Springer-Verlag, 1992.

[25] M.K. Molloy. *On the Integration of Delay and Throughput Measure In Distributed Processing Models.* PhD thesis, UCLA, Los Angeles, 1981.

[26] S. Natkin. *Les Reseaux de Petri Stochastiques et leur Application a l'Evaluation des Systemes Informatiques.* PhD thesis, CNAM, Paris, 1980.

[27] A. Pnueli and L. Zuck. Probabilistic verification by tableaux. In *Proc. First IEEE Symp. Logic in Comp. Sci.*, pages 322–331, 1986.

[28] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, MIT, 1995. Technical Report MIT/LCS/TR-676.

[29] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. In *CONCUR'94: Concurrency Theory. 5th Int. Conf.*, volume 836 of *LNCS*, pages 481–496. Springer-Verlag, 1994.

[30] F.J.W. Symons. Introduction to numerical Petri nets, a general graphical model for concurrent processing systems. *Australian Telecommunications Research*, 14(1):28–33, January 1980.

[31] M.Y. Vardi. Automatic verification of probabilistic concurrent finite-state systems. In *Proc. 26th IEEE Symp. Found. of Comp. Sci.*, pages 327–338, 1985.