Automated Game-Theoretic Verification of Security Systems

Chunyan Mu *

Department of Computer Science, Teesside University, UK
c.mu@tees.ac.uk

Abstract. Security-sensitive computerised communication systems are of increasing importance, however checking that they function correctly can be non-trivial. We propose automated verification techniques for the formal analysis of quantitative properties of such systems. Since communication networks typically require the collaboration of their participants to work effectively, we adopt a game-theoretic approach. Utility functions for each player, such as the degree of security offered and the communication costs incurred, are formally specified using quantitative temporal logics. Then, building upon probabilistic verification techniques for parametric Markov chains, we develop methods to identify Nash equilibria representing stable strategies for the participants. We implement our methods as an extension of the PRISM model checker, and illustrate their applicability by studying anonymity-cost trade-offs in the Crowds anonymity protocol.

Keywords: Quantitative verification, Game theory, security.

1 Introduction

Security properties have become essential requirements in today's computerised communication systems. Absolute guarantees on such properties are often impractical in real life, so we may instead tolerate a loss of anonymity or privacy in a system with low probability. Furthermore, system designs often need to trade off the degree of security offered against other practical concerns such as response time or power consumption. So, effective methods for the analysis of security also need to take quantitative aspects into account.

In this work, we present novel automatic verification techniques for the formal modelling and analysis of security properties in communication networks. Since such systems generally rely on the collaboration of their participants to work effectively, we adopt a game-theoretic approach to verification. We propose a new framework in which systems are modelled as *n-player parametric Markov chain games*, where each decision-maker chooses the value of a parameter, which

* The author thanks David Parker for many suggestions, help and insightful discussions. This work is supported in part by EPSRC (EP/K038575/1), and was partially performed when the author was at University of Birmingham.

is used to define the transition probabilities of a Markov chain. These parameters allow participants in the system to make strategic probabilistic decisions about their behaviour, such as the probability which they will act in a co-operative or unfriendly or malicious fashion during the execution of a security protocol. Since we model system behaviour in a probabilistic fashion, we can also capture a variety of other important stochastic aspects of a system, such as message loss, failures, or other sources of randomisation.

We apply game-theoretic notions and methods to study the behaviour of interacting decision-makers. We define utility functions for players capturing their preferences regarding the different system outcomes, and then define the expected value of these utility functions using our Markov chain models. Individual players can decide how to behave in order to maximise their utility, although their choices will typically influence the outcomes for other players too. In order to investigate the effectiveness of a protocol (e.g., for anonymity) that requires some cooperation between multiple individuals with conflicting objectives, we use the concept of Nash equilibria [14]. These represent the existence of situations where no system player can benefit by changing their own strategy, assuming that the other players keep their strategies unchanged.

We propose techniques to formally specify games and utility functions and to automatically compute Nash equilibria for them. We build upon existing techniques and tools for probabilistic model checking, which is a widely used technique for modelling and automatically verifying quantitative properties of systems with stochastic behaviour. In particular, we build upon *parametric probabilistic model checking* methods [6], in which transition probabilities of models can be given as functions over parameters, and an analysis of these models can yield results expressed as symbolic functions over parameters. We use probabilistic temporal logics as a means of specifying utility functions for individual players and then use parametric model checking to determine functions representing the expected utility. We then generate and solve a set of polynomial equations, the solutions to which yield Nash equilibria for the system.

We developed an implementation of this approach using the parametric model checking functionality of the PRISM model checker [10] and solving polynomial systems using the polyhedral homotopy continuation based PHCpack [21]. We describe how our approach can be used to model and analyse properties of security-sensitive communicating networks. In particular, we illustrate this on the Crowds [16] anonymity protocol, considering the trade-offs between the degree of anonymity provided and the corresponding communication cost.

Related work. Multiple efforts have been made to develop methods for game-theoretic analysis of communicating networks for security concern. Yang et. al [22] proposed a game theoretic framework to analyse users' behaviours in anonymity networks. Performance utilities were modelled as a combination of weighted cost and anonymity utilities. Simulations were performed to show the impact of users' cooperation level and the weights of the anonymity and cost factors to the optimisation of their utilities. However, this work did not compute the

Nash equilibria in an automatic way. In our work, we use exact solution methods, rather than simulation, and focus on automated methods to find equilibria.

Venkitasubramaniam [20] investigated the problem of maximising security properties from a game-theoretic perspective. The problem was formalised as a two-player zero-sum game between the network designer and the adversary. Given the adversary's observation, the anonymity degree was measured using conditional entropy of the routes. The adversary tried to choose a subset of the nodes to monitor in order to minimise the anonymity of the routes, while the designer aimed to maximise the anonymity. They did not deal with the problem of computing equilibria of games with multiple players with regard to performance analysis, however, which has been considered in our work.

Formal methods have played a significant important role in modelling and analysing security protocols. They include two main categories: proof-based theorem proving and state-exploration based model checking. Specifically, we focus on quantitative analysis of security properties. Mhamdi et. al [12] introduced two measures of information leakage: the information leakage degree and the conditional information leakage degree, to evaluate the anonymity and privacy properties of protocols. A theorem prover was applied to conduct a probabilistic and information-theoretic analysis for the evaluation of the anonymity and privacy properties. However, they did not tackle the problem of how the users should behave in order to optimise the security and performance properties from a game-theoretic point of view.

On the other hand, Shmatikov [17,18] applied the PRISM model checker to model and analyse the Crowds protocol for anonymity properties. By modelling the system behaviour as a discrete-time Markov chain, and formalising the anonymity properties in PCTL, the PRISM model checker was employed to perform automated probabilistic analysis and verify anonymity properties quantitatively. However, they did not study the problem of strategy decision-making or attempt a game-theoretic analysis of the system. Approaches of computing a Nash equilibrium in Stochastic games have been studied in [3,19]. It was not straightforward to adapt them in modelling and analysing security systems. Our framework can be naturally applied in such systems to synthesize optimal decision strategies.

2 Preliminaries

2.1 Game theory

We first recall some required definitions from game theory [2,15], beginning with the definitions of convexity and some related notions.

Definition 1 (Convex set). *A set S of vectors over real numbers is* convex *if $(1-\lambda)x + \lambda x' \in S$ whenever $x, x' \in S$, and $\lambda \in [0,1]$.*

Definition 2 (Upper level set). *Let f be a multivariate function defined on a set S. For $a \in \mathbb{R}$, the* upper level set *of f for a is $P_a = \{x \in S : f(x) \geq a\}$.*

Definition 3 (Quasi-concave function). *A multivariate function f defined on a convex set S is* quasi-concave *if every upper level set of f is convex.*

Definition 4 (Preference relation). *A* preference relation $\succeq$ *over a set S is a total, transitive and reflexive binary relation over S.*

Definition 5 (Strategic game). *A* strategic game $\langle N, (A_i), (\succeq_i)\rangle$ *consists of: a finite set of players N; for each player $i \in N$: a non-empty set of actions A_i available to i, and a preference relation $\succeq_i$ on $A = \times_{j\in N} A_j$ of i. The game is* finite *if the sets A_i of actions for player i are all finite.*

A *strategy* for player i is a choice of an action $a_i \in A_i$ and a *strategy profile* $\sigma = (a_1, \ldots, a_n) \in A$ is a choice of actions for all players. We write σ_i for the choice of player i in σ, σ_{-i} for the choices of all players except i, and (σ_{-i}, a_i) for the strategy profile that combines the choices from σ_{-i} and some $a_i \in A_i$.

Definition 6 (Nash equilibrium). *A* Nash equilibrium *of a strategic game $\langle N, (A_i), (\succeq_i)\rangle$ is a strategy profile $\sigma^* \in A$ with the property that, for every player $i \in N$ and strategy $a_i \in A_i$, we have $(\sigma^*_{-i}, \sigma^*_i) \succeq_i (\sigma^*_{-i}, a_i)$.*

Definition 7 (Best response function). *For any σ_{-i}, the* best response function *$B_i(\sigma_{-i})$ is defined as the set of player i's best actions given σ_{-i}:*

$$B_i(\sigma_{-i}) = \{a_i \in A_i : \forall a'_i \in A_i \,.\, (\sigma_{-i}, a_i) \succeq_i (\sigma_{-i}, a'_i)\}.$$

Note that, in the above, players' preferences with respect to strategies are defined by a preference relation $\succeq_i$. In the remainder of the paper, for modelling convenience and in order to allow players' choices to be probabilistic, we will instead use a *utility function* $u_i : A \to \mathbb{R}$ for each player i, which it aims to maximise. The basic strategic game can then be rewritten as a tuple $\langle N, (A_i), (u_i)\rangle$.

Proposition 1 ([15]). *The strategic game $\langle N, (A_i), (\succeq_i)\rangle$ has a Nash equilibrium if, for all $i \in N$: (i) the set A_i of actions of player i is a non-empty compact convex subset of Euclidian space; and (ii) the corresponding utility function u_i of the preference relation $\succeq_i$ is continuous and quasi-concave on A_i.*

In this paper, we model the communication systems in Markov chain games and discuss the existence of Nash equilibria for this model. Proposition 1 presents the requirements for the game to have a Nash equilibrium.

2.2 Parametric Markov chains

In this paper, we build on parametric model checking techniques for the model of parametric Markov chains [6]. We briefly review some relevant background.

Definition 8 (Rational function). *Let $V = \{x_1, \ldots, x_n\}$ be a finite set of* variables *with domain $\mathbb{R}$. A* polynomial *g over V is constructed via the following grammar:*

$$g ::= c \mid x \mid g + g \mid (g) \cdot (g)$$

where $c \in \mathbb{R}$, $x \in V$, $+$ and $\cdot$ are the standard addition and multiplication respectively. A rational function *f over V is a fraction of two polynomials g_1 and g_2 over V such as: $f = \frac{g_1}{g_2}$ where g_2 is not reducible to 0.*

Definition 9 (Evaluation). *Let $\mathbb{R}[x_1, \ldots, x_n]$ be the set of polynomials over the set of variables $V = \{x_1, \ldots, x_n\}$, let $\mathcal{F}_V : \mathbb{R}[x_1, \ldots, x_n] \to \mathbb{R}$ denote the* set of rational functions*, and $dom(f)$ denote the domain of function f. An* evaluation *$\mathcal{V} : X \to \mathbb{R}$ is a function for a subset $X \subseteq V$. For a rational function $f \in \mathcal{F}_V$, $f[X/\mathcal{V}]$ denotes the function obtained by substituting each $x \in (X \cap dom(\mathcal{V}))$ with its evaluation $\mathcal{V}(x)$.*

A *parametric Markov chain* is an extension of a discrete-time Markov chain, using rational functions instead of real numbers to label transition probabilities.

Definition 10 (Parametric Markov chains). *A parametric Markov chain is a tuple $\mathcal{M} = (S, I, \Delta, V, AP, L)$, where S is a countable set of states, $I : S \to \mathcal{F}_V$ is the initial distribution such that $\sum_{s \in S} I(s) = 1$, $\Delta : S \times S \to \mathcal{F}_V$ is the parametric transition probability matrix such that $\forall s \in S. \sum_{s' \in S} \Delta(s, s') = 1$, $V = \{x_1, \ldots, x_n\}$ is a finite set of parameters with domain $\mathbb{R}$, AP is a finite set of atomic propositions, and $L : S \to 2^{AP}$ is a labelling function mapping each state to a set of atomic propositions taken from a set AP.*

For an evaluation $\mathcal{V}$ and a parametric Markov chain $\mathcal{M}$, an *induced* parametric Markov chain $\mathcal{M}_\mathcal{V}$ is defined by substituting each variable in $dom(\mathcal{V})$ with its evaluation. By applying a total evaluation $\mathcal{V}$ with $dom(\mathcal{V}) = V$, we obtain real values for each probability instead of rational functions. Let $\texttt{Prob}^\mathcal{M} \subseteq \mathcal{F}_V$ denote the set of probabilities of $\mathcal{M}$, such as

$$\texttt{Prob}^\mathcal{M} := \{I(s) | s \in S\} \cup \{\Delta(s, s') | s, s' \in S\},$$

and similarly $\texttt{Prob}^\mathcal{M}_\mathcal{V} \subseteq \mathcal{F}_{V/\ dom(\mathcal{V})}$ for an evaluation $\mathcal{V}$. A total evaluation $\mathcal{V}$ of $\mathcal{M}$ is called *well-defined* if:

- $\forall f \in \texttt{Prob}^\mathcal{M}_\mathcal{V} : f[dom(\mathcal{V})/\mathcal{V}] \in [0, 1]$, which ensures every possible evaluation is a probability;
- $\forall f \in \texttt{Prob}^\mathcal{M}_\mathcal{V} : f \neq 0 \Leftrightarrow f[dom(\mathcal{V})/\mathcal{V}] \neq 0$, which ensures every non-zero rational function does not evaluate to 0.

A parametric Markov chain $\mathcal{M}$ can be viewed as a state transition system in which transitions are associated with parametric probabilities indicating their likelihood. We say there is a *transition* from state $s \in S$ to $s' \in S$ iff $\Delta_\mathcal{V}(s, s') > 0$ for all well-defined evaluations $\mathcal{V}$. A (finite or infinite) *path* describes one possible execution of $\mathcal{M}$, and is defined as a sequence of states $\rho = s_0, s_1, \ldots$ such that $\forall i \geq 0. \Delta(s_i, s_{i+1}) = f(X) > 0$, where $f(X)$ is a polynomial over $X \subseteq V$.

Let $\Omega_{\mathcal{M},s}$ denote the set of paths of $\mathcal{M}$ starting from state s (we omit the s when referring to all such paths). In order to reason about the behaviour of $\mathcal{M}$, it is required to formalise the probability of sets of paths taken. The construction is based on calculating the probability of individual finite paths induced by the parametric transition probability matrix Δ. The probability of the path $\rho = s_0, \ldots, s_k$ is given by: $Pr(\rho) \triangleq \prod_{i=0}^{k-1} \Delta(s_i, s_{i+1})$, where $\rho \in \Omega_{\mathcal{M},s_0}$.

2.3 Probabilistic temporal logics

For the purposes of probabilistic verification, properties to be checked against a model are typically specified in probabilistic extensions of temporal logic. In this paper we use the property specification language of the PRISM tool [9], the basis for which is the logic PCTL [7], plus various notions of reward.

Definition 11 (PCTL with rewards). *The syntax of PCTL with rewards is given by the grammar:*

$$\phi ::= \texttt{true} \mid a \mid \neg\phi \mid \phi \wedge \phi \mid P_{\bowtie q}[\psi] \mid R^r_{\bowtie x}[F\phi]$$
$$\psi ::= X\phi \mid \phi \,\mathcal{U}^{\leq k}\, \phi \mid \phi \,\mathcal{U}\, \phi$$

where $a \in AP$, r is a reward structure, $\bowtie \in \{<, \leq, >, \geq\}$, $q \in \mathbb{R} \cap [0,1]$, $x \in \mathbb{R}_{\geq 0}$, and $k \in \mathbb{N}$.

For example, $P_{\geq 0.5}[F\phi]$ means the probability of eventually reaching states satisfying ϕ is at least 0.5; $R^1_{\leq 10}[F\phi]$ means the expected cumulated reward (or cost) of reward structure 1 until reaching states satisfying ϕ is at most 10. We also often use notation such as $P_{=?}[\cdot]$ and $R^r_{=?}[\cdot]$ to represent numerically-valued probability or reward properties. We omit a full definition of the semantics of PCTL with rewards. Further details can be found in [9].

3 Game-theoretic Verification with Parametric Probabilistic Model Checking

We now present a framework for game-theoretic verification of quantitative properties, based on parametric model checking techniques for discrete-time Markov chains. In this section, we introduce a model called *parametric Markov chain games* (PMCGs), in which parameters represent (probabilistic) decisions taken by players, and then we discuss the existence of Nash equilibria for this model. Subsequently, we will show to automatically synthesise these Nash equilibria, based on formal specifications of a system model and utility functions, and by building upon existing parametric model checking techniques.

3.1 Parametric Markov chain games

We model systems as n-player games. We will assume a fixed set of players $N = \{1, \ldots, n\}$, each of which has m possible actions $A = \{a_1, \ldots, a_m\}$ (i.e., all players share the same action space A). Each $a_j \in A$ corresponds to a different possible course of action which can typically be decided upon multiple times during the execution of the system. We are generally interested in *mixed* strategies which are defined as a probability distribution π_i over A. Every time that player i needs to decide which action from A to take, it will do so by selecting action $a_j \in A$ with probability $\pi_i(a_j)$.

Given such strategies π_i for each player i, the subsequent behaviour of the system is necessarily probabilistic. Furthermore, we usually want to model other stochastic aspects of the system, for example, message transmission failure. So, we will assume that the behaviour of the system, under strategies π_i, can be modelled as a discrete-time Markov chain. In general, the transition probabilities of this Markov chain will be defined as expressions in terms of the probabilities $\pi_i(a_j)$ for each player i selecting each action a_j. This can be modelled as a *parametric* Markov chain whose parameters correspond to the probabilities $\pi_i(a_j)$. We refer to the resulting model as a *parametric Markov chain game* (PMCG), which is formally defined as follows.

Definition 12 (Parametric Markov chain game). *A* parametric Markov chain game *(PMCG) is a tuple of the form:*
$\mathcal{G}=((S, I, \Delta, V, AP, L), N, A, \{V_i\}_{i\in N}, \{u_i\}_{i\in N})$*, where:*

- (S, I, Δ, V, AP, L) *is a parametric Markov chain, which we will denote* $\mathcal{M}_\mathcal{G}$*;*
- N *is a finite set of players;*
- $A = \{a_1, \ldots, a_m\}$ *is a finite set of* m *actions;*
- (V_i) *is a partition of the parameter set* V *of* $\mathcal{M}_\mathcal{G}$*, assigning a subset* $V_i = \{x_{i,1}, \ldots, x_{i,m}\} \subseteq V$ *to each player* $i \in N$*;*
- $u_i : \Omega_{\mathcal{M}_\mathcal{G}} \to \mathbb{R}$ *is a utility function for each player* i*.*

A PMCG $\mathcal{G}$ incorporates a parametric Markov chain $\mathcal{M}_\mathcal{G}$, whose parameter set V is partitioned into subsets $V_i = \{x_{i,1}, \ldots, x_{i,m}\}$ for each player i. Each individual parameter $x_{i,j}$ represents the probability with which player i will choose action $a_j \in A$. The PMCG also defines a utility function u_i for each player i, represented as a function from (infinite) paths in $\mathcal{M}_\mathcal{G}$ to a real value.

3.2 Mixed strategies and Nash equilibria for PMCGs

Given a PMCG $\mathcal{G}$, the set of *pure strategies* for player i is the set of available actions A. A *mixed strategy* for player i, denoted by π_i, is given as a probability distribution over the set of pure strategies and written as a vector $\pi_i = (\pi_{i,1}, \pi_{i,2}, \ldots, \pi_{i,m})$ where $\pi_{i,j}$ denotes the probability of player i choosing action a_j. A (mixed) *strategy profile* $\pi = (\pi_1, \ldots, \pi_n) = (\pi_{i,1}, \pi_{i,2}, \ldots, \pi_{n,m})$ comprises a strategy for all players in the game.

The parametric Markov game $\mathcal{M}_\mathcal{G}$ of $\mathcal{G}$ represents the behaviour of the system under any possible strategy profile, with parameter $x_{i,j}$ representing the probability of player i choosing a_j. Thus, for a fixed strategy profile π, the resulting behaviour of the system is modelled by the induced Markov chain $\mathcal{M}_{\mathcal{G},V/\pi}$, in which each $v_{i,j}$ is assigned value $\pi_{i,j}$. This Markov chain gives us a probability measure, denoted $\texttt{Prob}^\pi_\mathcal{G}$ over the set of all paths in $\Omega_\mathcal{G}$ through $\mathcal{G}$.

Now, to reason about Nash equilibria of $\mathcal{G}$, we first need to specify a preference ordering over strategies. As discussed earlier, we do so implicitly by defining a utility function u_i whose expected value each player i aims to maximise. In a PMCG, a utility function u_i assigns a real value to each *path* through the model. The expected value of u_i under a mixed strategy profile π is then defined

by the Markov chain induced from $\mathcal{G}$ by π. More precisely, the expected utility for player i is $\mathbb{E}^{\pi}_{\mathcal{G}}(u_i)$, i.e., the expected value of function u_i with respect to the probability measure $\texttt{Prob}^{\pi}_{\mathcal{G}}$ over paths through $\mathcal{G}$. Abusing notation, we will often simply write $u_i(\pi)$ instead of $\mathbb{E}^{\pi}_{\mathcal{G}}(u_i)$. This allows us to give the following formal definition of a mixed strategy Nash equilibrium for a PMCG.

Definition 13 (Mixed strategy Nash equilibrium of $\mathcal{G}$). *Given a PMCG $\mathcal{G}=(\mathcal{M}_{\mathcal{G}}, N, A, (V_i), (u_i))$, a mixed strategy profile π for $\mathcal{G}$ is a* Nash equilibrium *if, for any player i and any mixed strategy π_i' of player i, we have $u_i(\pi_{-i}, \pi_i) \geq u_i(\pi_{-i}, \pi_i')$, where u_i gives the expected utility for player i under a strategy profile, as explained above.*

Theorem 1. *Let $\mathcal{G}=(\mathcal{M}_{\mathcal{G}}, N, A, (V_i), (u_i))$ be a PMCG. If, for all $i \in N$, the utility function u_i is continuous and quasi-concave over the set of mixed strategies for player i, then $\mathcal{G}$ has a mixed strategy Nash equilibrium.*

Proof. Consider the set of mixed strategies for player i, which is a set of distributions over the set A of m actions. This is a non-empty, convex and compact subset of $\mathbb{R}^m$. By Proposition 1, as long as each utility function u_i (and thus the preference relation $\succeq_i$) is continuous and quasi-concave over the set of mixed strategies, then $\mathcal{G}$ satisfies all the requirements to have a Nash equilibrium. □

The following result gives an important property of mixed strategy Nash equilibria for PMCGs $\mathcal{G}$ when calculating such equilibria:

Lemma 1. *Given a PMCG $\mathcal{G}=(\mathcal{M}_{\mathcal{G}}, N, A, \{V_i\}_{i\in N}, \{u_i\}_{i\in N})$, if u_i is monotonic on player i's mixed strategies, then: an n-tuple of mixed strategies profile $\pi = (\pi_1, \ldots, \pi_n)$ is a mixed strategy Nash equilibrium of $\mathcal{G}$* iff *for every player $i \in N$, every pure strategy in the support of π is a best response to π_{-i}.*

Proof. ("⇒"): Assume that there is an action a in the support of π_i which is not a best response to π_{-i}. Then by the monotonicity of the utility function u_i, player i can increase his utility by switching probability from a to an action that is a best response, so π_i is not a best response to π_{-i}, which leads to a contradiction.

("⇐"): Suppose that there is a mixed strategy π_i' that gives a higher expected utility than π_i does in response to π_{-i}. Then at least one action in the support of π_i' must give a higher utility than some action in the support of π_i, so that not all actions in the support of π_i are best responses to π_{-i}, which leads to a contradiction. □

One can imagine that, if a mixed strategy π_i is a best response, then each of the pure strategies involved in the mix must itself be a best response. Hence, all the pure strategies in the mix must yield the same expected utility. That is to say, every choice in the support of any player's equilibrium mixed strategy must yield the same utility value: $u_i(\pi_{-i}, a) = u_i(\pi_{-i}, a')$ for any two pure strategies $a, a' \in A$ such that the probabilities of player i choosing pure strategies a and a' are positive: $\pi_{i,a} > 0$ and $\pi_{i,a'} > 0$. We will use this fact to find mixed strategy

Nash equilibria of $\mathcal{G}=(\mathcal{M}_\mathcal{G}, N, A, \{V_i\}_{i\in N}, \{u_i\}_{i\in N})$ since we can therefore write the Nash equilibria conditions as follows:

$$\begin{cases} u_i(\pi_{-i}, a) = u_i(\pi_{-i}, a') \ \ \forall a, a' \in A, \ s.t. \ \pi_{i,a}, \pi_{i,a'} > 0 \\ \sum_{j=1}^{m} \pi_{i,j} = 1 \ \ \forall i \\ 0 \leq \pi_{i,j} \leq 1 \ \ \forall i, j. \end{cases} \tag{1}$$

By solving the equations above, we can find the equilibria.

4 Finding All Nash Equilibria

In this section we consider practical approaches to the synthesis of Nash equilibria for systems modelling using the parametric Markov chain game formalism introduced above. First, we need a formal specification of both the model of the system and the utility functions that are being used to define equilibria. Our work builds upon functionality in the PRISM model checker [10] for modelling and constructing parametric Markov chains, so systems are specified using the PRISM modelling language and utility functions for the models specified using PRISM's temporal logic notation, summarised in Section 2.3.

Using PRISM, we apply probabilistic verification to a parametric Markov chain, yielding rational functions that represent the expected values of utility functions. These rational functions are over the variables $x_{i,j}$ corresponding to the probabilities in the mixed strategies of each player. At this point we apply a simple optimisation to reduce the number of variables required. Given that we know $x_{i,1} + \cdots + x_{i,m} = 1$ for any i, one of the variables is redundant and we can rewrite, for example, $x_{i,m}$ as $1 - (x_{i,1} + \cdots + x_{i,m-1})$.

Next, we check the monotonicity of each player's utility function on the variables for its own mixed strategy.

Finally, we construct and solve, from the computed utility functions, a set of equalities in order to determine the set of Nash equilibria for the model. This is done based on the Nash equilibria conditions identified previously in (1), and is described in more detail below.

We assume the cases that each player's utility function is monotonic (and thus quasi-concave) for his own strategy from then on. The details of each step are discussed in the following subsections.

4.1 Nash equilibria conditions as polynomial equations

Given a set of players' utility properties from a PMCG, each of the form

$$u_i(x_{1,1}, \ldots, x_{1,m}, \ldots, x_{i,1}, \ldots, x_{i,m}, \ldots x_{n,1}, \ldots, x_{n,m}).$$

the Nash equilibria conditions can be used to construct a polynomial system of equations. Since player i's optimal choices of $x_{i,j}$ should equal the utility of the

other players from playing pure strategies, by Lemma 1, we can build a set of equations for each i:

$$\begin{aligned} & u_i(x_{1,1}, \ldots, x_{i-1,1}, \ldots, x_{i-1,m}, 1, 0, \ldots, 0, x_{i+1,1}, \ldots, x_{n,m}) \\ = {} & u_i(x_{11}, \ldots, x_{i-1,1}, \ldots, x_{i-1,m}, 0, 1, \ldots, 0, x_{i+1,1}, \ldots, x_{n,m}) \\ = {} & \ldots \quad \ldots \\ = {} & u_i(x_{11}, \ldots, x_{i-1,1}, \ldots, x_{i-1,m}, 0, 0, \ldots, 1, x_{i+1,1}, \ldots, x_{n,m}). \end{aligned}$$

In addition, for all $i \in N$, $\{x_{i,j} | 1 \leq j \leq m\}$ is a distribution so: $0 \leq x_{i,j} \leq 1$ and $\sum_{j=1}^{m} x_{i,j} = 1$. By solving the set of equations obtained above, we can find a set of complex solutions to $\{x_{ij} | i = 1, \ldots, n,\ j = 1, \ldots, m\}$, which, if they exist, yield the Nash equilibria, as required.

Solving the above system is non-trivial since the problem is typically non-linear. Herings and Peeters [8] show the feasibility of computing all Nash equilibria of general finite games in theory, and Datta provides an implementation in [5]. Homotopy continuation methods have been proven to be a reliable and powerful mathematical method to compute all isolated complex solutions [11,13] of polynomial systems. The method includes a number of main steps: use the algebraic structure to count the roots and to construct a start system, the root count determines the number of solution paths to be traced; the target system is embedded to solve in *homotopy* system, i.e., a family of systems connecting the start and the target system; following the solution paths of the homotopy system, the *continuation* methods are applied to extend the solutions of the start system to the desired solutions of the target system.

There are a number of software packages devoted to solving polynomial systems by using *homotopy continuation methods*. In this paper, we exploit the PHCpack [21] platform since it performs better than other software packages in terms of computational stability and capacity [4].

We conclude the Nash equilibrium generation with a few final checks. First, all of the mixing probabilities we have constructed must indeed be probabilities: $\forall i \in N. \sum_{j=1}^{m} x_{i,j} = 1$. Second, if there are no probability solutions, we need to check whether the player has a strictly profitable deviation. In the following sections, we illustrate the process on some examples.

5 Game-theoretic Modelling of Security Systems

Now, we move on to describe how game-theoretic verification approach described in the previous two sections can be applied specifically in the context of security-sensitive communicating systems. In particular, we show to model such systems as parametric Markov chain games (see Definition 12) equipped with appropriate utility functions. In the next section, we will demonstrate the approach on a case study: the Crowds anonymity protocol.

We consider computerised communication systems consisting of a set of *players* and a set of destinations. We focus on the core procedure of transmitting

messages through the system. This typically involves randomisation, and potentially other stochastic aspects such as message loss. Individual players may make certain strategic decisions about how to participate in the system. These are modelled as probabilistic actions, whose probabilities are parameters controlled by the player in question. The *adversary* is the set of malicious players who partially observe or participate in the transmission of messages and try to learn the sensitive information stored in the system states.

Players' *utility functions* are used to indicate preferences between strategies. Using our PMCG model, utility functions are defined as real-valued functions over paths, i.e., a mapping from each possible system execution to a real value. Strategies are then compared based on the expected value of this function. In the context of communication systems, we consider utility functions comprising two parts: *security* measurement and *cost* measurement. It is reasonable to expect that conflicts or trade-offs might exist between these two: for example, additional relaying of messages might improve security but at additional cost.

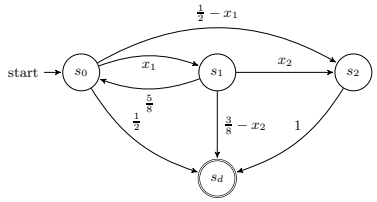
Security measurement. In a communicating network, the adversaries act as a player and make a series of partial observations over the communication network. There is a set of execution paths, and some of them release information to the adversaries. We say an execution path is *bad* to i if there is a transition from a sensitive node controlled by i to a node controlled by a malicious player along the path, which cause information leakage. In our quantitative setting, we consider (the information of) player i to be *secure* if the probability of *bad* paths for i is small enough. Specifically, letting $\psi \subseteq \Omega_{\mathcal{G}}$ denote the set of paths reaching the destination, and $\psi_i^* \subseteq \Omega_{\mathcal{G}}$ denote the set of *bad* paths of the player i, we define the measurement of security for mixed strategy π to be:

$$u_i^s(\pi) = \texttt{Prob}^{\pi}(\psi \setminus \psi_i^*) = \texttt{Prob}^{\pi}(\psi) - \texttt{Prob}^{\pi}(\psi_i^*), \tag{2}$$

where ψ and ψ^* can be specified as PCTL formulas. Note that the bigger the security metric the more secure the system is with regard to the security property of interest.

Example 1. Consider players $N = \{0, 1, 2\}$ and assume $0, 1$ are honest players and 2 is a malicious player. Consider a PMCG in which states s_i ($i \in \{0, 1, 2\}$) are controlled by player i (say i is sending or forwarding a message for instance), s_d denotes the message reaching its destination, and parameters x_i represent probabilities controllable by player i. Assume that player 0 starts a message at s_0 (sensitive), and any transitions from s_0 to s_2 will violate the security policy. The *bad* paths ψ^* contain all the paths including the transition from s_0 to s_2. The probabilistic transition graph and security metric computation by (2) is presented in Fig. 1.

Cost measurement. Markov chains with reward (or cost) structures allow us to specify two distinct types of rewards: *state* rewards, which are assigned to states by means of a function of the form $S \to \mathbb{R}_{\geq 0}$, and *transition* rewards, which



$$u_0^s(\pi) = 1 - (\frac{1}{2} - x_1)\left(1 + \frac{5}{8}x_1 + (\frac{5}{8}x_1)^2 + \dots\right) = \frac{4 + 3x_1}{8 - 5x_1}.$$

Fig. 1. Example: security measurement

are assigned to transitions by means of the function of the form $S \times S \to \mathbb{R}_{\geq 0}$. The state reward is the reward acquired in each state per time-step, and transition rewards are acquired each time a transition between states occurs. For the cost measurement, we consider the "expected cost consumed until a message reaches its destination". Let $r_i = (r_{i,\texttt{state}}, r_{i,\texttt{action}})$ be a reward structure for player i, and ϕ represent a set of target states (e.g., where a message has reached the destination). The required expected cost can be specified using the reward operator from PRISM's temporal logic (see Section 2.3): $R^{r_i}_{=?}[F\phi]$.

Example 2. Consider again the example presented in Fig.1. Let $r_{0,\texttt{state}}(s_0) = 0$, assume the cost of player 0 forwarding a message is 1. Letting $u^c_{i,s}$ represent the expected cost for player i from state s, we can calculate the cost metric by solving:

$$\begin{cases} u^c_{0,s_2} = u^c_{0,s_d} = 0 \\ u^c_{0,s_0} = (1 + x_1 u^c_{0,s_1}) + (\frac{1}{2} - x_1)u^c_{0,s_2} + \frac{1}{2}u^c_{0,s_d} \\ u^c_{0,s_1} = (0 + \frac{5}{8}u^c_{0,s_0}) + x_2 u^c_{0,s_2} + (\frac{3}{8} - x_2)u^c_{0,s_d} \end{cases}$$

We have $u^c_{0,s_0} = \frac{8}{8-5x_1}$, i.e., the parametric expected cost of player 0 is $\frac{8}{8-5x_1}$.

Expected utilities. Note that the performance of the system is in direct proportion to the security metric, and is in inverse proportion to the cost metric. So we define the *utility* of player i as a ratio of the security metric function over the cost metric function:

$$u_i(\pi) = \frac{u^s_i(\pi) * w_s}{u^c_{i,s_0}(\pi) * w_c} \tag{3}$$

where w_s denotes the weight of the security property, and w_c denotes the weight of the cost property.

Example 3. Let $w_s = 3$, $w_c = 1$, we calculate the expected utility of the previous example as: $u_0(\pi) = \frac{(4+3x_1)/(8-5x_1)*3}{24/(8-5x_1)} = \frac{3(4+3x_1)}{8}$.

6 Experimental Results: the Crowds protocol

We have implemented our parametric model checking based approach to game-theoretic verification as an extension of the PRISM model checker [1]. Our tool

can find the mixed strategy Nash equilibria for the players in a game when each player's utility function is monotonic w.r.t. his own mixed strategy. Building on the ideas set out in the previous section, we have used this approach to analyse the anonymity and cost of the Crowds protocol [16].

Crowds is a protocol allowing users to forward messages anonymously. The idea is that each user randomly chooses a user to forward a message rather than send their message to the destination directly. A forwarding route is therefore established within a collection of network members. In our game-theoretic model, each honest forwarder makes a decision whether to be cooperative or to be selfish. If he decides to be cooperative, then he flips a coin to decide to either send the message to the destination directly (with probability $1 - \texttt{PF}$), or relay it to another crowd member randomly (with probability $\texttt{PF}$); otherwise he discards the message directly. The malicious player behaves like a normal player but he will send the message he received to the destination directly. A malicious user can never be certain whether the observed user is the actual sender, or is simply forwarding another user's message.

Suppose in the protocol, there are two honest members (player-1, 2) and two malicious members (player-3, 4). Assume that sending and relaying a message costs $c_{s_1} = 1$ and $c_{r_1} = 2$ respectively for player-1, and $c_{s_2} = 2$ and $c_{r_2} = 3$ for player-2. Without loss of generality, we choose different cost values for different operations and players here for demonstration. Let x_i denote the probability of player i being cooperative, and $1 - x_i$ the probability of being selfish. Fig. 3 in Appendix B presents the transition graph of the model. We define the property specification for initiator (honest) player k as:

$$\frac{(\texttt{P}_{=?}[F\ \textit{destination}] - \texttt{P}_{=?}[F(\texttt{to}{=}3 \vee \texttt{to}{=}4) \wedge (\texttt{from}{=}k) \wedge (\texttt{sender}{=}k)]) * w_s}{(\texttt{R}_{k=?}[F\ \textit{deadlock}]) * w_c},$$

where $k \in \{1, 2\}$ is an honest player, $\texttt{to} \in \{1, 2, 3, 4\}$ is the player whom the message is sent to, $\texttt{from} \in \{1, 2, 3, 4\}$ is the player who is sending/forwarding the message, $\texttt{P}_{=?}[F\ \textit{destination}]$ calculates the probability of the set of runs ϕ reaching the destination; $\texttt{P}_{=?}[F\ (\texttt{to} = 3 \vee \texttt{to} = 4) \wedge (\texttt{from} = k) \wedge (\texttt{sender} = k)]$ calculates the probability of the set of runs ϕ^* violating anonymity property, i.e., sender k is sending or forwarding the message to a malicious player (3 or 4) directly; $\texttt{R}_{k=?}[F\ \textit{deadlock}]$ calculates the accumulated costs u_k^c reaching a terminating state; and w_s and w_c denote the weights for anonymity and costs, respectively. Assume $w_s = 3$, $w_c = 1$, let $u_1(x_1, x_2)$ and $u_2(x_1, x_2)$ denote the polynomial utility functions generated for player-1 and player-2 respectively, we get the parametric model checking results given as a set of polynomials as follows:

$$u_1(x_1, x_2) = \frac{15 * x_2 * x_1 - 36 * x_1 - 30 * x_2 - 144}{6 * x_2 * x_1 - 28 * x_1 + 12 * x_2 - 120}$$
$$u_2(x_1, x_2) = \frac{15 * x_2 * x_1 - 30 * x_1 - 36 * x_2 - 144}{8 * x_2 * x_1 + 24 * x_1 - 36 * x_2 - 216}$$

First let us check the monotonicity of $u_1(x_1, x_2)$ for player 1's mixed strategy:

$$\frac{\partial u_1}{\partial x_1} = \frac{6(15x_2^2 - 92x_2 + 12)}{(3x_1x_2 - 14x_1 + 6x_2 - 60)^2}$$

Let $\frac{\partial u_1}{\partial x_1} = 0$, we have $x_2 = \frac{2}{15}, 6$. Considering $x_2 \in [0,1]$, we have: when $0 \leq x_2 \leq \frac{2}{15}$, $\forall x_1 . \frac{\partial u_1}{\partial x_1} \geq 0$ i.e., u_1 is nondecreasing with x_1; similarly, when $\frac{2}{15} \leq x_1 \leq 1$, $\forall x_1 . \frac{\partial u_1}{\partial x_1} \leq 0$ i.e., u_1 is nonincreasing with x_1. Therefore, for $x_1, x_2 \in [0,1]$, u_1 is always monotonic w.r.t. player 1's mixed strategy; similarly, u_2 is monotonic w.r.t. player 2's mixed strategy.

To find the equilibria, we want to choose x_1 (x_2) so as to equalise the utility of player-2 (player-1) receives from playing either pure strategies. We therefore write the equations as: $u_1(0, x_2) = u_1(1, x_2),\ u_2(x_1, 0) = u_2(x_1, 1)$, i.e.,

$$(-30 * x_2 - 144) * (18 * x_2 - 148) = (-15 * x_2 - 180) * (12 * x_2 - 120)$$
$$(-30 * x_1 - 144) * (32 * x_1 - 252) = (-15 * x_1 - 180) * (24 * x_1 - 216).$$

By solving the above equations, we get the Nash equilibria and the relevant utility values: $x_1 = 0.72;\ x_2 = 0.13;\ u_1 = 1.25;\ u_2 = 0.83$.

In order to check the solutions obtained, let us look at the experiment results of the utilities of layer $i = 1, 2$ with constant variables $x_1 = 0 : 1$ and $x_2 = 0 : 1$ produced by PRISM. Fig. 2 presents the utilities to player 1's (cf. player 2's) pure strategies as functions of player 2's (cf. player 1's) mixed strategy. One can see

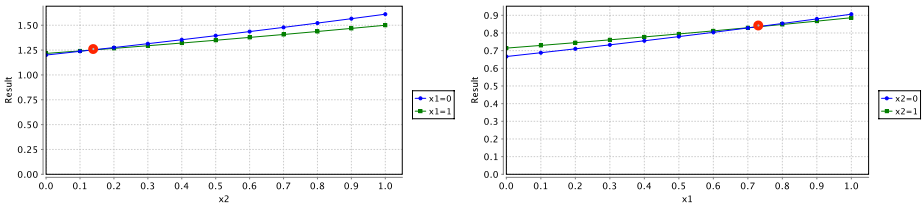


Fig. 2. Utilities to player 1's (left) and 2's (right) pure strategies as functions of player 2's and 1's mixed strategy

that the intersection of the two lines are mixed strategy equilibria: i.e., player-1 chooses $x_1 = 0.72$, player-2 chooses $x_2 = 0.13$, which also meets our equilibria results produced by our PMCG analyser. In addition, if we set up the reward structure symmetrically for player 1 and 2, say both transitions s_1 and s_2 cost 2, both transitions r_1 and r_2 cost 3, we obtain the symmetric Nash equilibria for player 1 and 2: $x_1 = x_2 = 0.72;\ u_1 = u_2 = 0.83$. Table 1 lists a group of experimental results produced by our PMCG analyser regarding to different number of players, honest players, reward structures (in which c_{s_i} and c_{r_i} denote the cost of player i for sending and relaying a message respectively) with the size of the state space of the parametric Markov chain and the total computation time spent to achieve the final equilibria results. This demonstrates our proposed approach can be used to automatically find the mixed Nash equilibria for the Crowds protocol with multiple players. The computation time increases with the number of honest players, and the result of the Nash equilibria is mainly affected by the specified reward structures.

Table 1. Experimental results with size of state space and time cost

N	N_h	Reward structure $(c_{s_1}, c_{r_1}; \ldots; c_{s_{N_h}}, c_{r_{N_h}})$	Size of PMC	Total time (sec.)	Nash equilibria $(x_1, x_2, \ldots)$
3	2	$(1, 2; 2, 3)$	19	1.18	$(0.62, 0.46)$
4	2	$(1, 2; 2, 3)$	23	1.181	$(0.72, 0.13)$
4	3	$(1, 1; 1, 1; 1, 1)$	48	1.857	$(0.32, 0.32, 0.32)$
5	2	$(2, 3; 2, 3)$	35	1.878	$(0.29, 0.29)$
5	3	$(1, 0.8; 1, 0.8; 1, 0.9)$	57	2.003	$(0.038, 0.038, 0.407)$
7	2	$(1, 1.2; 1, 1.3)$	35	1.131	$(0.186, 0.608)$
7	3	$(1, 0.6; 1, 0.6; 1, 0.6)$	75	1.964	$(0.469, 0.469, 0.469)$
7	4	$(1, 0.5; 1, 0.5; 1, 0.5; 1, 0.5)$	131	116.145	$(0.15, 0.15, 0.15, 0.15)$

We present more details of how the reward structures affect the Nash equilibria results in Table 2 in Appendix A. It can be seen that the players tend to be more cooperative (larger equilibria) when the ratio of the cost of sending a message s_i and relaying a message r_i is bigger (see Table 2 (a)). In the cases investigated, a range of such ratios results in *mixed* Nash equilibria, while the ratios outside of that range lead to *pure* Nash equilibria. The strategy of each player is also affected by other players' reward structures (see Table 2 (b&c)).

7 Conclusions

We have presented a new automated game-theoretic approach for quantitative verification of security properties of software systems. Security-sensitive communication networks typically require the collaboration of their participants to work effectively. We study the problem of how the participants *should* react regarding to collaborating strategies in order to improve the overall performance with a balance between security and cost. We apply a game-theoretic approach to capture such a balance represented as Nash equilibria. We propose methods to automatically find the equilibria under which no participants can benefit by changing their strategies.

To achieve our goal, we propose the model of parametric Markov chain games and apply parametric model checking techniques to compute utility functions, using models described in the PRISM modelling language and utilities specified in probabilistic temporal logic. We generate and solve a polynomial equation system, from which we identify the Nash equilibria. To illustrate the applicability of our approach, we have implemented our approach as an extension to the tool of PRISM model checker, and analysed the Crowds protocol, studying the trade-offs between anonymity/cost.

Both theoretical and experimental evidence are presented for the utility of the approach for quantitative security analysis. We believe this is a significant contribution to automatically analysing security systems from a *quantitative* and *game-theoretic* view. For future work, we plan to study the precise computational

complexity issues of the presented approach, and adapt our method in wider cases in addition to security systems.

References

1. Prototype tool and case studies. `http://www.prismmodelchecker.org/files/qest19/`
2. Carter, M.: Foundations of mathematical economics. MIT Press (2001)
3. Chatterjee, K., Majumdar, R., Jurdzinski, M.: On nash equilibria in stochastic games. In: CSL. pp. 26–40 (2004)
4. Datta, R.S.: Using computer algebra to find nash equilibria. In: ISSAC. pp. 74–79. ACM, New York, USA (2003)
5. Datta, R.S.: Finding all nash equilibria of a finite game using polynomial algebra. Econ. Theory 42(1), 55–96 (2009)
6. Hahn, E.M., Hermanns, H., Zhang, L.: Probabilistic reachability for parametric Markov models. STTT 13(1), 3–19 (2011)
7. Hansson, H., Jonsson, B.: A logic for reasoning about time and reliability. Formal Aspects of Computing 6(5), 512–535 (1994)
8. Herings, P.J., Peeters, R.J.A.P.: A globally convergent algorithm to compute all nash equilibria for n -person games. Annals OR 137(1), 349–368 (2005)
9. Kwiatkowska, M., Norman, G., Parker, D.: Stochastic model checking. In: SFM. vol. 4486, pp. 220–270 (2007)
10. Kwiatkowska, M., Norman, G., Parker, D.: PRISM 4.0: Verification of probabilistic real-time systems. In: Gopalakrishnan, G., Qadeer, S. (eds.) CAV. LNCS, vol. 6806, pp. 585–591. Springer (2011)
11. Li, T.: Solving polynomial systems. The Mathematical Intelligencer 8(3), 33–39 (1987)
12. Mhamdi, T., Hasan, O., Tahar, S.: Evaluation of anonymity and confidentiality protocols using theorem proving. Formal Methods in System Design 47(3), 265–286 (2015)
13. Morgan, A.: Solving Polynominal Systems Using Continuation for Engineering and Scientific Problems. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA (2009)
14. Nash, J.: Equilibrium points in n-person games. Proceedings of the National Academy of Sciences 36(1), 48–49 (1950)
15. Osborne, M.J.: An introduction to game theory. Oxford University Press (2009)
16. Reiter, M.K., Rubin, A.D.: Crowds: Anonymity for web transactions. ACM Transactions on Information and System Security 1, 66–92 (1998)
17. Shmatikov, V.: Probabilistic analysis of anonymity. In: CSFW. pp. 119–128. IEEE Computer Society Press (2002)
18. Shmatikov, V.: Probabilistic model checking of an anonymity system. Journal of Computer Security 12(3/4), 355–377 (2004)
19. Ummels, M., Wojtczak, D.: The complexity of nash equilibria in stochastic multi-player games. Logical Methods in Computer Science 7(3) (2011)
20. Venkitasubramaniam, P., Tong, L.: A game-theoretic approach to anonymous networking. IEEE/ACM Transactions on Networking 20(3), 892–905 (2012)
21. Verschelde, J.: Algorithm 795: PHCpack: A general-purpose solver for polynomial systems by homotopy continuation. ACM Trans. Math. Softw. 25(2), 251–276 (1999)
22. Yang, M., Sassone, V., Hamadou, S.: A game-theoretic analysis of cooperation in anonymity networks. In: POST. pp. 269–289 (2012)

Appendix A: Sensitivity study of the reward structures

Table 2. Sensitivity study of the reward structures to Nash equilibria (N.E.s) & utilities ($N = 4, N_h = 3$)

(a)

Reward structure$(c_{s_1}, c_{r_1}; c_{s_2}, c_{r_2}; c_{s_3}, c_{r_3})$	N.E. (x_1, x_2, x_3)	Utility (u_1, u_2, u_3)
(1.0, 0.2; 1.0, 0.2; 1.0, 0.2)	(1.0, 1.0, 1.0)	(6.06, 6.06, 6.06)
(1.0, 0.4; 1.0, 0.4; 1.0, 0.4)	(1.0, 1.0, 1.0)	(4.92, 4.92, 4.92)
(1.0, 0.6; 1.0, 0.6; 1.0, 0.6)	(0.908, 0.908, 0.908)	(3.93, 3.93, 3.93)
(1.0, 0.8; 1.0, 0.8; 1.0, 0.8)	(0.55, 0.55, 0.55)	(2.753, 2.753, 2.753)
(1.0, 1; 1.0, 1; 1.0, 1.0)	(0.32, 0.32, 0.32)	(2.113, 2.113, 2.113)
(1.0, 1.2; 1.0, 1.2; 1.0, 1.2)	(0.16, 0.16, 0.16)	(1.713, 1.713, 1.713)
(1.0, 1.4; 1.0, 1.4; 1.0, 1.4)	(0.048, 0.048, 0.048)	(1.44, 1.44, 1.44)
(1.0, 1.6; 1.0, 1.6; 1.0, 1.6)	(0.0, 0.0, 0.0)	(1.304, 1.304, 1.304)
(1.0, 1.8; 1.0, 1.8; 1.0, 1.8)	(0.0, 0.0, 0.0)	(1.25, 1.25, 1.25)
(1.0, 2.0; 1.0, 2.0; 1.0, 2.0)	(0.0, 0.0, 0.0)	(1.2, 1.2, 1.2)

(b)

Reward structure$(c_{s_1}, c_{r_1}; c_{s_2}, c_{r_2}; c_{s_3}, c_{r_3})$	N.E. (x_1, x_2, x_3)	Utility (u_1, u_2, u_3)
(1.0, 0.8; 1.0, 0.8; 1.0, 0.5)	(1.0, 1.0, 1.0)	(3.58, 3.58, 4.5)
(1.0, 0.8; 1.0, 0.8; 1.0, 0.6)	(0.908, 0.908, 0.191)	(2.78, 2.78, 3.93)
(1.0, 0.8; 1.0, 0.8; 1.0, 0.7)	(0.708, 0.708, 0.393)	(2.76, 2.76, 3.24)
(1.0, 0.8; 1.0, 0.8; 1.0, 0.8)	(0.55, 0.55, 0.55)	(2.75, 2.75, 2.75)
(1.0, 0.8; 1.0, 0.8; 1.0, 0.9)	(0.42, 0.42, 0.68)	(2.756, 2.756, 2.39)
(1.0, 0.8; 1.0, 0.8; 1.0, 1.0)	(0.321, 0.321, 0.779)	(2.763, 2.763, 2.113)
(1.0, 0.8; 1.0, 0.8; 1.0, 1.1)	(0.235, 0.235, 0.864)	(2.77, 2.77, 1.89)
(1.0, 0.8; 1.0, 0.8; 1.0, 1.2)	(0.16, 0.16, 0.94)	(2.78, 2.78, 1.713)
(1.0, 0.8; 1.0, 0.8; 1.0, 1.3)	(0.1, 0.1, 0.9997)	(2.79, 2.79, 1.56)
(1.0, 0.8; 1.0, 0.8; 1.0, 1.4)	(1.0, 1.0, 1.0)	(3.58, 3.58, 2.54)

(c)

Reward structure$(c_{s_1}, c_{r_1}; c_{s_2}, c_{r_2}; c_{s_3}, c_{r_3})$	N.E. (x_1, x_2, x_3)	Utility (u_1, u_2, u_3)
(1.0, 1.0; 1.0, 1.0; 1.0, 0.5)	(1.0, 1.0, 1.0)	(3.15, 3.15, 4.5)
(1.0, 1.0; 1.0, 1.0; 1.0, 0.6)	(1.0, 1.0, 1.0)	(3.15, 3.15, 4.14)
(1.0, 1.0; 1.0, 1.0; 1.0, 0.7)	(1.0, 1.0, 1.0)	(3.15, 3.15, 3.84)
(1.0, 1.0; 1.0, 1.0; 1.0, 0.8)	(0.55, 0.55, 0.09)	(2.12, 2.12, 2.75)
(1.0, 1.0; 1.0, 1.0; 1.0, 0.9)	(0.42, 0.42, 0.22)	(2.11, 2.11, 2.39)
(1.0, 1.0; 1.0, 1.0; 1.0, 1.0)	(0.32, 0.32, 0.32)	(2.113, 2.113, 2.113)
(1.0, 1.0; 1.0, 1.0; 1.0, 1.1)	(0.235, 0.235, 0.407)	(2.114, 2.114, 1.892)
(1.0, 1.0; 1.0, 1.0; 1.0, 1.2)	(0.16, 0.16, 0.48)	(2.116, 2.116, 1.713)
(1.0, 1.0; 1.0, 1.0; 1.0, 1.3)	(0.1, 0.1, 0.54)	(2.12, 2.12, 1.56)
(1.0, 1.0; 1.0, 1.0; 1.0, 1.4)	(1.0, 1.0, 1.0)	(2.123, 2.123, 1.44)

Appendix B: The PRISM model of Crowds protocol

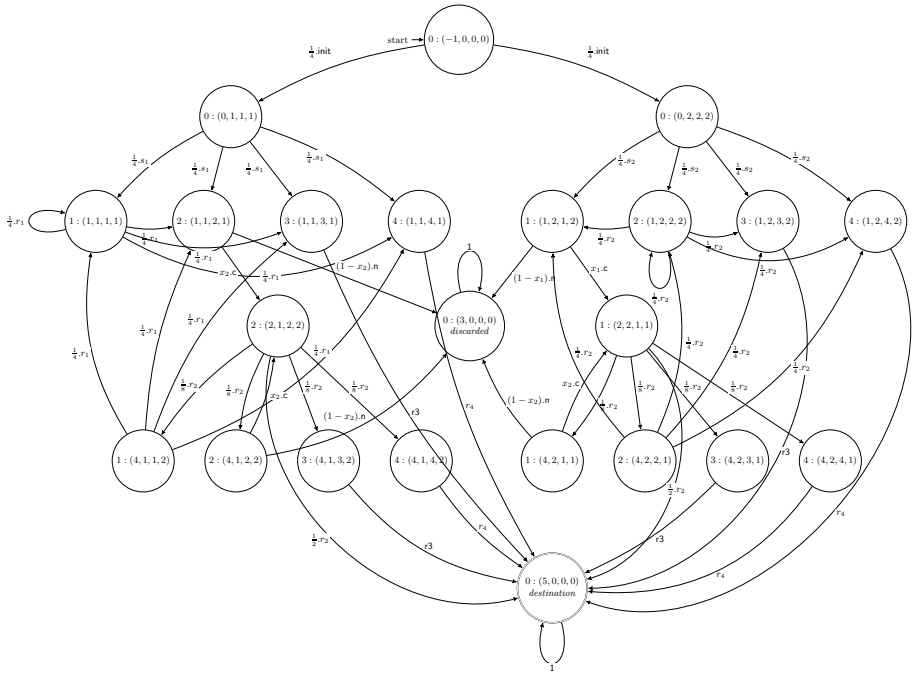


Fig. 3. Model of Crowds with 2 honest players and 2 malicious player with $PF = 0.5$. For $i \in \{1, 2, 3, 4\}$, transition label r_i denotes relaying a message by player i; for $i \in \{1, 2\}$, s_i denotes sending a message by (honest) player i, c_i denotes the player i decide to be cooperative, n_i denotes the i choose to be selfish. Label `init` denotes randomly pick up an honest player as a initiator to send out a message. State labelled as $i : (\texttt{status}, \texttt{from}, \texttt{to}, \texttt{sender})$ implies state $(\texttt{status}, \texttt{from}, \texttt{to}, \texttt{sender}) \in S_i$ for player $i \in \{0, 1, 2, 3, 4\}$, player $i = 0$ is used to model a coordinator, where the $\texttt{status} = 0, 1, 2, 3, 4, 5$ denotes that the sender is randomly picked up, the message is sent, the player decides to be cooperative, the player decides to be selfish and the message is discarded, and the message reaches the destination respectively.

1) Cost structures for honest players $i = 1, 2$: assigns a cost of 1 and 2 to all transitions labelled with 's_1' and 'r_1' to player 1 respectively; and assigns a cost of 2 and 3 to all transitions labelled with 's_2' and 'r_2' to player 2 respectively.
2) Property specification for honest players: the utility function of player i is defined as the probability of *good* behaviours/costs. We say a run is *good* if it reaches the destination without violating the anonymity properties.