Verification of Strong Nash-equilibrium for Probabilistic BAR Systems: 20th International Conference on Formal Engineering Methods, ICFEM 2018, Gold Coast, QLD, Australia, November...

Chapter · January 2018

DOI: 10.1007/978-3-030-02450-5_7

CITATIONS
0

READS
13

4 authors, including:

Dileepa Fernando
National University of Singapore
5 PUBLICATIONS **10** CITATIONS
SEE PROFILE

Cyrille Jegourel
Singapore University of Technology and Design
21 PUBLICATIONS **227** CITATIONS
SEE PROFILE

Verification of Strong Nash-equilibrium for Probabilistic BAR Systems

Dileepa Fernando[1,a], Naipeng Dong[1,b], Cyrille Jegourel[2,c], and Jin Song Dong[1,3,d]

[1] National University of Singapore
[2] Singapore University of Technology and Design
[3] Griffith University
[a] fdileepa@comp.nus.edu.sg [b] dcsdn@nus.edu.sg
[c] jegourelcyrille@yahoo.fr [d] dcsdjs@nus.edu.sg

Abstract. Verifying whether rational participants in a BAR system (a distributed system including *Byzantine*, *Altruistic* and *Rational* participants) would deviate from the specified behaviour is important but challenging. Existing works consider this as Nash-equilibrium verification in a multi-player game. If the game is probabilistic and non-terminating, verifying whether a coalition of rational players would deviate becomes even more challenging. There is no automatic verification algorithm to address it. In this article, we propose a formalization to capture that coalitions of rational players do not deviate, following the concept of Strong Nash-equilibrium (SNE) in game-theory, and propose a model checking algorithm to automatically verify SNE of non-terminating probabilistic BAR systems. We implemented a prototype and evaluated the algorithm in three case studies.

1 Introduction

In general, most real-world systems involve collaboration of many distributed parties, e.g., Internet routing [20], peer-to-peer file sharing [3], cooperative backup [14], etc. In these systems, agents are assumed to follow the rules or specifications in the system designs. These rules/specifications may have to be followed over infinite times, e.g. in operating systems [17] and medical computing [7].

In such non-terminating concurrent multi-agent systems a particular agent may deviate from the system specifications to maximise its self-interest. Following the motto "Unity makes strength", self-interested agents may also deviate as coalitions to improve their individual rewards simultaneously. For example, miners in block-chain form coalitions to reduce the cost of breaking block-chain security (i.e. increase profit) [11]. It is thus natural to consider the system as a game[1] in which self-interested agents are rational players, implying that an agent cooperates only if it improves her benefit.

However, rational behaviours are not the only source of deviation from the specifications. Some devices may not work properly for other reasons than self-interest.

[1] Here, *game* refers to the atomic concept of Game Theory, defined as the study of mathematical models of conflict and cooperation between intelligent and rational decision-maker agents.

The interacting components of a large system may be designed by different companies, that increases the probability of incompatibility or misconfiguration. Some devices may also have been intentionally designed to be malicious, in the sense that they aim more at the failures of other agents than the maximisation of their own rewards. Such agents/devices are named as Byzantine players in the game. Though a system is unlikely to work correctly in the presence of misconfigured agents, little has been done to verify such "damaged" system.

Byzantine-Altruistic-Rational (BAR)[2] models have been introduced in [2] in order to analyse the systems with selfish and "broken" agents. Later it has been extended in two directions: 1) Applying to probabilistic systems (PBAR) [9]; 2) Considering coalitions of rational players [16]. In this work we consider the combination - coalition of rational players in PBAR systems.

Verifying whether rational players deviate from the system specification in a BAR system is challenging due to the exhaustive strategy space of rational players. In PBAR, verification becomes even more challenging as the introduction of probabilities makes the calculations more complex. Moreover, in non-terminating PBAR, the convergence of rewards is not guaranteed. Especially, we consider the coalitions of rational players which require to reason about both group interest and individual interest.

Verification techniques like model checking have been used to guarantee whether a system satisfies some arbitrary properties derived from the specifications. For example, in the secret sharing protocol [10], model checking is used to guarantee that a secret is fairly distributed among entities that cooperate according to specified rules. Model checking algorithms are proposed to verify BAR [15], PBAR with stopping games [9] and BAR with a specific coalition [16]. However, there is no algorithm to automatically analyse coalitions in PBAR systems.

Contributions. We propose a formalization of Strong Nash Equilibrium (SNE) [19] to capture that any rational player or their coalition would not deviate from the specified behaviour in non-terminating probabilistic BAR systems (Section 5). We propose an approximation algorithm to automatically verify SNE (Section 6). We implement the algorithm as a prototype and evaluate the algorithm using case studies (Section 7).

2 Related Work

Nash-equilibrium. We observe two directions of research on Nash-equilibrium (NE): 1) Learning NE-strategy, e.g., in planning [18] and social behaviour analysis [13], where the environment and other players are dynamic; 2) analysing NE, where the environment and player behaviours are pre-defined, such as a PBAR system in this work. In analysing NE, we observe 3 sub-directions, namely, computing NE, e.g., PRALINE [4], finding optimal strategy of rational players, e.g., PRISM-games [5] and verifying NE, e.g. EAGLE [21]. Verifying NE is further divided into two categories: applying to games with 0 or 1 rewards e.g., [21], and applying to games with cumulative reward objectives e.g., [15]. BAR system verification resides in the later category.

[2] In BAR model, the agents are divided in three categories, altruistic, rational or Byzantine. Only altruistic agents follow the system specification.

Model Checking for NE. The tool EAGLE [21] and PRALINE [4] only handle non-probabilistic games. The closest to our work is PRISM-games 2.0 [12], which performs multi-objective optimal strategy synthesis for rational coalitions in probabilistic games. However, PRISM-games 2.0 does not handle non-terminating games with discounted long-run rewards (A detailed comparison is presented in section 7.1).

Verification of BAR Systems. The approach in [15] is only applicable for non-probabilistic BAR systems. Later, the work [9] extends it for probabilistic BAR systems stopping with probability 1; while the work [16] extends it with coalition but only considering coalitions with an agreed group reward. Our work combines both extensions in the concept level and further extends them to 1) probabilistic BAR systems in general (without the constraint to stop with probability 1) and 2) considering multi-objectives (individual interests in a coalition) rather than simply considering them as one agent/player.

3 Running Example

We illustrate a PBAR system using a simplified/artificial version of pricing game that is well-known in economics. Assume that sellers simultaneously set their prices for one type of product for an year. Buyers buy from the sellers who offer the cheapest price. The market share is thus divided equally between the sellers who offer the cheapest price. The profit of a seller in an year is the price times his market share for the year. The sellers make an agreement to reset the prices at the beginning of each year according to an agreed probabilistic distribution. A seller is altruistic if she follows the agreement, Byzantine if she chooses her price non-deterministically, rational if he chooses the price that maximizes her long-term profit. This game is played for infinite long time and the profit decreases each year at a rate β (to simulate the inflation or product depreciation).

For simplicity, we assume there are 3 sellers (d_1, d_2 and d_3) and 3 prices (p, $2p$ and $3p$), where p is a positive number. Initially, the sellers' prices are empty denoted as $\langle \perp, \perp, \perp \rangle$ respectively. The agreed price distribution is: resetting the price at p with probability 0.9 and resetting the price at $2p$ and $3p$ with probability 0.05 respectively. We assume d_1 and d_2 can be altruistic or rational and d_3 is Byzantine.

Reward Calculation Illustration. In the scenario that d_1 and d_2 are both rational, for a given year, assume d_1 and d_2 decide to set the price at p and d_3 decides to set the price at $2p$, then the prices for the first year are $\langle p, p, 2p \rangle$. Since buyers choose to buy at the lowest price, the buyers buy from d_1 and d_2. Assuming the total market share is 1, the market share for d_1, d_2 and d_3 is thus $1/2$,$1/2$,and 0 respectively. Hence, the profits for the sellers for the first year is $\langle 1/2 \times p, 1/2 \times p, 0 \times p \rangle = \langle p/2, p/2, 0 \rangle$. The profits of other years can be calculated in the same way. Given a trace specifying the price decisions of the sellers of each chronological year, we can calculate the profit of a seller, denoted as $\mathit{profit}_1, \mathit{profit}_2, \mathit{profit}_3, \cdots$. Since in the long-run, the actual profit for each seller decreases by β, the long-term profit for a seller following the trace is

$$\mathit{profit}_1 + \mathit{profit}_2 * \beta + \mathit{profit}_3 * \beta^2 + \cdots = \Sigma_{t=1}^{\infty} \mathit{profit}_t * \beta^t.$$

4 PBAR System Specification

Given n players, we denote the set of Byzantine players as Z and denote the non-Byzantine set as Z'. A PBAR-system can be formally represented as follows.

Definition 1. *Given a set of n players and a subset Z of Byzantine players, a PBAR-system is a tuple $\mathcal{M} = (S, I, A, T, P, H)$, where*

- *S is a set of states,*
- *I is the set of initial states,*
- *$A = \Gamma_1 \times \Gamma_2 \times \cdots \times \Gamma_n$ is a set of actions, where Γ_i is player i's local action set.*
- *$T : S \times A \to S$ specifies the transition function,*
- *$P : S \times A \times \mathbb{P}(Z') \to [0, 1]$ specifies the probability of taking an action at a state, given a set of altruistic players, which is a subset of non-Byzantine players. $\mathbb{P}(Z')$ is the power set of Z'. A subset of non-Byzantine players is an element in $\mathbb{P}(Z')$.*
- *$H : \langle H_1(s, a), \ldots, H_n(s, a)\rangle$ where, $H_i : S \times A \to \mathcal{R}$ is the reward function of player i*

Running Example Specification. According to the example specification, we have $n = 3$ and $Z = \{d_3\}$. A state represents the price combination of the sellers. Therefore there are 28 states in total, including $\{\langle t_1, t_2, t_3\rangle | t_1, t_2, t_3 \in \{p, 2p, 3p\}\}$ (27 states) and the initial state $\langle \perp, \perp, \perp\rangle$. The actions represent the chosen price $A = \{\langle \gamma_1, \gamma_2, \gamma_3\rangle | \gamma_1, \gamma_2, \gamma_3 \in \{p, 2p, 3p\}\}$. Since from any state it is possible to choose a price combination to go to any other state, the state transitions are edges in the complete graph of the 27 states, plus the edges from the initial state to each other states. The probability function is as follows: Since d_3 is Byzantine, the possible sets of altruistic players are $\emptyset$, $\{d_1\}$, $\{d_2\}$ and $\{d_1, d_2\}$. According to the probabilistic distribution of altruistic players' actions (only altruistic players follow the probabilistic distribution), we have for any $s \in S$ and $\langle \gamma_1, \gamma_2, \gamma_3\rangle \in A$, when only d_1 is altruistic[3],

$$P(s, \langle \gamma_1, \gamma_2, \gamma_3\rangle, \{d_1\}) = \begin{cases} 0.9 & \text{if } \gamma_1 = p \\ 0.05 & \text{if } \gamma_1 = 2p \text{ or } \gamma_1 = 3p. \end{cases}$$

By replacing γ_1 with γ_2, d_1 with d_2 in the above equation, we obtain $P(s, \langle \gamma_1, \gamma_2, \gamma_3\rangle, \{d_2\})$. When both d_1 and d_2 are altruistic,

$$P(s, \langle \gamma_1, \gamma_2, \gamma_3\rangle, \{d_1, d_2\}) = \begin{cases} 0.81 & \text{if } \gamma_1 = p \wedge \gamma_2 = p \\ 0.045 & \text{if } \gamma_1 = p \wedge \gamma_2 = 2p \text{ or } \gamma_1 = 2p \wedge \gamma_2 = p \text{ or} \\ & \quad \gamma_1 = p \wedge \gamma_2 = 3p \text{ or } \gamma_1 = 3p \wedge \gamma_2 = p \\ 0.0025 & \text{otherwise (4 cases).} \end{cases}$$

The pay-off function H is defined as $\langle H_1(s, a), H_2(s, a), H_3(s, a)\rangle$ where $a = \langle \gamma_1, \gamma_2, \gamma_3\rangle$,

$$H_i(s, \langle \gamma_1, \gamma_2, \gamma_3\rangle) = \begin{cases} \gamma_i * \frac{1}{m} & \text{if } \gamma_i = \min\{\gamma_1, \gamma_2, \gamma_3\} \\ & m \text{ is the number of minimum prices in } \langle \gamma_1, \gamma_2, \gamma_3\rangle \\ 0 & \text{otherwise.} \end{cases}$$

5 Formalizing Strong Nash-equilibrium

On top of the specification of a PBAR system, which captures the behaviour of altruistic, rational and Byzantine players, we now formalise the concept of Strong Nash-equilibrium (SNE) [19], capturing that coalition of rational players would not deviate from the specified altruistic behaviour for better pay-off in a PBAR system.

[3] We do not need to consider the case of $\emptyset$ as there is no probability in this case.

Intuitively, given a set of Byzantine players Z, for any coalition C ($C \subseteq Z'$), we compare the pay-off gained by the players in the coalition when they behave altruistically, named as the *altruistic game-reward*, and the maximum pay-off gained by the players in the coalition when they deviate, named as the *rational game-reward*. Only if a coalition's rational game-reward is better than the altruistic game-reward (i.e at least one player in the coalition gets better reward while the other do not lose reward), then the coalition would deviate from the altruistic behaviours. In the following part, we show how to formalise/calculate the two types of rewards.

Given a PBAR system $\mathcal{M} = (S, I, A, T, P, H)$, a path is an action sequence $\pi = \pi_a^0, \ldots, \pi_a^{|\pi|-1}$ ($|\pi|$ can be infinite) where, $\pi_a^l \in A$ is the action at step l. π corresponds to a valid state sequence $\pi_s^0, \ldots, \pi_s^{|\pi|-1}$ where validity is captured by transition function T ($T(\pi_s^l, \pi_a^l) = \pi_s^{l+1}$ $0 \leq l \leq |\pi| - 1$). We denote the set of paths starting from a state s with length k as $\Pi^k(s)$.

In non-terminating systems, both altruistic game-reward and rational game-reward are accumulated in infinite steps starting from the same initial state. To avoid the reward getting infinite and thus not comparable, we use the discounted pay-off, as shown in calculating the long-term pay-off in the running example. Thus, the pay-off of player i following path π is $\mathsf{R}_i(\pi) = H_i(\pi_s^0, \pi_a^0) + H_i(\pi_s^1, \pi_a^1)\beta + \ldots + H_i(\pi_s^{k-1}, \pi_a^{k-1})\beta^{k-1}$.

However, $\mathsf{R}_i(\pi)$ cannot be directly used in the formalisation of SNE. First, we need to consider the Byzantine players in the context of PBAR. Since Byzantine players behave arbitrarily, for a given sequence of joint actions of players in C (a strategy of C), there may be various paths due to the Byzantine players' s choices. Among these paths, we assume that the coalition always consider its minimum pay-off, which captures the guaranteed pay-off of the coalition no matter how the Byzantine players behave. To calculate the guaranteed pay-off, we define two basic concepts: 1) *a joint action sequence of a set of players Λ following a path π* is $\mathsf{P}_\Lambda^\pi = \mathsf{p}(\pi_a^0, \Lambda), \cdots, \mathsf{p}(\pi_a^{|\pi|-1}, \Lambda)$ with $\mathsf{p}(\langle\gamma_1, \ldots, \gamma_n\rangle, \{\lambda_1, \ldots, \lambda_l\}) = \langle\gamma_{\lambda_1}, \ldots, \gamma_{\lambda_l}\rangle$, $\Lambda = \{\lambda_1 \ldots, \lambda_l\}$ (λ_i is the index of the player)[4]; and 2) *a joint action choice of length k (possibly infinite) for a set of players $\Lambda = \{\lambda_1 \ldots, \lambda_l\}$* is defined as $\Sigma_\Lambda^k = \alpha^1, \ldots, \alpha^k$, where $\alpha^l = \langle\gamma_{\lambda_1}^l, \ldots, \gamma_{\lambda_l}^l\rangle$ is the joint action choice at step l. Given a starting state s and a joint action choice Σ_Λ^k, there may be a set of multiple paths adhering the joint action choice, denoted as $\Phi(s, \Sigma_\Lambda^k) = \{\pi | \pi \in \Pi^k(s), \mathsf{P}_\Lambda^\pi = \Sigma_\Lambda^k\}$. Given a starting state s, if we fix a joint action choice of non-Byzantine players Z', the resulting subset of paths will only contain the paths that vary due to the choices of Byzantine players, denoted as $\Phi(s, \Sigma_{Z'}^k)$ (i.e., $\Lambda = Z'$). In the set $\Phi(s, \Sigma_{Z'}^k)$, for a player $i \in Z'$, we first calculate its pay-off of each path and then choose the minimum pay-off. The result is the guaranteed pay-off of i starting from s of length k, given a fixed non-Byzantine players' choice $\Sigma_{Z'}^k$. Formally,

$$u_{i,Z}(s, \Sigma_{Z'}^k) = \min\{\mathsf{R}_i(\pi) | \pi \in \Phi(s, \Sigma_{Z'}^k)\}.$$

Second, due to the probabilistic behaviour of altruistic players, the system is probabilistic with multiple paths, and thus the pay-off of a player is always the expected pay-off, i.e., the pay-off of a path weighted by the probability of the path being taken. That is, if we release the constrain of fixing non-Byzantine players' choice to only fixing the rational/coalition[5] players' choice (Σ_C^k), for each player i, we will obtain a set

[4] Essentially, P_Λ^π projects each action to a part of it.

[5] Note that the rational players are exactly players in the coalition, capturing that the rational players assume the unknown players (not in C) are altruistic by default.

of guaranteed pay-offs $\{u_{i,Z}(s, \Sigma^k_{Z'})|\mathsf{P}_C^{\Sigma^k_{Z'}} = \Sigma^k_C\}$. Since the values in the set vary due to the probabilistic choice of altruistic players, they follow the same probabilistic distribution of altruistic players' joint choice. Thus we calculate the expected the pay-off of i by multiplying each value in the set with its corresponding probability and then adding up the results. Formally, the expected guaranteed pay-off of player i is

$$v_{i,Z}(s, \Sigma^k_C) = E(u_{i,Z}(s, \Sigma^k_{Z'})|\mathsf{P}_C^{\Sigma^k_{Z'}} = \Sigma^k_C).$$

This prepares us to calculate the altruistic and the rational game-reward for players in a coalition. We finally release the constraint of a specified joint action choice of players in the coalition, and there will be the following two cases.

- Altruistic: When the players in C behave altruistically, due to their probabilistic behaviour, when releasing the constraint of a given joint action choice of C, there is a set of $v_{i,Z}(s, \Sigma^k_C)$ values which follow a probabilistic distribution. We calculate the expected value of them, which captures the player i's reward starting from state s with length k, when considering all the possible behaviours of Byzantine and altruistic players' behaviours,

$$U^k_{i,Z}(s) = E(v_{i,Z}(s, \Sigma^k_C)|\forall \Sigma^k_C).$$

- Rational: When the players in C are rational, they follow a joint action choice (the best strategy) that leads to the best rewards, denoted as Υ. When the coalition only contains one player i, Υ can be easily calculated as the maximum value in the set $O = \{v_{i,Z}(s, \Sigma^k_C)|\forall \Sigma^k_C\}$, capturing that i chooses her actions that lead to the best rewards. When the coalition contains multiple players, we will need to find the best pay-off which is non-trivial, since the best choice for one player may not be the best choice for other players in the coalition (discussed in the next Section). No matter which case, Υ is one value in the set O, i.e., $\Upsilon = v_{i,Z}(s, \Sigma^k_C)$ for some Σ^k_C.

To summarise, when s is an initial state and k is infinite, $U^k_{i,Z}(s)$ is i's altruistic game-reward; and the player i's ration game-reward will be $\Upsilon = v_{i,Z}(s, \Sigma^k_C)$ for some Σ^k_C.

Note that differing from the coalition in [16], we do not consider the entire coalition as a single agent/player. That is, if, to achieve the best rational game-reward, it requires some players to sacrifice their pay-off (i.e., some players lose pay-off following the coalition while others in the coalition gain), this coalition will not be valid, since each player in the coalition is rational. In this work, we say a coalition would deviate if for every player in the coalition, the rational game-reward is no less than the altruistic game-reward, i.e., all players do not lose. One last point is that, we introduce a parameter ϵ to quantify the least significant reward gain for deviation (i.e., a player in a coalition gains if his rational game-reward in the coalition is greater than ϵ plus his altruistic game-reward) . This allows us to capture a variety of SNE parametrised by $\epsilon(> 0)$. As a side effect, we can achieve better termination of automatic verification by enlarging ϵ [15].

Therefore, a system satisfying ϵ-SNE is formalised as follows, capturing that there is NOT a joint action (Σ^∞_C) choice such that the coalition would deviate.

Definition 2 (ϵ-Strong Nash Equilibrium). *Let $\epsilon > 0$. A PBAR system $\mathcal{M} = (S, I, A, T, P, H)$ with Byzantine players Z is ϵ-SNE if $\forall C \subseteq Z', \nexists \Sigma^\infty_C$, s.t. $\forall i' \in C$, $\forall s \in I$,*

$$U^\infty_{i',Z}(s) + \epsilon < v_{i',Z}(s, \Sigma^\infty_C).$$

Table 1. Guaranteed profit of d_1

Action choice	$\langle 2p, 2p, p\rangle$	$\langle 2p, 3p, p\rangle$	$\langle 2p, p, p\rangle$
Guaranteed pay-offs ($\langle d_1, d_2, d_3\rangle$)	$\langle 0, 0, p\rangle$	$\langle 0, 0, p\rangle$	$\langle 0, p/2, p/2\rangle$
Action choice	$\langle 3p, 2p, p\rangle$	$\langle 3p, 3p, p\rangle$	$\langle 3p, p, p\rangle$
Guaranteed pay-offs ($\langle d_1, d_2, d_3\rangle$)	$\langle 0, 0, p\rangle$	$\langle 0, 0, p\rangle$	$\langle 0, p/2, p/2\rangle$
Action choice	$\langle p, 2p, p\rangle$	$\langle p, 3p, p\rangle$	$\langle p, p, p\rangle$
Guaranteed pay-offs($\langle d_1, d_2, d_3\rangle$)	$\langle p/2, 0, p/2\rangle$	$\langle p/2, 0, p/2\rangle$	$\langle p/3, p/3, p/3\rangle$

SNE in the Running Example. We illustrate the above ideas using the running example, by calculating the reward for the first year and the entire game-reward for two cases: coalition with size 1 and with size 2.

Calculating First Year Profits. As shown in Sect. 3, in the first year, given a fixed choice of d_1 and d_2 (non-Byzantine players), there are three pay-offs of d_1 (respectively d_2), depending on the choice of d_3. We choose the minimum value, i.e., the guaranteed pay-off of d_1 (respectively d_2). The guaranteed pay-offs of d_1 and d_2 and their corresponding action choices are shown in Table 1[6].

Consider d_1, we first calculate her pay-off when she is altruistic. Since altruistic sellers' behaviour is probabilistic, we calculate the expected guaranteed pay-off of d_1, i.e., d_1's pay-off for each action choice (see Table 1) times its probability, $p/2 \times 0.9 \times 0.05 + p/2 \times 0.9 \times 0.05 + p/3 \times 0.9 \times 0.9 = 0.315p$. That is, $U^1_{1,\{3\}}(\langle \perp, \perp, \perp\rangle) = 0.315p$.

Second, we calculate the pay-off of d_1 when she is rational. In this case, she plays the action which gives her the maximum profit. For example, since d_3's action is always p, if d_2 also chooses p, there are three choices for d_1 (the first column in Table 1). d_1 chooses action p which gives pay-off $p/2$, since the other actions provide 0 pay-off. In this example, it happens that the best strategy for d_1 is always playing p, no matter how d_2 acts. Since d_2 is altruistic, the choice of d_2 is probabilistic. Therefore, the pay-off of d_1 is the expected pay-off depending on the choice of d_2, which is $p/2 \times 0.05 + p/2 \times 0.05 + p/3 \times 0.9 = 0.35p$. That is for any $\Sigma^1_{\{1\}}$, $v_{1,\{3\}}(\langle \perp, \perp, \perp\rangle, \Sigma^1_{\{1\}}) \leq 0.35p$.

Since d_1 and d_2 are symmetric, the pay-off of d_2 is exactly the same as d_1. When d_2 is altruistic, her pay-off is $U^1_{2,\{3\}}(\langle \perp, \perp, \perp\rangle) = 0.315p$; and when d_2 is rational, $\forall \Sigma^1_{\{2\}}, v_{2,\{3\}}(\langle \perp, \perp, \perp\rangle, \Sigma^1_{\{2\}}) \leq 0.35p$.

Therefore, when $\epsilon \leq 0.035p$, $U^1_{i,\{3\}}(\langle p, p, p\rangle) + \epsilon < v_{i,\{3\}}(\langle p, p, p\rangle, \Sigma^1_{\{i\}})$ for both d_1 and d_2. That is, d_1 and d_2 would deviate and thus the game is not a SNE for coalitions of size 1 with path of length 1.

Coalition of d_1 and d_2. According to Table 1, both d_1 and d_2 choosing the action p in which d_1 and d_2 gain $p/3$ individually, which is better than being altruistic which gains $0.315p$. When $\epsilon \leq (p/3 - 0.315p)$, d_1 and d_2 both deviate. In this particular case, the coalition deviation corresponds to their individual deviation.

Calculating Game Profits. Assuming $\beta = 0.5$, in this example, since the pay-offs for each year are the same as in the first year, the long-term game-reward for d_1 and d_2 when they are altruistic and rational (without coalition) are as follows ($i \in \{1, 2\}$ to

[6] In this example, it happens (uncommonly) that given any fixed choice of d_1 and d_2, that d_3 chooses the lowest price p leads to the guaranteed pay-off of d_1 and d_2 in every case.

indicate d_1 and d_2 respectively): $U_{i,\{3\}}(\langle\perp,\perp,\perp\rangle) = 0.315p + 0.315p\beta + \ldots = \frac{0.315p}{1-\beta}$, and $\forall \Sigma^{\infty}_{\{i\}}, v_{i,\{3\}}(\langle\perp,\perp,\perp\rangle, \Sigma^{\infty}_{\{i\}}) \leq 0.35p + 0.35p\beta + \ldots = \frac{0.35p}{1-\beta}$. Thus SNE with $\epsilon \leq \frac{0.035p}{1-\beta}$ is violated for coalitions size 1. When d_1 and d_2 form a coalition, similarly, the best joint action is both d_1 and d_2 choosing p. Each seller's long-term game-reward is $v_{i,\{3\}}(\langle\perp,\perp,\perp\rangle, \Sigma^{\infty}_{\{1,2\}}) = \frac{p/3}{1-\beta}$ for $i \in \{1,2\}$. The coalition would deviate when $\epsilon \leq \frac{(p/3-0.315p)}{1-\beta}$, since $U_{i,\{3\}}(\langle\perp,\perp,\perp\rangle) + \epsilon \leq v_{i,\{3\}}(\langle\perp,\perp,\perp\rangle, \Sigma^{\infty}_{\{1,2\}})$.

Note that in this example, it happens that the actions maximizing the game-reward coincide with the best actions in each year. However, this is not always true in general. In many cases, the best strategy leading to the best long-term game-reward may not be the best action in each step. Automated verification is particularly useful in such cases.

6 Verification Algorithm

6.1 Reduction

Verifying ϵ-SNE is equivalent to finding the negation of the ϵ-SNE conditions defined in Def. 2, i.e., finding a joint action choice of some coalition so that the players in the coalition deviate. If such a joint action choice exists, then ϵ-SNE is violated. Thus the verification problem can be reduced to a multi-objective optimization problem of finding the existence of feasible joint action choices.

The negation of Def. 2 is as follows: For some coalitions $C \subseteq Z'$ there exist Σ^{∞}_{C}, for all $i \in C, s \in I, v_{i,Z}(s, \Sigma^{\infty}_{C}) - U_{i,Z}(s) > \epsilon$. The verification of the above inequality can be reduced to the following multi-objective optimization problem:

$$\underset{\Sigma^{\infty}_{C}}{\text{maximize}}\ \textbf{objective}\text{: } v_{i,Z}(s, \Sigma^{\infty}_{C}) - U_{i,Z}(s)\ \forall i \in C.$$

$$\text{subject to } \textbf{constraints}\text{: } v_{i,Z}(s, \Sigma^{\infty}_{C}) - U_{i,Z}(s) > \epsilon\ \forall i \in C.$$

Given a fixed Σ^{∞}_{C} and $s \in I$, the objectives of players in C form an objective vector of n-dimensions ($n = |C|$) where each dimension represents each player's objective. Depending on different Σ^{∞}_{C} and $s \in I$, there is a set of objective vectors. In multi-objective optimization, we say that vector $\boldsymbol{x}$ dominates vector $\boldsymbol{y}$ if the value of each dimension of $\boldsymbol{x}$ is no less than the corresponding value of $\boldsymbol{y}$. Thus the constraints can be represented by a vector $\boldsymbol{x}$ dominates the vector $\langle\epsilon,\ldots,\epsilon\rangle$. The verification of ϵ-SNE is now reduced to find whether there exists a vector which dominates $\langle\epsilon,\ldots\epsilon\rangle$ or not. Non-existence of this vector is equivalent to SNE. To find the vector satisfying the constraint, it is sufficient to compare $\langle\epsilon,\ldots,\epsilon\rangle$ with the set of vectors which are not dominated by any other vectors, which is exactly finding a solution for the optimization problem. Each vector can be considered as a point in a n-dimensional space. The set of vectors that are not dominated by any other vectors form a curve, called Pareto-curve in the n-dimensional space. Verification of ϵ-SNE can be reduced to checking whether or not a point in the Pareto-curve dominates $\langle\epsilon,\ldots,\epsilon\rangle$.

6.2 Approximation

In order to find a point which dominates $\langle\epsilon,\ldots,\epsilon\rangle$, one has to enumerate the points on the Pareto curve w.r.t all the strategy combinations (potentially infinite), which may not

be feasible in some cases. For simplicity, we choose only the coalition strategies corresponding to the optimal value of a single objective function which can be proven to be Pareto optimal strategies in multi-objective function [6]. The single objective function is a linear combination of each player's individual objectives parametrised by α as follows: $\alpha = \langle \alpha_1, \dots \alpha_{|C|} \rangle$ such that $\alpha_i \in [0, 1]$ and $\Sigma_{i=1}^{|C|} \alpha_i = 1$. Thus the linear combination is $\Sigma_{i=1}^{|C|} \alpha_i (v_{i,Z}(s, \Sigma_C^\infty) - U_{i,Z}(s))$. The Pareto-curve is approximated by the optimal objective vectors (corresponding to linear combinations) generated by various α values.

Since $U_{i,Z}(s)$ is a constant for a given i, s and Z, we can simplify the previous linear combination formula into $\Sigma_{i=1}^{|C|} \alpha_i (v_{i,Z}(s, \Sigma_C^\infty))$, which is named as the *joint objective function*. With a configuration C, Z and s, given an α, there is an optimal joint action that maximizes the joint objective function which can be denoted as $\sigma^m = argmax\{\Sigma_{i=1}^{|C|} \alpha_i (v_{i,Z}(s, \Sigma_C^\infty))\}$. With a joint action choice σ^m, we can reverse to the original vector representing $v_{i,Z}(s, \Sigma_C^\infty) - U_{i,Z}(s)$ for each player i in C. This vector corresponds to a point in the Pareto curve. If the vector dominates $\langle \epsilon, \dots, \epsilon \rangle$, this joint action choice is a strategy for the coalition to deviate.

Yet, another issue is that in general, the pay-off of each player (one value in each vector) cannot be calculated as an exact value due to the infinite length. Therefore, we approximate the pay-offs in a vector with some error bound $(\Xi_{u,i}(t) + \Xi_{v,i}(t))$ (uncertainty level) which leads to a region (named as uncertain region) rather than a point in the Pareto curve. For example, Fig. 1, given a joint action choice σ^m for t steps (dot t), there is the shaded area between two parallel dotted lines (the gradient is determined by α) representing the possible joint action choices that lead to the optimal rewards in infinite steps. Each point in this area is a vector $\langle V_{1,Z,C,\alpha}^\infty(s)$-$U_{1,Z}(s), \dots, V_{i,Z,C,\alpha}^\infty(s)$-$U_{i,Z}(s) \rangle (i \in C)$. For each i, we find the points that correspond to the lowest $V_{i,Z,C,\alpha}^\infty(s)$-$U_{i,Z}(s)$. For each such point ($l$), we calculate the uncertainty region that contains the points that dominate l (the triangular darker region). This region must contain some point which is not dominated by any point in the shaded area (For proofs see [8]). This region decreases when increasing t. We can choose $k = \min(t)$ s.t. every dimension i of a point in the uncertainty region is bounded by parameter δ. We provide details about finding k in our proof [8].

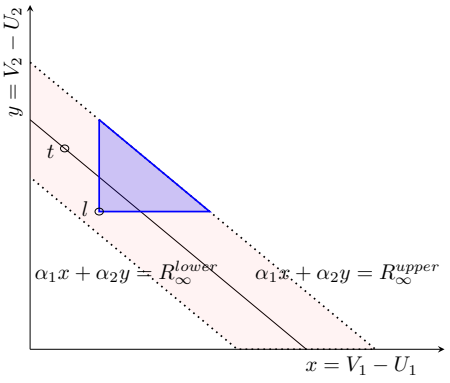

Fig. 1. Uncertain regions

6.3 Calculation of Rewards

We calculate the rewards for all players $i \in C$ in k steps as follows. Let $L = Z' \setminus C$, we first initialize the reward for 0 steps: $V_{i,Z,C,\alpha}^0(s) = 0$, and $U_{i,Z}^0(s) = 0$. Then we itera-

tively calculate the guaranteed state-reward (we refer to the game-reward starting from a state s as a state reward) for player i for a given non-Byzantine action combination $\langle a_C, a_L \rangle$[7] in $t+1$-steps starting from state s for a joint-objective function parameterized by α, in order to determine the optimal action combination. The above guaranteed state-reward is denoted by $g^t_{i,Z,\alpha}$, thus

$$g^{t+1}_{i,Z,\alpha}(s, \langle a_C, a_L \rangle) = \{min_{a_Z \in A_Z}((H_i(s,a) + \beta_i V^t_{i,Z,C,\alpha}(s'))|a = \langle a_C, a_Z, a_L \rangle, T(s,a) = s')\}.$$

Expected guaranteed state-reward of player i for joint action a_C for the last action in $t+1$-step action sequence is $v^{t+1}_{i,Z,\alpha}(s, a_C) = E_{a_L \in A_L}(g^{t+1}_{i,Z,\alpha}(s, \langle a_C, a_L \rangle))$.

We define expected guaranteed state-reward of player i in rational coalition C for $t+1$-step action sequence as,

$$V^{t+1}_{i,Z,C,\alpha}(s) = v^{t+1}_{i,Z,\alpha}(s, a^m_C),$$
$$\text{where } a^m_C = argmax\{\{\Sigma^{|C|}_{i=1}\alpha_i(v_{i,Z,\alpha}(s, a'_C))\} \mid a'_C \text{ is enabled at } s\}.$$

We define expected guaranteed state-reward of i when all coalitions C are altruistic for $t+1$-step action sequence as,

$$U^{t+1}_{i,Z}(s) = E_{a_{Z'} \in A_{Z'}}(g^{t+1}_{i,Z,\alpha}(s, a_{Z'})).$$

6.4 Algorithm

Given a regret value ϵ and parameter δ, we propose Algorithm 1 to decide whether a given model $\mathcal{M}$ with Byzantine players Z satisfies SNE. Line $1-6$ calculates the state rewards for each player i (not in Z) in k steps from the initial states (i.e., $V^k_{i,Z,C,\alpha}(s_0)$ and $U^k_{i,Z}(s_0)$, $s_0 \in I$). Using the results, we calculate the maximum gain of deviating in k steps (denoted as $\Delta_{i,\alpha}$), i.e., $max(V^k_{i,Z,C,\alpha}(s_0) - U^k_{i,Z}(s_0))$ (line 7). Then we use $\Delta_{i,\alpha}$ and the error bound $\Xi_{u,i}(k) + \Xi_{v,i}(k)$ to approximate the gain of deviating in infinite steps and obtain its lower and upper bounds (line 8). The final step is to decide the relation between the bounds and ϵ according to Def. 1. Given an α, the bounds of all players form the uncertainty region like in Fig. 1. If all points in the uncertainty region dominate $\langle \epsilon, \ldots, \epsilon \rangle$, we have $\epsilon_1(i, \alpha) > \epsilon, \forall i$, meaning that there is a strategy for C to deviate and gain more than ϵ, then $\mathcal{M}$ does not satisfy ϵ-SNE (line $9-10$). If all the points in the uncertainty region are dominated by $\langle \epsilon, \ldots, \epsilon \rangle$ ($\epsilon_2(i, \alpha) \leq \epsilon, \forall i$), then there is no joint action choice of C can deviate and gain, and thus the algorithm returns SNE (line $11-12$). Otherwise, we are uncertain about the domination relation, therefore we enlarge the regret value by δ, which will lead to an easier decision (line $13-14$). If by refining regret value, we still cannot decide, we try a different α value (line $15-18$). Given β and δ, the algorithm has complexity $O(|S| * |A|)$ where $|S|$ and $|A|$ are the sizes of state space and the action space.

7 Evaluation

In this section we evaluate our model and algorithm in three aspects: efficiency, scalability and applicability. 1) We illustrate the efficiency of our algorithm by comparing

[7] a_C is short for $\mathsf{p}(a, C)$, a_L is short hand for $\mathsf{p}(a, L)$.

Algorithm 1 $isSNE$(game $\mathcal{M}$, set Z, double ϵ, δ)

Let $s \in S, C \in Z'$ $a_C \in A_C$,$L = Z' \setminus C, i \in C$
Let k, $2(\Xi_{u,i}(k) + \Xi_{v,i}(k)) \leq \delta$, $\alpha_i \in (0,1)$
$V^0_{i,Z,C,\alpha}(s) \leftarrow 0; U^0_{i,Z}(s) \leftarrow 0;$
for t=1 to k **do**
    Update $U^t_{i,Z}(s)$,$V^t_{i,Z,C,\alpha}(s)$
end for
$\Delta_{i,\alpha} \leftarrow \max\{V^k_{i,Z,C,\alpha}(s) - U^k_{i,Z}(s) | s \in I\}$
$\epsilon_1(i,\alpha) \leftarrow \Delta_{i,\alpha} - \Xi_{u,i}(k) - \Xi_{v,i}(k)$; $\epsilon_2(i,\alpha) \leftarrow \Delta_{i,\alpha} + \Xi_{u,i}(k) + \Xi_{v,i}(k)$
if $(\epsilon_1(i,\alpha) > \epsilon \; \forall i \in C)$ **then**
    return NOT SNE
else if $(\epsilon_2(i,\alpha) \leq \epsilon, \forall i \in C)$ **then**
    return ϵ-SNE
else if $((\epsilon_1(i,\alpha) \leq \epsilon \; \forall i \in C)$,
$(\exists i \in C \; \epsilon_1(i,\alpha) \leq \epsilon \leq \epsilon_2(i,\alpha)))$ **then**
    return $\epsilon + \delta$-SNE
else
    $\delta^* \leftarrow \max\{(\epsilon_2(i) - \epsilon)\mathbb{1}(\epsilon_2(i) > \epsilon) | i \in C\}$
    return UNDECIDED
end if

with existing tools - PRISM-games in particular. We use the job scheduling as an example as it is representative of many real-world problems while being easy to illustrate. 2) To show the scalability of our algorithm, we take the apple-picking game as an illustration. The apple-picking game is a practical game that has been used to perform analysis in the game theory domain. Especially, it engages all elements that we would like to illustrate - probabilistic infinite game with rational players that may potentially form coalitions. 3) We apply our algorithm to a real-world scenario - analysing a probabilistic secret sharing scheme. The secret sharing scheme is an important building block in many security critical systems. Being a multi-party computation scheme, rational and Byzantine participants are key concepts. Moreover, it has been proved that probability is essential in the secret sharing scheme to counter the rational behaviour of participants [1]. Particularly, the secret sharing scheme we analyse considers coalitions.

7.1 Comparison with PRISM-games - Job Scheduling Case Study

As mentioned in Sect. 2, the most relevant work to ours is PRISM-games 2.0 [12], since it also handles NE in probabilistic game with coalitions under adversarial behaviour. However, it does not support infinite games. Moreover, it does not directly support SNE verification. In order to be able to adopt PRISM-games for PBAR verification, one has to apply some modelling tricks. We make the following modelling effort in using PRISM-games for verifying PBAR: first, we limited the game length/steps and encoded the discount factor in each state; second, we model different player configurations separately and each forms a model. In each model, we compute the optimal rational/altruistic rewards for the limited length using PRISM-games. Then we compare the rewards for all models to see whether SNE is satisfied. We use the following job scheduling game (adapted from [15] by adding probability) to show the comparison with our algorithm.

Table 2. Experimental results for job assignment

Configuration			Coalition	Coalition	Coalition	CPU	Our algorithm		PRISM-games	
players	jobs	$\|Z\|$	#1-ϵ	#1-$\epsilon+\delta$	#2-ϵ	(sec)	states	transitions	states	transitions
3	2	1	FAIL	PASS	PASS	0.891	3^6	3^9	451625	OutofMemory
3	3	1	FAIL	PASS	PASS	0.971	3^6	3^9	554134	OutofMemory
4	2	1	FAIL	PASS	PASS	40.453	3^8	3^{12}	OutofMemory	OutofMemory
4	2	2	FAIL	PASS	PASS	31.046	3^8	3^{12}	OutofMemory	OutofMemory
4	3	1	FAIL	PASS	PASS	49.53	3^8	3^{12}	OutofMemory	OutofMemory
4	3	2	FAIL	PASS	PASS	37.507	3^8	3^{12}	OutofMemory	OutofMemory

Job Scheduling. Suppose there are m jobs $J = \{J_1, \cdots, J_m\}$ each of which consists of a set of tasks; and there are q tasks $T = \{T_0, \cdots, T_q\}$. A system, comprising n workers, specifies a sequence of tasks for each worker. A task sequence for worker i is denoted as $\mathcal{T}_i = \langle \tau(i,0), \cdots, \tau(i,l_i) \rangle$, where $l_i + 1$ is the length of the task sequence. A set of tasks form a job, denoted by the function $\eta : J \to \mathbb{P}(T)$. Once a job is completed, the rewards are granted to the workers who finished the tasks that compose the job. When a worker finishes a task, she can choose to wait for her reward with probability 0.6 and not to wait with probability 0.4. Once she gets the rewards or decides not to wait, she can start the next task with probability 0.6, take a rest with probability 0.2 or skip the next task with probability 0.2. After finishing all tasks in the sequence (i.e., after finishing $\tau(i,l_i)$), a worker repeats the sequence of tasks from the beginning (i.e., task $\tau(i,0)$). Taking a task consumes the rewards (i.e., getting negative rewards). If more than one worker finish the same task which finishes some job, all of them will be rewarded. A worker deviates from the system by not following the probabilistic distribution.

Experimental Results. Given a set of Byzantine players Z, let $\epsilon = 1.1$ and $\delta = 0.1$, we verify SNE for different configurations of jobs and players. We run the verification using our algorithm and PRISM-games using a 64-bit x-86 Intel machine with 2.40 GHz processor and 8GB RAM. The experimental results are shown in Table 2. For the simplicity of representation, we use $|Z|$ to represent all the possible configurations with the same length of Z in Table 2, since in this example, the verification results for the same $|Z|$ are the same, due to symmetry of players' behaviour. For PRISM-games, we set the game length of each model as 9. From Table 2, we can see that PRISM-games throws out-of memory error in the model building phase even with a limited game length of 9 (with the same corresponding the values of ϵ and δ in each test configuration). The PRISM-games model can be found in [8]. As PRISM-games is not designed for PBAR, auxiliary variables are needed in modelling, which makes the model complicated and thus causes the out-of memory errors in generating the model.

7.2 Evaluating Scalability - Apple-picking Game

The apple-picking game is initially proposed in [13]. This game is easy to understand while has large state space. We use this game to test the scalability of our algorithm, by setting the parameter values that lead to certain limit of the algorithm execution.

Apple-picking Game. Assume there are 3 players, each player has 2 life points. In each round, there is a set of apples (fixed to 4 for the evaluation) and the number decreases

Table 3. Experimental results for apple game

$\|Z\|$	Coalition size	ϵ-SNE	$\epsilon + \delta$-SNE	No. states	No. transactions
1	1	PASS	PASS	3^{15}	4^3
0	1	FAIL	PASS	3^{15}	4^3
1	2	PASS	PASS	3^{15}	4^3
0	2	FAIL	FAIL	3^{15}	4^3

with a discount in each round. The number needs not to be an integer (i.e., this is an infinite game). In each round, a player chooses to shoot others or not. The set of apples are equally shared among the survivors at the end of each round. If a player is shot once, her life point decreases by 1. When a player's life point reaches 0, the player dies, but will reborn in 3 rounds. Each player i also has a confidence value towards another player j, denoted as c_{ij}. c_{ij} reduces by 1 when j shoots i, and increases by 1 every 3 rounds with the cap 2. This case study is a general abstraction of many real-world systems in which agents must find compromise between sharing limited resources, e.g., price fighting. In this game, an analyst may want to know whether a strategy is optimal for every player, for example, shooting based on trust - a player i chooses not to shoot with probability w/q, where w is her maximum confidence value ($Max(c_{ij})$, $i \neq j$) and q is her total confidence value (Σc_{ij}, $i \neq j$), and chooses to shoot with $1 - w/q$. Player i chooses to shoot j with probability c_{ij}/q.

Experimental Results. We model apple picking game as a PBAR-system (defined in Sect. 4). The model contains 3^{15} states and 4^3 transactions. We verified it using the proposed algorithm with $\epsilon = 1.1$ and $\delta = 0.1$, and the result is shown in Table 3.

In the experiment, we considered two scenarios: the presence ($|Z| = 1$ one of the players is Byzantine) or absence ($|Z| = 0$) of a Byzantine player. The algorithm returns "PASS" for ϵ-SNE at the presence of Byzantine player and returns FAIL at the absence of Byzantine player. This result suggests that coalition deviation is only profitable when no player is Byzantine. Moreover, when there is no Byzantine player, the system satisfies $\epsilon + \delta$-SNE when coalition size is 1 but fails when coalition size is 2. This suggests that it is profitable to form a coalition, when there is no Byzantine players. With the CPU configuration 10-core Intel Xeon E5-2630 v4 machine with 2.2GHz processor and 64GB RAM, the verification time for each of the test cases exceeds 4 hours, which indicates the practical limit of the scalability in terms of states and transitions.

7.3 Applicability - Secret Sharing Protocol ADGH06

A secret sharing protocol is a scheme for distributing a secret among a group of participants, each of whom has a share of the secret, and only a sufficient number of shares together can reconstruct the secret. It has been widely used in security related systems e.g., homomorphic secure computation. The most well-known secret sharing protocol is Shamir's secret sharing scheme. However, it is has been proved to not work when there are rational agents, and thus a new scheme with probability is introduced to counter the rational-fault [10]. However, this scheme only considers a single agent's deviation. Hence a new version of the probabilistic secret sharing protocol (ADGH06) is proposed which is able to tolerant to k coalitions of rational agents [1].

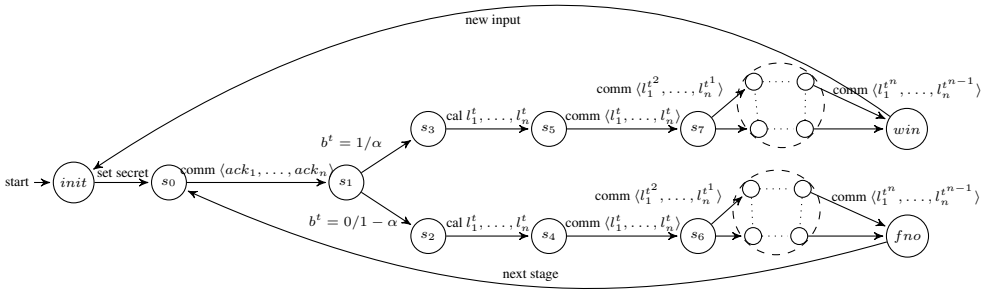

Fig. 2. All altruistic configuration.

Description. In the ADGH06 secret sharing protocol, a secret is encoded as $g(0)$ of an $m-1$ degree polynomial g (defined by a mediator), and it's shares are $g(i)$ which are distributed to each agent i ($i \neq 0$). In order to reconstruct the secret, agents broadcast their secret shares and use interpolation to recover $g(0)$. In details, the protocol repeats the following actions in a stage t until all of agents know the secret.

Each stage consists of 3 phases.

phase 1**.** A mediator collects ack from each agent for stage t.

phase 2**.** The mediator calculates a binary random variable b^t at stage t (the probability of $b^t = 1$ is α and the probability of $b^t = 0$ is $1-\alpha$), produces $l^t = b^t.g + h^t$ (h^t is a random polynomial s.t. $h^t(0) = 0$), and sends $l^t(i)$ to agent i.

phase 3**.** All the agents broadcast their shares to other agents.

Agent i's altruistic strategy is sending its share to all the other agents and reconstruct $l^t(0)$. If i does not receive sufficient number of shares from other agents, then cheating is detected and protocol stops. If $l^t(0) \neq 0$ then $l^t(0)$ is the secret and i gets the reward. If $l^t(0) = 0$ meaning that this round does not count for that the mediator chose meaningless shares (the secret cannot be reconstructed even with all the shares), then i proceeds to the next stage (re-running the above actions). When all the agents follow the protocol, all of them can reconstruct $g(0)$ with $\frac{1}{\alpha}$ expected number of stages.

In better evaluate our algorithm, we introduce Byzantine agent and extend the protocol to a non-terminating one (i.e. secret sharing procedure is repeated). We also introduce a discount factor to capture that the value of the secret decreases over time.

Modelling. Due to space limit, it is impossible to show the entire model of the protocol, which captures all configurations. Here in Fig. 2, we show a projection/part of the model representing the configuration when all agents are altruistic. Probabilities α and $1-\alpha$ are for mediator producing a meaningful polynomial or not. Probabilities of the other transitions is 1. Reward is only given when there is a transition to a win state, where each agent receive all the shares. The rational and Byzantine agents deviate by not sharing or sharing a wrong share. In addition, the rational agents deviate by guessing the secret when not all shares are received and when the mediator chooses meaningless

Table 4. Secret sharing verification results

$\|Z\|$	ϵ	δ	α	ϵ-SNE	$\epsilon + \delta$-SNE	CPU-time (s)	States	Transitions
0	0	0.1	0.3	FAIL	PASS	.658	734	11365
0	0	0.1	0.7	FAIL	PASS	.620	734	11365
0	1.1	0.1	0.3	PASS	PASS	.598	734	11365
0	1.1	0.1	0.7	PASS	PASS	.654	734	11365
1	0	0.1	0.3	FAIL	PASS	.576	734	11365
1	0	0.1	0.7	FAIL	PASS	.585	734	11365
1	1.1	0.1	0.3	PASS	PASS	.665	734	11365
1	1.1	0.1	0.7	PASS	PASS	.626	734	11365

shares, instead of stopping the protocol or restarting the protocol in the altruistic setting. The probability of an rational agent guess correct is p_1, while guessing wrong is $1 - p_1$ (due to the design of the polynomial g). Each rational player has one chance to use the guessed result to obtain the reward. Therefore, the larger the coalition size is, the bigger the chance of getting reward. The rational deviation is defined in [1].

Verification Results. Let the discount factor be $\beta = 0.5$ and the number of agents be $n = 4$, we verified the system using the same configuration as in 7.1. We considered two cases, with a Byzantine agent ($|Z| = 1$) and without Byzantine agents ($|Z| = 0$) (The original scheme does not claimed to be resilient to Byzantine faults). When there is no Byzantine players, the verification result (Table 4) shows that with higher regret allowance (ϵ), rational deviations become less profitable (with the same δ and α). Adding a Byzantine agent does not change the result in current parameters.

8 Conclusion

We aim to decide whether a rational agent in a probabilistic system would follow the specified behaviour in the presence of mis-configured agents. We express this problem as verifying whether the system is a SNE. We proposed a verification algorithm for non-terminating games and proved its correctness. We implemented our algorithm and evaluated using three case studies. Having a running time linear to the size of the state space times the action space, large number of players may consequently slow down the termination of our algorithm. We plan to provide in a future work optimisation and alternative methods to overcome this problem.

Acknowledgement. This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Corporate Laboratory@University Scheme, National University of Singapore, and Singapore Telecommunications Ltd.

References

1. Abraham, I., Dolev, D., Gonen, R., Halpern, J.: Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation. In: Proc. 25th annual ACM symposium on Principles of distributed computing. pp. 53–62 (2006)
2. Aiyer, A., Alvisi, L., Clement, A., Dahlin, M., Martin, J.P., Porth, C.: BAR Fault Tolerance for Cooperative Services. In: Proc. 20th ACM Symposium on Operating Systems Principles. pp. 45–58 (2005)

3. Backes, M., Ciobotaru, O., Krohmer, A.: RatFish: A File Sharing Protocol Provably Secure against Rational Users. In: Proc. 15th European Symposium on Research in Computer Security. LNCS, vol. 6345, pp. 607–625 (2010)
4. Brenguier, R.: PRALINE: A Tool for Computing Nash Equilibria in Concurrent Games. In: Proc. 25th Computer Aided Verification. LNCS, vol. 8044, pp. 890–895 (2013)
5. Chen, T., Forejt, V., Kwiatkowska, M.Z., Parker, D., Simaitis, A.: PRISM-games: A Model Checker for Stochastic Multi-Player Games. In: Proc. 19th Tools and Algorithms for the Construction and Analysis of Systems. LNCS, vol. 7795, pp. 185–191 (2013)
6. Coello, C.C.: A comprehensive survey of evolutionary-based multiobjective optimization techniques. Knowledge and Information systems **1**(3), 129–156 (1999)
7. Diamond, G., Rozanski, A., Steuer, M.: Playing doctor: Application of game theory to medical decision-making. Journal of Chronic Diseases (1986)
8. Fernando, D., Dong, N., Jegourel, C., Dong, J.: Verification of strong nash-equilibrium in probabilistic bar systems(extended with proof)., https://sites.google.com/view/verify-pbar
9. Fernando, D., Dong, N., Jegourel, C., Dong, J.: Verification of nash-equilibrium for probabilistic bar systems. In: ICECCS. pp. 53–62 (2016)
10. Halpern, J., Teague, V.: Rational Secret Sharing and Multiparty Computation: Extended Abstract. In: Proc. 36th Annual ACM Symposium on Theory of Computing. pp. 623–632 (2004)
11. Kiayias, A., Koutsoupias, E., Kyropoulou, M., Tselekounis, Y.: Blockchain mining games. In: Proc. 2016 ACM Conference on Economics and Computation. pp. 365–382 (2016)
12. Kwiatkowska, M., Parker, D., Wiltsche, C.: Prism-games 2.0: A tool for multi-objective strategy synthesis for stochastic games. In: Tools and Algorithms for the Construction and Analysis of Systems. pp. 560–566 (2016)
13. Leibo, J., Zambaldi, V., Lanctot, M., Marecki, J., Graepel, T.: Multi-agent reinforcement learning in sequential social dilemmas. In: Proc. 16th Conference on Autonomous Agents and MultiAgent Systems. pp. 464–473 (2017)
14. Lillibridge, M., Elnikety, S., Birrell, A., Burrows, M., Isard, M.: A Cooperative Internet Backup Scheme. In: Proc. the General Track: 2003 USENIX Annual Technical Conference. pp. 29–41 (2003)
15. Mari, F., Melatti, I., Salvo, I., Tronci, E., Alvisi, L., Clement, A., Li, H.C.: Model Checking Nash Equilibria in MAD Distributed Systems. In: Formal Methods in Computer-Aided Design. pp. 1–8 (2008)
16. Mari, F., Melatti, I., Salvo, I., Tronci, E., Alvisi, L., Clement, A., Li, H.: Model checking coalition nash equilibria in mad distributed systems. In: Self-Stabilizing Systems. pp. 531–546 (2009)
17. McNaughton, R.: Infinite games played on finite graphs. Annals of Pure and Applied Logic **65**(2), 149–184 (1993)
18. Mouaddib, A.I., Boussard, M., Bouzid, M.: Towards a formal framework for multi-objective multiagent planning. In: Proc. 6th international joint conference on Autonomous agents and multiagent systems. p. 123 (2007)
19. Shinohara, R.: Coalition-proof equilibria in a voluntary participation game. International Journal of Game Theory **39**(4), 603–615 (2010)
20. Shneidman, J., Parkes, D.C.: Specification Faithfulness in Networks with Rational Nodes. In: Proc. 23rd Annual ACM Symposium on Principles of Distributed Computing. pp. 88–97 (2004)
21. Toumi, A., Gutierrez, J., Wooldridge, M.: A Tool for the Automated Verification of Nash Equilibria in Concurrent Games. In: Proc. 12th Theoretical Aspects of Computing. LNCS, vol. 9399, pp. 583–594 (2015)