Efficient Qualitative Analysis of Classes of Recursive Markov Decision Processes and Simple Stochastic Games

Kousha Etessami[1] and Mihalis Yannakakis[2]

[1] LFCS, School of Informatics, University of Edinburgh
[2] Department of Computer Science, Columbia University

Abstract. Recursive Markov Decision Processes (RMDPs) and Recursive Simple Stochastic Games (RSSGs) are natural models for recursive systems involving both probabilistic and non-probabilistic actions. As shown recently [10], fundamental problems about such models, e.g., termination, are undecidable in general, but decidable for the important class of 1-exit RMDPs and RSSGs. These capture controlled and game versions of multi-type Branching Processes, an important and well-studied class of stochastic processes. In this paper we provide efficient algorithms for the *qualitative termination* problem for these models: does the process terminate almost surely when the players use their optimal strategies? Polynomial time algorithms are given for both maximizing and minimizing 1-exit RMDPs (the two cases are not symmetric). For 1-exit RSSGs the problem is in NP∩coNP, and furthermore, it is at least as hard as other well-known NP∩coNP problems on games, e.g., Condon's *quantitative* termination problem for finite SSGs ([3]). For the class of linearly-recursive 1-exit RSSGs, we show that the problem can be solved in polynomial time.

1 Introduction

In recent work [10], we introduced and studied Recursive Markov Decision Processes (RMDPs) and Recursive Simple Stochastic Games (RSSGs), which provide natural models for recursive systems (e.g., programs with procedures) involving both probabilistic and non-probabilistic actions. They define infinite-state MDPs and SSGs that extend Recursive Markov Chains (RMCs) ([8, 9]) with non-probabilistic actions that are controlled by a controller and/or the environment (the "players"). Informally, a recursive model (RMC, RMDP, RSSG) consists of a (finite) collection of finite state component models (resp. MC, MDP, SSG) that can call each other in a potentially recursive manner.

In this paper we focus on the important class of *1-exit RMDPs* and *1-exit RSSGs*, which we will denote by *1-RMDP* and *1-RSSG*. These are RMDPs and RSSGs where every component contains exactly 1 exit node. Without players, 1-RMCs correspond tightly to both Stochastic Context-Free Grammars (SCFGs) and Multi-Type Branching Processes (MT-BPs). Branching processes are an important class of stochastic processes, dating back to the early work of Galton and

B. Durand and W. Thomas (Eds.): STACS 2006, LNCS 3884, pp. 634–645, 2006.

Watson in the 19th century, and continuing in the 20th century in the work of Kolmogorov, Sevastianov, Harris and others for MT-BPs and beyond (see, e.g., [14]). MT-BPs model the growth of a population of objects of distinct types. In each generation each object of a given type gives rise, according to a probability distribution, to a multi-set of objects of distinct types. These stochastic processes have been used in a variety of applications, including in population genetics ([16]), nuclear chain reactions, ([7]), and RNA modeling in computational biology (based on SCFGs) ([22]). SCFGs are also fundamental models in statistical natural language processing (see, e.g., [19]). 1-RMDPs correspond to a controlled version of MT-BPs (and SCFGs): the reproduction of some types can be controlled, while the dynamics of other types is probabilistic as in ordinary MT-BPs; or the controller may be able to influence the reproduction of some types by choosing among a set of probability distributions (e.g., the branching Markov decision chains of [21]). The goal of the controller is either to maximize the probability of extinction or to minimize it (maximize survival probability). This model would also be suitable for analysis of population dynamics under worst-case (or best-case) assumptions for some types and probabilistic assumptions for others. Such controlled MT-BPs can be readily translated to 1-RMDPs, where the types of the MT-BP correspond to the components of the RMDP, extinction in the MT-BP corresponds to termination in the RMDP, and our results can be used for the design of strategies to achieve or prevent extinction.

Among our results in [10], we showed that for maximizing (minimizing) 1-RMDPs, the *Qualitative Termination Problem* (`Qual-TP`), is in NP (coNP, respectively), and that the same problem for 1-RSSGs is in $\Sigma_2^P \cap \Pi_2^P$. `Qual-TP` is the problem of deciding whether player 1 (the maximizer) has a strategy to force termination with probability 1, regardless of the strategy employed by player 2 (the minimizer). (In a maximizing 1-RMDP, the only player present is the maximizer, and in a minimizing 1-RMDP the only player present is the minimizer.)

In this paper we improve significantly on the above results. We show that `Qual-TP`, both for maximizing 1-RMDPs and for minimizing 1-RMDPs, can in fact be decided in polynomial time. It follows easily from this and strong determinacy results from [10], that for 1-RSSGs `Qual-TP` is in NP∩coNP. We show that one can not easily improve on this upper bound, by providing a polynomial time reduction from the *Quantitative Termination Problem* (`Quan-TP`) for finite SSGs ([3]) to the `Qual-TP` problem for 1-RSSGs. Condon [3] showed that for finite SSGs the `Quan-TP` problem, specifically the problem of deciding whether player 1 has a strategy to force termination with probability $\geq 1/2$, is in NP∩coNP. Whether the problem can be solved in P is a well-known open problem, that includes as special cases several other longstanding problems (e.g., "parity games" and "mean-payoff" games). We note (as is already known) that for finite SSGs, `Qual-TP` itself is in polynomial time. We in fact show a more general result, namely, that `Qual-TP` is in polynomial time for the class of 1-RSSGs that are *linearly-recursive.*

Thus, we provide a new class of infinite-state SSGs whose qualitative decision problem is at least as hard as the quantitative decision problem for finite SSGs, and quite possibly harder, but which we can still decide in NP∩coNP. We already showed in [10, 8] that the even harder `Quan-TP` problem for 1-RSSGs can be decided in PSPACE, and that improving that upper bound even to NP, even for 1-RMCs, would resolve a long standing open problem in the complexity of numerical computation, namely the square-root sum problem ([12]).

Most proofs are omitted due to space.

Related Work. Both MDPs and Stochastic Games have a vast literature (see [20, 11]). As mentioned, we introduced RMDPs and RSSGs and studied both quantitative and qualitative termination problems in [10]. We showed that for multi-exit models these problems are undecidable, and that (qualitative) model checking is undecidable even in the 1-exit case. Our earlier work [8, 9] developed the theory and algorithms for RMCs and [6, 2] studied the related model of probabilistic Pushdown Systems (pPDSs).

Our algorithms here were partly inspired by recent unpublished work by Denardo and Rothblum [4, 5] on *Multi-Matrix Multiplicative Systems*. They study families of square nonnegative matrices, which arise from choosing each matrix row independently from a choice of rows, and they give LP characterizations of when the spectral radius of all matrices in the family will be ≥ 1 or > 1. None of our results follow from theirs, but we use techniques similar to theirs, along with other techniques, to obtain our upper bounds.

2 Definitions and Background

A *Recursive Simple Stochastic Game (RSSG)*, A, is a tuple $A = (A_1, \dots, A_k)$, where each *component* $A_i = (N_i, B_i, Y_i, En_i, Ex_i, \mathtt{pl}_i, \delta_i)$ consists of:

- A set N_i of *nodes*, with a distinguished subset En_i of *entry* nodes and a (disjoint) subset Ex_i of *exit* nodes.
- A set B_i of *boxes*, and a mapping $Y_i : B_i \mapsto \{1, \dots, k\}$ that assigns to every box (the index of) a component. To each box $b \in B_i$, we associate a set of *call ports*, $Call_b = \{(b, en) \mid en \in En_{Y(b)}\}$, and a set of *return ports*, $Return_b = \{(b, ex) \mid ex \in Ex_{Y(b)}\}$. Let $Call^i = \cup_{b \in B_i} Call_b$, $Return^i = \cup_{b \in B_i} Return_b$, and let $Q_i = N_i \cup Call^i \cup Return^i$ be the set of all nodes, call ports and return ports; we refer to these as the *vertices* of component A_i.
- A mapping $\mathtt{pl}_i : Q_i \mapsto \{0, 1, 2\}$ that assigns to every vertex a player (Player 0 represents "chance" or "nature"). We assume $\mathtt{pl}_i(ex) = 0$ for all $ex \in Ex_i$.
- A transition relation $\delta_i \subseteq (Q_i \times (\mathbb{R} \cup \{\bot\}) \times Q_i)$, where for each tuple $(u, x, v) \in \delta_i$, the source $u \in (N_i \setminus Ex_i) \cup Return^i$, the destination $v \in (N_i \setminus En_i) \cup Call^i$, and x is either (i) a real number $p_{u,v} \in (0, 1]$ (the transition probability) if $\mathtt{pl}_i(u) = 0$, or (ii) $x = \bot$ if $\mathtt{pl}_i(u) = 1$ or 2. For computational purposes we assume that the given probabilities $p_{u,v}$ are rational. Furthermore they must satisfy the consistency property: for every $u \in \mathtt{pl}_i^{-1}(0)$, $\sum_{\{v' \mid (u, p_{u,v'}, v') \in \delta_i\}} p_{u,v'} = 1$, unless u is a call port or exit node, neither of which have outgoing transitions, in which case by default $\sum_{v'} p_{u,v'} = 0$.

We use the symbols (N, B, Q, δ, etc.) without a subscript, to denote the union over all components. Thus, e.g., $N = \cup_{i=1}^{k} N_i$ is the set of all nodes of A, $\delta = \cup_{i=1}^{k} \delta_i$ the set of all transitions, etc.

An RSSG A defines a global denumerable Simple Stochastic Game (SSG) $M_A = (V = V_0 \cup V_1 \cup V_2, \Delta, \mathtt{pl})$ as follows. The global *states* $V \subseteq B^* \times Q$ of M_A 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 $V \subseteq B^* \times Q$ 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 $\langle \beta, u \rangle \in V$ & $(u, x, v) \in \delta$, then $\langle \beta, v \rangle \in V$ and $(\langle \beta, u \rangle, x, \langle \beta, v \rangle) \in \Delta$.
3. if $\langle \beta, (b, en) \rangle \in V$ & $(b, en) \in Call_b$, then $\langle \beta b, en \rangle \in V$ & $(\langle \beta, (b, en) \rangle, 1, \langle \beta b, en \rangle) \in \Delta$.
4. if $\langle \beta b, ex \rangle \in V$ & $(b, ex) \in Return_b$, then $\langle \beta, (b, ex) \rangle \in V$ & $(\langle \beta b, ex \rangle, 1, \langle \beta, (b, ex) \rangle) \in \Delta$.

The mapping $\mathtt{pl} : V \mapsto \{0, 1, 2\}$ is given as follows: $\mathtt{pl}(\langle \beta, u \rangle) = \mathtt{pl}(u)$ if u is in $Q \setminus (Call \cup Ex)$, and $\mathtt{pl}(\langle \beta, u \rangle) = 0$ if $u \in Call \cup Ex$. The set of vertices V is partitioned into V_0, V_1, and V_2, where $V_i = \mathtt{pl}^{-1}(i)$. We consider M_A with various *initial states* of the form $\langle \epsilon, u \rangle$, denoting this by M_A^u. Some states of M_A are *terminating states* and have no outgoing transitions. These are states $\langle \epsilon, ex \rangle$, where ex is an exit node.

An RSSG where $V_2 = \emptyset$ ($V_1 = \emptyset$) is called a maximizing (minimizing, respectively) *Recursive Markov Decision Process* (RMDP); an RSSG where $V_1 \cup V_2 = \emptyset$ is called a *Recursive Markov Chain* (RMC) ([8, 9]); an RSSG where $V_0 \cup V_2 = \emptyset$ is called a Recursive Graph or *Recursive State Machine*(RSM) ([1]). Define *1-RSSGs* to be those RSSGs where every component has 1 exit, and likewise define *1-RMDPs* and *1-RMCs*. W.l.o.g., we can assume every component has 1 entry, because multi-entry RSSGs can be transformed to equivalent 1-entry RSSGs with polynomial blowup (similar to RSM transformations [1]). This is not so for exits, e.g., qualitative termination is undecidable for multi-exit RMDPs, whereas it is decidable for 1-RSSGs (see [10]). *This entire paper is focused on 1-RSSGs and 1-RMDPs.* Accordingly, some of our notation is simpler than that used for general RSSGs in [10]. We shall call a 1-RSSG (1-RMDP, etc.) *linear* if there in no path of transitions in any component from any return port to a call port.

Our basic goal is to answer qualitative termination questions for 1-RSSGs: "Does player 1 have a strategy to force the game to terminate at exit ex (i.e., reach $\langle \epsilon, ex \rangle$), starting at $\langle \epsilon, u \rangle$, with probability 1, regardless of how player 2 plays?". A *strategy* σ for player i, $i \in \{1, 2\}$, is a function $\sigma : V^* V_i \mapsto V$, where, given the history $ws \in V^* V_i$ of play so far, with $s \in V_i$ (i.e., it is player i's turn to play a move), $\sigma(ws) = s'$ determines the next move of player i, where $(s, \bot, s') \in \Delta$. (We could also allow randomized strategies.) Let Ψ_i denote the set of all strategies for player i. A pair of strategies $\sigma \in \Psi_1$ and $\tau \in \Psi_2$ induce in a straightforward way a Markov chain $M_A^{\sigma,\tau} = (V^*, \Delta')$, whose set of states is the set V^* of histories. Given an initial vertex u, suppose ex is the *unique* exit node of u's component. Let $q_u^{*,\sigma,\tau}$ be the probability that, in $M_A^{\sigma,\tau}$, starting at initial state $\langle \epsilon, u \rangle$ we will eventually terminate, by reaching some $w\langle \epsilon, ex \rangle$, for $w \in V^*$. From general determinacy results (e.g., [18]) it follows that $\sup_{\sigma \in \Psi_1} \inf_{\tau \in \Psi_2} q_u^{*,\sigma,\tau} = \inf_{\tau \in \Psi_2} \sup_{\sigma \in \Psi_1} q_u^{*,\sigma,\tau}$. This is the *value* of the game starting at u, which we denote by q_u^*. We are interested in the following problem: `Qual-TP`: Given A, a 1-RSSG (or 1-RMDP), and given a vertex u in A, is $q_u^* = 1$?

For a strategy $\sigma \in \Psi_1$, let $q_u^{*,\sigma} = \inf_{\tau \in \Psi_2} q_u^{*,\sigma,\tau}$, and for $\tau \in \Psi_2$, let $q_u^{*,\cdot,\tau} = \sup_{\sigma \in \Psi_1} q_u^{*,\sigma,\tau}$. We showed in [10] that 1-RSSGs satisfy a strong form of memoryless determinacy, namely, call a strategy *Stackless & Memoryless (S&M)* if it depends neither on the history of the game nor on the current call stack, i.e., only depends on the current vertex. We call a game *S&M-determined* if both players have S&M optimal strategies.

Theorem 1. *([10]) Every 1-RSSG termination game is S&M-determined. (Moreover, there is an S&M strategy $\sigma^* \in \Psi_1$ that maximizes the value of $q_u^{*,\sigma}$ for all u, and likewise a $\tau^* \in \Psi_2$ that minimizes the value of $q_u^{*,\cdot,\tau}$ for all u.)*

For multi-exit RMDPs and RSSGs things are very different. We showed that even memoryless determinacy fails badly (there may not exist any optimal strategy at all, only ϵ-optimal ones)), and furthermore `Qual-TP` is undecidable (see [10]).

Note that there are finitely many S&M strategies for player i: each picks one edge out of each vertex belonging to player i. For 1-RMCs, where there are only probabilistic vertices, we showed in [8] that `Qual-TP` can be decided in polynomial time, using a spectral radius characterization for certain moment matrices associated with 1-RMCs. It followed, by guessing strategies, that `Qual-TP` for both maximizing and minimizing 1-RMDPs is in NP, and that `Qual-TP` for 1-RSSGs is in $\Sigma_2^P \cap \Pi_2^P$. We obtain far stronger upper bounds in this paper. We will also use the following fact from [10].

Proposition 1. *([10]) We can decide in P-time if the value of a 1-RSSG termination game (and optimal termination probability in a maximizing or minimizing 1-RMDP) is exactly 0.*

In ([10]) we defined a monotone system S_A of nonlinear min-max equations for 1-RSSGs A, and showed that its *Least Fixed Point* solution yields the desired probabilities q_u^* . These systems generalize both the linear Bellman's equations for MDPs, as well as the nonlinear system of polynomial equation for RMCs studied in [8]. Here we recall these systems of equations (with a slightly simpler notation). Let us use a variable x_u for each unknown q_u^*, and let $\mathbf{x}$ be the vector of all $x_u, u \in Q$. The system S_A has one equation of the form $x_u = P(\mathbf{x})$ for each vertex u. Suppose that u is in component A_i with (unique) exit ex. There are 5 cases based on the "*Type*" of u.

1. $u \in Type_1$: $u = ex$. In this case: $x_u = 1$.
2. $u \in Type_{rand}$: $\mathtt{pl}(u)=0$ & $u \in (N_i \setminus \{ex\}) \cup Return^i$: $x_u = \sum_{\{v \mid (u,p_{u,v},v) \in \delta\}} p_{u,v} x_v$. (If u has no outgoing transitions, this equation is by definition $x_u = 0$.)
3. $u \in Type_{call}$: $u = (b, en)$ is a call port: $x_{(b,en)} = x_{en} \cdot x_{(b,ex')}$, where $ex' \in Ex_{Y(b)}$ is the unique exit of $A_{Y(b)}$.
4. $u \in Type_{max}$: $\mathtt{pl}(u) = 1$ & $u \in (N_i \setminus \{ex\}) \cup Return^i$: $x_u = \max_{\{v \mid (u,\bot,v) \in \delta\}} x_v$. (If u has no outgoing transitions, we define $\max(\emptyset) = 0$.)
5. $u \in Type_{min}$: $\mathtt{pl}(u) = 2$ and $u \in (N_i \setminus \{ex\}) \cup Return^i$: $x_u = \min_{\{v \mid (u,\bot,v) \in \delta\}} x_v$. (If u has no outgoing transitions, we define $\min(\emptyset) = 0$.)

In vector notation, we denote the system S_A by $\mathbf{x} = P(\mathbf{x})$.

Given 1-RSSG A, we can easily construct S_A in linear time. For vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, define $\mathbf{x} \leq \mathbf{y}$ to mean $x_j \leq y_j$ for every coordinate j. Let $\mathbf{q}^* \in \mathbb{R}^n$ denote the n-vector of q^*_u's.

Theorem 2. *([10]) Let* $\mathbf{x} = P(\mathbf{x})$ *be the system* S_A *associated with 1-RSSG A. Then* $\mathbf{q}^* = P(\mathbf{q}^*)$*, and for all* $\mathbf{q}' \in \mathbb{R}^n_{\geq 0}$*, if* $\mathbf{q}' = P(\mathbf{q}')$*, then* $\mathbf{q}^* \leq \mathbf{q}'$ *(in other words,* $\mathbf{q}^*$ *is the* Least Fixed Point*, of* $P : \mathbb{R}^n_{\geq 0} \mapsto \mathbb{R}^n_{\geq 0}$*).*

3 Qualitative Termination for 1-RMDPs in P-Time

We show that, for both maximizing 1-RMDPs and minimizing 1-RMDPs, qualitative termination can be decided in polynomial time. Please note: the two cases are not symmetric. We provide distinct algorithms for each of them. An important result for us is this:

Theorem 3. *([8])* `Qual-TP` *for 1-RMCs is decidable in polynomial time.*

We briefly indicate the key elements of that upper bound (please see [8] for more details). Our algorithm employed a spectral radius characterization of *moment matrices* associated with 1-RMCs. Given the system of polynomial equations $x = P(x)$ for a 1-RMC (no min and max types), its moment matrix B is the square Jacobian matrix of $P(x)$, whose (i, j)'th entry is the partial derivative $\partial P_i(x)/\partial x_j$, evaluated at the all 1 vector (i.e., $x_u \leftarrow 1$ for $u \in Q$). We showed in [8] that if the system $x = P(x)$ is decomposed into strongly connected components (SCCs) in a natural way, and we associate a moment matrix B_C to each SCC, C, then $q^*_u = 1$ for every u where x_u is in C, iff either u is of $Type_1$, or [C has successor SCCs and $q^*_v = 1$ for all nodes v in any successor SCC of C, *and* $\rho(B_C) \leq 1$, where $\rho(M)$ is the spectral radius of a square matrix M].

Theorem 4. *Given a maximizing 1-RMDP, A, and a vertex u of A, we can decide in polynomial time whether* $q^*_u = 1$*. In other words, for maximizing 1-RMDPs,* `Qual-TP` *is in* **P***.*

Proof. Given a maximizing 1-RMDP, A, we shall determine for all vertices u, whether $q^*_u = 1$, $q^*_u = 0$, or $0 < q^*_u < 1$. The system of equations $x = P(x)$ for A defines a labeled *dependency graph*, $G_A = (Q, \rightarrow)$, as follows: the nodes Q of G_A are the vertices of A, and there is an edge $u \rightarrow v$ iff x_v appears on the right hand side of the equation $x_u = P_u(x)$. Each node u is labeled by its *Type*. If $u \in Type_{rand}$, i.e., u is a probabilistic vertex, and x_v appears in the weighted sum $P_u(x)$ as a term $p_{u,v}x_v$, then the edge from u to v is labeled by the probability $p_{u,v}$. Otherwise, the edge is unlabeled.

We wish to partition the nodes of the dependency graph into three classes: $Z_0 = \{u \mid q^*_u = 0\}$, $Z_1 = \{u \mid q^*_u = 1\}$, and $Z_\$ = \{u \mid 0 < q^*_u < 1\}$. In our algorithm we will use a fourth partition, $Z_?$, to denote those nodes for which we have not yet determined to which partition they belong. We first compute Z_0. By proposition 1, this can be done easily in P-time even for 1-RSSGs. Once

we have computed Z_0, the remaining nodes belong either to Z_1 or $Z_\$$. Clearly, $Type_1$ nodes belong to Z_1.
Initialize: $Z_1 \leftarrow Type_1$; $Z_\$ \leftarrow \emptyset$; and $Z_? \leftarrow Q \setminus (Z_1 \cup Z_0)$;
Next, we do one "preprocessing" step to categorize some remaining "easy" nodes into Z_1 and $Z_\$$, as follows:

repeat
  if $u \in Z_? \cap (Type_{rand} \cup Type_{call})$ has all of its successors in Z_1
    then $Z_? \leftarrow Z_? \setminus \{u\};\ Z_1 \leftarrow Z_1 \cup \{u\}$;
  if $u \in Z_? \cap Type_{max}$ has some successor in Z_1
    then $Z_? \leftarrow Z_? \setminus \{u\};\ Z_1 \leftarrow Z_1 \cup \{u\}$;
  if $u \in Z_? \cap (Type_{rand} \cup Type_{call})$ has some successor in $Z_0 \cup Z_\$$
    then $Z_? \leftarrow Z_? \setminus \{u\}; Z_\$ \leftarrow Z_\$ \cup \{u\}$;
  if $u \in Z_? \cap Type_{max}$ has all successors in $(Z_0 \cup Z_\$)$
    then $Z_? \leftarrow Z_? \setminus \{u\}; Z_\$ \leftarrow Z_\$ \cup \{u\}$;
until (there is no change to $Z_?$)

The preprocessing step will not, in general, empty $Z_?$, and we need to categorize the remaining nodes in $Z_?$. We will construct a set of linear inequalities (an LP without an objective function) which has a solution iff there are any remaining node in $Z_?$ which belongs in Z_1, and if so, the solution we obtain to the LP will let us find and remove from $Z_?$ some more nodes that belong in Z_1.

Note that, if we can do this, then we can solve our problem, because all we need to do is iterate: we repeatedly do a preprocessing step, followed by the LP step to remove nodes from $Z_?$, until no more nodes can be removed, at which point we are done: the remaining nodes in $Z_?$ all belong to $Z_\$$.

For the LP step, restrict attention to the vertices remaining in $Z_?$. These vertices induce a subgraph of G_A, call it G'_A. Call a remaining probabilistic node u in $Z_?$ *leaky* if it does not have full probability on its outgoing transitions inside G'_A. Note that this happens if and only if some of u's outedges in G_A lead to nodes in Z_1 (otherwise, if u had an outedge to a node in Z_0 or $Z_\$$, it would already have been removed from $Z_?$ during preprocessing). Let $\mathcal{L}$ denote the set of remaining leaky nodes in $Z_?$. We add an extra terminal node t to G'_A, and for every $u \in \mathcal{L}$ we add a probabilistic edge $u \stackrel{p_{u,t}}{\rightarrow} t$, where $p_{u,t} = 1 - \sum_{v \in Z_?} p_{u,v}$.

W.l.o.g., assume that both entries of components and return nodes are probabilistic nodes (this can easily be assured by minor adjustments to the input 1-RSSG). The LP has a variable y_i for every node $i \in Z_?$ that is not $Type_{max}$, and has a variables $y_{i,j}$ for every $Type_{max}$ node $i \in Z_?$ and successor $j \in Z_?$ of i. In addition there are flow variables $f_{i,j,k}$ for each node $i \in Z_?$, and every edge $j \rightarrow k$ in G'_A. The constraints are as follows.

1. For every $j \in Type_{rand} \cup Type_{call}$ that is not a component entry or a return:
$$y_j \geq \sum_{i \rightarrow j \,\wedge\, i \in Type_{rand}} p_{i,j} y_i + \sum_{i \rightarrow j \,\wedge\, i \in Type_{max}} y_{i,j}$$

2. For every $j \in Type_{max}$:

$$\sum_{k} y_{j,k} \geq \sum_{i \to j \,\wedge\, i \in Type_{rand}} p_{i,j} y_i + \sum_{i \to j \,\wedge\, i \in Type_{max}} y_{i,j}$$

3. For every node i that is the entry of a component, say A_r:

$$y_i \geq \sum_{j=(b,en) \in Type_{call} \,\wedge\, Y(b)=r} y_j$$

4. For every node i that is a return node, say of box b: $y_i \geq y_j$, where j is the call node of b.
5. $\sum_i y_i + \sum_{i,j} y_{i,j} = 1$.
6. $y \geq 0$.

Regard the dependency graph as a network flow graph with capacity on each edge $i \to j$ coming out of a max node equal to $y_{i,j}$ and capacity of edges $i \to j$ coming out of the other vertices equal to y_i. We set up one flow problem for each $i \in Z_?$, with source i, sink t and flow variables f_{ijk}.

7. For every vertex i, we have flow conservation constraints on the variables $f_{i,j,k}$, i.e., $\sum_k f_{i,j,k} = \sum_k f_{i,k,j}$, for all nodes $j \in Z_?$, $j \neq i, t$.
8. Nonnegativity constraints: $f_{i,j,k} \geq 0$ for all i, j, k.
9. Capacity constraints: $f_{i,j,k} \leq y_{j,k}$ for every $j \in Type_{max}$ with successor k, and for every node i; and $f_{i,j,k} \leq y_j$ for every $j \in Type_{rand} \cup Type_{call}$ and successor k in G'_A and every node i.
10. Source constraints: $\sum_k f_{i,i,k} = y_i/2^{2m}$, for every $i \in Type_{rand} \cup Type_{call}$, and $\sum_k f_{i,i,k} = \sum_j y_{i,j}/2^{2m}$, for $i \in Type_{max}$, where m is defined as follows. Suppose our LP in constraints (1.-6.) has r variables and constraints, and that its rational entries have numerator and denominator with at most l bits. If there is a solution to (1.-6.), then (see, e.g., [13]), there is a rational solution whose numerators and denominators require at most $m = poly(r, l)$ bits to encode, where $poly(r, l)$ is a polynomial in r and l. Note $r \in O(|G'_A|)$, l is bounded by the number of bits required for the transition probabilities $p_{u,v}$ in A, hence m is polynomial in the input size.

The purpose of constraints (7-10) is to ensure that every vertex with a nonzero y variable can reach a leaky vertex in the subgraph of G'_A induced by the support of the y solution vector.

Lemma 1. *There exists a vertex $u \in Z_?$ such that $q^*_u = 1$ if and only if the LP constraints in (1.–10.) are feasible. Moreover, from a solution to the LP we can find a (partial) strategy for the maximizing player that forces termination from some such u with probability $= 1$.*

So to summarize, we set up and solve the LP. If there is no solution, then for all remaining vertices $u \in Z_?$, $q^*_u < 1$, and thus $u \in Z_\$$. If there is a solution, use the above partial (randomized) strategy for some of the max nodes, leaving the strategy for other nodes unspecified. This allows us to set to 1 some vertices (vertices in the bottom SCC's of the resulting 1-RMC), and thus to move them to Z_1. We can then iterate the preprocessing step and then the LP step until we reach a fixed point, at which point we have categorized all vertices u into one of Z_0, Z_1 or $Z_\$$. □

Theorem 5. *Given a minimizing 1-RMDP, A, and a vertex u of A, we can decide in polynomial time whether* $q^*_u = 1$. *In other words, for minimizing 1-RMDPs,* `Qual-TP` *is in* **P**.

Proof. As in the previous theorem, we want to classify the vertices into $Z_0, Z_\$, Z_1$, this time under optimal play of the minimizing player. We again consider the dependency graph G_A of A. We will again use $Z_?$ to denote those vertices that have not yet been classified.

Initialize: $Z_1 \leftarrow Type_1$; $Z_\$ \leftarrow \emptyset$; and $Z_? \leftarrow Q\setminus (Z_1 \cup Z_0)$;

Next, we again do a "preprocessing" step, which is "dual" to that of the preprocessing we did for maximizing 1-RMDPs, and categorizes some remaining "easy" nodes into Z_1 and $Z_\$$:

repeat
  if $u \in Z_? \cap (Type_{rand} \cup Type_{call})$ has all of its successors in Z_1
    then $Z_? \leftarrow Z_? \setminus \{u\};\ Z_1 \leftarrow Z_1 \cup \{u\}$;
  if $u \in Z_? \cap Type_{min}$ has some successor in $Z_\$$
    then $Z_? \leftarrow Z_? \setminus \{u\};\ Z_\$ \leftarrow Z_\$ \cup \{u\}$;
  if $u \in Z_? \cap (Type_{rand} \cup Type_{call})$ has some successor in $Z_0 \cup Z_\$$
    then $Z_? \leftarrow Z_? \setminus \{u\}; Z_\$ \leftarrow Z_\$ \cup \{u\}$;
  if $u \in Z_? \cap Type_{min}$ has all successors in (Z_1)
    then $Z_? \leftarrow Z_? \setminus \{u\}; Z_1 \leftarrow Z_1 \cup \{u\}$;
until (there is no change to $Z_?$)

Note that, after the preprocessing step, for every edge $u \to v$ in G_A from $u \in Z_?$ to $v \notin Z_?$, it must be the case that $v \in Z_1$ (otherwise, u would have already been moved to $Z_\$$ or Z_0). After preprocessing, we formulate a (different) LP, which has a solution iff there are more nodes currently in $Z_?$ which belong in $Z_\$$. Restrict attention to nodes in $Z_?$, and consider the subgraph G'_A of G_A induced by the nodes in $Z_?$. The LP has a variable y_i for every remaining vertex $i \in Z_?$ such that $i \notin Type_{min}$, and has a variable y_{ij} for every (remaining) node $i \in Type_{min}$, and successor j of i in G'_A. We shall need the following lemma:

Lemma 2. *Consider a square nonnegative matrices B with at most n rows and having rational entries with at most l bits each. If* $\rho(B) > 1$ *then* $\rho(B) \geq 1+1/2^m$ *where* $m = poly(n, l)$ *and* $poly(n, l)$ *is some polynomial in n and l.*

Let $d = (1 + 1/2^m)$. The constraints of our LP are as follows. For the LP we restrict attention to only those nodes j, i in $Z_?$.

1. For every $j \in Type_{rand}$ that is not a component entry or a return, as well as for every $j \in Type_{call}$:

$$dy_j \leq \sum_{i \in Type_{rand} \wedge i \to j} p_{i,j} y_i + \sum_{i \in Type_{min} \wedge i \to j} y_{i,j}$$

2. For every $j \in Type_{min}$:

$$d \sum_k y_{j,k} \leq \sum_{i \in Type_{rand} \wedge i \to j} p_{i,j} y_i + \sum_{i \in Type_{min} \wedge i \to j} y_{i,j}$$

3. For every node i that is the entry of a component, say A_r:

$$dy_i \leq \sum_{j=(b,en) \in Type_{call} \wedge Y(b)=r} y_j$$

4. For every node i that is a return node, say of box b: $dy_i \leq y_j$, where j is the entry node of b.
5. $\sum_i y_i + \sum_{i,j} y_{i,j} = 1$.
6. $y \geq 0$.

Lemma 3. *There exists a vertex $u \in Z_?$ such that $q_u^* < 1$ if and only if the LP in (1. – 6.) is feasible. Moreover, from a solution to the LP we can find a (partial) strategy that forces termination from some such u with probability < 1.*

To summarize, we find Z_0, then do preprocessing to determines the "easy" Z_1 and $Z_\$$ nodes. Then, we set up and solve the LP, finding some more $Z_\$$ vertices, removing them, and iterating again with a preprocessing and LP step, until we exhaust $Z_?$ or there is no solution to the LP; in the latter case the remaining vertices all belong to Z_1. As for a strategy that achieves these assignments, in each iteration when we solve the LP we fix the strategy for certain of the min nodes in a way that ensures that some new vertices will be added to $Z_\$$ and leave the other min nodes undetermined. Moreover, in preprocessing, if $Type_{min}$ nodes get assigned $Z_\$$ based on an outedge, we fix the strategy at that node accordingly. □

4 Qualitative Termination for 1-RSSGs in NP∩coNP

The following is a simple corollary of Theorems 1, 4, and 5.

Corollary 1. *Given a 1-RSSG, A, and given a vertex u of A, we can decide in both NP and coNP whether $q_u^* = 1$. In other words, the qualitative termination problem for 1-RSSGs is in NP∩coNP.*

As the following theorem shows, it will not be easy to improve this upper bound. Note that finite SSGs, defined by Condon [3], are a special case of 1-RSSGs (we can simply identify the terminal node "1" of the SSG with the unique exit of a single component with no boxes). Define the quantitative termination problem for finite SSGs to be the problem of deciding, given a finite SSG G, and a vertex u of G, whether $q_u^* \geq 1/2$. Condon [3] showed that this problem is in NP ∩ coNP, and it has been a major open problem whether this upper bound can be improved to P-time.

Theorem 6. *There is a P-time reduction from the quantitative termination problem for finite SSGs to the qualitative termination problem for 1-RSSGs.*

It is not at all clear whether there is a reduction from qualitative termination for 1-RSSGs to quantitative termination for finite SSGs. Thus, `Qual-TP` for 1-RSSGs appears to constitute a new harder game problem in NP∩coNP.

5 Qualitative Termination for Linear 1-RSSGs in P-Time

We now show that for linear 1-RSSGs, there is a P-time algorithm for deciding `Qual-TP`. This generalizes of course the case of flat games.

Theorem 7. *Given a linear 1-RSSG, and a vertex u, there is a polynomial time algorithm to decide whether $q^*_u = 1$.*

Proof. Given a linear 1-RSSG, A, consider its dependency graph G_A. The nodes of partitioned partitioned into 5 types: $Type_{max}$, $Type_{min}$, $Type_{rand}$, $Type_{call}$, and $Type_1$. Let Q be the set of all vertices of G_A. Our algorithm is depicted in Figure 1. We claim that a call to Prune(Q) returns precisely those vertices in $Z_1 = \{u \mid q^*_u = 1\}$. The proof is omitted due to space. □

Prune(Q)
  $W \leftarrow Q$;
  repeat
    $W \leftarrow W \setminus$PruneMin(W);
    $W \leftarrow$ PruneMax(W);
  until (there is no change in W);
  return W;

PruneMin(W)
  $S \leftarrow W \setminus Type_1$;
  repeat
    if there is a node u in $S \cap (Type_{rand} \cup Type_{max})$ that has a successor in $W \setminus S$, **then** $S \leftarrow S \setminus \{u\}$;
    if there is a node u in $S \cap (Type_{min} \cup Type_{call})$ that has no successor in S, **then** $S \leftarrow S \setminus \{u\}$;
  until (there is no change in S);
  return S;

PruneMax(W)
  $S \leftarrow W$;
  repeat
    if there is a node u in $S \cap (Type_{rand} \cup Type_{min} \cup Type_{call})$ that has a successor in $Q \setminus S$, **then** $S \leftarrow S \setminus \{u\}$;
    if there is a node u in $S \cap Type_{max}$ that has no successor in S, **then** $S \leftarrow S \setminus \{u\}$;
  until (there is no change in S);
  return S;

Fig. 1. P-time qualitative termination algorithm for linear 1-RSSGs

The algorithm applies more generally to *piecewise linear* 1-RSSGs, where every vertex $v \in Type_{call}$ has at most one successor in the dependency graph G_A that is in the same SCC as v.

Acknowledgment. Thanks to Uri Rothblum for providing us a copy of his unpublished work [4, 5]. Research partially supported by NSF Grant CCF-04-30946.

References

1. R. Alur, M. Benedikt, K. Etessami, P. Godefroid, T. W. Reps, and M. Yannakakis. Analysis of recursive state machines. In *ACM Trans. Progr. Lang. Sys.*, 27:786-818, 2005.
2. T. Brázdil, A. Kučera, and O. Stražovský. Decidability of temporal properties of probabilistic pushdown automata. In *Proc. of STACS'05*, 2005.
3. A. Condon. The complexity of stochastic games. *Inf. & Comp.*, 96(2):203–224, 1992.
4. E. V. Denardo and U. G. Rothblum. Totally expanding multiplicative systems. unpublished manuscript, submitted, 2005.
5. E. V. Denardo and U. G. Rothblum. A turnpike theorem for a risk-sensitive Markov decision process with stopping. unpublished manuscript, submitted, 2005.
6. J. Esparza, A. Kučera, and R. Mayr. Model checking probabilistic pushdown automata. In *Proc. of 19th IEEE LICS'04*, 2004.
7. C. J. Everett and S. Ulam. Multiplicative systems, part i., ii, and iii. Technical Report 683,690,707, Los Alamos Scientific Laboratory, 1948.
8. K. Etessami and M. Yannakakis. Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations. In *Proc. of 22nd STACS'05.* Springer, vol. 3404 of LNCS , 2005.
9. K. Etessami and M. Yannakakis. Algorithmic verification of recursive probabilistic state machines. In *Proc. 11th TACAS'05*, Springer, vol. 3440 of LNCS, 2005.
10. K. Etessami and M. Yannakakis. Recursive Markov decision processes and recursive stochastic games. In *Proc. of 32nd Int. Coll. on Automata, Languages, and Programming (ICALP'05)*, Springer, vol. 3580 of LNCS, 2005.
11. J. Filar and K. Vrieze. *Competitive Markov Decision Processes.* Springer, 1997.
12. M. R. Garey, R. L. Graham, and D. S. Johnson. Some NP-complete geometric problems. In *8th ACM STOC*, pages 10–22, 1976.
13. M. Grötschel, L. Lovász, and A. Schrijver. *Geometric Algorithms and Combinatorial Optimization.* Springer-Verlag, 2nd edition, 1993.
14. T. E. Harris. *The Theory of Branching Processes.* Springer-Verlag, 1963.
15. R. J. Horn and C. R. Johnson. *Matrix Analysis.* Cambridge U. Press, 1985.
16. P. Jagers. *Branching Processes with Biological Applications.* Wiley, 1975.
17. K. Mahler. An inequality for the discriminant of a polynomial. *Michigan Math J.*, 11:257–262, 1964.
18. D. A. Martin. Determinacy of Blackwell games. *J. Symb. Logic*, 63(4):1565–1581, 1998.
19. C. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999.
20. M. L. Puterman. *Markov Decision Processes.* Wiley, 1994.
21. U. G. Rothblum, P. Whittle. Growth optimality for branching Markov decision chains. Math. of Operations Research, 7: 582-601, 1982.
22. Y. Sakakibara, M. Brown, R Hughey, I.S. Mian, K. Sjolander, R. Underwood, and D. Haussler. Stochastic context-free grammars for tRNA modeling. *Nucleic Acids Research*, 22(23):5112–5120, 1994.