Checking LTL Properties of Recursive Markov Chains

Mihalis Yannakakis
Department of Computer Science
Columbia University

Kousha Etessami
LFCS, School of Informatics
University of Edinburgh

Abstract

We present algorithms for the qualitative and quantitative model checking of Linear Temporal Logic (LTL) properties for Recursive Markov Chains (RMCs). Recursive Markov Chains are a natural abstract model of procedural probabilistic programs and related systems involving recursion and probability. For the qualitative problem ("Given a RMC A and an LTL formula φ, do the computations of A satisfy φ almost surely?") we present an algorithm that runs in polynomial space in A and exponential time in φ. For several classes of RMCs, including RMCs with one exit (a special case that corresponds to well-studied probabilistic systems, e.g., multi-type branching processes and stochastic context-free grammars) the algorithm runs in polynomial time in A and exponential time in φ. On the other hand, we also prove that the problem is EXPTIME-hard, and hence it is EXPTIME-complete. For the quantitative problem ("does the probability that a computation of A satisfies φ exceed a given threshold p?", or approximate the probability within a desired precision) we present an algorithm that runs in polynomial space in A and exponential space in φ. For linearly-recursive RMCs, we can compute the exact probability in time polynomial in A and exponential in φ. These results improve by one exponential, in both the qualitative and quantitative case, the complexity that one would obtain if one first translated the LTL formula to a Büchi automaton and then applied the model checking algorithm for Büchi automata from [11]. Our results combine techniques developed in [10, 11] for analysis of RMCs, and in [6] for LTL model checking of flat Markov Chains, and extend them with new techniques.

1 Introduction

Recursive Markov Chains (RMCs), introduced in [10, 11], are a natural abstract model of procedural probabilistic programs and related systems involving recursion and probability. RMCs succinctly define a natural class of denumerable Markov chains that generalize well-studied stochastic processes such as multi-type Branching Processes. An RMC consists of a collection of finite state component Markov chains (MCs) that can call each other in a potentially recursive manner. Each component has a set of *nodes*, a set of *boxes* (each mapped to a component), an interface consisting of a set of *entry* and *exit* nodes (nodes in the component where we may start and terminate), and a set of probabilistic transitions connecting the nodes and boxes. A transition to a box specifies the entry node and models the invocation of the component MC associated with the box; when (and if) the component MC terminates at an exit, execution of the calling MC resumes from the corresponding exit of the box. An example RMC with 2 components, A_1 and A_2, each with one entry and two exits, is depicted in Figure 1. Note that, e.g., box b_1 of component A_1 maps to a copy of component A_2.

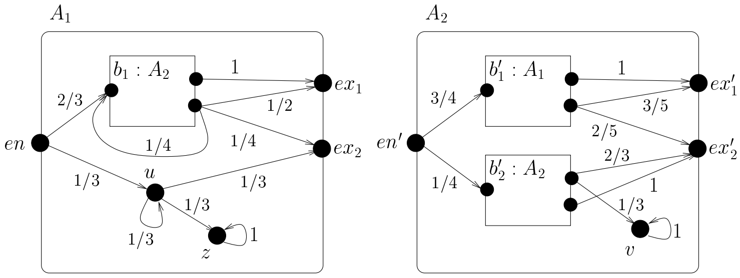

Figure 1. *A sample Recursive Markov Chain*

RMCs are a probabilistic version of Recursive State Machines (RSMs). RSMs ([1, 2]) and closely related models like Pushdown Systems (PDSs) (see, e.g., [3, 7]) have been studied extensively in research on model checking and program analysis, because of their applications to verification of sequential programs with procedures. RMCs are intimately related to probabilistic Pushdown Systems (pPDSs), and decidability and complexity of model checking questions for pPDSs has also been studied recently ([8, 4]).

RMCs generalize other well-studied models involving probability and recursion: *Stochastic Context-Free Grammars* (SCFGs) have been extensively studied, mainly in natural language processing (NLP) (see [16]). *Multi-Type Branching Processes* (MT-BPs), are an important family of stochastic processes with many applications in a variety of areas (see, e.g., [14]). Both SCFG's and MT-BP's are essentially equivalent to *1-exit* RMC's: the special case of RMC's in which all components have one exit.

Probabilistic models of programs and systems are of interest for several reasons. First, a program may use randomization, in which case the transition probabilities reflect the random choices of the algorithm. Second, we may want to model and analyse a program or system under statistical conditions on its behaviour (e.g., based on profiling statistics or on statistical assumptions), and to determine the induced probability of properties of interest.

In this paper, we study the model checking problems for RMCs when the specification is given by a formula φ in Linear Temporal Logic (LTL). As is well known, LTL is a standard logic for specifying properties of the executions of a system. Specifically, we are given an RMC A and an LTL formula φ. Every vertex of the RMC is labeled by the subset of the propositions of φ which hold at the vertex. Let $P_A(\varphi)$ denote the probability that an execution of A satisfies φ. The *qualitative* model checking problem is: determine whether almost all executions of A satisfy φ, i.e., whether $P_A(\varphi) = 1$. (Note that almost no executions of A satisfy φ, i.e. $P_A(\varphi) = 0$, iff $P_A(\neg\varphi) = 1$.)

In the *quantitative* model checking problems we wish to compare $P_A(\varphi)$ to a given rational threshold p, i.e., is $P_A(\varphi)\Delta p$?, where $\Delta \in \{<, \leq, =\}$, or alternatively, we may wish to approximate $P_A(\varphi)$ to within a given number of bits of precision. In general, as shown in [10], $P_A(\varphi)$ may be irrational. Hence we can not compute it exactly.

We show that the qualitative model checking problems can be solved in PSPACE in A and EXPTIME in φ. More specifically, the exponent depends only on the number of temporal operators in φ, which in practice is a small constant. For three classes of RMC's we obtain polynomial time in A (and exponential in φ): (1) The class of 1-exit RMCs (which correspond to SCFG's and MT-BP's); (2) The class of RMCs (denoted Bd-RMCs) with a bounded total number of entries and exits. In terms of probabilistic program abstractions, this class of RMC's corresponds to programs with a bounded number of distinct procedures, each of which has a bounded number of input/output parameter values. The internals of the components of the RMCs (i.e., the procedures) can be arbitrarily large and complex; (3) The class of "linearly-recursive" RMCs (`lr`-RMCs), where no component contains a path of transitions from a returnport of a box to an entry of a box.

For quantitative model checking, we show that, given a rational $p \in [0, 1]$, deciding whether $P_A(\varphi) \geq p$, (or $> p$, or any other comparison operator) can be decided in PSPACE in $|A|$ and EXPSPACE in $|\varphi|$. For A a `lr`-RMC we can compute the exact (rational) probability $P_A(\varphi)$ in polynomial time in A and exponential time in φ. For A a Bd-RMC and when φ is fixed, it follows already from the results of [11] (by translating φ to a Büchi automaton) that there is an algorithm that runs in polynomial time in $|A|$.

For lower bounds, we prove that qualitative LTL model checking, even for a fixed, single entry/exit RMC, is already EXPTIME-hard, and thus EXPTIME-complete.

As observed in [10, 11], our PSPACE complexities in $|A|$ for quantitative model checking of 1-exit RMCs, and for qualitative model checking of multi-exit RMCs, can not be improved without a breakthrough on the complexity of a long standing open problem in the complexity of exact numerical computation. Namely, we showed in [10] that the *Square Root Sum Problem*, reduces to quantitative reachability analysis for 1-exit RMCs and qualitative reachability analysis for multi-exit RMCs.

Related Work. Model checking of flat Markov chains has received extensive attention both in theory and practice (e.g. [6, 15, 19, 22]). It is known that both qualitative and quantitative model checking of a finite Markov chain A with respect to an LTL specification φ are PSPACE-complete, and furthermore the probability $P_A(\varphi)$ can be computed exactly in time polynomial in A and exponential in φ ([6]).

RMCs and the closely related model of probabilistic Pushdown Systems (pPDS) have been studied recently [10, 11, 12, 8, 4, 9]. Most relevant to the subject of the current paper are the results on model checking of properties specified by Büchi automata [4, 11]. In [11] we showed that qualitative model checking of an RMC A with respect to a Büchi automaton B can be done in PSPACE in $|A|$ and EXPTIME in $|B|$, and is also EXPTIME-complete in $|B|$, and that quantitative model checking can also be done in PSPACE in $|A|$, and in EXPSPACE in $|B|$. As is well known ([23]), any LTL formula φ can be translated to equivalent Büchi automaton B, and thus our prior algorithms also yield algorithms for checking LTL properties of RMCs. However, translation of LTL formulas to Büchi automata necessarily incurs a worst-case exponential blow up. Thus, if we use our earlier results directly, we would obtain a doubly exponential time algorithm for qualitative LTL model checking, and a doubly exponential space algorithm for quantitative LTL model checking. We show in this paper that these extra exponential costs in complexity are not necessary.

Our results build on both the techniques developed in [10, 11] for analysis of RMCs, as well as the techniques developed for LTL model checking of flat Markov Chains in [6]. There are a number of difficulties that have to be surmounted however. The algorithm of [6] for model checking LTL properties of flat Markov chains employs an iterative approach, whereby the chain is refined in each iteration and the formula is simplified by elimination of temporal operators, until at the end the formula becomes propositional and can be verified directly. There are serious technical obstacles however for effectively extending this approach to the recursive setting, and this is not what we do. Instead, we follow a different approach which is more global in nature. We use an idea from another method of [6], used there for another purpose, and we extend it with other techniques to handle recursion and LTL.

The paper is organized as follows. Sec. 2 gives definitions and background from [10, 11]; Sec.'s 3 and 4 give our upper bounds for qualitative and quantitative LTL model checking, respectively; Sec. 5 gives our EXPTIME-hardness result for qualitative LTL model checking.

2 Definitions and Background

A *Recursive Markov Chain (RMC)*, A, is a tuple $A = (A_1, \ldots, A_k)$, where each *component chain* $A_i = (N_i, B_i, Y_i, En_i, Ex_i, \delta_i)$ consists of:

- A set N_i of *nodes*.
- A subset of *entry* nodes $En_i \subseteq N_i$, and a subset of *exit* nodes $Ex_i \subseteq N_i$.
- A set B_i of *boxes*. Let $B = \cup_{i=1}^{k} B_i$ be the (disjoint) union of all boxes of A.
- A mapping $Y_i : B_i \mapsto \{1, \ldots, k\}$ assigns a component to every box. Let $Y = \cup_{i=1}^{k} Y_i$ be $Y : B \mapsto \{1, \ldots, k\}$ where $Y|_{B_i} = Y_i$, for $1 \leq i \leq k$.
- To each box $b \in B_i$, we associate a set of *call ports*, $\mathit{Call}_b = \{(b, en) \mid en \in En_{Y(b)}\}$, and a set of *return ports*, $\mathit{Return}_b = \{(b, ex) \mid ex \in Ex_{Y(b)}\}$.
- A transition relation δ_i, where transitions are of the form $(u, p_{u,v}, v)$ where:
  1. the source u is either a non-exit node $u \in N_i \setminus Ex_i$, or a return port $u = (b, ex) \in \mathit{Return}_b$, where $b \in B_i$.
  2. The destination v is either a non-entry node $v \in N_i \setminus En_i$, or a call port $v = (b, en) \in \mathit{Call}_b$, where $b \in B_i$.
  3. $p_{u,v} \in \mathbb{R}_{>0}$ is the probability of transition from u to v. (We assume $p_{u,v}$ is rational for computational purposes.)
  4. *Consistency of probabilities*: $\sum_{\{v' \mid (u, p_{u,v'}, v') \in \delta_i\}} p_{u,v'} = 1$, for each u that is not a call port or exit node (the latter do not have outgoing transitions).

We use the term *vertex* of A_i to refer collectively to its set of nodes, call ports, and return ports, and we denote this set by Q_i, and we let $Q = \bigcup_{i=1}^{k} Q_i$ be the set of all vertices of the RMC A. That is, the transition relation δ_i is a set of probability-weighted directed edges on the vertices Q_i of A_i. Let $\delta = \cup_i \delta_i$ be the set of all transitions of A.

RMCs where every component has at most one exit are called *1-exit* RMCs. RMCs where the total number of entries and exits is bounded by a constant c, (i.e., $\sum_{i=1}^{k} |En_i| + |Ex_i| \leq c$) are called *bounded total entry-exit* RMCs (Bd-RMCs, for short). The class of *linear* RMCs (`lr`-RMCs, for short) are those with no paths (in any component) from a return-port to a call-port.

An RMC A defines a global denumerable Markov chain $M_A = (V, \Delta)$ as follows. The global *states* $V \subseteq B^* \times Q$ are pairs of the form $\langle \beta, u \rangle$, where $\beta \in B^*$ is a (possibly empty) sequence of boxes and $u \in Q$ is a *vertex* of A. More precisely, the states V and transitions Δ are defined inductively as follows:

1. $\langle \epsilon, u \rangle \in V$, for $u \in Q$. (ϵ denotes the empty string.)
2. if $\langle \beta, u \rangle \in V$ and $(u, p_{u,v}, v) \in \delta$, then $\langle \beta, v \rangle \in V$ and $(\langle \beta, u \rangle, p_{u,v}, \langle \beta, v \rangle) \in \Delta$
3. if $\langle \beta, (b, en) \rangle \in V$ and $(b, en) \in \mathit{Call}_b$, then $\langle \beta b, en \rangle \in V$, and $(\langle \beta, (b, en) \rangle, 1, \langle \beta b, en \rangle) \in \Delta$.
4. if $\langle \beta b, ex \rangle \in V$ and $(b, ex) \in \mathit{Return}_b$, then $\langle \beta, (b, ex) \rangle \in V$ and $(\langle \beta b, ex \rangle, 1, \langle \beta, (b, ex) \rangle) \in \Delta$.

Item 1 corresponds to the possible initial states, 2 corresponds to a transition within a component, 3 is when a new component is entered via a box, 4 is when a component is exited and control returns to the calling component.

We will consider M_A with various *initial states* of the form $\langle \epsilon, v \rangle$, or we can consider a distribution on these initial states. Some states of M_A are *terminating*, i.e., have no outgoing transitions. Namely, states $\langle \epsilon, ex \rangle$, where ex is an exit. We want M_A to be a proper Markov chain, so we consider terminating states as *absorbing*, with a self-loop of probability 1. A *trace* (or *trajectory*) $t \in V^\omega$ of M_A is an infinite sequence of states $t = s_0 s_1 s_2 \ldots$ such that for all $i \geq 0$, there is a transition $(s_i, p_{s_i, s_{i+1}}, s_{i+1}) \in \Delta$.

Given a vertex $u \in Q_i$ and an exit $ex \in Ex_i$, both in the same component A_i, let $q^*_{(u,ex)}$ denote the probability that a trajectory starting at the state $\langle \epsilon, u \rangle$ eventually reaches the state $\langle \epsilon, ex \rangle$; we will often simply say that a trajectory of the RMC that starts at u exits the component at ex. As shown in [10], the probabilities $q^*_{(u,ex)}$ are the Least Fixed Point (LFP) of a system of multivariate polynomial equations $\mathbf{x} = P(\mathbf{x})$, where $\mathbf{x}$ is the vector of variables $x_{(u,ex)}$ corresponding to the vertex-exit pairs, $ex \in Ex_i$, for $1 \leq i \leq k$. The system contains one equation $x_{(u,ex)} = P_{(u,ex)}(\mathbf{x})$, for each variable $x_{(u,ex)}$, where $P_{(u,ex)}(\mathbf{x})$ is a multivariate polynomial with positive rational coefficients. There are 3 cases, based on the "type" of vertex u:

1. Type I: $u = ex$. In this case: $x_{(ex,ex)} = 1$.
2. Type II: either $u \in N_i \setminus \{ex\}$ or $u = (b, ex')$ is a return port. Then $x_{(u,ex)} = \sum_{\{v \mid (u, p_{u,v}, v) \in \delta\}} p_{u,v} \cdot x_{(v,ex)}$.
3. Type III: $u = (b, en)$ is a call port. In this case:
$$x_{((b,en),ex)} = \sum_{ex' \in Ex_{Y(b)}} x_{(en,ex')} \cdot x_{((b,ex'),ex)}$$

The vector $\mathbf{q}^*$ of the exit probabilities $q^*_{(u,ex)}$ is a solution to this system. Furthermore, it is the least nonnegative solution. That is, if $\mathbf{x} = P(\mathbf{x})$ and $\mathbf{x} \geq \mathbf{0}$, where $\mathbf{0}$ denotes the all 0 vector and comparison of vectors is done component-wise, then $\mathbf{x} \geq \mathbf{q}^*$.

The exit probabilities may in general be irrational and not even solvable by radicals [10], so they cannot be computed exactly. However, we can effectively compare them with given rational values. For notational convenience, let us enumerate the pairs (u, ex) from 1 to m, and identify the variable $x_{(u,ex)}$ of the jth pair with x_j. Suppose that we want to check whether the probability $q^*_{(u,ex)}$ of the jth pair is at most a given rational number p. Consider the following sentence in the *Existential Theory of Reals* (which we denote by ***ExTh***$(\mathbb{R})$):

$$\varphi \equiv \exists x_1, \ldots, x_m \bigwedge_{i=1}^{m} (x_i = P_i(x_1, \ldots, x_m)) \wedge \bigwedge_{i=1}^{m} 0 \leq x_i \wedge (x_j \leq p)$$

φ is true precisely when the LFP of $\mathbf{x} = P(\mathbf{x})$, i.e., $\mathbf{q}^*$, satisfies $q^*_{(u,ex)} \leq p$. We can check this using a decision procedure for ***ExTh***$(\mathbb{R})$. It is known that ***ExTh***$(\mathbb{R})$ can be decided in PSPACE and in exponential time, where the time exponent depends (linearly) only on the number of variables; thus for a fixed number of variables the algorithm runs in polynomial time [5, 20]. For Bd-RMCs, as shown in [10] it is possible to efficiently construct a system of equations in a *bounded* number of variables, whose LFP yields the entry-exit probabilities $q^*_{(en,ex)}$, and check it in polynomial time. Furthermore, for 1-exit RMCs (SCFGs), we can efficiently solve qualitative termination/reachability problems, i.e. determine whether $q^*_{(u,ex)}$ is 0, 1, or in-between; this algorithm does not use the ***ExTh***$(\mathbb{R})$ but rather an eigenvalue characterization and graph theoretic methods.

`lr`-RMCs were not studied explicitly in [10]. However, there we showed that a "decomposed" multi-variate Newton's method which operates bottom-up on the "SCCs" of the system $\mathbf{x} = P(\mathbf{x})$, converges monotonically from $\bar{0}$ to the solution $\mathbf{q}^*$ (see section 4 of [10]), and we observed that if these SCCs constitute linear systems, as in the case of finite Markov chains, then this decomposed Newton's method converges in 1 iteration to the (rational) solutions. It can be shown easily that for an `lr`-RMC the SCCs of the system $\mathbf{x} = P(\mathbf{x})$ that are encountered bottom-up are all linear systems. Thus our decomposed Newton's method again converges in one iteration on each of these systems, and we obtain the (rational) reachability probabilities $\mathbf{q}^*$ in polynomial time. The qualitative problem can be solved more efficiently with graph theoretic methods. We summarize these results in the following theorem:

Theorem 1 *([10]) 1. Given RMC A and rational p, there is a PSPACE algorithm to decide whether $q^*_{(u,ex)} \leq p$, with running time $O(|A|^{O(1)} \cdot 2^{O(m)})$ where m is the number of variables in the system $x = P(x)$ for A. Moreover $q^*_{(u,ex)}$ can be approximated to within j bits of precision within PSPACE and with running time at most j times the above.*
2. For a Bd-RMC these problems can be solved in polynomial time.
3. For a 1-exit RMC, we can determine in polynomial time which of the following holds:(1) $q^*_{(u,ex)} = 0$, (2) $q^*_{(u,ex)} = 1$, or (3) $0 < q^*_{(u,ex)} < 1$.
4. For `lr`-RMCs, we can compute in polynomial time the rational probability vector $\mathbf{q}^*$.

On the lower bound side, evidence of the difficulty of reachability problems for RMCs was given in [10] in the form of the *Square-root Sum Problem*, which is the following decision problem: given $(d_1, \ldots, d_n) \in \mathbb{N}^n$ and $k \in \mathbb{N}$, decide whether $\sum_{i=1}^{n} \sqrt{d_i} \leq k$. It is solvable in PSPACE, but it has been a major open problem since the 1970's (see, e.g., [13, 21]) whether it is solvable even in NP. The square-root sum problem is P-time reducible to deciding whether $q^*_{(u,ex)} \geq p$, in a 1-exit RMC, and to deciding whether $q^*_{(u,ex)} = 1$ for a 2-exit RMC [10].

The system $\mathbf{x} = P(\mathbf{x})$ of an RMC may have in general multiple (nonnegative) fixpoints. An augmented system can be constructed which has a unique fixpoint, as follows. For each vertex $v \in Q_i$, define the probability of *never exiting*: $ne(v) = 1 - \sum_{ex \in Ex_i} q^*_{(v,ex)}$. Call a vertex v *deficient* if $ne(v) > 0$, i.e. there is a nonzero probability that if the RMC starts at v it will never terminate (reach an exit of the component). We can determine if a vertex v is deficient by replacing the last constraint in the formula φ above with the constraints $ne(v) = 1 - \sum_{ex \in Ex_i} q^*_{(v,ex)}$ and $ne(v) > 0$ and testing the resulting formula. Doing this for all the vertices, determines the set Q' of deficient vertices, $Q' = \{v \in Q | ne(v) > 0\}$. It is shown in [11] that the system $\mathbf{x} = P(\mathbf{x}), \mathbf{x} \geq \mathbf{0}$ has a unique fixpoint that satisfies $\sum_{ex} q^*_{(v,ex)} < 1$ for every deficient vertex $v \in Q'$ and $\sum_{ex} q^*_{(v,ex)} \leq 1$ for every other vertex.

With a RMC A we can associate a finite-state flat Markov chain, called the *summary chain* M'_A. Assume, w.l.o.g., that the RMC has one initial state $s_0 = \langle \epsilon, en_{\text{init}} \rangle$, with en_{init} the only entry of a component A_0 that does not contain any exits. Any RMC can readily be converted to an "equivalent" one in this form, while preserving relevant probabilities. The summary chain $M'_A = (Q', \delta_{M'_A})$ is defined as follows. The set of states of M'_A is the set of deficient vertices: $Q' = \{v \in Q \mid ne(v) > 0\}$. For $u, v \in Q'$, there is a transition $(u, p'_{u,v}, v)$ in $\delta_{M'_A}$ if and only if one of the following conditions holds:

1. $u, v \in Q_i$ and $(u, p_{u,v}, v) \in \delta_i$, and $p'_{u,v} = \frac{p_{u,v} \cdot ne(v)}{ne(u)}$.
2. $u = (b, en) \in Call_b$, $v = (b, ex) \in Return_b$, $q^*_{(en,ex)} > 0$, and $p'_{u,v} = \frac{q^*_{(en,ex)} ne(v)}{ne(u)}$. We call these *summary* transitions.
3. $u = (b, en) \in Call_b$ and $v = en$, and $p'_{u,v} = \frac{ne(v)}{ne(u)}$. We call these *nesting* transitions.

In all three cases, $p'_{u,v}$ is well-defined (the denominator is nonzero) and positive. Recall that we assumed that the initial vertex en_{init} is the entry of a component A_0, and A_0 has no exits. Thus for all $u \in Q_0$, $ne(u) = 1$, and thus $Q_0 \subseteq Q'$, and if $(u, p_{u,v}, v) \in \delta_0$, then $(u, p_{u,v}, v) \in \delta_{M'_A}$.

Once the deficient vertices are determined, the transitions of the summary Markov chain can be constructed in polynomial time; the transition probabilities however may not be rational, thus they are not computed explicitly.

A mapping ρ can be defined that maps every trace t of the original (infinite) Markov chain M_A starting at $\langle \epsilon, en_{init} \rangle$, either to a unique trajectory $\rho(t)$ of the MC M'_A starting at en_{init}, or to the special symbol $\star$. Briefly, $\rho(t)$ is obtained from t by shortcutting recursive calls that eventually return, and replacing them with summary edges from the call port to the return port of the box; if any of the vertices after this shortcutting is not in the summary chain (i.e. is not in Q') then we set $\rho(t) = \star$. In more detail, let $t = s_0 s_1 \dots s_i \dots$ be a trajectory of M_A, starting at state $s_0 = \langle \epsilon, en_{\mathrm{init}} \rangle$. Its *summary image* $\rho(t)$ is defined sequentially based on prefixes of t, as follows. To start, map s_0 to en_{init}. Suppose $s_i = \langle \beta, u \rangle$, and, inductively, suppose that the prefix $s_0 \dots s_i$ has been mapped to $e_{\mathrm{init}} \dots u$. First, suppose u is not a call port, and that $s_{i+1} = \langle \beta, v \rangle$; if $v \in Q'$ then $s_0 \dots s_i s_{i+1}$ maps to $e_{\mathrm{init}} \dots uv$, otherwise define $\rho(t) = \star$ and quit. Next, suppose $u = (b, en)$ is a call port and $s_{i+1} = \langle \beta b, en \rangle$. If the trace eventually returns from this call (i.e., there exists $j > i+1$, such that $s_j = \langle \beta b, ex \rangle$ and $s_{j+1} = \langle \beta, (b, ex) \rangle$, and such that each of the states $s_{i+1} \dots s_j$, have βb as a prefix of the call stack), then $s_0 \dots s_j$ is mapped to $en_{\mathrm{init}} \dots u(b, ex)$ (provided $(b, ex) \in Q'$, otherwise $\rho(t) = \star$). If the trace never returns from this call, then $s_0 \dots s_i s_{i+1}$ maps to $en_{\mathrm{init}} \dots u\ en$, again provided $en \in Q'$, else $\rho(t) = \star$. This concludes the definition of ρ.

Important properties of the mapping ρ are that (i) with probability 1, a trajectory of M_A starting at $s_0 = \langle \epsilon, en_{\mathrm{init}} \rangle$ is mapped to a trajectory $\rho(t)$ of the summary chain M'_A (i.e. it is not mapped to $\star$), and (ii) the mapping preserves probabilities: for any measurable set D of trajectories of M'_A, its inverse image $\rho^{-1}(D)$ of trajectories of M_A is also measurable and has equal probability. The mapping ρ can be extended similarly to trajectories t of M_A that start at other states $s_0 = \langle \epsilon, z \rangle$, with $z \in Q'$ and which do not reach any exit ex of z's component, i.e. a state $\langle \epsilon, ex \rangle$.

The summary chain plays a central role in the analysis of properties of RMCs given by Büchi automata [11]. Note that for `lr`-RMCs, where probabilities are rational, M'_A can be computed explicitly, and in that case our model checking algorithms from [11] run in polynomial time in the size of an `lr`-RMC. The summary chain will be important also for our analysis of LTL properties.

Linear Temporal Logic.

Formulas in LTL [18] are built from a finite set $Prop$ of propositions using the usual Boolean connectives (eg. $\neg, \vee, \wedge$), the unary temporal connective *Next* (denoted $\bigcirc$) and the binary temporal connective *Until* ($\mathcal{U}$); thus, if ξ, ψ are LTL formulas then $\bigcirc \xi$ and $\xi \mathcal{U} \psi$ are also LTL formulas. Every vertex of the given RMC A is labelled with a subset of $Prop$: the set of propositions that hold at that vertex. That is, there is a given *valuation function* $Val : Q \mapsto 2^{Prop}$. The function can be extended naturally to the infinite Markov chain M_A that corresponds to A, by letting $Val(\langle \beta, u \rangle) = Val(u)$. If $t = s_0, s_1, s_2 \dots$ is a trajectory of M_A and φ is an LTL formula, then we define Satisfaction of the formula by t at step i, denoted $t, i \models \varphi$ inductively on the structure of φ as follows.

- $t, i \models p$ for $p \in Prop$ iff $p \in Val(s_i)$.
- $t, i \models \neg \xi$ iff not $t, i \models \xi$.
- $t, i \models \xi \vee \psi$ iff $t, i \models \xi$ or $t, i \models \psi$.
- $t, i \models \bigcirc \xi$ iff $t, (i+1) \models \xi$.
- $t, i \models \xi \mathcal{U} \psi$ iff there is a $j \geq i$ such that $t, j \models \psi$, and $t, k \models \xi$ for all k with $i \leq k < j$.

We say that the trajectory t satisfies φ iff $t, 0 \models \varphi$. Other useful temporal connectives can be defined from $\mathcal{U}$. The formula $True \mathcal{U} \psi$ means "eventually ψ holds" and is abbreviated $\Diamond \psi$. The formula $\neg(\Diamond \neg \psi)$ means "always ψ holds" and is abbreviated $\Box \psi$.

If φ is an LTL formula and A is an RMC with a valuation function to the propositions of φ, then the set of executions of A (i.e., trajectories of M_A) that satisfy φ is a measurable set. We use $P_A(\varphi)$ to denote the probability of this set. The *qualitative model checking* problem is the following: Given a LTL formula φ and an RMC A (including a valuation function), is $P_A(\varphi) = 1$, i.e. do the executions of A satisfy almost surely the property φ? This is the question of interest if φ is a required property of the model A. Note that the complementary question, i.e., is $P_A(\varphi) = 0$, corresponding to forbidden properties φ, is equivalent, and can be answered by simply negating the property and determining whether $P_A(\neg \varphi) = 1$. The *quantitative model checking* problem is the following: Given an LTL formula φ, an RMC A (including a valuation function), and a rational p, determine whether $P_A(\varphi) < p, = p$, or $P_A(\varphi) > p$.

3 Qualitative Model Checking

We are given RMC A and an LTL formula φ. We assume wlog that the RMC starts at the entry node en_{init} of component A_0 of A, which has no exit. First, we construct from A the graph of the summary Markov chain M'_A; we only need the nodes and edges of M'_A and not the precise transition probabilities. We identify the formula φ with its parse tree T. The leaves of the tree are labelled with atomic propositions and its nonleaf nodes are labelled with temporal or Boolean connectives. Let n be the number of propositions and internal nodes of T; number the propositions and internal nodes from 1 to n bottom-up: first the propositions and then the internal nodes. For each i, let φ_i be the subformula of φ corresponding to the tree T_i rooted at node i.

Let M_A be the (infinite) Markov chain represented by the RMC A. Let $X = x_0x_1x_2\dots$ be an infinite trajectory of M_A starting at some state $x_0 = \langle\beta, u\rangle$. We define the *type* of the trajectory to be a Boolean vector t of length n, where for each i, $t_i = 1$ iff X satisfies the formula φ_i. From the definition of the satisfaction of LTL formulas it follows that the pair (u, t) satisfies the following properties

1. If φ_i is a proposition p, then $t_i = 1$ if p holds at u, else $t_i = 0$.
2. If i is an internal node of the parse tree labelled with a Boolean connective $\neg$ (resp. $\vee$, $\wedge$) and has child j (resp. children j, l), then $t_i = \neg t_j$ (resp. $t_i = t_j \vee t_l$, $t_i = t_j \wedge t_l$).
3. If i is labelled with a temporal connective $\mathcal{U}$ and has children j, l, i.e., $\varphi_i = \varphi_j\mathcal{U}\varphi_l$, then (a) if $t_l = 1$ then also $t_i = 1$, and (b) if $t_j = t_l = 0$ then also $t_i = 0$.

We call any pair (u, t) consisting of a vertex u of the RMC A and a Boolean n-vector t *consistent* if it satisfies these three properties. Similarly we say that the pair (x_0, t) consisting of a state $x_0 = \langle\beta, u\rangle$ of M_A and a vector t is consistent if the pair (u, t) is consistent. Observe that if (u, t) is consistent then the temporal coordinates of t (those corresponding to nodes of φ labelled with temporal connective) determine uniquely the rest of the coordinates of t because of properties (1), (2).

Consider a trajectory $X = x_0x_1x_2\dots$ and suppose that we know the type s of its suffix $x_1x_2\dots$. Then we can determine uniquely the type t of X from s and the state x_0 (more precisely, the vertex u of x_0) as follows: The coordinates t_i corresponding to propositions are determined from u by property (1). For the internal nodes of the parse tree, proceed bottom-up in the tree. Let i be an internal node and suppose that we have determined the coordinates corresponding to the children of i. If i is labelled by a Boolean connective then t_i is determined by property (2) of consistency. If i is labelled by a temporal connective then t_i is determined by property (3) unless i is labelled (i) $\bigcirc$ (Next) or (ii) it is labelled $\mathcal{U}$ (Until) with children j, l and $t_j = 1, t_l = 0$. In case (i), if i has child j, i.e. $\varphi_i = \bigcirc\varphi_j$ then $t_i = s_j$; in case (ii) we must have $t_i = s_i$. Thus, these two properties (i), (ii) and the consistency conditions (1-3) above determine uniquely t from u and s. We will say that t is the type *backwards implied* for vertex u from type s.

The backward implication extends to finite paths: If $\pi = x_0x_1\dots x_k$ is a finite path of M_A and s is a type consistent with the final state x_k, then there is a unique type t that is backwards implied from s and π for the initial state x_0 of the path and its vertex.

We construct a graph G as follows. The nodes of G are all pairs (u, t) where u is a node of the summary chain M'_A and t is a Boolean n-vector such that the pair (u, t) is consistent. We include an edge $(u, t) \to (v, s)$ between two nodes of G if M'_A has an edge $u \to v$ and (a) either the edge is not a summary edge and t is the type that is backwards implied from s for node u, or (b) $u \to v$ is a summary edge, i.e. $u = (b, en), v = (b, ex)$ for some box b, and there is a path π in the RMC corresponding to the summary edge (i.e., goes from u to v with empty context) such that t is the type that is backwards implied from π and s.

We can check whether there exists a path π in the RMC from u to v satisfying the above requirement, as follows: Construct a Recursive State Machine (RSM) $\hat{A}$, called the *augmented RSM*, which has a component $\hat{A}_i$ for each component A_i of the RMC A. There is a vertex (u, t) for each vertex u of A and each type t that is consistent with u; if u is an entry or exit of a component A_i, then (u, t) is an entry or exit of the corresponding component $\hat{A}_i$. If b is a box of A_i mapped to A_j, then there is a corresponding box $\hat{b}$ in $\hat{A}_i$ that is mapped to $\hat{A}_j$; for every entry en of A_j and consistent tuple t, the box $\hat{b}$ has a corresponding call port $((\hat{b}, en), t)$ (the vertex is labelled with the same propositions as en), and we define similarly the return ports of $\hat{b}$. Note that the vertices of the form (u, t), where $u = (b, en)$ or $u = (b, ex)$ was a call port or return port of box b of A, are now ordinary nodes of $\hat{A}$. We include an edge $(u, t) \to (v, s)$ for each pair of vertices $(u, t), (v, s)$ such that t is the type backwards implied from s for u, and A contains an edge $u \to v$, or $u = (b, en)$ and $v = (\hat{b}, en)$ for some box b of A, or $u = (\hat{b}, ex)$ and $v = (b, ex)$. (Note: The reason that we introduced new call ports and return ports is that the trajectories of the Markov chain M_A contain explicit steps corresponding to the recursive calls and returns from the calls. This is a small technical detail.) It is easy to see now that there is a path π in the RMC A from $u = (b, en)$ to $v = (b, ex)$ (with empty context) that corresponds to the summary edge $u \to v$ and such that t is the type that is backwards implied from π and s iff the RSM $\hat{A}$ contains a path from (u, t) to (v, s) (with empty context).

Consider again a trajectory $X = x_0x_1x_2\dots$ of M_A. For each j, let t^j be the type of the path $x_jx_{j+1}\dots$. By our previous remarks, the pair (x_j, t^j) is consistent. Also, note that t^j is the type backwards implied by t^{j+1} and x_i. Let $\hat{X}$ be the sequence $(x_0, t^0), (x_1, t^1), (x_2, t^2)\dots$; we call this the *augmented trajectory* corresponding to X. It corresponds to a trajectory of the RSM $\hat{A}$.

Recall that there is a mapping ρ from trajectories X of the original Markov chain M_A to a trajectory of the summary chain M'_A, or to the symbol $\star$, with the property that $P_A(\rho^{-1}(\star)) = 0$. Suppose that $\rho(X) \neq \star$. Then $\rho(X)$ consists of the vertex parts $u_0u_{i1}u_{i2}\dots$ of a subsequence $x_0x_{i1}x_{i2}\dots$ of X obtained by shortcutting subpaths of X by summary edges. The mapping ρ can be extended to the augmented trajectory $\hat{X}$: $\rho(\hat{X}) = (u_0, t^0), (u_{i1}, t^{i1})\dots$ is obtained from the corresponding subsequence of $\hat{X}$ by keeping only the vertex parts and the types. By our construction of the graph G, $\rho(\hat{X})$ is a path of G.

If $(v_0, s_0), (v_1, s_1), (v_2, s_2) \ldots$ is a sequence of vertex-type pairs, then the *projection* of the sequence on the first component is the sequence $v_0, v_1, v_2 \ldots$ of vertices.

Lemma 2 *1. Every finite or infinite path of G projected on the first component yields a path of M'_A. 2. Conversely, every path of M'_A is the projection of at least one path in G.*

Recall, a vertex u of A is included in summary chain M'_A iff $ne(u) > 0$. Call a pair (u, t) *probable* if there is positive probability that a trajectory of A starting at u does not exit the component of u and has type t. We denote by $P'(u, t)$ the probability that a trajectory from u has type t conditioned on the event that it does not exit u's component.

Lemma 3 *1. If G contains an edge $(u, t) \to (v, s)$ and (v, s) is probable then (u, t) is also probable. 2. In particular, in every strongly connected component C of G, either all nodes are probable or none is.*

Let H be the subgraph of G consisting of probable nodes. By the above lemma, in order to compute H, it suffices to identify which strongly connected components of G are the bottom scc's of H. Then H consists of all the nodes that are ancestors of these bottom scc's. Once we compute the graph H, we can answer the qualitative model checking problem: The trajectories of the given RMC A satisfy the given formula φ almost surely if and only if H does not include any node of the form (en_{init}, t), where en_{init} is the initial node of A (the entry node of the top component) and t is a type with $t_n = 0$. Note that n corresponds to the root of the parse tree of φ, i.e., $\varphi_n = \varphi$, so (en_{init}, t) probable with $t_n = 0$ would mean that there is positive probability that a trajectory starting at en_{init} does not satisfy φ. (Recall that the top component has no exit, so all the trajectories from en_{init} do not exit its component.)

A trajectory X of the RMC (i.e. of the infinite chain M_A) maps with probability 1 to a trajectory $X' = \rho(X)$ of the summary chain M'_A, and the augmented trajectory $\hat{X}$ maps to an augmented trajectory $\hat{X}' = \rho(\hat{X})$ that is a path in G. Call a trajectory X of M_A *typical* if $X' = \rho(X)$ is defined and all pairs of $\hat{X}' = \rho(\hat{X})$ are probable, i.e. if $\hat{X}'$ is a path of the subgraph H. It follows easily from the Markov property that the set of typical trajectories of the RMC starting at the initial state has probability 1. More generally:

Lemma 4 *For every vertex u of the RMC A with $ne(u) > 0$, the probability that a trajectory starting at u does not exit its component and is typical with type t is $ne(u)P'(u, t)$.*

We wish to find the improbable nodes of G and remove them to obtain H. As we noted, it suffices to identify the bottom scc's of H. From the definition of G, if G contains a path from (u, t) to (v, s) then M'_A contains a path from u to v. Therefore, for every scc C of G, the first components of all the nodes of C belong to the same scc K of M'_A. We will say that the scc C *corresponds* to K.

Lemma 5 *If C is a bottom scc of H, then it corresponds to a bottom scc K of M'_A.*

We will now give a necessary condition for a node of G to be probable. Consider a summary edge $(u, t) \to (v, s)$ of G. We say that the edge is probable if the nodes are probable. We label the edge with a subset of $\{1, \ldots n\}$ as follows. A label $l \in \{1, \ldots n\}$ is included in the subset iff the augmented RMC $\hat{A}$ has a path from $\langle \epsilon, (u, t) \rangle$ to $\langle \epsilon, (v, s) \rangle$ that goes through some node $\langle \beta, (z, r) \rangle$ with $r_l = 1$. This can be determined in polynomial time in the size of $\hat{A}$ using the algorithm for Recursive State Machines of [1].

Lemma 6 *If (u, t) is probable, then it satisfies the following condition. For every node i of (the parse tree of) φ labelled $\mathcal{U}$, with corresponding subexpression $\varphi_i = \varphi_j \mathcal{U} \varphi_l$, if $t_i = 1$ then node (u, t) can reach in H (and in G) some probable node (v, s) with $s_l = 1$ or some probable summary edge whose label set includes l.*

It is convenient for the purposes of the analysis to refine the summary graph M'_A into a multigraph M''_A as follows. For each summary edge $u = (b, en) \to v = (b, ex)$ consider all paths of the RMC that give rise to the edge, i.e. paths of the form $\langle \epsilon, u \rangle \to \langle b, en \rangle \to \ldots \langle b, ex \rangle \to \langle \epsilon, v \rangle$. For every type s for the final state, each path implies backwards a type t for u. Let us call two paths *equivalent* if they induce the same mapping from types s at v to types t at u. This gives us a partition of the paths into equivalence classes. Replace the summary edge $u \to v$ with a set of parallel edges, one for each equivalence class. Do the same for all summary edges of M'_A and let M''_A be the resulting multigraph. We can view M''_A also as a (refined) Markov chain where the probability of the summary edges is divided among the parallel edges that replaced it according to the total probability of all paths in each equivalence class. (We do not actually perform this transformation; it is only for the purposes of the analysis.) The multigraph M''_A has the property that for every edge $u \to v$ (whether an ordinary, a summary, or a nesting edge) and every type s for v there is a unique type t that is implied for u by s and the edge. Note that, by construction, the graph G contains an edge $(u, t) \to (v, s)$; we will say that the edge $u \to v$ of M''_A is a projection of the edge $(u, t) \to (v, s)$ of G. (More than one parallel summary edges of M''_A from u to v may be the projection of the same edge of G.) We can extend the notion of projection to paths of G. Obviously M'_A and M''_A have the same scc's (replacing an edge by a set of parallel edges does not change the scc's).

Lemma 7 *Let C be a scc of G and let K be the corresponding scc of M''_A. The following are equivalent.*
1. For every node (v,s) of C, every edge $u \to v$ of K is a projection of some edge $(u,t) \to (v,s)$ of C into (v,s).
2. Every finite path in K is a projection of some path in C.
3. No other scc of G corresponding to K is ancestor of C.

The proof is omitted here; it is nontrivial but it is very similar to the proof of Lemma 5.10 of [6]. The characterization of bottom scc's of H is given by the following Theorem.

Theorem 8 *A scc C of G is a bottom scc of H if and only if the following three conditions are satisfied.*
1. C corresponds to a bottom scc K of M'_A.
2. No other scc of G corresponding to K is ancestor of C.
3. For every subexpression $\varphi_i = \varphi_j \mathcal{U} \varphi_l$ of φ, either all nodes (u,t) of C have $t_i = 0$ or there is a node $(v,s) \in C$ with $s_l = 1$ or there is a summary edge of C whose label set includes l.

Proof. Suppose that C is a bottom scc of H. Then C satisfies conditions 1 and 3 by Lemmas 5 and 6 respectively. Suppose that it does not satisfy (2). Then from Lemma 7 there is a finite path β of K that is not the projection of any path in C. Let (u,t) be any node of C. A trajectory of M''_A starting at u contains with probability 1 the path β (in fact the path occurs infinitely often in the trajectory). Such a trajectory is not the projection of any path in C. It follows that (u,t) is not probable.

Conversely, suppose C satisfies the three conditions. We show that C contains all probable pairs (u,t) whose first component u is in K. From this it follows that C is the only scc of H that corresponds to K, and C is a bottom scc of H because any descendant scc must then also correspond to K. To prove the above fact we show the following lemma. The proof uses induction with a very involved case analysis.

Lemma 9 *Suppose that C satisfies the three conditions of Theorem 8. For every probable pair (u,t) with $u \in K$, the following are true for each $i = 1, \ldots, n$.*
1. There is a node (u,t') of C such that t and t' agree in the first i coordinates.
2. There is a finite path $\alpha(u,t,i)$ of M''_A starting at u such that the type of almost all trajectories of the RMC from u that do not exit u's component, whose summary image has prefix $\alpha(u,t,i)$, agrees with t in the first i coordinates.

Summarizing, the qualitative model checking algorithm for a RMC A and a LTL formula φ works as follows.

1. Construct the graph of the summary chain M'_A.
2. Generate all consistent pairs (u,t), $u \in M'_A$, t a type.
3. Construct the graph G on the consistent pairs.
4. Find the strongly connected components of G, and construct the DAG of scc's.
5. While there is a bottom scc that violates one of the conditions of Theorem 8, remove it from G.
6. If the final graph H contains a node (en_{init}, t) with $t_n = 0$ then reject, else accept.

By our analysis, the final graph is the subgraph H of G induced by the probable pairs.

Step 1 (which depends only on the RMC A, not on the formula φ) can be done in polynomial space in A [11]. The rest of the steps can be done in polynomial time in the size of the graph G and the RSM $\hat{A}$, both of which are polynomial in $|A|$ and exponential in $|\varphi|$ (more specifically, the exponent only depends on the number of temporal operators in φ). If A is a 1-exit RMC, or Bd-RMC, or `lr`-RMC, then Step 1 can be done in polynomial time in A. Thus:

Theorem 10 *Given RMC A & LTL formula φ, we can check whether A satisfies φ with probability 1 in PSPACE in A & EXPTIME in φ. If A is a 1-exit RMC or a Bd-RMC or `lr`-RMC then the time complexity is polynomial in A.*

4 Quantitative Model Checking

We are given a Recursive Markov Chain A and an a LTL formula φ. We are also given a rational number p, and we want to determine whether the probability $P_A(\varphi)$ that a trajectory of A satisfies φ is at least (or at most) p. As mentioned in Section 2, the probability $P_A(\varphi)$ is in general irrational and thus it cannot be computed explicitly. We will construct a system of polynomial equations and inequalities in a set of real variables, one of which stands for the desired probability $P_A(\varphi)$. The system will be constructed in such a way that it has a unique solution. Then we will attach the inequality $P_A(\varphi) \geq p$ (or $P_A(\varphi) \leq p$) and invoke a procedure for the existential theory of the reals to check whether the resulting system is satisfiable.

First we set up the system (1a) $\mathbf{x} = P(\mathbf{x})$ of fixpoint equations for the RMC A which contains one variable $x_{(u,ex)}$ for every vertex u and exit ex of u's component. Recall that we can compute in PSPACE in $|A|$ the set $Q' = \{u \in Q \mid ne(u) > 0\}$ of deficient vertices. We add to (1a) the constraints (1b) $\mathbf{x} \geq 0$; (1c) $y_u = 1 - \sum_{ex \in Ex_{c(u)}} x_{(u,ex)}$ for every vertex u; (1d) $y_u > 0$ for every vertex u in Q'; and (1e) $y_u = 0$ for every vertex u in $Q - Q'$. Let (1) be the system of constraints (1a)-(1e). From the Unique Fixpoint Theorem of RMC's (Theorem 9 of [11]), system (1) has a unique solution $(\mathbf{x}, \mathbf{y})$, and this solution is $x_{(u,ex)} = q^*_{(u,ex)}$ and $y_u = ne(u)$.

Now, we carry out the algorithm for the qualitative model checking. As a result we compute all probable pairs (u,t). For a deficient vertex u and a type t, let $P'(u,t)$ be the

probability that a trajectory X starting at u has type t conditioned on the event that X does not exit u's component. We have a corresponding variable $z(u,t)$ (we only need to include the probable pairs, since the others have probability 0). These variables satisfy several constraints:
(2a) $\sum_t z(u,t) = 1$ for all $u \in Q'$.
(2b) If u is not a call port, then $z(u,t) = \sum_{(v,s)} p'_{u,v} z(v,s)$, where $p'_{u,v}$ is the probability of transition $u \to v$ in the summary Markov chain M'_A, and the sum ranges over all probable pairs (v,s) such that H contains an edge $(u,t) \to (v,s)$.
(2c) If u is a call port, $u = (b, en)$, then $z(u,t) = p'_{u,en} \sum_s z(en,s) + \sum_{(v,s)} p'_{u,v} f_{u,v,t,s} z(v,s)$, where the first sum ranges over all types s such that H contains an edge $(u,t) \to (en,s)$, and the second sum ranges over all exits $v = (b, ex)$ of the box b and types s such that H contains an edge $(u,t) \to (v,s)$ and $f_{u,v,t,s}$ is the fraction of the probability of $u - v$ paths of the RMC for which the type s at v implies backwards the type t at u.

These constraints are justified by the following lemma.

Lemma 11 *Probabilities $P'(u,t)$ satisfy constraints 2a-2c.*

The transition probabilities $p'_{u,v}$ of M'_A are rational functions of the probabilities captured by the variables $(\mathbf{x}, \mathbf{y})$ of system (1). The quantities $f_{u,v,t,s}$ are in general irrational, so we cannot compute them explicitly; however, we will later present a system of constraints with a unique solution that gives precisely these quantities. Suppose for now that we have also determined the parameters $f_{u,v,t,s}$. Then the constraints (2) form a linear system in the variables $z(u,t)$. It turns out that this system has a unique solution.

Lemma 12 *The system (2) of linear equations in the variables $z(u,t)$ has a unique solution.*

We will now construct a system of constraints that determines uniquely the parameters $f_{u,v,t,s}$. Recall the augmented Recursive State machine $\hat{A}$ that we constructed. We add weights to its edges and convert it to a weighted RSM; it will not necessarily be a RMC because the weights out of a node may not sum to 1. The edges of $\hat{A}$ are of the form $(u,t) \to (v,s)$. If A contains the edge $u \to v$ then we let the weight of $(u,t) \to (v,s)$ be the probability of the edge $u \to v$. The other cases are that $u = (b, en)$ and $v = (\hat{b}, en)$, or $u = (\hat{b}, ex)$ and $v = (b, ex)$; in these cases we give these edges weight 1.

Let $u = (b, en)$, $v = (b, ex)$ be a call port and a return port of a box b, and let π be a path in the RMC corresponding to the summary edge $u \to v$ in the summary graph, i.e. π is a path $\langle \epsilon, u \rangle \to \langle b, en \rangle \to \ldots \langle b, ex \rangle \to \langle \epsilon, v \rangle$, where all the intermediate nodes include b in the context. For every type s for the final vertex v, we can infer uniquely types for the vertices along the path, and in particular a type t for the initial vertex u. Thus, the augmented RSM $\hat{A}$ contains for every type s a unique path $\hat{\pi}_s$ corresponding to π which goes from a vertex (u,t) for some t (with empty context) to (v,s) and that path $\hat{\pi}_s$ has the same weight as the probability of the path π. The path $\hat{\pi}_s$ is composed of an edge from (u,t) to an entry $((\hat{b}, en), t')$ of the box $\hat{b}$, then a path that eventually reaches an exit $((\hat{b}, ex), s')$ of the box $\hat{b}$ and finally an edge from the exit to (v,s). Suppose that we have at hand for each entry (en, t') and exit (ex, s') of each component $\hat{A}_i$ of the weighted RSM $\hat{A}$ the sum $h(en, t', ex, s')$ of the weights of all the paths from the entry to the exit. Then we can use them to compute the quantity $x_{en,ex} \cdot f_{u,v,t,s}$ which is the sum of the probabilities of all the paths π corresponding to summary edges $u \to v$ for which type s at v is mapped back to type t at u. Namely, (3a) $x_{en,ex} \cdot f_{u,v,t,s} = \sum h(en, s', ex, t')$ where the summation ranges over all s', t' such that $\hat{A}$ has edges $(u,t) \to ((\hat{b}, en), t')$ and $((\hat{b}, ex), s') \to (v,s)$.

We introduce a variable $h(u,t,ex,s)$ for every pair consisting of a vertex (u,t) of $\hat{A}$ and an exit (ex,s) of its component, to represent the sum of the weights of all the paths from (u,t) that exit at (ex,s). We will construct a set of fixpoint equations, whose solution will be the desired weights. The fixpoint equations are similar to the system of equations for an RMC, given in Section 2. The only difference now is that the weights on the edges out of a vertex may not sum to 1. Let (3b) $\mathbf{h} = \hat{P}(\mathbf{h})$ be this system of equations. We add the constraints (3c): $\mathbf{h} \geq 0$. Finally we add the following constraints (3d): $\sum_t h(u,t,ex,s) = x(u,ex)$ for every triple u, ex, s where u is a vertex of component A_i of the RMC A, ex is an exit of the same component and s is a type. Note that (u,t) is a vertex of component $\hat{A}_i$ and (ex,s) is an exit of the component. The justification for these constraints is the following. For every path π from u to ex (with empty context) and every type s there is a unique corresponding path in $\hat{A}$ to (ex,s), and this path starts at a vertex (u,t) for some t and has weight equal to the probability of the path π. Summing over all such paths π gives the constraint (3d).

We claim now that having fixed the x variables (from constraints (1)), the system (3b-d) has a unique solution. First, note that the intended solution $\mathbf{h}$ representing the weights of the vertex-exit paths is the least fixed point solution of the system (3b-c). This can be shown in the same way as it is shown for Recursive Markov Chains. Namely, if we start with $\mathbf{h} = 0$ and apply repeatedly the operator $\hat{P}$ then the vector will converge to the least fixpoint solution and this coincides with the desired vector of weights. If we pick a fixpoint solution that is strictly greater in some component $h(u,t,ex,s)$ than the correct weights, then the solution will violate a constraint (3d). We conclude that the system (3b-d) has a unique solution. It follows then that (3a) determine uniquely the parameters $f_{u,v,t,s}$.

To summarize, we have three sets of constraints (1),(2),(3). The quantities $p'_{u,v}$ in constraints (2) (the transition probabilities of the summary chain) are ratios, so we first rewrite (2) to clear the denominators so that they become also polynomial equations. If we want to check whether the probability $P_A(\varphi)$, that a trajectory of A satisfies φ, is at least a given threshold p, then we add the constraint (4) $\sum z(en_{init}, t) \geq p$, where the summation ranges over all t with $t_n = 1$. Then we call a procedure for the existential theory of the reals on the system (1-4). Similarly we can determine if the probability is less than p. We can also approximate the probability $P_A(\varphi)$ within any number k of bits of precision by doing a binary search using the above procedure k times.

The size of the system of constraints is polynomial in $|A|$ and exponential in $|\varphi|$. It follows that the complexity is polynomial space in $|A|$ and exponential in $|\varphi|$. For linear RMCs, we can solve the constraints explicitly by solving a series of linear systems of equations.

Theorem 13 *Given RMC A, LTL formula φ and rational value p, we can determine whether the probability $P_A(\varphi)$ that a trajectory of A satisfies φ is $\geq$ (or $\leq$) p in space polynomial in A and exponential in φ. If A is a* `lr`*-RMC, then we can compute $P_A(\varphi)$ exactly in time polynomial in A and exponential in φ.*

5 Lower Bound

Theorem 14 *The qualitative problem of determining whether an RMC A satisfies an LTL formula φ with probability 1 (i.e., whether $P_A(\varphi) = 1$) is EXPTIME-hard (thus EXPTIME-complete). Furthermore, this holds even if the RMC is fixed and each component has 1 entry and 1 exit.*

The theorem is similar to theorems in [3, 17] showing that LTL model checking for Pushdown Systems and Basic Process Algebra (equivalent to RSMs, resp. RSMs with 1 exit) is EXPTIME-hard. We show that the same holds in the probabilistic case, and it holds even for a fixed RMC.

Acknowledgement: Research partially supported by NSF Grant CCF-04-30946. Thanks to Amir Pnueli for asking us about the special case of `lr`-RMCs.

References

[1] R. Alur, K. Etessami, and M. Yannakakis. Analysis of recursive state machines. In *Proc. of 13th Int. Conf. on Computer-Aided Verification*, pages 304–313, 2001.

[2] M. Benedikt, P. Godefroid, and T. Reps. Model checking of unrestricted hierarchical state machines. In *Proc. of ICALP'01*, volume 2076 of *LNCS*, pages 652–666, 2001.

[3] A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Applications to model checking. In *CONCUR'97*, pages 135–150, 1997.

[4] T. Brázdil, A. Kučera, and O. Stražovský. Decidability of temporal properties of probabilistic pushdown automata. In *Proc. of 22nd STACS'05*. Springer, 2005.

[5] J. Canny. Some algebraic and geometric computations in PSPACE. In *Prof. of 20th ACM STOC*, pages 460–467, 1988.

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

[7] J. Esparza, D. Hansel, P. Rossmanith, and S. Schwoon. Efficient algorithms for model checking pushdown systems. In *12th CAV*, volume 1855, pages 232–247. Springer, 2000.

[8] J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. In *Proc. of 19th LICS'04*, 2004.

[9] J. Esparza, A. Kučera, and R. Mayr. Quantitative analysis of probabilistic pushdown automata: expectations and variances. To appear in *Proc. of 20th IEEE LICS*, 2005.

[10] K. Etessami and M. Yannakakis. Recursive Markov Chains, Stochastic Grammars, and Monotone Systems of Nonlinear Equations. In *Proc. of 22nd STACS'05*. Springer, 2005. (Tech. Report, U. Edinburgh, June 2004.)

[11] K. Etessami and M. Yannakakis. Algorithmic Verification of Recursive Probabilistic State Machines. In *Proc. 11th Int. Conf. on Tools and Algorithms for the Construction and Analysis of Systems (TACAS'05)*, 2005.

[12] K. Etessami and M. Yannakakis. Recursive Markov Decision Processes and Recursive Stochastic Games. To appear in *ICALP'05*, 2005.

[13] M. Garey, R. Graham, and D. Johnson. Some NP-complete geometric problems. In *8th ACM STOC*, pages 10–22, 1976.

[14] T. E. Harris. *The Theory of Branching Processes*. Springer-Verlag, 1963.

[15] M. Kwiatkowska. Model checking for probability and time: from theory to practice. In *Proc. 18th IEEE LICS*, pages 351–360, 2003.

[16] C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[17] R. Mayr. Strict lower bounds for model checking BPA. In *Elec. Notes in Theor. Comp. Sci.*, 1998.

[18] A. Pnueli. The temporal logic of programs. In *Proc. 18th Symp. on Foundations of Comp. Sci.*, pages 46–57, 1977.

[19] A. Pnueli and L. D. Zuck. Probabilistic verification. *Inf. and Comp.*, 103(1):1–29, 1993.

[20] J. Renegar. On the computational complexity and geometry of the first-order theory of the reals. parts i,ii, iii. *J. of Symbolic Computation*, pages 255–352, 1992.

[21] P. Tiwari. A problem that is easier to solve on the unit-cost algebraic ram. *Journal of Complexity*, pages 393–397, 1992.

[22] M. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In *Proc. of 26th IEEE FOCS*, pages 327–338, 1985.

[23] M. Y. Vardi and P. Wolper. An automata-theoretic approach to automatic program verification. In *Proc. 1st Symp. on Logic in Comp. Sci. (LICS)*, pages 322–331, 1986.