Nash Equilibrium Reoptimization is Hard

A. Celli and A. Marchesi

Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano
Via Ponzio 34/5, 20133, Milano, Italy
{andrea.celli,alberto.marchesi}@polimi.it

Abstract. We provide, to the best of our knowledge, the first study about *reoptimization* complexity of game-theoretical solutions. In a reoptimization problem, we are given an instance, its optimal solution, and a local modification, and we are asked to find the exact or an approximate solution to the modified instance. Reoptimization is crucial whenever an instance needs to be solved repeatedly and, at each repetition, its parameters may slightly change. In this paper, we focus on Nash equilibrium, being the central game-theoretical solution. We study the reoptimization of Nash equilibria satisfying some properties (i.e., maximizing/minimizing the social welfare, the utility of a player or the support size) for some different local modifications of the game (i.e., modification of a payoff or addition/removal of an action), showing that such problems are NP-hard. Furthermore, we assess the approximation complexity of the aforementioned problems, showing that it matches the complexity of the original (non-reoptimization) problems. Finally, we show that, when finding a Nash equilibrium is thought as an optimization problem, reoptimization is useful for finding approximate solutions. Specifically, it allows one to find ϵ-Nash equilibria with smaller ϵ than that of the solutions returned by the best known approximation algorithms.

1 Introduction

The design of computational tools for tackling strategic scenarios has been a central problem in Artificial Intelligence for several years. The main goal is the development of software/physical agents capable of behaving optimally when facing strategic opponents. This is achieved by modelling a scenario by means of models from non-cooperative game theory [11] and by employing algorithmic tools [20] to search for optimal solutions (a.k.a. equilibria), each one specifying the best strategies the agents can play. A crucial issue is the study of the complexity of equilibrium-finding problems and the design of efficient algorithms scaling up in real-world applications, e.g., as in Security Games [23].

A key solution concept is the Nash Equilibrium (NE), which prescribes strategies so that no player has any incentive in deviating unilaterally. Given the importance of NE concept, there has been a growing interest on settling its computational properties. More precisely, assessing the computational complexity of finding an NE has required more than 10 years and the problem was finally shown to be PPAD-complete [6] even for 2-player games [3]. Recall that PPAD $\subseteq$ FNP but $\mathsf{PPAD} \not\subseteq \mathsf{FNP}$-complete unless $\mathsf{NP} = \text{co-}\mathsf{NP}$ [17], and it is unlikely that this last equivalence holds. Furthermore, it is generally believed that $\mathsf{PPAD} \neq \mathsf{FP}$ and thus it is unlikely that there is an algorithm finding an NE that requires only polynomial time in the size of the game. When, instead, one searches for an optimal NE, e.g., maximizing social welfare or the payoff of a single player, the problem becomes NP-hard [14, 5].

Frequently, the same equilibrium-finding problem must be repeated whenever the value of some parameter changes. Practical examples of such a scenario are, e.g., the problems of bargaining among economic agents [13] and games for security [23]. In these settings, the structure of the game may gradually change during time as, for example, costs in bargaining games could change or, in a security game, the available resources or the values of the targets may vary at different times of the interaction. Another interesting related scenario is when learning tools are paired with optimization algorithms (e.g., in an online fashion, see [19]) to continuously refine the estimations of some parameters, in the attempt to minimize the regret due to the initial lack of information. In all these scenarios, a central question is whether the knowledge of an optimal solution (e.g., the Nash equilibrium with a certain property) can make the equilibrium-finding problem of the modified game easier. This problem is commonly known as *reoptimization* problem (see [2, Chapter 4]). Most NP-hard problems maintain the same complexity in reoptimization but, for many of them, reoptimization allows one to find better approximations, changing in some cases the approximation computational complexity class or just improving the approximation ratio in others. This happens, for instance, in scheduling problems [21] and in the Travelling Salesman Problem [1]. To the best of our knowledge, the problem of reoptimizing equilibria in games is unexplored in the literature so far. In this paper, we provide the first study on the reoptimization complexity of game-theoretical solutions and, more precisely, of Nash equilibria.

Original contributions. The original contributions provided in this paper are as follows. For the sake of presentation, at first, we focus on the exact reoptimization problem and, subsequently, on the approximation problem, whose treatment is more involved. Specifically, we show that the NE reoptimization problem is NP-hard when searching for NEs maximizing/minimizing the social welfare, a player's utility or the support size, and the local modification is either the modification of the value of some payoffs of the game or the addition/removal of actions. Then, we prove that the aforementioned problems are also hard to approximate, unless $\mathsf{P} = \mathsf{NP}$. This happens even when the modification is as small as possible (i.e., a payoff modification arbitrarily close to zero or the addition/removal of one action). We show that, for every modification, maximizing/minimizing social welfare or a player's utility is not in Poly-APX, while maximizing/minimizing the support size is not in Log-APX. In doing that, we also provide a complete picture of the approximation problems for the non-reoptimization case, while, so far, the results provided by the literature are only partial and do not analyze the specific approximation complexity classes. Finally, we study how an NE of the original game maps to those of the locally modified one, proving that, in case of payoff modifications with maximum magnitude δ, an NE of the original game is a δ-NE of the modified game. This shows that reoptimization may be useful when searching for approximate NEs.

2 Preliminaries

A *normal-form* game [22] is a tuple (N, A, U) in which $N = \{1, \ldots, n\}$ is the set of players, $A = A_1 \times \ldots \times A_n$, where A_i is the set of actions available to player $i \in N$, and $U = \{U_1, \ldots, U_n\}$, where $U_i : A \to \mathbb{R}$ is the utility function of player $i \in N$. A (*mixed*) *strategy* s_i for player $i \in N$ is a probability distribution over A_i, where we define with $s_i(a)$ the probability that $a \in A_i$ is played by player i. We denote with S_i the set of all mixed strategies of player i, i.e., the $(|A_i| - 1)$-simplex. A strategy s_i is said to be *pure* if there exists an $a \in A_i$ s.t. $s_i(a) = 1$. Letting $S = S_1 \times \ldots \times S_n$, a *strategy profile* $s \in S$ is a tuple specifying a strategy for each player. Given strategy s_i, its *support* is the set of actions $\{a | s_i(a) > 0\}$. Moreover, the support of a strategy profile is made by the union of the supports of players' strategies. With a slight abuse of notation, let $U_i(s)$ be the *expected utility* of player i when s is played, i.e., $U_i(s) = \sum_{a \in A} U_i(a) \prod_{j \in N} s_j(a_j)$ (where a_j is the action of player j in a).

A *Nash equilibrium* (NE) [18] is a strategy profile s s.t., for each $i \in N$, for each $s'_i \in S_i$, $U_i(s_i, s_{-i}) \geq U_i(s'_i, s_{-i})$, where s_{-i} denotes the profile of players' strategies except for s_i. It is well known that the problem of finding an NE is PPAD-complete [6], even for 2-player games [3], and therefore it is unlikely to be solvable in polynomial time. Moreover, the problem is FIXP-complete for three or more players [10]. Additionally, [14] and [5] study the complexity of finding a NE optimizing certain properties—for instance the social welfare (i.e., the sum of players' utilities), the utility of a given player, and the size of the support at the equilibrium—showing that the problem is also inapproximable. The results that are relevant to our work are summarized in Table 1.

The literature also explores the idea of approximate NEs. Notice that the computation of an NE can be formulated as an optimization problem as follows. The NE constraints are relaxed so that players can play also non-optimal actions with strictly positive probability, provided they guarantee a regret, w.r.t. a best response, of at most ϵ (in additive sense), and the objective function to be minimized is ϵ. Multiple notions of approximate NE have been proposed. Most of the literature [16, 8, 24] studies the *ϵ-approximate Nash equilibrium* (ϵ-NE), which is a strategy profile s.t. no player can gain more than ϵ by unilaterally deviating from it. Formally, $s \in S$ is an ϵ-NE if, for each $i \in N$, for each $s'_i \in S_i$, $U_i(s) \geq U_i(s'_i, s_{-i}) - \epsilon$. Some works [15, 7] study a stronger notion of approximate equilibrium strategies, called *ϵ-well-supported Nash equilibrium* (ϵ-ws-NE). In an ϵ-ws-NE, a player plays with strictly positive probability only actions guaranteeing her an expected utility within ϵ from the best reachable value, given the strategies of the others. Formally, $s \in S$ is an ϵ-ws-NE if, for each $i \in N$, for each $a \in A_i$, if $s_i(a) > 0$ then, for every $a'_i \in A_i$, it holds $U_i(a, s_{-i}) \geq U_i(a'_i, s_{-i}) - \epsilon$. We remark that, while an ϵ-ws-NE is always an ϵ-

NE, the contrary is not generally true. However, given an ϵ-NE, one can construct in polynomial time an ϵ'-ws-NE, where ϵ' polynomially scales with ϵ [4].

Table 1. Known Complexity Results.

	Exact solution	Approximate solution
Max/Min social welfare	NP-hard [1]	$\notin$ Poly-APX[2]
Max/Min player utility	NP-hard [1]	$\notin$ Poly-APX[2]
Max support	NP-hard	$\notin$ Log-APX[2]
Min support	NP-hard	$\notin$ Log-APX[3]

In the derivation of our original complexity results, we employ two game gadgets introduced in [5] and [14], respectively. They are built starting from instances of SAT and SET-COVER, which are well-known NP-complete problems [12]. For clarity, we provide their brief definition.

Definition 1 (SAT). *SAT is defined as follows:*

- ***INSTANCE:*** *A set V of m variables and a collection C of c clauses.*
- ***QUESTION:*** *Is there a satisfying truth assignment for C?*

Definition 2 (SET-COVER). *SET-COVER is defined as follows:*

- ***INSTANCE:*** *A collection R of subsets of a set T, with $|R| = r$ and $|T| = t$, and a positive integer k.*
- ***QUESTION:*** *Does R contain a cover for T of size k i.e., a subset $R' \subseteq R$ with $|R'| = k$ and s.t. $\bigcup_{\rho \in R'} \rho = T$?*

Now, we highlight the structure of the two aforementioned gadgets.

Definition 3 (SAT-gadget). *Given a SAT instance with m variables and c clauses, and a real number $\epsilon > 0$, a* SAT-gadget *is a normal-form game $\Gamma_\epsilon^{SAT} = (\{1,2\}, A^{SAT}, U^{SAT})$, with $|A_1^{SAT}| = |A_2^{SAT}| = 3m + c + 1$, s.t.:*

- *there exists an NE where each player's expected utility is $m - 1$ iff the SAT instance is satisfiable;*
- *the only other NE provides both players an expected utility of ϵ.*

Remark 1 *In a SAT-gadget, the support size of players' strategies at an equilibrium is, respectively, m for the NE providing $m - 1$ utility, 1 for the other one.*

[1] Results on Min problems are novel, see Remark 2.

[2] Inapproximability results on Max problems refine those presented in [5], where the authors just show that the problems are not in the APX class. The refined results can be derived similarly to Theorems 5-7.

[3] The inapproximability of Min support is a novel result which has never been studied before. We prove this result in Lemma 1.

Definition 4 (SET-COVER-gadget). *Given a SET-COVER instance with t items and r subsets of items, a SET-COVER-gadget is a normal-form game $\Gamma^{SC} = (\{1, 2\}, A^{SC}, U^{SC})$, with $|A_1^{SC}| = t + 1$ and $|A_2^{SC}| = r + 1$, s.t. there exists an NE with support for each player no more than k iff the SET-COVER instance has a cover of size k.*

3 Reoptimization Results

We focus on the following question: does the knowledge of an optimal NE, w.r.t. a certain property, help one in finding a new optimal solution for a slightly modified game? This question is crucial every time the game is repeated in time and, at every repetition, a slight modification to the game may be introduced. Knowing the solution of the original (pre-modified) game could, in principle, avoid one to solve the game from scratch making the optimization problem easier. Roughly speaking, the answer to the above question is no, unless $\mathsf{P} = \mathsf{NP}$. To prove such a result, let us formally define the reoptimization framework in the context of NE optimization.

Definition 5. *Given a property π and a local modification μ, the reoptimization problem $\mathsf{RE\text{-}NE}(\pi, \mu)$ is defined as follows:*

- ***INPUT**: $(\Gamma, \Gamma', \hat{s})$, where Γ is a normal-form game, Γ' is the modified game obtained by applying μ to Γ, and $\hat{s}$ is the optimal NE w.r.t. π over Γ.*
- ***OUTPUT**: the optimal NE $\hat{s}'$ w.r.t. π over Γ'.*

In this work we focus on the following properties π characterizing a NE:

- maximum (minimum) social-welfare (MAX-SW, MIN-SW);
- maximum (minimum) utility for a given player (MAX-REV, MIN-REV);
- maximum (minimum) support size at the equilibrium (MAX-SUPP, MIN-SUPP).

We denote with Π the set of all properties. Furthermore, we focus on the following local modifications μ to the original game:[4]

- payoff modification of a single outcome (PAYOFF);
- addition and removal of an action (ADD, REM).

For the sake of presentation, let $\mathsf{NE}(\pi)$ be the problem of finding an optimal NE w.r.t. property π in a given normal-form game.

In Sections 4 and 5, we study the intractability of the problems of finding, respectively, an exact and an approximate solution to $\mathsf{RE\text{-}NE}(\pi, \mu)$, for each pair π, μ. Table 2 summarizes our results.

[4] We do not take into consideration the introduction/removal of a player since it cannot be considered a local modification of the game.

Table 2. Reoptimization Intractability Results.

	PAYOFF	ADD	REM
MAX/MIN-SW	NP-hard $\notin$ Poly-APX	NP-hard $\notin$ Poly-APX	NP-hard $\notin$ Poly-APX
MAX/MIN-REV	NP-hard $\notin$ Poly-APX	NP-hard $\notin$ Poly-APX	NP-hard $\notin$ Poly-APX
MAX/MIN-SUPP	NP-hard $\notin$ Log-APX	NP-hard $\notin$ Log-APX	NP-hard $\notin$ Log-APX

4 NP-Hardness Results

In this section, we give the formal proofs of the hardness of RE-NE(π, μ) for each π and μ. Initially, we focus on the case in which $\mu \in \{\text{PAYOFF}, \text{ADD}\}$.

Theorem 1. *RE-NE(π, μ) is NP-hard for each $\pi \in \Pi$ and for each $\mu \in \{\text{PAYOFF}, \text{ADD}\}$, even for 2-player games.*

Proof. We start by considering the case $\mu = \text{PAYOFF}$. We show that the existence of a polynomial-time algorithm $\mathcal{A}$ solving RE-NE(π, μ) would allow one to solve NE(π) in polynomial time, therefore leading to a contradiction. Given a generic normal-form game $\Gamma^k = (\{1, 2\}, A^k, U^k)$, where k denotes the number of outcomes of the game, we define $\Gamma^0 = (\{1, 2\}, A^0, U^0)$ s.t. $A_i^0 = A_i^k$, for each $i \in \{1, 2\}$, and $U_1^0(a) = U_2^0(a) = 0$ for each $a \in A^0$. Clearly, every strategy profile in Γ^0 is an NE. Therefore, we select an appropriate $\hat{s}_0$, according to π, as the initial optimal solution. Specifically, if $\pi = \text{MAX-SUPP}$, we set $\hat{s}_0$ to any strategy profile with full support for both players, otherwise, if $\pi = \Pi \setminus \{\text{MAX-SUPP}\}$, we set $\hat{s}_0$ to any pure strategy profile. Then, we can define a sequence of PAYOFF transformations that allows one to obtain Γ^k starting from game Γ^0. Specifically, a transformation that leads from Γ^t to Γ^{t+1}, with $t = 0, \dots, k-1$, is s.t., for a given $a' \in A^t$ for which $U_1^t(a') \neq U_1^k(a')$ or $U_2^t(a') \neq U_2^k(a')$, it sets $U_i^{t+1}(a') = U_i^k(a')$ for each $i \in \{1, 2\}$ and keeps the other payoffs unchanged. Notice that the sequence of games $\Gamma^0, \Gamma^1, \dots, \Gamma^k$ requires a number of transformations to reach Γ^k that is polynomial in the size of the game. Therefore, starting from $(\Gamma^0, \Gamma^1, \hat{s}_0)$, we can apply $\mathcal{A}$ to any $(\Gamma^t, \Gamma^{t+1}, \hat{s}_i)$ to produce, in polynomial time, $\hat{s}_{t+1}$, up to $\hat{s}_k$. Thus, $\hat{s}_k$ being the optimal NE according to π in Γ^k, we reach the contradiction.

Let us now consider $\mu = \text{ADD}$. A reasoning similar to the one used above applies. In particular, given a generic normal-form game $\Gamma^k = (\{1, 2\}, A^k, U^k)$, let Γ^0 be $(\{1, 2\}, \{x\} \times \{y\}, U^0)$ s.t. $x \in A_1^k$, $y \in A_2^k$ and $U_i^0(x, y) = U_i^k(x, y)$, for each $i \in \{1, 2\}$. The sequence of ADD transformations that allows one to obtain Γ^k from Γ^0 requires a number of steps polynomial in the size of Γ^k, where each step of the sequence adds an action to one of the players. Therefore, by assuming the existence of $\mathcal{A}$, we reach the same contradiction. □

In the following, we focus on the case $\mu = \text{REM}$, which is more involved since the reasoning underlying the proof of Theorem 1 cannot be applied.

Theorem 2. *$\mathsf{RE\text{-}NE}(\pi, \mu)$ is NP-hard for $\pi \in \{\mathsf{MAX\text{-}SW}, \mathsf{MIN\text{-}SW}, \mathsf{MAX\text{-}REV}, \mathsf{MIN\text{-}REV}\}$ and $\mu = \mathsf{REM}$, even for 2-player games.*

Proof. In order to prove the result, we show the hardness of the decision version of our problem, which asks for an NE of the modified game having value of π greater than or equal to a given constant. Let us first focus on the case $\pi \in \{\mathsf{MAX\text{-}SW}, \mathsf{MAX\text{-}REV}\}$, and consider a SAT-gadget Γ_ϵ^{SAT} with $0 < \epsilon < m - 1$, where m is the number of variables of the SAT instance embedded in Γ_ϵ^{SAT}, as in Definition 3 (in the following proofs we omit the definition of m, giving it the same meaning). Let us define a game $\Gamma = (\{1, 2\}, A, U)$ s.t. $A_1 = A_1^{SAT} \cup \{x\}$, $A_2 = A_2^{SAT} \cup \{y\}$ and U is equal to:

- $U_i(a) = U_i^{SAT}(a)$, $\forall i \in \{1, 2\}, \forall a \in A^{SAT}$;
- $U_1(x, a_2) = U_2(x, a_2) = -M$, $\forall a_2 \in A_2^{SAT}$;
- $U_1(a_1, y) = U_2(a_1, y) = -M$, $\forall a_1 \in A_1^{SAT}$;
- $U_1(x, y) = U_2(x, y) = m$;

where M is a sufficiently large constant (i.e., any $M \geq 4$), making $-M$ the lowest payoff in the game. Therefore, Γ preserves the NEs of Γ_ϵ^{SAT}, i.e., given an equilibrium s for Γ_ϵ^{SAT}, playing the actions of Γ corresponding to the support of s according to the same probability distribution leads to an equilibrium. Γ has also the new equilibrium (x, y), which is the optimal equilibrium w.r.t π. If we apply REM to Γ removing action x (or, equivalently, y), we obtain a new game Γ' s.t. its NEs are only those of Γ_ϵ^{SAT}. Therefore, Γ' has an equilibrium with value for the property π greater than or equal to $m-1$ iff the SAT instance embedded in Γ_ϵ^{SAT} is satisfiable.

Similarly, when $\pi = \{\mathsf{MIN\text{-}SW}, \mathsf{MIN\text{-}REV}\}$, we follow the same reasoning, by setting $\epsilon > m - 1$ and $U_1(x, y) = U_2(x, y) = m - 2$. □

The proof of the previous theorem suggests the following result about optimal NEs without reoptimization.

Remark 2 *Using a reduction similar to that due to [5], setting $\epsilon > m - 1$, it follows that $\mathsf{NE}(\pi)$, for $\pi \in \{\mathsf{MIN\text{-}SW}, \mathsf{MIN\text{-}REV}\}$, is NP-hard, even for 2-player games.*

Theorem 3. *$\mathsf{RE\text{-}NE}(\mathsf{MAX\text{-}SUPP}, \mathsf{REM})$ is NP-hard, even for 2-player games.*

Proof. Let us focus on the maximization of the overall support size at the equilibrium. The proof for the problem of maximizing the support of a single player follows the same reasoning.

Consider the decision version of $\mathsf{RE\text{-}NE}(\mathsf{MAX\text{-}SUPP}, \mathsf{REM})$, i.e., the problem of deciding whether Γ' has an NE with support size greater than or equal to a given constant. Let Γ_ϵ^{SAT} be a SAT-gadget, with $\epsilon > 0$. Γ is a normal-form game $(\{1, 2\}, A, U)$ s.t. $A_1 = A_1^{SAT} \cup \{x_i | 1 \leq i \leq m\}$, $A_2 = A_2^{SAT} \cup \{y_j | 1 \leq j \leq m\}$ and U is defined as:

- $U_i(a) = U_i(a)^{SAT}$, $\forall i \in \{1, 2\}$, $\forall a \in A^{SAT}$;

- $U_1(x_i, a_2) = U_2(x_i, a_2) = -M$, $\forall x_i \in A_1 \backslash A_1^{SAT}$, $\forall a_2 \in A_2^{SAT}$;
- $U_1(a_1, y_j) = U_2(a_1, y_j) = -M$, $\forall y_i \in A_2 \backslash A_2^{SAT}$, $\forall a_1 \in A_1^{SAT}$;
- $U_1(x_i, y_j) = U_2(x_i, y_j) = 1$, $\forall x_i \in A_1 \backslash A_1^{SAT}$, $\forall y_j \in A_2 \backslash A_2^{SAT}$;

where $-M$ is s.t. it is the lowest payoff in the game (i.e., any $M \geq 4$). Γ preserves the NEs of Γ_ϵ^{SAT} and also has a new set of equilibria given by all the possible probability distributions over actions in $A_1 \backslash A_1^{SAT}$ and $A_2 \backslash A_2^{SAT}$. Therefore, if the two players randomize uniformly over all x_i and y_j, $\hat{s}$ has support of size $2m$. Suppose to apply a REM transformation to a row in $A_1 \backslash A_1^{SAT}$ (or, equivalently, to a column in $A_2 \backslash A_2^{SAT}$). The resulting Γ' has an NE with support greater than or equal to $2m$ iff the SAT instance contained in Γ_ϵ^{SAT} is satisfiable. □

Theorem 4. *RE-NE(MIN-SUPP, REM) is NP-hard, even for 2-player games.*

Proof. We consider the problem of minimizing the overall support size at the equilibrium. The proof for the problem of minimizing the support size of a single player follows the same reasoning. We show that the reoptimization problem is hard by proving the hardness of its decisional counterpart.

Given a SET-COVER-gadget Γ^{SC}, we build $\Gamma = (\{1,2\}, A, U)$ s.t. $A_1 = A_1^{SC} \cup \{x\}$, $A_2 = A_2^{SC} \cup \{y\}$ and U is defined as:

- $U_i(a) = U_i^{SC}(a)$, $\forall i \in \{1,2\}$, $\forall a \in A^{SC}$;
- $U_i(x, a_2) = -M$, $\forall i \in \{1,2\}$, $\forall a_2 \in A_2^{SC}$;
- $U_i(a_1, y) = -M$, $\forall i \in \{1,2\}$, $\forall a_1 \in A_1^{SC}$;
- $U_i(x, y) = 1$, $\forall i \in \{1,2\}$;

where $M > 1$ (so that $-M$ is the lowest payoff in the game) and therefore, the set of NEs of Γ is equal to that of Γ^{SC} with the only addition of (x, y), which is the equilibrium minimizing support size. Γ' is obtained by applying REM to row x (or column y). Γ' has an NE with support size less than or equal to k iff the SET-COVER instance in Γ^{SC} has a cover of size k. □

5 Inapproximability Results

We show the inapproximability of RE-NE(π, μ), for every pair of π and μ. We initially focus on searching for an NE maximizing either the social welfare or the utility of a player.

Theorem 5. *RE-NE(π,μ) is not in Poly-APX for each $\pi \in \{$MAX-SW, MAX-REV$\}$ and $\mu \in \{$PAYOFF, REM, ADD$\}$, unless P = NP.*

Proof. We provide the proof for $\pi =$ MAX-SW. The proof for $\pi =$ MAX-REV follows the same reasoning. Let Γ_ϵ^{SAT} be a SAT-gadget, with $0 < \epsilon < \frac{m-1}{f(m)}$, where $f(m) = 2^m$ (the polynomiality of the reduction is preserved as $f(m)$ can be codified with m bits). Assume, by contradiction, that there exists a polynomial-time algorithm $\mathcal{A}$ providing an approximate solution to RE-NE(MAX-SW,μ), with approximation factor $r = \frac{1}{f(m)}$. Let $\Gamma = (\{1,2\}, A, U)$, where $A_1 = A_1^{SAT} \cup \{x\}$, $A_2 = A_2^{SAT} \cup \{y\}$, and U is so defined:

- $U_i(a) = U_i^{SAT}(a)$, $\forall i \in \{1, 2\}$, $\forall a \in A^{SAT}$;
- $U_i(x, a_2) = -M$, $\forall i \in \{1, 2\}$, $\forall a_2 \in A_2^{SAT}$;
- $U_1(a_1, y) = K$, $U_2(a_1, y) = -M$, $\forall a_1 \in A_1^{SAT}$;
- $U_1(x, y) = K$ and $U_2(x, y) = 0$;

where $M \geq 4$ and $K > 2(m-1)$. Let $\hat{s}$ be the NE with maximum social welfare, i.e., the pure strategy profile where $\hat{s}_1(x) = 1$ and $\hat{s}_2(y) = 1$. We define Γ' on the basis of the applied μ as follows:

- $\mu = \mathsf{PAYOFF}$. We build Γ' by applying a PAYOFF transformation to outcome (x, y), setting $U_1(x, y) = K - \delta$, where $\delta > 0$ is an arbitrarily small positive constant.
- $\mu = \mathsf{REM}$. We build Γ' by applying a REM transformation either to row x or column y.
- $\mu = \mathsf{ADD}$. We obtain Γ' by adding to Γ a new column y' s.t.:
  - $U_1(a_1, y') = K$, $\forall a_1 \in A_1^{SAT}$;
  - $U_2(a_1, y') = -M$, $\forall a_1 \in A_1^{SAT}$;
  - $U_1(x, y') = 0$ and $U_2(x, y') = 1$.

Notice that, in each Γ', the additional equilibrium of Γ w.r.t Γ_ϵ^{SAT} disappears, and no other equilibria are introduced. Therefore, in each of these cases, by employing $\mathcal{A}$, we would be able to obtain an approximate solution of value at least $\frac{2m-2}{f(m)} > 2\epsilon$ iff the SAT instance of Γ_ϵ^{SAT} is satisfiable. Otherwise, if it is unsatisfiable, the value of the approximate solution is less than or equal to 2ϵ. Thus, the existence of $\mathcal{A}$ would allow us to solve SAT in polynomial time. □

The case in which we search for an NE minimizing either the social welfare or the utility of a player is similar to the case studied above.

Theorem 6. *RE-NE(π,μ) is not in Poly-APX for each $\pi \in \{$MIN-SW, MIN-REV$\}$ and $\mu \in \{$ADD, REM, PAYOFF$\}$, unless P = NP.*

Proof. The proof follows from that of Theorem 5. The same reasoning applies if we choose $\epsilon > m - 1$ and $K < 2(m-1)$. □

The next three results focus on the case in which we search for an NE with maximum support.

Theorem 7. *RE-NE(MAX-SUPP, ADD) is not in Log-APX , unless P = NP.*

Proof. Consider a SAT-gadget Γ_ϵ^{SAT} (for any $\epsilon > 0$). Let us define a game $\Gamma = (\{1, 2\}, A, U)$ s.t. $A_1 = A_1^{SAT} \cup \{x_i | 1 \leq i \leq m\}$, $A_2 = A_2^{SAT} \cup \{y_j | 1 \leq j \leq m\}$, and U is defined as:

- $U_i(a) = U_i^{SAT}(a)$, $\forall i \in \{1, 2\}, \forall a \in A^{SAT}$;
- $U_1(x_i, a_2) = U_2(x_i, a_2) = -M$, $\forall x_i \in A_1 \backslash A_1^{SAT}$, $\forall a_2 \in A_2^{SAT}$;
- $U_1(a_1, y_j) = U_2(a_1, y_j) = -M$, $\forall y_j \in A_2 \backslash A_2^{SAT}$, $\forall a_1 \in A_1^{SAT}$;
- $U_1(x_i, y_j) = U_2(x_i, y_j) = 1$, $\forall x_i \in A_1 \backslash A_1^{SAT}$, $\forall y_j \in A_2 \backslash A_2^{SAT}$ with $i \neq j$;
- $U_1(x_i, y_j) = U_2(x_i, y_j) = -1$, $\forall x_i \in A_1 \backslash A_1^{SAT}$, $\forall y_j \in A_2 \backslash A_2^{SAT}$ with $i = j$;

where $M \geq 4$. Clearly, all the NEs of Γ_ϵ^{SAT} are preserved in Γ, and every additional NE of Γ does not have actions of A_1^{SAT} and A_2^{SAT} in its support. Moreover, Γ has always an equilibrium $\hat{s} = (\hat{s}_1, \hat{s}_2)$ s.t. $\hat{s}_1(x_i) = \frac{1}{m}$ for every $1 \leq i \leq m$ and $\hat{s}_2(y_j) = \frac{1}{m}$ for every $1 \leq j \leq m$, having support size $2m$. Also note that Γ has no NE with support size greater than $2m$. Suppose to apply an ADD transformation to Γ, leading to a new game, say Γ', by introducing an action x for player 1 s.t. $U_1(x, a_2) = U_2(x, a_2) = -M$ for any $a_2 \in A_2^{SAT}$, $U_1(x, y_j) = 2$ and $U_2(x, y_j) = 1$ for any $1 \leq j \leq m-1$, and $U_1(x, y_m) = U_2(x, y_m) = 2$. Notice that $\hat{s}$ is not an NE of Γ' since player 1 has an incentive to deviate from $\hat{s}_1$, playing x, which provides her a utility of 2, instead of $\frac{m-2}{m}$. In addition, simple arguments allow us to prove that the set of NEs of Γ' is equal to the one of Γ_ϵ^{SAT} with the addition of (x, y_m) (whose support size is 2).

By contradiction, assume there exists a polynomial-time approximation algorithm $\mathcal{A}$ for RE-NE(MAX-SUPP, ADD), which guarantees an approximation factor $r = \frac{1}{g(m)}$, where $g(\cdot)$ is a logarithmic function of the input. Clearly, if the SAT instance embedded in Γ_ϵ^{SAT} is satisfiable, then $\mathcal{A}$, when applied to $(\Gamma, \Gamma', \hat{s})$, produces an equilibrium $\hat{s}'$ with support size at least $\frac{2m}{g(m)} > 2$. Otherwise, the support size of $\hat{s}'$ is less than or equal to 2. Thus, the existence of such an algorithm would allow us to solve SAT in polynomial time. □

Theorem 8. *RE-NE(MAX-SUPP, PAYOFF) is not in Log-APX , unless P = NP.*

Proof. Let Γ be defined as in the proof of Theorem 7, where, in this case, U is changed so that:

- $U_1(x_1, y_j) = 0$ and $U_2(x_1, y_j) = (m-1)\delta$, $\forall y_j \in A_2 \backslash A_2^{SAT}, 1 \leq j \leq m-1$;
- $U_1(x_1, y_m) = 1$ and $U_2(x_1, y_m) = 0$;
- $U_1(x_2, y_1) = U_2(x_2, y_1) = 0$;
- $U_1(x_i, y_1) = 1$ and $U_2(x_i, y_1) = 0$, $\forall x_i \in A_1 \backslash A_1^{SAT}, 3 \leq i \leq m$;
- $U_1(x_i, y_j) = U_2(x_i, y_j) = 0, \forall x_i \in A_1 \backslash A_1^{SAT}, 2 \leq i \leq m, \forall y_j \in A_2 \backslash A_2^{SAT}, 2 \leq j \leq m-1$;
- $U_1(x_2, y_m) = 1$ and $U_2(x_2, y_m) = \delta$;
- $U_1(x_i, y_m) = 0$ and $U_2(x_i, y_m) = \delta$, $\forall x_i \in A_1 \backslash A_1^{SAT}, 3 \leq i \leq m$;

where δ is an arbitrarily small positive constant. As before, Γ has all the NEs of Γ_ϵ^{SAT}, with the addition of $\hat{s}$, which has support size $2m$ (the maximum possible). Now, let us apply a PAYOFF transformation to Γ, leading to Γ', by changing the utilities in (x_1, y_m), so that $U_1(x_1, y_m) = 1-\delta$ and $U_2(x_1, y_m) = 3\delta$. Simple considerations allow us to conclude that the NEs of Γ' are those of Γ_ϵ^{SAT}, with the addition of (x_2, y_m). The result easily follows, by contradiction, as in Theorem 7. □

Theorem 9. *RE-NE(MAX-SUPP, REM) is not in Log-APX , unless P = NP.*

Proof. Let Γ be defined as in the proof of Theorem 7 with the only difference being the definition of U, which is defined in the same way but for the subgame $\{x_1, \ldots, x_m\} \times \{y_1, \ldots, y_m\}$. Specifically:

- $U_1(x_1, y_1) = U_1(x_1, y_2) = -1$;
- $U_1(x_1, y_j) = 1$, $\forall y_j \in A_2 \setminus A_2^{SAT}, 3 \leq j \leq m$;
- $U_1(x_i, y_m) = -1$, $\forall x_i \in A_1 \setminus A_1^{SAT}, 2 \leq i \leq m-1$;
- $U_1(x_i, y_j) = \mathbb{1}_{i \neq j} - \mathbb{1}_{i=j}$, $\forall x_i \in A_1 \setminus A_1^{SAT}, 2 \leq i \leq m$, $\forall y_j \in A_2 \setminus A_2^{SAT}, 1 \leq j \leq m-1$;
- $U_1(x_m, y_m) = -3$.
- $U_2(x_i, y_j) = -1$, $\forall x_i \in A_1 \setminus A_1^{SAT}, 1 \leq i \leq m-1$, $\forall y_j \in A_2 \setminus A_2^{SAT}, 1 \leq j \leq m-1$;
- $U_2(x_m, y_j) = 1$, $\forall y_j \in A_2 \setminus A_2^{SAT}, 1 \leq j \leq m-1$;
- $U_2(x_i, y_m) = 1$, $\forall x_i \in A_1 \setminus A_1^{SAT}, 1 \leq i \leq m-1$;
- $U_2(x_m, y_m) = -2m + 3$.

Notice that Γ has always an optimal NE, say $\hat{s}$, with support size $2m$, s.t. $\hat{s}_1(x_i) = \frac{1}{m}$ for every $1 \leq i \leq m$ and $\hat{s}_2(y_j) = \frac{1}{m}$ for every $1 \leq j \leq m$. We build Γ' by applying a REM transformation to row x_m. Γ' has only an additional equilibrium w.r.t. Γ_ϵ^{SAT} (this can be shown by deleting strictly dominated rows/columns). The resulting additional equilibrium is (x_1, y_m) and, being in pure strategies, has support of size 2. Therefore, we can prove the theorem by contradiction, with the same reasoning as in the proof of Theorem 7. □

Now, we focus on the problem of searching for an NE with minimum support. Before stating our main results, let us introduce the following lemma, whose proof is useful in the sequel.

Lemma 1. *NE(MIN-SUPP) is not in Log-APX , unless P = NP.*

Proof. NE(MIN-SUPP) was shown to be NP-hard in [14]. We adapt the reduction to prove also the hardness of finding an approximate solution to the problem. The reduction is based on a SET-COVER gadget Γ^{SC} (for details see [14]), where, for our purpose, k is set equal to r (see Definition 4). By construction, each of its NEs corresponds to a valid cover for the SET-COVER instance embedded in the gadget. Moreover, the support size of an NE is precisely the size of the cover plus one, since player 1's actions correspond to the subsets composing the cover, whereas player 2 always plays her last action. Denoting with OPT_{NE} the optimal solution value of NE(MIN-SUPP), we have that $OPT_{SC} = OPT_{NE} - 1$, where OPT_{SC} is the size of an optimal cover. By contradiction, assume $\mathcal{A}$ is a polynomial-time approximation algorithm for NE(MIN-SUPP), providing an approximation ratio $r = \frac{1}{g(t)}$, where $g(\cdot)$ is a logarithmic function of the input. Let APX_{NE} be the value (in terms of support size) of the approximate solution which is returned by $\mathcal{A}$ on Γ^{SC}. Clearly, such solution is an NE, and it corresponds to a valid cover for the SET-COVER instance, with size $APX_{SC} = APX_{NE} - 1$ and, thus, we have $\frac{OPT_{SC}}{APX_{SC}} = \frac{OPT_{NE}-1}{APX_{NE}-1}$. Given that $\frac{OPT_{NE}}{APX_{NE}} \geq r$, we have $OPT_{NE} \geq rAPX_{NE}$. Thus, it follows $\frac{OPT_{NE}-1}{APX_{NE}-1} \geq \frac{rAPX_{NE}-1}{APX_{NE}-1} \geq 2r$ and therefore we have $\frac{OPT_{SC}}{APX_{SC}} \geq \frac{2}{g(t)}$. The last inequality leads to a contradiction, since SET-COVER cannot be approximated within a factor $\frac{c}{g(t)}$, where c is a constant greater than one [9]. □

Now, we can state the main results.

Theorem 10. *RE-NE(MIN-SUPP, μ) is not in Log-APX for each $\mu \in \{$PAYOFF, ADD, REM$\}$, unless P = NP.*

Proof. Let us first prove the case μ=PAYOFF. Given a SET-COVER gadget Γ^{SC}, where we set $k = r$ (see Definition 4), let us define $\Gamma = (\{1,2\}, A, U)$ s.t. $A_1 = A_1^{SC} \cup \{x\}$, $A_2 = A_2^{SC} \cup \{y\}$, and U is so defined:

- $U_i(a) = U_i^{SC}(a)$, $\forall i \in \{1,2\}, \forall a \in A^{SC}$;
- $U_1(x, a_2) = U_2(x, a_2) = -M$, $\forall a_2 \in A_2^{SC}$;
- $U_1(a_1, y) = 1$, $U_2(a_1, y) = -M$, $\forall a_1 \in A_1^{SAT}$;
- $U_1(x, y) = U_2(x, y) = 2$;

where $M > 1$ and therefore, $-M$ is lower than any other payoff in the game. Observe that the NEs of Γ are the same as those of Γ^{SC}, with the only addition of (x, y) (having support size 2). Suppose Γ' is built by applying a PAYOFF transformation to Γ, by changing U so that $U_1(x, y) = 1 - \delta$. Clearly, the set of NEs of Γ' coincides with the one of Γ^{SC}. Suppose, by contradiction, there exists a polynomial-time approximation algorithm solving RE-NE(MIN-SUPP, PAYOFF), providing an approximation factor $r = \frac{1}{g(t)}$, where $g(\cdot)$ is a logarithmic function of the input. Following a reasoning similar to that adopted in the proof of Lemma 1, we can show that the existence of such an algorithm would allow us to obtain an $\frac{1}{g(t)}$-approximate solution for a generic SET-COVER instance, which is a contradiction.

The proofs for the other cases follow a similar structure: Γ is defined as before, while Γ' is obtained by applying a specific transformation to Γ, which makes the set of NEs of Γ' equal to the one of Γ^{SC}. In particular, for μ = ADD, the result is achieved by applying an ADD transformation which introduces a new action x' into A_1, s.t. $U_1(x', a_2) = U_2(x', a_2) = -M$, for any $a_2 \in A_2^{SC}$, $U_1(x', y) = 3$, and $U_2(x', y) = -M - 1$. Instead, for μ = REM, the considered REM operation simply removes action x from A_1. □

Given the negative results presented in this section, a natural question is whether the knowledge of an optimal NE in the original game may help in approximating the optimum in the locally modified one when we upper bound the magnitude of the modification. Given that ADD and REM are already elementary operations, since they consider the addition (respectively, the removal) of a single action, we concentrate on PAYOFF modifications where payoffs are either increased or decreased at most of $\delta > 0$, which we call δ-PAYOFF.

The following example shows how the value of an optimal NE of the modified game (in terms of MAX-SW) can be arbitrarily worse than that for the original game, even for δ arbitrarily close to zero.

Example 1 *Consider the game Γ in Figure 1, where M is an arbitrarily large value. In this game, the optimal NE for MAX-SW is the bottom right outcome, which has social welfare $M + 1$. Suppose to apply a δ-PAYOFF transformation*

$$\begin{array}{c|c|c|}
 & a_2^1 & a_2^2 \\ \hline
a_1^1 & 1,1 & 1,0 \\ \hline
a_1^2 & 0,0 & 1,M \\ \hline
\end{array}
\qquad
\begin{array}{c|c|c|}
 & a_2^1 & a_2^2 \\ \hline
a_1^1 & 1,1 & 1,0 \\ \hline
a_1^2 & 0,0 & 1-\delta,M \\ \hline
\end{array}$$

$$\Gamma \qquad\qquad\qquad \Gamma'$$

Fig. 1. Arbitrarily small PAYOFF modifications may disrupt social welfare.

to such outcome, where $\delta > 0$ is a constant arbitrarily close to zero, reducing player 1's utility from 1 to $1-\delta$ (see Γ' in Figure 1). Clearly, (a_1^2, a_2^2) is no more an NE, and the new optimal NE becomes the top left outcome, which has social welfare 2. Therefore, the ratio between the two optimal values is $\frac{M+1}{2}$, which goes to infinity as M grows.

We can construct similar examples showing that the same holds for maximization/minimization of the revenue and of the support size. Indeed, simple modifications of the previous proofs (for $\mu = \mathsf{PAYOFF}$) show that the same inapproximability results hold even for $\mu = \delta\text{-}\mathsf{PAYOFF}$ with $\delta > 0$ arbitrarily close to zero, as stated in the following remarks.

Remark 3 *For any $\delta > 0$, RE-NE(π, δ-PAYOFF) is not in Poly-APX for each $\pi \in \{$MAX-SW, MIN-SW, MAX-REV, MIN-REV$\}$, unless P = NP.*

Remark 4 *For any $\delta > 0$, RE-NE(π, δ-PAYOFF) is not in Log-APX for each $\pi \in \{$MAX-SUPP, MIN-SUPP$\}$, unless P = NP.*

The above remarks directly follow, respectively, from the construction adopted in the proof of Theorems 5-6 and Theorems 8-10.

6 Approximate Nash Equilibria

We observed that even arbitrarily small modifications in the payoffs may generate considerable changes in the equilibria of the game. In particular, it may happen that, given a generic NE s for Γ, a PAYOFF transformation results in s no longer being an equilibrium point (disregarding any concept of optimality). Notice that this may happen even for arbitrarily small payoff modification, as shown in Example 1. Furthermore, the computational complexity of the reoptimization version of the problem of finding an NE (without requiring optimality w.r.t. a certain property) is the same as that of its non-reoptimization counterpart (i.e., finding an ϵ-approximate equilibrium minimizing ϵ). This can be shown with arguments similar to those of Section 4.

A question with useful practical implications, is whether knowing that s is a Nash equilibrium for Γ becomes a completely useless piece of information after a small δ-PAYOFF perturbation is applied to the original game. The answer is that, if we consider approximate Nash equilibria, strategy profile s preserves interesting properties and should be kept into account when considering whether computing a new solution for Γ' is worth it or not. Specifically, if we focus on normalized $[0,1]$-games[5] (a common assumption in the literature on approximate NEs), we can state the following theorem, which is a refinement of [4].[6]

Theorem 11. *Consider a generic normal-form game $\Gamma = (\{1,2\}, A, U)$ and a new game $\Gamma' = (\{1,2\}, A, U')$, obtained by applying a PAYOFF modification of magnitude (δ_1, δ_2), $\delta_i \in [-1,1]$ for each $i \in \{1,2\}$, to an outcome $(\overline{a}_1, \overline{a}_2) \in A$ of Γ so that $U'_i(\overline{a}_1, \overline{a}_2) = U_i(\overline{a}_1, \overline{a}_2) + \delta_i$ and $U'_i(\overline{a}_1, \overline{a}_2) \in [0,1]$, for each $i \in \{1,2\}$. Let s^* be an NE of Γ. Then, s^* is a δ-well supported Nash equilibrium for Γ', where $\delta = \max\{\delta_1, \delta_2\}$.*

Proof. First, we show that each action played with positive probability by the first player in s^*_1 leads to a payoff at most δ_1 smaller than the payoff at her best response against s^*_2. We denote with $S : A \to [-1,1]$ a function s.t. $S(\overline{a}_1, \overline{a}_2) = \delta_1$ and $S(a) = 0$ for every $a \in A \setminus \{(\overline{a}_1, \overline{a}_2)\}$. Moreover, let $\hat{a}_1 \in A_1$ be an action s.t. strategy $s_1(\hat{a}_1) = 1$ is a best response to s^*_2. For each $a_1 \in A_1$ s.t. $s^*_1(a_1) > 0$ it holds:

$$\begin{aligned}
&\sum_{a_2 \in A_2} U'_1(\hat{a}_1, a_2) s^*_2(a_2) - \sum_{a_2 \in A_2} U'_1(a_1, a_2) s^*_2(a_2) = \\
&= \sum_{a_2 \in A_2} (U_1(\hat{a}_1, a_2) + S(\hat{a}_1, a_2)) s^*_2(a_2) - \sum_{a_2 \in A_2} (U_1(a_1, a_2) + S(a_1, a_2)) s^*_2(a_2) = \\
&= \sum_{a_2 \in A_2} U_1(\hat{a}_1, a_2) s^*_2(a_2) - \sum_{a_2 \in A_2} U_1(a_1, a_2) s^*_2(a_2) + \\
&\qquad + \sum_{a_2 \in A_2} S(\hat{a}_1, a_2) s^*_2(a_2) - \sum_{a_2 \in A_2} S(a_1, a_2) s^*_2(a_2) \leq \\
&\leq \sum_{a_2 \in A_2} U_1(\hat{a}_1, a_2) s^*_2(a_2) + \sum_{a_1 \in A_1} \sum_{a_2 \in A_2} U_1(a_1, a_2) s^*_1(a_1) s^*_2(a_2) + |\delta_1| \leq |\delta_1|
\end{aligned}$$

The same reasoning holds for the second player, with a final upper bound of $|\delta_2|$. If we set $\delta = \max\{\delta_1, \delta_2\}$, the previous inequalities show that s^* is a δ-well supported NE for Γ'. □

Notice that, for $\delta < 0.3393$, s^* has better guarantees over Γ' than what we could obtain by applying the best polynomial-time approximation algorithm currently known [24]. Therefore, on a mildly modified game, strategy profile s^*

[5] Normalized $[0,1]$-games are games in which all the payoffs of the players are in $[0,1]$. We recall that any game is equivalent to a normalized $[0,1]$-game by means of an affine transformation.

[6] They obtain an ϵ-approximate NE with $\epsilon \leq 4\delta$, when perturbing each payoff with an arbitrary probability distribution over $[-\delta, \delta]$.

is still a valuable prescription for both players. This is an example in which reoptimization is useful, allowing one to outperform the best approximation algorithm known so far.

7 Conclusions

In this paper, we provide, to the best of our knowledge, the first study of the reoptimization complexity of game-theoretical solutions. We focus on Nash equilibria satisfying some specific properties (i.e., maximizing/minimizing the social welfare, the utility of a player, the support size), showing that the reoptimization complexity is NP-hard for some local modifications (i.e., modification of a payoff or addition/removal of an action). Furthermore, we show that reoptimization does not help even when one is searching for approximate solutions and this holds also when the modification is as small as possible. Instead, when one searches for approximate ϵ-Nash equilibria, reoptimization can help, allowing one to find approximations better than those returned by the best known algorithms.

In the future, we are interested in empirically evaluating reoptimization techniques. Although we show in this paper that reoptimization does not help in the worst case when searching for optimal Nash equilibria, preliminary results suggest that in the average case it is very useful. Furthermore, we are interested in investigating reoptimization complexity in Security Games models.

References

1. Claudia Archetti, Luca Bertazzi, and M Grazia Speranza. Reoptimizing the traveling salesman problem. *NETWORKS*, 42(3):154–159, 2003.
2. Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer Science & Business Media, 2012.
3. Xi Chen and Xiaotie Deng. Settling the complexity of two-player Nash equilibrium. In *ANN IEEE SYMP FOUND*, pages 261–272, 2006.
4. Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Computing Nash equilibria: Approximation and smoothed complexity. In *FOCS*, pages 603–612, 2006.
5. Vincent Conitzer and Tuomas Sandholm. New complexity results about Nash equilibria. *GAME ECON BEHAV*, 63(2):621–641, 2008.
6. Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM J COMPUT*, 39(1):195–259, 2009.
7. Constantinos Daskalakis, Aranyak Mehta, and Christos Papadimitriou. A note on approximate Nash equilibria. In *WINE*, pages 297–306. Springer, 2006.
8. Constantinos Daskalakis, Aranyak Mehta, and Christos Papadimitriou. Progress in approximate Nash equilibria. In *EC*, pages 355–358, 2007.
9. Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *STOC*, pages 624–633, 2014.
10. Kousha Etessami and Mihalis Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM J COMPUT*, 39(6):2531–2597, 2010.

11. Drew Fudenberg and Jean Tirole. *Game theory*. MIT Press, 1991.
12. MR Garey and DS Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, CA, 1979.
13. Nicola Gatti, Francesco Di Giunta, and Stefano Marino. Alternating-offers bargaining with one-sided uncertain deadlines: an efficient algorithm. *ARTIF INTELL*, 172(8-9):1119–1157, 2008.
14. Itzhak Gilboa and Eitan Zemel. Nash and correlated equilibria: Some complexity considerations. *GAME ECON BEHAV*, 1(1):80–93, 1989.
15. Spyros C Kontogiannis and Paul G Spirakis. Efficient algorithms for constant well supported approximate equilibria in bimatrix games. In *ICALP*, pages 595–606, 2007.
16. Richard J Lipton, Evangelos Markakis, and Aranyak Mehta. Playing large games using simple strategies. In *EC*, pages 36–41, 2003.
17. Nimrod Megiddo and Christos H Papadimitriou. On total functions, existence theorems and computational complexity. *THEOR COMPUT SCI*, 81(2):317–324, 1991.
18. J. Nash. Non-cooperative games. *ANN MATH*, pages 286–295, 1951.
19. Thanh Hong Nguyen, Rong Yang, Amos Azaria, Sarit Kraus, and Milind Tambe. Analyzing the effectiveness of adversary modeling in security games. In *AAAI*, 2013.
20. Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*, volume 1. Cambridge University Press Cambridge, 2007.
21. Markus W Schäffter. Scheduling with forbidden sets. *DISCRETE APPL MATH*, 72(1-2):155–166, 1997.
22. Yoav Shoham and Kevin Leyton-Brown. *Multiagent systems: Algorithmic, game-theoretic, and logical foundations*. Cambridge University Press, 2008.
23. Milind Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge University Press, 2011.
24. Haralampos Tsaknakis and Paul G Spirakis. An optimization approach for approximate Nash equilibria. In *WINE*, pages 42–56, 2007.