Stochastic Game Logic

Christel Baier[a] Tomáš Brázdil[b] Marcus Größer[a] Antonín Kučera[b]*

[a]Institut für Informatik, Technische Universität Dresden

[b]Faculty of Informatics, Masaryk University, Czech Republic

E-mail: {baier,groesser}@tcs.inf.tu-dresden.de, {xbrazdil,kucera}@fi.muni.cz

Abstract

*Stochastic game logic (*SGL*) is a new temporal logic that combines features of alternating temporal logic (to formalize the individual views and cooperation and reaction facilities of agents in a multiplayer game), probabilistic computation tree logic and extended temporal logic (to reason about qualitative and quantitative, linear or branching time winning objectives). The paper presents the syntax and semantics of* SGL *and discusses its model checking problem. The model checking problem of* SGL *turns out to be undecidable when the strategies are history-dependent. We show* PSPACE *completeness for memoryless deterministic strategies and the* EXPSPACE *upper bound for memoryless randomized strategies. For the qualitative fragment of* SGL *we show* PSPACE *completeness for memoryless strategies.*

1. Introduction

In this paper, we introduce *Stochastic Game Logic (SGL)*, a new formalism aimed at expressing properties of probabilistic multiplayer games. The logic SGL is closely related to the existing temporal logics such as ECTL [6] and ATL [1]. The syntax of SGL is rather similar to the one of ATL, but there is a conceptual difference in the semantics. The main ingredient of both ATL and SGL is the $\langle\langle A \rangle\rangle$ operator, where A is a set of cooperating players (agents). Intuitively, the formula $\langle\langle A \rangle\rangle\Phi$ says *"there is a strategy for the agents in A such that the formula Φ holds no matter what the other agents do"*. The difference between ATL and SGL becomes apparent when these operators are *nested*. To avoid notation overloading, from now on we use $\langle\langle A \rangle\rangle_{ATL}$ to denote the ATL version of the operator, and reserve the simpler notation $\langle\langle A \rangle\rangle$ exclusively for SGL.

The ATL semantics relies on the standard CTL-like approach where all subformulae are interpreted over the "full" game. For instance, the formula $\langle\langle A \rangle\rangle_{ATL}\Box\langle\langle B \rangle\rangle_{ATL}\Diamond \mathsf{p}$[1] asserts the existence of a strategy α for the agents in A such that $\langle\langle B \rangle\rangle_{ATL}\Diamond \mathsf{p}$ holds (in the "full" game) for all states s that can be reached when the agents in A make their decisions according to α, i.e., from these states s the agents in B have a strategy β in the original game (neglecting the strategy α) which ensures that a state where p holds is reached. Thus in ATL a strategy chosen by the $\langle\langle . \rangle\rangle_{ATL}$ operator *is not propagated* to the inner ATL state formulae. Therefore, properties stating that a certain agent can react on the choices made by another agent cannot be formalized in ATL.

Following the approach of [2, 12, 5], the semantics of the SGL formula $\langle\langle A \rangle\rangle\Phi$ is defined differently. The operator $\langle\langle A \rangle\rangle$ imposes a binding of the strategy α chosen by the agents in A in the same way as first-order quantification $\exists x\phi$ binds the variable x. The scope of the binding is the full formula Φ including its subformulae. However, the nested $\langle\langle A' \rangle\rangle$ operators can revise the binding for the agents in $A \cap A'$. In SGL, the $\langle\langle . \rangle\rangle$ operator can be used in combination with PCTL-like properties [4] that might express qualitative or quantitative probability bounds on path-events, or Boolean combinations thereof. We follow here the concept of extended temporal logics [18, 15, 6], especially the concept of ECTL [6] and use deterministic Rabin automata to describe path properties.

With this concept we can formalize typical multiplayer game properties such as *"the agents in A have a strategy such that whatever strategy the agents in B choose, the agents in C can react to that strategy so that the winning condition holds"*. This is formalized by SGL formula

$$\langle\langle A \rangle\rangle \| B \| \langle\langle C \rangle\rangle \text{"\textit{the winning condition holds}"},$$

where $\|B\|\Phi = \neg\langle\langle B \rangle\rangle\neg\Phi$. This property might or might not be expressible in ATL, depending on the winning condition and whether the game is turn-based or concurrent. In general, the SGL formulae $\langle\langle A \rangle\rangle\|B\|\langle\langle C \rangle\rangle$*"win.cond."* and $\langle\langle A \cup C \rangle\rangle$*"win.cond."* and *not* equivalent, because C's strategies can depend on B's decisions.

* Supported by the research intent MSM0021622419.

1 $\Box$ and $\Diamond$ denote the "always" and "eventually" operator, respectively.

For an example that illustrates the usefulness of the revision of a strategy chosen for a formula $\langle\langle A\rangle\rangle\Phi$ by another $\langle\langle A\rangle\rangle$ operator inside Φ, we consider the following scenario. A banker or money broker has a certain amount x (say 1 Mio Dollars) to work with. His/her goal is to design a strategy (of buying and selling stock, fixed-term deposit, subscription warrants, etc.) for the upcoming months that guarantees with a given probability (e.g. 90) his/her earnings to become larger than 100.000 Dollars in the next year. On the other hand, if everything goes haywire, the banker wants to be able to have at least 120.000 Dollars at his/her disposal within a day, no matter what happens to the rest of the money.

These are two requirements, that cannot be expressed in a formula of the kind $\mathcal{P}_{\geq\vartheta}(\ldots)$. The second one is rather a postulation that allows for a change in the strategy of our banker, which explains the need of the $\langle\langle.\rangle\rangle$ operator in nested form. Thus the appropriate formula looks like this:

$$\langle\langle A\rangle\rangle\Big[(\mathcal{P}_{\geq 0.9}(\Diamond^{\leq 365}(\mathit{earnings} \geq \mathit{100.000}))\wedge$$
$$\mathcal{P}_{\geq 1}\big(\Box\langle\langle A\rangle\rangle\mathcal{P}_{\geq 1}(\mathcal{X}(\mathit{available\ money} \geq \mathit{120.000})\big)\big)\Big]$$

Here A represents the banker, $\mathcal{X}$ represents the "NextStep" operator and one step corresponds to one day.

Note that except for the probabilistic operator $\mathcal{P}(.)$ a formula like the one above can also be expressed in ATL.

Our contribution. This paper introduces a new temporal logic SGL. This logic provides a uniform framework for reasoning about qualitative and quantitative properties of multi-agent systems. We study the decidability and complexity of SGL for various types of strategies. Although parts of our results rely on known results for stochastic games with branching time winning objectives [12, 5], to the best of our knowledge this is the first attempt for defining an ATL-like logic that can express quantitative (PCTL-like) properties. Former approaches with ATL-like modalities have been studied by de Alfaro et al, e.g. [10, 9], for concurrent stochastic games. However, these papers concentrate on qualitative properties and they do not consider Boolean combination of qualitative properties or the nesting of $\langle\langle.\rangle\rangle$ operators. From [12, 5] we deduce that SGL model checking is undecidable for history dependent strategies. For memoryless randomized strategies we give a reduction from the model checking problem into the Tarski algebra which proves the problem to be in EXPSPACE. Moreover, we show that the model checking problem of SGL for memoryless deterministic strategies as well as the model checking problem of the qualitative fragment of SGL for memoryless strategies is PSPACE complete.

Organization. Section 2 introduces our model of probabilistic multiplayer games (PMG) and related notions. The syntax and semantics of our logic SGL will be introduced in Section 3. The model checking problem for SGL on multiplayer games is addressed in Section 4. Section 5 concludes the paper.

2. Probabilistic multiplayer games

We consider turn-based multiplayer games where in each state only one agent makes a move.

Definition 2.1. [Probabilistic multiplayer game]
A probabilistic multiplayer game (PMG) is a tuple $\mathcal{M} = (\mathsf{Agents}, S, \rightarrow, \mathsf{P}, \mathsf{AP}, L)$ where

- $\mathsf{Agents} = \{1,\ldots,n\}$ is a finite set of agents,
- S is a set of states, disjointly partitioned into $S = S_{prob} \cup \bigcup_{a\in\mathsf{Agents}} S_a$,
- $\rightarrow \subseteq S\times S$ is a transition relation[2],
- $\mathsf{P} : S_{prob} \times S \rightarrow [0,1]$ is a function such that $\sum_{u\in S}\mathsf{P}(s,u) = 1$ and $\mathsf{P}(s,t) = 0$ iff $s \not\rightarrow t$ for all $s \in S_{prob}$,
- AP is a finite set of atomic propositions,
- $\mathsf{L} : S \rightarrow 2^{\mathsf{AP}}$ is a labeling function that assigns to each state s the set $\mathsf{L}(s)$ of atomic proposition $\mathsf{p} \in \mathsf{AP}$ which hold in s.

We may regard S as a function that assigns to each agent a a set S_a such that $S_a \cap S_b = \emptyset$ if $a \neq b$. $\Box$

The states $s \in S_a$ are called a-states. The idea is that in the a-states, it is agents a turn to choose a transition $s \rightarrow t$. In the probabilistic states $s \in S_{prob}$, the successor state is chosen randomly according to P.

So far, no restrictions on the cardinality of S have been made. When addressing the model checking problem, only finite PMG, i.e., PMG with finite state space, will be considered.

We write $\mathit{Paths}(s)$ for the set of all sequences $s_0\,s_1\,s_2\ldots \in S^{\omega}$ where $s_0 = s$ and $s_i \rightarrow s_{i+1}$ for all $i \geq 0$. We denote by $\mathit{Succ}(s)$ the set of all successors of s, i.e. $\mathit{Succ}(s) = \{t \in S \mid s \rightarrow t\}$. For a path $\pi = s_0, s_1, \ldots$ we denote by $\pi(j) = s_j$ the j^{th} state of π.

Given a countable set T, let $\mathsf{Distr}(T)$ be the set of all distributions on T, i.e., functions $\mu : T \rightarrow [0,1]$ such that $\sum_{t\in T}\mu(t) = 1$.

For an agent-set $A \subseteq \mathsf{Agents}$, we write S_A for $\bigcup_{a\in A} S_a$ and refer to the states $s \in S_A$ as A-states.

Definition 2.2. [Strategy]
A history-dependent randomized A-strategy (briefly HR-strategy, or simply strategy) is a function $\alpha : S^*S_A \rightarrow \mathsf{Distr}(S)$ such that $\alpha(s_1\ldots s_n\,s)(t) = 0$ if $s \not\rightarrow t$. An α-path denotes a path $s_0\,s_1\,s_2\ldots$ which is consistent with α's decisions, i.e., for all $i \geq 0$, $s_i \in S_A$

2 In the rest of this paper we write $s \rightarrow t$ instead of $(s,t) \in \rightarrow$.

implies $\alpha(s_0 \dots s_i)(s_{i+1}) > 0$. Strategy α is called memoryless (or an MR-strategy) if $\alpha(s_1 \dots s_n s) = \alpha(s)$ for all state-sequences $s_1 \dots s_n$. It is called deterministic (or a HD-strategy) if for all $s_1 \dots s_n s \in S^* S_A$, the distribution $\alpha(s_1 \dots s_n s)$ assigns probability 1 to one successor of s (and 0 to the others). An MD-strategy means a deterministic MR-strategy. □

Given an MR-strategy α, the game $\mathcal{M}^\alpha$ induced by α arises from $\mathcal{M}$ by fixing the decisions for the agents in A according to α. Formally, we define $\mathcal{M}^\alpha = (\mathsf{Agents} \setminus A, S, \to^\alpha, \mathsf{P}^\alpha, \mathsf{AP}, L)$ where

- $s \to^\alpha t$ iff $s \in S_A$ and $\alpha(s)(t) > 0$, or $s \notin S_A$ and $s \to t$;
- $\mathsf{P}^\alpha(s,t) = \alpha(s)(t)$ if $s \in S_A$, and $\mathsf{P}^\alpha(s,t) = \mathsf{P}(s,t)$ if $s \notin S_A$.

An analogous definition of $\mathcal{M}^\alpha$ can be provided for HR-strategies α, in which case we have to deal with the state space S^+ (rather than S) for $\mathcal{M}^\alpha$. Thus, for HR-strategies the induced game might be infinite, while $\mathcal{M}^\alpha$ is finite if $\mathcal{M}$ is finite and α memoryless.

For notational reasons let $\alpha_\emptyset$ denote the empty strategy for the empty agent set.

3. The logic SGL

The logic SGL uses ω-regular languages to specify path properties in the style of the extended computation tree logic ECTL [6]. These languages are expressed by deterministic Rabin automata.

Definition 3.1. [Deterministic Rabin Automata]
A deterministic Rabin automaton $\mathcal{A}$ is a tuple $(Q, \Sigma, q_{init}, \delta, (L_i, R_i)_{i=1}^m)$, where

- Q is a finite set of states,
- Σ is a finite alphabet,
- $q_{init} \in Q$ is the initial state,
- $\delta : Q \times \Sigma \longrightarrow Q$ is a transition function and
- $(L_i, R_i), 1 \leq i \leq m$ is the acceptance condition where $L_i, R_i \subseteq Q$.

Given an infinite word $\pi = \pi_1, \pi_2, \dots \in \Sigma^\omega$ over the alphabet Σ, we call $r(\pi) = q_1, q_2, q_3, \dots$ where $q_1 = q_{init}$ and $q_{i+1} = \delta(q_i, \pi_i)$ the run of $\mathcal{A}$ for the input word π. By

$$\lim(r(\pi)) = \{q \in Q \mid q_j = q \text{ for infinitely many } j\}$$

we denote the limit of $r(\pi)$, i.e., the set of states that occur infinitely often in $r(\pi)$. We say that a set of states $T \subseteq Q$ is accepting iff there exists an index $j \in \{1, \dots, m\}$ such that $T \cap L_j \neq \emptyset$ and $T \cap R_j = \emptyset$. The language accepted by the Rabin automaton $\mathcal{A}$ is defined as

$$L(\mathcal{A}) = \{\pi \in \Sigma^\omega \mid \lim(r(\pi)) \text{ is accepting}\}.$$

□

Stochastic game logic. Our logic, called stochastic game logic (SGL), borrows ideas from ATL (and the ATL-like formalisms for stochastic games [10, 9]), extended computation tree logic ECTL [6], and the game logic GL of [1]. The probabilistic fragment of SGL contains a PCTL-like probabilistic operator which allows to reason about the probabilities for ω-regular properties, expressed by a deterministic Rabin automaton. This leads to the following abstract syntax for SGL formulae.

$$\Phi ::= \mathsf{p} \;\Big|\; \Phi_1 \wedge \Phi_2 \;\Big|\; \neg\Phi \;\Big|\; \langle\!\langle A \rangle\!\rangle \Phi \;\Big|\; \mathcal{P}_{\bowtie\lambda}(\mathcal{A}; \Phi_1, \dots, \Phi_k)$$

where $\mathsf{p} \in \mathsf{AP}$ is an atomic proposition, $A \subseteq \mathsf{Agents}$ a set of agents, $\bowtie \in \{<, \leq, >, \geq\}$ a comparison operator, and $\lambda \in [0,1]$ a probability bound. $\mathcal{A}$ is a deterministic Rabin automaton over the alphabet $2^{\{1,\dots,k\}}$.

Semantics. The formula $\langle\!\langle A \rangle\!\rangle \Phi$ requires the existence of an XY strategy[3] α for the A agents, such that the subformula Φ is satisfied in the game induced by α. Thus, Φ is interpreted over $\mathcal{M}^\alpha$, but decisions already made by α can be changed by another $\langle\!\langle B \rangle\!\rangle$ operator in Φ for the agents in $A \cap B$. This is the crucial difference from the standard ATL-semantics where the state subformulae of Φ are interpreted over the full game $\mathcal{M}$. As decisions once made by a $\langle\!\langle . \rangle\!\rangle$ operator might be changed by a nested $\langle\!\langle . \rangle\!\rangle$ operator, we need to keep track on the strategy decisions that have already been made.

The $\mathcal{P}_{\bowtie\lambda}(\mathcal{A}; \Phi_1, \dots, \Phi_k)$ operator has the standard PCTL* semantics, meaning that in the current game $\mathcal{M}^\alpha$, for all HR strategies β of the remaining agents, the probability measure of all paths accepted by the automaton $\mathcal{A}$ in the Markov chain $(\mathcal{M}^\alpha)^\beta$ matches the probability bound λ. Here, a path is accepted by the automaton $\mathcal{A}$ if its projection to words over $2^{\{1,\dots,k\}}$ indicating which of the formulae $\Phi_1, \dots, \Phi_k$ are satisfied in each of the states of the path, is in $L(\mathcal{A})$.

Let $\mathcal{M} = (\mathsf{Agents}, S, \to, \mathsf{P}, \mathsf{AP}, L)$ be a probabilistic multiplayer game and XY a class of strategies (i.e., XY is either MD, MR, HD, or HR). We define a satisfaction relation $s, A, \alpha \models^{\mathcal{M}}_{\mathrm{XY}} \Phi$ where s is a state in $\mathcal{M}$ and α is an XY strategy for A. The intuitive meaning is that s satisfies Φ in the induced game M^α. Note that we need to keep track of the strategy decisions already made. The rules for the satisfaction relation are given in Figure 1. The meaning of the newly employed symbols is the following:

- $\alpha_{|A \setminus B}$ is the strategy for the agent-set $A \setminus B$ that coincides with strategy α for all paths ending in a state $s \in S_A \setminus S_B$.
- $(\alpha \leftarrow \beta)$ denotes the strategy for the agents in $A \cup B$ such that the agents in $A \setminus B$ behave according to the strategy α and the agents in B behave according to the

3 Here XY stands for MD, MR, HD, or HR.

$s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \mathsf{p}$	iff	$\mathsf{p} \in L(s)$
$s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \Phi_1 \wedge \Phi_2$	iff	$s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \Phi_1$ and $s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \Phi_2$
$s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \neg\Phi$	iff	$s, A, \alpha \not\models^{\mathcal{M}}_{\text{XY}} \Phi$
$s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \langle\langle B \rangle\rangle \Phi$	iff	there exists an XY-strategy β for B in the PMG $\mathcal{M}^{\alpha_{\|A \setminus B}}$ such that $s, A \cup B, (\alpha \leftarrow \beta) \models^{\mathcal{M}}_{\text{XY}} \Phi$
$s, A, \alpha \models^{\mathcal{M}}_{\text{XY}} \mathcal{P}_{\bowtie\lambda}(\mathcal{A}; \Phi_1, \ldots, \Phi_k)$	iff	for all HR-strategies β for $\mathsf{Agents} \setminus A$ in the PMG $\mathcal{M}^{\alpha}$: $Prob_{(\mathcal{M}^{\alpha})^{\beta}}\left(\{\pi \in Paths(s) \mid \tilde{\pi}^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;XY} \in L(\mathcal{A})\}\right) \bowtie \lambda$

Figure 1. Semantics of SGL

strategy β, i.e., decisions already made by an agent in $A \cap B$ are neglected and a new strategy is chosen. Formally, given a path $\pi = s_1, \ldots, s_n$, we put

$$(\alpha \leftarrow \beta)(\pi) = \begin{cases} \alpha(\pi) & \text{if } s_n \in S_A \setminus S_B \\ \beta(\pi) & \text{if } s_n \in S_B \end{cases}$$

- $\tilde{\pi}^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;XY} \in (2^{\{1,\ldots,k\}})^{\omega}$ is defined as follows.

$$\tilde{\pi}^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;XY}(i) = \{j \mid \pi(i), A, \alpha \models^{\mathcal{M}}_{XY} \Phi_j\}.$$

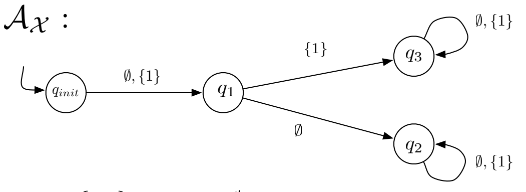
Given a formula Φ, we denote by $Sat^{\mathcal{M}}_{XY}(\Phi)$ the set of states of $\mathcal{M}$ that satisfy Φ, i.e.,

$$Sat^{\mathcal{M}}_{XY}(\Phi) = \{s \in S : s, \emptyset, \alpha_{\emptyset} \models^{\mathcal{M}}_{XY} \Phi\}.$$

Note that the class of strategies for agents that explicitly appear in some $\langle\langle . \rangle\rangle$ operator is restricted to XY, while the remaining agents can always use unrestricted (i.e., HR) strategies. Intuitively, this is because the remaining agents are usually interpreted as unpredictable intruders, and hence their worst possible behaviour must be taken into account. On the other hand, the strategy for cooperating agents should be as simple as possible. The results in [2] yield that the satisfaction relations $\models_{\text{HD}}$, $\models_{\text{HR}}$, $\models_{\text{MD}}$ and $\models_{\text{MR}}$ are pairwise distinct. However, it follows from the results of [11, 14, 4] that the semantics of the $\mathcal{P}_{\bowtie\lambda}(\mathcal{A}; \Phi_1, \ldots, \Phi_k)$ operator is the same, no matter whether we quantify over all HR strategies or over all HD strategies for the remaining agents.

Example 3.2. *Since the syntax of* SGL *does not contain the standard temporal operators such as "NextStep" (denoted $\mathcal{X}$) or "Always" (denoted $\Box$), it is worth noting how to express these operators in* SGL*. For example, the formula $\mathcal{P}_{\bowtie\lambda}(\mathcal{X}\Phi_1)$ can be expressed in* SGL *as $\mathcal{P}_{\bowtie\lambda}(\mathcal{A}_{\mathcal{X}}; \Phi_1)$, where the Rabin automaton $\mathcal{A}_{\mathcal{X}}$ is shown below.*

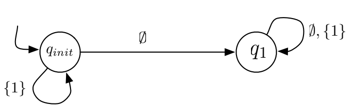
$\mathcal{A}_{\mathcal{X}}$:



$L_1 = \{q_3\},\ R_1 = \emptyset$

The $\Box$ operator is expressed by a Rabin automaton $\mathcal{A}_{\Box}$ defined as follows.

$\mathcal{A}_{\Box}$:



$L_1 = \{q_{init}\},\ R_1 = \emptyset$

Similarly, we can define automata for other temporal operators such as "Until", "WeakUntil", and "Eventually".

The relationship between SGL **and other logics.** In this paragraph we show that formulae of other well-known logics such as CTL, CTL*, PCTL, PCTL*, ATL, etc., can effectively be translated into SGL.

The standard (non-probabilistic) CTL is expressible in SGL. CTL is interpreted over labelled transition systems (Kripke structures) which can be seen as a PMG with no probabilistic states and only one agent. In the CTL semantics each path quantifier $\exists, \forall$ is interpreted over the "full" system. Since in SGL strategies chosen by the $\langle\langle\{1\}\rangle\rangle$ operator can be overwritten by another $\langle\langle\{1\}\rangle\rangle$ operator, we can embed CTL as follows. Given a labelled transition system T and a CTL formula Φ, let Φ' be the SGL formula obtained from Φ by substituting each occurrence of the existential quantifier $\exists(.)$ by $\langle\langle\{1\}\rangle\rangle\mathcal{P}_{\geq 1}(.)$, and each occurrence of the universal quantifier $\forall(.)$ by $\|\{1\}\|\mathcal{P}_{\geq 1}(.)$, where $\|.\| = \neg\langle\langle . \rangle\rangle\neg$. Then it holds that $Sat_{CTL}(\Phi) = Sat^{T}_{MD}(\Phi')$. Note that Φ' is not conform to our SGL syntax as it uses temporal operators like "Always" and "NextStep" instead of Rabin automata to express path properties. But, as indicated in Example 3.2, the formula Φ' can be transformed into an equivalent SGL formula.

The same transformation embeds CTL* into SGL, but in this case, the SGL formula has to be interpreted over the HD-semantics. That is, given a CTL* formula Φ, it holds that $Sat_{CTL^*}(\Phi) = Sat^{T}_{HD}(\Phi')$. As CTL* uses LTL path formulae we need more complicated automata as the ones introduced in example 3.2. However, this does not pose any real problems as the languages expressible by LTL formulae are contained in the ω-regular languages, and deterministic Rabin automata are as expressive as ω-regular languages [13, 16, 17].

The standard PCTL (interpreted over Markov decision processes (MDP)) can also be embedded into SGL. Each Markov decision process M can be seen as a PMG with only one agent. In PCTL, there are no path quantifiers like $\exists$ and $\forall$. The semantics of the PCTL $\mathcal{P}_{\bowtie\lambda}(.)$ operator implicitly quantifies over all strategies in the given MDP M. This is the same as in our SGL semantics. Moreover, given a formula $\mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$, the formulae $\Phi_1,\ldots,\Phi_k$ are interpreted over the same system as the formula $\mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$. Hence, we do not need a transformation from PCTL to SGL as in the CTL case above; we only need the transformation from LTL path formulae to Rabin automata. Given a PCTL formula Φ and an MDP M, it holds that $Sat_{PCTL}(\Phi) = Sat^{M}_{MD}(\Phi)$. Again, the temporal operators have to be substituted by the appropriate automata.

Similarly, PCTL* embeds into SGL. Let M be an MDP and Φ be a PCTL* formula. Then $Sat_{PCTL^*}(\Phi) = Sat^{M}_{HD}(\Phi)$.

Remark 3.3. *Let M be and MDP (which can be understood as PMG with one agent 1), and let Φ be a* SGL *formula that is obtained from some PCTL* formula in the way indicated above. In particular, note that Φ does not contain the $\langle\langle.\rangle\rangle$ operator. Assume that Φ has nested $\mathcal{P}_{\bowtie\lambda}(.)$ operators, so it might look like this: $\mathcal{P}_{\bowtie\lambda}(\ldots\mathcal{P}_{\bowtie'\lambda'}(\ldots)\ldots)$. Let Φ' be the formula obtained from Φ by substituting each occurrence of $\mathcal{P}_{\bowtie\lambda}(.)$ by $\|\{1\}\|\mathcal{P}_{\bowtie\lambda}(.)$. Then $Sat^{M}_{XY}(\Phi) = Sat^{M}_{HD}(\Phi')$ for each strategy class XY[4], whereas $Sat^{M}_{HD}(\Phi) \neq Sat^{M}_{HD}(\|\{1\}\|\Phi)$ in general. This is because although in Φ' the $\|\{1\}\|$ operator of the outermost $\mathcal{P}_{\bowtie\lambda}(.)$ operator fixes a strategy for the only agent, this strategy can be overwritten by the $\|\{1\}\|$ operator of a nested $\mathcal{P}_{\bowtie\lambda}(.)$ operator. Thus, we get the standard PCTL semantics. On the other hand, the formula $\|\{1\}\|\Phi$ fixes a strategy α by the $\|\{1\}\|$ operator and evaluates the outermost $\mathcal{P}_{\bowtie\lambda}(.)$ operator on the Markov chain M^{α}. This means that also the nested $\mathcal{P}_{\bowtie\lambda}(.)$ operators are evaluated over $\mathcal{M}^{\alpha}$ which gives the above inequality.*

Even ATL is expressible in SGL. In standard ATL, the $\langle\langle.\rangle\rangle_{ATL}$ operator is followed by a path formula. The ATL semantics of the formula $\langle\langle A\rangle\rangle_{ATL}\varphi$ yields the existence of an HD strategy for the A-agents such that for all HD strategies of the agents not in A, the path formula φ holds for the unique path that is determined by the chosen strategies. As already mentioned, the strategy chosen for the A-agents is not propagated to the subformulae. Given a PMG $\mathcal{M}$ without any probabilistic states and an ATL formula Φ, let Φ' be the SGL formula obtained from Φ by substituting each occurrence of $\langle\langle A\rangle\rangle_{ATL}\varphi$ by $\langle\langle A\rangle\rangle\|\mathsf{Agents}\setminus A\|\mathcal{P}_{\geq 1}(\varphi)$. It holds that $Sat_{ATL}(\Phi) = Sat^{\mathcal{M}}_{HD}(\Phi')$. The corresponding results hold also for ATL* and the SGL satisfaction under the HD-semantics.

In [1], the authors introduce an extension of ATL called game logic (GL). In contrast to ATL, where the operator $\langle\langle.\rangle\rangle_{ATL}$ is followed by a path formula and the semantics implicitly quantifies over all paths, the $\langle\langle.\rangle\rangle$ operator in game logic can also be followed by an existential path quantifier $\exists$. A formula of the kind $\Phi = \langle\langle A\rangle\rangle(\exists\Box\varphi_1 \wedge \exists\Box\varphi_2)$ is expressible in GL. Φ assert the existence of a strategy α for the agents in A, such that for some behavior of the remaining agents φ_1 is always true, and for some (possibly different) behavior of the remaining agents φ_2 is always true. Thus, the chosen strategy α is propagated to the inner subformulae. Nevertheless, the semantics of GL does not propagate strategies chosen by $\langle\langle.\rangle\rangle$ operators to nested $\langle\langle.\rangle\rangle$ operators. For example, the GL formula $\langle\langle A\rangle\rangle\langle\langle B\rangle\rangle\Phi$ is equivalent to $\langle\langle B\rangle\rangle\Phi$. Hence, the GL semantics is more alike to the standard CTL* semantics and differs crucially from our SGL semantics. Therefore, GL fails to express typical game properties like *"player B can react to the strategy chosen by player A"*.

ATL-like approaches to reason about stochastic games and qualitative winning objectives have been introduced by de Alfaro et al [10, 9]. They use ATL-like formulae, such as $\langle\langle A\rangle\rangle_{\mathrm{almost}}\psi$ or $\langle\langle A\rangle\rangle_{\mathrm{positive}}\psi$, to formalize the existence of a strategy for agents in A such that the condition specified by ψ holds almost surely or with positive probability. Our framework generalizes these concepts to the quantitative setting and allows to express, e.g., properties asserting that the agents in A can cooperate so that the probability of the event specified by ψ is within a certain interval, or so that a Boolean combination of such PCTL-like formulae holds, no matter how the other agents behave. The ATL-like formulae $\langle\langle A\rangle\rangle_{\mathrm{almost}}\psi$ or $\langle\langle A\rangle\rangle_{\mathrm{positive}}\psi$ of [10, 9] are encoded in SGL by the formulae $\langle\langle A\rangle\rangle\mathcal{P}_{=1}(\psi)$ and $\langle\langle A\rangle\rangle\mathcal{P}_{>0}(\psi)$, respectively. However, SGL cannot express the limit operator $\langle\langle A\rangle\rangle_{\mathrm{limit}}$ of [10].

4. Model checking SGL

The model checking problem for SGL addresses the question whether for a given finite PMG $\mathcal{M}$, a state s of $\mathcal{M}$ and SGL-formula Φ it holds that $s \in Sat^{\mathcal{M}}_{\mathrm{XY}}(\Phi)$ for a given strategy class XY. We first state our main result:

Theorem 4.1. *The* SGL *model checking problem is*

- *undecidable for HD and HR semantics;*
- PSPACE-*complete for MD semantics;*
- PSPACE-*hard and in* EXPSPACE *for MR semantics.*

4 Realize again that Φ does not contain any $\langle\langle.\rangle\rangle$ operator.

For the qualitative fragment[5] *of* SGL *and MR semantics, the model-checking problem is* PSPACE*-complete.*

The undecidability result for history-dependent strategies follows from the undecidability result for $1\frac{1}{2}$-player games and PCTL stated in [5]. More precisely, [5] yields the undecidability of the model checking problem for PMG with a singleton agent set $\{a\}$ and SGL formulae of the form $\langle\langle a\rangle\rangle\Phi$ where Φ is a PCTL formula.

We now turn to the proof of the decidability of the SGL model checking problem for the MR semantics. For this we extend the results of [12] on the MR-controller synthesis for MDPs (viewed as $1\frac{1}{2}$-player games) and PCTL specifications by showing that the model checking problem for SGL with MR semantics can effectively be encoded by closed formulae of $(\mathbb{R},*,+,\leq)$. The EXPSPACE upper bound is then obtained by analyzing the size of the resulting formula of $(\mathbb{R},*,+,\leq)$.

The major difficulty lies in the treatment of nested $\langle\langle\cdot\rangle\rangle$ operators. This requires an appropriate representation of the induced games that serve to evaluate subformulae in the scope of a given $\langle\langle\cdot\rangle\rangle$ operator. Moreover, in the encoding that we provide, we assume that the set of probabilistic states S_{prob} is empty. This is no restriction, as every PMG $\mathcal{M} = (\mathsf{Agents}, S, \rightarrow, \mathsf{P}, \mathsf{AP}, \mathsf{L})$ and every SGL formula Φ can efficiently be transformed into a PMG $\mathcal{M}'$ and a SGL formula Φ' where $\mathcal{M}'$ does not have any probabilistic states and

$$Sat^{\mathcal{M}}_{MR}(\Phi) \;=\; Sat^{\mathcal{M}'}_{MR}(\Phi').$$

This transformation works as follows. We add a new agent *PROB* that controls the former probabilistic states, i.e., $\mathcal{M}' = (\mathsf{Agents}', S', \rightarrow, \mathsf{P}', \mathsf{AP}', \mathsf{L}')$, where

- $S' = S$, where $S'_{PROB} = S_{prob}$ and $S'_a = S_a$ for every agent $a \neq PROB$ (hence, $S'_{prob} = \emptyset$),
- $\mathsf{Agents}' = \mathsf{Agents} \cup \{PROB\}$,
- $\mathsf{P}' = \emptyset$,
- $\mathsf{AP}' = \mathsf{AP} \cup S$,
- $\mathsf{L}'(s) = \mathsf{L}(s) \cup \{s\}$

The formula Φ' looks as follows:

$$\Phi' = \langle\langle PROB\rangle\rangle \left(\Phi \wedge \bigwedge_{\substack{s\in S'_{PROB}\\ t\in Succ(s)}} \mathcal{P}_{\geq 1}\big(\Box(s \Rightarrow \mathcal{P}_{=\mathsf{P}(s,t)}(\mathcal{X}t))\big)\right)$$

Here $\Box$ denotes the "Always" operator and $\mathcal{X}$ denotes the "NextStep" operator. Note that in $\mathcal{M}'$, each state $s \in S$ is also an atomic proposition. Thus, the above transformation just adds a new agent *PROB* for the probabilistic states of S (and views those states as non-probabilistic ones in S') and requires in Φ' that whenever a state $s \in S'_{PROB}$ is visited, the agent *PROB* chooses a successor $t \in Succ(s)$ with the probability that the transition $s \rightarrow t$ had in $\mathcal{M}$. It is easy to see that $Sat^{\mathcal{M}}_{MR}(\Phi) = Sat^{\mathcal{M}'}_{MR}(\Phi')$ (note that there is no application of $\langle\langle PROB\rangle\rangle$ in the formula Φ). In SGL syntax, Φ' is written as

$$\Phi' = \langle\langle PROB\rangle\rangle \left(\Phi \wedge \bigwedge_{\substack{s\in S_{prob}\\ t\in Succ(s)}} \mathcal{P}_{\geq 1}\big(\mathcal{A}_{\Box}; \Phi^{s,t}\big) \right),$$

where $\Phi^{s,t} = \big(s \Rightarrow \mathcal{P}_{=\mathsf{P}(s,t)}\,(\mathcal{A}_{\mathcal{X}}; t)\big)$ and the automata $\mathcal{A}_{\Box}$ and $\mathcal{A}_{\mathcal{X}}$ are as shown in Example 3.2. Thus, in the following we assume that the given PMG has no probabilistic states.

Let $\mathcal{M} = (\mathsf{Agents}, S, \rightarrow, \mathsf{P}, \mathsf{AP}, \mathsf{L})$ be a finite PMG and Φ a SGL formula. According to the semantics of SGL, we need a formalism to handle the satisfaction relation $., A, \alpha \models^{\mathcal{M}}_{\mathrm{MR}}$ for SGL formulae where α is an MR-strategy for the agent set A. Thus, we need to keep track of the set A of agents that have already chosen a strategy. This strategy will be encoded by first-order variables $Y_{s\rightarrow t}$ for $s \in S_A, t \in Succ(s)$. The variable $Y_{s\rightarrow t}$ represents the probability $\alpha(s)(t)$ with which the strategy α chooses the s-successor t.

Following the semantics, we inductively define first order formulae $\tau_C(s,\Phi)$ over $(\mathbb{R},*,+,\leq)$ such that $\tau_C(s,\Phi)$ is valid iff $s, C, \gamma \models^{\mathcal{M}}_{MR} \Phi$, where the strategy γ is given by the values of the variables $Y_{s\rightarrow t}$ for $s \in S_C, t \in Succ(s)$. Thus,

$$s \in Sat^{\mathcal{M}}_{MR}(\Phi) \quad \text{iff} \quad \tau_{\emptyset}(s,\Phi) \text{ is valid.}$$

The definition of $\tau_C(s,\Phi)$ is given in Figure 2.

The cases when Φ is an atomic proposition, a conjunction or a negation are obvious. In the case when $\Phi = \langle\langle A\rangle\rangle\Phi'$, the formula $\tau_C(s, \langle\langle A\rangle\rangle\Phi')$ existentially quantifies over the variables $Y_{t\rightarrow u}$ for $t \in S_A, u \in Succ(t)$, requiring that the choice of the $Y_{t\rightarrow u}$ represents a strategy for the agent set A and that with this choice, the formula $\tau_{C\cup A}(s,\Phi')$ holds. Note that if $C \cap A \neq \emptyset$, then $C \cap A$ strategies chosen in an earlier phase will be overwritten by the new choice of the $Y_{t\rightarrow u}$ variables.

The most complicated case is $\Phi = \mathcal{P}_{\unlhd\lambda}(\mathcal{A}; \Phi_1, \ldots, \Phi_k)$. First, let us explain how we could effectively solve the problem whether $s, A, \alpha \models^{\mathcal{M}}_{MR} \Phi$ for a given set of agents $A \subseteq \mathsf{Agents}$ and a given A-strategy α. Let us assume that the sets $Sat^{\mathcal{M}}_{MR}(\Phi_j, A, \alpha) = \{s \in S \mid s, A, \alpha \models^{\mathcal{M}}_{MR} \Phi_j\}, 1 \leq j \leq k$ are known. The semantics requires that for all *MR* strategies β for the remaining agents $\notin A$ it holds that the set of all paths in M^{α} that start in s and are accepted by the deterministic Rabin automaton $\mathcal{A}$ has probability $\unlhd\lambda$ in $(\mathcal{M}^{\alpha})^{\beta}$. We denote by $\mathcal{M}^{\alpha} \times \mathcal{A}$ the standard product of the PMG $\mathcal{M}^{\alpha}$ and the automaton $\mathcal{A}$, i.e., if $\mathcal{A} = (Q, 2^{\{1,\ldots,k\}}, q_{init}, \delta, (L_i,R_i)_{i=1}^{m})$, then $\mathcal{M}^{\alpha} \times \mathcal{A} = (S \times Q, \rightarrow_{\mathcal{M}^{\alpha}\times\mathcal{A}}, \mathcal{P}^{\mathcal{M}^{\alpha}\times\mathcal{A}})$, where

5 The qualitative fragment of SGL is obtained by restricting the syntax of the $\mathcal{P}_{\bowtie\lambda}(.)$ operator so that it can only use the probability bounds > 0 or ≥ 1.

$$
\begin{array}{lcl}
\tau_C(s,\mathsf{p}) & = & \text{true, if } \mathsf{p}\in L(s)\\
\tau_C(s,\Phi_1\wedge\Phi_2) & = & \tau_C(s,\Phi_1)\ \wedge\ \tau_C(s,\Phi_2)\\
\tau_C(s,\neg\Phi) & = & \neg\ \tau_C(s,\Phi)\\
\tau_C(s,\langle\!\langle A\rangle\!\rangle\Phi) & = & \exists_{\substack{t\in S_A\\ u\in Succ(t)}} Y_{t\to u}.\ \big[\ \tau_{C\cup A}(s,\Phi)\ \wedge \bigwedge_{t\in S_A,u\in Succ(t)} (0\le Y_{t\to u}\le 1)\ \wedge\ \bigwedge_{t\in S_A}\big(\sum_{u\in Succ(t)} Y_{t\to u}=1\big)\ \big]\\
\tau_C(s,\mathcal{P}_{\trianglelefteq\lambda}(\mathcal{A}(\Phi_1,\ldots,\Phi_k))) & = & \exists_{\substack{t\in S\\ 1\le i\le k}} X_{t,\Phi_i}\ \exists_{\substack{t\in S\\ q\in Q}} AEC_{t,q}\ \exists_{\substack{q,p\in Q\\ t\in S}} \Delta(q,t,p).\\
 & & \Big[\bigwedge_{t\in S,1\le i\le k}\big[\,X_{t,\Phi_i}=1\leftrightarrow\tau_C(t,\Phi_i)\,\big]\ \wedge \bigwedge_{t\in S,q\in Q}\big[\,AEC_{t,q}=1\leftrightarrow f^{\mathcal{A}_{Rabin}}_{AEC,C}(t,q)\,\big]\ \wedge\\
 & & \bigwedge_{q,p\in Q,t\in S}\big[\,\Delta(q,t,p)\in\{0,1\}\wedge\big(\,\Delta(q,t,p)=1\leftrightarrow \bigvee_{\substack{I\subseteq\{1,\ldots,k\}\\ \delta(q,I)=p}}(\wedge_{i\in I}X_{t,\Phi_i}=1\wedge\wedge_{i\notin I}X_{t,\Phi_i}\neq 1)\,\big)\big]\\
 & & \exists_{\substack{t\in S\\ q\in Q}} Z_{t,q}.\Big[\,(solution_C(\bar Z)\wedge\forall_{\substack{t\in S\\ q\in Q}} Z'_{t,q}.\big[\,solution_C(\bar Z')\to\bigwedge_{t\in S,q\in Q} Z_{t,q}\le Z'_{t,q}\,\big]\wedge Z_{s,q_{init}}\trianglelefteq\lambda\,\Big]\Big]
\end{array}
$$

$$
\begin{array}{lcll}
solution_C(\bar Z) & = & \bigwedge_{t\in S,q\in Q}\big[\,0\le Z_{t,q}\le 1\wedge(AEC_{t,q}=1\to Z_{t,q}=1)\,\big]\ \wedge & \text{(i)}\\
 & & \bigwedge_{\substack{t\in S\setminus S_C,u\in Succ(t)\\ q,p\in Q}}\big[\,(AEC_{t,q}\neq 1\wedge\Delta(q,t,p)=1)\ \to\ Z_{t,q}\ge Z_{u,p}\,\big]\wedge & \text{(ii)}\\
 & & \bigwedge_{t\in S_C,q\in Q}\big[\,Z_{t,q}=\sum_{u\in Succ(t),p\in Q} Y_{t\to u}\cdot\Delta(q,t,p)\cdot Z_{u,p}\,\big] & \text{(iii)}
\end{array}
$$

$$
\begin{array}{lcll}
f^{\mathcal{A}_{Rabin}}_{AEC,C}(s_0,p_0) & = & \exists_{\substack{t\in S\\ p\in Q}} W_{t,p}\ \exists_{\substack{t,u\in S,\ q,p\in Q\\ 0\le n\le|S|\cdot|Q|}} R^{\le n}_{<t,p><u,q>}. & \\
 & & \Big[W_{s_0,p_0}=1\ \wedge \bigwedge_{\substack{p,q\in Q\\ t\in S_C,u\in Succ(t)}}\big[\,(W_{t,p}=1\ \wedge\ Y_{t\to u}>0\ \wedge\ \Delta(p,t,q)=1)\to W_{u,q}=1\,\big]\ \wedge & \text{(I)}\\
 & & \bigwedge_{t,u\in S,p,q\in Q}\big[\,(W_{t,p}=1\wedge W_{u,q}=1)\to R^{\le|S|\cdot|Q|}_{<t,p><u,q>}=1\,\big]\ \wedge & \text{(II)}\\
 & & \bigvee_{1\le i\le m}\big[\,(\bigvee_{\substack{p\in L_i\\ t\in S}} W_{t,p}=1)\ \wedge \bigwedge_{\substack{p\in Q\setminus R_i\\ t\in S}} W_{t,p}=0\,\big]\ \wedge & \text{(III)}\\
 & & \bigwedge_{\substack{t\in S\setminus S_C,u\in Succ(t)\\ p,q\in Q}}(R^{\le 1}_{<t,p><u,q>}=1\leftrightarrow\Delta(p,t,q)=1)\wedge\bigwedge_{\substack{t\in S_C,u\in Succ(t)\\ p,q\in Q}}(R^{\le 1}_{<t,p><u,q>}=1\leftrightarrow\Delta(p,t,q)=1\ \wedge\ Y_{t\to u}>0)\ \wedge & \\
 & & \bigwedge_{t,u\in S,p,q\in Q}\Big[\bigwedge_{0\le n<|S|\cdot|Q|} R^{\le n+1}_{<t,p><u,q>}=1\leftrightarrow[R^{\le n}_{<t,p><u,q>}=1\ \vee & \\
 & & \bigvee_{\substack{v\in S\setminus S_C,u\in Succ(v)\\ r\in Q}}(W_{v,r}=1\ \wedge\ R^{\le n}_{<t,p><v,r>}=1\ \wedge\ \Delta(r,v,q)=1)\ \vee & \\
 & & \bigvee_{\substack{v\in S_C,u\in Succ(v)\\ r\in Q}}(W_{v,r}=1\ \wedge\ R^{\le n}_{<t,p><v,r>}=1\ \wedge\ \Delta(r,v,q)=1\ \wedge\ Y_{v\to u}>0)]\Big]\Big] &
\end{array}
$$

Figure 2. Inductive definition of the formula $\tau_C(s,\Phi)$.

- $(s,p)\to_{\mathcal{M}^\alpha\times\mathcal{A}}(t,q)$ iff $s\to^\alpha t$ and $\delta(p,\bar s^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;MR})=q$[6]
- $\mathcal{P}^{\mathcal{M}^\alpha\times\mathcal{A}}((s,p),(t,q))=\alpha(s)(t)$

We view $\mathcal{M}^\alpha\times\mathcal{A}$ as a Markov decision process (MDP). The states (s,p) where $s\in S_A$ are purely probabilistic and all other states are purely nondeterministic. Let AEC denote the set of states of $\mathcal{M}^\alpha\times\mathcal{A}$ that are contained in an accepting end component with the acceptance condition being a Rabin condition with the pairs $(S\times L_i,S\times R_i)_{i=1}^m$. Here, an end component in $\mathcal{M}^\gamma\times\mathcal{A}$ is a set of states $U\subseteq S\times Q$ such that

(I) for every $(s,p)\in U$ such that $s\in S_C$ and every $t\in S$ such that $\gamma(s)(t)>0$ we have that $(t,\delta(p,L(s)))\in U$. This means that U is closed under the probabilistic choice imposed by the strategy γ.

(II) U is strongly connected, i.e., for each pair of states of U we have that the first state is reachable from the second state by a finite path leading only through the states of U.

An end component is accepting if it satisfies the given acceptance condition.

It is well known [4, 7] that $s,A,\alpha\models^{\mathcal{M}}_{MR}\Phi$ iff $z_{(s,q_{init})}\trianglelefteq\lambda$, where the $(z_{(s,q)})_{s\in S,q\in Q}$ form the least solution of the following system of inequalities.

$$z_{(s,q)}=1 \text{ for all } (s,q)\in AEC \qquad \text{(i)}$$

$$z_{(s,q)}\ge z_{(t,p)} \text{ if } s\notin S_A \text{ and } (t,p)\in Succ((s,q)) \qquad \text{(ii)}$$

6 Remember that $\bar s^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;MR}=\{j\mid s,A,\alpha\models^{\mathcal{M}}_{MR}\Phi_j\}$

$$z_{(s,q)} = \sum_{(t,p)\in Succ((s,q))} \alpha(s)(t)\cdot z_{(t,p)} \text{ if } s \in S_A \qquad \text{(iii)}$$

Our encoding into $(\mathbb{R},*,+,\leq)$ simulates the above method. Let us have a closer look at the formula $\tau_C(s,\mathcal{P}_{\trianglelefteq\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k))$. First, we existentially quantify over the variables $X_{t,\Phi_i}, t \in S, 1 \leq i \leq k$. These variables indicate whether $t,C,\gamma \models^{\mathcal{M}}_{MR} \Phi_i$, where γ is represented by the variables $Y_{t\to u}, t \in S_C, u \in Succ(t)$. The subformula $(X_{t,\Phi_i} = 1 \leftrightarrow \tau_C(t,\Phi_i))$ serves for this purpose. Then we existentially quantify over the variables $AEC_{t,q}, t \in S, q \in Q$, indicating whether (t,q) is contained in an accepting end component. Last but not least we existentially quantify over the variables $\Delta(q,t,p)$ encoding the transition relation in $\mathcal{A}$. If the automaton $\mathcal{A}$ moves from the state q to the state p when reading $\vec{t}^{\Phi_1,\ldots,\Phi_k}_{C,\gamma}$, then $\Delta(q,t,p) = 1$; otherwise, $\Delta(q,t,p) = 0$ (Again, γ is represented by the variables $Y_{t\to u}, t \in S_C, u \in Succ(t)$.) Having set the conditions for $X_{t,\Phi_i}, AEC_{t,q}$, and $\Delta(q,t,p)$, we existentially quantify over the variables $Z_{t,q}, t \in S, q \in Q$ requiring that they not only form a solution of the system of inequalities, but also form the least solution. At last, the formula $\tau_C(s,\mathcal{P}_{\trianglelefteq\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k))$ requires that $Z_{s,q_{init}} \trianglelefteq \lambda$.

The formula $f^{Rabin}_{AEC,C}(s_0,p_0)$ evaluates to true iff the state (s_0,q_0) in $\mathcal{M}^\gamma \times \mathcal{A}$ is contained in some end component that satisfies the acceptance condition of the Rabin automaton $\mathcal{A}$. We quantify over the variables $W_{t,p}, t \in S, p \in Q$ that serve to indicate whether (t,p) is contained in some accepting end component and the variables $R^{\leq n}_{<t,p>,<u,q>}$ that indicate whether the state (u,q) is reachable from the state (t,p) within the end component represented by the choice of $W_{t,p}, t \in S, p \in Q$ in at most n steps. In Figure 2, formula (III) serves to ensure that the end component satisfies the Rabin acceptance condition.

The formula $\tau_C(s,\Phi)$, where Φ is an application of the probability operator $\mathcal{P}_{\bowtie\lambda}(\ldots)$ has only been defined for upper probability bounds, i.e., $\bowtie \in \{<,\leq\}$. For the definition of $\tau_C(s,\mathcal{P}_{\trianglerighteq\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$ where $\trianglerighteq \in \{>,\geq\}$ we use the duality

$$s,A,\alpha \models^{\mathcal{M}}_{MR} \mathcal{P}_{\trianglerighteq\lambda}(\mathcal{A};\bar{\Phi}) \longleftrightarrow s,A,\alpha \models^{\mathcal{M}}_{MR} \mathcal{P}_{\trianglelefteq 1-\lambda}(\bar{\mathcal{A}};\bar{\Phi}),$$

where $(\trianglerighteq,\trianglelefteq) \in \{(>,<),(\geq,\leq)\}$, $\bar{\mathcal{A}}$ is the complement automaton of $\mathcal{A}$, and $\bar{\Phi}$ abbreviates $\Phi_1,\ldots,\Phi_k$. In our case, $\mathcal{A}$ is a deterministic Rabin automaton $(Q,2^{\{1,\ldots,k\}},q_{init},\delta,(L_i,R_i)^m_{i=1})$, therefore the deterministic Streett automaton $\mathcal{A}_{Streett} = (Q,2^{\{1,\ldots,k\}},q_{init},\delta,(L_i,R_i)^m_{i=1})$ accepts the complement language of $\mathcal{A}$. Hence, the formula $\mathcal{P}_{\trianglerighteq\lambda}(\ldots)$ is alike to the formula $\mathcal{P}_{\trianglelefteq\lambda}(\ldots)$ except for the subformula

$$AEC_{t,q} = 1 \leftrightarrow f^{\mathcal{A}_{Streett}}_{AEC,C}(t,q).$$

and the threshold condition

$$Z_{s,q_{init}} \trianglelefteq 1-\lambda.$$

Furthermore, $f^{\mathcal{A}_{Streett}}_{AEC,C}(t,q)$ is a variant of the formula $f^{\mathcal{A}_{Rabin}}_{AEC,C}(t,q)$ where only the acceptance condition (III) is changed to the Streett acceptance

$$\bigwedge_{1\leq i\leq m} \Big[\big(\bigvee_{\substack{p\in L_i\\ t\in S}} W_{t,p} = 1 \big) \rightarrow \big(\bigvee_{\substack{p\in R_i\\ t\in S}} W_{t,p} = 1 \big) \Big] .$$

The following lemma states the correctness of our construction:

Lemma 4.2 (Correctness)**.** *Let $A \subseteq$ Agents and let α be a MR-strategy for A. Then $s,A,\alpha \models^{\mathcal{M}}_{MR} \Phi$ iff $\tau_A(s,\Phi)$ evaluates to true when the variables $Y_{s,t}, s \in S_A, t \in Succ(s)$ are interpreted by $\alpha(s)(t)$. In particular, $s \in Sat^{\mathcal{M}}_{MR}(\Phi)$ iff $\tau_\emptyset(s,\Phi)$ is valid.*

Proof. By induction on the structure of Φ using the previous explanations. □

A direct corollary to Lemma 4.2 is the following:

Corollary 4.3. *The* SGL *model checking problem with the MR-semantics is in* EXPSPACE*.*

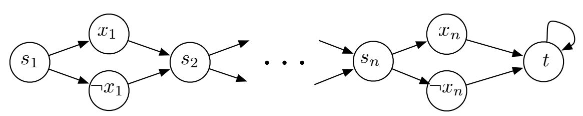
Proof. Although the quantifier alternation depth in $\tau_\emptyset(s,\Phi)$ is not bounded by a fixed constant, its size is still polynomial in the size of Φ. Hence, we can apply the result of [3] which says that $(\mathbb{R},*,+,\leq)$ is decidable in exponential space. □

Proposition 4.4. *The model checking problem of* SGL *is* PSPACE*-hard for memoryless strategies.*

Proof. We show the PSPACE-hardness by reducing QSAT, the satisfaction problem for fully quantified Boolean formulae (QBF) to our model checking problem. Let

$$\varphi = \exists x_1 \forall x_2 \exists x_3 \ldots \exists x_n \varphi_{CNF}$$

be a QBF in prenex normal form, where φ_{CNF} is a propositional formula over the variables $x_1,\ldots,x_n$ in conjunctive normal form, i.e,, $\varphi_{CNF} = c_1 \wedge \ldots \wedge c_k$ with c_j being a disjunction of literals in $\{x_1,\ldots,x_n,\neg x_1,\ldots,\neg x_n\}$. We define a PMG $\mathcal{M}$ as follows.



The set of agents is $\{1,2,\ldots,n,n{+}1\}$ and each agent $i \leq n$ controls exactly the state s_i. The agent $n+1$ controls the deterministic states $t, x_i, \neg x_i$. The set of atomic propositions is $\mathsf{AP} = \{x_1,\ldots,x_n,\neg x_1,\ldots,\neg x_n\}$ and each literal state is labelled with the corresponding atomic proposition. Let

$$\Phi = \langle\langle 1\rangle\rangle \| 2 \| \ldots \langle\langle n\rangle\rangle \left[\bigwedge_{\substack{i=1\\ i\ odd}}^{n} (\mathcal{P}_{\geq 1}(\Diamond x_i) \vee \mathcal{P}_{\geq 1}(\Diamond \neg x_i)) \wedge \bigwedge_{j=1}^{k} \mathcal{P}_{>0}(\Diamond c_j)\right]$$

It holds that φ is valid iff $s_1 \in Sat^{\mathcal{M}}_{XY}(\Phi)$ for any strategy type XY in $\{MD, MR\}$. Here $\Diamond$ denotes the "eventually" operator and $\|.\|$ stands for $\neg\langle\langle.\rangle\rangle\neg$ $\square$

The *qualitative fragment* of SGL is obtained by restricting the syntax of SGL such that the $\mathcal{P}_{\bowtie\lambda}(.)$ operator is only used with the probability bounds > 0 or ≥ 1.

Proposition 4.5. *The model checking problem for the qualitative fragment of* SGL *is* PSPACE-*complete for the MR and MD strategy semantics.*

Proof. PSPACE-hardness follows from the proof of Proposition 4.4. The membership to PSPACE follows from a closer analysis of Algorithm 1 that solves the model checking problem for the qualitative fragment of SGL and MR semantics. Note that for the qualitative fragment of SGL, the exact probabilities in the PMG $\mathcal{M}$ do not matter (as we are only dealing with finite systems). Just the topology of the underlying graph (a transition is taken with probability zero or with positive probability) is of importance. Thus, given an MR-strategy α for an agent set A, it is only important which transitions are chosen with a positive probability by α. Therefore, we declare two MR-strategies α and α' for the agent set A as equivalent if and only if the following holds.

$$\alpha(s)(t) \neq 0 \quad \leftrightarrow \quad \alpha'(s)(t) \neq 0 \qquad \forall s \in S_A, t \in S$$

The corresponding equivalence classes are called *symbolic* MR-strategies. There are exponentially many symbolic MR-strategies for A (in the size of $\mathcal{M}$), denoted MR^A_{symb}. Hence, when the formula Φ is an application of the $\langle\langle.\rangle\rangle$ operator, the **FOR** loop in the algorithm might loop exponentially often. However, the recursion depth of the algorithm is bounded by the length of the formula on input. Having computed the T_i's, the computation of $Sat^{\mathcal{M}}(\mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k),A,\alpha)$ can be done by simple graph algorithms on the product $\mathcal{M}^{\alpha} \times \mathcal{A}$ (see [8, 17, 7]). Thus, we obtain the PSPACE upper bound for the MR semantics. Note that Algorithm 1 also works for MD-strategies. A symbolic MD-strategy contains exactly one MD-strategy, so in this case the **FOR** loop ranges over all MD-strategies for B. $\square$

Proposition 4.6. *The* SGL *model checking problem is* PSPACE *complete for the MD semantics.*

Proof. PSPACE-hardness was shown in Proposition 4.4. The membership to PSPACE is obtained by a slight modification of Algorithm 1. As we are dealing with MD-strategies, let the **FOR** loop (in the case of the $\langle\langle.\rangle\rangle$ operator) range over all MD-strategies. In the case when $\Phi = \mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$ we use the following procedure.

Algorithm 1 $Sat^{\mathcal{M}}(\Phi,A,\alpha)$

Input: PMG $\mathcal{M}$, qualitative SGL formula Φ, set of agents A, α a symbolic MR strategy for A
Output: set of all $s \in S$ such that $s,A,\alpha \models^{\mathcal{M}}_{MR} \Phi$

```
CASE $\Phi$ is
    p                  :  return $\{s \in S : p \in L(s)\}$;
    $\Phi_1 \wedge \Phi_2$            :  return $Sat^{\mathcal{M}}(\Phi_1,A,\alpha) \cap Sat^{\mathcal{M}}(\Phi_2,A,\alpha)$;
    $\neg\Phi'$                :  return $S \setminus Sat^{\mathcal{M}}(\Phi',A,\alpha)$;
    $\langle\langle B\rangle\rangle\Phi'$             :  $T := \emptyset$
                       :  FOR ALL $\beta \in MR^B_{symb}$ DO
                       :      $T := T \cup Sat^{\mathcal{M}}(\Phi',A \cup B,\alpha \leftarrow \beta)$
                       :  OD
                       :  return $T$
    $\mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$   :     (* $\bowtie\lambda \in \{>0, =1\}$ *)
                  :  FOR $i = 1$ TO $k$ DO
                  :      $T_i := Sat^{\mathcal{M}}(\Phi_i,A,\alpha)$
                  :  OD
                  :  apply graph algorithm to compute the set
                  :  $T = \{s \mid s,A,\alpha \models^{\mathcal{M}}_{MR} \mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)\}$
                  :  return $T$
END CASE
```

```
$\mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$  :
        :  FOR $i = 1$ TO $k$ DO
        :      $T_i := Sat^{\mathcal{M}}(\Phi_i,A,\alpha)$
        :  OD
        :  solve linear programming problem to compute
           the set $T$ of states $s$ such that
           $s,A,\alpha \models^{\mathcal{M}}_{MD} \mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k)$
        :  return $T$
```

The linear programming problem consists of the system of inequalities (i), (ii) and (iii) described earlier in this section[7]. Gaining the least (resp. the greatest) solution of the system of inequalities is achieved by using an appropriate objective function. Having computed the T_i's, $Sat^{\mathcal{M}}(\mathcal{P}_{\bowtie\lambda}(\mathcal{A};\Phi_1,\ldots,\Phi_k),A,\alpha)$ is computable in time polynomial in the size of $\mathcal{M}$ and $\mathcal{A}$. The PSPACE membership follows as the recursion depth of the algorithm is bounded by the length of the formula on input. $\square$

5. Conclusion

We introduced a new stochastic game logic (SGL) interpreted over probabilistic multiplayer games (PMG). It combines features of alternating time logic (ATL), probabilistic computation tree logic and extended temporal logics. Our

7 Of course, only if $\bowtie \in \{<, \leq\}$. For $\bowtie \in \{>, \geq\}$ everything can easily be adapted.

logic uses an existential strategy quantifier $\langle\langle . \rangle\rangle$ that, unlike in ATL, propagates the chosen strategies to the subformulae. This enables us to state game properties like *"player B can react to the strategy chosen by player A"*. Whereas the ATL model checking problem is known to be solvable in PTIME [1], modifying the semantics of the $\langle\langle . \rangle\rangle$ operator so that the strategy decisions are propagated to the subformulae makes the model checking problem PSPACE-hard. In this paper we established the following results.

The model-checking problem for finite state PMG and general SGL is

- undecidable for HR and HD strategies,
- PSPACE-complete for MD strategies,
- PSPACE-hard and in EXPSPACE for MR strategies.

The model-checking problem for finite state MPG and the qualitative fragment of SGL is

- PSPACE-complete for MD and MR strategies

The decidability of the qualitative fragment of SGL with respect to history dependent strategies remains open.

References

[1] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49:672–713, 2002.

[2] C. Baier, M. Größer, M. Leucker, F. Ciesinski, and B. Bollig. Controller synthesis for probabilistic systems. In *IFIP Worldcongress, Theoretical Computer Science*, 2004.

[3] Michael Ben-Or, Dexter Kozen, and John Reif. The complexity of elementary algebra and geometry. *J. Comput. Syst. Sci.*, 32(2):251–264, 1986.

[4] Bianco and de Alfaro. Model checking of probabilistic and nondeterministic systems. In *Proc. FSTTCS*, volume 15 of *Lecture Notes in Computer Science*, pages 499–513, 1995.

[5] Tomáš Brázdil, Václav Brožek, Vojtěch Forejt, and Antonín Kučera. Stochastic games with branching-time winning objectives. In *Proc. LICS*. IEEE Computer Society Press (to appear), 2006.

[6] Edmund M. Clarke, Orna Grumberg, and Robert P. Kurshan. A synthesis of two approaches for verifying finite state concurrent systems. *J. Log. Comput.*, 2(5):605–618, 1992.

[7] Costas Courcoubetis and Mihalis Yannakakis. The complexity of probabilistic verification. *Journal of the ACM*, 42(4):857–907, July 1995.

[8] Luca de Alfaro. Formal verification of probabilistic systems. Thesis CS-TR-98-1601, Stanford University, Department of Computer Science, June 1998.

[9] Luca de Alfaro and Thomas A. Henzinger. Concurrent omega-regular games. In *Proc. LICS*, pages 141–154, 2000.

[10] Luca de Alfaro, Thomas A. Henzinger, and Orna Kupferman. Concurrent reachability games. In *IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 564–575, 1998.

[11] Sergiu Hart, Micha Sharir, and Amir Pnueli. Termination of probabilistic concurrent programs. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 5(3):356–380, July 1983.

[12] Antoní Kučera and Oldřich Stražovský. On the controller synthesis for finite-state Markov decision processes. In *Proc. FSTTCS*, volume 3821 of *Lecture Notes in Computer Science*, pages 541–552, 2005.

[13] Shmuel Safra. On the complexity of ω-automata. In *29th Annual Symposium on Foundations of Computer Science*, pages 319–327, White Plains, New York, 24–26 October 1988. IEEE.

[14] R. Segala and N. Lynch. Probabilistic simulations for probabilistic processes. *Lecture Notes in Computer Science*, 836:481–496, 1994.

[15] Wolfgang Thomas. Computation tree logic and regular omega-languages. *Lecture Notes in Computer Science*, 354:690–713, 1988.

[16] Wolfgang Thomas. Automata on infinite objects. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, chapter 4, pages 133–191. Elsevier Science Publishers B. V., 1990.

[17] Moshe Y. Vardi. Probabilistic linear-time model checking: An overview of the automata-theoretic approach. *Lecture Notes in Computer Science*, 1601:265–276, 1999.

[18] Moshe Y. Vardi and Pierre Wolper. Yet another process logic (preliminary version). *Lecture Notes in Computer Science*, 164:501–512, 1983.