Equilibria-based Probabilistic Model Checking for Concurrent Stochastic Games

Marta Kwiatkowska[1], Gethin Norman[2], David Parker[3], and Gabriel Santos[1]

[1] Department of Computing Science, University of Oxford, UK
[2] School of Computing Science, University of Glasgow, UK
[3] School of Computer Science, University of Birmingham, UK

Abstract. Probabilistic model checking for stochastic games enables formal verification of systems that comprise competing or collaborating entities operating in a stochastic environment. Despite good progress in the area, existing approaches focus on zero-sum goals and cannot reason about scenarios where entities are endowed with different objectives. In this paper, we propose probabilistic model checking techniques for concurrent stochastic games based on Nash equilibria. We extend the temporal logic rPATL (probabilistic alternating-time temporal logic with rewards) to allow reasoning about players with distinct quantitative goals, which capture either the probability of an event occurring or a reward measure. We present algorithms to synthesise strategies that are subgame perfect social welfare optimal Nash equilibria, i.e., where there is no incentive for any players to unilaterally change their strategy in any state of the game, whilst the combined probabilities or rewards are maximised. We implement our techniques in the PRISM-games tool and apply them to several case studies, including network protocols and robot navigation, showing the benefits compared to existing approaches.

1 Introduction

Probabilistic model checking is a technique for formally verifying systems that exhibit uncertainty or feature randomisation. Quantitative system requirements, which express, e.g., safety, reliability or performance, are formally specified in temporal logic. These are then automatically checked against a probabilistic model, such as a Markov chain, capturing the possible behaviour of the system being verified. Closely related is strategy synthesis, which uses probabilistic models with nondeterminism, for example Markov decision processes (MDPs), to automatically generate policies or controllers which guarantee that pre-specified system requirements are satisfied. Thanks to mature tool support [28,20], the methods have been successfully applied to many domains, from autonomous vehicles, to computer security, to task scheduling.

Stochastic games are a modelling formalism that incorporates probability, nondeterminism and multiple players, who can compete or collaborate to achieve their goals. A variety of verification algorithms for these models have been devised, e.g., [13,46,2,3,14]. More recently, probabilistic model checking and strategy synthesis techniques for stochastic games have been proposed [17,6,29,25]

and implemented in the PRISM-games tool [32]. This has allowed modelling and verification of stochastic games to be used for a variety of non-trivial applications, in which competitive or collaborative behaviour between entities is a crucial ingredient, including computer security and energy management.

Initial work in this direction focused on *turn-based* stochastic games (TSGs), where each state is controlled by a single player [17], and proposed the logic rPATL, an extension of the well known logic ATL [4]. The logic can specify that a coalition of players is able to achieve a quantitative objective regarding the probability of an event's occurrence or the expectation of a reward measure, regardless of the strategies of the other players. Recently [29], this was extended to *concurrent* stochastic games (CSGs), in which players make decisions simultaneously. This allows more realistic modelling of interactive agents operating concurrently. In another direction, *multi-objective* model checking of TSGs [18,6] enabled reasoning about coalitions aiming to satisfy a *Boolean combination* of objectives, regardless of the remaining players' behaviour.

A limitation of these approaches is that they focus on *zero-sum* properties, in which a coalition aims to satisfy some requirement or to optimise some objective, while the remaining players have the directly opposing goal. In this paper, we consider CSGs in which two coalitions of players have distinct quantitative objectives. For this, we use the notion of subgame perfect *Nash equilibria* [38], i.e., scenarios in which it is not beneficial for any player to unilaterally change their strategy in any state. Furthermore, amongst these, we consider *social welfare* optimal equilibria, which maximise the sum of the objectives of the players.

We propose an extension to rPATL which allows reasoning about subgame perfect social welfare optimal Nash equilibria between two coalitions of players, with respect to probabilistic or reward objectives, expressed using a variety of temporal operators. We then give a model checking algorithm for the logic against CSGs which employs a combination of backwards induction (for finite-horizon operators) and value iteration (for infinite-horizon operators). A key ingredient of the computation is finding social welfare optimal Nash equilibria for bimatrix games, which we perform using labelled polytopes [33] and a reduction to SMT. We implement our techniques as an extension of the PRISM-games [32] model checker and develop a selection of case studies, including robot navigation, communication protocols and power control, to evaluate its performance and applicability. We show that we are able to synthesise strategies that outperform those derived using existing techniques.

Related Work. Game-theoretic models are used in many contexts within verification, as summarised above. In addition, the existence of and the complexity of finding Nash equilibria for stochastic games are studied in [16,46], but without practical algorithms. In [41], a learning-based algorithm for finding Nash equilibria for discounted properties of CSGs is presented and evaluated. Similarly, [34] studies Nash equilibria for discounted properties and introduces iterative algorithms for strategy synthesis. A theoretical framework for price-taking equilibria of CSGs is given in [5], where players try to minimise their costs which include a price common to all players and dependent on the decisions of all players. A no-

tion of strong Nash equilibria for a restricted class of CSGs is formalised in [21] and an approximation algorithm for checking the existence of such equilibria for discounted properties is introduced and evaluated. We also mention [9], which studies the existence of stochastic equilibria with imprecise deviations for CSGs and proposes a PSPACE algorithm to compute such equilibria.

For non-stochastic games, model checking tools such as PRALINE [10], EAGLE [45] and EVE [24] support Nash equilibria, as does MCMAS-SLK [11] via strategy logic. General purpose tools such as Gambit [35] can compute a variety of equilibria but, again, not for stochastic games.

2 Preliminaries

We first provide some background material on game theory and stochastic games. We let $Dist(X)$ denote the set of probability distributions over set X.

Definition 1 (Normal form game). *A (finite, n-person)* normal form game *(NFG) is a tuple* $\mathsf{N} = (N, A, u)$ *where:* $N = \{1, \ldots, n\}$ *is a finite set of players;* $A = A_1 \times \cdots \times A_n$ *and* A_i *is a finite set of actions available to player* $i \in N$*;* $u = (u_1, \ldots, u_n)$ *and* $u_i \colon A \to \mathbb{R}$ *is a utility function for player* $i \in N$*.*

For an NFG N, the players choose actions at the same time, where the choice for player $i \in N$ is over the action set A_i. When each player i choose a_i, the utility received by player j equals $u_j(a_1, \ldots, a_n)$. A (mixed) strategy σ_i for player i is a distribution over its action set. A *strategy profile* $\sigma = (\sigma_1, \ldots, \sigma_n)$ is a tuple of strategies for each player and the expected utility of player i under σ is:

$$u_i(\sigma) \stackrel{\text{def}}{=} \sum_{(a_1,\ldots,a_n)\in A} u_i(a_1, \ldots, a_n) \cdot \left(\prod_{j=1}^{n} \sigma_j(a_j) \right) .$$

For profile $\sigma = (\sigma_1, \ldots, \sigma_n)$ and player i strategy σ_i', we define the sequence $\sigma_{-i} = (\sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots, \sigma_n)$ and profile $\sigma_{-i}[\sigma_i'] = (\sigma_1, \ldots, \sigma_{i-1}, \sigma_i', \sigma_{i+1}, \ldots, \sigma_n)$. For player i and strategy sequence σ_{-i}, a *best response* for player i to σ_{-i} is a strategy $\sigma_i^\star$ for player i such that $u_i(\sigma_{-i}[\sigma_i^\star]) \geqslant u_i(\sigma_{-i}[\sigma_i])$ for all strategies σ_i of player i. We now introduce the concept of *Nash equilibria* and a particular variant called *social welfare optimal*, which are equilibria that maximise the total utility, i.e. maximise the sum of players' individual utilities.

Definition 2 (Nash equilibrium). *For NFG* N*, a strategy profile* $\sigma^\star$ *is a* Nash equilibrium *(NE) if* $\sigma_i^\star$ *is a best response to* $\sigma_{-i}^\star$ *for all* $i \in N$*. Furthermore* $\sigma^\star$ *is a* social welfare optimal NE *(SWNE) if* $u_1(\sigma^\star) + \cdots + u_n(\sigma^\star) \geqslant u_1(\sigma) + \cdots + u_n(\sigma)$ *for all Nash equilibria* σ *of* N*.*

A two-player NFG is *constant-sum* if there exists $c \in \mathbb{R}$ such that $u_1(\alpha) + u_2(\alpha) = c$ for all $\alpha \in A$ and *zero-sum* if $c = 0$. For general two-player NFGs, we have a *bimatrix game* which can be represented by two distinct matrices $\mathsf{Z}_1, \mathsf{Z}_2 \in \mathbb{R}^{l \times m}$ where $A_1 = \{a_1, \ldots, a_l\}$, $A_2 = \{b_1, \ldots, b_m\}$, $z_{ij}^1 = u_1(a_i, b_j)$ and $z_{ij}^2 = u_2(a_i, b_j)$.

Example 1. We consider a stag hunt game [39] where, if players decide to cooperate, this can yield a large payoff, but, if the others do not, then the

cooperating player gets nothing while the remaining players get a small payoff. A scenario with 3 players, where two form a coalition, yields a bimatrix game:

$$\mathsf{Z}_1 = \begin{array}{c} \\ a_0 \\ a_1 \end{array} \begin{array}{c} \begin{array}{ccc} b_0 & b_1 & b_2 \end{array} \\ \begin{pmatrix} 2 & 2 & 2 \\ 0 & 4 & 6 \end{pmatrix} \end{array} \qquad \mathsf{Z}_2 = \begin{array}{c} \\ a_0 \\ a_1 \end{array} \begin{array}{c} \begin{array}{ccc} b_0 & b_1 & b_2 \end{array} \\ \begin{pmatrix} 4 & 2 & 0 \\ 4 & 6 & 9 \end{pmatrix} \end{array}$$

where a_0 and a_1 represent player 1 not cooperating and cooperating respectively and b_i that i players in the coalition cooperate. There are three Nash equilibria:

- player 1 and the coalition select a_0 and b_0, respectively with utilities $(2, 4)$;
- player 1 selects a_0 and a_1 with probabilities $5/9$ and $4/9$ and the coalition selects b_0 and b_2 with probabilities $2/3$ and $1/3$ with utilities $(2, 4)$;
- player 1 and the coalition select a_1 and b_2 respectively with utilities $(6, 9)$.

For instance, in the first case, neither player 1 nor the coalition thinks the other will cooperate: the best they can do is act alone. The third is the only SWNE.

Concurrent stochastic games. In this paper, we use CSGs, in which players repeatedly make simultaneous (probabilistic) choices that update the game state.

Definition 3 (Concurrent stochastic game). *A* concurrent stochastic multi-player game *(CSG) is a tuple* $\mathsf{G} = (N, S, \bar{S}, A, \Delta, \delta, AP, L)$ *where:*

- $N = \{1, \ldots, n\}$ *is a finite set of players;*
- S *is a finite set of states and* $\bar{S} \subseteq S$ *is a set of initial states;*
- $A = (A_1 \cup \{\bot\}) \times \cdots \times (A_n \cup \{\bot\})$ *where* A_i *is a finite set of actions available to player* $i \in N$ *and* $\bot$ *is an idle action disjoint from the set* $\cup_{i=1}^{n} A_i$*;*
- $\Delta \colon S \to 2^{\cup_{i=1}^{n} A_i}$ *is an action assignment function;*
- $\delta \colon S \times A \to \mathit{Dist}(S)$ *is a probabilistic transition function;*
- AP *is a set of atomic propositions and* $L \colon S \to 2^{AP}$ *is a labelling function.*

A CSG G starts in an initial state $\bar{s} \in \bar{S}$ and, when in state s, each player $i \in N$ selects an action from its available actions $A_i(s)$ given by $\Delta(s) \cap A_i$ if this set is non-empty and $\{\bot\}$ otherwise. Supposing player i selects action a_i, the state of the game is updated according to the distribution $\delta(s, (a_1, \ldots, a_n))$. We augment CSGs with *reward structures* of the form $r = (r_A, r_S)$ where $r_A \colon S \times A \to \mathbb{R}_{\geqslant 0}$ is an action reward function and $r_S \colon S \to \mathbb{R}_{\geqslant 0}$ is a state reward function.

Definition 4 (End component). *An end component of a CSG* G *is a pair* (S', δ') *comprising a subset* $S' \subseteq S$ *of states and a partial probabilistic transition function* $\delta' \colon S' \times A \to \mathit{Dist}(S)$ *satisfying the following conditions:*

- (S', δ') *defines a sub-CSG of* G*, i.e., for all* $s' \in S'$ *and* $\alpha \in A$*, if* $\delta'(s', \alpha)$ *is defined, then* $\delta'(s', \alpha) = \delta(s', \alpha)$ *and* $\delta'(s', \alpha)(s) = 0$ *for all* $s \in S \setminus S'$*;*
- *the underlying graph of* (S', δ') *is strongly connected.*

It is non-terminal *if* $\delta(s, \alpha)(s') > 0$ *for some* $s \in S'$*,* $\alpha \in A$ *and* $s' \in S \setminus S'$*.*

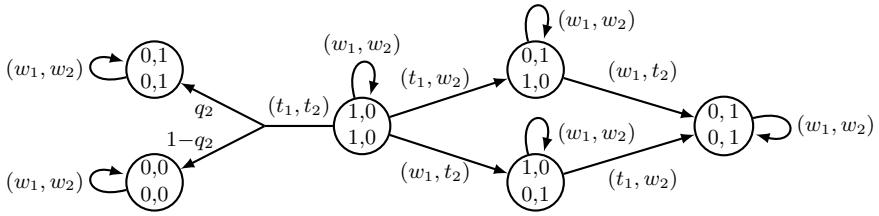

Fig. 1: CSG model of a medium access control problem.

A path of G represents a resolution of both the players' and probabilistic choices and is given by a sequence $\pi = s_0 \xrightarrow{\alpha_0} s_1 \xrightarrow{\alpha_1} \cdots$ such that $s_i \in S$, $\alpha_i = (a_1^i, \ldots, a_n^i) \in A$, $a_j^i \in A_j(s_i)$ for $j \in N$ and $\delta(s_i, \alpha_i)(s_{i+1}) > 0$ for all $i \geqslant 0$. For a path π, the $(i{+}1)$th state is denoted $\pi(i)$, the $(i{+}1)$th action $\pi[i]$, and if π is finite, the final state by $last(\pi)$. The sets of finite and infinite paths (starting in state s) are given by $FPaths_{\mathsf{G}}$ and $IPaths_{\mathsf{G}}$ ($FPaths_{\mathsf{G},s}$ and $IPaths_{\mathsf{G},s}$).

CSG strategies and equilibria. A *strategy* for player i in a CSG G resolves the player's choices. More precisely, it is a function $\sigma_i \colon FPaths_{\mathsf{G}} \to Dist(A_i \cup \{\bot\})$ such that if $\sigma_i(\pi)(a_i){>}0$, then $a_i \in A_i(last(\pi))$. We denote by Σ_{G}^i the set of strategies of player i.

As for NFGs, a *strategy profile* for G is a tuple $\sigma{=}(\sigma_1, \ldots, \sigma_n)$ of strategies for all players and, for a player i strategy σ_i', we define the sequence σ_{-i} and profile $\sigma_{-i}[\sigma_i']$ in the same way. For strategy profile σ and state s, we let $IPaths_{\mathsf{G},s}^{\sigma}$ denote the infinite paths from s under the choices of σ. We can define a probability measure $Prob_{\mathsf{G},s}^{\sigma}$ over the infinite paths $IPaths_{\mathsf{G},s}^{\sigma}$ [27] and, for random variable $X \colon IPaths_{\mathsf{G}} \to \mathbb{R}_{\geqslant 0}$, the expected value $\mathbb{E}_{\mathsf{G},s}^{\sigma}(X)$ of X in s with respect to σ.

An *objective* (or utility function) for player i of G is a random variable $X_i \colon IPaths_{\mathsf{G}} \to \mathbb{R}_{\geqslant 0}$. This can encode, e.g., the probability or expected cumulative reward for reaching a target. NE for CSGs can be defined as for NFGs. Since our model checking algorithm is based on backwards induction [43,36], we restrict attention to *sub-game perfect* NE [38], which are NE in *every state* of the CSG. In addition, for infinite-horizon objectives, the existence of NE is an open problem [8] so, for such objectives, we use ε-NE, which exist for any $\varepsilon{>}0$.

Definition 5 (Subgame perfect ε-NE). *For CSG G and $\varepsilon{>}0$, a strategy profile $\sigma^\star$ is a* subgame perfect ε-Nash equilibrium *for objectives $\langle X_i \rangle_{i \in N}$ if and only if $\mathbb{E}_{\mathsf{G},s}^{\sigma^\star}(X_i) \geqslant \sup_{\sigma_i \in \Sigma_i} \mathbb{E}_{\mathsf{G},s}^{\sigma^\star_{-i}[\sigma_i]}(X_i) - \varepsilon$ for all $i \in N$ and $s \in S$.*

Social welfare optimal variants of these equilibria (SWNEs and ε-SWNEs) are defined for CSGs as for NFGs above (see Definition 2).

Example 2. In [10] a deterministic concurrent game is used to model medium access control. Two users with limited energy share a wireless channel and choose to transmit (t) or wait (w) and, if both transmit, the transmissions fail due to interference. We extend this to a CSG by assuming that transmissions succeed with probability q_2 if both transmit. Figure 1 presents a CSG where each user has energy for one transmission (the first value of tuples labelling states represents if a user has energy and the second if it has successfully transmitted).

If the objectives are to maximise the probability of a successful transmission, there are two SWNEs when one user waits for the other to transmit and then transmits. This means both successfully transmit. If the objectives are to maximise the probability of being one of the first to transmit, then there is only one SWNE corresponding to both immediately trying to transmit.

3 Extending rPATL with Nash Formulae

We now extend the logic rPATL, previously proposed for zero-sum properties of both TSGs [29] and CSGs [29], to allow the analysis of equilibria-based properties. Since we are limited to considering ε-SWNE for infinite-horizon properties, we assume some ε has been fixed in advance when considering such properties.

Definition 6 (Extended rPATL syntax). *The syntax of our extended version of* rPATL *is given by the grammar:*

$$\begin{aligned}
\phi &:= \texttt{true} \mid \texttt{a} \mid \neg\phi \mid \phi \wedge \phi \mid \langle\!\langle C \rangle\!\rangle \texttt{P}_{\sim q}[\,\psi\,] \mid \langle\!\langle C \rangle\!\rangle \texttt{R}^{r}_{\sim x}[\,\rho\,] \mid \langle\!\langle C{:}C' \rangle\!\rangle_{\max \sim x}(\theta) \\
\theta &:= \texttt{P}[\,\psi\,]{+}\texttt{P}[\,\psi\,] \mid \texttt{R}^{r}[\,\rho\,]{+}\texttt{R}^{r}[\,\rho\,] \\
\psi &:= \texttt{X}\,\phi \mid \phi\ \texttt{U}^{\leqslant k}\ \phi \mid \phi\ \texttt{U}\ \phi \\
\rho &:= \texttt{I}^{=k} \mid \texttt{C}^{\leqslant k} \mid \texttt{F}\ \phi
\end{aligned}$$

where $\texttt{a}$ *is an atomic proposition,* C *and* C' *are coalitions of players such that* $C'{=}N\backslash C$, $\sim \in \{<, \leqslant, \geqslant, >\}$, $q \in [0,1]$, $x \in \mathbb{R}$, r *is a reward structure and* $k \in \mathbb{N}$.

The logic rPATL is a branching-time temporal logic that combines the probabilistic operator $\texttt{P}$ of PCTL [23], PRISM's reward operator $\texttt{R}$ [28], and the coalition operator $\langle\!\langle C \rangle\!\rangle$ of ATL [4]. The formula $\langle\!\langle C \rangle\!\rangle \texttt{P}_{\geqslant q}[\,\psi\,]$ states that the coalition C has strategies which, when followed, regardless of the strategies of $N\backslash C$, guarantee that the probability of satisfying path formula ψ is at least q. Such properties are inherently *zero-sum* in nature as one coalition tries to maximise an objective (here the probability of ψ) and the other to minimise it.

We extend rPATL with the ability to reason about *equilibria* through *Nash formulae* of the form $\langle\!\langle C{:}C' \rangle\!\rangle_{\max \sim x}(\theta)$. In addition to the usual state (ϕ), path (ψ) and reward (ρ) formulae, we distinguish *non-zero sum* formulae (θ), which comprise a sum of probability or reward objectives. The formula $\langle\!\langle C{:}C' \rangle\!\rangle_{\max \sim x}(\theta)$ is satisfied if there exists a subgame perfect SWNE strategy profile between coalitions C and $C'(=N\backslash C)$ under which the *sum* of the two objectives in θ is $\sim x$. As is common for probabilistic temporal logics, we allow numerical queries of the form $\langle\!\langle C{:}C' \rangle\!\rangle_{\max=?}[\theta]$ which return the sum of SWNE values.

For probabilistic objectives ($\theta{=}\texttt{P}[\,\psi^1\,]{+}\texttt{P}[\,\psi^2\,]$), each ψ^i can be a "next" ($\texttt{X}$), "bounded until" ($\texttt{U}^{\leqslant k}$) or "until" ($\texttt{U}$) operator, with the usual equivalences such as $\texttt{F}\ \phi \equiv \texttt{true}\ \texttt{U}\ \phi$. For reward objectives ($\theta{=}\texttt{R}^{r_1}[\,\rho^1\,]{+}\texttt{R}^{r_2}[\,\rho^2\,]$), each ρ^i refers to the expected reward with respect to reward structure r_i: the instantaneous reward after k steps ($\texttt{I}^{=k}$); the reward accumulated over k steps ($\texttt{C}^{\leqslant k}$); or the reward accumulated until a state satisfying ϕ is reached ($\texttt{F}\ \phi$).

Example 3. Recall the medium access control CSG of Example 2. Formula $\langle\!\langle p_1{:}p_2 \rangle\!\rangle_{\max \geqslant 2}(\mathtt{P}[\,\mathtt{F}\ \mathsf{send}_1\,]{+}\mathtt{P}[\,\mathtt{F}\ \mathsf{send}_2\,])$ means players p_1 and p_2 send their packets with probability 1, while $\langle\!\langle p_1{:}p_2 \rangle\!\rangle_{\max=?}(\mathtt{P}[\,\neg\mathsf{send}_2\ \mathtt{U}\ \mathsf{send}_1\,]{+}\mathtt{P}[\,\neg\mathsf{send}_1\ \mathtt{U}\ \mathsf{send}_2\,])$ asks what is the sum of subgame perfect SWNE values when the objectives are to maximise the probability of being one of the first to successfully transmit.

Before we give the semantics, we define *coalition games* which, given a CSG and coalition (set of players), reduce the CSG to a two-player CSG. Without loss of generality we assume the coalition of players is of the form $C = \{1, \ldots, n'\}$.

Definition 7 (Coalition game). *For CSG* $\mathsf{G}{=}(N, S, \bar{s}, A, \Delta, \delta, AP, L)$ *and coalition* $C{=}\{1, \ldots, n'\} \subseteq N$*, the* coalition game $\mathsf{G}^C{=}(\{1,2\}, S, \bar{s}, A^C, \Delta, \delta^C, AP, L)$ *is a two-player game where:* $A^C = A_1^C \times A_2^C$*,* $A_1^C = (A_1 \cup \{\bot\}) \times \cdots \times (A_{n'} \cup \{\bot\})$*,* $A_2^C = (A_{n'+1} \cup \{\bot\}) \times \cdots \times (A_n \cup \{\bot\})$ *and for any* $s \in S$*,* $a_1^C = (a_1, \ldots, a_{n'}) \in A_1^C$ *and* $a_2^C = (a_{n'+1}, \ldots, a_n) \in A_2^C$ *we have* $\delta^C(s, (a_1^C, a_2^C)) = \delta(s, (a_1, \ldots, a_n))$.

Furthermore, for a reward structure r *of* G*, by abuse of notation we use* r *for the corresponding reward structure of* G^C *which is constructed similarly.*

Definition 8 (Extended rPATL semantics). *The satisfaction relation* $\models$ *of our rPATL extension is defined inductively on the structure of the formula. The propositional logic fragment* ($\mathtt{true}$, a, $\neg$, $\wedge$) *is defined in the usual way. For temporal operators and a state* $s \in S$ *in CSG* G*, we have:*

$$s \models \langle\!\langle C \rangle\!\rangle \mathtt{P}_{\sim q}[\,\psi\,] \;\Leftrightarrow\; \exists \sigma_1 \in \Sigma^1.\, \forall \sigma_2 \in \Sigma^2.\, Prob^{\sigma_1,\sigma_2}_{\mathsf{G}^C,s}\{\pi \in IPaths^{\sigma_1,\sigma_2}_{\mathsf{G}^C,s} \mid \pi \models \psi\} \sim q$$

$$s \models \langle\!\langle C \rangle\!\rangle \mathtt{R}^r_{\sim x}[\,\rho\,] \;\Leftrightarrow\; \exists \sigma_1 \in \Sigma^1.\, \forall \sigma_2 \in \Sigma^2.\, \mathbb{E}^{\sigma_1,\sigma_2}_{\mathsf{G}^C,s}[rew(r,\rho)] \sim x$$

$$s \models \langle\!\langle C{:}C' \rangle\!\rangle_{\max \sim x}(\theta) \;\Leftrightarrow\; \exists \sigma_1^\star \in \Sigma^1, \sigma_2^\star \in \Sigma^2.\, \left(\mathbb{E}^{\sigma_1^\star,\sigma_2^\star}_{\mathsf{G}^C,s}(X_1^\theta) + \mathbb{E}^{\sigma_1^\star,\sigma_2^\star}_{\mathsf{G}^C,s}(X_2^\theta)\right) \sim x$$

and $(\sigma_1^\star, \sigma_2^\star)$ *is a subgame perfect SWNE*[4] *for the objectives* (X_1^θ, X_2^θ) *in* G^C *where, for* $1 \leqslant i \leqslant 2$ *and* $\pi \in IPaths^{\sigma_1,\sigma_2}_{\mathsf{G}^C,s}$:

$$\begin{aligned}
X_i^{\mathtt{P}[\,\psi^1\,]+\mathtt{P}[\,\psi^2\,]}(\pi) &= 1 \text{ if } \pi \models \psi^i \text{ and } 0 \text{ otherwise} \\
X_i^{\mathtt{R}^{r_1}[\,\rho^1\,]+\mathtt{R}^{r_2}[\,\rho^2\,]}(\pi) &= rew(r_i, \rho^i)(\pi) \\
\pi \models \mathtt{X}\,\phi &\Leftrightarrow \pi(1) \models \phi \\
\pi \models \phi_1\ \mathtt{U}^{\leqslant k}\ \phi_2 &\Leftrightarrow \exists i \leqslant k.\, (\pi(i) \models \phi_2 \wedge \forall j < i.\, \pi(j) \models \phi_1) \\
\pi \models \phi_1\ \mathtt{U}\ \phi_2 &\Leftrightarrow \exists i \in \mathbb{N}.\, (\pi(i) \models \phi_2 \wedge \forall j < i.\, \pi(j) \models \phi_1) \\
rew(r, \mathtt{I}^{=k})(\pi) &= r_S(\pi(k)) \\
rew(r, \mathtt{C}^{\leqslant k})(\pi) &= \textstyle\sum_{i=0}^{k-1} \big(r_A(\pi(i), \pi[i]) + r_S(\pi(i))\big) \\
rew(r, \mathtt{F}\ \phi)(\pi) &= \begin{cases} \infty & \text{if } \forall j \in \mathbb{N}.\, \pi(j) \not\models \phi \\ \sum_{i=0}^{k_\phi} \big(r_A(\pi(i), \pi[i]) + r_S(\pi(i))\big) & \text{otherwise} \end{cases}
\end{aligned}$$

and $k_\phi = \min\{k{-}1 \mid \pi(k) \models \phi\}$.

[4] In the case of infinite-horizon properties, this is a subgame perfect ε-SWNE.

4 Model Checking CSGs against Nash Formulae

Since rPATL is a branching-time logic, the basic model checking algorithm works by recursively computing the set $Sat(\phi)$ of states satisfying formula ϕ over the structure of ϕ. So, to extend the existing rPATL model checking algorithm for CSGs [29] to the logic from Section 3, we need only consider Nash formulae $\langle\langle C{:}C'\rangle\rangle_{\max\sim x}(\theta)$. This requires computation of subgame perfect SWNE values of the objectives (X_1^θ, X_2^θ) and a comparison of their sum to the threshold x.

We first explain how we compute SWNE values in bimatrix games, then subgame perfect SWNE values for finite-horizon objectives and lastly approximate subgame perfect ε-SWNE values for infinite-horizon objectives. We also discuss how to synthesise SWNE profiles. Our algorithm requires the following assumption on CSGs, which can be checked using standard graph-based methods. Without this assumption the presented value iteration algorithms are not guaranteed to converge (for further details, see [30]).

Assumption 1 *For any infinite-horizon probabilistic properties, there are no non-terminal end components. For infinite-horizon reward properties, the targets are reached with probability 1 under all strategy profiles.*

Computing SWNE Values of Bimatrix Games. Finding Nash equilibria in bimatrix games is in the class of *linear complementarity* problems (LCPs). More precisely, a profile (σ_1, σ_2) is a Nash equilibrium of the bimatrix game $\mathsf{Z}_1, \mathsf{Z}_2 \in \mathbb{R}^{l\times m}$ where where $A_1{=}\{a_1, \ldots, a_l\}$, $A_2{=}\{b_1, \ldots, b_m\}$ if and only if there exists $u, v \in \mathbb{R}$ such that, for the column vectors $x \in \mathbb{R}^l$v and $y \in \mathbb{R}^m$ where $x_i{=}\sigma_1(a_i)$ and $y_j{=}\sigma_2(b_j)$ for $1{\leqslant}i{\leqslant}l$ and $1{\leqslant}j{\leqslant}m$, we have:

$$x^T(\mathbf{1}u - \mathsf{Z}_1 y) = 0, \quad y^T(\mathbf{1}v - \mathsf{Z}_2^T x) = 0, \quad \mathbf{1}u - \mathsf{Z}_1 y \geqslant \mathbf{0}, \quad \mathbf{1}v - \mathsf{Z}_2^T x \geqslant \mathbf{0}$$

and $\mathbf{0}$ and $\mathbf{1}$ are vectors or matrices with all components 0 and 1, respectively.

The Lemke-Howson algorithm [33] can be applied for finding Nash equilibria and is based on the method of *labelled polytopes* [37]. Other well-known methods include those based on *support enumeration* [40] and *regret minimisation* [42].

SWNE via Labelled Polytopes. Given a bimatrix game $\mathsf{Z}_1, \mathsf{Z}_2 \in \mathbb{R}^{l\times m}$, we denote the sets of deterministic strategies of players 1 and 2 by $I{=}\{1, \ldots, l\}$ and $M{=}\{1, \ldots, m\}$ and define $J{=}\{l{+}1, \ldots, l{+}m\}$ by mapping $j \in M$ to $l{+}j \in J$. A *label* is then defined as element of $I \cup J$. The sets of strategies for players 1 and 2 can be represented by:

$$X = \{x \in \mathbb{R}^l \mid \mathbf{1}x = 1 \ \wedge \ x \geqslant \mathbf{0}\} \quad \text{and} \quad Y = \{y \in \mathbb{R}^m \mid \mathbf{1}y = 1 \ \wedge \ y \geqslant \mathbf{0}\}\,.$$

The strategy set Y is then divided into regions $Y(i)$ and $Y(j)$ (polytopes) for $i \in I$ and $j \in J$ such that $Y(i)$ contains strategies for which the deterministic strategy i of player 1 is a best response and $Y(j)$ contain strategies which choose action j with probability zero:

$$Y(i) = \{y \in Y \mid \forall k \in I.\, \mathsf{Z}_1(i,:)y \geqslant \mathsf{Z}_1(k,:)y\} \ \text{ and } \ Y(j) = \{y = Y \mid y_{j-l} = 0\}$$

where $\mathsf{Z}_1(i,:)$ is the ith row vector of Z_1. A vector y is then said to have label k if $y \in Y(k)$, for $k \in I \cup J$. The strategy set X is divided analogously into regions $X(j)$ and $X(i)$ for $j \in J$ and $i \in I$ and a vector x has label k if $x \in X(k)$, for $k \in I \cup J$. A pair of vectors $(x,y) \in X \times Y$ is *completely labelled* if the union of the labels of x and y equals $I \cup J$. The Nash equilibria of the game equal the vector pairs that are completely labelled [33,44].

Once all completely labelled vector pairs have been computed, one can calculate the corresponding set of values through matrix-vector multiplication. The pairs that maximise the sum of values correspond to SWNE strategies. In case of multiple SWNEs, we choose the values that are maximal for the first player, unless both players can get equal payoff, in which case we choose these.

Computing Values of Nash Formulae. For a formula $\langle\langle C{:}C' \rangle\rangle_{\max\sim x}(\theta)$, if the objectives of the non-zero sum formula θ are both finite-horizon, we can use *backwards induction* [43,36] to compute (precise) subgame perfect SWNE values. Below, we give the cases for bounded probabilistic reachability and bounded cumulative reward objectives; the remaining cases can be found in [30]. If both of the objectives are infinite-horizon, we use *value iteration* [15] to approximate subgame perfect SWNE values. Since there is not necessarily a unique pair of such values, the convergence criterion is applied to the sum of the two values computed, which *is* unique. Below, we give details for probabilistic and expected reachability objectives; the remaining cases can be found in [30]. Finally, for cases where there is a combination of finite- and infinite-horizon objectives, we convert to having both infinite-horizon by modifying the game and formula in a standard manner for probabilistic model checking; see [30] for the construction. The two key aspects of the value iteration algorithm are using SWNE to ensure uniqueness and solving an MDP when the target of one player has been reached.

We use the notation $\mathtt{V}_{\mathsf{G}^C}(s,\theta)$ for SWNE values of the objectives (X_1^θ, X_2^θ) in state s of G^C. We also use $\mathtt{P}^{\max}_{\mathsf{G},s}(\psi)$ and $\mathtt{R}^{\max}_{\mathsf{G},s}(r,\rho)$ for the maximum probability of satisfying ψ and maximum expected reward for the random variable $rew(r,\rho)$, respectively, in state s when all players collaborate. These can be computed through standard MDP model checking [7,1].

Bounded Probabilistic Reachability. If $\theta{=}\mathtt{P}[\,\mathtt{F}^{\leqslant k_1}\ \phi^1\,]{+}\mathtt{P}[\,\mathtt{F}^{\leqslant k_2}\ \phi^2\,]$, then we compute values of the objectives for the formulae $\theta_{n+n_1,n+n_2} = \mathtt{P}[\,\mathtt{F}^{\leqslant n+n_1}\ \phi^1\,] + \mathtt{P}[\,\mathtt{F}^{\leqslant n+n_2}\ \phi^2\,]$ for $0{\leqslant}n{\leqslant}k$ recursively, where $k{=}\min\{k_1,k_2\}$, $n_1{=}k_1{-}k$ and $n_2 = k_2{-}k$. For state s, if $n{=}0$:

$$\mathtt{V}_{\mathsf{G}^C}(s,\theta_{n_1,n_2}) = \begin{cases} (\eta_{\phi^1}(s), \eta_{\phi^2}(s)) & \text{if } n_1{=}n_2{=}0 \\ (\eta_{\phi^1}(s), \mathtt{P}^{\max}_{\mathsf{G},s}(\mathtt{F}^{\leqslant n_2}\ \phi^2) & \text{else if } n_1{=}0 \\ (\mathtt{P}^{\max}_{\mathsf{G},s}(\mathtt{F}^{\leqslant n_1}\ \phi^1), \eta_{\phi^2}(s)) & \text{otherwise} \end{cases}$$

and if $n{>}0$:

$$\mathtt{V}_{\mathsf{G}^C}(s,\theta_{n+n_1,n+n_2}) = \begin{cases} (1,1) & \text{if } s \in \mathit{Sat}(\phi^1) \cap \mathit{Sat}(\phi^2) \\ (1, \mathtt{P}^{\max}_{\mathsf{G},s}(\mathtt{F}^{\leqslant n+n_2}\ \phi^2)) & \text{else if } s \in \mathit{Sat}(\phi^1) \\ (\mathtt{P}^{\max}_{\mathsf{G},s}(\mathtt{F}^{\leqslant n+n_1}\ \phi^1), 1) & \text{else if } s \in \mathit{Sat}(\phi^2) \\ \mathit{val}(\mathsf{Z}_1,\mathsf{Z}_2) & \text{otherwise} \end{cases}$$

where $\eta_{\phi^i}(s)$ equals 1 if $s \in Sat(\phi^i)$ and 0 otherwise for $1 \leqslant i \leqslant 2$, and $val(\mathsf{Z}_1, \mathsf{Z}_2)$ equals SWNE values of the bimatrix game $(\mathsf{Z}_1, \mathsf{Z}_2) \in \mathbb{R}^{l \times m}$:

$$z^l_{i,j} = \textstyle\sum_{s' \in S} \delta(s,(a_i,b_j))(s') \cdot v^{s',l}_{(n-1)+n_l}$$

$1 \leqslant l \leqslant 2$ and $(v^{s',1}_{(n-1)+n_1}, v^{s',2}_{(n-1)+n_2}) = \mathtt{V}_{\mathsf{G}^C}(s', \theta_{(n-1)+n_1,(n-1)+n_2})$ for all $s' \in S$.

Bounded Cumulative Rewards. If $\theta = \mathtt{R}^{r_1}[\,\mathtt{C}^{\leqslant k_1}\,] + \mathtt{R}^{r_2}[\,\mathtt{C}^{\leqslant k_2}\,]$, then we compute values of the objectives for the formulae $\theta_{n+n_1,n+n_2} = \mathtt{R}^{r_1}[\,\mathtt{C}^{\leqslant n+n_1}\,] + \mathtt{R}^{r_2}[\,\mathtt{C}^{\leqslant n+n_2}\,]$ for $0 \leqslant n \leqslant k$ recursively, where $k = \min\{k_1, k_2\}$, $n_1 = k_1 - k$ and $n_2 = k_2 - k$. For state s, if $n=0$:

$$\mathtt{V}_{\mathsf{G}^C}(s, \theta_{n_1,n_2}) = \begin{cases} (0,0) & \text{if } n_1 = n_2 = 0 \\ (0, \mathtt{R}^{\max}_{\mathsf{G},s}(r_2, \mathtt{C}^{\leqslant n_2})) & \text{else if } n_1 = 0 \\ (\mathtt{R}^{\max}_{\mathsf{G},s}(r_1, \mathtt{C}^{\leqslant n_1}), 0) & \text{otherwise} \end{cases}$$

and if $n>0$, then $\mathtt{V}_{\mathsf{G}^C}(s, \theta_{n+n_1,n+n_2})$ equals SWNE values of the bimatrix game $(\mathsf{Z}_1, \mathsf{Z}_2) \in \mathbb{R}^{l \times m}$:

$$z^l_{i,j} = r^l_S(s) + r^l_A(s,(a_i,b_j)) + \textstyle\sum_{s' \in S} \delta(s,(a_i,b_j))(s') \cdot v^{s',l}_{(n-1)+n_l}$$

$1 \leqslant l \leqslant 2$ and $(v^{s',1}_{(n-1)+n_1}, v^{s',2}_{(n-1)+n_2}) = \mathtt{V}_{\mathsf{G}^C}(s', \theta_{(n-1)+n_1,(n-1)+n_2})$ for all $s' \in S$.

Probabilistic Reachability. If $\theta = \mathtt{P}[\,\mathtt{F}\ \phi^1\,] + \mathtt{P}[\,\mathtt{F}\ \phi^2\,]$, values can be computed through value iteration as the limit $\mathtt{V}_{\mathsf{G}^C}(s,\theta) = \lim_{n\to\infty} \mathtt{V}_{\mathsf{G}^C}(s,\theta,n)$ where:

$$\mathtt{V}_{\mathsf{G}^C}(s,\theta,n) = \begin{cases} (1,1) & \text{if } s \in Sat(\phi^1) \cap Sat(\phi^2) \\ (1, \mathtt{P}^{\max}_{\mathsf{G},s}(\mathtt{F}\ \phi^2)) & \text{else if } s \in Sat(\phi^1) \\ (\mathtt{P}^{\max}_{\mathsf{G},s}(\mathtt{F}\ \phi^1), 1) & \text{else if } s \in Sat(\phi^2) \\ (0,0) & \text{else if } n=0 \\ val(\mathsf{Z}_1, \mathsf{Z}_2) & \text{otherwise} \end{cases}$$

where $val(\mathsf{Z}_1, \mathsf{Z}_2)$ equals SWNE values of the bimatrix game $(\mathsf{Z}_1, \mathsf{Z}_2) \in \mathbb{R}^{l \times m}$:

$$z^l_{i,j} = \textstyle\sum_{s' \in S} \delta(s,(a_i,b_j))(s') \cdot v^{s',l}_{n-1}$$

$1 \leqslant l \leqslant 2$ and $(v^{s',1}_{n-1}, v^{s',2}_{n-1}) = \mathtt{V}_{\mathsf{G}^C}(s', \theta, n-1)$ for all $s' \in S$.

Expected Reachability. If $\theta = \mathtt{R}^{r_1}[\,\mathtt{F}\ \phi^1\,] + \mathtt{R}^{r_2}[\,\mathtt{F}\ \phi^2\,]$, values can be computed through value iteration as the limit $\mathtt{V}_{\mathsf{G}^C}(s,\theta) = \lim_{n\to\infty} \mathtt{V}_{\mathsf{G}^C}(s,\theta,n)$ where:

$$\mathtt{V}_{\mathsf{G}^C}(s,\theta,n) = \begin{cases} (0,0) & \text{if } s \in Sat(\phi^1) \cap Sat(\phi^2) \text{ or } n=0 \\ (0, \mathtt{R}^{\max}_{\mathsf{G},s}(r_2, \mathtt{F}\ \phi^2)) & \text{else if } s \in Sat(\phi^1) \\ (\mathtt{R}^{\max}_{\mathsf{G},s}(r_1, \mathtt{F}\ \phi^1), 0) & \text{else if } s \in Sat(\phi^2) \\ val(\mathsf{Z}_1, \mathsf{Z}_2) & \text{otherwise} \end{cases}$$

where $val(\mathsf{Z}_1, \mathsf{Z}_2)$ equals SWNE values of the bimatrix game $(\mathsf{Z}_1, \mathsf{Z}_2) \in \mathbb{R}^{l \times m}$:

$$z^l_{i,j} = r^l_S(s) + r^l_A(s,(a_i,b_j)) + \textstyle\sum_{s' \in S} \delta(s,(a_i,b_j))(s') \cdot v^{s',l}_{n-1}$$

$1 \leqslant l \leqslant 2$ and $(v_{n-1}^{s',1}, v_{n-1}^{s',2}) = \mathtt{V}_{\mathsf{G}^C}(s', \theta, n-1)$ for all $s' \in S$.

Strategy Synthesis. In addition to property verification, it is usually beneficial to perform *strategy synthesis*, that is, construct a witness of the satisfaction of a property. In the case of a formula $\langle\langle C{:}C' \rangle\rangle_{\max \sim x}(\theta)$, we can return a subgame perfect SWNE for the objectives (X_1^θ, X_2^θ). This is achieved using the approach above, both keeping track of a SWNE for the bimatrix game solved in each state and, when computing optimal values for MDPs, also performing strategy synthesis [31] (a strategy of the MDP is equivalent to a strategy profile of the CSG). We can then combine these generated profiles to yield a subgame perfect SWNE. The synthesised strategies require randomisation and memory. Memory is needed since choices change after a path formulae becomes true or a target reached and is required for finite-horizon properties. For infinite-horizon properties, the use of value iteration means only approximate ε-NE profiles are synthesised. However, for the case studies in Section 6, we find that all synthesised profiles are NE.

Correctness and Complexity. The proof of correctness is given in the extended version of the paper [30] and shows that the values computed during value iteration correspond to subgame perfect SWNE values of finite game trees, and the values of these game trees converge uniformly and are bounded from below and above by the finite approximations of G^C and actual values of G^C, respectively. A limitation of our approach, as for standard value iteration [22,26], is that convergence of the values does not give guarantees on the precision. Complexity is linear in the size of the formula, while finding NE for reachability objectives is EXPTIME [12]. Value iteration requires solving an LCP problem of size $|A|$ for each state at every iteration, with the number of iterations depending on the convergence criterion. Section 6 reports on efficiency in practice.

5 Implementation and Tool Support

We have extended PRISM-games [32] with support for modelling and verification of CSGs against equilibria-based properties, building upon the CSG extension of [29]. The tool and files for the case studies of Section 6 are available from [47].

Modelling. CSGs are specified using an extension of the PRISM modelling language, in which behaviour is defined using probabilistic guarded commands of the form $[a]\ g \to u$, where a is an action label, g is a guard (a predicate over states) and u is a probabilistic state update. If it is enabled (i.e., g is true), an a-labelled transition can probabilistically update the model's state.

This language is adapted to CSGs in [29] by assigning modules to players and, in any state, letting each player choose between enabled commands of the corresponding modules (if no command is enabled, the player idles). One requirement of [29] was that the updates of all player were independent of each other; we extend the language to remove this requirement, by allowing commands to be labelled with lists of actions $[a_1, \ldots, a_n]$, and thus represent behaviour dependent on other players' choices. Rewards are extended similarly so that an individual player's rewards can depend on the choices taken by multiple players.

Case study & property [parameters]	Param. values	CSG statistics				Constr. time(s)	Verif. time (s)	
		Players	States	Choices	Trans.		MDP	CSG
Aloha $\mathtt{P}[\mathtt{F}\ sent_1]+\mathtt{P}[\mathtt{F}\ sent_{2,3}]$ $[b_{max},D]$	2,8	3	17,057	19,713	42,654	0.6	0.6	21.4
	3,8	3	89,114	97,326	264,172	2.2	2.1	32.8
	4,8	3	449,766	474,898	1,655,479	10.9	10.6	49.9
	5,8	3	2,308,349	2,385,537	10,362,711	97.7	90.0	121.7
Robot coordination $\mathtt{P}[\mathtt{F}^{\leqslant k}goal_1]+\mathtt{P}[\mathtt{F}^{\leqslant k}goal_2]$ $[l,k]$	10,10	3	9,802	66,514	543,524	1.4	2.0	27.2
	15,15	3	50,177	375,549	3,175,539	5.0	19.8	131.8
	20,20	3	159,202	1,249,434	10,738,004	15.4	136.3	928.7
	25,25	3	389,377	3,142,669	27,267,419	48.3	548.8	4,837.0
Medium access control $\mathtt{R}^{r_1}[\mathtt{C}^{\leqslant k}]+\mathtt{R}^{r_2}[\mathtt{C}^{\leqslant k}]$ $[e_{max},k]$	10,20	2	441	1,600	2,759	0.1	-	17.2
	20,40	2	1,681	6,400	11,119	0.2	-	127.5
	40,80	2	6,561	25,600	44,639	0.7	-	991.7
	80,160	2	25,921	102,400	178,879	1.3	-	6,937.0
Power control $\mathtt{R}^{r_1}[\mathtt{F}\ e_1{=}0]+\mathtt{R}^{r_2}[\mathtt{F}\ e_2{=}0]$ $[e_{max},k]$	4,20	2	2,346	6,802	13,574	0.2	0.2	3.0
	4,40	2	10,746	30,700	60,854	0.4	1.0	12.7
	8,20	2	4,010	14,545	31,654	0.3	0.4	5.2
	8,40	2	32,812	119,694	260,924	1.2	3.9	64.8

Table 1: Statistics for a representative set of CSG verification instances.

Implementation. We have implemented model construction of CSGs for the language described above, and the model checking and strategy synthesis algorithms of Section 4, extending the PRISM-games implementation of rPATL verification [29]. We build on PRISM's Java-based 'explicit' engine which uses sparse matrices, and add an SMT-based implementation for solving bimatrix games using Z3 [19]. The set of all Nash equilibria for a bimatrix game are found by progressively querying the SMT solver for new profiles until the model becomes unsatisfiable. Structuring the problem using labelled polytopes, which can be expressed through conjunctions, disjunctions and linear inequalities, avoids non-linear arithmetic. As an optimisation, we also search for and filter out *dominated strategies* as a precomputation step to reduce the calls to the solver.

6 Case Studies and Experimental Results

We now present case studies and results to demonstrate the applicability of our approach and implementation, as well as the benefits of using equilibria.

Efficiency and Scalability. Before describing the case studies, we first discuss the performance of the implementation. In Table 1, we show experiments run on a 2.10 GHz Intel Xeon using 16GB RAM. The table includes model statistics (players, states and transitions) and the time to construct the CSG and verify it; the latter is split between CSG verification (including solving the bimatrix games) and the instances of MDP verification. Our tool can analyse models with over 2 million states and 20 million transitions; all are solved in under 2 hours and most are considerably quicker. However, for models where players have choices in almost all states, only models with up to tens of thousands of states can be verified within 2 hours. The majority of the time is spent solving bimatrix games, and therefore it is the number of choices of each coalition, rather than the number of players, that affects performance.

Investigating the Benefits of Equilibria Properties. In each case study, we compare our results with the corresponding zero-sum properties [29]. E.g., for

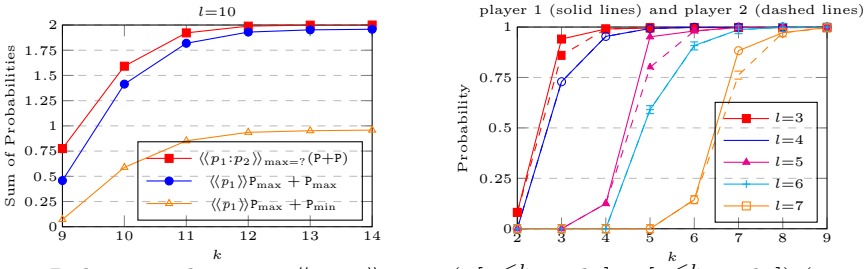

Fig. 2: Robot coordination: $\langle\langle p_1{:}p_2\rangle\rangle_{\max=?}(\mathtt{P}[\,\mathtt{F}^{\leqslant k}\ goal_1\,]{+}\mathtt{P}[\,\mathtt{F}^{\leqslant k}\ goal_2\,])$ (q=0.1)

$\langle\langle C{:}C'\rangle\rangle_{\max=?}(\mathtt{P}[\,\mathtt{F}\ \phi_1\,]{+}\mathtt{P}[\,\mathtt{F}\ \phi_2\,])$, we compute the value and an optimal strategy σ_C for coalition C of the formula $\langle\langle C\rangle\rangle\mathtt{P}_{\max=?}[\,\mathtt{F}\ \phi_1\,]$, and then find the value of an optimal strategy for the coalition C' for $\mathtt{P}_{\min=?}[\,\mathtt{F}\ \phi_2\,]$ and $\mathtt{P}_{\max=?}[\,\mathtt{F}\ \phi_2\,]$ in the MDP induced by CSG when C follows σ_C. The aim is to showcase the advantages of cooperation as, in many real-world applications, agents' goals are not strictly opposed. As will be seen, all the presented results demonstrate that by using equilibrium properties at least one of the players gains and in almost all cases neither player loses (in the one case study where this is not the case the gains far outweigh the losses). The individual SWNE values for players need not be unique and, for all case studies (except Aloha in which the players goals are not symmetric), the values can be swapped to give alternative SWNE values.

Robot Coordination. Our first case study models a scenario in which two robots move concurrently over a grid of size $l{\times}l$. The robots start in diagonally opposite corners and try to reach the corner from which the other starts. A robot can move either diagonally, horizontally or vertically towards its goal and when it moves there is a probability (q) that it instead moves in an adjacent direction. E.g., if a robot moves north east, then with probability $q/2$ it will move north or east. If the robots enter the same cell, they crash and are unable to move again.

We suppose the robots try to maximise the probability of reaching their individual goals eventually and within a given number of steps (k). If there is no bound and $l{\geqslant}4$, the SWNE strategies allow each robot to reach its goal with probability 1 (as time is not an issue, they can collaborate to avoid crashing). For the bounded case, in Figure 2 we have plotted both the sum of the probabilities for a grid of size 10 (left) and the probabilities of the individual players for different grid sizes (right) as k varies. When there is only one route to each goal within the bound (along the diagonal), i.e. when $k = l{-}1$, the SWNE strategies of both robots take this route. In odd grids, there is a high chance of crashing, but also a chance one will deviate and the other reaches their goal. Initially, as the bound k increases, for odd grids the SWNE values for the players are not equal (see Figure 2 right). Here, it is better overall for one to follow the diagonal and the other to take a longer route, as if both took the diagonal route, the chance of crashing increases, decreasing the chance of reaching their goals.

Aloha. This case study concerns three users trying to send packets using the slotted ALOHA protocol. In a time slot, if a single user tries to send a packet,

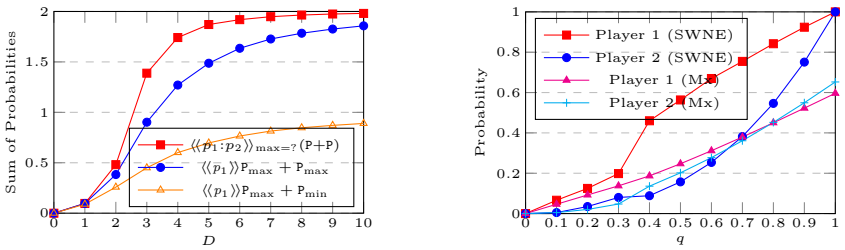

Fig. 3: Aloha: $\langle\!\langle p_1{:}\{p_2,p_3\}\rangle\!\rangle_{\max=?}(\mathtt{P}[\,\mathtt{F}\ (sent_1 \wedge t \leqslant D)\,]+\mathtt{P}[\,\mathtt{F}\ (sent_2 \wedge sent_3 \wedge t \leqslant D)\,])$

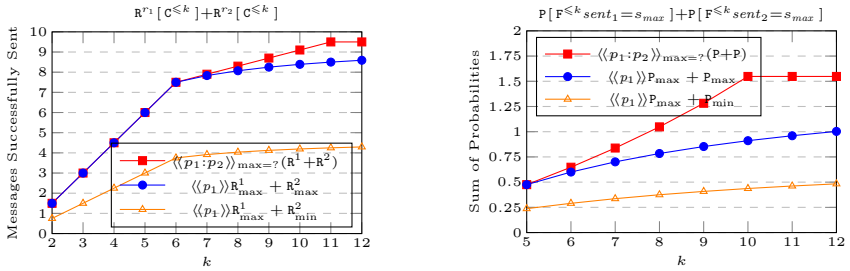

Fig. 4: Medium access control (e_{max}=5, s_{max}=5, q_1=0.9 and q_2=0.75)

there is a probability (q) that the packet is sent; as more users try and send, then the probability of success decreases. If sending a packet fails, the number of slots a user waits before resending is set according to an exponential backoff scheme. More precisely, each user maintains a backoff counter which it increases each time there is a failure (up to b_{max}) and, if the counter equals k, randomly chooses the slots to wait from $\{0, 1, \ldots, 2^k{-}1\}$.

We suppose three users try to maximise the probability of sending packets before a deadline D, with users 2 and 3 forming a coalition. Figure 3 presents total values as D varies (left) and individual values as q varies (right). Through synthesis, we find the collaboration is dependent on D and q. Given more time there is more chance for the users to collaborate sending in different slots, while if q is large it is unlikely users need to repeatedly send, so again can send in different slots. As the coalition has more messages to send, their probabilities are lower. However, even for two users, the probabilities are different, since, although it is advantageous to collaborate and only one user tries first, if transmission fails, then both users try to send as this is the best option for their individual goals.

Medium Access Control. Our third case study extends the CSG model from Example 2 by assuming the probability of a successful transmission when a single user tries to transmit equals q_1 and the energy of each user is bounded by e_{max}.

We consider two Nash properties for this model, both bounded by the number of time slots (k). The goal for each user in the first property is to maximise their expected number of successful transmissions and the second to maximise the probability of successfully transmitting a certain number (s_{max}) of messages. Figure 4 presents results for these properties as the bound k varies. For both properties, the SWNE strategies yield equal values for the players. Synthesising strategies we see that for small values of k there is not sufficient time to col-

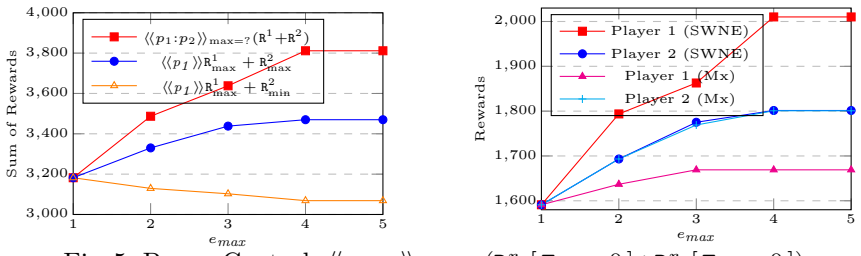

Fig. 5: Power Control: $\langle\!\langle p_1{:}p_2 \rangle\!\rangle_{\max=?}(\mathtt{R}^{r_1}[\mathtt{F}\ e_1{=}0]{+}\mathtt{R}^{r_2}[\mathtt{F}\ e_2{=}0])$

laborate (both users always try and transmit); however, as k increases there is time for the users to collaborate and try to transmit in different slots, and hence improve their values. Since the users have limited energy, Figure 4 shows that eventually adding steps does not increase the reward or probability.

Power Control. Our final case study is based a model of power control in cellular networks from [10]. In the model, phones emit signals over a cellular network and the signals can be strengthened by increasing the power level up to a bound (pow_{max}). A stronger signal can improve transmission quality, but uses more energy and lowers the quality of other transmissions due to interference. We extend this model by adding a failure probability (q_{fail}) when a power level is increased and assume each phone has a limited battery capacity (e_{max}). Based on [10], we associate a reward structure with each phone representing transmission quality dependent both on its power level and that of other phones due to interference.

We consider two players, each trying to maximise their reward before their battery is empty. Figure 5 presents, for p_{max}=5 and e_{max}=5, the sum of the SWNE values (left) and the values of the individual players (right) as the battery capacity varies. The values of the players are different because if one increases their power level this increases the overall reward (their reward increases, while the other's decreases by a lesser amount due to interference), whereas if both increase the overall reward decreases (both rewards decrease due to interference).

7 Conclusions

We have presented a logic, algorithms and tool for model checking and strategy synthesis of concurrent stochastic games using Nash equilibria-based properties. In comparison to existing methods, which support only zero-sum properties, we demonstrate, on a range of case studies, that our approach produces strategies that are collectively more beneficial for all players in the game. Future work will investigate other techniques for Nash equilibria synthesis, non-coalitional multi-player games and mechanism design.

Acknowledgements. This work is partially supported by the EPSRC Programme Grant on Mobile Autonomy and the PRINCESS project, under the DARPA BRASS programme. We would like to thank the reviewers of an earlier version of this paper for finding a flaw in the correctness proof.

References

1. de Alfaro, L.: Computing minimum and maximum reachability times in probabilistic systems. In: Proc. CONCUR'99, *LNCS*, vol. 1664, pp. 66–81. Springer (1999)
2. de Alfaro, L., Henzinger, T., Kupferman, O.: Concurrent reachability games. Theoretical Computer Science **386**(3), 188–217 (2007)
3. de Alfaro, L., Majumdar, R.: Quantitative solution of omega-regular games. Journal of Computer and System Sciences **68**(2), 374–397 (2004)
4. Alur, R., Henzinger, T.A., Kupferman, O.: Alternating-time temporal logic. Journal of the ACM **49**(5), 672–713 (2002)
5. Arslan, G., Yüksel, S.: Distributionally consistent price taking equilibria in stochastic dynamic games. In: Proc. CDC'17, pp. 4594–4599. IEEE (2017)
6. Basset, N., Kwiatkowska, M., Wiltsche, C.: Compositional strategy synthesis for stochastic games with multiple objectives. Information and Computation **261**(3), 536–587 (2018)
7. Bianco, A., de Alfaro, L.: Model checking of probabilistic and nondeterministic systems. In: Proc. FSTTCS'95, *LNCS*, vol. 1026, pp. 499–513. Springer (1995)
8. Bouyer, P., Markey, N., Stan, D.: Mixed Nash equilibria in concurrent games. In: Proc. FSTTCS'14, *LIPICS*, vol. 29, pp. 351–363. Leibniz-Zentrum für Informatik (2014)
9. Bouyer, P., Markey, N., Stan, D.: Stochastic equilibria under imprecise deviations in terminal-reward concurrent games. In: Proc. GandALF'16, *EPTCS*, vol. 226, pp. 61–75. Open Publishing Association (2016)
10. Brenguier, R.: PRALINE: A tool for computing Nash equilibria in concurrent games. In: Proc. CAV'13, *LNCS*, vol. 8044, pp. 890–895. Springer (2013)
11. Čermák, P., Lomuscio, A., Mogavero, F., Murano, A.: MCMAS-SLK: A model checker for the verification of strategy logic specifications. In: Proc. CAV'14, *LNCS*, vol. 8559, pp. 525–532. Springer (2014)
12. Chatterjee, K.: Nash equilibrium for upward-closed objectives. In: Proc. CSL'06, *LNCS*, vol. 4027, pp. 271–286. Springer (2006)
13. Chatterjee, K.: Stochastic ω-regular games. Ph.D. thesis, University of California at Berkeley (2007)
14. Chatterjee, K., de Alfaro, L., Henzinger, T.: Strategy improvement for concurrent reachability and turn-based stochastic safety games. Journal of Computer and System Sciences **79**(5), 640–657 (2013)
15. Chatterjee, K., Henzinger, T.: Value iteration. In: 25 Years of Model Checking, *LNCS*, vol. 5000, pp. 107–138. Springer (2008)
16. Chatterjee, K., Majumdar, R., Jurdziński, M.: On Nash equilibria in stochastic games. In: Proc. CSL'04, *LNCS*, vol. 3210, pp. 26–40. Springer (2004)
17. Chen, T., Forejt, V., Kwiatkowska, M., Parker, D., Simaitis, A.: Automatic verification of competitive stochastic systems. Formal Methods in System Design **43**(1), 61–92 (2013)
18. Chen, T., Forejt, V., Kwiatkowska, M., Simaitis, A., Wiltsche, C.: On stochastic games with multiple objectives. In: Proc. MFCS'13, *LNCS*, vol. 8087, pp. 266–277. Springer (2013)
19. De Moura, L., Bjørner, N.: Z3: An efficient SMT solver. In: Proc. TACAS'08, *LNCS*, vol. 4963, pp. 337–340. Springer (2008). github.com/Z3Prover/z3
20. Dehnert, C., Junges, S., Katoen, J.P., Volk, M.: A Storm is coming: A modern probabilistic model checker. In: Proc. CAV'17, *LNCS*, vol. 10427, pp. 592–600. Springer (2017)

21. Dileepa, F., Dong, N., Jegourel, C., Dong, J.S.: Verification of strong Nash-equilibrium for probabilistic BAR systems. In: Proc. ICFEM'18, *LNCS*, vol. 11232, pp. 106–123. Springer (2018)
22. Haddad, S., Monmege, B.: Interval iteration algorithm for MDPs and IMDPs. Theoretical Computer Science **735**, 111–131 (2018)
23. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing **6**(5), 512–535 (1994)
24. J. Gutierrez, J., Najib, M., Perelli, G., Wooldridge, M.: Eve: A tool for temporal equilibrium analysis. In: Proc. ATVA'18, *LNCS*, vol. 11138, pp. 551–557. Springer (2018)
25. Kelmendi, E., Krämer, J., Kretínský, J., Weininger, M.: Value iteration for simple stochastic games: Stopping criterion and learning algorithm. In: Proc. CAV'18, *LNCS*, vol. 10981, pp. 623–642. Springer (2018)
26. Kelmendi, E., Krämer, J., Křetínský, J., Weininger, M.: Value iteration for simple stochastic games: Stopping criterion and learning algorithm. In: Proc CAV'18, *LNCS*, vol. 10981, pp. 623–642. Springer (2018)
27. Kemeny, J., Snell, J., Knapp, A.: Denumerable Markov Chains. Springer (1976)
28. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Proc CAV'11, *LNCS*, vol. 6806, pp. 585–591. Springer (2011)
29. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Automated verification of concurrent stochastic games. In: Proc. QEST'18, *LNCS*, vol. 11024, pp. 223–239. Springer (2018)
30. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Equilibria-based probabilistic model checking for concurrent stochastic games (2018). arXiv:1811.07145
31. Kwiatkowska, M., Parker, D.: Automated verification and strategy synthesis for probabilistic systems. In: Proc. ATVA'13, *LNCS*, vol. 8172, pp. 5–22. Springer (2013)
32. Kwiatkowska, M., Parker, D., Wiltsche, C.: PRISM-games: Verification and strategy synthesis for stochastic multi-player games with multiple objectives. Software Tools for Technology Transfer **20**(2), 195–210 (2018)
33. Lemke, C., J. Howson, J.: Equilibrium points of bimatrix games. Journal of the Society for Industrial and Applied Mathematics **12**(2), 413–423 (1964)
34. Lozovanu, D., Pickl, S.: Determining Nash equilibria for stochastic positional games with discounted payoffs. In: Proc. ADT'17, *LNAI*, vol. 10576, pp. 339–343. Springer (2017)
35. McKelvey, R., McLennan, A., Turocy, T.: Gambit: Software tools for game theory, version 16.0.1 (2016). gambit-project.org
36. von Neumann, J., Morgenstern, O., Kuhn, H., Rubinstein, A.: Theory of Games and Economic Behavior. Princeton University Press (1944)
37. Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.: Algorithmic Game Theory. CUP (2007)
38. Osborne, M., Rubinstein, A.: An Introduction to Game Theory. Oxford University Press (2004)
39. Pacheco, J., Santos, F., Souza, M., Skyrms, B.: Evolutionary dynamics of collective action. In: The Mathematics of Darwin's Legacy, pp. 119–138. Springer (2011)
40. Porter, R., Nudelman, E., Shoham, Y.: Simple search methods for finding a Nash equilibrium. In: Proc. AAAI'04, pp. 664–669. AAAI Press (2004)
41. Prasad, H., Prashanth, L., Bhatnagar, S.: Two-timescale algorithms for learning Nash equilibria in general-sum stochastic games. In: Proc. AAMAS'15, pp. 1371–1379. IFAAMAS (2015)

42. Sandholm, T., Gilpin, A., Conitzer, V.: Mixed-integer programming methods for finding Nash equilibria. In: Proc. AAAI'05, pp. 495–501. AAAI Press (2005)
43. Schwalbe, U., Walker, P.: Zermelo and the early history of game theory. Games and Economic Behavior **34**(1), 123–137 (2001)
44. Shapley, L.: A note on the Lemke-Howson algorithm. In: Pivoting and Extension: In honor of A.W. Tucker, *Mathematical Programming Studies*, vol. 1, pp. 175–189. Springer (1974)
45. Toumi, A., Gutierrez, J., Wooldridge, M.: A tool for the automated verification of Nash equilibria in concurrent games. In: Proc. ICTAC'15, *LNCS*, vol. 9399, pp. 583–594. Springer (2015)
46. Ummels, M.: Stochastic multiplayer games: Theory and algorithms. Ph.D. thesis, RWTH Aachen University (2010)
47. Supporting material. prismmodelchecker.org/files/fm19nash/