Correlated Equilibria and Fairness in Concurrent Stochastic Games

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

[1] Department of Computer Science, University of Oxford, Oxford, UK
{marta.kwiatkowska,gabriel.santos}@cs.ox.ac.uk
[2] School of Computing Science, University of Glasgow, Glasgow, UK
gethin.norman@glasgow.ac.uk
[3] School of Computer Science, University of Birmingham, Birmingham, UK
d.a.parker@cs.bham.ac.uk

Abstract. Game-theoretic techniques and equilibria analysis facilitate the design and verification of competitive systems. While algorithmic complexity of equilibria computation has been extensively studied, practical implementation and application of game-theoretic methods is more recent. Tools such as PRISM-games support automated verification and synthesis of zero-sum and (ε-optimal subgame-perfect) social welfare Nash equilibria properties for concurrent stochastic games. However, these methods become inefficient as the number of agents grows and may also generate equilibria that yield significant variations in the outcomes for individual agents. We extend the functionality of PRISM-games to support *correlated equilibria*, in which players can coordinate through public signals, and introduce a novel optimality criterion of *social fairness*, which can be applied to both Nash and correlated equilibria. We show that correlated equilibria are easier to compute, are more equitable, and can also improve joint outcomes. We implement algorithms for both normal form games and the more complex case of multi-player concurrent stochastic games with temporal logic specifications. On a range of case studies, we demonstrate the benefits of our methods.

1 Introduction

Game-theoretic verification techniques can support the modelling and design of systems that comprise multiple agents operating in either a cooperative or competitive manner. In many cases, to effectively analyse these systems we also need to adopt a probabilistic approach to modelling, for example because agents operate in uncertain environments, use faulty hardware or unreliable communication mechanisms, or explicitly employ randomisation for coordination.

In these cases, *probabilistic model checking* provides a convenient unified framework for both formally modelling probabilistic multi-agent systems and specifying their required behaviour. In recent years, progress has been made in this direction for several models, including turn-based and concurrent stochastic games (TSGs and CSGs), and for multiple temporal logics, such as rPATL [10] and its extensions [24]. Tool support has been developed, in the form of PRISM-games [22], and successfully applied to case studies across a broad range of areas.

Initially, the focus was on *zero-sum* specifications [24], which can be natural for systems whose participants have directly opposing goals, such as the defender and attacker in a security protocol minimising or maximising the probability of a successful attack, respectively. However, agents often have objectives that are distinct but not directly opposing, and may also want to cooperate to achieve these objectives. Examples include network protocols and multi-robot systems.

For these purposes, *Nash equilibria* (NE) have also been integrated into probabilistic model checking of CSGs [24], together with *social welfare* (SW) optimality criterion, resulting in social welfare Nash equilibria (SWNE). An SWNE comprises a strategy for each player in the game where no player has an incentive to deviate unilaterally from their strategy and the sum of the individual objectives over all players is maximised.

One key limitation of SWNE, however, is that, as these techniques are extended to support larger numbers of players [21], the efficiency and scalability of synthesising SWNE is significantly reduced. In addition, simply aiming to maximise the sum of individual objectives may not produce the best performing equilibrium, either collectively or individually; for example, they can offer higher gains for specific players, reducing the incentive of the other players to collaborate and instead motivating them to deviate from the equilibrium.

In this paper, we adopt a different approach and introduce, for the first time within formal verification, both *social fairness* as an optimality criterion and *correlated equilibria*, and the insights required to make these usable in practical applications. Social fairness (SF) is particularly novel, as it is inspired by similar concepts used in economics and distinct from the fairness notions employed in verification. Correlated equilibria (CE) [3], in which players are able to coordinate through public *signals*, are easier to compute than NE and can yield better outcomes. Social fairness, which minimises the differences between the objectives of individual players, can be considered for both CE and NE.

We first investigate these concepts for the simpler case of normal form games, illustrating their differences and benefits. We then extend the approach to the more powerful modelling formalism of CSGs and extend the temporal logic rPATL to formally specify agent objectives. We present algorithms to synthesise equilibria, using linear programming to find CE and a combination of backwards induction or value iteration for CSGs. We implement our approach in the PRISM-games tool [22] and demonstrate significant gains in computation time and that quantifiably more fair and useful strategies can by synthesised for a range of application domains. An extended version of this paper, with the complete model checking algorithm, is available [23].

Related work. Nash equilibria have been considered for concurrent systems in [18], where a temporal logic is proposed whose key operator is a novel path quantifier which asserts that a property holds on all Nash equilibrium computations of the system. There is no stochasticity and correlated equilibria are not considered. In [2], a probabilistic logic that can express equilibria is formulated, along with complexity results, but no implementation has been provided.

The notion of fairness studied here is inspired by fairness of equilibria from economics [33,34] and aims to minimise the difference between the payoffs, as opposed to maximising the lowest payoff among the players in an NE [25]. Our notion of fairness can be thought of as a constraint applied to equilibria strategies, similar in style to social welfare, and used to select certain equilibria based on optimality. This is distinct from fairness used in verification of concurrent processes, where (strong) fairness refers to a property stating that, whenever a process is enabled infinitely often, it is executed infinitely often. This notion is typically defined as a constraint on infinite execution paths expressible in logics LTL and CTL* and needed to prove liveness properties. For probabilistic models, verification under fairness constraints has been formulated for Markov decision processes and the logic PCTL* [5,4]. For games on graphs, fairness conditions expressed as ω-regular winning conditions can be used to synthesise reactive processes [8]. Algorithms for strong transition fairness for ω-regular games have been recently studied in [6]. Both qualitative and quantitative approaches have been considered for verification under fairness constraints, but no equilibria.

2 Normal Form Games

We start by considering normal form games (NFGs), then define our equilibria concepts for these games, present algorithms and an implementation for computing them, and finally summarise some experimental results.

We first require the following notation. Let $\mathit{Dist}(X)$ denote the set of probability distributions over set X. For any vector $v \in \mathbb{R}^n$, we use $v(i)$ to refer to the ith entry of the vector. For any tuple $x = (x_1, \ldots, x_n) \in X^n$, element $x' \in X$ and $i \leqslant n$, we define the tuples $x_{-i} \stackrel{\text{def}}{=} (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ and $x_{-i}[x'] \stackrel{\text{def}}{=} (x_1, \ldots, x_{i-1}, x', x_{i+1}, \ldots, x_n)$.

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 $i \in N$.*

We fix an NFG $\mathsf{N} = (N, A, u)$ for the remainder of this section. In a play of N, each player $i \in N$ chooses an action from the set A_i at the same time. If each player i chooses a_i, then the utility received by player j equals $u_j(a_1, \ldots, a_n)$. We next define the *strategies* for players of N and *strategy profiles* comprising a strategy for each player. We also define *correlated profiles*, which allow the players to coordinate their choices through a (probabilistic) *public signal.*

Definition 2 (Strategy and profile). *A* strategy *σ_i for player i is an element of $\Sigma_i = \mathit{Dist}(A_i)$ and a* strategy profile *σ is an element of $\Sigma_{\mathsf{N}} = \Sigma_1 \times \cdots \times \Sigma_n$.*

For strategy σ_i of player i, the *support* is the set of actions $\{a_i \in A_i \mid \sigma_i(a_i) > 0\}$ and the support of a profile is the product of the supports of the strategies.

Definition 3 (Correlated profile). *A* correlated profile *is a tuple* (τ, ς) *comprising* $\tau \in \mathit{Dist}(D)$, *where* $D = D_1 \times \cdots \times D_n$, D_i *is a finite set of* signals *for player* i, *and* $\varsigma = (\varsigma_1, \ldots, \varsigma_n)$, *where* $\varsigma_i \colon D_i \to A_i$.

For a correlated profile (τ, ς), the public signal τ is a joint distribution over signals D_i for each player i such that, if player i receives the signal $d_i \in D_i$, then it chooses action $\varsigma_i(d_i)$. We can consider any correlated profile (τ, ς) as a *joint strategy*, i.e., a distribution over $A_1 \times \cdots \times A_n$ where:

$$(\tau, \varsigma)(a_1, \ldots, a_n) = \textstyle\sum\{\tau(d_1, \ldots, d_n) \mid d_i \in D_i \wedge \varsigma(d_i) = a_i \text{ for all } i \in N\}\,.$$

Conversely, any joint strategy $\tau \in \mathit{Dist}(A_1 \times \cdots \times A_n)$ can be considered as a correlated profile (τ, ς) where $D_i = A_i$ and ς_i is the identity function for $i \in N$.

Any strategy profile σ can be mapped to an equivalent correlated profile (in which τ is the joint distribution $\sigma_1 \times \cdots \times \sigma_n$ and ς_i is the identity function). On the other hand, there are correlated profiles with no equivalent strategy profile. Under profile σ and correlated profile (τ, ς) the expected utilities of player i are:

$$\begin{aligned} u_i(\sigma) &\stackrel{\text{def}}{=} \textstyle\sum_{(a_1,\ldots,a_n)\in A} u_i(a_1, \ldots, a_n) \cdot \left(\prod_{j=1}^n \sigma_j(a_j)\right) \\ u_i(\tau, \varsigma) &\stackrel{\text{def}}{=} \textstyle\sum_{(d_1,\ldots,d_n)\in D} \tau(d_1, \ldots, d_n) \cdot u_i(\varsigma_1(d_1), \ldots, \varsigma_n(d_n))\,. \end{aligned}$$

Example 1. Consider the two-player NFG where $A_i = \{a_1^i, a_2^i\}$ and a correlated profile corresponding to the joint distribution $\tau \in \mathit{Dist}(A_1 \times A_2)$ where $\tau(a_1^1, a_2^1) = \tau(a_1^2, a_2^2) = 0.5$. Under this correlated profile the players share a fair coin and both choose their first action if the coin is heads and their second action otherwise. This has no equivalent strategy profile. ■

Optimal equilibria of NFGs. We now introduce the notions of *Nash equilibrium* [27] and *correlated equilibrium* [3], as well as different definitions of optimality for these equilibria: *social welfare* and *social fairness*. Using the notation introduced above for tuples, for any profile σ and strategy $\sigma_i^\star$, the strategy tuple σ_{-i} corresponds to σ with the strategy of player i removed and $\sigma_{-i}[\sigma_i^\star]$ to the profile σ after replacing player i's strategy with $\sigma_i^\star$.

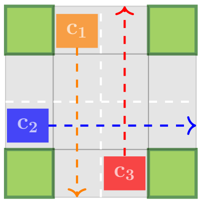
Definition 4 (Best response). *For a profile* σ *and correlated profile* (τ, ς), *a* best response *for player* i *to* σ_{-i} *and* (τ, ς_{-i}) *are, respectively:*

- *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* $\sigma_i \in \Sigma_i$;
- *a function* $\varsigma_i^\star \colon D_i \to A_i$ *for player* i *such that* $u_i(\tau, \varsigma_{-i}[\varsigma_i^\star]) \geqslant u_i(\tau, \varsigma_{-i}[\varsigma_i])$ *for all functions* $\varsigma_i \colon D_i \to A_i$.

Definition 5 (NE and CE). *A strategy profile* $\sigma^\star$ *is a* Nash equilibrium *(NE) and a correlated profile* $(\tau, \varsigma^\star)$ *is a* correlated equilibrium *(CE) if:*

- $\sigma_i^\star$ *is a best response to* $\sigma_{-i}^\star$ *for all* $i \in N$;
- $\varsigma_i^\star$ *is a best response to* $(\tau, \varsigma_{-i}^\star)$ *for all* $i \in N$;

respectively. We denote by Σ^N *and* Σ^C *the set of NE and CE, respectively.*



α	$u_1(\alpha)$	$u_2(\alpha)$	$u_3(\alpha)$
(pro_1, pro_2, pro_3)	-1000	-1000	-100
(pro_1, pro_2, yld_3)	-1000	-100	-5
(pro_1, yld_2, pro_3)	5	-5	5
(pro_1, yld_2, yld_3)	5	-5	-5
(yld_1, pro_2, pro_3)	-5	-1000	-100
(yld_1, pro_2, yld_3)	-5	5	-5
(yld_1, yld_2, pro_3)	-5	-5	5
(yld_1, yld_2, yld_3)	-10	-10	-10

Fig. 1: Example: Cars at an intersection and the corresponding NFG.

Any NE of N is also a CE, while there can exist CEs that cannot be represented by a strategy profile and therefore are not NEs. For each class of equilibria, NE and CE, we introduce two optimality criteria, the first maximising *social welfare* (SW), defined as the *sum* of the utilities, and the second maximising *social fairness* (SF), which minimises the *difference* between the players' utilities. Other variants of fairness have been considered for NE, such as in [25], where the authors seek to maximise the lowest utility among the players.

Definition 6 (SW and SF). *An equilibrium $\sigma^\star$ is a* social welfare *(SW) equilibrium if the sum of the utilities of the players under $\sigma^\star$ is maximal over all equilibria, while $\sigma^\star$ is a* social fair *(SF) equilibrium if the difference between the player's utilities under $\sigma^\star$ is minimised over all equilibria.*

We can also define the dual concept of *cost equilibria* [24], where players try to minimise, rather than maximise, their expected utilities by considering equilibria of the game $\mathsf{N}^- = (N, A, -u)$ in which the utilities of N are negated.

Example 2. Consider the scenario, based on an example from [32], where three cars meet at an intersection and want to proceed as indicated by the arrows in Figure 1. Each car can either *proceed* or *yield*. If two cars with intersecting paths proceed, then there is an accident. If an accident occurs, the car having the right of way, i.e., the other car is to its right, has a utility of -100 and the car that should yield has a utility of -1000. If a car proceeds without causing an accident, then its utility is 5 and the cars that yield have a utility of -5. If all cars yield, then, since this delays all cars, all have utility -10. The 3-player NFG is given in Figure 1. Considering the different optimal equilibria of the NFG:

– the SWNE and SWCE are the same: for c_2 to yield and c_1 and c_3 to proceed, with the expected utilities $(5, -5, 5)$;
– the SFNE is for c_1 to yield with probability 1, c_2 to yield with probability 0.863636 and c_3 to yield with probability 0.985148, with the expected utilities $(-9.254050, -9.925742, -9.318182)$;
– the SFCE gives a joint distribution where the probability of c_2 yielding and of c_1 and c_3 yielding are both 0.5 with the expected utilities $(0, 0, 0)$.

Modifying u_2 such that $u_2(pro_1, pro_2, pro_3) = -4.5$ to, e.g., represent a reckless driver, the SWNE becomes for c_1 and c_3 to yield and c_2 to proceed with the expected utilities $(-5, 5, -5)$, while the SWCE is still for c_2 to yield and c_1 and c_3 to proceed. The SFNE and SFCE also do not change. ■

Algorithms for computing equilibria. Before we give our algorithm to compute correlated equilibria, we briefly describe the approach of [21,24] for Nash equilibria computation that this paper builds upon. Finding NE in two-player NFGs is in the class of *linear complementarity* problems (LCPs) and we follow the algorithm presented in [24], which reduces the problem to SMT via labelled polytopes [28] by considering the regions of the strategy profile space, iteratively reducing the search space as positive probability assignments are found and added as restrictions on this space. To find SWNE and SFNE, we can enumerate all NE and then find the optimal NE.

When there are more than two players, computing NE values becomes a more complex task, as finding NE within a given support no longer reduces to a linear programming (LP) problem. In [21] we presented an algorithm using support enumeration [31], which exhaustively examines all sub-regions, i.e., supports, of the strategy profile space, one at a time, checking whether that sub-region contains NEs. For each support, finding SWNE can be reduced to a *nonlinear programming problem* [21]. This nonlinear programming problem can be modified to find SFNE in each support, similarly to how the LP problem for SWCEs is modified to find SFCEs below.

In the case of CE we can first find a joint strategy for the players, i.e., a distribution over the action tuples, which, as explained above, can then be mapped to a correlated profile. A SWCE can be found by solving the following LP problem. Maximise: $\sum_{i \in N} \sum_{\alpha \in A} u_i(\alpha) \cdot p_\alpha$ subject to:

$$\textstyle\sum_{\alpha_{-i} \in A_{-i}} (u_i(\alpha_{-i}[a_i]) - u_i(\alpha_{-i}[a'_i])) \cdot p_{\alpha_{-i}[a_i]} \geqslant 0 \tag{1}$$

$$0 \leqslant p_\alpha \leqslant 1 \tag{2}$$

$$\textstyle\sum_{\alpha \in A} p_\alpha = 1 \tag{3}$$

for all $i \in N$, $\alpha \in A$, $a_i, a'_i \in A_i$, $\alpha_{-i} \in A_{-i}$ where $A_{-i} \overset{\text{def}}{=} \{\alpha_{-i} \mid \alpha \in A\}$. The variables p_α represent the probability of the joint strategy corresponding to the correlated profile selecting the action-tuple α. The above LP has $|A|$ variables, one for each action-tuple, and $\sum_{i \in N}(|A_i|^2 - |A_i|) + |A| + 1$ constraints. Computation of SFCE can be reduced to the following optimisation problem. Minimise $p^{\max} - p^{\min}$ subject to: (1), (2) and (3) together with:

$$p^i = \textstyle\sum_{\alpha \in A} p_\alpha \cdot u_i(\alpha) \tag{4}$$

$$\left(\wedge_{m \in N} p^i \geqslant p^m\right) \rightarrow (p^{\max} = p^i) \tag{5}$$

$$\left(\wedge_{m \in N} p^i \leqslant p^m\right) \rightarrow (p^{\min} = p^i) \tag{6}$$

for all $i \in N$, $m \neq i$, $\alpha \in A$, $a_j, a_l \in A_i$, $\alpha_{-i} \in A_{-i}$. Again, the variables p_α in the program represent the probability of the players playing the joint action α. The constraint (4) requires p^i to equal the utility of player i. The constraints (5) and (6) set $p^{\max}$ and $p^{\min}$ as the maximum and minimum values within the utilities of the players, respectively. Given we use the constraints (1), (2) and (3), we start with the same number of variables and constraints as needed to compute SWCEs and incur an additional $|N|+2$ variables and $3 \cdot |N|$ constraints.

Game	Players	$\lvert A_i \rvert$	$\lvert A \rvert$	NE		CE	
				Supports	SW	SW	SF
Majority voting games	2	4	16	225	0.07	0.02	0.08
		6	36	3,969	0.1	0.02	0.1
		8	64	65,025	0.4	0.03	0.3
		10	100	1,046,529	5.8	0.07	0.7
	3	3	27	343	1.2	0.07	0.1
		4	81	3,375	25.8	0.08	0.3
Covariant games	3	3	27	343	8.7	0.08	1.7
		4	81	3,375	598.5	0.08	2.9
	8	2	256	6,561	TO	0.3	TO
		3	6,561	5,764,801	TO	22.8	TO
	10	2	1,024	59,049	TO	1.2	TO

Table 1: Times (s) for synthesis of equilibria in NFGs (timeout 30 mins).

Implementation. To find SWNE or SFNE of two-player NFGs, we adopt a similar approach to [24], using labelled polytopes to characterise and find NE values through a reduction to SMT in both Z3 [13] and Yices [14]. As an optimised precomputation step, when possible we also search for and filter out *dominated strategies*, which speeds up the computation and reduces solver calls.

For NFGs with more than two players, solving the nonlinear programming problem based on support enumeration has been implemented in [21] using a combination of the SMT solver Z3 [13] and the nonlinear optimisation suite Ipopt [38]. To mitigate the inefficiencies of an SMT solver for such problems, we used Z3 to filter out unsatisfiable support assignments with a timeout and then Ipopt is called to find SWNE values using an interior-point filter line-search algorithm [39]. To speed up the overall computation, the support assignments are analysed in parallel. Computing SFNE increases the complexity of the nonlinear program and, due to the inefficiency in this approach [21], we have not extended the implementation to compute SFNE.

As shown above, computing SWCE for NFGs reduces to solving an LP, and we implement this using either the optimisation solver Gurobi [17] or the SMT solver Z3 [13]. In the case of SFCE, the constraints (5) and (6) include implications, and therefore the problem does not reduce directly to an LP. When using Z3, we can encode these constraints directly as it supports assertions that combine inequalities with logical implications, a feature that linear solvers such as Gurobi do not have. Section 5 discusses implementing SFCE computation in Gurobi. Both solvers support the specification of *lower priority* or *soft* objectives, which makes it possible to have a consistent ordering for the players' payoffs in cases where multiple equilibria exist.

Efficiency and scalability. Table 1 presents experimental results for solving a selection of NFGs randomly generated with GAMUT [29], using Gurobi for SWCE and NE of two-player NFGs, Z3 for SFCE and both Ipopt and Z3 for NFGs of more than two players, and running on a 2.10GHz Intel Xeon Gold with 32GB of JVM memory. For each instance, Table 1 lists the number of players, actions for each player, joint actions and supports that need to be enumerated when finding NE, as well as the time to find SWNEs, SWCEs and SFCEs (the time for finding SFNEs of two-player games is the same as for SWNEs). As the results demonstrate, due to a simpler problem being solved and the fact that we do not need to enumerate the solutions, computing CEs scales far better than NEs as the number of players and actions increases. Finding NEs in games with more than two players is particularly hard as the constraints are nonlinear. We also see that SFCE computation is slower than SWCE, which is caused by the additional variables and constraints required when finding SFCE and using Z3 rather than Gurobi for the solver.

3 Concurrent Stochastic Games

We now further develop our approach to support concurrent stochastic games (CSGs) [36], in which players repeatedly make simultaneous action choices that cause the game's state to be updated probabilistically. We extend the previously introduced definitions of optimal equilibria to such games, focusing on subgame-perfect equilibria, which are equilibria in every state of a CSG. We then present algorithms to reason about and synthesise such equilibria.

Definition 7 (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:*

- $N = \{1, \dots, n\}$ *is a finite set of players;*
- S *is a finite set of states and* $\bar{S} \subseteq S$ *is a set of initial states;*
- $A = (A_1 \cup \{\bot\}) \times \cdots \times (A_n \cup \{\bot\})$ *and* 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 (partial) probabilistic transition function;*
- AP *is a set of atomic propositions and* $L \colon S \to 2^{AP}$ *is a labelling function.*

For the remainder of this section we fix a CSG G as in Definition 7. The game G starts in one of its initial states $\bar{s} \in \bar{S}$ and, supposing G is in a state s, then each player i of G chooses an action from the set that are available, defined as $A_i(s) \overset{\text{def}}{=} \Delta(s) \cap A_i$ if $\Delta(s) \cap A_i$ is non-empty and $A_i(s) \overset{\text{def}}{=} \{\bot\}$ otherwise. Supposing each player chooses a_i, then the game transitions to state s' with probability $\delta(s, (a_1, \dots, a_n))$. To enable quantitative analysis of G we augment it with *reward structures*, which are tuples $r{=}(r_A, r_S)$ of an action reward function $r_A \colon S{\times}A \to \mathbb{R}$ and state reward function $r_S \colon S \to \mathbb{R}$.

A *path* of G is a sequence $\pi = s_0 \xrightarrow{\alpha_0} s_1 \xrightarrow{\alpha_1} \cdots$ where $s_k \in S$, $\alpha_k = (a_1^k, \dots, a_n^k) \in A$, $a_i^k \in A_i(s_k)$ for $i \in N$ and $\delta(s_k, \alpha_k)(s_{k+1}) > 0$ for all $k \geqslant 0$. We denote by $\mathit{FPaths}_{\mathsf{G},s}$ and $\mathit{IPaths}_{\mathsf{G},s}$ the sets of finite and infinite paths starting in state s of G respectively and drop the subscript s when considering all finite and infinite paths of G. As for NFGs, we can define *strategies* of G that resolve the choices of the players. Here, a strategy for player i is a function $\sigma_i \colon \mathit{FPaths}_{\mathsf{G}} \to \mathit{Dist}(A_i \cup \{\bot\})$ such that, if $\sigma_i(\pi)(a_i){>}0$, then $a_i \in A_i(\mathit{last}(\pi))$ where $\mathit{last}(\pi)$ is the final state of π. Furthermore, we can define strategy profiles, correlated profiles and joint strategies analogously to Definitions 2 and 3.

The utility of a player i of G is defined by a random variable $X_i \colon \mathit{IPaths}_{\mathsf{G}} \to \mathbb{R}$ over infinite paths. For a profile[4] σ and state s, using standard techniques [20], we can construct a probability measure $\mathit{Prob}^{\sigma}_{\mathsf{G},s}$ over the paths with initial state s corresponding to σ, denoted $\mathit{IPaths}^{\sigma}_{\mathsf{G},s}$ and the expected value $\mathbb{E}^{\sigma}_{\mathsf{G},s}(X_i)$ of player i's utility from s under σ. Given utilities $X_1, \dots, X_n$ for all the players of G, we can then define NE and CE (see Definition 5) as well as the restricted classes of SW and SF equilibria as for NFGs (see Definition 6). Following [24,21], we focus on *subgame-perfect* equilibria [30], which are equilibria in *every state* of G.

Nonzero-sum properties. As in [24] (for two-player CSGs) and [21] (for n-player CSGs) we can specify equilibria-based properties using temporal logic. For simplicity, we restrict attention to nonzero-sum properties without nesting, allowing for the specification of NE and CE against either SW or SF optimality.

Definition 8 (Nonzero-sum specifications). *The syntax of nonzero-sum specifications θ for CSGs is given by the grammar:*

$$\begin{aligned}
\phi &\coloneqq \langle\!\langle \mathbb{C} \rangle\!\rangle (\star_1, \star_2)_{\mathrm{opt}\sim x}(\theta) \\
\theta &\coloneqq \mathtt{P}[\,\psi\,] + \cdots + \mathtt{P}[\,\psi\,] \;\mid\; \mathtt{R}^r[\,\rho\,] + \cdots + \mathtt{R}^r[\,\rho\,] \\
\psi &\coloneqq \mathtt{X}\,\mathtt{a} \;\mid\; \mathtt{a}\ \mathtt{U}^{\leqslant k}\ \mathtt{a} \;\mid\; \mathtt{a}\ \mathtt{U}\ \mathtt{a} \\
\rho &\coloneqq \mathtt{I}^{=k} \;\mid\; \mathtt{C}^{\leqslant k} \;\mid\; \mathtt{F}\ \mathtt{a}
\end{aligned}$$

where $\mathbb{C} = C_1{:}\cdots{:}C_m$, $C_1, \dots, C_m$ are coalitions of players such that $C_i \cap C_j = \varnothing$ for all $1 \leqslant i \neq j \leqslant m$ and $\cup_{i=1}^m C_i = N$, $(\star_1, \star_2) \in \{\textsc{ne}, \textsc{ce}\} \times \{\textsc{sw}, \textsc{sf}\}$, $\mathrm{opt} \in \{\min, \max\}$, $\sim \in \{<, \leqslant, \geqslant, >\}$, $x \in \mathbb{Q}$, r is a reward structure, $k \in \mathbb{N}$ and $\mathtt{a}$ is an atomic proposition.

The nonzero-sum formulae of Definition 8 extend the logic of in [24,21] in that we can now specify the type of equilibria, NE or CE, and optimality criteria, SW or SF. A probabilistic formula $\langle\!\langle C_1{:}\cdots{:}C_m \rangle\!\rangle (\star_1, \star_2)_{\max\sim x}(\mathtt{P}[\,\psi_1\,] + \cdots + \mathtt{P}[\,\psi_m\,])$ is true in a state if, when the players form the coalitions $C_1, \dots, C_m$, there is a subgame-perfect equilibrium of type $\star_1$ meeting the optimality criterion $\star_2$ for which the *sum* of the values of the objectives $\mathtt{P}[\,\psi_1\,], \dots, \mathtt{P}[\,\psi_m\,]$ for the coalitions $C_1, \dots, C_m$ satisfies $\sim x$. The objective ψ_i of coalition C_i is either a next ($\mathtt{X}\,\mathtt{a}$), bounded until ($\mathtt{a}_1\ \mathtt{U}^{\leqslant k}\ \mathtt{a}_2$) or until ($\mathtt{a}_1\ \mathtt{U}\ \mathtt{a}_2$) formula, with the usual equivalences, e.g., $\mathtt{F}\ \mathtt{a} \equiv \mathtt{true}\ \mathtt{U}\ \mathtt{a}$.

For a reward formula $\langle\!\langle C_1{:}\cdots{:}C_m \rangle\!\rangle (\star_1, \star_2)_{\mathrm{opt}\sim x}(\mathtt{R}^{r_1}[\,\rho_1\,] + \cdots + \mathtt{R}^{r_m}[\,\rho_m\,])$ the meaning is similar; however, here the objective of coalition C_i refers to a reward formula ρ_i with respect to reward structure r_i and this formula is either a bounded instantaneous reward ($\mathtt{I}^{=k}$), bounded accumulated reward ($\mathtt{C}^{\leqslant k}$) or reachability reward ($\mathtt{F}\ \mathtt{a}$).

For formulae of the form $\langle\!\langle C_1{:}\cdots{:}C_m \rangle\!\rangle (\star_1, \star_2)_{\min\sim x}(\theta)$, the dual notions of cost equilibria are considered. We also allow *numerical* queries of the form $\langle\!\langle C_1{:}\cdots{:}C_m \rangle\!\rangle (\star_1, \star_2)_{\mathrm{opt}=?}(\theta)$, which return the sum of the optimal subgame-perfect equilibrium's values.

[4] We can also construct such a probability measure and expected value given a correlated profile or joint strategy.

Model checking nonzero-sum specifications. Similarly to [24,21], to allow model checking of nonzero-sum properties we consider a restricted class of CSGs. We make the following assumption, which can be checked using graph algorithms with time complexity quadratic in the size of the state space [1].

Assumption 1. *For each subformula* $\mathtt{P}[\,\mathtt{a}_1 \ \mathtt{U}\ \mathtt{a}_2\,]$*, a state labelled* $\neg\mathtt{a}_1 \vee \mathtt{a}_2$ *is reached with probability 1 from all states under all strategy profiles and correlated profiles. For each subformula* $\mathtt{R}^r[\,\mathtt{F}\ \mathtt{a}\,]$*, a state labelled* $\mathtt{a}$ *is reached with probability 1 from all states under all strategy profiles and correlated profiles.*

We now show how to compute the optimal values of a nonzero-sum formula $\phi = \langle\langle C_1{:}\cdots{:}C_m\rangle\rangle(\star_1,\star_2)_{\mathrm{opt}\sim x}(\theta)$ when opt $=$ max. The case when opt $=$ min can be computed by negating all utilities and maximising.

The model checking algorithm broadly follows those presented in [24,21], with the differences described below. The problem is reduced to solving an m-player *coalition game* $\mathsf{G}^{\mathcal{C}}$ where $\mathcal{C} = \{C_1,\dots,C_m\}$ and the choices of each player i in $\mathsf{G}^{\mathcal{C}}$ correspond to the choices of the players in coalition C_i in G. Formally, we have the following definition in which, without loss of generality, we assume $\mathcal{C}$ is of the form $\{\{1,\dots,n_1\},\{n_1{+}1,\dots n_2\},\dots,\{n_{m-1}{+}1,\dots n_m\}\}$ and let $j_{\mathcal{C}}$ denote player j's position in its coalition.

Definition 9 (Coalition game). *For CSG* $\mathsf{G} = (N,S,\bar{S},A,\Delta,\delta,AP,L)$ *and partition* $\mathcal{C} = \{C_1,\dots,C_m\}$ *of the players into* m *coalitions, we define the* coalition game $\mathsf{G}^{\mathcal{C}} = (\{1,\dots,m\},S,\bar{S},A^{\mathcal{C}},\Delta^{\mathcal{C}},\delta^{\mathcal{C}},AP,L)$ *as an* m*-player CSG where:*

- $A^{\mathcal{C}} = (A^{\mathcal{C}}_1 \cup \{\bot\}) \times \cdots \times (A^{\mathcal{C}}_m \cup \{\bot\})$;
- $A^{\mathcal{C}}_i = \big(\prod_{j\in C_i}(A_j \cup \{\bot\}) \setminus \{(\bot,\dots,\bot)\}\big)$ *for all* $1 \leqslant i \leqslant m$;
- *for any* $s \in S$ *and* $1 \leqslant i \leqslant m$: $a^{\mathcal{C}}_i \in \Delta^{\mathcal{C}}(s)$ *if and only if either* $\Delta(s) \cap A_j = \varnothing$ *and* $a^{\mathcal{C}}_i(j_{\mathcal{C}}) = \bot$ *or* $a^{\mathcal{C}}_i(j_{\mathcal{C}}) \in \Delta(s)$ *for all* $j \in C_i$;
- *for any* $s \in S$ *and* $(a^{\mathcal{C}}_1,\dots,a^{\mathcal{C}}_m) \in A^{\mathcal{C}}$: $\delta^{\mathcal{C}}(s,(a^{\mathcal{C}}_1,\dots,a^{\mathcal{C}}_m)) = \delta(s,(a_1,\dots,a_n))$ *where for* $i \in M$ *and* $j \in C_i$ *if* $a^{\mathcal{C}}_i{=}\bot$*, then* $a_j{=}\bot$ *and otherwise* $a_j{=}a^{\mathcal{C}}_i(j_{\mathcal{C}})$.

If all the objectives in θ are finite-horizon, *backward induction* [35,27] can be applied to compute (precise) optimal equilibria values with respect to the criterion $\star_2$ and equilibria type $\star_1$. On the other hand, if all the objectives are infinite-horizon, *value iteration* [9] can be used to approximate optimal equilibria values and, when there is a combination of objectives, the game under study is modified in a standard manner to make all objectives infinite-horizon.

Backward induction and value iteration over the CSG $\mathsf{G}^{\mathcal{C}}$ both work by iteratively computing new values for each state s of $\mathsf{G}^{\mathcal{C}}$. The values for each state, in each iteration, are found by computing optimal equilibria values of an NFG N whose utility function is derived from the outgoing transition probabilities from s in the CSG and the values computed for successor states of s in the previous iteration. The difference here, with respect to [21], is that the NFGs are solved for the additional equilibria and optimality conditions considered in this paper, which we compute using the algorithms presented in Section 2.

Algorithm for probabilistic until. Because of space limitations, we only present here the details of value iteration for (unbounded) probabilistic until, i.e., for $\phi = \langle\langle C_1{:}\cdots{:}C_m\rangle\rangle(\star_1,\star_2)_{\max\sim x}(\theta)$ where $\theta = \mathtt{P}[\,\mathsf{a}_1^1 \;\mathtt{U}\; \mathsf{a}_2^1\,] + \cdots + \mathtt{P}[\,\mathsf{a}_1^m \;\mathtt{U}\; \mathsf{a}_2^m\,]$. The complete model checking algorithm can be found in [23].

Following [21], we use $\mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,\theta,n)$ to denote the vector of computed values, at iteration n, in state s of $\mathsf{G}^{\mathcal{C}}$ for optimality criterion $\star_2$ (SW or SF), equilibria type $\star_1$ (NE or CE) and (until) objectives θ. We also use $\mathbf{1}_m$ and $\mathbf{0}_m$ to denote a vector of size m whose entries all equal to 1 or 0, respectively. For any set of states S', atomic proposition a and state s we let $\eta_{S'}(s)$ equal 1 if $s \in S'$ and 0 otherwise, and $\eta_{\mathsf{a}}(s)$ equal 1 if $\mathsf{a} \in L(s)$ and 0 otherwise.

Each step of value iteration also keeps track of two sets $D, E \subseteq M$, where $M = \{1,\dots,m\}$ are the players of $\mathsf{G}^{\mathcal{C}}$. We use D for the subset of players that have already reached their goal (by satisfying a_2^i) and E for the players who can no longer can satisfy their goal (having reached a state that fails to satisfy a_1^i). It can then be ensured that their payoffs no longer change and are set to 1 or 0, respectively. In these cases, we effectively consider a modified game where, although the payoffs for these players are set, we still need to take their strategies into account in order to guarantee an optimal equilibrium.

Optimal values for all states s in the CSG $\mathsf{G}^{\mathcal{C}}$ can be computed as the following limit: $\mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,\theta) = \lim_{n\to\infty} \mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,\theta,n)$, where $\mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,\theta,n) = \mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,\varnothing,\varnothing,\theta,n)$ and, for any $D, E \subseteq M$ such that $D \cap E = \varnothing$:

$$\mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,D,E,\theta,n) = \begin{cases} (\eta_D(1),\dots,\eta_D(m)) & \text{if } D \cup E = M \\ (\eta_{\mathsf{a}_2^1}(s),\dots,\eta_{\mathsf{a}_2^m}(s)) & \text{else if } n = 0 \\ \mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,D\cup D',E,\theta,n) & \text{else if } D' \neq \varnothing \\ \mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s,\star_1,\star_2,D,E\cup E',\theta,n) & \text{else if } E' \neq \varnothing \\ \mathit{val}(\mathsf{N},\star_1,\star_2) & \text{otherwise} \end{cases}$$

where $D' = \{l \in M\backslash(D\cup E) \mid \mathsf{a}_2^l \in L(s)\}$, $E' = \{l \in M\backslash(D\cup E) \mid \mathsf{a}_1^l \notin L(s) \text{ and } s \in L(\mathsf{a}_2^l)\}$ and $\mathit{val}(\mathsf{N},\star_1,\star_2)$ equals optimal values of the NFG $\mathsf{N} = (M, A^{\mathcal{C}}, u)$ with respect to the criterion $\star_2$ and of equilibria type $\star_1$ in which for any $1\leqslant l\leqslant m$ and $\alpha \in A^{\mathcal{C}}$:

$$u_l(\alpha) = \begin{cases} 1 & \text{if } l \in D \\ 0 & \text{else if } l \in E \\ \sum_{s'\in S} \delta^{\mathcal{C}}(s,\alpha)(s') \cdot v_{n-1}^{s',l} & \text{otherwise} \end{cases}$$

and $(v_{n-1}^{s',1}, v_{n-1}^{s',2},\dots,v_{n-1}^{s',m}) = \mathtt{V}_{\mathsf{G}^{\mathcal{C}}}(s',\star_1,\star_2,D,E,\theta,n{-}1)$ for all $s' \in S$.

Since this paper considers equilibria for any number of coalitions (in particular, for more than two), the above follows the algorithm of [21] in the way that it keeps track of the coalitions that have satisfied their objective (D) or can no longer do so (E). By contrast the CSG algorithm of [24] was limited to two coalitions, which enabled the exploitation of efficient MDP analysis techniques for such coalitions. As explained in [21], in such a scenario we cannot reduce the analysis from an n-coalition game to an $(n-1)$-coalition game, as otherwise we would give one of the remaining coalitions additional power (the action choices of the coalition that has satisfied their objective or can no longer do so), which would therefore give this coalition an advantage over the other coalitions.

Strategy synthesis. As in [24,21] we can extend the model checking algorithm to perform *strategy synthesis*, generating a witness (i.e., a profile or joint strategy) representing the corresponding optimal equilibrium. This is achieved by storing the profile or joint strategy for the NFG solved in each state. Both the profiles and joint strategies require finite memory and are probabilistic. Memory is required as choices change after a path formula becomes true or a target is reached and to keep track of the step bound in finite-horizon properties. Randomisation is required for both NE and CE of NFGs.

Correctness and complexity. The correctness of the algorithm follows directly from [24,21], as changing the class of equilibria or optimality criterion does not change the proof. The complexity of the algorithm is linear in the formula size and value iteration requires finding optimal NE or CE for an NFG in each state of the model. Computing NEs of an NFG with two (or more) players is PPAD-complete [12,11], while finding optimal CEs of an NFG is in P [15].

4 Case Studies and Experimental Results

We have developed an implementation of our techniques for equilibria synthesis on CSGs, described above, building on top of the PRISM-games [22] model checker. Our implementation extends the tool's existing support for construction and analysis of CSGs, which is contained within its sparse matrix based "explicit" engine written in Java. We have considered a range of CSG case studies (supplementary material can be found at [40]). Below, we summarise the efficiency and scalability of our approach, again running on a 2.10GHz Intel Xeon Gold with 32GB JVM memory, and then describe our findings on individual case studies.

Efficiency and scalability. Table 2 summarises the performance of our implementation on the case studies that we have considered. It shows the statistics for each CSG, and the time taken to build it and perform equilibria synthesis, for several different variants (NE vs. CE, SW vs. SF). Comparing the efficiency of synthesising SWNE and SWCE, we see that the latter is typically much faster. For two-player NE, the social fairness variant is no more expensive to compute as we enumerate all NEs. For CE, which uses Z3 rather than Gurobi for finding SF, we note that, although Z3 is able to find optimal equilibria, it is not primarily developed as an optimisation suite, and therefore generally performs poorly in comparison with Gurobi. The benefits of the social fair equilibria, in terms of the values yielded for individual players, are discussed in the in-depth coverage of the different case studies below.

Aloha. In this case study, introduced in [24], a number of users try to send packets using the slotted Aloha protocol. We suppose that each user has one packet to send and, in a time slot, if k users try and send their packet, then the probability that each packet is successfully sent is q/k where $q \in [0,1]$. If a user fails to send a packet, then the number of slots it waits before resending the packet is set according to Aloha's exponential backoff scheme. The scheme requires that each user maintains a backoff counter, which it increases each time

Case study & property [parameters]	Players	$\star_1,\star_2$	Param. values	CSG statistics		Constr. time(s)	Verif. time (s)
				States	Trans.		
Aloha $(\star_1,\star_2)_{\min=?}(\mathtt{R}^{time}[\,\mathtt{F}\ \mathsf{s}_i\,])$ $[b_{max}, q]$	2	NE,SW	4,0.8	2,778	6,285	0.1	2.2
		CE,SW					2.1
		NE,SF					2.1
		CE,SF					23.3
	3	CE,SW	4,0.8	107,799	355,734	3.0	80.1
		CE,SF					114.6
	4	NE,SW	2,0.8	68,689	161,904	1.9	1042.9
		CE,SW					58.8
Aloha $(\star_1,\star_2)_{\max=?}(\mathtt{P}_{\max=?}[\,\mathtt{F}\ \mathsf{s}_i \wedge t \leqslant D\,])$ $[b_{max}, q, D]$	4	NE,SW	2,0.8,8	159,892	388,133	3.9	1027.5
		CE,SW					224.5
	5	CE,SW	2,0.8,8	1,797,742	5,236,655	54.5	4,936.8
		CE,SF					TO
Power control $(\star_1,\star_2)_{\max=?}(\mathtt{R}^{r}[\,\mathtt{F}\ \mathsf{e}_i\,])$ $[pow_{max}, e_{max}, q_{fail}]$	2	NE,SW	8,40,0.2	32,812	260,924	1.2	564.5
		NE,SF					566.3
		CE,SW					177.9
	3	CE,SW	5,15,0.2	42,156	740,758	3.5	147.0
		CE,SF					TO
Public good $(\star_1,\star_2)_{\max=?}(\mathtt{R}^{c}[\,\mathtt{I}^{=r_{max}}\,])$ $[f, r_{max}]$	3	NE,SW	2.5,3	16,202	35,884	0.8	27.5
		CE,SW					1.9
	4	NE,SW	3,3	391,961	923,401	13.0	71.9
		CE,SW					35.3
	5	CE,SW	4,2	59,294	118,342	3.1	5.2
Investors $(\star_1,\star_2)_{\max=?}(\mathtt{R}^{prof}[\,\mathtt{F}\ \mathsf{cin}_i\,])$ $[p_{bar}, months]$	2	CE,SW	0.2,8	71,731	315,804	2.4	47.5
		CE,SF					2,401.9
	3	CE,SW	0.2,5	83,081	462,920	3.6	79.3
		CE,SF					861.2

Table 2: Statistics for a set of CSG verification instances (timeout 2 hours).

there is a packet failure (up to $b_{\max}$) and, if the counter equals k and a failure occurs, randomly chooses the slots to wait from $\{0, 1, \ldots, 2^k - 1\}$.

We suppose that the objective of each user is to minimise the expected time to send their packet, which is represented by the nonzero-sum formula $\langle\langle usr_1{:}\cdots{:}usr_m\rangle\rangle(\star_1,\star_2)_{\min=?}(\mathtt{R}^{time}[\,\mathtt{F}\ \mathsf{s}_1\,]+\cdots+\mathtt{R}^{time}[\,\mathtt{F}\ \mathsf{s}_m\,])$. Synthesising optimal strategies for this specification, we find that the cases for SWNE and SWCE coincide (although SWCE returns a joint strategy for the players, this joint strategy can be separated to form a strategy profile). This profile requires one user to try and send first, and then for the remaining users to take turns to try and send afterwards. If a user fails to send, then they enter backoff and allow all remaining users to try and send before trying to send again. There is no gain to a user in trying to send at the same time as another, as this will increase the probability of a sending failure, and therefore the user having to spend time in backoff before getting to try again. For SFNE, which has only been implemented for the two-player case, the two users follow identical strategies, which involve randomly deciding whether to wait or transmit, unless they are the only user that has not transmitted, and then they always try to send when not in backoff. In the case of SFCE, users can employ a shared probabilistic signal to coordinate which user sends next. Initially, this is a uniform choice over the users, but as time progresses the signal favours the users with lower backoff counters as these users have had fewer opportunities to send their packet previously.

In Figure 2 we have plotted the optimal values for the players, where SW_i correspond to the optimal values (expected times to send their packets) for player

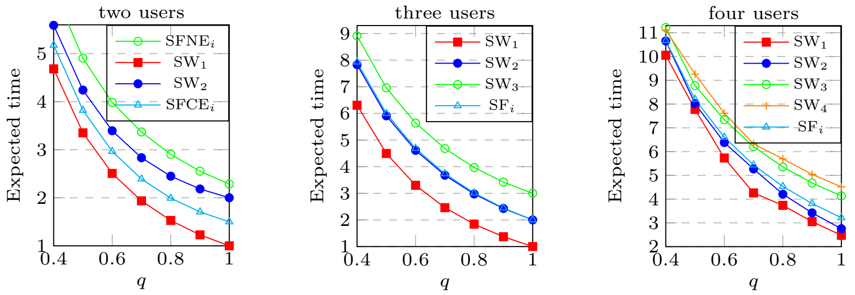


Fig. 2: Aloha: $\langle\langle usr_1{:}\cdots{:}usr_m\rangle\rangle(\star_1,\star_2)_{\min=?}(\mathtt{R}^{time}[\,\mathtt{F}\ \mathsf{s}_1\,]+\cdots+\mathtt{R}^{time}[\,\mathtt{F}\ \mathsf{s}_m\,])$

i for both SWNE and SWCE for the cases of two, three and four users. We see that the optimal values for the different users under SFNE and SFCE coincide, while under SWNE and SWCE they are different for each user (with the user sending first having the lowest and the user sending last the highest). Comparing the sum of the SWNE (and SWCE) values and that of the SFCE values, we see a small decrease in the sum of less than 2% of the total, while for SFNE there is a greater difference as the players cannot coordinate, and hence try and send at the same time.

Power control. This case study is based on a model of power control in cellular networks from [7]. In the network there are a number of users that each have a mobile phone. The phones emit signals that the users can strengthen by increasing the phone's power level up to a bound ($pow_{\max}$). A stronger signal can improve transmission quality, but uses more energy and lowers the quality of the transmissions of other phones due to interference. We use the extended model from [22], which adds a probability of failure (q_{fail}) when a power level is increased and assumes each phone has a limited battery capacity ($e_{\max}$). There is a reward structure associated with each phone representing transmission quality, which is dependent on both the phone's power level and the power levels of other phones due to interference. We consider the nonzero-sum property $\langle\langle p_1{:}\cdots{:}p_m\rangle\rangle(\star_1,\star_2)_{\max=?}(\mathtt{R}^{r_1}[\,\mathtt{F}\ \mathsf{e}_1\,]+\cdots+\mathtt{R}^{r_m}[\,\mathtt{F}\ \mathsf{e}_m\,])$, where each user tries to maximise their expected reward before their phone's battery is depleted.

In Figure 3 we have presented the expected rewards of the players under the synthesised SWCE and SFCE joint strategies. When performing strategy synthesis, in the case of two users the SWNE and SWCE yield the same profile in which, when the users' batteries are almost depleted, one user tries to increase their phone's power level and, if successful, in the next step, the second user then tries to increase their phone's power level. Since the first user's phone battery is depleted when the second tries to increase, this increase does not cause any interference. On the other hand, if the first user fails to increase their power level, then both users increase their battery levels. For the SFCE, the users can coordinate and flip a coin as to which user goes first: as demonstrated by Figure 3 this yields equal rewards for the users, unlike the SWCE. In the case of three users, the SWNE and SWCE differ (we were only able to synthesise SWNE for $pow_{\max} = 2$ as for larger values the computation had not completed within

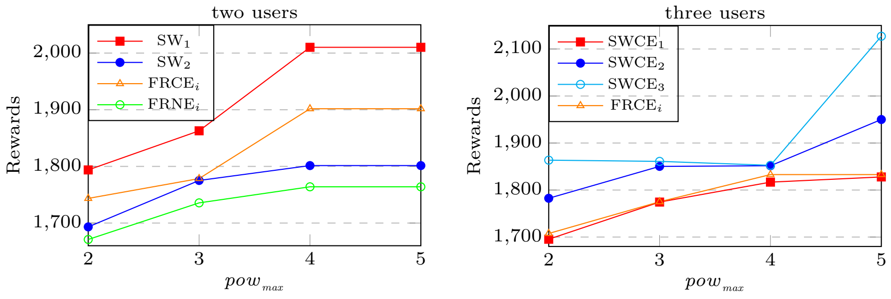


Fig. 3: Power control: $\langle\langle p_1{:}\cdots{:}p_m\rangle\rangle(\star_1,\star_2)_{\max=?}(\mathtt{R}^{r_1}[\,\mathtt{F}\ \mathtt{e}_1\,]+\cdots+\mathtt{R}^{r_m}[\,\mathtt{F}\ \mathtt{e}_m\,])$

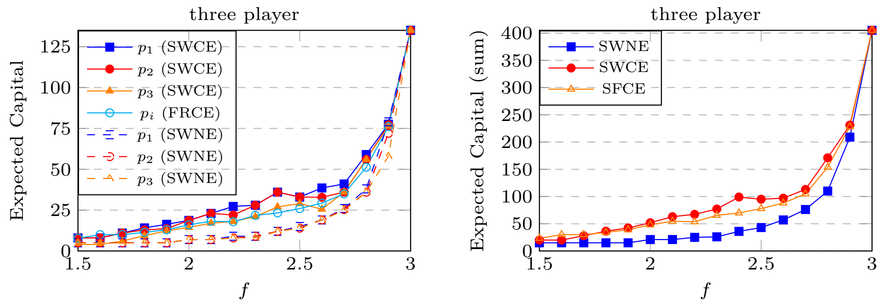


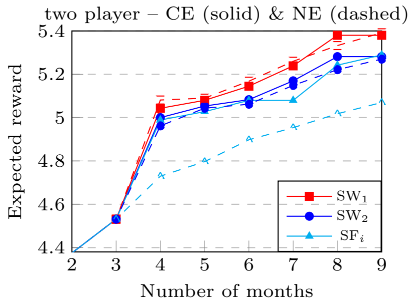
Fig. 4: Public good: $\langle\langle p_1{:}\cdots{:}p_m\rangle\rangle(\star_1,\star_2)_{\max=?}(\mathtt{R}^{c_1}[\,\mathtt{I}^{=r_{max}}\,]+\cdots+\mathtt{R}^{c_m}[\,\mathtt{I}^{=r_{max}}\,])$

the timeout), again users take turns to try and increase their phone's power level. However, here if the users are unsuccessful the SWCE can coordinate as to which user goes next trying to increase their phone's battery level. Through this coordination, the users' rewards can be increased as the battery level of at most one phone increases at a time, which limits interference. On the other hand, for the SWNE users must decide independently whether to increase their phone's battery level and they each randomly decide whether to do so or not.

Public good. We next consider a variant of a *public good* game [19], based on the one presented in [22] for the two-player case. In this game a number of players each receive an initial amount of capital (e_{init}) and, in each of r_{max} months, can invest none, half or all of their current capital. The total invested by the players in a month is multiplied by a factor f and distributed equally among the players before the start of the next month. The aim of the players is to maximise their expected capital which is represented by the formula: $\langle\langle p_1{:}\cdots{:}p_m\rangle\rangle(\star_1,\star_2)_{\max=?}(\mathtt{R}^{c_1}[\,\mathtt{I}^{=r_{max}}\,]+\cdots+\mathtt{R}^{c_m}[\,\mathtt{I}^{=r_{max}}\,])$.

Figure 4 plots, for the three-player model, both the expected capital of individual players and the total expected capital after three months for the SWNE, SWCE and SFNE as the parameter f varies. As the results demonstrate the players benefit, both as individuals and as a population, by coordinating through a correlated strategy. In addition, under the SFCE, all players receive the same expected capital with only a small decrease in the sum from that of the SWCE.

Investors. The final case study concerns a concurrent multi-player version of futures market investor model of [26], in which a number of investors (the players)



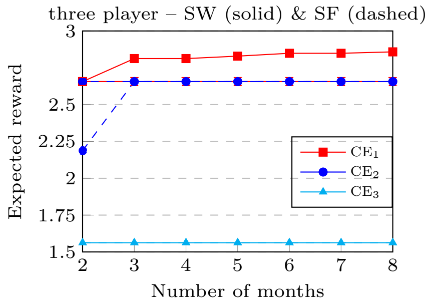


Fig. 5: Investors: $\langle\!\langle inv_1{:}\cdots{:}inv_m \rangle\!\rangle(\star_1,\star_2)_{\max=?}(\mathtt{R}^{pf_1}[\,\mathtt{F}\ \mathsf{cin}_1\,]{+}\cdots{+}\mathtt{R}^{pf_m}[\,\mathtt{F}\ \mathsf{cin}_m\,])$

interact with a probabilistic stock market. In successive months, the investors choose whether to invest, wait or cash in their shares, while at the same time the market decides with probability *pbar* to bar each investor, with the restriction that an investor cannot be barred two months in a row or in the first month, and then the values of shares and cap on values are updated probabilistically.

We consider both two- and three-player models, where each investor tries to maximise its individual profit represented by the following nonzero-sum property: $\langle\!\langle inv_1{:}\cdots{:}inv_m \rangle\!\rangle(\star_1,\star_2)_{\max=?}(\mathtt{R}^{pf_1}[\,\mathtt{F}\ \mathsf{cin}_1\,]{+}\cdots{+}\mathtt{R}^{pf_m}[\,\mathtt{F}\ \mathsf{cin}_m\,])$. In Figure 5 we have plotted the different optimal values for NE and CE of the two-player game and the different optimal values for CE of the three-player game (the computation of NE values timed out for the three player case). As the results demonstrate, again we see that the coordination that CEs offer can improve the returns of the players and that, although considering social fairness does decrease the returns of some players, this is limited, particularly for CEs.

5 Conclusions

We have presented novel techniques for game-theoretic verification of probabilistic multi-agent systems, focusing on correlated equilibria and a notion of social fairness. We began with the simpler case of normal form games and then extended this to concurrent stochastic games, and used temporal logic to formally specify equilibria. We proposed algorithms for equilibrium synthesis, implemented them and illustrated their benefits, in terms of efficiency and fairness, on case studies from a range of application domains.

Future work includes exploring the use of further game-theoretic topics within this area, such as techniques for mechanism design or other concepts such as Stackelberg equilibria. We plan to implement SFCE computation in Gurobi using the *big-M method* [16] to encode implications and techniques from [37] to encode conjunctions, which should yield a significant speed-up in their computation.

Acknowledgements. This project was funded by the ERC under the European Union's Horizon 2020 research and innovation programme (FUN2MODEL, grant agreement No. 834115).

References

1. de Alfaro, L.: Formal Verification of Probabilistic Systems. Ph.D. thesis, Stanford University (1997)
2. Aminof, B., Kwiatkowska, M., B. Maubert, B., Murano, A., Rubin, S.: Probabilistic strategy logic. In: Proc. IJCAI'19. pp. 32–38 (2019)
3. Aumann, R.: Subjectivity and correlation in randomized strategies. Journal of Mathematical Economics **1**(1), 67–96 (1974)
4. Baier, C., Katoen, J.P.: Principles of Model Checking. MIT Press (2008)
5. Baier, C., Kwiatkowska, M.: Model checking for a probabilistic branching time logic with fairness. Distributed Computing **11**(3), 125–155 (1998)
6. Banerjee, T., Majumdar, R., Mallik, K., Schmuck, A.K., Soudjani, S.: Fast symbolic algorithms for omega-regular games under strong transition fairness. Tech. Rep. MPI-SWS-2020-007r, Max Planck Institute (2021)
7. Brenguier, R.: PRALINE: A tool for computing Nash equilibria in concurrent games. In: Sharygina, N., Veith, H. (eds.) Proc. CAV'13. LNCS, vol. 8044, pp. 890–895. Springer (2013), lsv.fr/Software/praline/
8. Chatterjee, K., Fijalkow, N.: A reduction from parity games to simple stochastic games. EPTCS **54**, 74–86 (2011)
9. Chatterjee, K., Henzinger, T.: Value iteration. In: 25 Years of Model Checking. LNCS, vol. 5000, pp. 107–138. Springer (2008)
10. 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)
11. Chen, X., Deng, X., Teng, S.H.: Settling the complexity of computing two-player Nash equilibria. J. ACM **56**(3) (2009)
12. Daskalakis, C., Goldberg, P., Papadimitriou, C.: The complexity of computing a Nash equilibrium. Communications of the ACM **52**(2), 89–97 (2009)
13. 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
14. Dutertre, B.: Yices 2.2. In: Biere, A., Bloem, R. (eds.) Proc CAV'14. LNCS, vol. 8559, pp. 737–744. Springer (2014), yices.csl.sri.com
15. Gilboa, I., Zemel, E.: Nash and correlated equilibria: Some complexity considerations. Games and Economic Behavior **1**(1), 80–93 (1989)
16. Griva, I., Nash, S., Sofer, A.: Linear and Nonlinear Optimization: Second Edition. CUP (01 2009)
17. Gurobi Optimization, LLC: Gurobi Optimizer Reference Manual (2021), www.gurobi.com
18. Gutierrez, J., Harrenstein, P., Wooldridge, M.J.: Reasoning about equilibria in game-like concurrent systems. In: Proc. 14th International Conference on Principles of Knowledge Representation and Reasoning (KR'14) (2014)
19. Hauser, O., Hilbe, C., Chatterjee, K., Nowak, M.: Social dilemmas among unequals. Nature **572**, 524–527 (2019)
20. Kemeny, J., Snell, J., Knapp, A.: Denumerable Markov Chains. Springer (1976)
21. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Multi-player equilibria verification for concurrent stochastic games. In: Gribaudo, M., Jansen, D., Remke, A. (eds.) Proc. QEST'20. LNCS, Springer (2020)
22. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: PRISM-games 3.0: Stochastic game verification with concurrency, equilibria and time. In: Proc. CAV'20. pp. 475–487. LNCS, Springer (2020)

23. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Correlated equilibria and fairness in concurrent stochastic games (2022), arXiv:2201.09702
24. Kwiatkowska, M., Norman, G., Parker, D., Santos, G.: Automatic verification of concurrent stochastic systems. Formal Methods in System Design pp. 1–63 (2021)
25. Littman, M., Ravi, N., Talwar, A., Zinkevich, M.: An efficient optimal-equilibrium algorithm for two-player game trees. In: Proc. UAI'06. pp. 298–305. AUAI Press (2006)
26. McIver, A., Morgan, C.: Results on the quantitative mu-calculus qMu. ACM Trans. Computational Logic **8**(1) (2007)
27. von Neumann, J., Morgenstern, O., Kuhn, H., Rubinstein, A.: Theory of Games and Economic Behavior. Princeton University Press (1944)
28. Nisan, N., Roughgarden, T., Tardos, E., Vazirani, V.: Algorithmic Game Theory. CUP (2007)
29. Nudelman, E., Wortman, J., Shoham, Y., Leyton-Brown, K.: Run the GAMUT: A comprehensive approach to evaluating game-theoretic algorithms. In: Proc. AAMAS'04. pp. 880–887. ACM (2004), gamut.stanford.edu
30. Osborne, M., Rubinstein, A.: An Introduction to Game Theory. OUP (2004)
31. Porter, R., Nudelman, E., Shoham, Y.: Simple search methods for finding a Nash equilibrium. In: Proc. AAAI'04. pp. 664–669. AAAI Press (2004)
32. Prisner, E.: Game Theory Through Examples. Mathematical Association of America, 1 edn. (2014)
33. Rabin, M.: Incorporating fairness into game theory and economics. The American Economic Review **83**(5), 1281–1302 (1993)
34. Rabin, M.: Fairness in repeated games. working paper 97–252, University of California at Berkeley (1997)
35. Schwalbe, U., Walker, P.: Zermelo and the early history of game theory. Games and Economic Behavior **34**(1), 123–137 (2001)
36. Shapley, L.: Stochastic games. PNAS **39**, 1095–1100 (1953)
37. Stevens, S., Palocsay, S.: Teaching use of binary variables in integer linear programs: Formulating logical conditions. INFORMS Transactions on Education **18**(1), 28–36 (2017)
38. Wächter, A.: Short tutorial: Getting started with IPOPT in 90 minutes. In: Combinatorial Scientific Computing. No. 09061 in Dagstuhl Seminar Proceedings, Leibniz-Zentrum für Informatik (2009), github.com/coin-or/Ipopt
39. Wächter, A., Biegler, L.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming **106**(1), 25–57 (2006)
40. Supporting material, www.prismmodelchecker.org/files/tacas22equ/

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.