Formal Verification of Performance and Reliability of Real-Time Systems*

Luca de Alfaro

Department of Computer Science
Stanford University

Abstract

In this paper we propose a methodology for the specification and verification of performance and reliability properties of real-time systems within the framework of temporal logic. The methodology is based on the system model of *stochastic real-time systems* (SRTSs), and on branching-time temporal logics that are extensions of the probabilistic logics pCTL and pCTL*. SRTSs are discrete-time transition systems that can model both probabilistic and nondeterministic behavior. The specification language extends the branching-time logics pCTL and pCTL* by introducing an operator to express bounds on the average time between events. We present model-checking algorithms for the algorithmic verification of system specifications, and we discuss their complexity.

*This research was supported in part by the National Science Foundation under grant CCR-92-23226, the Advanced Research Projects Agency under NASA grant NAG2-892, the United States Air Force Office of Scientific Research under grant F49620-93-1-0139, and the Department of the Army under grant DAAH04-95-1-0317.

1 Introduction

Temporal logic has proved to be a valuable tool for the formal analysis of systems, and has been applied to the specification and verification of a wide range of system properties, from safety and response of reactive systems to reliability of probabilistic systems. In this paper, we extend the range of its application to include the study of performance properties of real-time systems, and we propose a unified methodology for the specification and verification of both performance and reliability properties of these systems within the framework of temporal logic. The methodology we introduce relies upon a probabilistic model of the system, a specification language based on temporal logic, and model-checking algorithms for the verification of system specifications.

The system model we propose is that of *Stochastic Real-Time Systems* (SRTSs). Similar to the models of [HJ89, Han94, Seg95], SRTSs can describe the probabilistic and nondeterministic components of the system behavior. The probabilistic characterization provides information about the transition probabilities and the discrete-time delay distributions of the transitions; nondeterminism can be used to model systems with incomplete probabilistic characterizations, and enables us to consider reactive modules as components of SRTSs. Unlike other system models, based on automata or augmented Markov chains [HJ89, Seg95], SRTSs consist of a collection of transitions defined over a state space, together with mechanisms for choosing and scheduling transitions for execution. This approach, derived from [MP93], leads to compact representations of concurrent probabilistic systems.

The specification of system properties is based on the logics pTL and pTL*, and on the use of *instrumentation clocks* to measure the length of intervals of time. The logics pTL and pTL*, extensions of pCTL and pCTL* [HJ89, ASB+95, BdA95], are obtained by adding the probabilistic operators P and D to the branching-time temporal logics CTL and CTL*. The operator P is used to express bounds on the probability of system behaviors, and the operator D is used to express bounds on the average time between events. For example, the formula $\mathrm{A}\Box[\mathrm{P}_{\geq a}(\phi \triangleright \psi)]$ specifies that whenever a transition satisfying ϕ is taken, the ensuing system behavior satisfies ψ with probability at least a [HJ89, ASB+95]. Similarly, the formula $\mathrm{A}\Box[\mathrm{D}_{<b}(\phi \triangleright \gamma)]$ specifies that whenever a transition satisfying ϕ is taken, the average time needed to reach a state satisfying γ is less than b. The instrumentation clocks, related to the clocks used in timed automata [AD91], are stopped or reset depending on the transitions taken by the system, and their value can be used in the logic to reason about the timing behavior of the system.

We present model-checking algorithms that enable us to verify whether an SRTS satisfies a property written in pTL or pTL*. For model checking the P operator, we give an alternative algorithm to the one presented in [CY90], based on the automata-theoretic constructions of [Saf88] and on the properties of *stable* sets of states. These sets are related to the closed classes of Markov chains, and are a unifying concept for all our model-checking algorithms. By using the properties of stable sets, we reduce model checking for the D operator to the computation of minimum and maximum first-passage costs in Markov decision processes [Der70], and we give criteria for the convergence of these costs in our setting.

We show that model checking of full pTL and pTL* is a hard problem, equivalent to the problem of computing the minimum expected total cost of a Markov decision process, which is not known to be solvable in polynomial time in the size of the system. On the other hand, we show that model checking of the restricted logics $\mathrm{pTL}_<$ and $\mathrm{pTL}^*_<$, in which the D operator is restricted to its upper-bound forms $D_{<b}$ and $D_{\leq b}$, has polynomial time-complexity in the size of the probabilistic description of the state space.

Related Work

The first applications of temporal logic to probabilistic systems studied which temporal logic properties are satisfied with probability 1 by systems modeled either as finite Markov chains [LS82, Pnu83, HS84, PZ93] or as augmented Markov models exhibiting both nondeterministic and probabilistic behavior [Var85, PZ86, CY88, PZ93, CY95]. This approach has been extended to real-time systems in [ACD92], where the specification is represented by a timed automaton.

Subsequently, [HJ89, HJ94] considered a restricted class of real-time systems, in which the amount of time elapsed is equated to the number of transitions taken by the Markov chain, and introduced the probabilistic logic pCTL. The specifications are written in the logic pCTL, which extends CTL by introducing the P operator and adding subscripts to the temporal operators to denote time intervals, similarly to metric temporal logic [Koy90]. The model-checking algorithms they present are derived from results from the theory of Markov chains.

The logic pCTL*, the corresponding extension of CTL*, has been studied in [ASB+95], which proposes model-checking algorithms for Markov chains derived from ideas in [CY88]. The work of [ASB+95] also considers generalized Markov processes, representing families of Markov chains, and shows that the model-checking problem for pCTL* formulas on generalized Markov processes is decidable using results from the theory of real closed fields. However, no efficient computational method is given for this latter problem.

The work of [Han94] constitutes the first study of probabilistic properties of general real-time systems within the framework of temporal logic. The systems are modeled by process algebra terms that are translated into real-time extensions of the concurrent Markov chains of [Var85], and the specifications are expressed in pCTL. This work also discusses properties related to the average behavior of real-time systems, and presents algorithms to compute some quantities of interest. However, no logic is proposed for the specification of these properties, and the algorithms can be applied only if the systems can be transformed to eliminate nondeterminism.

The work of [BdA95] studies the logics pCTL and pCTL* for systems exhibiting both probabilistic and nondeterministic behavior, and proposes model-checking algorithms based on canonical forms for linear-time temporal formulas which, unlike those of [Han94], have polynomial time-complexity in the size of the description of the system. Algorithms for pCTL and pCTL* model checking based on automata theory can be derived from [CY90].

In the study of [Seg95], real-time systems are modeled by timed probabilistic automata, and the verification is based on the notions of probabilistic simulation and bisimulation. The study also provides a broad overview of work on probabilistic systems done in the context of process algebras. The relationship between these notions of simulation and temporal logics related to pCTL is examined in [SL94].

2 Stochastic Real-Time Systems

The system model we use, Stochastic Real-Time Systems, is derived from the transition systems of [MP93] and the probabilistic real-time systems of [ACD92].

Definition 1 (SRTS) A *Stochastic Real-Time System* (SRTS) is a quadruple $A = (\mathcal{V}, S, s_{in}, \mathcal{T})$ consisting of the following components.

1. A set $\mathcal{V}$ of state variables, where each variable has its own type.
2. A finite *state space* S: each state $s \in S$ is a type-consistent value assignment of all the variables in $\mathcal{V}$. For every $x \in \mathcal{V}$, we denote by $s(x)$ the value of x at state s.
3. An *initial state* $s_{in} \in S$.
4. A set $\mathcal{T}$ of *transitions*. Each transition $\tau \in \mathcal{T}$ is a partial function $\tau : S \mapsto S$ with domain $\mathrm{Dom}(\tau)$. If $s \in \mathrm{Dom}(\tau)$, we say that τ is *enabled* at s. Each transition $\tau \in \mathcal{T}$ has an associated *weight* $w_\tau \in \mathbb{R}^+ \cup \{\perp\}$, where $\mathbb{R}^+$ is the set of positive real numbers and $\perp$ is a value used to denote an unspecified weight. Weights are used to choose the transition to be taken, and unspecified weights are used to model nondeterministic choices.

   The set $\mathcal{T}$ is partitioned into the sets $\mathcal{T}_i$ of *immediate* transitions and $\mathcal{T}_d$ of *delayed* transitions. Immediate transitions can be taken as soon as they are enabled; delayed transitions must wait for an amount of time specified by their waiting-time distributions.

   We distinguish three types of delayed transitions, depending on whether the probability distribution of waiting times is unknown, is geometrical or has finite support. Accordingly, we partition $\mathcal{T}_d$ into the sets $\mathcal{T}_u$, $\mathcal{T}_g$, $\mathcal{T}_f$. To each $\tau \in \mathcal{T}_g$ we associate the instantaneous probability q_τ with which τ is taken when enabled; to each $\tau \in \mathcal{T}_f$ we associate a maximum delay u_τ and a waiting-time distribution $f_\tau : \mathbb{N} \mapsto [0, 1]$ such that $f_\tau(0) = 0$, $f_\tau(n) = 0$ for $n > u_\tau$, and $\sum_{i=1}^{u_\tau} f_\tau(i) = 1$. ∎

A computation of an SRTS consists of an alternation of *delayed phases*, in which delayed transitions are taken, and *immediate phases*, in which immediate transitions are taken. The model of time we choose is that of discrete time, and a time step is composed of a delayed phase and of the following immediate phase. Time does not advance within the phases, and is advanced by one unit at the end of the immediate phase that concludes a time step.

Each delayed transition is taken at most once during a time step. At the beginning of each delayed phase, a subset $\mathcal{T}_s \subseteq \mathcal{T}_d$ of delayed transitions is scheduled for execution. The set $\mathcal{T}_s$ is chosen so that every delayed transition is scheduled with the probability dictated by its waiting-time distribution and by the amount of time for which it has been enabled.

During the delayed phase, the transitions in $\mathcal{T}_s$ are chosen for execution, taken, and removed from $\mathcal{T}_s$. The choice is made according to the weights $\{w_\tau\}_{\tau \in \mathcal{T}_s}$. Transitions $\tau \in \mathcal{T}_s$ with $w_\tau = \perp$ are chosen nondeterministically; transitions with $w_\tau \neq \perp$ are taken with probability proportional to w_τ. Once taken, the transition is removed from $\mathcal{T}_s$, along with all the transitions that have become disabled as a consequence. This process continues until $\mathcal{T}_s$ is empty, at which point the delayed phase is concluded. Thus, if two scheduled transitions are independent, they are both taken (and the order in which they are taken depends on their weights); if the transitions are dependent (i.e. if they can disable each other), the weights determine which ones are taken.

During the immediate phase, the immediate transition to be taken is again chosen among the enabled ones according to the transition weights. Every immediate transition can be taken any number of times during an immediate phase, and the phase itself continues as long as there are immediate transitions enabled. When no enabled immediate transition is left, the immediate phase is concluded, the time is advanced by one unit, and the computation of the SRTS continues with the following delayed phase.

The first phase of the execution of an SRTS is an immediate phase: this enables the use of immediate transitions to set up the initial conditions of the system before the first time step.

The two-phase execution semantics of SRTSs can be used to model systems consisting of a physical system controlled by a software module. The physical system can be modeled by delayed transitions, and the software module by immediate ones. At each time step, certain system transitions occur, after which the controlling software computes the next control output. This model is appropriate whenever the reaction time of the control software is negligible compared to the time unit used in the description of the physical system. The scheduling and choice mechanisms for the transitions also enable us to compose SRTSs in parallel by taking the union of their sets of transitions, simplifying the task of modeling complex concurrent systems.

To give a formal definition of the semantics of an SRTS, we provide a translation from SRTSs to Probabilistic Nondeterministic Systems.

2.1 Probabilistic Nondeterministic Systems

A Probabilistic Nondeterministic System (PNS) is a Markov decision process without the costs associated to the actions [Der70, Put94], and generalizes a Markov chain by introducing nondeterminism in the system model.

Definition 2 (PNS) A PNS is a quadruple $\Pi = (\mathcal{V}, S, s_{in}, \kappa, p)$, where:

1. $\mathcal{V}$ is the set of state variables.
2. S is the state space of the system. The value of $x \in \mathcal{V}$ at $s \in S$ is denoted by $s(x)$.
3. $s_{in} \in S$ is the initial state.
4. κ is a function that associates with each $s \in S$ the set $\kappa(s) = \{a_1, \ldots, a_{k_s}\}$ of *actions* that can be taken at state s.
5. p is a function that associates with each $s, t \in S$ and $a \in \kappa(s)$ the probability $p(t \mid s, a)$ of a transition from s to t under action a. For all $s \in S$ and $a \in \kappa(s)$, we require $\sum_{t \in S} p(t \mid s, a) = 1$. We let $\text{Supp}(s, a) = \{t \in S \mid p(t \mid s, a) > 0\}$. ∎

In a computation of a PNS, the successor of a state $s \in S$ is chosen in two steps: first, an action $a \in \kappa(s)$ is selected nondeterministically; second, a successor state $t \in S$ is chosen according to the probability distribution $p(t \mid s, a)$. A Markov chain is thus a PNS s.t. $|\kappa(s)| = 1$ for all $s \in S$. We define a *reachability* relation $\rho \subseteq S \times S$ by $\rho = \{(s, t) \mid \exists a \in \kappa(s) \,.\, p(t \mid s, a) > 0\}$. Then, we associate with each state $s \in S$ the set

$$\Omega_s = \{s_0 s_1 s_2 \cdots \mid s = s_0 \wedge \forall n \in \mathbb{N} \,.\, \rho(s_n, s_{n+1})\}$$

of *behaviors* from s. Given a behavior $\omega \in \Omega_s$, we denote by ω_i the i-th state of ω, with $\omega_0 = s$, and we denote by $\omega_{\geq i}$ the behavior $\omega_i \omega_{i+1} \omega_{i+2} \cdots$. For each $s \in S$, we let $\mathcal{B}_s \subseteq 2^{\Omega_s}$

be the σ-algebra of *measurable* subsets of Ω_s, following the classical definition of [KSK66]. To be able to talk about the probability of system behaviors, we would like to associate to each $\Delta \in \mathcal{B}_s$ its probability measure $\mu(\Delta)$. However, this measure is not well-defined, since the probability that a behavior $\omega \in \mathcal{B}_s$ belongs to Δ may depend on the criterion by which the actions are chosen.

To represent these choice criteria, we use the concept of *policy* [Der70, Put94] (see also [Var85, PZ86, Han94, BdA95]). A policy η is a set of conditional probabilities $Q_\eta(a \mid s_0 s_1 \cdots s_n)$, where $a \in \kappa(s_n)$.[1] A policy dictates the probabilities with which the actions are chosen: according to policy η, after the finite prefix $s_0 \cdots s_n$ starting at the root $s = s_0$ of Ω_s, action $a \in \kappa(s_n)$ is chosen with probability $Q_\eta(a \mid s_0 s_1 \cdots s_n)$. Thus, the probability of a direct transition to $t \in S$ after $s_0 \cdots s_n$ is given by

$$\Pr_\eta(t \mid s_0 \cdots s_n) = \sum_{a \in \kappa(s_n)} p(t \mid s, a)\, Q_\eta(a \mid s_0 s_1 \cdots s_n) \ .$$

These transition probabilities give rise to a unique probability measure $\mu_{s,\eta}$ on $\mathcal{B}_s$. We write $\Pr_{s,\eta}(A)$ to denote the probability of event A in Ω_s under policy η and probability measure $\mu_{s,\eta}$, and we adopt the usual conventions to denote conditional probabilities and expectations [KSK66].

A policy is *Markovian* if $Q_\eta(a \mid s_0 s_1 \cdots s_n) = Q_\eta(a \mid s_n)$ for all $s_0, s_1, \ldots, s_n \in S$ and all $a \in \kappa(s_n)$. A policy η is *deterministic* if it is Markovian, and if for all $s \in S$ there is exactly one action $a \in \kappa(s)$ for which $Q_\eta(a \mid s) = 1$. Given a Markovian policy η, the PNS under η gives rise to a Markov chain with transition probabilities $p_{st} = \sum_{a \in \kappa(s)} p(t \mid s, a) Q_\eta(a \mid s)$, for all $s, t \in S$.

2.2 Translation of SRTS into PNS

The semantics of an SRTS $A = (\mathcal{V}, S, s_{in}, \mathcal{T})$ is defined by translating A into a PNS $\Pi_A = (\hat{\mathcal{V}}, \hat{S}, \hat{s_{in}}, \kappa, p)$, and by taking the set of computations of the SRTS A to be $\Omega_{\hat{s_{in}}}$. The translation consists of several steps.

State space. The first step consists in extending the set of variables $\mathcal{V}$ of the SRTS A to $\hat{\mathcal{V}} = \mathcal{V} \cup \{d, e\} \cup \{c_\tau \mid \tau \in \mathcal{T}_f\}$, where:

1. d is an integer variable that has value 1 at the beginning of each delayed phase (when the delayed transitions have to be scheduled), and value 0 otherwise;
2. e is a variable having $2^{\mathcal{T}_d}$ as the set of possible values: the value $\hat{s}(e) \subseteq \mathcal{T}_d$ represents the set of scheduled transitions at $\hat{s}$.
3. For each $\tau \in \mathcal{T}_f$, c_τ is a clock that counts the amount of time for which τ has been enabled; its possible values are $\{0, \ldots, u_\tau\}$.

The state space $\hat{S}$ consists of all the possible value assignments to the variables in $\hat{\mathcal{V}}$. The initial state $\hat{s_{in}}$ is s.t. $\hat{s_{in}}(c_\tau) = 0$ for all $\tau \in \mathcal{T}_f$, $\hat{s_{in}}(x) = s_{in}(x)$ for all $x \in \mathcal{V}$, and $\hat{s_{in}}(d) = 0$, $\hat{s_{in}}(e) = \emptyset$ (i.e. execution starts at the beginning of an immediate phase).

[1] A more general definition of policy enables the choice of the next action to depend also on the sequence of previous actions [Put94]. This level of generality is unnecessary for the present work, in which the behavior of the system is considered to be fully described by the sequence of system states.

Extending the transitions. The next step consists in extending each transition $\tau \in \mathcal{T} : S \mapsto S$ to a transition $\hat{\tau} : \hat{S} \mapsto \hat{S}$. Let $\hat{s}$, $\hat{s}'$ be two states of $\hat{S}$, and let s, s' be their restrictions to the variables in $\mathcal{V}$. Then $\hat{\tau}(\hat{s}) = \hat{s}'$ iff all of the following conditions hold:

1. $\tau(s) = s'$;
2. if $\tau \in \mathcal{T}_d$ then $\tau \in \hat{s}(e)$, and if $\tau \in \mathcal{T}_i$ then $\hat{s}(e) = \emptyset$;
3. $\hat{s}'(d) = 0$;
4. $\hat{s}'(e) = \{\tau' \in \hat{s}(e) \mid s' \in \mathrm{Dom}\,(\tau') \wedge s' \neq \tau'(s)\}$;
5. for all $\tau' \in \mathcal{T}_f$, if $s' \in \mathrm{Dom}\,(\tau') \wedge s' \neq \tau'(s)$ then $\hat{s}'(c_{\tau'}) = \hat{s}(c_{\tau'})$, else $\hat{s}'(c_{\tau'}) = 0$.

We let $\widehat{\mathcal{T}} = \{\hat{\tau} \mid \tau \in \mathcal{T}\}$. By Condition 2, a delayed transition can be taken only when it is scheduled, and an immediate transition can be taken only when no delayed transitions are scheduled (i.e. during an immediate phase). Condition 4 deschedules all transitions that are taken or disabled as a result of taking $\hat{\tau}$; Condition 5 resets the clocks of the transitions with finite support waiting-times distributions that have been descheduled.

Constructing the action sets. Finally, we construct $\kappa(\hat{s})$ for each $\hat{s} \in \hat{S}$.

Scheduling the delayed transitions. If $\hat{s}(d) = 1$, we are at the beginning of a delayed phase, and we must schedule for execution a subset of the set $\mathcal{T}_d$ of delayed transitions. Let s be the restriction of $\hat{s}$ to the variables in $\mathcal{V}$, and let $\mathcal{T}_n = \mathcal{T}_g \cup \mathcal{T}_f$ be the set of transitions with known delay distributions. A transition $\tau \in \mathcal{T}_n$ should be scheduled with a probability r_τ given by:

$$\tau \in \mathcal{T}_g : \; r_\tau = \begin{cases} q_\tau & \text{if } s \in \mathrm{Dom}\,(\tau), \\ 0 & \text{otherwise;} \end{cases} \qquad \tau \in \mathcal{T}_f : \; r_\tau = \frac{f_\tau(\hat{s}(c_\tau))}{\sum_{i=\hat{s}(c_\tau)}^{u_\tau} f_\tau(i)} \; . \tag{1}$$

The scheduling of the transitions takes place in two steps. First, a subset $T \subseteq \mathcal{T}_u$ of transitions with unknown delay distributions is chosen nondeterministically; then, a subset $T' \subseteq \mathcal{T}_n$ is chosen according to the probabilities given in (1). The transitions in $T \cup T'$ are then scheduled. Accordingly, $\kappa(\hat{s}) = \{a_T \mid T \subseteq \mathcal{T}_u\}$, where each action a_T schedules a subset $T \subseteq \mathcal{T}_u$. Scheduling a set of transitions $T \cup T'$ leads from $\hat{s}$ to the single state $\hat{s}' = \gamma(\hat{s}, T, T')$ such that

$$\hat{s}'(d) = 0 \qquad \hat{s}'(e) = T \cup T' \qquad \forall x \in \mathcal{V}' - \{d, e\} \,.\, \hat{s}'(x) = \hat{s}(x) \; .$$

The probability distribution corresponding to action a_T is given by

$$p(\hat{s}' \mid \hat{s}, a_T) = \begin{cases} \Big(\prod_{\tau \in T'} r_\tau\Big)\Big(\prod_{\tau \in \mathcal{T}_n - T'} (1 - r_\tau)\Big) & \text{if } \hat{s}' = \gamma(\hat{s}, T, T') \text{ for } T' \subseteq \mathcal{T}_n; \\ 0 & \text{otherwise.} \end{cases}$$

Choosing from $\widehat{\mathcal{T}}$ the transition to be taken. If $\hat{s}(d) = 0$ and the set $T = \{\hat{\tau} \in \widehat{\mathcal{T}} \mid \hat{s} \in \mathrm{Dom}\,(\hat{\tau})\}$ of transitions enabled at $\hat{s}$ is non-empty, let $U = \{\hat{\tau} \in T \mid w_{\hat{\tau}} = \bot\}$ be the enabled transitions having undefined weight, and let $D = T - U$. The system makes a nondeterministic choice, choosing either a single transition from U, or the set D as a whole, to cause the next state transition. If a single transition $\hat{\tau} \in U$ is chosen, the successor of $\hat{s}$ is $\hat{\tau}(\hat{s})$; if the set D is chosen, the transitions in D are combined according to their weights, yielding a probability distribution for the successor of $\hat{s}$. The set $\kappa(\hat{s})$ will thus consist of one action $a_{\hat{\tau}}$ for each $\hat{\tau} \in U$ and, if $D \neq \emptyset$, of one additional action a_D.

For each $\hat{\tau} \in U$, we let $p(\hat{s}' \mid \hat{s}, a_{\hat{\tau}}) = \mathbb{D}_{\hat{\tau}(\hat{s})}(\hat{s}')$, where $\mathbb{D}_x(y)$ is the distribution having value 1 if $x = y$ and 0 otherwise. If the state transition is caused by transitions in D, we must combine the probabilities of the transitions that cause the same state change. Each transition $\hat{\tau} \in D$ is taken with probability $w_{\hat{\tau}} / \sum_{\hat{\tau} \in D} w_{\hat{\tau}}$. Denoting with $D(\hat{s}, \hat{s}') = \{\hat{\tau} \in D \mid \hat{s}' = \hat{\tau}(\hat{s})\}$ the subset of transitions in D that lead from $\hat{s}$ to $\hat{s}'$, we let

$$p(\hat{s}' \mid \hat{s}, a_D) = \frac{\sum_{\hat{\tau} \in D(\hat{s}, \hat{s}')} w_{\hat{\tau}}}{\sum_{\hat{\tau} \in D} w_{\hat{\tau}}} .$$

Advancing the clocks. If $\hat{s}(d) = 0$ and no transition $\hat{\tau} \in \widehat{\mathcal{T}}$ is enabled at $\hat{s}$, the system is at the end of an immediate phase, and the clocks of the enabled transitions in $\mathcal{T}_f$ are advanced. We let $\kappa(\hat{s}) = \{a_{clk}\}$, where $p(\hat{s}' \mid \hat{s}, a_{clk}) = \mathbb{D}_{\hat{t}'}(\hat{s}')$, where $\hat{t}'$ is the unique state in $\hat{S}$ such that

1. $\hat{t}'(d) = 1$;
2. $\hat{t}'(e) = \emptyset$;
3. for all $\tau \in \mathcal{T}_f$, if $s \in \mathrm{Dom}(\tau)$ then $\hat{t}'(c_\tau) = \max\{\hat{s}(c_\tau) + 1, u_\tau\}$, else $\hat{t}'(c_\tau) = 0$;
4. for all $x \in \mathcal{V}$, $\hat{t}'(x) = \hat{s}(x)$.

Condition 1 signals the beginning of a new delayed phase once the clocks have been advanced. Condition 3 advances the clocks of the enabled transitions in $\mathcal{T}_f$. The max in Condition 3 serves no purpose if Π_A is constructed in such a way as to include only the states reachable from $\hat{s_{in}}$: if no $\hat{\tau}$ is enabled at a reachable $\hat{s}$, it is $\hat{s}(c_\tau) < u_\tau$.

It is easy to modify the above translation so that states not reachable from $\hat{s_{in}}$ are not included in $\hat{S}$. In the following, we assume that the state space $\hat{S}$ of Π_A contains only reachable states.

3 Specification

The specification of SRTS properties is based on the use of *instrumentation clocks* and on temporal logics derived from pCTL and pCTL* [HJ89, ASB+95, BdA95].

3.1 Instrumentation Clocks and Instrumented PNS

An *instrumentation clock* δ over a state space S is a pair $(\delta_\mathcal{T}, \delta_s)$, where $\delta_\mathcal{T} \subseteq S^2$ is a transition relation dictating when the clock is reset, and $\delta_s \subseteq S$ is a set of states on which the clock is advanced alongside the other system clocks. Thus, δ measures the amount of time spent by the system in δ_s since the last $\delta_\mathcal{T}$-transition.

Given an SRTS $A = (\mathcal{V}, S, s_{in}, \mathcal{T})$ and a set C of instrumentation clocks over S, we construct an *instrumented* PNS $\Pi_{A,C}$, whose states keep track of the value of the clocks. The construction starts from the PNS $\Pi_A = (\hat{\mathcal{V}}, \hat{S}, \hat{s_{in}}, \kappa, p)$, from which we construct $\Pi_{A,C} = (\mathcal{V}^*, S^*, s^*_{in}, \kappa^*, p^*)$ as follows.

1. $\mathcal{V}^* = \hat{\mathcal{V}} \cup C$, where each $\delta \in C$ is an integer variable, so that $S^* = S \times \mathbb{N}^{|C|}$.

2. $s^*_{in}(x) = \hat{s_{in}}(x)$ for $x \in \hat{\mathcal{V}}$, and $s^*_{in}(\delta) = 0$ for $\delta \in C$.
3. For any state $s^* \in \mathcal{V}^*$, let $\hat{s}$, s be its restrictions to the variables in $\hat{\mathcal{V}}$, $\mathcal{V}$ respectively. We let $\kappa^*(s^*) = \kappa(\hat{s})$, and for $a \in \kappa^*(s^*)$, $p^*(t^* \mid s^*, a) = p(\hat{t} \mid \hat{s}, a)$ if the following conditions are satisfied for all $\delta \in C$:

$$t^*(d) = 1 \ : \qquad \text{if } t \in \delta_s \text{ then } t^*(\delta) = s^*(\delta) + 1 \text{ else } t^*(\delta) = s^*(\delta) \tag{2}$$

$$t^*(d) = 0 \ : \qquad \text{if } (s,t) \in \delta_{\mathcal{T}} \text{ then } t^*(\delta) = 0 \text{ else } t^*(\delta) = s^*(\delta). \tag{3}$$

If one of (2), (3) is not satisfied, $p^*(t^* \mid s^*, a) = 0$.

Note that if $C \neq \emptyset$, the state space of $\Pi_{A,C}$ is infinite.

3.2 Probabilistic Temporal Logic

Syntax. The system specifications are written in the logic pTL and pTL*. We distinguish three classes of pTL and pTL* formulas: the class *Stat* of *state formulas* (whose truth value is evaluated on the states), the class *Trans* of *transition formulas* (expressing relationships between states) and the class *Seq* of *sequence formulas* (whose truth value is evaluated on infinite sequences of states). To define these classes, we first define the set *BaStat* of *basic state formulas*. Given an SRTS $A = (\mathcal{V}, S, s_{in}, \mathcal{T})$ and a set C of instrumentation clocks, the set *BaStat* contains:

1. all formulas built using propositional connectives, interpreted function and predicate symbols on the set $\mathcal{V}$ of variables;
2. all formulas of the form $\delta > b$, $\delta = b$, where $\delta \in C$ and b is an integer constant.

The class *Trans* contains all formulas built using propositional connectives, interpreted function and predicate symbols on the set $\mathcal{V} \cup \mathcal{V}'$ of variables, where $\mathcal{V}'$ denotes the set obtained by priming each variable of $\mathcal{V}$. For pTL*, the classes *Stat* and *Seq* are defined inductively.

State formulas:

$$\phi \in \mathit{BaStat} \implies \phi,\ \neg\phi \in \mathit{Stat} \qquad \phi \in \mathit{Trans}, \psi \in \mathit{Seq} \implies \mathrm{P}_{\bowtie b}(\phi \rhd \psi) \in \mathit{Stat} \tag{4}$$

$$\phi,\ \psi \in \mathit{Stat} \implies \phi \wedge \psi \in \mathit{Stat} \qquad \phi \in \mathit{Trans}, \psi \in \mathit{Stat} \implies \mathrm{D}_{\bowtie b'}(\phi \rhd \psi) \in \mathit{Stat} \tag{5}$$

$$\phi \in \mathit{Seq} \implies \mathrm{A}\phi,\ \mathrm{E}\phi \in \mathit{Stat}\,. \tag{6}$$

Sequence formulas:

$$\phi \in \mathit{Stat} \implies \phi,\ \neg\phi,\ \Box\phi,\ \Diamond\phi \in \mathit{Seq} \qquad \phi,\ \psi \in \mathit{Seq} \implies \phi \wedge \psi,\ \phi\,\mathcal{U}\,\psi \in \mathit{Seq}\,. \tag{7}$$

In the above definition, $\bowtie$ stands for one of $\{<, \leq, \geq, >\}$, $b \in [0,1]$ and $b' \geq 0$. The logic pTL is a restricted version of pTL*, and its definition can be obtained by replacing the clauses (7) with the single clause

$$\phi,\ \psi \in \mathit{Stat} \implies \Box\phi,\ \Diamond\phi,\ \phi\,\mathcal{U}\,\psi \in \mathit{Seq}\,. \tag{8}$$

Semantics. The truth value of pTL and pTL* formulas is defined with respect to the instrumented PNS $\Pi_{A,C} = (\mathcal{V}^*, S^*, s^*_{in}, \kappa^*, p^*)$. For $\phi \in \mathit{Stat}$, $\psi \in \mathit{Trans}$, $\gamma \in \mathit{Seq}$, we indicate with $s \models \phi$, $(s, s') \models \psi$, $\omega \models \gamma$ their satisfaction on $s \in S^*$, $(s, s') \in S^* \times S^*$, $\omega \in \bigcup_{s \in S^*} \Omega_s$.

For a basic state formula ϕ and $s \in S^*$, $s \models \phi$ iff ϕ is true in the interpretation in which every $x \in \mathcal{V}$ is assigned the value $s(x)$. For $\phi \in \mathit{Trans}$ and $s, s' \in S^*$, $(s, s') \models \phi$ iff ϕ is true in the interpretation that assigns to $x \in \mathcal{V}$ the value $s(x)$, and to $x' \in \mathcal{V}'$ the value $s'(x)$. The semantics of the logical connectives, temporal operators and path quantifiers is defined in the usual way.

Given $\phi \in \mathit{Trans}$ and $s \in S$, let $A_{s,\phi} = \{s' \in S \mid \rho^*(s, s') \wedge (s, s') \models \phi\}$ be the set of states reachable from s by a ϕ-transition. The truth value of $s \models \mathrm{P}_{\bowtie b}(\phi \triangleright \psi)$ is defined by

$$s \models \mathrm{P}_{\bowtie a}(\phi \triangleright \psi) \quad \textit{iff} \quad A_{s,\phi} = \emptyset \ \textit{or} \ \ \forall \eta \,.\, \mathrm{Pr}_{s,\eta}\Big(\omega_{\geq 1} \models \psi \;\Big|\; \omega \in \Omega_s \wedge \omega_1 \in A_{s,\phi}\Big) \bowtie b \,. \quad (9)$$

The intuitive meaning of this definition is that $\mathrm{P}_{\bowtie b}(\phi \triangleright \psi)$ holds at $s \in S$ if, after taking the transition ϕ, a behavior has probability $\bowtie b$ of satisfying ψ, regardless of the policy guiding the nondeterministic behavior of the system.

Given $s \in S$ and $\psi \in \mathit{Stat}$, we define $T_{\omega,\psi} = \min\{i \mid \omega_i \models \psi\}$ to be the first position at which ψ holds along ω, with $T_{\omega,\psi} = \infty$ if $\forall i \,.\, \omega_i \not\models \psi$. Let $H_{\omega,\psi} = \sum_{i=1}^{T_{\omega,\psi}} \omega_i(d)$ be the number of time steps along ω before $T_{\omega,\psi}$. The truth value of $s \models \mathrm{D}_{\bowtie b'}(\phi \triangleright \psi)$ is defined by:

$$s \models \mathrm{D}_{\bowtie b}(\phi \triangleright \psi) \quad \textit{iff} \quad A_{s,\phi} = \emptyset \ \textit{or} \ \ \forall \eta \,.\, \mathrm{E}_{s,\eta}\Big\{H_{\omega_{\geq 1},\psi} \;\Big|\; \omega \in \Omega_s \wedge \omega_1 \in A_{s,\phi}\Big\} \bowtie b' \quad (10)$$

where $\mathrm{E}_{s,\eta}\{\cdot\}$ denotes as usual the expectation with respect to the measure $\mu_{s,\eta}$. The intuitive meaning of this definition is that $\mathrm{D}_{\bowtie b'}(\phi \triangleright \psi)$ holds at $s \in S$ if, after transition ϕ, the PNS reaches a ψ-state in average time $\bowtie b'$, regardless of the policy guiding the nondeterministic behavior of the system.

To see that the semantics is well defined, it is possible to show by induction on the structure of formulas that the events mentioned in the above definitions are measurable, and that $T_{\omega,\psi}$ is a random time and that $H_{\omega,\psi}$ is a random variable.

Definition 3 We say that an instrumented PNS $\Pi_{A,C}$ having initial state s^*_{in} satisfies $\phi \in \mathit{Stat}$, written $\Pi_{A,C} \models \phi$, if $s^*_{in} \models \phi$. We say that an SRTS A with instrumentation clocks C satisfies ϕ, written $A, C \models \phi$, if $\Pi_{A,C} \models \phi$. ∎

3.3 Invariance Theorems and Non-Zeno Systems

In the above definition of $\Pi_{A,C} \models \phi$, while $\Pi_{A,C}$ must contain all the clocks mentioned by ϕ, it can also contain additional instrumentation clocks. Our first invariance theorem states that the presence in $\Pi_{A,C}$ of instrumentation clocks not mentioned by ϕ does not influence the satisfaction of ϕ. Given a formula ϕ of pTL or pTL*, let $\mathit{clocks}(\phi)$ be the set of instrumentation clocks mentioned in ϕ.

Theorem 1 *For any SRTS A, $\phi \in \mathit{Stat}$, set C of clocks s.t. $\mathit{clocks}(\phi) \subseteq C$ and clock $\delta \notin C$, $\Pi_{A,C} \models \phi$ iff $\Pi_{A,C \cup \{\delta\}} \models \phi$.*

Since we are interested in working with the smallest possible PNS, we will assume that for $\Pi_{A,C}$ it is $C = \mathit{clocks}(\phi)$, i.e. that $\Pi_{A,C}$ contains only the instrumentation clocks mentioned

m	State of A_1	l	state of A_2
0	idle	0	waiting for item
1	item in production	1	item received
2	item ready to send	2	ack sent, item being consumed
3	waiting for acknowledge	3	item consumed

Table 1: States of A_1 and A_2 as encoded by the variables l and m.

in ϕ. In general, the PNS $\Pi_{A,C}$ has an infinite state space, and it is not possible to verify directly whether $\Pi_{A,C} \models \phi$ using conventional model-checking algorithms. However, we can construct a reduced PNS $\Pi^{\phi}_{A,C}$ such that $\Pi_{A,C} \models \phi$ iff $\Pi^{\phi}_{A,C} \models \phi$. The key observation is that if M_δ is the largest constant with which δ is compared to in ϕ, we need to keep track of the value of δ only up to $M_\delta + 1$, since no inequality of ϕ changes truth value when the value of δ increases beyond $M_\delta + 1$. The PNS $\Pi^{\phi}_{A,C}$ is constructed from Π_A, C and $\{M_\delta \mid \delta \in C\}$ by using the construction of Section 3.1, substituting condition (2) with, for all $\delta \in C$:

$$t^*(d) = 1 \;: \qquad \text{if } t \in \delta_s \text{ then } t^*(\delta) = 1 + \max(s^*(\delta), M_\delta) \text{ else } t^*(\delta) = s^*(\delta). \qquad (11)$$

Note that $\Pi^{\phi}_{A,C}$ has a finite state space. Our second invariance theorem states that $A, C \models \phi$ can be computed using $\Pi^{\phi}_{A,C}$ instead of $\Pi_{A,C}$.

Theorem 2 $\Pi_{A,C} \models \phi$ *iff* $\Pi^{\phi}_{A,C} \models \phi$.

Non-Zeno systems. We say that a PNS $\Pi_A = (\mathcal{V}, S, s_{in}, \kappa, p)$ corresponding to an SRTS A is *non-Zeno* if a behavior from s_{in} has probability 1 of going through infinitely many time steps, under any policy. Formally, we require that $\Pr_{s_{in},\eta}(\sum_{i=0}^{\infty} \omega_i(d) = \infty \mid \omega \in \Omega_{s_{in}}) = 1$ for any policy η. In general, we are only interested in non-Zeno systems, and therefore the proof of the above property should precede the proof of any other system specification. We will present algorithms to check whether a PNS is non-Zeno.

4 A Specification Example

Consider a producer-consumer system composed of two processes A_1 and A_2 communicating over a channel. The system structure is depicted in Figure 1. The states of A_1 and A_2 are represented by the variables l and m, respectively, as detailed in Table 1. The communication channel is composed of two *send buffers* s_1, s_2 and two *receive buffers* r_1 and r_2 (see Figure 1). The variables corresponding to the buffers have value 0 when the buffer is empty, and 1 when it contains a message. The initial state s_{in} of the system corresponds to $l = m = 0$ and $s_1 = s_2 = r_1 = r_2 = 0$. The transitions of the system are summarized in Table 2. For brevity, we do not specify the probability distributions of the waiting times of the delayed transitions.

We add an instrumentation clock δ having δ_s defined by the formula *true*, and δ_τ defined by the formula $s_1 = 0 \wedge s'_1 = 1$, so that δ measures the amount of time elapsed since an item has been placed in the send buffer of A_1. Then, the property: *"If A_1 sends an item,*

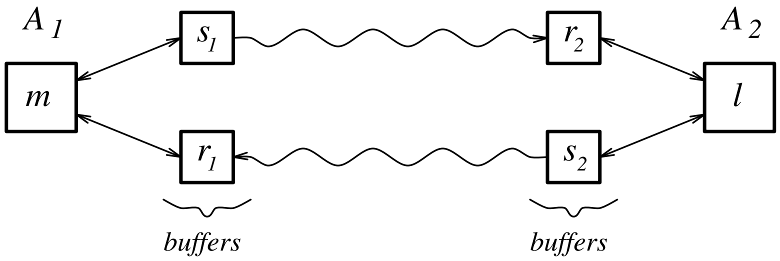

Figure 1: A producer-consumer system, consisting of a producer A_1 and a consumer A_2 connected by a bidirectional communication channel. The channel consists of two send buffers and two receive buffers. Inside each system component we indicate the variable used to represent the state of the component. The lines represent interaction between the components: the solid lines represent local interactions, the wavy lines represent communication over the channel.

Transitions of A_1:

Name	Type	Domain	Transition Formula
τ_{on}	$\in \mathcal{T}_u \subseteq \mathcal{T}_d$	$l = 0$	$l' = 1$
$\tau_{produce}$	$\in \mathcal{T}_f \subseteq \mathcal{T}_d$	$l = 1$	$l' = 2$
$\tau_{senditem}$	$\in \mathcal{T}_i$	$l = 2 \wedge s_1 = 0$	$l' = 3 \wedge s_1' = 1$
τ_{getack}	$\in \mathcal{T}_i$	$l = 3 \wedge r_1 = 1$	$l' = 0 \wedge r_1' = 0$

Transitions of A_2:

Name	Type	Domain	Transition Formula
$\tau_{getitem}$	$\in \mathcal{T}_i$	$m = 0 \wedge r_2 = 1$	$m' = 1 \wedge r_2' = 0$
$\tau_{giveack}$	$\in \mathcal{T}_i$	$m = 1 \wedge s_2 = 0$	$m' = 2 \wedge s_2' = 1$
$\tau_{consume}$	$\in \mathcal{T}_f \subseteq \mathcal{T}_d$	$m = 2$	$m' = 3$
τ_{reset}	$\in \mathcal{T}_i$	$m = 3$	$m' = 0$

Channel Transitions:

Name	Type	Domain	Transition Formula
τ_{1to2}	$\in \mathcal{T}_g \subseteq \mathcal{T}_d$	$s_1 = 1 \wedge r_2 = 0$	$s_1' = 0 \wedge r_2' = 1$
τ_{2to1}	$\in \mathcal{T}_g \subseteq \mathcal{T}_d$	$s_2 = 1 \wedge r_1 = 0$	$s_2' = 0 \wedge r_1' = 1$

Table 2: Transitions of the producer-consumer system. Each transition is specified by a formula that defines its domain, and by a transition formula that describes the state change due to the transition. In the transition formula, primed variables refer to the values of the state variables after the transition; for brevity, variables whose value is not changed by the transition are not mentioned.

A_1 *receives an acknowledge after an average delay less than 7 time units"* is expressed by the pTL formula

$$\mathrm{A}\Box\Big[\mathrm{D}_{\leq 7}\Big((s_1 = 0 \wedge s_1' = 1) \triangleright m = 0\Big)\Big] . \tag{12}$$

The property: *"If A_1 sends an item, then with probability at least 0.6 A_2 finishes the processing within 8 time units"* is expressed by the pTL* formula:

$$\mathrm{A}\Box\Big[\mathrm{P}_{\geq 0.6}\Big((s_1 = 0 \wedge s_1' = 1) \triangleright s_1 = 1\, \mathcal{F}\, r_2 = 1\, \mathcal{F}\, m = 1\, \mathcal{F}\, m = 2\, \mathcal{F}\, (m = 3 \wedge \delta \leq 8)\Big)\Big] , \tag{13}$$

where we have used the abbreviation $\mathcal{F}$ for the *non-skipping until*:

$$\phi\, \mathcal{F}\, \psi \quad := \quad \phi \wedge (\phi\, \mathcal{U}\, \psi)$$

which, like $\mathcal{U}$, associates to the right.

The motivation for transition formulas. Previous probabilistic logics, such as pCTL and pCTL* of [HJ89, ASB+95], do not include transition formulas in their semantics, and apply the operator P directly to a path formula. When reasoning about behavior probabilities, it is essential to specify precisely the initial conditions from which the behaviors originate. Often, these initial conditions correspond to transitions of the system; for example, in the above system the interesting starting conditions coincide with the sending of a message. Our syntax can easily encode these conditions, without requiring the introduction of auxiliary variables indicating when a transition has just been taken.

Moreover, our syntax can express properties that cannot be expressed in pCTL or pCTL*. To see this, consider a system having a state s_0 with three possible successors s_1, s_2, s_3. Assume that the state formulas $\phi_0, \ldots, \phi_3$ characterize these states, so that $\gamma : \phi_0 \wedge (\phi_1' \vee \phi_2') \in \mathit{Trans}$ characterizes the transition from s_0 to $\{s_1, s_2\}$. It is easy to see that the property $\mathrm{P}_{\bowtie a}(\gamma \triangleright \psi)$, for $\psi \in \mathit{Seq}$, cannot be expressed directly in pCTL*.

5 Model Checking

We now present algorithms to decide whether a PNS $\Pi^{\phi}_{A,C} = (\mathcal{V}, S, s_{in}, \kappa, p)$ with finite state space S is non-Zeno, and whether it satisfies a specification ϕ written in pTL or pTL*.

The model checking algorithms share the same basic structure of those proposed in [ES84, EL85] for CTL and CTL*. Given a formula $\phi \in \mathit{Stat}$, they recursively evaluate the truth values of the state subformulas $\psi \in \mathit{Stat}$ of ϕ at all states $s \in S$, starting from the basic state formulas of ϕ and following the recursive definitions (4)–(6) of state formulas, until the truth value of ϕ itself can be computed at all $s \in S$. Since the logics pTL and pTL* are obtained by adding the P and D operators to CTL and CTL*, we need to examine only the cases corresponding to these additional operators. For brevity, we will consider only the logic pTL*, since pTL model checking can be done by combining the results of [BdA95] with the methods presented below for the D operator.

We begin with a discussion of the properties of certain subsets of states of a PNS, called *stable sets*. The properties of these sets will be instrumental in the development of the model-checking algorithms.

5.1 Stable Sets

Intuitively, a subset of the state space of a PNS is *stable* if there is a policy such that all behaviors that enter the subset will never leave it [BdA95]. The definition is as follows.

Definition 4 (stable sets) Given a PNS $\Pi = (\mathcal{V}, S, s_{in}, \kappa, p)$, a subset $B \subseteq S$ is *stable* if, for all $s \in B$, there is $a \in \kappa(s)$ such that $p(t \mid s, a) > 0$ implies $t \in B$. If B is stable, we define

$$\rho_B = \Big\{(s,t) \in B \times B \mid \exists a \in \kappa(s) \,.\, \Big(p(t \mid s,a) > 0 \wedge \mathrm{Supp}(s,a) \subseteq B\Big)\Big\} \,.$$

Thus, relation ρ_B holds between $s, t \in B$ if there is an action $a \in \kappa(s)$ that leads to t with positive probability, and that leads outside of B with probability 0. If the graph (B, ρ_B) is strongly connected, B is said to be a *strongly connected stable set* (SCSS). An SCSS B is *maximal* in $C \subseteq S$ if $B \subseteq C$ and if there is no other SCSS $B' \subseteq C$ such that $B \subseteq B'$. ∎

An equivalent definition of SCSSs is provided by the following remark, that we state without proof (for the definition of recurrent classes, see [KSK66]).

Remark (SCSSs and closed recurrent classes) *A set B of states is an SCSS iff there is a Markovian policy η such that B is a closed recurrent class of the Markov chain arising from η.*

The following lemmas summarize the relevant properties of stable sets and SCSSs.

Lemma 1 *The following assertions hold.*

1. *Let B be a stable set. Then, there is a Markovian policy defined on B such that any behavior that enters B will stay in B forever with probability 1.*
2. *Let B be an SCSS. Then, there is a Markovian policy defined on B such that any behavior that enters B with probability 1 will stay in B forever and will visit all states of B infinitely often.*

Proof. We prove the second assertion, since the first one can be proved similarly. Let η be the Markovian policy defined on B that chooses at state $s \in B$ each action in $\{a \in \kappa(s) \mid \mathrm{Supp}(s,a) \subseteq B\}$ with uniform probability. If a behavior after entering B follows policy η, it will never leave B. Moreover, B under policy η forms a closed ergodic Markov chain, and from classical results on Markov chains it follows that it will visit every state in B infinitely often with probability 1. ∎

Given an infinite behavior ω, we denote by $\mathrm{inft}(\omega)$ the set of states that appears infinitely often along ω. The next lemma states that this set is stable with probability 1, under any policy.

Lemma 2 *For all states s and for all policies η, $\mu_{s,\eta}\Big(\{\omega \in \Omega_s \mid \mathrm{inft}(\omega) \text{ is stable}\}\Big) = 1$.*

Proof. Assume towards the contradiction that $\mu_{s,\eta}\Big(\{\omega \in \Omega_s \mid \mathrm{inft}(\omega) \text{ is stable}\}\Big) < 1$. Then, there is a non-stable set B such that the set $\Omega_s^B = \{\omega \in \Omega_s \mid \mathrm{inft}(\omega) = B\}$ has positive measure under η. Since B is not stable, there is a state $t \in B$ such that $\mathrm{Supp}(t,a) \not\subseteq B$ for all $a \in \kappa(t)$. Let $b = \max_{a \in \kappa(t)} \sum_{t' \in B} p(t' \mid t, a)$ be the maximum probability of staying in B after visiting t; notice that $b < 1$. Since all behaviors in Ω_s^B visits t infinitely often staying in B, we have $\mu_{s,\eta}(\Omega_s^B) \le b^k$ for all $k > 0$, which leads to the desired contradiction $\mu_{s,\eta}(\Omega_s^B) = 0$. ∎

We say that a behavior ω is *eventually confined* to a subset C of the state space if $\mathrm{inft}(\omega) \subseteq C$.

Corollary 1 *Let S be the finite state space of a PNS, and let $C \subseteq S$ be any subset of states. For any state s, the following assertions hold.*

1. *Let B be the maximal stable subset of C. Under any policy η, the probability that a behavior from s is eventually confined to $C - B$ is 0.*
2. *Let $D_1, \ldots, D_n$ be the SCSSs that are maximal in C, and let $D = \bigcup_{i=1}^n D_i$. Under any policy η, the probability that a behavior from s is eventually confined to $C - D$ is 0.*

Proof. We prove the first assertion, since the second one is an easy consequence of the first. Assume, towards the contradiction, that the probability that a behavior is eventually confined to $C - B$ is greater than 0. Then, there is a set $D \subseteq C - B$ such that

$$\mu_{s,\eta}\Big(\{\omega \in \Omega_s \mid \mathrm{inft}(\omega) = D\}\Big) > 0 \ .$$

By Lemma 2, D must be stable. Thus, also $B \cup D$ is stable, contradicting the maximality of B. ∎

Define the size $|\Pi|$ of a PNS $\Pi = (\mathcal{V}, S, s_{in}, \kappa, p)$ to be the length of its encoding, in which we assume that the next-state distributions are represented by lists of rational numbers, each represented as the ratio between two integers. Given a subset $C \subseteq S$, the following algorithm computes the maximal stable set $B \subseteq C$ and the maximal SCSSs in C in time polynomial in $|\Pi|$.

Algorithm 1 (stable sets and SCSSs)

Input: PNS $\Pi = (\mathcal{V}, S, s_{in}, \kappa, p)$, and a subset $C \subseteq S$.

Output: The stable set B maximal in C, and the list$E_1, \ldots, E_n$ of SCSSs maximal in C.

Procedure: Define the functional $\Lambda : 2^S \mapsto 2^S$ by

$$\Lambda(D) = \Big\{ s \in D \ \Big|\ \exists a \in \kappa(s) \ . \ \mathrm{Supp}(s,a) \subseteq D \Big\} \ .$$

Then $B = \lim_{k\to\infty} \Lambda^k(C) = \Lambda^\infty(C)$, and the computation of the limit requires at most $|C|$ iterations, since the functional is monotonic. The SCSSs $E_1, \ldots, E_n$ maximal in C can be computed by computing the maximal strongly connected components of the graph (B, ρ_B). ∎

5.2 Model Checking for the Operator P

Given $s \in S$, $\phi \in \mathit{Trans}$, $\psi \in \mathit{Seq}$, $b \in [0,1]$, we present an algorithm to decide whether $s \models \mathrm{P}_{\bowtie b}(\phi \triangleright \psi)$ holds. As usual, we assume that the truth value of the state subformulas of ψ has already been computed at all states. Let $A_{s,\phi} = \{s' \in S \mid \rho(s,s') \wedge (s,s') \models \phi\}$

be the set of states reachable from s via a ϕ-transition; by definition (9), if $A_{s,\phi} = \emptyset$ then $s \models \mathrm{P}_{\bowtie b}(\phi \rhd \psi)$ holds. If $A_{s,\phi} \neq \emptyset$, let

$$\begin{aligned}
\nu_s^-(\phi \rhd \psi) &= \inf_\eta \mathrm{Pr}_{s,\eta}\Big(\omega_{\geq 1} \models \psi \;\Big|\; \omega \in \Omega_s \wedge \omega_1 \in A_{s,\phi}\Big) \\
&= \inf_\eta \frac{\sum_{t \in A_{s,\phi}} \mathrm{Pr}_{t,\eta}(\omega \models \psi \mid \omega \in \Omega_t)\, \mathrm{Pr}_\eta(t \mid s)}{\sum_{t \in A_{s,\phi}} \mathrm{Pr}_\eta(t \mid s)} \\
&= \inf_\eta \frac{\sum_{t \in A_{s,\phi}} \sum_{a \in \kappa(s)} Q_\eta(a \mid s)\, p(t \mid s,a)\, \mathrm{Pr}_{t,\eta}(\omega \models \psi \mid \omega \in \Omega_t)}{\sum_{t \in A_{s,\phi}} \sum_{a \in \kappa(s)} Q_\eta(a \mid s)\, p(t \mid s,a)} \\
&= \inf_\eta \frac{\sum_{a \in \kappa(s)} Q_\eta(a \mid s) \sum_{t \in A_{s,\phi}} p(t \mid s,a)\, \mathrm{Pr}_{t,\eta}(\omega \models \psi \mid \omega \in \Omega_t)}{\sum_{a \in \kappa(s)} Q_\eta(a \mid s) \sum_{t \in A_{s,\phi}} p(t \mid s,a)} \\
&= \min_{a \in \kappa(s)} \frac{\sum_{t \in A_{s,\phi}} p(t \mid s,a) \inf_\eta \mathrm{Pr}_{t,\eta}(\omega \models \psi \mid \omega \in \Omega_t)}{\sum_{t \in A_{s,\phi}} p(t \mid s,a)} \,. \qquad (14)
\end{aligned}$$

A similar relation holds for $\nu_s^+(\phi \rhd \psi) = \inf_\eta \mathrm{Pr}_{s,\eta}(\omega_{\geq 1} \models \psi \mid \omega \in \Omega_s \wedge \omega_1 \in A_{s,\phi})$. Thus, to compute $\nu_s^-(\phi \rhd \psi)$ and $\nu_s^+(\phi \rhd \psi)$ it suffices to compute

$$\mathrm{Pr}_t^- \psi = \inf_\eta \mathrm{Pr}_{t,\eta}(\omega \models \psi \mid \omega \in \Omega_t) \qquad \mathrm{Pr}_t^+ \psi = \sup_\eta \mathrm{Pr}_{t,\eta}(\omega \models \psi \mid \omega \in \Omega_t) \qquad (15)$$

for all $t \in A_{s,\phi}$. These quantities represent the minimum and maximum probability that a behavior from t satisfies ψ. By the results of [CY90, BdA95] there are policies η^- and η^+ that realize respectively the minimum and maximum values for $\mathrm{Pr}_t^- \psi$ and $\mathrm{Pr}_t^+ \psi$, so that we can substitute min and max to inf and sup in (14) and (15). The truth value of $s \models \mathrm{P}_{\bowtie a}(\phi \rhd \psi)$ can then be computed from $\nu_s^-(\phi \rhd \psi)$ and $\nu_s^+(\phi \rhd \psi)$ by definition (9).

The minimum and maximum probabilities $\mathrm{Pr}_t^- \psi$ and $\mathrm{Pr}_t^+ \psi$ can be computed in time polynomial in $|\Pi_{A,C}^\phi|$ and doubly exponential in $|\psi|$ using an algorithm presented in [CY90]. Below, we present an alternative algorithm, derived from the above properties of stable sets, and from results on the determinization of ω-automata [Saf88]. The running time of the proposed algorithm is again polynomial in $|\Pi_{A,C}^\phi|$ and doubly exponential in $|\psi|$: in fact, it is shown in [CY90] that this is the lower bound for the time-complexity of the problem.

Computation of minimum and maximum probabilities. Let t_0 be a designated state in S, and let ψ be a sequence formula. Since $\mathrm{Pr}_{t_0}^- \psi = 1 - \mathrm{Pr}_{t_0}^+ \neg\psi$, it suffices to provide an algorithm to compute $\mathrm{Pr}_{t_0}^+ \psi$. Let $\alpha_1, \ldots, \alpha_n \in \mathit{Stat}$ be the maximal state subformulas of ψ, i.e. the state subformulas of ψ that are not proper subformulas of any other state subformula of ψ. Define $\psi' = \psi[r_1/\alpha_1] \ldots [r_n/\alpha_n]$ to be the result of replacing each α_i with a new propositional symbol r_i: the formula ψ' is therefore a linear-time temporal formula constructed from $r_1, \ldots, r_n$ using the temporal operators $\Box$, $\Diamond$, $\mathcal{U}$. As the truth value of $\alpha_1, \ldots, \alpha_n$ has already been computed at all states, we can define the *label* $l(s)$ of $s \in S$ by $l(s) = \{r_i \mid 1 \leq i \leq n \wedge s \models \alpha_i\}$.

It is known from automata theory that ψ' can be translated into a deterministic Rabin automaton $DR_{\psi'} = (Q, q_{in}, \Sigma, \delta, U)$ with state space Q, initial state $q_{in} \in Q$, alphabet $\Sigma = 2^{\{r_1,\ldots,r_n\}}$, transition relation $\delta : Q \times \Sigma \mapsto Q$, and acceptance condition U [VW86, Saf88].

The acceptance condition is a list $U = \{(H_1, L_1), \ldots, (H_m, L_m)\}$ of pairs of subsets of Q. An infinite sequence $\sigma : b_0 b_1 b_2 \cdots$ of symbols of Σ is accepted by $DR_{\psi'}$ if it induces a sequence $\omega_\sigma : q_0 q_1 q_2 \cdots$ of states of Q s.t. $q_0 = q_{in}$, $\delta(q_i, b_i) = q_{i+1}$ for all $i \geq 0$ and, for some $1 \leq j \leq m$, it is $\mathrm{inft}(\omega_\sigma) \subseteq H_j$ and $\mathrm{inft}(\omega_\sigma) \cap L_j \neq \emptyset$.

From $\Pi^\phi_{A,C} = (\mathcal{V}, S, s_{in}, \kappa, p)$ and $DR_{\psi'} = (Q, q_{in}, \Sigma, \delta, U)$ we construct the *product PNS* $\Pi' = (\mathcal{V}', S', s'_{in}, \kappa', p')$, where:

1. $\mathcal{V}' = \mathcal{V} \cup \{\xi\}$, where ξ is a new variable with domain Q;
2. $S' = S \times Q$, where $(s, q) \in S'$ is s.t. $(s, q)(\xi) = q$ and $(s, q)(x) = s(x)$ for $x \in \mathcal{V}$;
3. $s'_{in} = (t_0, \delta(q_{in}, l(t_0)))$;
4. for $(s, q) \in S'$, it is $\kappa'(s, q) = \kappa(s)$.
   For each action $a \in \kappa(s)$, the probability $p'((s', q') \mid (s, q), a)$ of a transition to $(s', q') \in S'$ is equal to $p(s' \mid s, a)$ if $\delta'(q, l(s')) = q'$, and is equal to 0 otherwise.

Each pair (H_i, L_i), $1 \leq i \leq m$ induces a related pair (H'_i, L'_i), defined by $H'_i = S \times H_i$, $L'_i = S \times L_i$. For each pair (H'_i, L'_i), $1 \leq i \leq m$, we let $B^{(i)}_1, \ldots, B^{(i)}_{n_i}$ be the SCSS maximal in H'_i and having non-empty intersection with L'_i, and we let $T = \bigcup_{i=1}^m \bigcup_{j=1}^{n_i} B^{(i)}_j$. By the results of the previous section, the set T can be computed in time polynomial in $|\Pi'|$.

Let $R_{\omega,T}$ denote the event $\exists k \,.\, \omega_k \in T$, i.e. the event of ω reaching T. The following theorem states that to compute $\mathrm{Pr}^+_{t_0}\, \phi$ it suffices to compute the maximum probability of reaching T from s'_{in} in Π'. Maximum reachability probabilities can be computed by solving a linear programming problem, using the methods of [Der70, CY90, Put94].

Theorem 3 $\mathrm{Pr}^+_{t_0}\, \psi = \sup_\eta \mathrm{Pr}_{s'_{in},\eta}(R_{\omega,T} \mid \omega \in \Omega_{s'_{in}})$.

Proof. By construction of Π', it is $\mathrm{Pr}^+_{t_0}(\psi) = \sup_\eta \mathrm{Pr}_{s'_{in},\eta}(\omega \models \psi' \mid \omega \in \Omega_{s'_{in}})$. We can write

$$\begin{aligned}\mathrm{Pr}_{s'_{in},\eta}(\omega \models \psi' \mid \omega \in \Omega_{s'_{in}}) &= \mathrm{Pr}_{s'_{in},\eta}\Big(R_{\omega,T} \wedge \omega \models \psi' \;\Big|\; \omega \in \Omega_{s'_{in}}\Big) \\ &\quad + \mathrm{Pr}_{s'_{in},\eta}\Big(\neg R_{\omega,T} \wedge \omega \models \psi' \;\Big|\; \omega \in \Omega_{s'_{in}}\Big) \,. \end{aligned} \tag{16}$$

A behavior $\omega \in \Omega_{s'_{in}}$ that satisfies ψ' must, for some $1 \leq i \leq m$, (a) be eventually confined to H'_i, (b) visit infinitely often L'_i. From (a), by Corollary 1, with probability 1 the behavior is eventually confined to $\bigcup_{j=1}^{k_i} D^{(i)}_j$, where $D^{(i)}_1$, …, $D^{(i)}_{k_i}$ are the SCSSs maximal in H'_i. From (b), the behavior can be eventually confined only to the SCSSs that have non-empty intersection with L'_i, that is, to $B^{(i)}_1$, …, $B^{(i)}_{n_i}$. Since $\bigcup_{j=1}^{n_i} B^{(i)}_j \subseteq T$, a behavior that satisfies ψ' will enter T with probability 1, and the second term on the right side of (16) is 0. Thus, (16) reduces to

$$\mathrm{Pr}_{s'_{in},\eta}(\omega \models \psi' \mid \omega \in \Omega_{s'_{in}}) = \mathrm{Pr}_{s'_{in},\eta}\Big(R_{\omega,T} \wedge \omega \models \psi' \;\Big|\; \omega \in \Omega_{s'_{in}}\Big) \,.$$

Taking $\sup_\eta$ of both sides and using Lemma 1, we have

$$\begin{aligned}\sup_\eta \mathrm{Pr}_{s'_{in},\eta}(\omega \models \psi' \mid \omega \in \Omega_{s'_{in}}) &= \sup_\eta \mathrm{Pr}_{s'_{in},\eta}\Big(R_{\omega,T} \wedge \omega \models \psi' \;\Big|\; \omega \in \Omega_{s'_{in}}\Big) \\ &= \sup_\eta \mathrm{Pr}_{s'_{in},\eta}(R_{\omega,T} \mid \omega \in \Omega_{s'_{in}}) \,,\end{aligned}$$

and the result follows. ∎

5.3 Checking Non-Zenoness

To check whether the PNS $\Pi^{\phi}_{A,C} = (\mathcal{V}, S, s_{in}, \kappa, p)$ is non-Zeno, we convert $\Pi^{\phi}_{A,C}$ into a *Markov decision process* (MDP) $\Pi_{mdp} = (\mathcal{V}, S, s_{in}, \kappa, p, c)$ by associating to each pair $\{(s,a)\}_{s\in S, a\in\kappa(s)}$ a cost $c(s,a)$ [Der70, Put94]. Intuitively, the cost of a pair (s,a) corresponds to the amount of time elapsed during the action: all actions that advance the clocks have cost 1, while all others have cost 0. More precisely, for all $s \in S$ and $a \in \kappa(s)$,

$$c(s,a) = \begin{cases} 1 & \text{if } s(d) = 0 \wedge \exists t \in S \,.\, (p(t \mid s,a) = 1 \wedge t(d) = 1); \\ 0 & \text{otherwise.} \end{cases}$$

To check for non-Zenoness, we introduce the notion of *zero-cost* stable sets.

Definition 5 (zero-cost stable sets) Given an MDP $\Pi_{mdp} = (\mathcal{V}, S, s_{in}, \kappa, p, c)$, a *zero-cost stable set* (ZCSS) is a stable subset $B \subseteq S$ such that for every $s \in B$, there is an action $a \in \kappa(s)$ with $c(s,a) = 0$ and $\text{Supp}(s,a) \subseteq B$. A zero-cost stable set is *maximal* if it is not the proper subset of any other zero-cost stable set. ∎

Note that since the union of two ZCSSs is still a ZCSS, every MDP has a single (possibly empty) maximal ZCSS. The following algorithms computes this set in time polynomial in the size of the MDP.

Algorithm 2 (computation of maximal zero-cost stable set)

Input: An MDP $(\mathcal{V}, S, s_{in}, \kappa, p, c)$.

Output: The maximal ZCSS B of the MDP.

Procedure: Define the functional $\Lambda : 2^S \mapsto 2^S$ by

$$\Lambda(D) = \left\{ s \in D \;\middle|\; \exists a \in \kappa(s) \,.\, c(s,a) = 0 \wedge \text{Supp}(s,a) \subseteq D \right\} .$$

Then $B = \Lambda^{\infty}(S)$, and the computation requires at most $|S|$ iterations. ∎

The following theorem shows that to check that the PNS $\Pi^{\phi}_{A,C}$ is non-Zeno it suffices to check that there are no reachable ZCSSs, which by the above results can be done in time polynomial in $|\Pi^{\phi}_{A,C}|$.

Theorem 4 (checking non-Zenoness) *A PNS is non-Zeno iff the corresponding MDP does not contain any non-empty ZCSS reachable from the initial state.*

Proof. Assume that an MDP contains a ZCSS reachable from the initial state. Then there is a policy such that with positive probability a behavior from the initial state is eventually confined to the ZCSS, and once confined, it never takes an action with positive cost. Thus, if an MDP contains a reachable ZCSS, the corresponding PNS is not non-Zeno.

On the other hand, assume that the MDP does not contain any ZCSS reachable from the initial state. Let $B_1, \ldots, B_n$ be the stable sets of the MDP, and let $\Omega^i_{s_{in}} = \{\omega \in \Omega_{s_{in}} \mid \text{inft}(\omega) = B_i\}$, for $1 \le i \le n$. By Lemma 2, for any policy η it is $\mu_{s_{in},\eta}(\bigcup_{i=1}^{n} \Omega^i_{s_{in}}) = 1$, so we need to consider only the behaviors in $\Omega^i_{s_{in}}$ for some $1 \le i \le n$.

If $\mu_{s_{in},\eta}(\Omega^i_{s_{in}}) > 0$, with probability 1 a behavior $\omega \in \Omega^i_{s_{in}}$ executes only finitely many pairs (s,a) such that $\text{Supp}(s,a) \not\subseteq B_i$. Since B_i is not zero-cost, ω must with probability 1 execute infinitely often pairs (s,a) having $c(s,a) > 0$. As these pairs correspond to time steps of the system, the system is non-Zeno. ∎

5.4 Model Checking for the Operator D

Given $\phi \in \mathit{Trans}$, $\psi \in \mathit{Stat}$ and $b \geq 0$, we now consider the problem of computing the truth value of $\mathrm{D}_{\bowtie b}(\phi \triangleright \psi)$ at state $s \in S$ of a PNS $\Pi^{\phi}_{A,C}$. We assume that $\Pi^{\phi}_{A,C}$ is non-Zeno, and that the truth value of ψ has already been evaluated at all states $s \in S$. We let $A_{s,\phi}$ be as before, and we let $B = \{s \in A_{s,\phi} \mid s \not\models \psi\}$ be the set of $\neg\psi$-states reachable from s via a ϕ-transition. Remembering definition (10), if $A_{s,\phi} \neq \emptyset$ we can follow a reasoning similar to (14) and write

$$\inf_{\eta} \mathrm{E}_{s,\eta}\Big\{H_{\omega_{\geq 1},\psi} \;\Big|\; \omega \in \Omega_s \wedge \omega_1 \in A_{s,\psi}\Big\} = \min_{a \in \kappa(s)} \frac{\displaystyle\sum_{t \in B} p(t \mid s,a) \, \inf_{\eta} \mathrm{E}_{t,\eta}\{H_{\omega,\psi} \mid \omega \in \Omega_t\}}{\displaystyle\sum_{t \in A_{s,\phi}} p(t \mid s,a)} \tag{17}$$

$$\sup_{\eta} \mathrm{E}_{s,\eta}\Big\{H_{\omega_{\geq 1},\psi} \;\Big|\; \omega \in \Omega_s \wedge \omega_1 \in A_{s,\psi}\Big\} = \max_{a \in \kappa(s)} \frac{\displaystyle\sum_{t \in B} p(t \mid s,a) \, \sup_{\eta} \mathrm{E}_{t,\eta}\{H_{\omega,\psi} \mid \omega \in \Omega_t\}}{\displaystyle\sum_{t \in A_{s,\phi}} p(t \mid s,a)} . \tag{18}$$

Let $T = \{s \in S \mid s \models \psi\}$ be the set of states satisfying ψ, and let $W_{\omega,i}$ be a random variable that indicates the cost at position i of behavior ω: that is, $W_{\omega,i} = c(s,a)$ if $\omega_i = s$ and action $a \in \kappa(s)$ has been selected by the policy. For $t \notin T$, the *first-passage cost* $C^T_{\eta,t}$ is then defined as

$$C^T_{t,\eta} = \mathrm{E}_{t,\eta}\Big\{\sum_{i=0}^{T_{\omega,\psi}} W_{\omega,i} \mid \omega \in \Omega_t\Big\} .$$

As a consequence of our choice of costs for the actions, we have $C^T_{t,\eta} = \mathrm{E}_{t,\eta}\{H_{\omega,\psi} \mid \omega \in \Omega_t\}$, and since $\Pi^{\phi}_{A,C}$ is non-Zeno,

$$C^T_{\eta,t} < \infty \quad \textit{iff} \quad \mathrm{Pr}_{t,\eta}(R_{\omega,T} \mid \omega \in \Omega_t) = 1 .$$

The model-checking problem is thus reduced to the computation of $\inf_{\eta} C^T_{t,\eta}$ and $\sup_{\eta} C^T_{t,\eta}$ for all $t \in B$. We examine the two cases separately.

5.4.1 Minimization of First-Passage Cost

Classical results on Markov decision processes insure the existence of a policy η^- that minimizes the first-passage cost $C^T_{\eta,t}$; the minimum cost $C^T_{\eta^-,t}$ can be determined using the value-iteration methods reviewed in [Put94]. Thus, it is possible to substitute inf with min in (17), and the value of $C^T_{\eta^-,t}$ at all $t \in B$ can be used to compute the truth value of $s \models \mathrm{D}_{>b}(\phi \triangleright \psi)$ and $s \models \mathrm{D}_{\geq b}(\phi \triangleright \psi)$. Unfortunately, computing the minimum cost $C^T_{\eta^-,t}$ is computationally expensive. The following theorem states that it is exactly as difficult as computing the minimum expected total cost of a general finite positive model of Markov decision problem. This latter problem cannot be solved using linear-programming methods, and is not known to be solvable in polynomial time in $|\Pi_{mdp}|$ [Put94].

Theorem 5 *The following two problems are polynomially equivalent, i.e. are inter-reducible by polynomial transformations:*

1. *Given a PNS $\Pi^{\phi}_{A,C}$ corresponding to an SRTS A, a subset T of states and a single state $s_{in} \notin T$, compute $C^{T}_{\eta^-,s_{in}}$ on the corresponding Markov decision process.*
2. *Given a Markov decision process with finite state space and non-negative costs, and given a state s_{in}, compute the minimum expected total cost from s_{in}.*

Proof. The reduction from Problem 1 to Problem 2 is straightforward. The reduction from Problem 2 to Problem 1 proceeds as follows.

Let $\Pi_{mdp} = (\mathcal{V}, S, s_{in}, \kappa, p, c)$ be the MDP. First, note that if Π_{mdp} does not contain non-empty ZCSSs, the minimum expected total cost from any state is infinite under any policy. This can be shown by a reasoning analogous to the one used in the proof of Theorem 4. We can check whether Π_{mdp} contains non-empty ZCSSs in time polynomial in $|\Pi_{mdp}|$; if it does not contain them, we can construct in constant time a non-Zeno SRTS in which the set T is not reachable from the initial state. Otherwise, let C be the maximal ZCSS of Π_{mdp}.

If $s_{in} \in C$, we know that the minimum expected cost from s_{in} is 0, and again it is easy to construct in constant time an appropriate SRTS. Otherwise, we replace C with a single new state s_C, updating the transition probabilities by $p(s_C \mid s,a) = \sum_{t \in C} p(t \mid s,a)$. We set $\kappa(s_C) = \{a_C\}$ and $p(s_C \mid s_C, a_C) = 1$, where a_C is a new action, and we take $T = \{s_C\}$. We can assume without loss of generality that $c(s,a) = 0 \vee c(s,a) \geq 1$ for all $s \in S, a \in \kappa(s)$: otherwise, we can enforce this condition by multiplying all costs by the reciprocal of the least non-zero cost, obtaining an equivalent problem.

Let $Act = \{(s,a) \mid s \in S \wedge a \in \kappa(s)\}$. We construct an SRTS $(\mathcal{V}', S', s'_{in}, \mathcal{T})$ as follows.

1. $S' = S \cup Act \times \{0,1\}$.
2. $\mathcal{V}' = \{x\}$, where x is an integer variable whose value is used to enumerate the states in S'.
3. $s'_{in} = s_{in}$.
4. $\mathcal{T}_i = \{\tau^{(1)}_{s,a}\}_{(s,a)\in Act} \cup \{\tau^{(2)}_{s,a,t}\}_{(s,a)\in Act, t\in S}$.
5. $\mathcal{T}_g = \{\tau^{(3)}_{s,a}\}_{(s,a)\in Act}$.

Every transition $\tau^{(1)}_{s,a} \in \mathcal{T}_i$ corresponds to the choice of action $a \in \kappa(s)$ at state s. It is enabled on the state $s \in S \subset S'$, and $\tau^{(1)}_{s,a}(s) = ((s,a),0)$. Every $\tau^{(1)}_{s,a} \in \mathcal{T}_i$ has weight $\perp$, so that actions can be selected nondeterministically.

For all $(s,a) \in Act$, if $c(s,a) = 0$ then transition $\tau^{(3)}_{s,a}$ can never be taken. Otherwise, $\tau^{(3)}_{s,a}$ is enabled on state $((s,a),0)$, and $\tau^{(3)}_{s,a}((s,a),0) = ((s,a),1)$. We set $q_{\tau^{(3)}_{s,a}} = 1/c(s,a)$, so that before $\tau^{(3)}_{s,a}$ is taken a time on average $c(s,a)$ elapses.

Finally, the transitions $\{\tau^{(2)}_{s,a,t}\}_{(s,a)\in Act, t\in S}$ are used to simulate the effect of the actions on the state of the MDP. For all $(s,a) \in Act$, transition $\tau_{s,a,t}$ is enabled on $((s,a),0)$ if $c(s,a) = 0$, and is enabled on $((s,a),1)$ if $c(s,a) \geq 1$. For $t \in S$, transition $\tau^{(2)}_{s,a,t}$ leads to state t, and has weight $p(t \mid s,a)$.

Since every action of the MDP is simulated in the SRTS by a transition that is scheduled in an average time equal to the cost of the action, it can be proved that the minimum expected total cost for the MDP is equal to the minimum expected time for the SRTS from s_{in} to T. ∎

5.4.2 Maximization of First-Passage Cost

Since we assume that the PNS is non-Zeno, if there is a policy η such that $\Pr_{t,\eta}(R_{\omega,T} \mid \omega \in \Omega_t) < 1$, then $\sup_\eta C^T_{\eta,t} = \infty$. Otherwise $\sup_\eta C^T_{\eta,t} < \infty$, and there is a policy η^+ that maximizes the first-passage cost $C^T_{\eta,t}$. The maximum cost $C^T_{\eta^+,t} < \infty$ can be computed by solving a linear programming problem in time polynomial in the size of the MDP using the methods of [Der70, Put94]. To determine whether the maximum first-passage cost converges, we can use the following corollary.

Corollary 2 *Let U be the largest stable subset of $S - T$. Then there is a path from t to U in (S, ρ) iff there is a policy η such that* $\Pr_{t,\eta}(R_{\omega,T} \mid \omega \in \Omega_t) < 1$.

Proof. By Lemma 1, if there is a path from t to U in (S, ρ), there is a policy under which a behavior from t has positive probability of being eventually confined to U, thus never reaching T.

Conversely, by Corollary 1, under any policy the probability that a behavior reaches $T \cup U$ is 1. Thus, under any policy, if a behavior has probability less than 1 of reaching T, it must have probability greater than 0 of reaching U, implying that U is reachable from t in the graph (S, ρ). ∎

5.5 Complexity of pTL* Model Checking

Let $\text{pTL}^*_<$ be the logic obtained from pTL* by allowing only the versions $\mathrm{D}_{<b}$ and $\mathrm{D}_{\leq b}$ of the operator $\mathrm{D}_{\bowtie b}$. The following theorem combines the results of [CY90] with the previous analysis of pTL* model checking and checking for non-Zenoness.

Theorem 6 *The following assertions hold:*

1. *Checking whether a PNS $\Pi^\phi_{A,C}$ is non-Zeno has time complexity polynomial in $|\Pi^\phi_{A,C}|$.*
2. *Model checking of a $\text{pTL}^*_<$ formula ϕ over a PNS $\Pi^\phi_{A,C}$ has time-complexity polynomial in $|\Pi^\phi_{A,C}|$ and doubly exponential in $|\phi|$.*

On the other hand, the size of $\Pi^\phi_{A,C}$ depends exponentially on the number of delayed transitions and instrumentation clocks of A, C and ϕ. This exponential dependency is typical of most model-checking approaches to the formal verification of real-time systems.

6 Extending the D Operator to Past Formulas

The D operator of pTL and pTL* enables us to express bounds on the average time needed, after a transition, to reach a given set of system states. Thus, it is somewhat less general than the P operator, which can refer to the probability of a general sequence formula. For instance, in the producer-consumer example of Section 4 the specification *"If A_1 sends an item, A_2 finishes the processing in an average time no greater than 8 time units"* cannot be expressed in pTL*, in contrast with (13). In fact, the formula

$$\mathrm{A}\Box\Big[\mathrm{D}_{\leq 8}\Big((s_1 = 0 \wedge s'_1 = 1) \triangleright m = 3\Big)\Big]$$

encodes the (different) specification *"If A_1 sends an item, A_2 finishes processing some item in an average time no greater than 8 time units"*. Without using sequence formulas it is not possible to distinguish between the cases in which $m = 3$ as a result of processing the item that has been sent, or the previous one.

To overcome this limitation we generalize the definition of the D operator, allowing the formula ψ in $\mathrm{D}_{\bowtie b}(\phi \triangleright \psi)$ to be a *past* temporal formula, instead of a state formula. A *past* temporal formula is a formula constructed from state formulas in *Stat* using the temporal operators $\boxminus$, $\diamondminus$ and $\mathcal{S}$ [MP91][2]

The semantics of this extension can be defined as follows. Given a behavior ω and a past formula ψ, let $T_{\omega,\psi} = \min\{i \mid \omega_0 \cdots \omega_i \models_i \psi\}$, where $\omega_0 \cdots \omega_i \models_i \psi$ indicates that formula ψ holds at the last position i of the finite sequence $\omega_0 \cdots \omega_i$. Then, $H_{\omega,\psi} = \sum_{i=1}^{T_{\omega,\psi}} \omega_i(d)$ is the time that elapses along ω before ψ is satisfied, and the truth value of $\mathrm{D}_{\bowtie b}(\phi \triangleright \psi)$ can be again defined as in (10).

Using this extended version of pTL*, we can encode the above specification with the formula

$$\mathrm{A}\Box\Big[\mathrm{D}_{\leq 8}\Big((s_1 = 0 \wedge s_1' = 1) \triangleright m = 3 \, \mathcal{G} \, m = 2 \, \mathcal{G} \, m = 1 \, \mathcal{G} \, r_2 = 1 \, \mathcal{G} \, s_1 = 1\Big)\Big] ,$$

where we have used the abbreviation $\mathcal{G}$ for the *non-skipping* since:

$$\phi \, \mathcal{G} \, \psi \quad := \quad \phi \wedge (\phi \, \mathcal{S} \, \psi)$$

which associates to the right. The past formula insures that item whose processing is finished at $m = 3$ is the same that is sent by the transition on the left of $\triangleright$.

This extended version of the D operator can be model checked by combining the techniques of [BdA95] with the algorithms presented in the previous section. Specifically, given a PNS $\Pi_{A,C}^{\gamma}$ it is possible to construct a PNS $\Pi_{A,C}^{\gamma,\psi}$ in which the states keep track of the truth values of the past subformulas of ψ (ψ itself included) [BdA95]. The truth value of $\mathrm{D}_{\bowtie b}(\phi \triangleright \psi)$ can then be computed by applying the algorithms of the previous section to the PNS $\Pi_{A,C}^{\gamma,\psi}$. Letting $\mathrm{pTL}_{<}^{*p}$ be the logic obtained from $\mathrm{pTL}_{<}^{*}$ by adopting the version of D generalized to past formulas, we can state the following result about the complexity of $\mathrm{pTL}_{<}^{*p}$ model checking.

Theorem 7 ($\mathrm{pTL}_{<}^{*p}$ model checking) $|\Pi_{A,C}^{\gamma,\psi}| = O(2^{|\psi|}\,|\Pi_{A,C}^{\gamma}|)$. *Therefore, model checking of a $\mathrm{pTL}_{<}^{*p}$ formula ϕ over a PNS $\Pi_{A,C}^{\phi}$ has time-complexity polynomial in $|\Pi_{A,C}^{\phi}|$ and doubly exponential in $|\phi|$.*

Thus, the complexity of the model-checking problem is not changed by the extension, as it is dominated by the doubly-exponential dependency arising from the operator P.

7 Conclusions

In this work we have presented the system model of SRTSs, which provide a synthetic representation of probabilistic real-time systems, and we have extended the probabilistic logics of [HJ89, Han94, ASB+95, BdA95] by adding an operator D that specifies the average time

[2]The use of past temporal operators in branching-time logics has been discussed in depth in [KP95].

between events. We have then presented model-checking algorithms for this extended logic, based on results from automata theory and relatively simple concepts about probabilistic systems. With these extensions and algorithms, temporal logic can be applied to the study of the performance and reliability of real-time systems.

On the other hand, no efficient algorithm has been given for model checking the lower-bound version of the D operator, for which in fact we have presented relative hardness results. Further research is needed to investigate methods to deal with this complexity, as well as to study optimizations and average-case behavior for the algorithms presented in this paper. Further research is also needed to determine if the methods developed in this paper, based on a discrete model of time, can be adapted to a continuous-time model.

Acknowledgements. We wish to thank Andrea Bianco for many inspiring discussions and suggestions. We also thank Arjun Kapur, Henny Sipma and Tomás Uribe for several helpful comments.

References

[ACD92] R. Alur, C. Courcoubetis, and D. Dill. Verifying automata specifications of probabilistic real-time systems. In *Real Time: Theory in Practice*, volume 600 of *Lect. Notes in Comp. Sci.*, pages 28–44. Springer-Verlag, 1992.

[AD91] R. Alur and D. Dill. The theory of timed automata. In *Real-Time: Theory in Practice*, volume 600 of *Lect. Notes in Comp. Sci.*, pages 45–73. Springer-Verlag, 1991.

[ASB+95] 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 *Lect. Notes in Comp. Sci.* Springer-Verlag, 1995.

[BdA95] 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 *Lect. Notes in Comp. Sci.*, pages 499–513. Springer-Verlag, 1995.

[CY88] C. Courcoubetis and M. Yannakakis. Verifying temporal properties of finite-state probabilistic programs. In *Proc. 29th IEEE Symp. Found. of Comp. Sci.*, 1988.

[CY90] C. Courcoubetis and M. Yannakakis. Markov decision processes and regular events. In *ICALP'90*, volume 443 of *Lect. Notes in Comp. Sci.*, pages 336–349. Springer-Verlag, 1990.

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

[Der70] C. Derman. *Finite State Markovian Decision Processes.* Acedemic Press, 1970.

[EL85] E.A. Emerson and C.L. Lei. Modalities for model checking: Branching time strikes back. In *Proc. 12th ACM Symp. Princ. of Prog. Lang.*, pages 84–96, 1985.

[ES84] E.A. Emerson and A.P. Sistla. Deciding branching time logic. In *Proc. 16th ACM Symp. Theory of Comp.*, pages 14–24, 1984.

[Han94] H. Hansson. *Time and Probability in Formal Design of Distributed Systems.* Elsevier, 1994.

[HJ89] 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.

[HJ94] H. Hansson and B. Jonsson. A logic for reasoning about time and probability. *Formal Aspects of Computing*, 6(5):512–535, 1994.

[HS84] S. Hart and M. Sharir. Probabilistic temporal logic for finite and bounded models. In *Proc. 16th ACM Symp. Theory of Comp.*, pages 1–13, 1984.

[Koy90] R. Koymans. Specifying real-time properties with metric temporal logic. *Real-time Systems*, 2(4):255–299, 1990.

[KP95] O. Kupferman and A. Pnueli. Once and for all. In *Proc. 10th IEEE Symp. Logic in Comp. Sci.*, pages 25–35, 1995.

[KSK66] J.G. Kemeny, J.L. Snell, and A.W. Knapp. *Denumerable Markov Chains*. D. Van Nostrand Company, 1966.

[LS82] D. Lehman and S. Shelah. Reasoning with time and chance. *Information and Control*, 53(3):165–198, 1982.

[MP91] Z. Manna and A. Pnueli. *The Temporal Logic of Reactive and Concurrent Systems: Specification*. Springer-Verlag, New York, 1991.

[MP93] Z. Manna and A. Pnueli. Models for reactivity. *Acta Informatica*, 30:609–678, 1993.

[Pnu83] A. Pnueli. On the extremely fair treatment of probabilistic algorithms. In *Proc. 15th ACM Symp. Theory of Comp.*, pages 278–290, 1983.

[Put94] M.L. Puterman. *Markov Decision Processes*. John Wiley and Sons, 1994.

[PZ86] A. Pnueli and L. Zuck. Probabilistic verification by tableaux. In *Proc. First IEEE Symp. Logic in Comp. Sci.*, pages 322–331, 1986.

[PZ93] A. Pnueli and L.D. Zuck. Probabilistic verification. *Information and Computation*, 103:1–29, 1993.

[Saf88] S. Safra. On the complexity of ω-automata. In *Proc. 29th IEEE Symp. Found. of Comp. Sci.*, 1988.

[Seg95] R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. PhD thesis, MIT, June 1995. Technical Report MIT/LCS/TR-676.

[SL94] R. Segala and N.A. Lynch. Probabilistic simulations for probabilistic processes. In *CONCUR '94: Concurrency Theory*, volume 836, pages 481–496. Springer-Verlag, 1994.

[Var85] M.Y. Vardi. Automatic verification of probabilistic concurrent finite-state systems. In *Proc. 26th IEEE Symp. Found. of Comp. Sci.*, pages 327–338, 1985.

[VW86] M.Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. First IEEE Symp. Logic in Comp. Sci.*, pages 332–344, 1986.