Acta Informatica (2012) 49:203–224
DOI 10.1007/s00236-012-0156-0

ORIGINAL ARTICLE

# Stochastic game logic

**Christel Baier · Tomáš Brázdil · Marcus Größer · Antonín Kučera**

Received: 6 September 2011 / Accepted: 30 April 2012 / Published online: 6 June 2012


**Abstract** Stochastic game logic (SGL) is a new temporal logic for multi-agent systems modeled by turn-based multi-player games with discrete transition probabilities. It combines features of alternating-time temporal logic (ATL), probabilistic computation tree logic and extended temporal logic. SGL contains an ATL-like modality to specify the individual cooperation and reaction facilities of agents in the multi-player game to enforce a certain winning objective. While the standard ATL modality states the existence of a strategy for a certain coalition of agents without restricting the range of strategies for the semantics of inner SGL formulae, we deal with a more general modality. It also requires the existence of a strategy for some coalition, but imposes some kind of strategy binding to inner SGL formulae. This paper presents the syntax and semantics of SGL and discusses its model checking problem for different types of strategies. The model checking problem of SGL turns out to be undecidable when dealing with the full class of history-dependent strategies. We show that the SGL model checking problem for memoryless deterministic strategies as well as the model checking problem of the qualitative fragment of SGL for memoryless randomized strategies is PSPACE-complete, and we establish a close link between natural syntactic fragments of SGL and the polynomial hierarchy. Further, we give a reduction from the SGL model checking problem under memoryless randomized strategies into the Tarski algebra which proves the problem to be in EXPSPACE.

C. Baier · M. Größer
Faculty for Computer Science, Institute for Theoretical Computer Science,
Technische Universität Dresden, 01062 Dresden, Germany
e-mail: baier@tcs.inf.tu-dresden.de

M. Größer
e-mail: groesser@tcs.inf.tu-dresden.de

T. Brázdil · A. Kučera (✉)
Faculty of Informatics, Masaryk University, Botanická 68a,
60200 Brno, Czech Republic
e-mail: kucera@fi.muni.cz

T. Brázdil
e-mail: brazdil@fi.muni.cz

## 1 Introduction

Traditional temporal logics, such as linear temporal logic (LTL) or computation tree logic (CTL) are widely used to specify properties of parallel systems [14]. In the classical approach, the semantics of LTL and CTL relies on an operational model (transition system) where the paths represent the possible interleavings of the processes running in parallel and formulae assert conditions on all paths (LTL) or on the branching structure of states (CTL). This approach is adequate for closed systems where the transition system describes the potential system behaviors from a global perspective. In contrast, the semantics of open systems relies on a *game-based view* where the individual components are considered as *players* (also called *agents*) that interact with each other and their environment. Alur et al. [2] introduced alternating-time temporal logic (ATL) as a variant of CTL with modalities expressing the existence of strategies for coalitions of agents that enforce a certain event. More precisely, ATL extends CTL by formulae of the form $\langle\langle A\rangle\rangle_{\text{ATL}}\varphi$ and $[\![A]\!]_{\text{ATL}}\varphi$ where $A$ is a set of cooperating players (agents) and $\varphi$ an ATL path formula. [1] Intuitively, the formula $\langle\langle A\rangle\rangle_{\text{ATL}}\varphi$ asserts that there is a strategy for the agents in $A$ such that the event specified by $\varphi$ holds, no matter how the opponents (i.e., the other agents) behave. The dual operator, denoted by $[\![\cdot]\!]_{\text{ATL}}$, can be understood as universal quantification over strategies. That is, formula $[\![A]\!]_{\text{ATL}}\varphi$ states that $\varphi$ holds along at least one path under all strategies for $A$. Stated differently, $[\![A]\!]_{\text{ATL}}\varphi$ asserts the absence of a strategy for the coalition $A$ to avoid $\varphi$ to hold. The ATL semantics relies on the standard CTL-like approach where all subformulae are interpreted over the "full" structure. That is, the ATL modalities $\langle\langle\cdot\rangle\rangle_{\text{ATL}}$ and $[\![\cdot]\!]_{\text{ATL}}$ do not restrict the range of strategies for nested $\langle\langle\cdot\rangle\rangle_{\text{ATL}}$ and $[\![\cdot]\!]_{\text{ATL}}$ modalities. For instance, the formula $\langle\langle A\rangle\rangle_{\text{ATL}}\Box\langle\langle B\rangle\rangle_{\text{ATL}}\Diamond\mathsf{p}$ asserts the existence of a strategy $\alpha$ for the agents in $A$ such that $\langle\langle B\rangle\rangle_{\text{ATL}}\Diamond\mathsf{p}$ holds (in the "full" game) for all states $s$ that can be reached when the agents in $A$ make their decisions according to $\alpha$, i.e., from these states $s$ the agents in $B$ have a strategy $\beta$ in the original game (neglecting the strategy $\alpha$) which ensures that a state where $\mathsf{p}$ holds is reached. [2] Thus in ATL a strategy chosen by the $\langle\langle\cdot\rangle\rangle_{\text{ATL}}$ 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 are not expressible in ATL. Several variants of ATL have been proposed that contain modalities for strategy quantification imposing certain bindings of strategies to inner formulae. Examples are game logic [2], ATL with strategy contexts [1,9], or strategy logic of [12] that contains first-order quantification over strategies.

In this paper, we introduce a probabilistic variant of ATL, called SGL for short. It serves to specify properties of multi-agent systems where actions can have a probabilistic effect. More precisely, we deal with finite-state turn-based games where the game arena is based on a graph structure with several annotations. Each state is either a game configuration where a single agent is declared to choose a successor state (based on some deterministic or randomized strategy) or a probabilistic state where the next state is chosen randomly according to some fixed probabilistic distribution. The logic SGL contains a variant of the ATL-modality for reasoning about the existence of strategies for a coalition of agents $A$, denoted by $\langle\langle A\rangle\rangle$, to achieve a certain objective. The dual modality to $\langle\langle A\rangle\rangle$ is denoted by $[\![A]\!]$ in SGL. The difference between the ATL-modalities $\langle\langle A\rangle\rangle_{\text{ATL}}$ and $[\![A]\!]_{\text{ATL}}$ and the SGL modalities $\langle\langle A\rangle\rangle$ and $[\![A]\!]$ becomes apparent when these operators are *nested*. Following the approach of [4,8,22],

[1] Our logic SGL uses other modalities for existential and universal quantification over strategies. To avoid notation overloading, we use the notation $\langle\langle A\rangle\rangle_{\text{ATL}}$ and $[\![A]\!]_{\text{ATL}}$ to denote the standard ATL modalities, while $\langle\langle A\rangle\rangle$ and $[\![A]\!]$ will be used for the SGL-modalities. Later on, we will explain how the ATL-modalities can be derived from $\langle\langle A\rangle\rangle$ and $[\![A]\!]$.

[2] $\Box$ and $\Diamond$ denote the "always" and "eventually" operator, respectively.

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 $\alpha$ 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 $\Phi$ 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 objectives of coalitions of agents can be linear-time or branching-time properties with qualitative or quantitative probability bounds. For this purpose, the SGL syntax combines features of probabilistic computation tree logic (PCTL) for Markov decision processes [6] with features of extended temporal logic (ECTL*) [13,27,30]. More precisely, the SGL-operators $\langle\langle\cdot\rangle\rangle$ and $\|\cdot\|$ can be used in combination with qualitative or quantitative probability bounds on path-events that are specified by finite-state $\omega$-automata. For technical convenience, we use deterministic Rabin automata to describe path properties.

With this concept we can formalize typical multi-player 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

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

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.

To illustrate the advantages of strategy binding and revision, consider the following scenario. A broker (coalition $B$) has a certain amount of money (say 1 Mio Dollars) to work with. The broker's 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 %) the earnings to become larger than 100.000 Dollars in the next year. Further, the broker must act so that his potential losses caused by unpredictable events (e.g., earthquake or oil embargo) are acceptable. That is, if some unpredictable event happens, the broker should be able to change his behaviour (depending on the actual event) so that he has at least 500.000 Dollars at his/her disposal within a day, no matter what happens to the rest of the money. Let us assume that the unpredictable events are fired by a coalition $E$ of players. Then the appropriate SGL formula formalizing the above requirements is

$$\langle\langle B\rangle\rangle(Earn \wedge Safe)$$

where

$$\begin{aligned} Earn &\equiv \mathscr{P}_{\geq 0.9}\left(\Diamond^{\leq 365}(earnings \geq 100.000)\right) \\ Safe &\equiv \mathscr{P}_{\geq 1}\left(\Box\|E\|\langle\langle B\rangle\rangle\mathscr{P}_{\geq 1}(\mathscr{X}(available\ money \geq 500.000))\right) \end{aligned}$$

Here, $\Box$ denotes the "always" operator, $\Diamond^{\leq 365}$ denotes the "in at most 365 steps" operator, and $\mathscr{X}$ represents the next operator (see Sect. 3). One step corresponds to one day.

*Our contribution* The novel logic SGL provides a uniform framework for reasoning about qualitative and quantitative linear- and branching-time properties of probabilistic multi-agent systems. We present the syntax and semantics of SGL and study the decidability and complexity of the SGL model checking problem for various types of strategies. As a consequence of known results for stochastic games with branching-time winning objectives [8,22], we obtain the undecidability of the SGL model checking problem for history-dependent strategies. However, decidability can be established when restricting the range of the strategies for

the modalities $\langle\langle A \rangle\rangle$ and $\|A\|$ to memoryless ones. We present a classification of SGL formulae into "types" that yield a perfect match with the polynomial hierarchy when dealing with memoryless strategies only. This also yields PSPACE-completeness of the SGL model checking problem under memoryless deterministic strategies and the model checking problem for the qualitative fragment of SGL under memoryless randomized strategies. Using an encoding of the semantics for (arbitrary) SGL formulae in first-order theory of the reals, we obtain an exponentially space-bounded model checking algorithm for full SGL and memoryless randomized strategies.

To the best of our knowledge, this paper (and its preceding conference version [3]) represents the first attempt for defining an ATL-like logic that can express quantitative (PCTL-like) properties combined with strategy binding. Former approaches with ATL-like modalities for reasoning about concurrent stochastic games have been studied by de Alfaro et al. [18,19] in, e.g. However, these papers concentrate on qualitative properties and they do not consider Boolean combination of qualitative properties or the nesting of $\langle\langle\cdot\rangle\rangle$ operators. A probabilistic variant of ATL, denoted by pATL, has been studied in [10]. The logic pATL contains a modality $\langle\cdot\rangle^p$ that can express the existence of a strategy for the agents in $A$ that achieves the satisfaction probability at least $p$. However, there is no strategy binding in the logic pATL.

The reduction of the SGL model checking problem for memoryless randomized strategies to Tarski algebra reuses ideas that have been presented in [22] for the synthesis of controller from PCTL specifications. However, several non-trivial adaptions to our more general logical framework with potential nestings of ATL-like modalities are required.

*Organization* Section 2 introduces our model of probabilistic multi-player games (PMG) and related notions. The syntax and semantics of SGL is introduced in Sect. 3. The model checking problem for SGL on multi-player games is addressed in Sects. 4, and 5 concludes the paper.

## 2 Preliminaries

In this section we briefly explain our model of probabilistic multi-player games (Sect. 2.1) and summarize the relevant features of deterministic Rabin automata (Sect. 2.2).

### 2.1 Probabilistic multi-player games

In this paper, we deal with turn-based multi-player games where in each state only one agent makes a move.

**Definition 1** *Probabilistic multi-player game (PMG)* A probabilistic multi-player game (PMG) is a tuple $\mathcal{M} = (\mathsf{Agents}, S, \to, \mathsf{P}, \mathsf{Props}, \nu)$ where

- $\mathsf{Agents}$ is a finite set of agents,
- $S$ is a set of states, disjointly partitioned into $S = S_{prob} \cup \bigcup_{a \in \mathsf{Agents}} S_a$,
- $\to \subseteq S \times S$ is a total transition relation,[3] i.e., for every $s \in S$ there is $t \in S$ such that $s \to t$,
- $\mathsf{P} : S_{prob} \times S \to [0, 1]$ is a probability assignment such that, for all $s \in S_{prob}$, $\sum_{u \in S} \mathsf{P}(s, u) = 1$ and $\mathsf{P}(s, t) = 0$ iff $s \not\to t$,
- $\mathsf{Props}$ is a finite set of atomic propositions,

[3] In the rest of this paper, we will write $s \to t$ instead of $(s, t) \in \to$.

– $v : S \to 2^{\mathsf{Props}}$ is a labeling function that assigns to each state $s$ the set $v(s)$ of atomic propositions which hold in $s$.

A Markov decision process (MDP) is a PMG where the set of agents is a singleton.

We may regard $S$ as a function that assigns to each agent $a$ a set $S_a$ such that $S_a \cap S_b = \varnothing$ if $a \neq b$. The states $s \in S_a$ are called $a$-states. 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. Note that in the $a$-states, it is agent's $a$ turn to choose a transition $s \to t$. In the probabilistic states $s \in S_{prob}$, the successor state is chosen randomly according to $\mathsf{P}$.

So far, no restrictions on $\mathscr{M}$ have been made. When addressing the model checking problem, we consider only finite-state PMG with rational probability assignment (i.e., $S$ is a finite set and $\mathsf{P}(s,t)$ is rational for all $(s,t) \in S_{prob} \times S$).

We write $Paths(s)$ for the set of all infinite sequences $s_0\, s_1\, s_2 \ldots \in S^\omega$ where $s_0 = s$ and $s_i \to s_{i+1}$ for all $i \geq 0$. More generally, for a finite sequence $w = s_0 \ldots s_k$ of states we use $Paths(w)$ to denote the set of all $\pi \in Paths(s_0)$ that start with $w$ (note that $Paths(w)$ can be empty). We denote by $Succ(s)$ the set of all successors of $s$, i.e., $Succ(s) = \{t \in S \mid s \to t\}$. For a path $\pi = s_0, s_1, \ldots$ and $j \geq 0$, we denote by $\pi(j)$ the state $s_j$ of $\pi$.

Given a finite or countably infinite set $T$, let $\mathsf{Distr}(T)$ be the set of all distributions on $T$, i.e., functions $\mu : T \to [0,1]$ such that $\sum_{t \in T} \mu(t) = 1$. A distribution $\mu$ is Dirac if $\mu(t) = 1$ for some $t \in T$.

**Definition 2** *Strategy* Let $A \subseteq \mathsf{Agents}$. A history-dependent randomized $A$-strategy (briefly HR strategy, or simply strategy) is a function $\alpha : S^* S_A \to \mathsf{Distr}(S)$ such that $\alpha(s_1 \ldots s_n\, s)(t) = 0$ if $s \not\to t$. An $\alpha$-path denotes a path $s_0\, s_1\, s_2 \ldots$ which is consistent with $\alpha$'s decisions, i.e., for all $i \geq 0$, $s_i \in S_A$ implies $\alpha(s_0 \ldots s_i)(s_{i+1}) > 0$.

A strategy $\alpha$ is called deterministic (or a HD strategy) if for all $s_1 \ldots s_n\, s \in S^* S_A$, the distribution $\alpha(s_1 \ldots s_n\, s)$ is Dirac. We say that $\alpha$ is memoryless (or an MR strategy) if $\alpha(s_1 \ldots s_n\, s) = \alpha(s)$ for all state-sequences $s_1 \ldots s_n$. An MD strategy means a memoryless deterministic strategy. A special type of HR strategies are finite-memory (FR) strategies, where the decision depends only on the control state entered by some fixed finite-state automaton after reading the sequence $s_1 \ldots s_n\, s \in S^* S_A$. An FD strategy is a deterministic FR strategy.

Given a strategy $\alpha$ for all agents and a state $s \in S$, we define the probability space $(Paths(s), \mathscr{F}, Prob^\alpha)$ in the standard way, i.e.,

– $\mathscr{F}$ is the $\sigma$-field generated by all $Paths(w)$ where $w$ is a finite sequence of states initiated in $s$,
– $Prob^\alpha$ is the unique probability measure such that for all $w = s_0 \ldots s_n$ where $s = s_0$ and $n \geq 1$ we have that $Prob^\alpha(Paths(w)) = \prod_{i=0}^{n-1} x_i$. Here $x_i$ is equal to $\mathsf{P}(s_i, s_{i+1})$ or $\alpha(s_0 \ldots s_i)(s_{i+1})$, depending on whether $s_i \in S_{prob}$ or not, respectively.

Sometimes we also need to consider games induced by $A$-strategies, where $A$ is just a subset of agents. In particular, this is useful for memoryless strategies. For a given MR $A$-strategy $\alpha$, the game $\mathscr{M}^\alpha$ induced by $\alpha$ arises from $\mathscr{M}$ by fixing the decisions for the agents in $A$ according to $\alpha$ (note that the $A$-states of $\mathscr{M}$ become probabilistic states in $\mathscr{M}^\alpha$). Formally, we define $\mathscr{M}^\alpha = (\mathsf{Agents} \setminus A, S, \to^\alpha, \mathsf{P}^\alpha, \mathsf{Props}, v)$ 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 $s \notin S_A$ is a probabilistic state of $\mathscr{M}$.

### 2.2 Deterministic Rabin automata

The logic SGL introduced in the next section uses $\omega$-regular languages to specify path properties in the style of the extended computation tree logic ECTL [13]. These languages are expressed by deterministic Rabin automata. We briefly recall here the basic concepts.

**Definition 3** [*Deterministic Rabin Automata (DRA)*] A deterministic Rabin automaton (DRA) $\mathscr{A}$ is a tuple $(Q, \Sigma, q_{init}, \delta, (L_i, R_i)_{i=1}^m)$, where

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

Given an infinite word $\pi = \pi_1 \pi_2 \ldots \in \Sigma^\omega$ over the alphabet $\Sigma$, we call $r(\pi) = q_1 q_2 q_3 \ldots$ where $q_1 = q_{init}$ and $q_{i+1} = \delta(q_i, \pi_i)$ the run of $\mathscr{A}$ for the input word $\pi$. 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 $j \in \{1, \ldots, m\}$ such that $T \cap L_j \neq \varnothing$ and $T \cap R_j = \varnothing$. The language accepted by the Rabin automaton $\mathscr{A}$ is defined as

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

In this paper (particularly in Sect. 4), we rely on well-known results about optimal values in maximizing and minimizing MDPs with DRA objectives (see [16,17] or Chapter 10 in [5]). To make this paper self-contained, we briefly recall these results (in a form which suits our purposes). Let $\mathscr{M} = (\{a\}, S, \to, \mathsf{P}, \mathsf{Props}, \nu)$ be an MDP and $\mathscr{A} = (Q, \Sigma, q_{init}, \delta, (L_i, R_i)_{i=1}^m)$ a DRA where $\Sigma = 2^{\mathsf{Props}}$. For every infinite path $\pi = s_1 s_2 \ldots$, let $\hat{\pi} = \nu(s_1)\, \nu(s_2) \ldots$ be the corresponding infinite word over $\Sigma$. For every $s \in S$, we define its upper and lower $\mathscr{A}$-value as follows:

$$val^+(\mathscr{A}, s) = \sup_\alpha Prob^\alpha(\{\pi \in Paths(s) \mid \hat{\pi} \in L(\mathscr{A})\}),$$
$$val^-(\mathscr{A}, s) = \inf_\alpha Prob^\alpha(\{\pi \in Paths(s) \mid \hat{\pi} \in L(\mathscr{A})\}).$$

Here $\alpha$ ranges over all HR strategies of the only agent $a$.

The next proposition says that $val^+(\mathscr{A}, s)$ and $val^-(\mathscr{A}, s)$ correspond to components of the least solution of efficiently constructible systems of linear (in)equalities.

**Proposition 1** (see [5,6,16,17]) *Let $\mathscr{M} = (\{a\}, S, \to, \mathsf{P}, \mathsf{Props}, \nu)$ be an MDP and $\mathscr{A} = (Q, \Sigma, q_{init}, \delta, (L_i, R_i)_{i=1}^m)$ a DRA where $\Sigma = 2^{\mathsf{Props}}$. There are systems $\Delta_{acc}$, $\Delta_{rej}$ of linear (in)equalities over the set of variables $Var = \{x_{s,q} \mid s \in S, q \in Q\}$, constructible in polynomial time, such that for every $s \in S$ we have that*

- $val^+(\mathscr{A}, s) = \mu[\Delta_{acc}](x_{s,q_{init}})$,
- $val^-(\mathscr{A}, s) = 1 - \mu[\Delta_{rej}](x_{s,q_{init}})$.

*Here $\mu[\Delta_{acc}]$ and $\mu[\Delta_{rej}]$ denote the least solution of $\Delta_{acc}$ and $\Delta_{rej}$ in $[0, 1]$, and $\mu[\Delta_{acc}](x_{s,q})$ and $\mu[\Delta_{rej}](x_{s,q})$ denote the value of $x_{s,q}$ in $\mu[\Delta_{acc}]$ and $\mu[\Delta_{rej}]$, respectively.*

*Proof (sketch)* First, we construct the synchronized product of $\mathscr{M}$ and $\mathscr{A}$, which is an MDP $\mathscr{M} \times \mathscr{A} = (\{a\}, S \times Q, \rightarrow_\otimes, \mathsf{P}_\otimes, \mathsf{Props}_\otimes, \nu_\otimes)$, where

- $(S \times Q)_a = S_a \times Q$, $(S \times Q)_{Prob} = S_{Prob} \times Q$,
- $(s, q) \rightarrow_\otimes (s', q')$ iff $s \rightarrow s'$ and $\delta(q, \nu(s)) = q'$,
- $\mathsf{P}_\otimes((s, q), (s', q'))$ is equal either to $\mathsf{P}(s, s')$ or 0, depending on whether $(s, q) \rightarrow_\otimes (s', q')$ or not, respectively.
- $\mathsf{Props}_\otimes = \mathsf{Props}$,
- $\nu_\otimes(s, q) = \nu(s)$.

An end component of $\mathscr{M} \times \mathscr{A}$ is a set $U \subseteq S \times Q$ such that

- for all $(s, q) \in U$ and $(s', q') \in S \times Q$ such that $s \in S_{Prob}$ and $(s, q) \rightarrow_\otimes (s', q')$ we have that $(s', q') \in U$, i.e., $U$ is closed under successors of probabilistic vertices;
- for every $(s, q) \in U$, where $s \in S_a$, there is $(s', q') \in U$ such that $(s, q) \rightarrow_\otimes (s', q')$ and $(s', q') \in U$;
- $U$ 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 $U$.

Obviously, for each end component $U$, there is a strategy $\alpha$ for the only agent $a$ such that each state of $U$ is visited infinitely often with probability one, assuming that we start in a state of $U$. For a given end component $U$, we use $Q_U$ to denote the set $\{q \in Q \mid (s, q) \in U \text{ for some } s \in S\}$. We say that a given end component $U$ is *accepting* if there is $1 \leq i \leq m$ such that $Q_U \cap L_i \neq \varnothing$ and $Q_U \cap R_i = \varnothing$. Similarly, we say that $U$ is *rejecting* if for all $1 \leq i \leq m$ we have that $Q_U \cap L_i = \varnothing$ or $Q_U \cap R_i \neq \varnothing$. The systems $\Delta_{acc}$ and $\Delta_{rej}$ are constructed as follows:

(1) If $(s, q)$ belongs to some accepting/rejecting end component, we add to $\Delta_{acc}/\Delta_{rej}$ the equality $x_{s,q} = 1$;
(2) otherwise, we add to $\Delta_{acc}/\Delta_{rej}$ either

$$\text{the equality } x_{s,q} = \sum_{(s,q) \rightarrow_\otimes (s',q')} \mathsf{P}_\otimes((s, q), (s', q')) \cdot x_{s',q'}, \text{ if } s \in S_{Prob};$$

$$\text{the inequality } x_{s,q} \geq x_{s',q'} \text{ for every transition } (s, q) \rightarrow_\otimes (s', q'), \text{ if } s \in S_a.$$

It follows from the results of [5,6,16,17] that $val^+(\mathscr{A}, s) = \mu[\Delta_{acc}](x_{s,q_{init}})$ and $val^-(\mathscr{A}, s) = 1 - \mu[\Delta_{rej}](x_{s,q_{init}})$. Further, the systems $\Delta_{acc}$ and $\Delta_{rej}$ are constructible in time polynomial in the size of $\mathscr{M}$ and $\mathscr{A}$, because the condition of (1) is solvable in polynomial time. □

## 3 The logic SGL

For specifying properties of probabilistic multi-player games, we introduce a new temporal logic called "stochastic game logic" (SGL). The logic SGL borrows ideas from ATL (and the ATL-like formalisms for stochastic games [18,19]), extended computation tree logic ECTL [13], and the game logic GL of [2]. The probabilistic fragment of SGL contains a PCTL-like probabilistic operator which allows to reason about the probabilities for $\omega$-regular properties, expressed by a deterministic Rabin automaton.

We start with the syntax of SGL (Sect. 3.1), then present its formal semantics by interpreting SGL formulae over the states of a probabilistic multi-player game (Sect. 3.2) and then discuss the expressiveness of SGL (Sect. 3.3).

### 3.1 SGL syntax

Throughout the paper, let AP and AG be countably infinite sets of atomic propositions and agents, respectively.

The abstract syntax of SGL formulae is given by the following equation:

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

Here $\mathsf{p}$ and $A$ range over AP and finite subsets of AG, respectively, $\bowtie \in \{<, \leq, >, \geq\}$ is a comparison operator, $\lambda \in [0, 1]$ is a rational probability bound, and $\mathscr{A}$ is a DRA over the alphabet $2^{\{1,\ldots,k\}}$.

Note that SGL does not contain the usual Boolean connectives such as $\wedge$, $\vee$, etc. This is no restriction, because, as we shall see in the next section, $\Phi_1 \wedge \Phi_2$ is semantically equivalent to $\mathscr{P}_{>0}(\mathscr{A}_\wedge; \Phi_1, \Phi_2)$, where $\mathscr{A}_\wedge$ is the fixed DRA of Example 1. Hence, we can freely use Boolean connectives in SGL formulae as symbolic abbreviations *without* influencing the upper complexity bounds for the SGL model checking problem presented in Section 4. The same can be said about the standard PCTL operators $\mathscr{P}_{\bowtie\lambda}(\mathscr{X}(\Phi))$ and $\mathscr{P}_{\bowtie\lambda}(\Phi_1 \mathscr{U} \Phi_2)$ which are equivalent to $\mathscr{P}_{\bowtie\lambda}(\mathscr{A}_\mathscr{X}; \Phi_1)$ and $\mathscr{P}_{\bowtie\lambda}(\mathscr{A}_\mathscr{U}; \Phi_1, \Phi_2)$, respectively (see Example 1).

The real power of DRA becomes apparent when dealing with more complicated path properties. Since DRA can accept an arbitrary $\omega$-regular language, the class of path properties expressible in SGL is exactly the class of all $\omega$-regular properties, and hence SGL is strictly more expressive than PCTL* even on purely probabilistic systems. The idea of automata connectives is borrowed from the logic ECTL* [13,27,30]. For our purposes, it is more convenient to use DRA (rather than non-deterministic Büchi automata), because this yields a better match with standard complexity classes. Alternatively, we could also use deterministic Müller or deterministic Street automata.

An important syntactic fragment of SGL is *qualitative SGL* where the constant $\lambda$ in the probabilistic operator $\mathscr{P}_{\bowtie\lambda}$ may only take the value 0 or 1.

### 3.2 SGL semantics

Let $\mathscr{M} = (\mathsf{Agents}, S, \rightarrow, \mathsf{P}, \mathsf{Props}, \nu)$ be a PMG where $\mathsf{Agents} \subseteq \mathsf{AG}$, $\mathsf{Props} \subseteq \mathsf{AP}$, and XY a class of strategies (i.e., XY is either MD, MR, HD, or HR). We define a satisfaction relation

$$s, A, \alpha \models_{\mathrm{XY}} \Phi$$

where $s$ is a state in $\mathscr{M}$, $A \subseteq \mathsf{Agents}$, $\alpha$ is an XY $A$-strategy, and $\Phi$ is an SGL formula such that for every $\langle\!\langle B \rangle\!\rangle$ operator used in $\Phi$ we have that $B \subseteq \mathsf{Agents}$. The intuitive meaning is that $s$ satisfies $\Phi$ in the game induced by $\alpha$. The $A$ and $\alpha$ on the left-hand side of $\models_{\mathrm{XY}}$ are used to keep track of the strategy decisions already made. Note that SGL is parametrized by the strategy class XY that can be used by the agents whose strategy choice

The formal rules for the satisfaction relation are given below, followed by a more detailed explanation of the newly employed symbols.

$$
\begin{array}{lll}
s, A, \alpha \models_{\mathrm{XY}} \mathsf{p} & \text{iff} & \mathsf{p} \in \nu(s) \\
s, A, \alpha \models_{\mathrm{XY}} \neg\Phi & \text{iff} & s, A, \alpha \not\models_{\mathrm{XY}} \Phi \\
s, A, \alpha \models_{\mathrm{XY}} \langle\!\langle B \rangle\!\rangle \Phi & \text{iff} & \text{there is an XY } B\text{-strategy } \beta \text{ such that} \\
 & & s, A \cup B, (\alpha \leftarrow \beta) \models_{\mathrm{XY}} \Phi \\
s, A, \alpha \models_{\mathrm{XY}} \mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Phi_1, \ldots, \Phi_k) & \text{iff} & \text{for all HR strategies } \beta \text{ for } \mathsf{Agents} \setminus A \text{ we have that} \\
 & & Prob^{\alpha \leftarrow \beta}\left(\{\pi \in Paths(s) \mid \tilde{\pi}^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;XY} \in L(\mathscr{A})\}\right) \bowtie \lambda
\end{array}
$$

The semantics of $\mathsf{p}$ and $\neg\Phi$ is standard. The formula $\langle\langle B\rangle\rangle\Phi$ requires the existence of an XY $B$-strategy $\beta$ such that the subformula $\Phi$ is satisfied in the game induced by $\alpha$ and $\beta$. For the agents in $A \cap B$, the previous decisions made by $\alpha$ are changed by $\beta$. That is, $(\alpha \leftarrow \beta)$ denotes the strategy for the agents in $A \cup B$ such that the agents in $A \setminus B$ behave according to the strategy $\alpha$ and the agents in $B$ behave according to the strategy $\beta$. Formally, given a path $\pi = s_1, \ldots, s_n$, we put

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

Finally, the $\mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Phi_1, \ldots, \Phi_k)$ formula has the standard PCTL* semantics, meaning that for all *unrestricted (i.e., HR)*, strategies $\beta$ of the "remaining" agents in $\mathsf{Agents} \setminus A$, the probability measure of all paths accepted by the automaton $\mathscr{A}$ in the Markov chain induced by combining the "current" strategy $\alpha$ with $\beta$ matches the probability bound $\lambda$. Here, a path is accepted by the automaton $\mathscr{A}$ if its projection to words over $2^{\{1,\ldots,k\}}$ indicating which of the formulae $\Phi_1, \ldots, \Phi_k$ are satisfied in each of the states of the path, is in $L(\mathscr{A})$. Formally, $\tilde{\pi}^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;XY} \in (2^{\{1,\ldots,k\}})^\omega$ is defined as follows:

$$\tilde{\pi}^{\Phi_1,\ldots,\Phi_k}_{A,\alpha;XY}(i) = \left\{ j \mid 1 \le j \le k \text{ and } \pi(i), A, \alpha \models_{XY} \Phi_j \right\}.$$

Given a formula $\Phi$, we denote by $Sat_{XY}(\Phi)$ the set of all states of $\mathscr{M}$ that satisfy $\Phi$, i.e.,

$$Sat_{XY}(\Phi) = \left\{ s \in S \mid s, \varnothing, \alpha_\varnothing \models_{XY} \Phi \right\}.$$

Note that the class of strategies for agents that explicitly appear in some $\langle\langle\cdot\rangle\rangle$ 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, because implementing memory/randomization might be costly. The results in [4] yield that the satisfaction relations $\models_{\text{HD}}$, $\models_{\text{HR}}$, $\models_{\text{MD}}$, and $\models_{\text{MR}}$ are pairwise distinct. For example, it can happen that $s, \varnothing, \alpha_\varnothing \not\models_{MD} \langle\langle B\rangle\rangle\Phi$ and $s, \varnothing, \alpha_\varnothing \models_{MR} \langle\langle B\rangle\rangle\Phi$, which means that the agents of $B$ can achieve the property $\Phi$ without memory but need to randomize.

*Example 1* As we already mentioned, the syntax of SGL does not contain the standard Boolean connectives and temporal operators such as "NextStep" (denoted by $\mathscr{X}$), "Always" (denoted by $\Box$), or "Until" (denoted by $\mathscr{U}$). It is perhaps worth noting how to express these operators in SGL. For example, the formula $\mathscr{P}_{\bowtie\lambda}(\Box\Phi_1)$ can be expressed in SGL as $\mathscr{P}_{\bowtie\lambda}(\mathscr{A}_\Box; \Phi_1)$, where the DRA $\mathscr{A}_\Box$ is shown in Fig. 1 (together with DRA for some other connectives).

### 3.3 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, ATL*, etc., can effectively be translated into SGL. The translation is linear for CTL, PCTL, and ATL; for CTL*, PCTL*, and ATL*, there is an exponential blowup caused by translating LTL properties into deterministic Rabin automata.

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

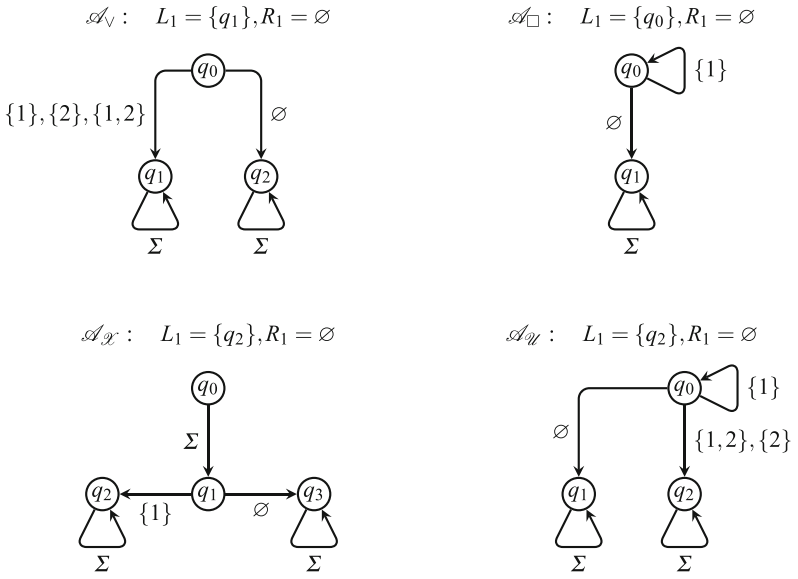


**Fig. 1** DRA implementing some Boolean and modal connectives. The initial state is $q_0$

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 $\Phi$, let $\Phi'$ be the SGL formula obtained from $\Phi$ by substituting each occurrence of the existential quantifier $\exists(\cdot)$ by $\langle\langle\{1\}\rangle\rangle\mathscr{P}_{\geq 1}(\cdot)$, and each occurrence of the universal quantifier $\forall(\cdot)$ by $[\![\{1\}]\!]\mathscr{P}_{\geq 1}(\cdot)$. Then,

$$Sat_{CTL}(\Phi) = Sat_{MD}(\Phi').$$

In the proof of the above equality, one has to realize that if there is a path which satisfies/violates a formula of the form $\Phi_1 \mathscr{U} \Phi_2$, then there is also a path with the same property obtained by fixing exactly one outgoing transition in every state. Also note that $\Phi'$ does not strictly 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 1, the formula $\Phi'$ 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 HD strategies. That is, given a CTL* formula $\Phi$, it holds that $Sat_{CTL^*}(\Phi) = Sat_{HD}(\Phi')$. As CTL* uses LTL path formulae, we need more complicated automata than the ones introduced in Example 1. However, this does not pose any real problems as the languages expressible by LTL formulae are $\omega$-regular, and deterministic Rabin automata are as expressive as $\omega$-regular languages [25,28,29].

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 $\mathscr{P}_{\bowtie\lambda}(\cdot)$ operator implicitly quantifies over all strategies in the given MDP $M$. This is the same as in our SGL semantics. Moreover, given a formula

$$\mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Phi_1, \ldots, \Phi_k),$$

the formulae $\Phi_1, \ldots, \Phi_k$ are interpreted over the same system as the formula $\mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Phi_1, \ldots, \Phi_k)$. 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 deterministic Rabin automata. Given a PCTL formula $\Phi$ and an MDP $M$, it holds that

$$Sat_{PCTL}(\Phi) \;=\; Sat_{MD}(\Phi).$$

Again, the temporal operators have to be substituted by the appropriate automata.

Similarly, PCTL* embeds into SGL. Let $M$ be an MDP and $\Phi$ be a PCTL* formula. Then,

$$Sat_{PCTL^*}(\Phi) \;=\; Sat_{HD}(\Phi).$$

*Remark 1* Let $M$ be an MDP (which can be understood as PMG with one agent 1), and let $\Phi$ be an SGL formula that is obtained from some PCTL* formula in the way indicated above. In particular, note that $\Phi$ does not contain the $\langle\langle\cdot\rangle\rangle$ operator. Assume that $\Phi$ has nested $\mathscr{P}_{\bowtie\lambda}(\cdot)$ operators, so it might look like this: $\mathscr{P}_{\bowtie\lambda}(\ldots\mathscr{P}_{\bowtie'\lambda'}(\ldots)\ldots)$. Let $\Phi'$ be the formula obtained from $\Phi$ by substituting each occurrence of $\mathscr{P}_{\bowtie\lambda}(\cdot)$ by $\|\{1\}\|\mathscr{P}_{\bowtie\lambda}(\cdot)$. Then,

$$\begin{array}{lcll} Sat_{XY}(\Phi) & = & Sat_{HD}(\Phi') & \text{for each strategy class } XY\text{, whereas} \\ Sat_{HD}(\Phi) & \neq & Sat_{HD}(\|\{1\}\|\Phi) & \text{in general.} \end{array}$$

This is because although in $\Phi'$ the $\|\{1\}\|$ operator of the outermost $\mathscr{P}_{\bowtie\lambda}(\cdot)$ operator fixes a strategy for the only agent, this strategy can be overwritten by the $\|\{1\}\|$ operator of a nested $\mathscr{P}_{\bowtie\lambda}(\cdot)$ operator. Thus, we get the standard PCTL semantics. On the other hand, the formula $\|\{1\}\|\Phi$ fixes a strategy $\alpha$ by the $\|\{1\}\|$ operator and evaluates the outermost $\mathscr{P}_{\bowtie\lambda}(\cdot)$ operator on the Markov chain $M^\alpha$. This means that also the nested $\mathscr{P}_{\bowtie\lambda}(\cdot)$ operators are evaluated over $M^\alpha$ which gives the above inequality.

Even ATL is expressible in SGL. In standard ATL, the $\langle\langle\cdot\rangle\rangle_{\text{ATL}}$ operator is followed by a path formula. The ATL semantics of the formula $\langle\langle A\rangle\rangle_{\text{ATL}}\varphi$ 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 $\varphi$ 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 $\mathscr{M}$ without any probabilistic states and an ATL formula $\Phi$, let $\Phi'$ be the SGL formula obtained from $\Phi$ by substituting each occurrence of $\langle\langle A\rangle\rangle_{\text{ATL}}\varphi$ by $\langle\langle A\rangle\rangle\|\mathsf{Agents}\backslash A\|\mathscr{P}_{\geq 1}(\varphi)$. It holds that

$$Sat_{ATL}(\Phi) \;=\; Sat_{HD}(\Phi').$$

The corresponding results hold also for ATL*.

In [2], the authors introduce an extension of ATL called game logic (GL). In contrast to ATL, where the operator $\langle\langle\cdot\rangle\rangle_{\text{ATL}}$ is followed by a path formula and the semantics implicitly quantifies over all paths, the $\langle\langle\cdot\rangle\rangle$ 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. $\Phi$ asserts the existence of a strategy $\alpha$ for the agents in $A$, such that for some behavior of the remaining agents $\varphi_1$ is always true, and for some (possibly different) behavior of the remaining agents $\varphi_2$ is always true. Thus, the chosen strategy $\alpha$ is propagated to the inner subformulae. Nevertheless, the semantics of GL does not propagate strategies chosen by $\langle\langle\cdot\rangle\rangle$ operators to nested $\langle\langle\cdot\rangle\rangle$ 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. [18,19]. They use ATL-like formulae, such as $\langle\langle A\rangle\rangle_{\text{almost}}\psi$ or $\langle\langle A\rangle\rangle_{\text{positive}}\psi$, to formalize the existence of a strategy for agents in $A$ such that

the condition specified by $\psi$ 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 $\psi$ 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 $\langle\!\langle A \rangle\!\rangle_{\text{almost}} \psi$ or $\langle\!\langle A \rangle\!\rangle_{\text{positive}} \psi$ of [18,19] are encoded in SGL by the formulae $\langle\!\langle A \rangle\!\rangle \mathscr{P}_{=1}(\psi)$ and $\langle\!\langle A \rangle\!\rangle \mathscr{P}_{>0}(\psi)$, respectively. However, SGL cannot express the limit operator $\langle\!\langle A \rangle\!\rangle_{\text{limit}}$ of [19].

## 4 Model checking SGL

The model checking problem for SGL addresses the question whether for a given finite-state PMG $\mathscr{M}$, a state $s$ of $\mathscr{M}$, and SGL-formula $\Phi$ it holds that $s \in Sat_{\text{XY}}(\Phi)$ for a given strategy class XY. We analyze the complexity of the model checking problem for SGL and its natural fragments with respect to HR, HD, FR, FD, MR, and MD strategy classes.

We start by recalling the standard notion of polynomial hierarchy [23,24]. For every $\ell \geq 0$, the complexity classes $\Delta_\ell$, $\Sigma_\ell$, and $\Pi_\ell$ are defined inductively as follows:

$$\begin{aligned} &\Delta_0,\ \Sigma_0,\ \Pi_0 \text{ are equal to P,} \\ &\Delta_{i+1} = \mathrm{P}^{\Sigma_i},\ \Sigma_{i+1} = \mathrm{NP}^{\Sigma_i},\ \Pi_{i+1} = \mathrm{coNP}^{\Sigma_i}. \end{aligned}$$

A complete problem for $\Sigma_\ell$, where $\ell \geq 1$, is $\mathsf{QBF}_\ell$. An instance of $\mathsf{QBF}_\ell$ is a quantified Boolean formula (with $\ell$ quantifier alternations) of the form

$$\exists X_1\, \forall X_2\, \exists X_3\, \forall X_4 \cdots Q X_\ell\, \varphi$$

where the variables which appear in the propositional formula $\varphi$ are disjointly partitioned into $X_1, \ldots, X_n$, and $Q$ is either $\forall$ or $\exists$ depending on whether $\ell$ is even or odd, respectively. The question is whether the formula is valid. Without restrictions, we may assume that $\varphi$ takes the form

$$\begin{aligned} \varphi &= \bigwedge_{i=1}^{I} \psi_i, && \text{where } I \geq 1 \text{ and} \\ \psi_i &= \bigvee_{j=1}^{J_i} \xi_{i,j}, && \text{where } J_i \geq 1 \text{ and} \\ \xi_{i,j} &= \bigwedge_{k=1}^{K_{i,j}} \varrho_{i,j,k}, && \text{where } K_{i,j} \geq 1 \text{ and} \\ \varrho_{i,j,k} &= \bigvee_{m=1}^{M_{i,j,k}} \delta_{i,j,k,m}, && \text{where } M_{i,j,k} \geq 1 \text{ and} \\ \delta_{i,j,k,m} &= \bigwedge_{n=1}^{N_{i,j,k,m}} L_{i,j,k,m,n}, && \text{where } N_{i,j,k,m} \geq 1 \text{ and } L_{i,j,k,m,n} \text{ is a literal.} \end{aligned} \tag{1}$$

Here, a literal is a propositional variable or its negation. This assumption is safe because the propositional formula constructed in the proof of Cook's theorem [15] also has the same fixed structure.[4]

[4] The propositional formula constructed in the proof of Cook's theorem is satisfiable iff a given non-deterministic Turing machine $\mathscr{M}$ running in polynomial time accepts a given input word $w$. The formula depends on $\mathscr{M}$ and $w$, but the nesting depth (and structure) of conjunctions and disjunctions is fixed. An explicit construction of the formula can be found in, e.g., [21], and a full justification of our assumption about $\varphi$ follows from the proof of $\Sigma_\ell$-hardness of $\mathsf{QBF}_\ell$; see, e.g., [24], Theorem 17.10.

We show that there is a close correspondence between the polynomial hierarchy and natural syntactic fragments of SGL. For every SGL formula $\Phi$, we define the "type" of $\Phi$, denoted by $\mathsf{Type}(\Phi)$, inductively as follows:

$$\mathsf{Type}(\mathsf{p}) = \Delta_0$$

$$\mathsf{Type}(\neg\Phi) = \begin{cases} \Delta_\ell & \text{if } \mathsf{Type}(\Phi) = \Delta_\ell, \\ \Sigma_\ell & \text{if } \mathsf{Type}(\Phi) = \Pi_\ell, \\ \Pi_\ell & \text{if } \mathsf{Type}(\Phi) = \Sigma_\ell. \end{cases}$$

$$\mathsf{Type}(\langle\langle B\rangle\rangle\Phi) = \begin{cases} \Sigma_{\ell+1} & \text{if } \mathsf{Type}(\Phi) = \Sigma_\ell \text{ or } \mathsf{Type}(\Phi) = \Pi_\ell, \\ \Sigma_\ell & \text{if } \mathsf{Type}(\Phi) = \Delta_\ell. \end{cases}$$

$$\mathsf{Type}(\mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Phi_1, \ldots, \Phi_k)) = \begin{cases} \Delta_{\ell+1} & \text{if } \mathrm{Max}\{\mathsf{Type}(\Phi_1), \ldots, \mathsf{Type}(\Phi_k)\} = \Sigma_\ell \\ \Delta_\ell & \text{if } \mathrm{Max}\{\mathsf{Type}(\Phi_1), \ldots, \mathsf{Type}(\Phi_k)\} = \Delta_\ell. \end{cases}$$

Here $\mathrm{Max}\{\mathsf{Type}(\Phi_1), \ldots, \mathsf{Type}(\Phi_k)\}$ denotes the "maximal" type in the set. More precisely, let $\ell \geq 0$ be the least number such that for every $1 \leq i \leq k$ we have that $\mathsf{Type}(\Phi_i) = \Lambda_j$ where $j \leq \ell$ and $\Lambda \in \{\Sigma, \Pi, \Delta\}$. If for all $1 \leq i \leq k$ such that $\mathsf{Type}(\Phi_i) = \Lambda_\ell$ we have that $\Lambda = \Delta$, then $\mathrm{Max}\{\mathsf{Type}(\Phi_1), \ldots, \mathsf{Type}(\Phi_k)\} = \Delta_\ell$. Otherwise, $\mathrm{Max}\{\mathsf{Type}(\Phi_1), \ldots, \mathsf{Type}(\Phi_k)\} = \Sigma_\ell$. Note that we do not distinguish between the types $\Sigma_\ell$ and $\Pi_\ell$ when defining the maximal type. Intuitively, this is because the maximal type is used as an oracle for a deterministic polynomial-time algorithm which checks the validity of $\mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Phi_1, \ldots, \Phi_k)$, and $P^{\Sigma_\ell} = P^{\Pi_\ell}$ (see Theorem 1).

Now we can define the promised hierarchy of SGL syntactic fragments.

- $\mathrm{SGL}(\Delta_0)$, $\mathrm{SGL}(\Sigma_0)$, and $\mathrm{SGL}(\Pi_0)$ consist of all SGL formulae of type $\Delta_0$, $\Sigma_0$, and $\Pi_0$, respectively.
- $\mathrm{SGL}(\Delta_{i+1})$, $\mathrm{SGL}(\Sigma_{i+1})$, and $\mathrm{SGL}(\Pi_{i+1})$ consist of all formulae in $\mathrm{SGL}(\Delta_i) \cup \mathrm{SGL}(\Sigma_i) \cup \mathrm{SGL}(\Pi_i)$, and all formulae of type $\Delta_{i+1}$, $\Sigma_{i+1}$, and $\Pi_{i+1}$, respectively.

Observe that every SGL formula $\Phi$ has the unique type $\mathsf{Type}(\Phi)$, but there can be formulae equivalent to $\Phi$ whose type is different. For example, the type of $\langle\langle A\rangle\rangle\langle\langle B\rangle\rangle\langle\langle C\rangle\rangle\mathsf{p}$ is $\Sigma_2$, while the types of semantically equivalent formulae $\langle\langle A \cup C\rangle\rangle\neg\neg\langle\langle B\rangle\rangle\mathsf{p}$ and $\langle\langle A \cup B \cup C\rangle\rangle\mathsf{p}$ are $\Sigma_1$ and $\Sigma_0$, respectively. Let us also note that we could alternatively define

$$\mathsf{Type}(\langle\langle B\rangle\rangle\Phi) = \begin{cases} \Sigma_{\ell+1} & \text{if } \mathsf{Type}(\Phi) = \Pi_\ell, \\ \Sigma_\ell & \text{if } \mathsf{Type}(\Phi) = \Delta_\ell \text{ or } \mathsf{Type}(\Phi) = \Sigma_\ell, \end{cases}$$

without influencing the validity of our results. Perhaps, this alternative variant better corresponds to the standard intuition about the $\Sigma_\ell$ and $\Pi_\ell$ classes, but it also causes complications in the proof of Lemma 1 and its follow-up arguments. For the sake of technical simplicity, we decided to keep the original variant.

### 4.1 MD strategies

In this section we examine the SGL model checking problem with respect to MD semantics.

**Lemma 1** *Let $\mathscr{M} = (\mathsf{Agents}, S, \to, \mathsf{P}, \mathsf{Props}, v)$ be a PMG where $\mathsf{Agents} \subseteq \mathsf{AG}$, $\mathsf{Props} \subseteq \mathsf{AP}$, and let $\Phi$ be an SGL formula. The problem whether $s, A, \alpha \models_{MD} \Phi$, where $s \in S$, $A \subseteq \mathsf{Agents}$, and $\alpha$ is an MD $A$-strategy, is solvable in $\mathsf{Type}(\Phi)$.*

*Proof* We proceed by induction on the structure of $\Phi$. The cases when $\Phi = \mathsf{p}$ and $\Phi = \neg\Psi$ are immediate.

If $\Phi = \langle\langle B \rangle\rangle \Psi$, it suffices to guess an appropriate MD $B$-strategy $\beta$ and check whether $s, A \cup B, (\alpha \leftarrow \beta) \models_{MD} \Psi$. Hence, by applying induction hypothesis, the problem whether $s, A, \alpha \models_{MD} \Phi$ is in $\mathrm{NP}^{\mathsf{Type}(\Psi)}$, which is equal to $\mathsf{Type}(\Phi)$ by the definition of $\mathsf{Type}$. Recall that $\mathrm{NP}^{\Delta_\ell} = \Sigma_\ell$ and $\mathrm{NP}^{\Sigma_\ell} = \mathrm{NP}^{\Pi_\ell} = \Sigma_{\ell+1}$.

If $\Phi = \mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Psi_1, \ldots, \Psi_k)$, we first apply the current MD $A$-strategy $\alpha$ to $\mathscr{M}$ and construct the PMG $\mathscr{M}^\alpha$ (see Sect. 2.1). Further, for every state $t$ we compute the set $\sigma(t)$ of all $j \in \{1, \ldots, k\}$ such that $t, A, \alpha \models_{MD} \Psi_j$. By induction hypothesis, this is achievable by a deterministic polynomial-time algorithm with $\mathrm{Max}\{\mathsf{Type}(\Psi_1), \ldots, \mathsf{Type}(\Psi_k)\}$ oracle. Since the agents in $\mathsf{Agents} \setminus A$ are now considered adversarial, we interpret $\mathscr{M}^\alpha$ as an MDP where the only agent $\{a\}$ controls all $(\mathsf{Agents} \setminus A)$-states. Further, we interpret $\{1, \ldots, k\}$ as the set of atomic propositions of $\mathscr{M}^\alpha$, and $\sigma$ as the corresponding valuation. If $\bowtie \in \{>, \geq\}$, we simply check if $val^-(\mathscr{A}, s) \bowtie \lambda$, where $s$ is considered as a state of $\mathscr{M}^\alpha$. Note that this is achievable in time polynomial in the size of $\mathscr{M}^\alpha$ and $\mathscr{A}$ by solving the linear program of Proposition 1.

Similarly, if $\bowtie \in \{<, \leq\}$, we check if $val^+(\mathscr{A}, s) \bowtie \lambda$. Hence, the problem whether $s, A, \alpha \models_{MD} \mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Psi_1, \ldots, \Psi_k)$ is in $P^{\mathrm{Max}\{\mathsf{Type}(\Psi_1), \ldots, \mathsf{Type}(\Psi_k)\}}$ which is equal to $\mathsf{Type}(\mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Psi_1, \ldots, \Psi_k))$ by the definition of $\mathsf{Type}$. □

Now we prove the corresponding lower complexity bound. For the sake of readability, we adopt the following abbreviations, where $\mathscr{A}_\vee$, $\mathscr{A}_{\mathscr{X}}$, $\mathscr{A}_\Box$, and $\mathscr{A}_{\mathscr{U}}$ are the automata of Fig. 1.

$$\Phi_1 \Rightarrow \Phi_2 = \mathscr{P}_{=1}(\mathscr{A}_\vee; \neg\Phi_1, \Phi_2)$$
$$\mathscr{X}^{\bowtie\lambda}\Phi = \mathscr{P}_{\bowtie\lambda}(\mathscr{A}_{\mathscr{X}}; \Phi)$$
$$\Box^{\bowtie\lambda}\Phi = \mathscr{P}_{\bowtie\lambda}(\mathscr{A}_\Box; \Phi)$$
$$\Phi_1 \,\mathscr{U}^{\bowtie\lambda}\, \Phi_2 = \mathscr{P}_{\bowtie\lambda}(\mathscr{A}_{\mathscr{U}}; \Phi_1, \Phi_2)$$
$$\|A\|\Phi = \neg\langle\langle A \rangle\rangle\neg\Phi$$

We also write $\langle\langle a \rangle\rangle \Phi$ instead of $\langle\langle \{a\} \rangle\rangle \Phi$ for a single agent $a$.

**Lemma 2** *For every $\ell \geq 0$, there is a fixed formula $\Phi \in SGL(\Sigma_\ell)$ such that the model checking problem for $\Phi$ is $\Sigma_\ell$-hard.*

*Proof* Let $\exists X_1 \forall X_2 \exists X_3 \forall X_4 \cdots Q X_\ell \varphi$ be a quantified Boolean formula with $\ell$ quantifier alternations, where the propositional formula $\varphi$ takes the special form (1) introduced at the beginning of Sect. 4. We construct a PMG $\mathscr{M} = (\mathsf{Agents}, S, \to, \mathsf{P}, \mathsf{Props}, \nu)$, a state $s(\varphi) \in S$, and a fixed formula $\Phi \in \mathrm{SGL}(\Sigma_\ell)$ such that $\exists X_1 \forall X_2 \exists X_3 \forall X_4 \cdots Q X_\ell \varphi$ is valid iff $s(\varphi) \in Sat_{MD}(\Phi)$.

Let $\{x_1, \ldots, x_r\}$ be the set of all propositional variables that appear in $\varphi$. The PMG $\mathscr{M}$ is constructed as follows. We put $\mathsf{Agents} = \{a_1, \ldots, a_\ell\}$ and $\mathsf{Props} = \{t, f, d, a, e, b\}$. The probability assignment $\mathsf{P}$ can be chosen arbitrarily (the precise values of transition probabilities do not influence our arguments; note that $\mathsf{P}$ is required to be positive by Definition 1). The states, transitions, and labelling function of $\mathscr{M}$ are defined incrementally. For all $u \in \{1, \ldots, r\}$, we put to $S$ a fresh stochastic state $B_u$ which satisfies $b$ and no other proposition. Further, for every $\delta_{i,j,k,m}$ subformula of $\varphi$, we add to $\mathscr{M}$ the gadget with initial state $\delta$ depicted in Fig. 2. All states of the gadget are fresh, except for $B_1, \ldots, B_r$ that are shared by all gadgets (note that the number of outgoing transitions of every $B_u$ is the same and it is equal to the number of $\delta_{i,j,k,m}$ subformulae of $\varphi$). The states $p_1, \ldots, p_r$ are controlled by the agents, and the other states are stochastic. Intuitively, the agent controlling $p_u$ can set the variable $x_u$ to true or false by selecting the transition leading to $x_u$ or $\bar{x}_u$, respectively. The state $p_u$ satisfies either the atomic proposition $e$ or $a$, depending on whether $x_u$ is

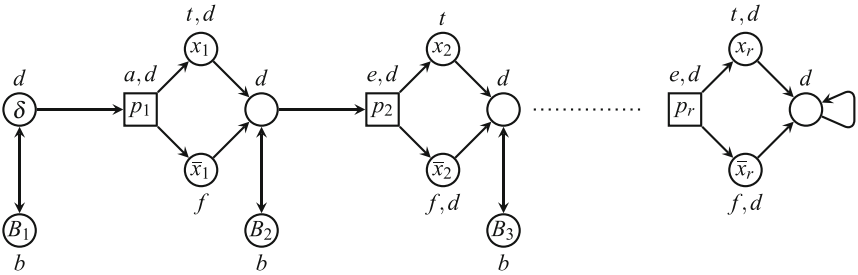


**Fig. 2** The gadget for $\delta_{i,j,k,m}$ subformula

quantified existentially (i.e., $x_u \in X_i$ where $i$ is odd) or universally, respectively. For every $i \in \{1, \ldots, \ell\}$, the agent $a_i$ sets the variables in $X_i$, i.e., if $x_u \in X_i$, then $p_u$ is an $a_i$-state. The propositions $t$ and $f$ are satisfied exactly in $x_1, \ldots, x_r$ and $\bar{x}_1, \ldots, \bar{x}_r$, respectively, and the proposition $d$ encodes the structure of $\delta_{i,j,k,m}$ in the following way: for every variable $x_u$, where $1 \leq u \leq r$,

- if $\delta_{i,j,k,m}$ contains both $x_u$ and $\neg x_u$ (as literals), then $d \notin \nu(x_u)$ and $d \notin \nu(\bar{x}_u)$;
- if $\delta_{i,j,k,m}$ does not contain $x_u$ and $\neg x_u$, then $d \in \nu(x_u)$ and $d \in \nu(\bar{x}_u)$;
- if $\delta_{i,j,k,m}$ contains $x_u$ and does not contain $\neg x_u$, then $d \in \nu(x_u)$ and $d \notin \nu(\bar{x}_u)$;
- if $\delta_{i,j,k,m}$ does not contain $x_u$ and contains $\neg x_u$, then $d \notin \nu(x_u)$ and $d \in \nu(\bar{x}_u)$.

The other states of the gadget, different from $x_1, \ldots, x_r, \bar{x}_1, \ldots, \bar{x}_r, B_1, \ldots, B_r$, satisfy $d$ as well.

Note that in general, agents can play *inconsistently* by setting the same variable to true and false in different gadgets. However, this can be easily detected. Consider the following formulae:

$$Surely_t = (\neg t \wedge \neg f)\, \mathscr{U}^{=1}\, t$$

$$Surely_f = (\neg t \wedge \neg f)\, \mathscr{U}^{=1}\, f$$

$$Cons_e = \Box^{=1}\Big((\mathscr{X}^{>0} b \wedge \mathscr{X}^{>0} e) \Rightarrow \big(Surely_t \vee Surely_f\big)\Big)$$

$$Cons_a = \Box^{=1}\Big((\mathscr{X}^{>0} b \wedge \mathscr{X}^{>0} a) \Rightarrow \big(Surely_t \vee Surely_f\big)\Big)$$

Let $\delta$ be the initial state of some of the constructed gadgets, and let $\alpha$ be an MD Agents-strategy. We claim that $\alpha$ is consistent iff

$$\delta, \mathsf{Agents}, \alpha \models_{\mathrm{MD}} Cons_e \wedge Cons_a.$$

The formula $Cons_e$ encodes the consistency of $\alpha$ with respect to existentially quantified variables, and the formula $Cons_a$ does the same for universally quantified variables (the reason why we treat the existentially/universally quantified variables separately becomes clear later). To see this, consider, e.g., the choice made by $\alpha$ in the states $p_u$, where $x_u$ is quantified universally. Let $q$ be a predecessor of $p_u$ in some gadget. Then $q, \mathsf{Agents}, \alpha \models_{\mathrm{MD}} \mathscr{X}^{>0} b \wedge \mathscr{X}^{>0} a$. If $\alpha$ behaves consistently in all $p_u$, i.e., selects either always the transition to $x_u$ or always the transition to $\bar{x}_u$, then $q, \mathsf{Agents}, \alpha \models_{\mathrm{MD}} Surely_t$ or $q, \mathsf{Agents}, \alpha \models_{\mathrm{MD}} Surely_f$, respectively. On the other hand, if $\alpha$ behaves inconsistently in $p_u$, then $q, \mathsf{Agents}, \alpha \not\models_{\mathrm{MD}} Surely_t$ and $q, \mathsf{Agents}, \alpha \not\models_{\mathrm{MD}} Surely_f$. These observations are easy to verify by examining the structure of the gadgets, and also explain the role of $B_u$ states.

Observe that every assignment $\mu$ for the propositional variables $x_1, \ldots, x_r$ determines a unique consistent MD Agents-strategy $\alpha_\mu$, and vice versa. For every assignment $\mu$ and every $\delta_{i,j,k,m}$ subformula we have that $\delta_{i,j,k,m}$ is true in $\mu$ iff

$$\delta, \mathsf{Agents}, \alpha_\mu \models_{\mathrm{MD}} \neg b \, \mathscr{U}^{=0} \, \neg d.$$

This follows directly from the construction of the gadgets.

Now we complete the construction of $\mathscr{M}$ by adding the following states and transitions:

- we add a state $s(\varphi)$; further, for every $\psi_i$ subformula we add a state $s(\psi_i)$, for every $\xi_{i,j}$ subformula we add a state $s(\xi_{i,j})$, and for every $\varrho_{i,j,k}$ subformula we add a state $s(\varrho_{i,j,k})$;
- we add a transition $s(\varphi) \to s(\psi_i)$ for every $1 \leq i \leq I$;
- for every $\psi_i$ subformula, we add a transition $s(\psi_i) \to s(\xi_{i,j})$ for all $1 \leq j \leq J_i$;
- for every $\xi_{i,j}$ subformula, we add a transition $s(\xi_{i,j}) \to s(\varrho_{i,j,k})$ for all $1 \leq k \leq K_{i,j}$;
- for every $\varrho_{i,j,k}$ subformula and every $1 \leq m \leq M_{i,j,k}$, we add a transition $s(\varrho_{i,j,k}) \to \delta$ where $\delta$ is the initial state of the gadget for $\delta_{i,j,k,m}$.

Finally, we define the formula $\Phi$ as follows:

$$\langle\!\langle a_1 \rangle\!\rangle \, \|a_2\| \, \langle\!\langle a_3 \rangle\!\rangle \cdots \big[a_\ell\big] \, \mathscr{X}^{=1} \, \mathscr{X}^{>0} \, \mathscr{X}^{=1} \, \mathscr{X}^{>0} \big(\neg \mathit{Cons}_a \vee (\mathit{Cons}_e \wedge \neg b \, \mathscr{U}^{=0} \, \neg d)\big)$$

Here $\big[a_\ell\big]$ is either $\langle\!\langle a_\ell \rangle\!\rangle$ or $\|a_\ell\|$, depending on whether $\ell$ is odd or even, respectively. Observe that $\Phi \in \mathrm{SGL}(\Sigma_\ell)$ and $\Phi$ is a fixed formula for a fixed $\ell$. We claim that

$$\exists X_1 \, \forall X_2 \, \exists X_3 \, \forall X_4 \cdots \, Q X_\ell \, \varphi \text{ is valid iff } s(\varphi) \in \mathit{Sat}_{MD}(\Phi).$$

The "$\Rightarrow$" direction follows by observing that

- if $i$ is even, then the agent $a_i$ does not gain anything by using an inconsistent strategy, because this inevitably makes the subformula $\neg \mathit{Cons}_a$ valid;
- if $i$ is odd and all of the previous choices of the agents were consistent (i.e., encode an assignment for the variables in $X_1 \cup \cdots \cup X_{i-1}$), then $a_i$ simply takes an assignment for $X_i$ which makes the formula $\forall X_{i+1} \cdots \, Q X_\ell \, \varphi$ valid and encodes this assignment in his strategy. If some of the previous choices of the agents were inconsistent, then $a_i$ can play arbitrarily.

The "$\Rightarrow$" direction follows similarly. □

A direct consequence of Lemmas 1 and 2 is the following:

**Theorem 1** *Let $\ell \geq 0$. Then*

- *the model checking problem for SGL($\Sigma_\ell$) is $\Sigma_\ell$-complete for MD semantics, and the hardness result holds even for a fixed SGL($\Sigma_\ell$) formula;*
- *the model checking problem for SGL($\Pi_\ell$) is $\Pi_\ell$-complete for MD semantics, and the hardness result holds even for a fixed SGL($\Pi_\ell$) formula;*
- *the model checking problem for SGL($\Delta_\ell$) is in $\Delta_\ell$ for MD semantics.*

*Finally, the model checking problem for SGL is* PSPACE*-complete for MD semantics.*

Observe that PSPACE-hardness of the model checking problem for SGL follows immediately from the proof of Lemma 2 (we can use the same construction to reduce the QBF problem to SGL model checking). However, the constructed SGL formula is not fixed anymore because the quantifier alternation depth in QBF instances is not fixed. In fact, it is unlikely that SGL model checking is PSPACE-hard for some fixed SGL formula $\Phi$, because this would imply collapse of the polynomial hierarchy at $\mathsf{Type}(\Phi)$ level. Let us note that one can also give a simple direct proof for PSPACE-hardness of SGL model checking which avoids many of the technicalities presented in the proof of Lemma 1.

### 4.2 MR strategies

We start by observing that if we restrict ourselves to the *qualitative fragment* of SGL (see Sect. 3.1), then randomization brings only a limited extra power to the agents, and the corresponding results are the same as for MD strategies.

**Theorem 2** *Let $\ell \geq 0$. Then*

- *the model checking problem for qualitative SGL($\Sigma_\ell$) is $\Sigma_\ell$-complete for MR semantics, and the hardness result holds even for a fixed qualitative SGL($\Sigma_\ell$) formula;*
- *the model checking problem for qualitative SGL($\Pi_\ell$) is $\Pi_\ell$-complete for MR semantics, and the hardness result holds even for a fixed qualitative SGL($\Pi_\ell$) formula;*
- *the model checking problem for qualitative SGL($\Delta_\ell$) is in $\Delta_\ell$ for MR semantics.*

*Finally, the model checking problem for qualitative SGL is* PSPACE*-complete for MR semantics.*

Note that the formula $\Phi$ constructed in the proof of Lemma 2 is qualitative, and the agents do not gain anything by using randomized strategies. Hence, the proof of Lemma 2 works also for qualitative SGL($\Sigma_\ell$) and MR strategies without any change, and thus we obtain the lower bounds of Theorem 2. The upper complexity bounds follow from Lemma 1, which is valid also for MR strategies and qualitative SGL($\Sigma_\ell$). However, the subcase $\Phi \equiv \langle\langle B \rangle\rangle \Psi$ in the proof of Lemma 1 requires a slight modification. Realize that for the qualitative fragment of SGL, the exact values of transition probabilities chosen by the agents do not really matter; it is only important which of them are chosen with positive/zero probability. Hence, instead of guessing a single outgoing transition for every $B$-state, which was enough for MD strategies, we now guess for every $B$-state a subset of outgoing transitions that are assigned a positive probability (and choose an arbitrary positive distribution over the chosen successors). The rest of the proof of Lemma 1 does not require any modification.

For general SGL formulae, the exact values of transition probabilities are of course relevant, and our decidability proof is based on encoding the SGL model checking problem into first-order theory of the reals, i.e., $(\mathbb{R}, *, +, \leq)$, which is known to be decidable [26]. Our proof can be seen as an extension of the previous result [22] where it was shown that the MR-controller synthesis for MDPs (viewed as $1\frac{1}{2}$-player games) and PCTL specifications can effectively be encoded by closed formulae of $(\mathbb{R}, *, +, \leq)$.

In the proof of our next lemma we often construct some finite index set $I = \{i_1, \ldots, i_n\}$ and then fix a fresh first-order variable $X_i$ for every $i \in I$. To simplify our notation, we define

$$\widetilde{\exists} X_j . \psi = (\exists X_{i_1} \exists X_{i_2} \cdots \exists X_{i_n}) . \psi .$$

Similarly, if $J \subseteq I$ where $J = \{j_1, \ldots, j_m\}$, we write just $(\widetilde{\exists} X_j : j \in J) . \psi$ instead of $(\exists X_{j_1} \exists X_{j_2} \cdots \exists X_{j_m}) . \psi$.

**Lemma 3** *Let $\mathscr{M} = (\mathsf{Agents}, S, \to, \mathsf{P}, \mathsf{Props}, \nu)$ be a PMG where $\mathsf{Agents} \subseteq \mathsf{AG}$, $\mathsf{Props} \subseteq \mathsf{AP}$, and let $\Phi$ be an SGL formula. For every $s \in S$ there is a closed formula $\tau(s, \Phi)$ of $(\mathbb{R}, *, +, \leq)$ such that $s \in Sat_{MR}(\Phi)$ iff $\tau(s, \Phi)$ holds.*

*Proof* For every transition $s \to t$ of $\mathscr{M}$, we use $Y_{s,t}$ to denote either the constant whose value is equal to $\mathsf{P}(s, t)$ if $s$ is stochastic, or a fresh first-order variable that encodes the probability of $s \to t$ chosen by the responsible agent. For every subset $C$ of agents, we construct a closed formula $\tau_C(s, \Phi)$ of $(\mathbb{R}, *, +, \leq)$ such that $\tau_C(s, \Phi)$ is valid iff $s, C, \gamma \models_{\mathrm{MR}} \Phi$, where the strategy $\gamma$ is given by the values of the variables $Y_{s,t}$ for $s \in S_C$. Then, we simply put $\tau(s, \Phi) = \tau_\varnothing(s, \Phi)$.

The formula $\tau_C(s, \Phi)$ is defined by induction on the structure of $\Phi$. The first three subcases are immediate. We put

$$\begin{aligned}
\tau_C(s, p) &= \begin{cases} \mathit{true} & \text{if } p \in \nu(s), \\ \mathit{false} & \text{otherwise;} \end{cases} \\
\tau_C(s, \neg\Phi) &= \neg\tau_C(s, \Phi), \\
\tau_C(s, \langle\!\langle A \rangle\!\rangle \Phi) &= \big(\widetilde{\exists} Y_{r,t} : r \in S_A, r \to t\big) . \left( \tau_{C \cup A}(s, \Phi) \wedge \bigwedge_{r \in S_A} \mathit{Dist}_r \right),
\end{aligned}$$

where

$$\mathit{Dist}_r = \left( \bigwedge_{r \to t} 0 \leq Y_{r,t} \leq 1 \right) \wedge \left( \sum_{r \to t} Y_{r,t} = 1 \right).$$

Now let $\Phi = \mathscr{P}_{\bowtie\lambda}(\mathscr{A}; \Psi_1, \ldots, \Psi_k)$, where $\mathscr{A} = (Q, \Sigma, q_{init}, \delta, (L_i, R_i)_{i=1}^m)$. Let us consider the MDP $\mathscr{M}^\gamma$ with atomic propositions $\{1, \ldots, k\}$ and a labelling function $\eta$ such that $\eta(t)$ consists of all $i$ where $1 \leq i \leq k$ and $t, C, \gamma \models^{\mathscr{M}}_{\mathrm{MR}} \Psi_i$. We need to encode either the property $val^+(\mathscr{A}, s) \bowtie \lambda$ or $val^-(\mathscr{A}, s) \bowtie \lambda$ (where $s$ is interpreted as a state of $\mathscr{M}^\gamma$), depending on whether $\bowtie \in \{\leq, <\}$ or $\bowtie \in \{\geq, >\}$, respectively. According to Proposition 1, this means to encode

$$\text{either } \mu[\Delta_{acc}](x_{s,q_{init}}) \bowtie \lambda \text{ or } 1 - \mu[\Delta_{rej}](x_{s,q_{init}}) \bowtie \lambda,$$

respectively. Note that the first inequality holds iff there exists *some* solution *sol* of $\Delta_{acc}$ in $[0, 1]$ such that $sol(x_{s,q_{init}}) \bowtie \lambda$. Realize that $\mu[\Delta_{acc}] \leq sol$. Similarly, the second inequality holds iff there exists some solution *sol* of $\Delta_{rej}$ in $[0, 1]$ such that $1 - sol(x_{s,q_{init}}) \bowtie \lambda$.

For all $t \in S$ and $q \in Q$, let $Z_{t,q}$ be a fresh first-order variable. The formula $\tau_C(s, \Phi)$ takes the form

$$\tau_C(s, \Phi) = \widetilde{\exists} Z_{t,q} . \left( \bigwedge_{t \in S, q \in Q} 0 \leq Z_{t,q} \leq 1 \right) \wedge \mathit{Sol} \wedge \mathit{Req}$$

where

$$\mathit{Req} = \begin{cases} Z_{s,q_{init}} \bowtie \lambda & \text{if } \bowtie \in \{\leq, <\}, \\ 1 - Z_{s,q_{init}} \bowtie \lambda & \text{otherwise.} \end{cases}$$

The subformula *Sol* says that the variables $Z_{t,q}$ form a solution of $\Delta_{acc}$ and $\Delta_{rej}$ in $[0, 1]$, respectively.

The construction of *Sol* is not trivial, because the labeling $\eta$ depends on the concrete form of $\Psi_1, \ldots, \Psi_k$. Hence, we first need to encode $\eta$ and the structure of $\mathscr{M}^\gamma \times \mathscr{A}$ symbolically, and then we can proceed with encoding the (in)equalities of $\Delta_{acc}$ and $\Delta_{rej}$ (cf. the proof of Proposition 1).

Let us fix a fresh variable $X_{t,i}$ for all $t \in S$ and $i \in \{1, \ldots, k\}$. We construct a formula *Eta* which ensures that $X_{t,i}$ is positive iff $t, C, \gamma \models_{\mathrm{MR}} \Psi_i$.

$$\mathit{Eta} = \bigwedge_{t \in S, i \in \{1,\ldots,k\}} \big(X_{t,i} > 0 \Leftrightarrow \tau_C(t, \Psi_i)\big)$$

Further, we fix a fresh variable $T_{(t,q),(t',q')}$ for all $(t, q), (t', q') \in S \times Q$ where $(t, q) \neq (t', q')$, and construct a formula *Tran* which say that $T_{(t,q),(t',q')}$ is either 1 or 0, depending

on whether $(t,q) \to_{\otimes} (t',q')$ in $\mathscr{M}^{\gamma} \times A$ or not, respectively.

$$Tran = \bigwedge_{\substack{t,t' \in S \\ q,q' \in Q}} \left(T_{(t,q),(t',q')} = 1 \vee T_{(t,q),(t',q')} = 0\right) \wedge \bigwedge_{\substack{t,t' \in S \\ q,q' \in Q \\ t \not\to t'}} T_{(t,q),(t',q')} = 0$$

$$\wedge \bigwedge_{\substack{t,t' \in S \\ q,q' \in Q \\ t \not\to t'}} \left( T_{(t,q),(t',q')} = 1 \Leftrightarrow \bigvee_{\substack{A \subseteq \{1,\ldots,k\} \\ \delta(q,A) = q'}} \left( \bigwedge_{i \in A} X_{t,i} > 0 \wedge \bigwedge_{i \notin A} X_{t,i} \leq 0 \right) \right)$$

Now we encode the existence of an accepting/rejecting end component of $\mathscr{M}^{\gamma} \times \mathscr{A}$. For all $(t,q) \in S \times Q$, we fix a fresh variable $V_{t,q}$. We construct a formula $Acc$ which says that the set of all $(t,q)$, where $V_{t,q} > 0$, forms an accepting/rejecting end component.

$$AccRej = \bigwedge_{\substack{t \in S_{Prob} \cup S_C \\ q \in Q}} V_{t,q} > 0 \Rightarrow \left( \bigwedge_{\substack{t' \in S \\ q' \in Q}} T_{(t,q)(t',q')} \Rightarrow V_{t',q'} > 0 \right) \tag{2}$$

$$\wedge \bigwedge_{\substack{t \in S_{\mathsf{Agents} \setminus C} \\ q \in Q}} V_{t,q} > 0 \Rightarrow \left( \bigvee_{\substack{t' \in S \\ q' \in Q}} T_{(t,q)(t',q')} \wedge V_{t',q'} > 0 \right) \tag{3}$$

$$\wedge \bigwedge_{\substack{t,t' \in S \\ q,q' \in Q}} \left(V_{t,q} > 0 \wedge V_{t',q'} > 0\right) \Rightarrow Reach^{\leq N}(t,q)(t',q') \tag{4}$$

$$\wedge\ Rabin \tag{5}$$

Formula (2) says that the set of states encoded by $V_{t,q}$ variables is closed under successors of stochastic states, and (3) says that each non-deterministic state in the set has at least one successor in the set. The condition of strong connectedness is encoded by (4), where $N = |S| \cdot |Q|$, and (5) says that the end component is accepting or rejecting, depending on whether $\bowtie \in \{\leq, <\}$ or $\bowtie \in \{\geq, >\}$, respectively. The formula $Reach^{\leq N}(t,q)(t',q')$ says that $(t,q)$ can reach $(t',q')$ by a sequence of at most $N$ transitions, and it is constructed inductively as follows:

$$Reach^{\leq 0}(t,q)(t',q') = \begin{cases} true & \text{if } t = t' \text{ and } q = q', \\ false & \text{otherwise;} \end{cases}$$

$$Reach^{\leq i+1}(t,q)(t',q') = \bigvee_{\substack{t'' \in S \\ q'' \in Q}} \left( Reach^{\leq i}(t,q)(t'',q'') \wedge T_{(t'',q'')(t',q')} = 1 \right)$$

The formula $Rabin$ is easy to construct. We put either

$$Rabin = \bigvee_{1 \leq i \leq m} \left( \bigvee_{\substack{q \in L_i \\ t \in S}} (V_{t,q} > 0) \wedge \bigwedge_{\substack{q \in R_i \\ t \in S}} (V_{t,q} \leq 0) \right)$$

or

$$Rabin \;=\; \bigwedge_{1\leq i\leq m} \left( \bigwedge_{\substack{q\in L_i\\ t\in S}} (V_{t,q} \leq 0) \vee \bigvee_{\substack{q\in R_i\\ t\in S}} (V_{t,q} > 0) \right)$$

depending on whether $\bowtie \in \{\leq, <\}$ or $\bowtie \in \{\geq, >\}$, respectively. Now we can finally express that the $Z_{t,q}$ variables form a solution of $\Delta_{acc}$ or $\Delta_{rej}$. Let

$$Sat \;=\; \bigwedge_{\substack{r\in S\\ p\in Q}} \big(\widetilde{\exists} V_{t,q}.\, (AccRej \wedge V_{r,p})\big) \Rightarrow (Z_{r,p} = 1)$$

$$\wedge \bigwedge_{\substack{r\in S_{Prob}\cup S_C\\ p\in Q}} \neg\big(\widetilde{\exists} V_{t,q}.\, (AccRej \wedge V_{r,p})\big) \Rightarrow \left( Z_{r,p} = \sum_{\substack{r'\in S\\ p'\in Q}} T_{(r,p)(r',p')} \cdot Y_{r,r'} \cdot Z_{r',p'} \right)$$

$$\wedge \bigwedge_{\substack{r\in S_{\mathsf{Agents}\setminus C}\\ p\in Q}} \neg\big(\widetilde{\exists} V_{t,q}.\, (AccRej \wedge V_{r,p})\big) \Rightarrow \left( \bigwedge_{\substack{r'\in S\\ p'\in Q}} T_{(r,p)(r',p')} = 1 \;\Rightarrow\; Z_{r,p} \geq Z_{r',p'} \right)$$

Now we put

$$Sol \;=\; \widetilde{\exists} X_{t,i}.\, \widetilde{\exists} T_{(t,q)(t',q')}.\, (Eta \wedge Tran \wedge Sat).$$

The correctness of our construction follows by verifying that all subformulae have the intended meaning, which is straightforward. □

The following theorem is a direct consequence of Theorem 2 and Lemma 3.

**Theorem 3** *Let $\ell \geq 0$. Then*

- *the model checking problem for $SGL(\Sigma_\ell)$ is in* EXPTIME *and $\Sigma_\ell$-hard for MR semantics; the hardness result holds even for a fixed $SGL(\Sigma_\ell)$ formula;*
- *the model checking problem for $SGL(\Pi_\ell)$ is in* EXPTIME *and $\Pi_\ell$-hard for MR semantics; the hardness result holds even for a fixed $SGL(\Pi_\ell)$ formula;*
- *the model checking problem for $SGL(\Delta_\ell)$ is in* EXPTIME *for MR semantics.*

*Finally, the model checking problem for SGL is in* EXPSPACE *and* PSPACE*-hard for MR semantics.*

The lower bounds of Theorem 3 are just inherited from Theorem 2. The upper bounds are obtained by analyzing the size and structure of the formula $\tau(s, \Phi)$ constructed in the proof of Lemma 3, and applying known results about the complexity of $(\mathbb{R}, *, +, \leq)$ and its fragments. Note that the size of $\tau(s, \Phi)$ is *polynomial* in the size of $\mathscr{M}$ and $\Phi$, and the quantifier alternation depth of $\tau(s, \Phi)$ is fixed for every $\ell \geq 0$ (after pushing all negations inside). The general upper bound for $(\mathbb{R}, *, +, \leq)$ is EXPSPACE. The existential fragment of $(\mathbb{R}, *, +, \leq)$ can be decided in polynomial space [11], and every fragment of $(\mathbb{R}, *, +, \leq)$ obtained by restricting the quantifier alternation depth to some fixed level is solvable is exponential time [20]. Thus, we obtain the upper bounds of Theorem 3. Also note that for $\ell = 0$, the upper bounds trivially improve to P (cf. Proposition 1), and for $\ell = 1$, they improve to PSPACE.

### 4.3 FR, FD, HR, and HD strategies

For history-dependent strategies, the SGL model checking problem becomes undecidable. This follows immediately from the undecidability result for $1\frac{1}{2}$-player games and PCTL stated in [8]. More precisely, [8] 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 $\Phi$ is a PCTL formula. This holds for FR, FD, HR, and HD semantics. For HR and HD semantics, the above problem is even shown to be *highly* undecidable, i.e., beyond the arithmetical hierarchy. Thus, we obtain the following:

**Theorem 4** *The SGL model checking problem is undecidable for FR, FD, HR, and HD semantics. This result holds even for the SGL($\Sigma_1$) and SGL($\Pi_1$) fragments.*

The results of [8] do not apply to the qualitative fragment of SGL. In [7], it was shown that the controller synthesis problem for finite-state MDPs and qualitative PECTL* objectives is decidable. Since the underlying argument is quite involved, the question whether this result can be generalized to qualitative SGL is postponed to future work.

## 5 Conclusion

We introduced a new SGL interpreted over probabilistic multi-player games (PMG). It combines features of ATL, PCTL and ECTLs. Our logic uses an existential strategy quantifier $\langle\langle\cdot\rangle\rangle$ 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 by a polynomially time-bounded algorithm [2], modifying the semantics of the $\langle\langle\cdot\rangle\rangle$ operator so that the strategy decisions are propagated to the subformulae makes the model checking problem PSPACE-hard. The main results of this paper can be summarized as follows.

The model checking problem for finite-state PMG and (full) 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 PMG 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.

**Acknowledgments** Tomáš Brázdil and Antonín Kučera are supported by the Czech Science Foundation, grant No. P202/12/G061. Christel Baier is supported by the projects PROBPOR and ROCKS funded by the German Research Foundation.

## References

1. Ågotnes, T., Goranka, V., Jamroga, W.: Alternating-time temporal logic with irrevocable strategies. In: 11th Confernce on Theoretical Aspects of Rationality and Knowledge (TARK'07), pp. 15–24 (2007)
2. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. J. ACM **49**, 672–713 (2002)
3. Baier, C., Brázdil, T., Größer, M., Kučera, A.: Stocahstic game logic. In: Fourth International Conference on Quantitative Evaluation of Systems (QEST'07), pp. 227–236. IEEE Computer Society Press, Los Alamos (2007)

4. Baier, C., Größer, M., Leucker, M., Ciesinski, F., Bollig, B.: Controller synthesis for probabilistic systems. In: IFIP Worldcongress, Theoretical Computer Science (2004)
5. Baier, C., Katoen, J.-P.: Principles of Model Checking. MIT Press, Cambridge (2008)
6. Bianco, A., de Alfaro, L.: Model checking of probabilistic and nondeterministic systems. In: Thiagarajan, P.S. (ed.) Proceedings of Foundations of Software Technology and Theoretical Computer Science, 15th Conference, Bangalore, India, December 18–20, Lecture Notes in Computer Science 1026, pp. 499–513 (1995). ISBN 3-540-60692-0
7. Brázdil, T., Forejt, V., Kučera, A.: Controller Synthesis and Verification for Markov Decision Processes with Qualitative Branching Time Objectives. In: Proceedings of the ICALP 2008, Lecture Notes in Computer Science, vol. 5126, pp. 148–159 (2008)
8. Brázdil, T., Brožek, V., Forejt, V., Kučera, A.: Stochastic games with branching-time winning objectives. In: Proceedings if the LICS 2006, pp. 349–358 (2006)
9. Brihaye, T., Da Costa Lopes, A., Laroussinie, F., Markey, N.: ATL with strategy contexts and bounded memory. In: International Symposium on Logical Foundations of Computer Science (LFCS), Lecture Notes in Computer Science, vol. 5407, pp. 92–106 (2009)
10. Bulling, N., Jamroga, W.: What agents can probably enforce. Fundam. Inform. **93**(1–3), 81–96 (2009)
11. Canny, J.: Some algebraic and geometric computations in PSPACE. In: Proceedings of the STOC'88, pp. 460–467 (1988)
12. Chatterjee, K., Henzinger, T.A., Piterman, N.: Strategy logic. Inf. Comput. 208(6): 677–693 (2010)
13. Clarke, E.M., Grumberg, O., Kurshan, R.P.: A synthesis of two approaches for verifying finite state concurrent systems. J. Log. Comput. **2**(5), 605–618 (1992)
14. Clarke, E.M., Peled, D., Grumberg, O.: Model Checking. MIT Press, Cambridge (1999)
15. Cook, S.A.: The complexity of theorem-proving procedures. In: IEEE Symposium on Foundations of Computer Science (FOCS), pp. 151–158 (1971)
16. Courcoubetis, C., Yannakakis, M.: The complexity of probabilistic verification. J. ACM **42**(4), 857–907 (1995)
17. De Alfaro, L.: Formal Verification of Probabilistic Systems. Ph.D.Thesis. Department of Computer Science, Stanford University (1997)
18. de Alfaro, L., Henzinger, T.A.: Concurrent omega-regular games. In: Proceedings of the LICS, pp. 141–154 (2000)
19. de Alfaro, L., Henzinger, T.A., Kupferman, O.: Concurrent reachability games. In: IEEE Symposium on Foundations of Computer Science (FOCS), pp. 564–575 (1998)
20. Grigoriev, D.: Complexity of deciding Tarski algebra. J. Symb. Comput. **5**(1–2), 65–108 (1988)
21. Hopcroft, J.E., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley, Reading (1979)
22. Kučera, A., Stražovský, O.: On the controller synthesis for finite-state Markov decision processes. In: Proceedings of the FSTTCS, Lecture Notes in Computer Science, vol. 3821, pp. 541–552 (2005)
23. Meyer, A.R., Stockmeyer, L.: The equivalence problem for regular expressions with squaring requires exponential space. In: Proceedings of the 13th Annual IEEE Symposium on Switching and Automata Theory, pp. 125–129 (1972)
24. Papadimitriou, C.: Computational Complexity. Addison-Wesley, Reading (1994)
25. Safra, S.: On the complexity of $\omega$-automata. In: 29th Annual IEEE Symposium on Foundations of Computer Science, pp. 319–327. White Plains, New York (1988)
26. Tarski, A.: A lattice-theoretical fixpoint theorem and its applications. Pac. J. Math. **5**(2), 285–309 (1955)
27. Thomas, W.: Computation tree logic and regular omega-languages. In: Lecture Notes in Computer Science, vol. 354, pp. 690–713 (1988)
28. Thomas, W.: Automata on infinite objects. In: van Leeuwen, J. (ed.) Handbook of Theoretical Computer Science, chapt. 4, pp. 133–191. Elsevier Science Publishers B.V., Amsterdam (1990)
29. Vardi, M.Y.: Probabilistic linear-time model checking: an overview of the automata-theoretic approach Lecture Notes in Computer Science, vol. 1601, pp. 265–276 (1999)
30. Vardi, M.Y., Wolper, P.:Yet another process logic (preliminary version). In: Lecture Notes in Computer Science, vol. 164, pp. 501–512 (1983)