Approximation of (k, t)-robust equilibria

Tudor Dan Mihoc, Noémi Gaskó, Rodica Ioana Lung, Mihai Suciu

Babeş-Bolyai University, Cluj-Napoca, Romania

Abstract

Game theory models strategic and conflicting situations offering several solution concepts known as game equilibria, among which probably the most popular one is the Nash equilibrium. A less known equilibrium, called (k, t)-robust, has recently been used in the context of distributed computing. The (k, t)-robust equilibrium combines the concepts of k-resiliency and t-immunity: a strategy profile is k-resilient if there is no coalition of k players that can benefit from improving their payoffs by collective deviation, and it is t-immune if any action of any t players does not decrease the payoffs of the others. A strategy profile is (k, t)-robust if it is both k-resilient and t-immune. In this paper an evolutionary method of approximating (k, t)-robust equilibria is proposed and tested by means of numerical experiments on a benchmark constructed from a game that studies node behavior in a distributed system.

Keywords: game theory; distributed systems; (k, t)-robust equilibria

1 Introduction

The popularity of solution concepts offered by game theory depends on their practical properties and applicability, which is directly related to their computability. One of the most accepted postulates in game theory is related to the players rationality, interpretable also as selfishness, as each player is pursuing its own interests. The Nash equilibrium [15], which relies on this principle, is a state of the game such that all players reach the highest value for their utility function that can not be improved by unilateral deviation. However the Nash equilibrium cannot model every possible situation: in real life players may not act rational, they often (apparently) cooperate and may adopt unselfish behavior, creating a need for other concepts and leading to several Nash equilibrium refinements and other solution concepts.

One of the most appealing Nash refinement is the strong equilibrium [6] that considers combinations of strategies from which no group of players can profitably deviate and introducing the idea of resiliency. In [3] a state of the game is described to be k-resilient if the members of no group of k players can all do better by deviating from their strategies. On the other hand, the t-immune equilibrium models the situation in which fault arbitrary (or coordinated) behavior can occur to up to t players in a game without negatively affecting the rest of non-deviating players [8]. Combining these two concepts - resilience and immunity - leads to the (k, t)-robust equilibrium, that allows the study of different systems where some of the components adopt a Byzantine comportment [11], in order to determine their stability or even to predict their outcomes [1, 4, 10]. Thus, the (k, t)-robust equilibrium is appealing when dealing with large systems in which irrational behavior of some elements is expected, or for over simplified games that ignore some components in player's utility functions.

In particular, (k, t)-robust equilibria have been studied related to distributing computing problems [2, 3], cryptography [4], and concurrent games [7]. However, practical approaches are limited by two important factors: (i) the existence of (k, t)-robust equilibria is not guaranteed by theoretical results, and (ii) the lack of methods of computing it renders it useless even in situations in which it may exist.

In distributed computing, a problem approached with game theoretic tools is the coordination of a set of concurrent processes in spite of the faulty behavior of some of them, with the (k, t)-robust equilibrium a suitable solution concept, as the formulation in terms of Nash equilibria raised some questions about the accuracy of the model as well as the possibility to compute the solution [5]. While it is generally agreed that the (k, t)-robust equilibrium models more accurate this problem [2], the problem of its computability is still open.

In this paper we present a computational intelligence based method of approximating (k, t)-robust equilibria by using a differential evolution algorithm with fitness assignment based on non-domination with respect to a binary relation between strategy profiles. A simple steady state method is also used to support the statement that the search is actually guided by the relation between strategies. One of the challenges of presenting this approach comes from the lack of other methods of computing this equilibria type and that of actual benchmarks for testing it. In this context, to illustrate possible results, we constructed a game based on the distributed computing systems model presented in [14] by generalizing it to n players and using it as a benchmark for our tests. Each solution provided by our method is validated through a procedure that checks by enumerating all possible deviations if it is indeed a (k, t)-robust equilibrium.

2 (k,t)-robust equilibria

We consider a game in normal form $\Gamma = (N, S, U)$, where $N = \{1, ..., n\}$ is the set of players, $S = S_1 \times ... \times S_n$ is the set of strategy profiles of the game, and $U = (u_1, ..., u_n)$ are the players' utility functions, $u_i : S \to \mathbb{R}$. A state of the game is a strategy profile represented by a vector $s \in S$, $s = (s_1, ..., s_n)$ that contains the players options ($s_i \in S_i$). If $X \subset N$ and $p, q \in S$, (p_{-X}, q_X) denotes the strategy profile in which players from X play their strategies from q and those from $N \setminus X$ their strategies from p. We denote by $|K|$ the cardinality of set K.

To describe the (k,t)-robust equilibrium, we will use the k-resilient and t-immune equilibrium concepts.

Definition 1 *[7] A strategy s^* is a k-**resilient** equilibrium, if for all $K \subseteq N$, with $|K| = k$,*

$$u_i(s^*_K, s^*_{-K}) \geq u_i(s_K, s^*_{-K}),$$

for all $s_K \in S_K$, and for all $i \in K$.

Definition 2 *[7] A strategy $s^* \in S$ is t-**immune** if for all $T \subseteq N$ with $|T| = t$, all $s_T \in S_T$, and all $i \notin T$ we have:*

$$u_i(s^*_{-T}, s_T) \geq u_i(s^*).$$

By combining these two concepts, the (k,t)-robust equilibrium captures the situation of the game in which no coalition of k players is affected by the actions of any other t players, i.e. no matter what any t players choose, there will be k players who will have no incentive to deviate, as any deviation will decrease their payoff.

Definition 3 *[7] A strategy $s^* \in S$ is (k,t)-**robust** if for all $K, T \subseteq N$, such that $K \cap T = \emptyset$, $|K| = k$, $|T| = t$, for all $x_T \in S_T$, for all $y_K \in S_K$, for all $i \in K$:*

$$u_i(s^*_{-T}, x_T) \geq u_i(s^*_{-(K \cup T)}, y_K, x_T).$$

We remark that the $(1,0)$-robust equilibrium is a Nash equilibrium of a game [2].

Remark 1 *Other resources ([3, 2]) indicate a variation of the definitions presented above, in which the size of coalition K can be $|K| \leq k$ and the same with T, i.e. $|T| \leq t$. From our perspective, considering coalitions of size less then k and t respectively poses a different problem that remains to be studied in future work.*

3 Approximation of (k,t)-robust equilibria

A general framework for computing game equilibria can be constructed by introducing generative relations that, by allowing the comparison of two strategy profiles, can be used to guide the search of an heuristic methods towards a certain equilibrium type.

Generative relation for (k,t)-robust equilibrium A generative relation that characterizes an equilibrium concept is a relation $\tau \subset S \times S$ that has the property that the set of non-dominated solutions with respect to τ is equal to the set of equilibria of the game. A strategy profile $s \in S$ is non-dominated with respect to relation τ if there does not exist a $q \in S$ such that $(q,s) \in \tau$. The main purpose in defining such a relation is to use it for fitness assignment within an evolutionary algorithm in order to guide the search towards non-dominated solutions and thus compute game equilibria. A generative relation for characterizing Nash equilibria of the game can be found in [12]. In what follows we propose a generative relation for (k,t)-robust equilibria.

Let p and q two profile strategies from S, and $K \subset N, T \subset N$ with $K \cap T = \emptyset$. We denote by:

$$r_{K,T}(p,q) = |\{i \in K | u_i(p_{-T}, q_T) < u_i(p_{-(K \cup T)}, q_{(K \cup T)})\}$$

the number of players from K that can improve their payoffs by deviating from p to q, if all players in T deviate from p to q. If p is a (k,t)-robust equilibrium, then $r_{K,T}(p,q) = 0$ for any $K, T \subset N, K \cap T = \emptyset$ and any $q \in S$. Then we write:

$$r(p,q) = \sum_{\substack{K \subset N \\ |K|=k}} \sum_{\substack{T \subset N \\ K \cap T = \emptyset \\ |T|=t}} r_{K,T}(p,q)$$

It is also obvious that if p is a (k,t)-robust equilibrium then $r(p,q) = 0$ for any $q \in S$.

Definition 4 *We say the strategy p* ***is better than strategy*** *q* ***with respect to*** *(k,t)****-robust equilibrium****, and we write $p \prec_{kt} q$, iff:*

$$r(p,q) < r(q,p).$$

Thus, strategy p is considered better than q if less players from any subset K of size k can improve their payoff by deviating from p to q when all players in any T of size t deviate from p to q than vice-versa.

Definition 5 *The strategy profile $s^* \in S$ is called (k,t)* ***non-dominated****, or non-dominated with respect to the relation $\prec_{kt}$, iff $\nexists s \in S, s \neq s^*$ such that*

$$s \prec_{kt} s^*.$$

We consider $\prec_{kt}$ as a potential generative relation of (k,t)-robust equilibrium, i.e. the set of non-dominated strategies with respect to $\prec_{kt}$ approximates the (k,t)*-robust equilibria*. The proof that $\prec_{kt}$ is a generative relation is beyond the scope of this paper. In what follows we will use it to compute non-dominated solutions with respect to $\prec_{kt}$ and further check (by using the definition and enumerating all possible coalitions and situation) if the obtained solutions are indeed (k,t)-robust.

Algorithm 1 kt-CrDE

Randomly generate initial population P_0 of strategies;
while (not termination condition) **do**
  for each $l = \{1, ..., population\ size\}$ **do**
    create offspring $O[l]$ from parent l;
    if $O[l]$ *(k,t) dominates* the most similar parent j **then**
      $O[l]$ replaces parent j;
    end if
  end for
end while

3.1 Methods

As underlying algorithms we have used two methods: the Crowding based Differential Evolution (CrDE) [13], that has been used to compute Nash equilibria, was adapted to compute (k,t)-robust nondominated solutions by replacing the Nash ascendancy relation with relation $\prec_{kt}$; and a simple steady state evolutionary algorithm.

CrDE [16] is a method designed to compute multiple equilibria in one run that was already tested of multiple game settings, the algorithm is outlined in 1. CrDE uses a *DE/rand/1/exp* scheme with the modification that an offspring replaces the closest (using Euclidean distance) if it is better than it with respect to the searched equilibrium (algorithm 2).

Algorithm 2 CrDE - the *DE/rand/1/exp* scheme

Create offspring $O[l]$ **from parent** $P[l]$
$O[l] = P[l]$
randomly select parents $P[i_1]$, $P[i_2]$, $P[i_3]$, where $i_1 \neq i_2 \neq i_3 \neq i$
$n = U(0, dim)$
for $j = 0$; $j < dim \wedge U(0,1) < pc$; $j = j + 1$ **do**
  $O[l][n] = P[i_1][n] + F * (P[i_2][n] - P[i_3][n])$
  $n = (n + 1) \bmod dim$
end for

The steady state evolutionary algorithm (StEA) is a simple algorithm used to show that (k,t)-robust equilibria can be computed with almost any method by using the $\prec_{kt}$ relation and consequently that CrDE is not the only method that can be adapted in this manner. StEA evolves a population of strategy profiles (of length n)represented as binary arrays and randomly generated at the beginning of the search. We consider a one point crossover and strong mutation operators. Each iteration three individuals from the population are selected randomly, compared to each other and are sorted with respect to the $\prec_{kt}$ generative relation. Crossover is applied to the best two of them and one offspring is selected at random. We apply to the resulting individual the mutation operator. If the resulting offspring dominates the third individual, it will replace it in the original population, and if not the population will remain unchanged. We repeat this until a predefined number of iterations is reached in order to obtain a final population of profile strategies. The algorithm reports the set of non-dominated individuals with respect to $\prec_{kt}$ in the final population.

4 A distributed computing game

Modeling and solving games with many players/agents having different goals and/or under uncertainty represent some common traits of game theory and distributed computing. Thus, problems from the distributed computing field, such as Internet routing, security, resource allocation, and fault tolerance can be tackled by game theory [9]. In most cases, in order to better model the system, solutions of the game must take into account groups or coalitions of players and be robust to the deviation of non faulty/byzantine players.

In this paper we consider the game in [14] that constructs a model for nodes behavior in a distributed computing system (grid computing) with two nodes and generalize it to n players as a non-cooperative game in normal form $\Gamma = (N, S, U)$. The players $i \in \{1, ..., n\}$ are the grid's nodes and form the set N. Each player can Cooperate (c-strategy) or can refuse taking tasks from the other nodes (Defect, d-strategy). The time needed by a node to complete a task can be divided in two components: a serial time, with duration a_i and a parallel time b_i, respectively.

A central authority rewards the nodes that are willing to cooperate - quantified in the utility functions with m_c - and will give a bonus of m_t to the nodes that reduce their computing costs. The players utility functions are:

$$
U_i = \begin{cases}
-\left(a_i + p_i^1\right), & \text{if } |C| = 0 \\
-\left(a_i + p_i^2\right) + m_c, & \text{if } |C| = |N| \text{ and } p_i^2 >= b_i \\
-\left(a_i + p_i^2\right) + m_t + m_c, & \text{if } |C| = |N| \text{ and } p_i^2 < b_i \\
-\left(a_i + p_i^3\right) + m_t, & \text{if } |C| < |N| \text{ and } |C| \neq |0| \\
 & \text{ and } p_i^3 < b_i \text{ and } i \in D \\
-\left(a_i + p_i^3\right), & \text{if } |C| < |N| \text{ and } |C| \neq |0| \\
 & \text{ and } p_i^3 = b_i \text{ and } i \in D \\
-\left(a_i + p_i^4\right) + m_c, & \text{if } |C| < |N| \text{ and } |C| \neq |0| \\
 & \text{ and } i \in C \text{ and } p_i^4 >= b_i \\
-\left(a_i + p_i^4\right) + m_t + m_c, & \text{if } |C| < |N| \text{ and } |C| \neq |0| \\
 & \text{ and } i \in C \text{ and } p_i^4 < b_i
\end{cases} \tag{1}
$$

where:

$$
p_i^1 = b_i, p_i^2 = \sum_{j=1}^{n} \frac{b_j}{|N|}, p_i^3 = \frac{b_i}{|C|+1},
$$

$$
p_i^4 = \sum_{j \in C} \frac{b_j}{|C|} + \sum_{j \in D} \frac{b_j}{|C|+1}.
$$

If $n = 2$, $m_c = 0$, and $m_t = m$, game Γ is the same with the one proposed in [14].

5 Numerical experiments

To validate this approach to detecting (k, t)-robust equilibria both algorithms described above were tested on the distributed computer game.

Experimental set-up Game Γ was considered with 2, 5 and 10 players and different values of k and t. For the numerical experiments, we have used the following parameters $a_i = 1, b_i = 2, \forall i \in N$, and $m_c, m_t \in \{0, 1, 2, 3\}$. The results for this game may allow a decision maker estimate for different m_c and m_t the chances to achieve a system immune to random defections of some of the grid's nodes.

Parameters used by both methods are standard and were not fine tuned for this problem. There was no special effort involved in designing and adapting

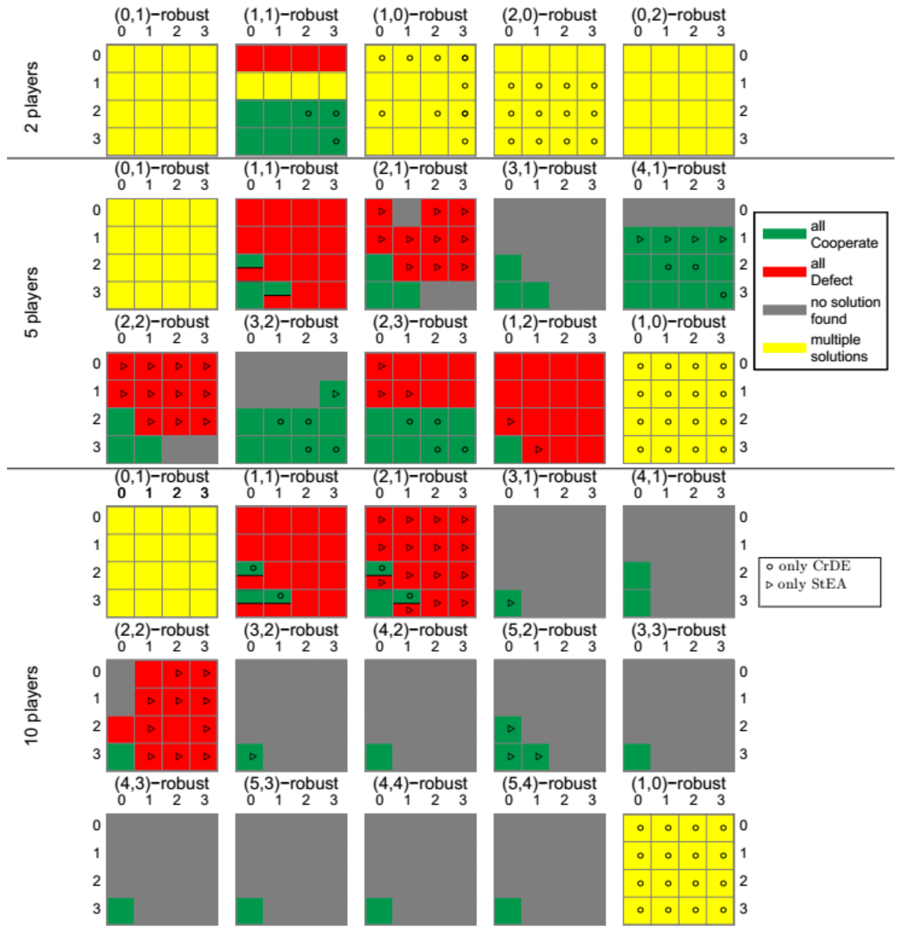

Figure 1: Results obtained by the two methods for 2, 5, and 10 players and different (k, t) settings. A red square indicates that the corresponding (k, t)-robust strategy is for all players to defect; a green square indicates that the robust strategy is cooperation for all players. A yellow square indicates multiple solutions; a square divided in two represents two strategy profiles (all cooperate and all defect). A circle indicates that only CrDE detected that particular solution; a triangle that only StEA reported that solution; a gray square marks no convergence.

the methods for computing the (k,t)-robust equilibria other than using the non-dominated selection based on relation $\prec_{kt}$.

The parameters for the CrDE algorithm are: $CF = 50$, $F = 0.5$ and $Pc = 0.9$, with a population of 50 strategies and 500 generations. StEA uses a population of 30 individuals. For 2 and 5 players the maxim number of iterations was 500, and for the game with 10 players 1900 iterations, respectively. The crossover probability is 0.80 and the mutation rate is 0.04. For each game setting both methods were run 10 independent times.

Results and discussion Results are presented as color matrices representing obtained (k,t)-robust strategies reported by the two methods in Figure 1. For each (k,t) pair the matrix represents results obtained for different m_c and m_t values. Detected solutions were validated using the definition of (k,t)-robust equilibria (definition 3) showing that the generative relation can be used for fitness assignment within evolutionary algorithms to detect this equilibrium type.

As depicted in the Figure 1 both algorithms guided by the generative relation detected almost the same solutions, with the remark that CrDE performed much better then StEA in terms of number of detected equilibria (as expected considering that it was designed to compute multiple solutions). However, for small number of players both methods were capable to compute multiple equilibria.

There were several cases where the two algorithms detected completely different solutions that were all (k,t)-robust; the difference might be caused by specific characteristics of each method.

For this particular game, having a tool that allows these equilibria to be detected, could be very practical. For example is important in a grid of 5 units to know that a specific configuration where 3 units play as a supervisor requests is stable even if the 2 others do as they like ($(3,2)$-robust) – see for example the 5 players table with $m_c = 2$ and $m_t = 1$ from the Figure 1: a stable configuration is for all to share tasks, otherwise the players will have the tendency to deviate in order to secure their gains, even if the new payoffs would not be as high as the current configuration.

6 Conclusions

The aim of this paper was to propose a computational tool to allows users to gain an insight of the phenomena taking place in complex systems - like distributed computing - by using a less known equilibrium concept, the (k,t)-robust equilibrium.

As many other equilibria concepts, the (k,t)-robust equilibrium "suffers" from limited applicability due to the lack of practical methods to compute it. Our approach to approximate (k,t)-robust equilibria, based on a generative relation and using a differential evolution algorithm, was tested on a game constructed from a distributed computing problem. The knowledge offered by

(k,t)-robust equilibria can be used also as a learning mechanism for players that can form coalitions of k players choosing the (k,t)-robust strategy and be safe from the deviation of t players (which is the t immune strategy). Further investigations are necessary to assess the possibilities of computing and using this equilibrium concept in practical, real-world applications.

Acknowledgement

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS - UEFISCDI, project number PN-II-RU-TE-2014-4-2560.

References

[1] I. Abraham. On Implementing Robust and Resilient Mediators. pages 1–3, 2007.

[2] I. Abraham, L. Alvisi, and J. Y. Halpern. Distributed computing meets game theory. *ACM SIGACT News*, 42(2):69, 2011.

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

[4] I. Abraham, D. Dolev, and J. Y. Halpern. *Lower bounds on implementing robust and resilient mediators*, volume 4948 LNCS. 2008.

[5] J. Aspnes, C. Busch, S. Dolev, P. Fatourou, C. Georgiou, and A. A. Shvartsman. Eight open problems in distributed computing. 2006.

[6] R. J. Aumann. Acceptable points in general cooperative n-person games. In R. D. Luce and A. W. Tucker, editors, *Contribution to the theory of game IV, Annals of Mathematical Study 40*, pages 287–324. University Press, 1959.

[7] R. Brenguier. Robust Equilibria in Concurrent Games. page 17, nov 2016.

[8] N. Gaskó, M. Suciu, R. I. Lung, T. Mihoc, and D. Dumitrescu. Players with unexpected behavior: t-immune strategies. An evolutionary approach. *Studia Universitatis Babes-Bolyai, Informatica*, 58(2), 2013.

[9] J. Y. Halpern. A computer scientist looks at game theory. *Games and Economic Behavior*, 45(1):114 – 131, 2003. First World Congress of the Game Theory Society.

[10] J. Y. Halpern. Beyond nash equilibrium: Solution concepts for the 21st century. *Proceedings of the twenty-seventh ACM symposium on Principles of distributed computing (PODC '08)*, pages 1–9, 2008.

[11] L. Lamport, R. Shostak, and M. Pease. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982.

[12] R. I. Lung and D. Dumitrescu. Computing nash equilibria by means of evolutionary computation. *Int. J. of Computers, Communications & Control*, 6:364–368, 2008.

[13] R. I. Lung, T. D. Mihoc, and D. Dumitrescu. Nash equilibria detection for multi-player games. In *Evolutionary Computation (CEC), 2010 IEEE Congress on*, pages 1–5. IEEE, 2010.

[14] I. Mishkovski, S. Filiposka, D. Trajanov, A. Grnarov, and L. Kocarev. Using Game Theory to Analyze Distributed Computing Systems. *2008.Telfor.Rs*, (November 2015), 2008.

[15] J. F. Nash. Non-cooperative games. *Annals of Mathematics*, 54:286–295, 1951.

[16] R. Thomsen. Multimodal optimization using crowding-based differential evolution. In *IEEE Proceedings of the Congress on Evolutionary Computation (CEC 2004)*, pages 1382–1389, 2004.