Solving security models with perfect observability

Paolo Zappala[1,2], Amal Benhamiche[1], Matthieu Chardy[1], Francesco De Pellegrini[2], and Rosa Figueiredo[2]

[1] Orange Innovation, Orange, 44 Avenue de la République, Châtillon, 92320, France
`{name.surname}@orange.com`
[2] LIA, Avignon Université, 301 Rue Baruch de Spinoza, Avignon, 84140, France
`{name.surname}@univ-avignon.fr`

Extended abstract

Sequential models with perfect observability represent situations in which communication is public and observable by all the agents. Such models are applied within different domains of security, such as intrusion detection [1], blockchain protocols [4] and wiretap channels [2]. The extensive-form game with perfect information is the representation used to identify the solution of these models. To date, the literature provides methods to identify specific solutions for small-size extensive-form games. We provide the first method to identify all solutions that is also scalable with the size of the games.

We consider a game in extensive form with perfect information.

Definition 1 (extensive-form game). *An extensive-form game is a tuple $\Gamma = \langle \mathcal{I}, \mathcal{A}, H', H, P, u \rangle$, where:*

- $\mathcal{I} = \{1, \ldots, N\}$ *is the set of players;*
- H' *is the set of histories with* $\emptyset \in H'$*;*
- $\mathcal{A} : h' \in H' \to A$ *is a function that provides for every history a set of actions* A*, i.e., for all* $a \in A$*, we have* $h' + (a) \in H'$*;*
- $H = \{h \in H' | \mathcal{A}(h) = \emptyset\} \subset H'$ *is the set of outcomes;*
- $P : H' \setminus H \to \mathcal{I}$ *is a function that indicates which player* $P(h') \in \mathcal{I}$ *acts after observing the history* $h' \in H' \setminus H$*;*
- $u = (u_i)_{i \in \mathcal{I}}$*, with* $u_i : H \to \mathbb{R}$ *the utility function of player* $i \in \mathcal{I}$*.*

Every player picks a strategy, i.e., she chooses an action for every history observed. Formally, a *strategy* s_i of a player $i \in \mathcal{I}$ is a function $s_i \in S_i = \{s_i : h \in H_i \mapsto a \in \mathcal{A}(h)\}$ that gives for every history $h \in H_i = \{h \in H : P(h) = i\}$ an available action $a \in \mathcal{A}(h)$. A strategy profile $\overline{s} \in S := \times_i S_i$ is a *Nash equilibrium* if no player can increase her utility by changing unilaterally her strategy, i.e, if for every $i \in \mathcal{I}$ and for all $s_i \in S_i$ it holds $u_i(\overline{s}_i, \overline{s}_{-i}) \geq u_i(s_i, \overline{s}_{-i})$. A specific equilibrium can be computed efficiently with the backward induction algorithm (cf., e.g., [3]). The absence of methods to compute all the other Nash equilibria limits the insights for security models [3, 4]. Hereafter, we provide a method to enumerate all the Nash equilibria of an extensive-form game.

We define $I : H \times H \to \mathcal{I}$, where $I(h, h') = P(h \cap h')$, the function that maps the pair of outcomes $h, h' \in H$ with $h \neq h'$ to the player $I(h, h')$ that separates their paths from the root of the game tree. This relation between outcomes and strategies allows us to define a graph-based method to identify all the outcomes of Nash equilibria. We consider the following problem from graph theory.

Problem (MC)**.** *Existence of a maximal clique excluding a set of vertices*
Input. *$\langle H, E, X\rangle$ defining a graph $\langle H, E\rangle$ and a subset of vertices $X \subset H$.*
Output. *Is there a vertex set $C \subset H \setminus X$ that induces a maximal clique?*

We introduce the following algorithm to determine whether a target outcome is a Nash equilibrium. The iteration of the algorithm over all outcomes permits the enumeration of the equilibria.

```
ALGORITHM: NASH
Input: A game $\Gamma$, the set of its outcomes $H$, the function $I : H \times H \to \mathcal{I}$ and
        an outcome $h \in H$.
Output: Is $h$ a realisation of a Nash equilibrium?
$boolean = True$ ;
for $i \in \mathcal{I}$ do
    $H_i \leftarrow \{h' \in H \setminus \{h\} | I(h, h') = i\}$;
    $X_i \leftarrow \{h' \in H_i | u_i(h) < u_i(h')\}$;
    $E|_{H_i} = \{(h, h') \in H_i | I(h, h') = i\}$;
    if Output of Problem (MC) with input $\langle H_i, X_i, E|_{H_i}\rangle$ is negative then
        $boolean = False$;
    end
end
```

The algorithm has an efficient representation, i.e., it does not require to enumerate the strategies of the game. Furthermore, it relies on a graph theoretical problem whose instances are often easy to compute, as verified for the vast majority of outcomes we have tested. Finally, this algorithm is parallelizable, a condition that is key in order to scale the solution method for larger games.

References

1. Kantzavelou, I., Katsikas, S.: A generic intrusion detection game model in it security. In: Trust, Privacy and Security in Digital Business: 5th International Conference, TrustBus 2008 Turin, Italy, September 4-5, 2008 Proceedings 5. pp. 151–162. Springer (2008)
2. Mukherjee, A., Swindlehurst, A.L.: Jamming games in the mimo wiretap channel with an active eavesdropper. IEEE Transactions on Signal Processing **61**(1), 82–91 (2012)
3. Raya, M., Manshaei, M.H., Félegyházi, M., Hubaux, J.P.: Revocation games in ephemeral networks. In: Proceedings of the 15th ACM conference on Computer and communications security. pp. 199–210 (2008)
4. Zappalà, P., Belotti, M., Potop-Butucaru, M., Secci, S.: Game theoretical framework for analyzing blockchains robustness. In: 35th International Symposium on Distributed Computing. p. 25 (2021)