Verification of Nash-Equilibrium for Probabilistic BAR Systems

Conference Paper · November 2016

DOI: 10.1109/ICECCS.2016.016

CITATIONS
2

READS
40

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 Nash-Equilibrium for Probabilistic BAR Systems

Fernando Dileepa*, Naipeng Dong†, Cyrille Jegourel‡, Jin Song Dong§
National University of Singapore
{fdileepa*, dongnp†, jegourel‡, dongjs§}@comp.nus.edu.sg

***Abstract*—A BAR system specifies a cooperation between agents who can be altruistic whether they follow the specified behaviours, Byzantine whether they randomly deviate from specifications and rational whether they deviate to increase their own benefits. We consider whether a rational agent indeed follows the specification of a probabilistic BAR system as verifying whether the system is a Nash-equilibrium in the corresponding stochastic games. In this article, we propose an intuitive specification for probabilistic BAR systems and an algorithm to automatically verify Nash-equilibrium. To validate our implementation of the algorithm, we present two case studies – the three-player Rock-paper-scissors game and a probabilistic secret sharing protocol.**

I. Introduction

Game Theory is the study of mathematical models of conflict and cooperation between multiple agents who are considered as intelligent and rational decision-makers, and is used in a large range of domains (economics, psychology, computer science...). In communication networks, the protocols are seen as games in which the involved entities of the network must cooperatively follow the protocol rules. However, it may be possible that a particular entity is corrupted or does not work correctly with respect to the protocol. Some device may have been designed to deviate from the protocol to maximise a self-interest function or may be simply misconfigured. A protocol is unlikely to work correctly with respect to a property in the presence of selfish or "broken" agents. However, it remains of first importance to guarantee some safety property under these conditions. For example, in the secret sharing protocol, we expect that a corrupted agent cannot reconstruct highly sensitive information before the other agents. Byzantine-Altruistic-Rational (BAR) models have been introduced in [1] in order to analyse the protocol correctness of systems under these conditions. In these systems, the agents are divided in three categories of players. They are said to be altruistic if they follow the rules of the protocol, even if a rational choice is to deviate from it. The agents are rational if they may deviate from the protocol in order to maximise their self-interest, captured in a utility function. Finally, Byzantine agents characterise the players who arbitrarily deviate from the protocol because of component failures, malicious intent, security compromise. The general architecture of BAR systems have been introduced in [1]. In the BAR framework, the correctness of the protocol with respect to some property is satisfied if the property remains true in spite of the presence of rational and Byzantine players. Such protocols are said BAR-tolerant. Some of them have been already proposed to implement cooperative services in peer-to-peer data streaming applications [2].

BAR-tolerant protocols are thus important because they guarantee that selfish agents actually play as expected by the protocol even if some agents are identified as broken or misconfigured. However, it remains challenging to prove that a protocol is BAR-tolerant. For that purpose, it is relevant to make use of Nash-equilibriums, a common game theory concept characterising strategies from which rational players should not deviate in order to maximise their utility. In [3], the authors presented a symbolic Model Checking algorithm that automatically verifies whether the proposed protocol is a *Nash-equilibrium*. In [4], PRALINE has been presented as a tool for computing Nash-equilibrium in non-probabilistic concurrent games played on graphs. By concurrent games, we mean games where the next state is defined at the beginning of a turn by the individual choices of all the players. However, these previous works are limited to non-*probabilistic protocols*. Moreover, the game must be modelled with perfect information, meaning that the rational agents are assumed to have a perfect knowledge of the player utilities and of the global state of the game.

In our article, we focus on a particular class of probabilistic protocols, that are protocols in which the agents take decisions accordingly to a specified probability distribution. The altruistic agents play thus with respect to this distribution, the rational agents play with respect to the probability distribution that maximises their expected utility and the Byzantine agents play accordingly to any probability distribution. Note that we do not cover the case in which illegal decisions with respect to the protocol may be taken by the rational and Byzantine players. Moreover, we consider probabilistic games in which the utilities are known by the rational agents, as well as the number of Byzantine players.

Related to our work, the tool PRISM-games [5] computes an optimal strategy for a coalition of rational players in a probabilistic game. This strategy can then be used to verify that the global strategy is a Nash-equilibrium. However, PRISM-games is limited to the analysis of turn-based games, that are games where agents select their moves in turns. Synchronisation between agents is thus not available directly in PRISM-games. Moreover, the protocols are with perfect information: the agents know the whole reward structure as well as the current global state of the game and the probability distri-

butions of all the players. Finally, PRISM-games computes an optimal strategy up to some bounded number of iterations but it does not compute explicitly the potentially larger bound guaranteeing that a particular strategy is a Nash-equilibrium.

A. Contributions and structure of the article

In Section II, we present the specification of parallel and synchronous probabilistic BAR systems. In Section III, we propose a verification algorithm to verify exact Nash-equilibrium of the system. Finally, we apply the algorithm in two case studies, one with perfect information and one with imperfect information in Section IV and we present interesting results with regard of Byzantine agents. We summarise our results in Section V. For the best of our knowledge, this is the first attempt of verifying Nash-equilibrium of probabilistic BAR systems. Nash-equilibrium is verified for imperfect information scenario as well as the perfect information scenario. This differs from previous works e.g., [5] and [3], since we consider imperfect information games and since we verify Nash-equilibriums explicitly instead of approximate equilibriums [5]. Finally, our verification algorithm determines a bound for the number of iterations that guarantees the Nash-equilibrium property.

II. Specification

In BAR systems, the non-altruistic agents do not necessarily follow the protocol rules. It is thus necessary to capture their possible deviations through adequate specifications. Moreover, we consider probabilistic BAR systems. In this setting, the agents may have to choose a local action among several enabled actions. Their choices are governed by a probability distribution. In this section, we present a framework which allows us to formally specify a probabilistic BAR system as a game in an intuitive way. We highlight that our framework facilitates modelling of concurrent games with some imperfect information. In our case, we restrict the imperfect information for a player to the knowledge of her opponents' local states. It means that each agent knows her own local state, the entire reward structure of the game, as well as, given a global state, the outgoing probability distribution.

We first show the formal modelling of each type of players as Finite State Machines (FSM) and then depict the combination of the FSMs to form a global transition system modelling the global behaviour of the whole system.

A. Altruistic players

An altruistic player's behaviours that are specified in a system, can be formally captured by a FSM defined as a tuple $\mathcal{M}_i^a = (S_i^a, I_i^a, A_i^a, G_i^a, T_i^a, P_i^a, H_i^a)$ where

- S_i^a is the (finite) set of states of altruistic player i,
- I_i^a denotes the set of initial states,
- A_i^a is the set of actions,
- G_i^a is the set of atomic propositions defined on global states S that serves as guard conditions of taking an action or gaining rewards,
- $T_i^a : [\phi]S_i^a \times A_i^a \to S_i^a$, with ϕ being a first-order logic formula over G_i^a, specifies the transitions between states and the condition ϕ that needs to be satisfied for taking this transition,
- $P_i^a : S_i^a \times A_i^a \to [0,1]$ such that $\forall s_i \in S_i^a, \sum_{a_i \in A_i^a} P_i^a(s_i, a_i) = 1$ specifies the probabilities of taking an action at state s_i,
- H_i^a has two functions $AH_i^a : S_i^a \times A_i^a \to \mathbb{R}$ specifies the pay-off of an action $a_i \in A_i^a$ at state $s_i \in S_i^a$, and $SH_i^a : [\phi]S \to \mathbb{R}$ specifies the pay-off obtained after reaching states S satisfying formula ϕ.

When the system is well-defined, an altruistic players has either a probabilistic or a deterministic behaviour, that is, at state $s_i \in S_i^a$, there is either one action $a_i \in A_i^a$ or a set of probabilistic actions whose probabilities sum up to 1. When the action is deterministic, the probability is omitted, since the probability is always 1.

Example 1: A three-player Rock-paper-scissors game Consider a 3-player machine-game version of Rock-paper-scissors derived from [6]. We model playing rock, paper, and scissors as playing 0, 1, and 2, respectively. We denote (i, j, k) as the respective outcomes of players p_1, p_2 and p_3, denote $+$ as the operator for the addition modulo 3 and denote $a = b \oplus c$ as $a = b$ or, exclusively, $a = c$. The pay-off of player p_1 for the outcome (i, j, k) is 2 if $j = k$ and $i = j+1$ (p_1 wins uniquely), 1 if $i = k \oplus j$ and $i - 1 = (j \oplus k)$ (both p_1 and another player win), -1 if $i + 1 = j \oplus k$ (p_1 loses to one player), -2 if $j = k$ and $j = i + 1$ (p_1 loses to both players), and 0 if $i = j = k$ or $i + 1 = j \oplus k \wedge i - 1 = k \oplus j$ (no one wins and three players play the same, or p_1 wins one player but loses to another). Player p_2's pay-off is similar to the pay-off of player p_1 after replacing i, j, k by j, k, i in player p_2's pay-off conditions. Player p_3's pay-off is the negatives of player p_1's plus player p_2's pay-off.

We model the altruistic player's behaviour in Example 1 as follows.

- $S_i^a = \{s_1^a, s_2^a, s_3^a, s_4^a\}$, $I_i^a = \{s_1^a\}$, $A_i^a = \{0, 1, 2, \mathit{replay}\}$,
- $G_i^a = \{i = j, j = k, i = j+1, j = i+1, i = k \oplus j, i-1 = (j \oplus k), i + 1 = j \oplus k\}$,
- $T_i^a(s_1^a, 0) = s_2^a$, $T_i^a(s_1^a, 1) = s_3^a$, $T_i^a(s_1^a, 2) = s_4^a$, $[\phi]\, T_i^a(s_4^a, \mathit{replay}) = s_1^a$, $[\phi]\, T_i^a(s_3^a, \mathit{replay}) = s_1^a$, $[\phi]\, T_i^a(s_2^a, \mathit{replay}) = s_1^a$, with $\phi = \neg\big((j = k \wedge i = j+1) \vee (i = k \wedge j = i+1) \vee (i = j \wedge k = i+1)\big)$, meaning that the game terminates when there is a unique winner, and continues otherwise,
- $P_i^a(s_1^a, 0) = 1/3, P_i^a(s_1^a, 1) = 1/3, P_i^a(s_1^a, 2) = 1/3$, and action *replay* is deterministic,
- $SH_i^a(S) = \begin{cases} 2 \text{ if } S \models (j = k \wedge i = j + 1) \\ 1 \text{ if } S \models (i = k \oplus j \wedge i - 1 = (j \oplus k)) \\ 0 \text{ if } S \models ((i = j = k) \vee \\ \qquad (i + 1 = j \oplus k \wedge i - 1 = k \oplus j)) \\ -1 \text{ if } S \models (i + 1 = j \oplus k) \\ -2 \text{ if } S \models (j = k \wedge j = i + 1). \end{cases}$

B. Byzantine players

Byzantine players may randomly deviate from the system by 1) performing actions that are not specified in the system, or 2) performing a set of probabilistic actions with a different probabilistic distribution. The deviating actions at a state shall be specified by the BAR system. As we want to ensure the BAR-tolerance of protocols, we only consider the worst-case scenario for the rational players, meaning that we assume that the Byzantine players always perform an action that minimizes the pay-off of the rational player. This action is unique, for that any distribution over the actions would not minimise the rational player's pay-off larger than one single action. Hence, there is no probability distribution needed for Byzantine players. In addition, since Byzantine players randomly deviate, the guard conditions and pay-off function are not needed neither.

Similar to the altruistic player, a Byzantine player can be formally modelled by a FSM, a tuple $\mathcal{M}_i^b = (S_i^b, I_i^b, A_i^b, T_i^b)$, where

- S_i^b is the (finite) set of states of Byzantine player i,
- I_i^b denotes the set of initial states,
- A_i^b is the set of actions,
- $T_i^b : S_i^b \times A_i^b \to S_i^b$ specifies the transitions between states.

Assume that the Byzantine players in Example 1 can only deviate from the specification by playing according to a different probability distribution than the uniform distribution. A Byzantine player is modelled as follows:

- $S_i^b = \{s_1^b, s_2^b, s_3^b, s_4^b\}$, $I_i^b = \{s_1^b\}$, $A_i^b = \{0, 1, 2, replay\}$,
- $T_i^b(s_1^b, 0) = s_2^b$, $T_i^b(s_1^b, 1) = s_3^b$, $T_i^b(s_1^b, 2) = s_4^b$,
  $T_i^b(s_4^b, replay) = s_1^b$, $T_i^b(s_3^b, replay) = s_1^b$,
  $T_i^b(s_2^b, replay) = s_1^b$.

C. Rational players

Similar to Byzantine players, the rational players may also deviate from the system specification. Unlike Byzantine players, whose actions are random, the rational players' actions are driven by pay-offs, that is, at a state, the rational players choose the action that leads to larger pay-off. Similarly to the Byzantine players, this action is unique. In fact, a rational player has the same allowed transitions, states and actions as a Byzantine player. Unlike to the Byzantine players, the rational players need the pay-off function. Hence, a rational player can be formally modelled by a FSM, a tuple $\mathcal{M}_i^r = (S_i^r, I_i^r, A_i^r, G_i^r, T_i^r, H_i^r)$, where

- S_i^r is the (finite) set of states of rational player i,
- I_i^r denotes the set of initial states,
- A_i^r is the set of actions,
- G_i^r is the set of atomic propositions defined on global states S that serves as guard conditions of taking an action,
- $T_i^r : S_i^r \times A_i^r \to S_i^r$ specifies the transitions between states,
- H_i^r has two functions: $AH_i^r : S_i^r \times A_i^r \to \mathbb{R}$ specifies the pay-off of an action $a_i \in A_i^r$ at state $s_i \in S_i^r$, and $SH_i^r : [\phi]S \to \mathbb{R}$ specifies the pay-off of reaching states S satisfying formula ϕ.

A rational player in Example 1 is modelled as follows:

- $S_i^r = \{s_1^r, s_2^r, s_3^r, s_4^r\}$, $I_i^r = \{s_1^r\}$, $A_i^r = \{0, 1, 2, replay\}$,
- $T_i^r(s_1^r, 0) = s_2^r$, $T_i^r(s_1^r, 1) = s_3^r$,
  $T_i^r(s_1^r, 2) = s_4^r$, $T_i^r(s_4^r, replay) = s_1^r$,
  $T_i^r(s_3^r, replay) = s_1^r$, $T_i^r(s_2^r, replay) = s_1^r$,
- $G_i^r = G_i^a$,
- $H_i^r(S) = \begin{cases} 2 \text{ if } S \models (j = k \wedge i = j + 1) \\ 1 \text{ if } S \models (i = k \oplus j \wedge i - 1 = (j \oplus k)) \\ 0 \text{ if } S \models (i = j = k) \vee \\ \qquad (i + 1 = j \oplus k \wedge i - 1 = k \oplus j)) \\ -1 \text{ if } S \models (i + 1 = j \oplus k) \\ -2 \text{ if } S \models (j = k \wedge j = i + 1). \end{cases}$

D. Global graph

Given a finite set of players modelled as FSMs ($\{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$), one can construct a global FSM $\mathcal{M} = (S, I, A, G, T, P, H)$ modelling that the players executing their FSM concurrently. Note that in our algorithm, the global graph is constructed implicitly on-the-fly. For the clearer presentation of our verification in the next section, we explicitly construct a global graph here to provide an intuition how a global graph is constructed.

The global states in the global FSM is the product of the local states of each player, i.e., $S = S_1 \times S_2 \times \cdots \times S_n$; the set of initial states is $I = I_1 \times I_2 \times \cdots \times I_n$; the set of global actions is $A = A_1 \times A_2 \times \cdots \times A_n$; the set of atomic propositions is $G = G_1 \cup G_2 \cup \cdots \cup G_n$ (G_i is empty if $\mathcal{M}_i$ models a Byzantine player).

Intuitively, the transition function T is defined as $T(\langle s_1, \ldots, s_n\rangle, \langle a_1, \ldots, a_n\rangle) = \langle s'_1 \ldots, s'_n\rangle$ where $T_i(s_i, a_i) = s'_i$ in $\mathcal{M}_i$ meaning that at state $\langle s_1, \ldots, s_n\rangle$ every player takes an action according to their local FSMs, then there is a transition in the global FSM. Note that we require that every player has to take an action: Altruistic players always follow the system specification to take an action; Byzantine and rational players may deviate from the specification by doing nothing or taking other actions, where the deviation actions shall be defined and modelled in the local FSMs. We emphasize that we model each player's local actions rather than requiring the global FSM as input like in [3]. Hence, we need to additionally consider the special cases where multiple users need to take actions at the same time, i.e., action synchronization, e.g., in Example 1, three players need to take the $replay$ action at the same time, in order to re-start the game. We support two ways of modelling synchronized actions: one is to model action as one name with an arity, (*e.g.* the number of players), such that more than the arity number of players taking the action can be synchronized, *e.g.*, the $replay/3$ means that at least three players want to re-start the game; the other is to define a pair of actions, for example $(send, receive)$ in communication, meaning that whenever there is a $send$ action and a $receive$

action are enabled at a state, the synchronized action can be taken. The synchronized actions need to be tagged in a set A_s. In Example 1, $A_s = \{replay/3\}$. When an action $\langle a_1, \ldots, a_n\rangle$ is enabled at a state s_i, if a_i is in A_s, but there is no other action to synchronize with, for the simplicity of modelling, we assume a failure state s_f exists and $T(s, \langle a_1, \ldots, a_n\rangle) = s_f$. Formally, $T(s, \langle a_1, \ldots, a_n\rangle) = s_f$, if $|\{a|a \in \{a_1, \ldots, a_n\} \wedge a/arity \in A_s\}| < arity$ or $(a, b) \in A_s \wedge (|\{a|a \in \{a_1, \ldots, a_n\}| \neq |\{b|b \in \{a_1, \ldots, a_n\}|)$, where $|A|$ denotes the number of elements in set A.

Generally, the probabilistic function is $P(\langle s_1, \ldots, s_n\rangle, \langle a_1, \ldots, a_n\rangle)) = P_1(s_1, a_1) \times \ldots \times P_n(s_n, a_n)$. If $P_i(s_i, a_i)$ is not defined i.e., in the case of deterministic or non-deterministic actions, we set the probability value to be 1. It is straightforward in the case that a_i is a deterministic action. In the case of non-deterministic actions, once a_i is chosen, it becomes deterministic, and thus the value is also 1. In such a way, the probability of the global action $\langle a_1, \ldots, a_n\rangle$ at state $\langle s_1, \ldots, s_n\rangle$ i.e., $\prod_{i=1}^{n} P_i(s_i, a_i)$, actually captures the product of the probabilities of actions in $\{a_1, \ldots, a_n\}$, and thus the probabilities of all global actions at a state sum up to 1. When no action is probabilistic in the global action $\langle a_1, \ldots, a_n\rangle$, then we do not sign a probability to the action at the state, denoted as $P(\langle s_1, \ldots, s_n\rangle, \langle a_1, \ldots, a_n\rangle)) = \circ$, meaning the probability is not defined.

The pay-off function H is defined as $H(\langle a_1, \ldots, a_n\rangle, s) = \langle H_1(a_1, s), \ldots, H_n(a_n, s)\rangle$ where $H_i(a_i, s)$ can be empty, denoted as $\star$, meaning that function H_i is not defined for action a_i at state s.

Given one altruistic player, one Byzantine player, one rational player, the global FSM for Example 1 is constructed using the above way. Since there are too many states in the global FSM, we only demonstrate some interesting parts of the global FSM, for example, we only shows the states with incoming and outgoing transitions.

- $S = \{\langle s_1^a, s_1^b, s_1^r\rangle, \langle s_2^a, s_2^b, s_2^r\rangle, \langle s_3^a, s_2^b, s_2^r\rangle, \langle s_2^a, s_3^b, s_2^r\rangle, \langle s_2^a, s_2^b, s_3^r\rangle, \langle s_3^a, s_3^b, s_2^r\rangle, \langle s_3^a, s_2^b, s_3^r\rangle, \langle s_2^a, s_3^b, s_3^r\rangle, \langle s_3^a, s_3^b, s_3^r\rangle, \langle s_4^a, s_2^b, s_2^r\rangle, \langle s_2^a, s_4^b, s_2^r\rangle, \langle s_2^a, s_2^b, s_4^r\rangle, \langle s_4^a, s_4^b, s_2^r\rangle, \langle s_4^a, s_2^b, s_4^r\rangle, \langle s_2^a, s_4^b, s_4^r\rangle, \langle s_4^a, s_4^b, s_4^r\rangle, \langle s_3^a, s_4^b, s_4^r\rangle, \langle s_4^a, s_3^b, s_4^r\rangle, \langle s_4^a, s_4^b, s_3^r\rangle, \langle s_3^a, s_3^b, s_4^r\rangle, \langle s_3^a, s_4^b, s_3^r\rangle, \langle s_4^a, s_3^b, s_3^r\rangle\}$,
- $I = \{\langle s_1^a, s_1^b, s_1^r\rangle\}$,
- $A = \{\langle a_1, a_2, a_3\rangle | a_1, a_2, a_3 \in \{0, 1, 2, replay\}$ and $A_s = \{replay/3\}$,
- $G = G_i^a$,
- The playing transitions are:
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 0, 0\rangle) = \langle s_2^a, s_2^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 0, 0\rangle) = \langle s_3^a, s_2^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 1, 0\rangle) = \langle s_2^a, s_3^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 0, 1\rangle) = \langle s_2^a, s_2^b, s_3^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 1, 0\rangle) = \langle s_3^a, s_3^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 0, 1\rangle) = \langle s_3^a, s_2^b, s_3^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 1, 1\rangle) = \langle s_2^a, s_3^b, s_3^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 1, 1\rangle) = \langle s_3^a, s_3^b, s_3^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 0, 0\rangle) = \langle s_4^a, s_2^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 2, 0\rangle) = \langle s_2^a, s_4^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 0, 2\rangle) = \langle s_2^a, s_2^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 2, 0\rangle) = \langle s_4^a, s_4^b, s_2^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 0, 2\rangle) = \langle s_4^a, s_2^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, 2, 2\rangle) = \langle s_2^a, s_4^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 2, 2\rangle) = \langle s_4^a, s_4^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 2, 2\rangle) = \langle s_3^a, s_4^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 1, 2\rangle) = \langle s_4^a, s_3^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 2, 1\rangle) = \langle s_4^a, s_4^b, s_3^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 1, 2\rangle) = \langle s_3^a, s_3^b, s_4^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, 2, 1\rangle) = \langle s_3^a, s_4^b, s_3^r\rangle$,
  $T(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, 1, 1\rangle) = \langle s_4^a, s_3^b, s_3^r\rangle$,
- and the re-start transitions are:
  $T(\langle s_2^a, s_2^b, s_2^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_3^a, s_3^b, s_2^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_3^a, s_2^b, s_3^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_2^a, s_3^b, s_3^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_3^a, s_3^b, s_3^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_4^a, s_2^b, s_2^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_2^a, s_4^b, s_2^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_2^a, s_2^b, s_4^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_4^a, s_4^b, s_4^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_3^a, s_4^b, s_4^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_4^a, s_3^b, s_4^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$,
  $T(\langle s_4^a, s_4^b, s_3^r\rangle, \langle replay, replay, replay\rangle) = \langle s_1^a, s_1^b, s_1^r\rangle$.
  The re-start transitions only happen when there is no unique winner. When there is a unique winner, the game terminates. In this case, although the Byzantine player and the rational player *replay* actions are enabled, there will not be a global transition, because the rational player will not take an action.
- $P(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 0, a_i, a_j\rangle) = 1/3 \times 1 \times 1 = 1/3$ (the Byzantine and rational player actions a_i, a_j are non-deterministic, and thus $P^b(s_1^b, a_i) = 1$, $P^r(s_1^r, a_j) = 1$), Similarly, $P(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 1, a_i, a_j\rangle) = 1/3$, $P(\langle s_1^a, s_1^b, s_1^r\rangle, \langle 2, a_i, a_j\rangle) = 1/3$. In addition, $P(\langle s_i^a, s_j^b, s_k^r\rangle, \langle replay, replay, replay\rangle) = \circ$
- $H(\langle a_1, a_2, a_3\rangle, s) =$
$$\begin{cases} \langle 2, -1, \star\rangle \text{ if } S \models (j = k \wedge i = j + 1) \\ \langle -1, 2, \star\rangle \text{ if } S \models (i = k \wedge i + 1 = j) \\ \langle -1, -1, \star\rangle \text{ if } S \models (i = j \wedge i + 1 = k) \\ \langle 0, 0, \star\rangle \text{ if } S \models (i = j = k) \\ \langle 0, 0, \star\rangle \text{ if } S \models (i + 1 = j, j + 1 = k) \\ \langle 0, 0, \star\rangle \text{ if } S \models (i + 1 = k, k + 1 = j) \\ \langle -2, 1, \star\rangle \text{ if } S \models (j = k \wedge j = i + 1). \end{cases}$$

Note that we do not distinguish the global event and the global state since a global event always leads to a unique global state.

III. Verification

In this section, we present our algorithm to verify whether a probabilistic BAR system modelled in the previous specification frame is a Nash-equilibrium. The input of the algorithm is m Byzantine players (possibly empty), denoted as $\mathcal{M}^b = \{\mathcal{M}_1^b, \mathcal{M}_2^b, \ldots, \mathcal{M}_m^b\}$, $n - m$ non-Byzantine players (non-empty), either altruistic, denoted as $\mathcal{M}^a = \{\mathcal{M}_{m+1}^a, \mathcal{M}_{m+2}^a, \ldots, \mathcal{M}_n^a\}$, either rational, denoted

as $\mathcal{M}^r = \{\mathcal{M}^r_{m+1}, \mathcal{M}^r_{m+2}, \ldots, \mathcal{M}^r_n\}$. We assign to each altruistic player $\mathcal{M}^a_{m+j} \in \mathcal{M}^a$ a corresponding rational player $\mathcal{M}^r_{m+j} \in \mathcal{M}^r$. We prove that for any altruistic player, if she deviates from the specified behaviour, i.e., the altruistic player becomes a rational player, her pay-off would not increase. This property captures that the specified altruistic behaviour is a Nash-equilibrium, which is formally defined as follows.

Nash-equilibrium A specification $\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$ with perfect information is a Nash-equilibrium if,

$$\forall init \in I, \forall i > m, U_i(init) \geq V_i(init). \quad (1)$$

A specification $\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$ with perfect information is a Nash-equilibrium if,

$$\forall init \in I_i, \forall i > m, Ul_i(init) \geq Vl_i(init). \quad (2)$$

$\mathcal{M}^b$, $\mathcal{M}^a, \mathcal{M}^r$ are defined the same as the input of the algorithm. I is the set of global initial states, i.e., all the combinations of the initial states of the total n players. m is the number of Byzantine players, i is the index of a player, since the Byzantine players are indexed from 1 to m, $i > m$ means that the player i is not a Byzantine player. In a perfect information game, $U_i : S \times \mathbb{N} \to \mathbb{R}$ denotes the minimum guaranteed pay-off for player i starting from global state $s \in S$ and doing exactly $k \in \mathbb{N}$ actions, when i is *altruistic*. $V_i : S \times \mathbb{N} \to \mathbb{R}$ denotes the maximum guaranteed expected pay-off achieved by player i, starting from global state $s \in S$ and doing exactly $k \in \mathbb{N}$ actions, when i is *rational*. When k tends to ∞, we simply denote the converged value as $V_i(s)$ and $U_i(s)$ respectively. Hence, $U_i(init) \geq V_i(init)$ ensures that a rational player does not deviate because the player would not gain pay-off by deviating. For imperfect information games, the minimum guaranteed expected pay-off of an altruistic player and maximum guaranteed pay-off of a rational player are defined locally and are denoted as $Ul_i : S \times \mathbb{N} \to \mathbb{R}$ and $Vl_i : S \times \mathbb{N} \to \mathbb{R}$, respectively.

We focus on the games that terminate with probability 1 in the long-run, since otherwise, the pay-off of a rational player and the pay-off of the player deviating can be infinite and thus cannot be compared. In the case that a game terminates with probability 1, the pay-off of a player converges to a certain value in the long-run, since the probability of an infinite sequence of actions is negligible.

Given a specification of a BAR system, i.e., a set of players, $\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$, the maximum non-terminating bound ϵ with $0 < \epsilon < 1$, and a boolean value *perfect* indicating whether it is a perfect information game. The algorithm 1 first calculates the iteration steps *iter* by ensuring that the non-terminating probability is smaller than ϵ. Then we call the algorithm 2 by providing $\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$ and *iter* as parameters. The algorithm 2 decides whether $\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$ is a Nash-equilibrium. There are three possible outcomes of algorithm 2: *PASS* meaning that the specification is a Nash-equilibrium, *FAIL* meaning that the specification is NOT a Nash-equilibrium, and *TRY SMALLER* ϵ meaning that the error is too large, and in order to reduce the error, ϵ needs to be set to a smaller value, so that more iteration steps are involved and thus the algorithm returns more accurate (Details of algorithm 2 will be explained in the next paragraph). If the result is *TRY SMALLER* ϵ, we call the algorithm 1 again with a smaller non-terminating bound $\frac{\epsilon}{2}$. To ensure that the loop terminates, we set a large iteration bound $iter_{max}$. When the iteration bound is reached without any definite answers, the algorithm 1 returns *NOT DECIDED* meaning that the algorithm cannot terminate with a definite result.

Algorithm 1 *VerifyNash*($\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$, ϵ, $iter_{max}$, *perfect*)

1: Let $p_{min} = min_{s \in S, a \in A}\{P(s,a)\}$
2: $iter = \lceil \frac{log(\epsilon)}{log(1-p_{min})} \rceil$
3: **if** $iter > iter_{max}$ **then**
4: **return** NOT DECIDED
5: **end if**
6: $Result \leftarrow CheckNash(\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle, iter, perfect)$
7: **if** $Result = TRYSMALLER\epsilon$ **then**
8: $VerifyNash(\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle, \frac{\epsilon}{2}, perfect)$
9: **else**
10: **return** $Result$
11: **end if**

The key part of algorithm 2 is to calculate the $V_i(init, iter)$, $V_i(init, iter)$ for perfect information game and calculate $Vl_i(init, iter)$ and $Ul_i(init, iter)$ for imperfect information game. Firstly, we initialize the four values to be 0 (line 9 and line 10). Then we iteratively calculate the four values. In each iteration, we update the expected pay-offs of the current iteration (step t) to the corresponding values accumulated in the previous $t-1$ steps (line 11 to line 16). In details, to calculate V_i, we first compute a set of possible expected pay-offs at a global state for each action combination of Byzantine players and rational player i. This set is then grouped by the possible next local states of rational player i ($s_i \in S_i$). For each possible s_i, the Byzantine players together choose a combined action for the next state which minimises the expected pay-off of i. This gives the guaranteed pay-off at a given global state for each next local state s_i of player i. Then player i can choose the s_i which gives the maximum guaranteed pay-off at global state s. The difference in calculating U_i is that we calculate the expected pay-off over all the possible next local states for the player i at a global state, instead of taking the maximum, because an altruistic player does not make choice but just follows the specification. Compare to $V_i(init, iter)$ and $V_i(init, iter)$, we calculate $Vl_i(init, iter)$ and $Ul_i(init, iter)$ using a different reward function in the algorithm, which is the transition based reward structure (AH_i). Because given the local state ls. $Vl_i(init, iter)$ and $Ul_i(init, iter)$ represent the expected pay-offs at all the possible global states.

Algorithm 2 *CheckNash*($\langle \mathcal{M}^b, \mathcal{M}^a, \mathcal{M}^r \rangle$, *iter*, *perfect*)

1: **for all** $i \in Al$ **do**
2: Let $s \in S$
3: Let $ls \in S_i$
4: Let $ha_{max} = max\{|AH_i^r(s,a)||s \in S \wedge a \in A\}$
5: Let $hs_{max} = max\{|SH_i^r(s)||s \in S\}$
6: Let $p_{min} = min_{s \in S, a \in A}\{P(s,a)\}$
7: $max_{Aerror} = 2ha_{max}(1-p_{min})^{iter}$
8: $max_{Serror} = 2hs_{max}(1-p_{min})^{iter}$
9: $V_i(s,0) \leftarrow 0; U_i(s,0) \leftarrow 0;$
10: $Vl_i(ls,0) \leftarrow 0; Ul_i(ls,0) \leftarrow 0;$
11: **for** t=1 to $iter$ **do**
12: $V_i(s,t) \leftarrow max_{s'_i \in S_i}\{\ min_{s'_Z \in S_Z}\ E_{s'_{Al\setminus\{i\}} \in S_{Al\setminus\{i\}}}\{SH_i^r(s') + V_i(s', t-1) | T(s,a) = s' for\ some\ a \in A\}\}$
13: $U_i(s,t) \leftarrow E_{s'_i \in S_i}\ \{min_{s'_Z \in S_Z}\ E_{s' Al\setminus\{i\} \in S_{Al\setminus\{i\}}}\{SH_i^a(s') + U_i(s', t-1) | T(s,a) = s' for\ some\ a \in A\}\}$
14: $Vl_i(ls,t) \leftarrow max_{a_i \in A_i}\{\ \{E_{\{s \in S | s_i = ls\}}\ min_{a_Z \in A_Z}\{E_{a_{Al\setminus\{i\}} \in A_{Al\setminus\{i\}}}\{AH_i^r(s,a) + V_i(s', t-1) | T(s,a) = s' \wedge s_i = ls\}\}\}$
15: $Ul_i(ls,t) \leftarrow E_{a_i \in A_i}\ \{E_{\{s \in S | s_i = ls\}}\ min_{a_Z \in A_Z}\{E_{a_{Al\setminus\{i\}} \in A_{Al\setminus\{i\}}}\{AH_i^a(s,a) + U_i(s', t-1) | T(s,a) = s' \wedge s_i = ls\}\}\}$
16: **end for**
17: **end for**
18: **if** !perfect **then**
19: **if** $\forall i \in Al,\ \forall init \in I_i.\ Vl_i(init, iter) \leq Ul_i(init, iter) \wedge Ul_i(init, iter) - Vl_i(init, iter) \geq max_{Aerror}$ **then**
20: **return** $PASS$
21: **else if** $\forall i \in Al, \forall init \in I_i. Vl_i(init, iter) > Ul_i(init, iter) \wedge Vl_i(init, iter) - Ul_i(init, iter) > max_{Aerror}$ **then**
22: **return** $FAIL$
23: **end if**
24: **end if**
25: **if** perfect **then**
26: **if** $\forall i \in Al,\ \forall init \in I.\ V_i(init, iter) \leq U_i(init, iter) \wedge U_i(init, iter) - V_i(init, iter) \geq max_{Serror}$ **then**
27: **return** $PASS$
28: **else if** $\forall i \in Al,\ \forall init \in I.\ V_i(init, iter) > U_i(init, iter) \wedge V_i(init, iter) - U_i(init, iter) > max_{Serror}$ **then**
29: **return** $FAIL$
30: **end if**
31: **end if**
32: **return** TRY SMALLER ϵ

Once the four values are calculated, we can use them to perform the verification. If game is an imperfect information game, Nash-equilibrium definition 2 is applied. If it is a perfect information game, Nash-equilibrium definition 1 is applied. Since we limit the iteration steps, the above calculated value is an approximation of the pay-offs in infinite steps. That implies that an error exists. We calculate the maximum error bound in line 4 to line 8. Intuitively, we obtain the maximum absolute pay-off of the player i, i.e., ha_{max} and hs_{max}, and then calculate the upper bound of the pay-off in any infinite path starting from step $iter$ (we denote the upper bound as e), by summing up the pay-offs weighted by the probability of the corresponding actions in the path. The maximum error bound is computed by doubling the upper bound of pay-off e due to the use of absolute value. The reason is that $U_i(init)$ is in the range of $[U_i(init, iter) - e, U_i(init, iter) + e]$; $V_i(init)$ is in the range of $V_i(init, iter) - e, V_i(init, iter) + e]$; by ensuring $U_i(init, iter) - e \geq V_i(init, iter) + e$, we ensure that $U_i(init) \geq V_i(init)$. Given the calculated error bound, we compare $|V_i(init) - U_i(init)|$ with the error bound (or compare $|Vl_i(init) - Ul_i(init)|$ with the error bound). If $|V_i(init) - U_i(init)|$ (or $|Vl_i(init) - Ul_i(init)|$) is no less than the error bound, meaning that the error is NOT too large, the algorithm terminates; otherwise the algorithm returns $TRY\ \ SMALLER\ \epsilon$. When the algorithm terminates, if $V_i(init) \geq U_i(init)$, the algorithm returns $PASS$, meaning that the specification is a Nash-equilibrium, otherwise, the algorithm returns $FAIL$ meaning that the specification is not a Nash-equilibrium.

Theoretically, if the iteration number is not bounded with $iter_{max}$, the algorithm should terminate with $PASS$ or $FAIL$, for any scenario where $V_i(init) - U_i(init)$ converges to a non-zero value. If $V_i(init) - U_i(init)$ converges to zero, then the algorithm is not guaranteed to terminate. It terminates only if $|V_i(init, iter) - U_i(init, iter)| = max_{Serror}$ ($|Vl_i(init, iter) - Ul_i(init, iter)| = max_{Aerror}$ correspondingly) for some $iter$ value.

IV. Case study

We use our algorithm on a case study with perfect information, the three-player Rock-Paper-Scissors, and on an other with imperfect information, the secret sharing protocol from Halpern and al. [7].

A. Rock-Paper-Scissors game

We analyse two strategies of the Rock-Paper-Scissors game in Example 1: each player has equally distributed probability on playing rock, paper and scissors ($1/3$ for playing rock, $1/3$ for playing paper, and $1/3$ for playing scissors); and each player has unequal probability distribution ($1/5$ for playing rock, $1/5$ for playing paper, and $3/5$ for playing scissors). For each strategy, we analyse two scenarios: no Byzantine players, and having one Byzantine player.

1) Equal probability strategy:

a) No Byzantine Players: When there is no Byzantine players ($m = 0$), to verify whether the equally distributed probability strategy is a Nash-equilibrium, we actually need to verify the following conditions, according to the definition III in the previous section.

1) $U_1(\langle s_1^a, s_1^a, s_1^a \rangle) \geq V_1(\langle s_1^r, s_1^a, s_1^a \rangle)$
2) $U_2(\langle s_1^a, s_1^a, s_1^a \rangle) \geq V_2(\langle s_1^a, s_1^r, s_1^a \rangle)$
3) $U_3(\langle s_1^a, s_1^a, s_1^a \rangle) \geq V_3(\langle s_1^a, s_1^a, s_1^r \rangle)$

Symbol s_1^a is the initial state for an altruistic player as defined in the altruistic player modelling in Section II-A. Similarly, s_r^1 is the initial state for a rational player. Assuming three altruistic players, the global initial state is $\langle s_1^a, s_1^a, s_1^a \rangle$. When the first player is a rational player, the initial global state is $\langle s_1^r, s_1^a, s_1^a \rangle$. $U_1(\langle s_1^a, s_1^a, s_1^a \rangle)$ is the maximum guaranteed expected pay-off for player p_1 when p_1 is altruistic; $V_1(\langle s_1^r, s_1^a, s_1^a \rangle)$ is the maximum guaranteed expected pay-off for player p_1 when p_1 is rational, i.e., may deviate from the equal probability strategy. Hence, the first condition says that player p_1 would not deviate from the specified strategy since the deviation pay-off is smaller. Similar, the second condition ensures that p_2 would not deviate and the third condition ensures that p_3 would not deviate.

Since the behaviour of all altruistic players are exactly the same, the three conditions are symmetrical. Hence, it is sufficient to prove one of the above conditions. Assume the game only ends, only if there is a unique winner. The verification result shows that the equal probabilistic strategy is a Nash-equilibrium. We provide the intuition of the result by illustrating the first condition, i.e., reasoning on player p_1's behaviour, as follows.

For each iteration, we need to the following two cases, corresponding to $U_1(\langle s_1^a, s_1^a, s_1^a \rangle)$ and $V_1(\langle s_1^r, s_1^a, s_1^a \rangle)$ respectively. We show that the probability of ending the game in both of the two cases are the same and the pay-off of p_1 in both cases are the same as well.

- *Player p_1 is altruistic.* Since each player has three actions (plying rock, paper, or scissors) and the probability of each action is $\frac{1}{3}$, there are in total 27 outcomes and each outcome has probability $\frac{1}{3} \times \frac{1}{3} \times \frac{1}{3} = \frac{1}{27}$. Among all the outcomes, there are 9 outcomes that lead to a unique winner, which end the game. Thus, the probability that the game ends in one iteration is $9 \times \frac{1}{27} = \frac{1}{3}$. The expected pay-off of p_1 is $\frac{1}{27} * ep_a^1 + \ldots + \frac{1}{27} * ep_a^{27}$ where ep_a^i is the expected pay-off for outcome i. We show that the value of the expected pay-off is 0. For instance, if we fix p_1's action as playing rock 0, the outcomes together with the pay-offs are showing in Table I. Since, the probability of each outcome is the same, the expected pay-off in this case is obviously 0 (Pay-offs cancel out, see Table I). Similarly when p_1 plays other actions. Hence the total pay-off for p_1 is 0.
- *Player p_1 is rational.* Since p_1 is rational, he can choose any action with any probability. To show the ending probability of the game, we show that for any action of p_1, the probability to end the game is exactly $\frac{1}{3}$ as follows. Given a certain action of p_1, there are 9 outcomes and each outcome has probability $\frac{1}{3} \times \frac{1}{3} = \frac{1}{9}$. Among these 9 outcomes, there are 3 outcomes with a unique winner. Hence, the probability of having a unique winner for any choice of player p_1 is $3 \times \frac{1}{9} = \frac{1}{3}$. Similarly, for calculating the pay-offs, given one action of p_1, her expected pay-off is 0 (see from Table I). Hence, for any probabilistic distribution of p_1's actions, the expected pay-off is 0.

TABLE I
PAY-OFFS OF PLAYER p_1 WHEN PLAYING ROCK

i	0	0	0	0	0	0	0	0	0
j	0	0	0	1	1	1	2	2	2
k	0	1	2	0	1	2	0	1	2
pay-off	0	-1	1	-1	-2	0	1	0	2

TABLE II
PAY-OFFS OF PLAYER p_2 WHEN p_1 PLAYS ROCK

i	0	0	0	0	0	0	0	0	0
j	0	0	0	1	1	1	2	2	2
k	0	1	2	0	1	2	0	1	2
pay-off	0	-1	1	2	1	0	-2	0	-1

When the iteration is infinite, $U_1(\langle s_1^a, s_1^a, s_1^a \rangle)$ is calculated as $ep_a + \frac{2}{3} * ep_a + \frac{2}{3}^2 * ep_a + \ldots$ where ep_a is the expected pay-off of each iteration, $\frac{2}{3}$ is the probability of continue the game. Since ep_a is 0, $U_1(\langle s_1^a, s_1^a, s_1^a \rangle) = 0$. Similarly, $V_1(\langle s_1^r, s_1^a, s_1^a \rangle) = 0$. Thus, the first condition $U_1(\langle s_1^a, s_1^a, s_1^a \rangle) \geq V_1(\langle s_1^r, s_1^a, s_1^a \rangle)$ is satisfied. The other two conditions can be proved in a similar way, especially they are symmetrical to the first condition.

b) One Byzantine Player: Assume player p_1 is Byzantine ($m = 1$), to verify whether the equally distributed probability strategy is a Nash-equilibrium, we need to verify the following conditions, according to the definition III.

1) $U_2(\langle s_1^b, s_1^a, s_1^a \rangle) \geq V_2(\langle s_1^b, s_1^r, s_1^a \rangle)$
2) $U_3(\langle s_1^b, s_1^a, s_1^a \rangle) \geq V_3(\langle s_1^b, s_1^a, s_1^r \rangle)$

The verification result shows that the equal probabilistic strategy is a Nash-equilibrium as well when p_1 is Byzantine. The intuitions are as follows.

Similarly, for each iteration, we consider the following two cases. Like the previous case with no Byzantine player, the probability of ending the game in both cases are the same $\frac{1}{3}$. Unlike the previous case, the pay-off in the two cases differ from each other.

- *Player p_2 is altruistic.* Recall that the Byzantine player only has non-deterministic actions at each state. According to the algorithm, we always consider the action of the Byzantine player that leads to the worse case of pay-off for player p_2, i.e., we only consider the p_1's action that minimize p_2's expected pay-off.For each action of Byzantine player p_1, we calculate the the expected pay-off of p_2. For example, if p_1 plays rock 0, the pay-offs of p_2 for each outcome are shown in Table II. The probability of each outcome is exactly $\frac{1}{9}$. Hence the expected pay-off of p_2 is 0. Similarly, for the other two actions of p_1, the expected pay-offs of p_2 are 0 as well.
- *Player p_2 is rational.* Since p_2 is rational, her actions can have any probabilistic distribution. Similar to the case with no Byzantine player, we consider the pay-off for each action of p_2. Technically, we consider the pay-off of p_2 given any given combination of p_1's action and p_2's action. Recall that we only need to consider p_1's action that leads to the worse case for p_2. Given p_2's

TABLE III
PAY-OFFS OF PLAYER p_2 WHEN PLAYING ROCK

i	1	1	1
j	0	0	0
k	0	1	2
pay-off	-1	-2	0

action, p_1's action is decided. For example, when p_2 plays rock, the action of p_1 that leads to the worst case for p_2 is playing paper. The pay-offs of p_2 for each outcome is given in Table III. Hence, the expected pay-off of p_2 when playing rock $\frac{1}{3} \times (-1 + -2 + 0) = -1$. For the other two actions of p_2, the expected pay-off is calculated in a similar way. Due to the symmetric property, the expected pay-off of p_2 when playing other two actions are -1 as well. Hence, the expected pay-off of p_2 is -1, no matter which probabilistic distribution of p_2's actions is. Note that unlike the previous case with no Byzantine players, the rational player's pay-off decreased when a Byzantine player is presented.

Therefore, $U_1(\langle s_1^b, s_1^a, s_1^a \rangle) = ep_a + \frac{2}{3} * ep_a + \frac{2}{3}^2 * ep_a + \ldots$, where ep_a is the expected pay-off of p_2 in each iteration. Since $ep_a = 0$, we have $U_1(\langle s_1^b, s_1^a, s_1^a \rangle) = 0$. Similarly, $V_1(\langle s_1^b, s_1^r, s_1^a \rangle) = -1 + \frac{2}{3} * (-1) + \frac{2}{3}^2 * (-1) + \cdots \approx -3$ meaning that the value converges to -3. Thus, we have $U_1(\langle s_1^b, s_1^a, s_1^a \rangle) \geq V_1(\langle s_1^b, s_1^r, s_1^a \rangle)$. It means that the equal probability strategy is a Nash-equilibrium, when p_1 is Byzantine.

2) Unequal probability strategy: Given the probabilistic strategy of the altruistic player $\langle \frac{1}{5}, \frac{1}{5}, \frac{3}{5} \rangle$, we analyse the above two scenarios – without Byzantine players and with one Byzantine player. Differing from the case of equal probability strategy, in the unequal probability strategy, the probability of each outcome becomes unequal as well. This introduces difficulties in calculation of the expected pay-offs. For the simplicity of calculation, we group the outcomes with the same probability and obtain the following classes, such that we do not need to consider the total 27 outcomes every time.

- Unique winner and the winner plays 0
- Unique winner and the winner plays 1
- Unique winner and the winner plays 2
- Unique loser and the loser plays 0
- Unique loser and the loser plays 1
- Unique loser and the loser plays 2
- Each player plays 0
- Each player plays 1
- Each player plays 2
- other cases

a) Without Byzantine.: Similarly, we show that in each of the above classes, the expected pay-off of any altruistic play is 0. For example, in the first class – unique winner and the winner plays 0, there are 3 cases: p_1 wins, p_2 wins, or p_3 wins. An altruistic player p_1's pay-offs are 2, -1 and -1 respectively. In this class the probability of each outcome is the same. Thus, the expected pay-off of p_1 is 0. Similarly, other players' pay-offs are 0 as well. Similarly, the expected pay-offs of any altruistic player is 0 in other classes. Considering all the classes, the expected pay-offs of an altruistic player is 0.

Next, we calculate the expected pay-off of a rational player using a similar way as in the equal probability strategy. Assuming p_1 is rational, given that p_1 plays rock, there are 9 outcomes. The pay-off of each outcome is shown in Table I. However, the probability of each outcome is unequal. We calculate the probabilities for each outcome and obtain the expected pay-off of p_1 is 0.8, and calculate the probability of restarting the game as $\frac{86}{125}$. Hence, for each iteration, the expected pay-off of p_1 is no less than 0.8. If any other action of p_1 leads to a pay-off less than 0.8, p_1 will play rock instead of the action for achieving better pay-off. Hence, the expected pay-off of p_1 in each iteration is no less than 0.8. For infinite number of iterations, the expected pay-off of p_1 is obviously larger than 0. This implies that $V_1(\langle s_1^r, s_1^a, s_1^a \rangle) > U_1(\langle s_1^a, s_1^a, s_1^a \rangle)$. Therefore, the strategy $\langle \frac{1}{5}, \frac{1}{5}, \frac{3}{5} \rangle$ is NOT a Nash-equilibrium, when there is no Byzantine player.

b) With one Byzantine player.: Assume player p_1 is Byzantine, unlike the case with equal probability strategy, the unequal probability strategy provides interesting and counter-intuitive scenarios. We cannot simply consider one iteration and derive the Nash-equilibrium by extending it to multiple iterations. Essentially, the probability of ending the game plays more important role in the calculation. An action of rational player which leads to bigger pay-off than other actions in one iteration may not always lead to a bigger pay-off in the long run, in the case that the action leads to a significant larger probability to restart the game. The converged value for this case can be smaller than a slightly smaller pay-off per iteration, but the action that leads to the smaller pay-off (possibly negative) leads to small probability of restarting.
In the algorithm, $U_i(\langle s_1^b, s_1^a, s_1^a \rangle, iter)$ was approximately equal to $\frac{-10}{11}$ when p_2 is altruistic and $V_i(\langle s_1^b, s_1^r, s_1^a \rangle, iter)$ was approximately equal to $\frac{-5}{3}$ when p_2 was rational. The algorithm terminated by returning the result $PASS$. Though we do not have a rigorous proof to find the convergence value of $U_i(\langle s_1^b, s_1^a, s_1^a \rangle) - V_i(\langle s_1^b, s_1^r, s_1^a \rangle)$ for this particular example, the algorithm anyway implies that the strategy $\langle \frac{1}{5}, \frac{1}{5}, \frac{3}{5} \rangle$ is a Nash-equilibrium, when p_1 is Byzantine.

B. Probabilistic secret sharing

Secret sharing is a way for a group of users to share a secret: the secret is divided into many parts, named *shares*, and each member has a share; the secret can only be reconstructed with a sufficient number (threshold m) of shares. In other words, the secret is kept confidential, even in the presence of a limited number of traitors (compromised members or non-cooperating members who prevent the reconstruction of the secret). This property makes secret sharing an ideal scheme for storing high sensitive information, such as encryption keys and launch codes. In the parametric form, a n-m ($n \geq m$) secret sharing ensures that it requires at least m honest (here altruistic) players to reconstruct the secret within a group of n players.

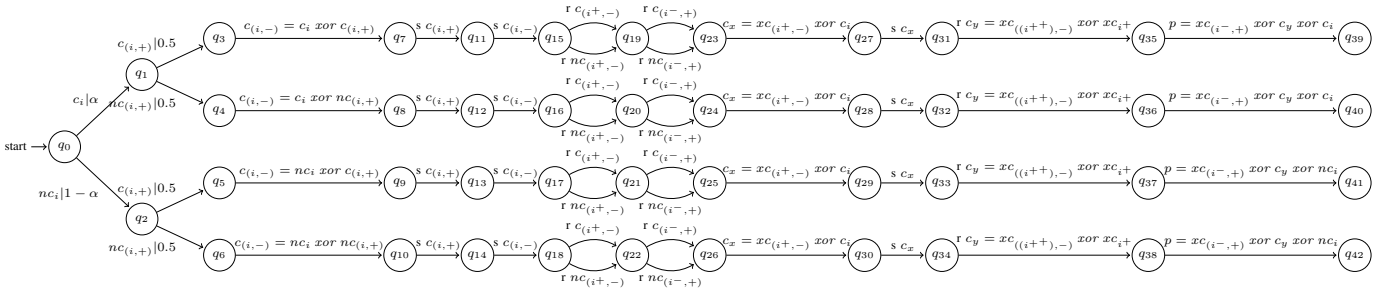

Fig. 1. Altruistic player in probabilistic secret sharing

That is, less than m compromised players together cannot reconstruct the secret; and less than $n - m$ non-cooperating members together cannot stop the honest players to reconstruct of the secret.

The most well-known secret sharing scheme is the Shamir's secret sharing [8]. However, the scheme does not work when the players are rational – a player follows the protocol only if it increases her utility. It is shown that a rational player will not share her secret with anyone else. [9]. In fact, it is proved that in general, in all practical mechanisms for shared-secret reconstruction with an upper bound on the running time, the best strategy for each rational player is to deviate from the specification by doing nothing [9]. Hence, a randomized mechanism (the probabilistic secret sharing scheme) is proposed to ensure that rational players would reveal their shares to reconstruct the secret, i.e., achieving a Nash-equilibrium with a constant expected running time.

1) Protocol description: The probabilistic scheme is a 3-3 secret sharing. Given the 3 players, we index each player is indexed with a distinct number in $\{1, 2, 3\}$. We use i to denote one player, then the other two players are $(i+1)\%3$, denoted as i^+, and $(i-1)\%3$, denoted as i^-. There is an issuer who assign each player a share of secret. Assuming the share cannot be changed; for example, it is signed by the issuer. The scheme works as follows.

1) Each player i
   - chooses a bit c_i (c_i can only be 1 or 0) such that $c_i = 1$ with probability α and $c_i = 0$ with probability $1 - \alpha$;
   - chooses a bit $c_{(i,+)}$ ($c_{(i,+)}$ can only be 1 or 0) randomly, so that $c_{(i,+)} = 1$ with probability $1/2$ and $c_{(i,+)} = 0$ with probability $1/2$;
   - calculates $c_{(i,-)} = c_i \; xor \; c_{(i,+)}$.

   Then player i sends the bit $c_{(i,+)}$ to player i^+ and sends the bit $c_{(i,-)}$ to player i^-. This means that the player i receives a bit $c_{(i^+,-)}$ from player i^+ and a bit $c_{(i^-,+)}$ from player i^-.
2) On receiving $c_{(i^+,-)}$, player i computes $c_{(i^+,-)} \; xor \; c_i$ and sends it to player i^-. Correspondingly, i receives $c_{((i^+)^+,-)} \; xor \; c_{i^+}$ from i^+. Since there are only 3 players, player $(i^+)^+$ is the player i^-. Hence $c_{((i^+)^+,-)} \; xor \; c_{i^+} = c_{(i^-,-)} \; xor \; c_{i^+}$.
3) When receiving both $c_{(i^-,+)}$ from player i^- (step 1) and receiving $c_{(i^-,-)} \; xor \; c_{i^+}$ from i^+ (step 2), i computes $p = c_{(i^-,+)} \; xor \; (c_{(i^-,-)} \; xor \; c_{i^+}) \; xor \; c_i$. Note that $c_{(i^-,+)} \; xor \; (c_{(i^-,-)} \; xor \; c_{i^+}) \; xor \; c_i = c_{i^-} \; xor \; c_{i^+} \; xor \; c_i$, i.e., $p = c_1 \; xor \; c_2 \; xor \; c_3$ (It holds for all players). If $p = c_i = 1$ then player i sends her share to the others.
4) Depending on the value of p and c_i, player i decides whether to send her share.
   - If $p = 1$, i receives 3 shares, then he construct the secret.
   - If $p = 0$ and i received no secret shares, or if $p = 1$ and i received exactly one share (possibly from itself; that is, we allow the case that i did not receive any shares from other players but sent its own), the issuer restarts the protocol.
   - Otherwise, player i stops the protocol. Because someone must have been cheating.

   The intuition behind the last step is as follows: When $p = 0$, no one should send her share; and thus i receives no secret share. Otherwise, some one calculated wrong (cheating). When $p = 1$ either $c_i = c_{i^+} = c_{i^-} = 1$ or there exists exactly one player chooses 1, i.e., $c_i = c_{i^+} = 0$ and $c_{i^-} = 1$, $c_i = c_{i^-} = 0$ and $c_{i^+} = 1$, or $c_{i^+} = c_{i^-} = 0$ and $c_i = 1$. In the former case, each player can construct the secret. In the later case, i should receive exactly one share. Otherwise, some player calculated wrong (cheating).

2) Modelling: The model of the altruistic players who follow the protocol specification is straight forward. For the sake of space saving, we present the FMS for altruistic players in the graphical manner as in Figure 1. In the Figure, we use 's' to stand for sending and 'r' to stand for receiving, use nc_i to represent the opposite value of c_i; the value after symbol | is the probability of the action, for example, $c_i|\alpha$ stands for the probability of choosing the value c_i is α. For simplicity of presentation, we merged the states and traces caused by received values and use the prefix xc to represent to represent either c or nc depending on which one is received. The payoff function is defined as follows:

$$SH_i^a(S) = \begin{cases} 3 & \text{if only the player gets secret} \\ 2 & \text{if the player and another player get secret} \\ 1 & \text{if everyone knows the secret} \\ 0 & \text{if no one knows the secret} \\ -1 & \text{if two other players know the secret} \\ -2 & \text{if only one other player knows the secret} \end{cases}$$

The Byzantine players deviate from the protocol specification by sending the opposite value of the one that the player should send. For example, in a Byzantine player's model, we add a transition from q_7 to q'_{11} labelled with $nc_{(i,+)}$, and copy the transition from q_{11} to q_{39} as q'_{11} to q'_{39}. Similar for other states which enables sending actions. Due to space limit, we omit the figure for Byzantine players. The model of rational players are the model of Byzantine players together with the above pay-off function.

Notice that we fixed the order of all players' sending and receiving actions such that the three players' sending and receiving actions can be synchronised. That is, we did not model the full interleaving of all the possible orders of sending and receiving actions of the three players. We show that this does not influence the verification result.

3) Verification results: We verified whether a player deviates from the specification in two settings: without Byzantine player and with one Byzantine player. Essentially, our algorithm calculates the pay-off of a player in the following four configurations and compare the pay-offs.

1) All the three players are altruistic ($m = 0$)
2) Two players are altruistic and one player is rational ($m = 0$)
3) Two players are altruistic and one player is Byzantine ($m = 1$)
4) One player is altruistic, one player is Byzantine and one player is rational ($m = 1$)

The verification result shows that the probabilistic secret sharing protocol is a Nash-equilibrium when there is no Byzantine player, which confirms the manual analysis in its original paper. In addition, the verification result shows that even with a Byzantine player, the probabilistic secret sharing scheme is still a Nash-equilibrium, which has not been proved in existing works.

V. Conclusion

In this article, we presented a framework for probabilistic BAR systems. We introduced an algorithm to verify Nash-equilibrium properties in concurrent games with perfect and imperfect information at the level of global state knowledge. We performed our algorithm on two case studies, the 3-player Rock-Paper-Scissors example and Halpern and al.'s Secret Sharing protocol. Some results are interesting as they illustrate the necessity of a Byzantine player in a game to "force" the rational player to behave altruistically notably in the Rock-Paper-Scissors example. This opens a new door for Byzantine players, currently only considered as misconfigured or corrupted agents. Our algorithm is guaranteed to terminate (due to the maximum number of iterations) but the convergence may be not guaranteed if V_i and U_i converge to the same value. The analysis of this limit case is left to future work.

References

[1] A. S. Aiyer, L. Alvisi, A. Clement, M. Dahlin, J. Martin, and C. Porth, "BAR Fault Tolerance for Cooperative Services," in *Proceedings of the 20th ACM Symposium on Operating Systems Principles 2005, SOSP 2005, Brighton, UK, October 23-26, 2005*, 2005, pp. 45–58.

[2] H. C. Li, A. Clement, E. L. Wong, J. Napper, I. Roy, L. Alvisi, and M. Dahlin, "BAR Gossip," in *7th Symposium on Operating Systems Design and Implementation (OSDI '06), November 6-8, Seattle, WA, USA*, 2006, pp. 191–204.

[3] F. Mari, I. Melatti, I. Salvo, E. Tronci, L. Alvisi, A. Clement, and H. C. Li, "Model Checking Nash Equilibria in MAD Distributed Systems," in *Formal Methods in Computer-Aided Design, FMCAD 2008, Portland, Oregon, USA, 17-20 November 2008*, 2008, pp. 1–8.

[4] R. Brenguier, "PRALINE: A tool for computing nash equilibria in concurrent games," in *Proc. 25th Computer Aided Verification*, ser. LNCS, vol. 8044. Springer, 2013, pp. 890–895.

[5] T. Chen, V. Forejt, M. Z. Kwiatkowska, D. Parker, and A. Simaitis, "Prism-games: A model checker for stochastic multi-player games," in *Proc. 19th Tools and Algorithms for the Construction and Analysis of Systems*, ser. LNCS, vol. 7795. Springer, 2013, pp. 185–191.

[6] R. Pass and J. Halpern, "Game Theory with Costly Computation: Formulation and Application to Protocol Security," in *Proceedings of the Behavioral and Quantitative Game Theory - Conference on Future Directions, BQGT '10, Newport Beach, California, USA, May 14-16, 2010*, 2010.

[7] I. Abraham, D. Dolev, R. Gonen, and J. Halpern, "Distributed computing meets game theory: robust mechanisms for rational secret sharing and multiparty computation," in *Proceedings of the twenty-fifth annual ACM symposium on Principles of distributed computing*. ACM, 2006, pp. 53–62.

[8] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[9] J. Halpern and V. Teague, "Rational secret sharing and multiparty computation: Extended abstract," in *Proc. 36th Annual ACM Symposium on Theory of Computing*. ACM, 2004, pp. 623–632.