Constrained Pure Nash Equilibria in Graphical Games

Gianluigi Greco[1] and **Francesco Scarcello** [2]

Abstract. The rational behavior of non-cooperative players is often formalized by means of the game theoretic notion of Nash equilibrium. However, there are several practical applications (e.g., multi-agent planning, mechanism design, and routing protocols design), where the computation of any Nash equilibrium could not be satisfactory, since we are often interested only in equilibria that satisfy some additional requirements. Even though such Nash equilibria, called *constrained Nash equilibria*, received a great deal of interest, a comprehensive formalization and a deep investigation of the complexity issues related to their computation are still missing.

In this paper, we present a formal framework for specifying and working with these constraints, by focusing on graphical games, where a player p may be directly interested only on part of the other players, called neighbors of p. We study the computational complexity of the main problems arising in this framework for pure strategies. Furthermore, we identify some restrictions on players' interaction that make some of these problems easy, by exploiting the graph representation of the structure of the game, and a generalization of graph acyclicity called treewidth. In these tractable cases, we also provide highly-parallelizable algorithms for the computation of constrained Nash equilibria.

1 Introduction

Game theory (see, e.g., [15]) is a mathematical framework for representing the interaction of rational agents that try to achieve their (possibly contrasting) goals. Roughly, a *game* $\mathcal{G}$ consists of a set P of players each one having to perform some actions (or *strategies*). After a player $p \in P$ had chosen a strategy, she gets a payoff determined by her chosen action, as well as by the actions pursued by the other players she is interested in, called the *neighbors of* p. Then, the aim of p is to maximize such a payoff.

A widely accepted rational behavior for non-cooperative players is based on the notion of Nash equilibrium, which is guaranteed to exist for some kind of randomized strategies, after the famous Nash's theorem [14]. According to Papadimitriou, determining the complexity of computing a Nash equilibrium is one of the most important open questions [16], and in fact several interesting results about the computation of equilibria in particular settings had already appeared in the literature (see, e.g., [5, 3, 2, 12, 19]).

In this paper, we focus on the setting of *pure* Nash equilibria (see, e.g., [3, 6]), i.e., we assume that each player have to chose exactly one strategy among a set of possible ones in order to achieve her goal — conversely in the setting of *mixed* Nash equilibria, players express probabilities of choosing individual strategies. Recently, complexity issues related to pure Nash equilibria have been deeply investigated in [6], where it is proven that, even in very restrictive settings, determining whether a game has a pure Nash Equilibrium is NP-hard. In the same paper, some tractable classes of games have been identified, and efficient algorithms for the computation of their pure Nash equilibria have been proposed.

However, in many real world cases, such as multi-agent planning, mechanism design, and routing protocols design, it is not sufficient to compute any (pure) Nash equilibrium. Sometimes, we may want to ensure that a player does not choose some particular action, or we may be interested in Nash equilibria whose global outcomes satisfy some particular requirement.

Such Nash equilibria with additional properties, called *constrained Nash equilibria* in this paper, received a great deal of interest, because they can be applied fruitfully in many practical contexts. For instance, Koutsoupias and Papadimitriou [13] studied the problem of routing the traffic over a network, and recognized that these systems are often non-cooperative, since each player uses to selfish route its traffic, and aims at the maximization of its own utility. Then, the authors proposed the study of the *worst* Nash equilibrium, i.e., the configuration with the maximum social cost (summation of the expected players' utilities). The idea is that, by computing this parameter, it is possible to get equilibria corresponding to network configurations with a link-latency upper bound below some given threshold.

Other important applications of constrained Nash equilibria have been suggested by Conitzer and Sandohlm [2] in the context of automatic mechanism design. In this case, the focus is on the design of the game rules, which should guarantee that a "desirable" outcome (according to a given objective) is reached in equilibria, despite the fact that each agent is driven by her own interest.

Of course, most of the well-known results on Nash equilibria do not apply to this new setting. Thus, we would like to know what happens if we add constraints to the game, and whether computing such an equilibrium is a tractable problem, or under which restrictions it could be feasible in polynomial time.

In this paper, we face the above problems in the setting of **graphical games** [12, 8, 9, 11, 10, 19] where each player may be directly interested only on part of the other players — thus, for each player p, we encode its payoffs by a table containing a cell for each possible combination of the actions of p's neighbors and of p herself. This setting turned out to be very useful in large population games (modelling, e.g., the interactions of agents over the internet), where representing all utility functions extensively (as implicitly assumed in traditional applications of game theory [15]) is either inconvenient or unfeasible. Our main contributions are the followings.

- In Section 3, we provide a formal framework for working with games comprising (i) constraints on the outcomes, which are enforced over the payoffs of players, (ii) constraints on the actions, that may force players to choose or to avoid some action, and (iii)

[1] DEIS, University of Calabria, Rende, Italy. email: ggreco@si.deis.unical.it
[2] DEIS, University of Calabria and ICAR-CNR, Rende, Italy. email: scarcello@unical.it

objective functions on the global outcome, which allow us to compute equilibria that are optimal w.r.t. some (global) criterium.
- For each of the above class of constraints, we prove novel complexity results for the problem of deciding the existence of Nash equilibria. Specifically, we are able to identify hard cases (in Section 4) and new relevant tractable cases (in Section 5, that we believe correspond to situations that are likely to occur in real application scenarios.
- Specifically, in Section 5 we identify some restrictions on players' interaction based on the graphical representation of the structure of the game, by exploiting a generalization of graph acyclicity, called treewidth [17], and we prove that computing a pure Nash equilibrium that satisfy all the constraints is feasible in polynomial time for games having bounded treewidth, if the constraints are smooth – very roughly, with polynomially bounded output values.
- Finally, for all the tractable cases, we provide efficient algorithms that can be implemented on a logspace alternating Turing machine, showing that the computation is highly-parallelizable.

2 Games and Nash Equilibria

A *game* $\mathcal{G}$ is a tuple $\langle P, Neigh, Act, U\rangle$, where P is a non-empty set of distinct players, $Neigh$ and Act are functions, and U is a set of functions. In particular, for each player $p \in P$: $Neigh$ provides a set of players $Neigh(p) \subseteq P - \{p\}$, called neighbors of p; $Act(p)$ defines the set of her possible actions (also *strategies*); and U contains a table encoding her utility function $u_p : Act(p) \times_{j\in Neigh(p)} Act(j) \rightarrow \Re$. Intuitively, $Neigh(p)$ contains the players who potentially matter w.r.t. to p's utility function. Indeed, in general, a player may be not directly interested in all other players, and thus her utility function is defined only in terms of the actions played by her neighbors and by herself. These games are known as *graphical games* [12, 8, 9, 11, 10, 19], or games in *graphical normal form* (short: GNF) [6].

A possible outcome $\mathbf{x}$ of $\mathcal{G}$ is a *profile* (also known as *global strategy*), i.e., a set containing exactly one strategy for each player. Let $\mathbf{x}$ be a profile, p a player, and u_p the utility function of p. Then, we denote by $pay_p(\mathbf{x})$ the payoff of a player p w.r.t. $\mathbf{x}$, i.e., the output of u_p applied to the actions played by p and her neighbors according to the profile $\mathbf{x}$.

Example 1 *Consider the game $\mathcal{G}_1$ for the players p_1 and p_2, with $Act(p_1) = \{a, b, c\}$ and $Act(p_2) = \{d, e\}$, whose utility functions are shown in the table on the left in Figure 1. For each profile, the first (resp. second) number in the table is the utility function of p_1 (resp. p_2). For instance, for the profile $\mathbf{x}$ in which p_1 plays a and p_2 plays d, they get payoff 2 and 3, respectively.* □

Let us now formally define the main concept of equilibrium studied in this paper. Given a profile $\mathbf{x}$, a player p, and an individual strategy $\mathbf{y}$ for p we denote by $\mathbf{x}_{-p}[\mathbf{y}]$ the profile where the individual strategy of player p in $\mathbf{x}$ is replaced by $\mathbf{y}$. Then, $\mathbf{x}$ is a **pure Nash equilibrium** for $\mathcal{G}$ if, $\forall p \in P,\ \nexists \mathbf{y} \in Act(p)$ such that $pay_p(\mathbf{x}) < pay_p(\mathbf{x}_{-p}[\mathbf{y}])$.

Example 2 *Consider again the game $\mathcal{G}_1$ shown in Figure 1. The strategy in which p_1 and p_2 play a and d, respectively, is a pure Nash equilibrium. Indeed, it is easy to see that neither of them may improve her payoff by changing her action (keeping the action of the other player fixed). Conversely, the game $\mathcal{G}_2$, whose utility functions are reported in the table on the right of the same figure, does not admit any pure Nash equilibrium.* □

	a	b	c
d	2,3	0,1	1,1
e	1,2	3,3	0,1

$\mathcal{G}_1$

	t	f
t	1,−1	−1,1
f	−1,1	1,−1

$\mathcal{G}_2$

Figure 1. Tabular representation of games $\mathcal{G}_1$ and $\mathcal{G}_2$.

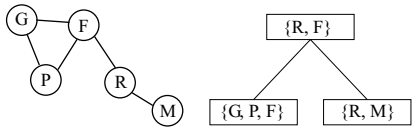


Figure 2. A dependency graph for the game $\mathcal{G}_3$ and a tree decomposition.

The interaction among players of $\mathcal{G}$ is usually represented by an undirected graph $\mathrm{G}(\mathcal{G}) = (P, E)$, called *dependency graph*, whose vertices coincide with the players of $\mathcal{G}$, and $\{p, q\} \in E$ if p is a neighbor of q, i.e., $p \in Neigh(q)$. Note that this graph is undirected, even if the neighborhood relationship is not necessarily symmetric. Indeed, it can be observed that, as far as Nash equilibria are concerned, any player's choice may also affect the strategies of the players she depends on.

As an example of a simple dependency graph, the game $\mathcal{G}_1$ is represented by $\mathrm{G}(\mathcal{G}_1)$ consisting of two vertices connected by means of an edge. In the left of Figure 2 we show the more involved interactions of players G, F, P, R and M in a game $\mathcal{G}_3$. Note that M is directly interested only in R, G is directly interested in both F and P, while there is no player directly interested in the strategies of all the other players in $\mathcal{G}_3$. It is worthwhile noting that, even if each player depends on a small number of other players directly, all of them depend on each other transitively. However, by encoding just the explicit direct relationships, the graphical normal form (GNF) provides a natural and compact representation of such games.

A fundamental structural property of graphs is *acyclicity*. Many hard problems turned out to be easy for acyclic structures. However, in many practical contexts, the graphs are in fact cyclic, but not very intricate. In these case, it can be useful to consider some generalizations of graph acyclicity, that allow us to identify and deal with structures having the same nice properties as acyclic graphs. In particular, we will use the notion of *treewidth* [17], which provides a measure of the degree of cyclicity of graphs, and is currently the broadest known generalization of graph acyclicity.

Definition 3 ([17]) A *tree decomposition* of a graph $G = (V, E)$ is a pair $\langle T, \chi\rangle$, where $T = (N, F)$ is a tree, and χ is a labelling function assigning to each vertex $p \in N$ a set of vertices $\chi(p) \subseteq V$, such that the following conditions are satisfied: (1) for each vertex b of G, there exists $p \in N$ such that $b \in \chi(p)$; (2) for each edge $\{b, d\} \in E$, there exists $p \in N$ such that $\{b, d\} \subseteq \chi(p)$; (3) for each vertex b of G, the set $\{p \in N \mid b \in \chi(p)\}$ induces a connected subtree of T. □

For instance, Figure 2 shows on the right a tree decomposition for $\mathrm{G}(\mathcal{G}_3)$ having width 2.

The *width* of the tree decomposition $\langle T, \chi\rangle$ is $\max_{p\in N} |\chi(p) - 1|$. The *treewidth* of G is the minimum width over all its tree decompositions. Let now $k > 0$ be a fixed constant. A tree decomposition of width at most k (if any) can be computed in linear time. We say that a game $\mathcal{G}$ has k-bounded treewidth if the treewidth of $\mathrm{G}(\mathcal{G})$ is at most k. A (possibly infinite) class of games C is said to have bounded treewidth if there is a k such that, for each game $\mathcal{G} \in C$, $\mathcal{G}$ has k-bounded treewidth.

3 Nash Equilibria with Additional Properties

In this section, we provide a formal framework for specifying additional properties of Nash equilibria, in terms of constraints on the payoffs of individual players, on the actions to be played, and on the global outcome of the game.

Let $\mathcal{G} = \langle P, Neigh, Act, U\rangle$ be a game and P' be a non-empty subset of the players. An *evaluation function* $f_{P'}$ for players in P' is any polynomial-time computable function that, for each combined strategy $\mathbf{x}$ for the players in $\bigcup_{p\in P'} Neigh(p) \cup P'$, maps the set $\{pay_i(\mathbf{x}) \mid i \in P'\}$ to a real number. Moreover, $f_{P'}$ is said *smooth* w.r.t. $\mathcal{G}$ if it is computable in logspace and, for each $\mathbf{x}$, $f_{P'}(\mathbf{x}) = O(\mathrm{poly}(|\mathcal{G}|))$, where $\mathrm{poly}(|\mathcal{G}|)$ is any polynomial in the size $|\mathcal{G}|$ of the game. Note that, since the output values of smooth functions are bounded by a polynomial in $|\mathcal{G}|$, then logarithmic space is sufficient for their encodings.

We next build constraints on profiles by exploiting different types of evaluation functions. A function $f_{P'}$ is said *local* w.r.t. a player $p \in P'$ if $P' \subseteq Neigh(p) \cup \{p\}$, i.e., if it can be evaluated by looking at the outcome of the neighbors of player p only.

Notice that local functions can be profitably used for modelling comparison among payoffs of (directly) interacting players. Otherwise, i.e., if there exists no player p such that $f_{P'}$ is local w.r.t. it, $f_{P'}$ is said *global*.

An interesting class of evaluation functions are those generated by binary operators, as described next. Let $\oplus$ be a commutative and associative binary operator over the set of real numbers, and let $\perp$ be its neuter element (e.g., 0 for $+$, 1 for $\times$). Then, the evaluation function induced by $\oplus$, referred to as the *aggregation function* $\mathsf{F}^{\oplus}$, is inductively defined as follows: $\mathsf{F}^{\oplus}_{\emptyset}(\mathbf{x}) = \perp$, and $\mathsf{F}^{\oplus}_{P'}(\mathbf{x}) = \mathsf{F}^{\oplus}_{P'-\{i\}}(\mathbf{x}) \oplus pay_i(\mathbf{x})$, with $i \in P'$. For instance, the aggregation function F^{+} induced by the $+$ operator is the *utilitarist social welfare*, while $\mathsf{F}^{\min}$ returns the minimum payoff, and $\mathsf{F}^{\max}$ returns the maximum payoff — note that all these examples are smooth functions if the payoffs are small (logarithmic-space encodable) w.r.t. the game size, or if they are bounded by some fixed constant.

We consider three types of constraints:

1. A *constraint on the payoffs* of the players in a set P' is an expression c of the form $[f_{P'}\ \mathtt{op}(c)\ k]$, with $k \in \Re$ and $\mathtt{op}(c) \in \{<, >, =, \neq, \leq, \geq\}$. Moreover, if f is a (smooth) aggregation function, then c is said *(smooth) aggregation constraint*. The semantics is as follows: a Nash equilibrium $\mathbf{x}$ satisfies c, denoted by $\mathbf{x} \models c$, if $f_{P'}(\mathbf{x})\ \mathtt{op}(c)\ k$. For instance, if $op(c)$ is $<$ then we require that the evaluation of $f_{P'}$ on the Nash equilibrium $\mathbf{x}$ is less than k. Finally, the constraint is said *local* if $f_{P'}$ is local.
2. A *constraint on the actions* of a player p is an expression c of the form $[\mathtt{op}(c)\ a]$, with $a \in Act(p)$ and $\mathtt{op}(c) \in \{=, \neq\}$. A Nash equilibrium $\mathbf{x}$ satisfies c of the form $[=\ a]$ (resp. $[\neq\ a]$), denoted by $\mathbf{x} \models c$, if p plays (resp. does not play) a in $\mathbf{x}$.
3. An *objective function* for the players in a set P' is an expression o of the form $[\mathtt{op}\ f_{P'}]$, where $\mathtt{op} \in \{\min, \max\}$ and $f_{P'}$ is an evaluation function. If $f_{P'}$ is a (smooth) function, then o is also said a (smooth) *objective function*. If $f_{P'}$ is an aggregation function $\mathsf{F}^{\oplus}_{P'}$, then we additionally require that $\min$ and $\max$ distribute over $\oplus$. A Nash equilibrium $\mathbf{x}$ is said *optimal* w.r.t. an objective function o of the form $[\min f_{P'}]$ (resp., $[\max f_{P'}]$), denoted by $\mathbf{x} \models o$, if there exists no Nash equilibrium $\mathbf{y}$ such that $f_{P'}(\mathbf{y}) < f_{P'}(\mathbf{x})$ (resp., $f_{P'}(\mathbf{y}) > f_{P'}(\mathbf{x})$).

Finally, a *constrained game* is a pair $\mathcal{G} = (G, C)$, where G is a game and C is a possibly empty set of constraints denoted also by $constr(\mathcal{G})$. The subset of global constraints in C is denoted by $constr^{glob}(\mathcal{G})$. A game is said an *optimization game*, if it is also equipped with an objective function, denoted by $obj(\mathcal{G})$.

An example of optimization game is the computation of the Nash equilibria that guarantee the best utilitarist social welfare, i.e., the optimal Nash equilibria with respect to the objective function $[\max \mathsf{F}^{+}]$.

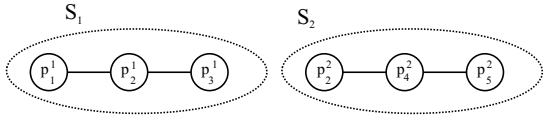


Figure 3. Reduction from EXACT COVER.

4 Hard Cases

In this section, we study the problems of deciding the existence of pure Nash equilibria in the presence of constraints, and of computing a Nash equilibrium (if any) that optimizes a given objective function. Following [8, 9, 6], we consider in our analysis games in graphical normal form.

Let us first focus on games with no objective function to be optimized. For this setting, deciding the existence of a pure Nash equilibrium is known to be NP-complete for general games. However, the problem is LOGCFL-complete for acyclic games and bounded treewidth games without constraints [6].

We recall that LOGCFL consists of all decision problems that are logspace reducible to a context-free language. We have: $\mathrm{AC}_0 \subseteq \mathrm{NC}_1 \subseteq \mathrm{LOGSPACE} \subseteq \mathrm{NL} \subseteq \mathrm{LOGCFL} \subseteq \mathrm{AC}_1 \subseteq \mathrm{NC}_2 \subseteq \mathrm{P}$. It follows that all problems in LOGCFL are highly parallelizable.

Note that, in this paper, we are interested in Nash equilibria satisfying additional properties, which is a more general and somehow more intricate case. In fact, such properties are expressed as constraints that may relate players that do not directly interact, i.e., which are not necessarily neighbors of each other. In this case, we are able to sharpen the hardness results of [6], by proving that computing constrained Nash equilibria is hard even for simple kinds of players' interactions, as in the acyclic games. We remark that previous results cannot be exploited here, since they were based on complicated interactions with cyclic dependency graphs.

Theorem 4 *Deciding whether a constrained game $\mathcal{G}$ has a Nash equilibrium satisfying all the constraints is* NP*-complete. Hardness holds even for acyclic games with a fixed number of actions, and with* $|constr(\mathcal{G})| = 1$.

Proof *(Sketch). Membership:* We can guess a profile $\mathbf{x}$, and verify in polynomial time that $\mathbf{x}$ is a Nash equilibrium for $\mathcal{G}$ and that all the constraints are satisfied.

Hardness: Recall that given the elements $I_1, ..., I_n$ and a number of sets $S_1, ..., S_m$, each one containing exactly three elements in $\{I_1, ..., I_n\}$, deciding whether there exists an exact cover, i.e., a set $\mathcal{C} \subseteq \{S_1, ..., S_m\}$ such that $\bigcup_{S_i\in\mathcal{C}} S_i = \{I_1, ..., I_n\}$ and $S_i \cap S_j = \emptyset$ for each $S_i, S_j \in \mathcal{C}$ with $i \neq j$, is NP-complete [4].

We define a GNF game $\mathcal{G}$ as follows: The set of players, denoted by P_I, contains exactly one player p^k_j for each item I_j in S_k. Let I_i, I_j and I_h be three elements in S_k. We define the neighborhood as follows: $Neigh(p^k_i) = \{p^k_j\}$, $Neigh(p^k_j) = \{p^k_i, p^k_h\}$, and $Neigh(p^k_h) = \{p^k_j\}$ — see Figure 3 for the acyclic undirected graph representation of the game associated to the sets $S_1 = \{p^1_1, p^1_2, p^1_3\}$ and $S_2 = \{p^2_2, p^2_4, p^2_5\}$.

Players in P_I may choose a strategy in the set $\{IN, OUT\}$. Let $\mathbf{x}$ be a profile. Then, each player $p \in P_I$ is such that

- $u_p(\mathbf{x}) = 1$, if p plays IN and all her neighbors play IN;

- $u_p(\mathbf{x}) = 0$, if p plays OUT and all her neighbors play OUT;
- $u_p(\mathbf{x}) = -2$, in all other cases.

Finally, $constr(\mathcal{G})$ contains the one constraint $[\Pi_{i=1}^{n} \sum_{k|I_i \in S_k} pay_{p_i^k}(\mathbf{x}) = 1]$.

Then, it is easy to see that there exists an exact cover $\Leftrightarrow$ $\mathcal{G}$ admits a Nash equilibrium satisfying $constr(\mathcal{G})$. □

Since general constraints with associated general evaluation functions lead to intractable games, one may wonder whether the use of aggregation functions make the problem any easier. Unfortunately, we next show that deciding the existence of constrained Nash equilibria is hard even in this restricted case. However, this time, hardness occurs only if we can have an unbounded number of constraints.

Theorem 5 *Let $\mathcal{G}$ be a game, and $constr(\mathcal{G})$ be a set of aggregation constraints. Then, deciding whether there exists a Nash equilibrium is* NP*-complete. Hardness holds even for acyclic games with a fixed number of actions.*

Proof *(Sketch).* Membership follows from Theorem 4. The hardness part is based on the same construction as in the proof of Theorem 4, by replacing the one constraint in $constr(\mathcal{G})$ by the following n constraints: $ac_i : [F^{+}_{\{p_k^i | I_i \in S_k\}} = 1], \quad \forall i.$ □

Optimization Games. We next consider the problem of computing the Nash equilibrium that optimizes a given objective $obj(\mathcal{G})$, by characterizing its complexity in terms of classes of search problems.

An NP *metric Turing machine* MT is a polynomial-time nondeterministic Turing machine that, on every computation branch, halts with the encoding of a binary number on its output tape. The output of MT is the maximum number over its computations. The class OptP contains all integer functions that are computable by an NP metric Turing machine. Moreover, $\mathrm{OptP}[O(\log n)]$ is the subclass of OptP containing all functions f whose value $f(x)$ has $O(\log n)$ bits, where $n = |x|$ (see [1]). For instance, computing the cardinality of a maximum clique or of a minimum vertex cover in a graph are $\mathrm{OptP}[O(\log n)]$ functions. Then, $\mathrm{FNP}//\mathrm{OptP}$ (resp., $\mathrm{FNP}//\mathrm{OptP}[O(\log n)]$) is the class of all partial multi-valued functions g computed by polynomial-time nondeterministic Turing machines T such that, for every x, $g(x) = T(x \cdot h(x))$, where $\cdot$ denotes the concatenation operator, and h is a function in OptP (resp., in $\mathrm{OptP}[O(\log n)]$).

Theorem 6 *Let $\mathcal{G}$ be a constrained game. Then, computing a Nash equilibrium (if any) that optimizes an objective function (resp., a smooth objective function) $obj(\mathcal{G})$ is* $\mathrm{FNP}//\mathrm{OptP}$*-complete (resp.,* $\mathrm{FNP}//\mathrm{OptP}[O(\log n)]$*-complete). Hardness holds even for the optimization of the utilitarist social welfare, and for games with a fixed number of actions and with $|constr(\mathcal{G})| = 0$.*

Proof *(Sketch). Membership:* The value v that optimizes the objective function can be computed by an NP metric Turing machine. Then, we can guess a profile $\mathbf{x}$ and verify in polynomial time that $\mathbf{x}$ is a Nash equilibrium for $\mathcal{G}$, that all the constraints are satisfied, and that the value of the objective function is exactly v. Moreover, note that if $obj(\mathcal{G})$ is a smooth objective function, then v can be computed in $\mathrm{OptP}[O(\log n)]$.

Hardness: Consider the $\mathrm{FNP}//\mathrm{OptP}[O(\log n)]$-complete problem X-MAXIMAL MODEL: Given a formula ϕ in conjunctive normal form on the variables $Y = \{Y_1, ..., Y_n\}$ and a subset $X \subseteq Y$, compute a satisfying truth assignment M for ϕ whose X-part is maximal, i.e., for every other satisfying assignment M' there exists a variable in X which is true in M and false in M'.

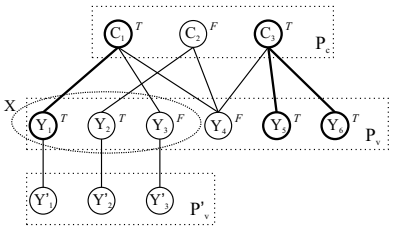


Figure 4. Reduction from X-MAXIMAL MODEL.

We define a game $\mathcal{G}(\phi)$ with an objective function $obj(\mathcal{G})$ such that (i) there is a one-to-one correspondence between Nash equilibria of $\mathcal{G}(\phi)$ and satisfying truth-value assignments of ϕ, and (ii) each Nash equilibrium $\mathbf{x}$ that optimizes $obj(\mathcal{G})$ corresponds to a X-MAXIMAL MODEL. The players of $\mathcal{G}(\phi)$ are partitioned in three sets: P_v, corresponding to all variables of ϕ; P'_v, corresponding to the X variables of ϕ; and P_c, corresponding to the clauses of ϕ. All players may choose an action in $\{t, f, u\}$ (read: true, false, undefined). Figure 4 shows the dependency graph and a Nash equilibrium of $\mathcal{G}(\phi)$ for a formula $\phi = (Y_1 \vee Y_3 \vee Y_4) \wedge (\neg Y_2 \vee Y_4) \wedge (Y_4 \vee Y_5 \vee Y_6)$. The utility functions are designed as follows: The strategies of "variable" players in P_v encode truth-value assignments for the corresponding variables in ϕ. At Nash equilibria, these assignments satisfy ϕ. Similarly, the strategies of "clause" players encode the evaluation of the corresponding clauses of ϕ, given the choices of their variable-players neighbors. Finally, each player x' in P'_v is a copy of a player x in P_v corresponding to an X-variable occurring in ϕ. Player x' plays the same action as x, but gets her maximum payoff 1 if she can play t, while all other variable players have no incentive to change their actions as long as their profile encodes a satisfying assignment. The objective function is $[\max f^{+}_{P'_v}]$, so that any optimal Nash equilibrium corresponds to a satisfying assignment for ϕ with a maximum number of X variables made true, which is clearly an X-MAXIMAL MODEL of ϕ.

Finally, in order to prove that the general (non-smooth) case is $\mathrm{FNP}//\mathrm{OptP}$-hard, we may show a reduction from the problem X-LEXICOGRAPHICALLY MAXIMAL MODEL. In this case, we use the same game as above, but we have to define a more complex and non-smooth objective function. □

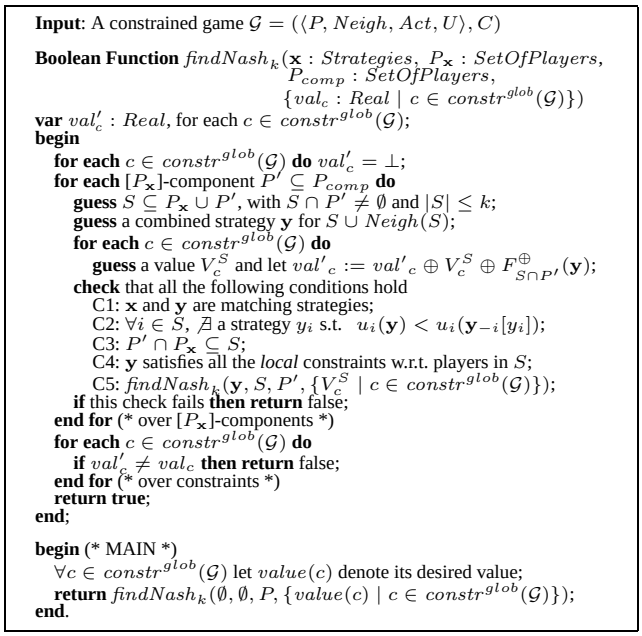
5 Tractable Cases

The hardness proof of Theorem 5 relies on the use of an unbounded number of global aggregation constraints. Thus, it is natural to investigate what happens if the number of these constraints is fixed, or bounded by some constant.

Let $h > 0$ be a fixed constant. We say that a game $\mathcal{G}$ is *h-weakly constrained* if $|constr^{glob}(\mathcal{G})| \leq h$ and all constraints are smooth. A class C of games is *weakly constrained* if there is a constant h such that all graphs in C are h-weakly constrained — notice that there is no bound on the number of local constraints.

We next show that, for classes of weakly constrained games having acyclic or bounded-treewidth dependency graphs (and hence also for acyclic games), the problem of deciding the existence of Nash equilibria is feasible in polynomial time, and it is also parallelizable.

Before explaining the details of the algorithm, we give some preliminary definitions and notations. Let $\mathcal{G}$ be a game, and let S be a subset of nodes of $\mathrm{G}(\mathcal{G})$. Note that nodes of $\mathrm{G}(\mathcal{G})$ correspond one-to-one to players of $\mathcal{G}$, and thus we will use the two terms interchangeably, hereafter. A non-empty set P' of players is [S]-*connected* (in $\mathrm{G}(\mathcal{G})$) if, for each pair $h, h' \in P'$, there exists a



Input: A constrained game $\mathcal{G} = (\langle P, Neigh, Act, U\rangle, C)$

Boolean Function $findNash_k(\mathbf{x} : Strategies,\ P_{\mathbf{x}} : SetOfPlayers,\ P_{comp} : SetOfPlayers,\ \{val_c : Real \mid c \in constr^{glob}(\mathcal{G})\})$
var $val'_c : Real$, for each $c \in constr^{glob}(\mathcal{G})$;
begin
  for each $c \in constr^{glob}(\mathcal{G})$ **do** $val'_c = \bot$;
  for each $[P_{\mathbf{x}}]$-component $P' \subseteq P_{comp}$ **do**
    guess $S \subseteq P_{\mathbf{x}} \cup P'$, with $S \cap P' \neq \emptyset$ and $|S| \leq k$;
    guess a combined strategy $\mathbf{y}$ for $S \cup Neigh(S)$;
    for each $c \in constr^{glob}(\mathcal{G})$ **do**
      guess a value V_c^S and let $val'_c := val'_c \oplus V_c^S \oplus F^{\oplus}_{S\cap P'}(\mathbf{y})$;
    check that all the following conditions hold
      C1: $\mathbf{x}$ and $\mathbf{y}$ are matching strategies;
      C2: $\forall i \in S$, $\nexists$ a strategy y_i s.t. $u_i(\mathbf{y}) < u_i(\mathbf{y}_{-i}[y_i])$;
      C3: $P' \cap P_{\mathbf{x}} \subseteq S$;
      C4: $\mathbf{y}$ satisfies all the *local* constraints w.r.t. players in S;
      C5: $findNash_k(\mathbf{y}, S, P', \{V_c^S \mid c \in constr^{glob}(\mathcal{G})\})$;
    if this check fails **then return** false;
  end for (* over $[P_{\mathbf{x}}]$-components *)
  for each $c \in constr^{glob}(\mathcal{G})$ **do**
    if $val'_c \neq val_c$ **then return** false;
  end for (* over constraints *)
  return true;
end;

begin (* MAIN *)
  $\forall c \in constr^{glob}(\mathcal{G})$ let $value(c)$ denote its desired value;
  return $findNash_k(\emptyset, \emptyset, P, \{value(c) \mid c \in constr^{glob}(\mathcal{G})\})$;
end.

Figure 5. Algorithm DecideNashExistence$_k$.

path $h = h_1, \ldots, h_n = h'$ in $\mathrm{G}(\mathcal{G})$ such that no player h_i in the path belongs to S. The [S]-*components* of $\mathrm{G}(\mathcal{G})$ are the maximal [S]-*connected* sets of players in $\mathcal{G}$. Finally, let $\mathbf{x}_1$ and $\mathbf{x}_2$ be two combined strategies for two sets of players P_1 and P_2. We say that $\mathbf{x}_1$ and $\mathbf{x}_2$ are *matching strategies* if all the players in $P_1 \cap P_2$ play the same action in x_1 and x_2.

Let k and h be fixed constants, where k will be the bound for the treewidth, and h the bound for the number of global constraints. Figure 5 shows the nondeterministic algorithm **DecideNashExistence**$_k$ that decides whether there exists a pure Nash equilibrium for a constrained game $\mathcal{G}$ that satisfies all the aggregation constraints (and may be also used for computing such an equilibrium), if the treewidth of the game is at most k. Otherwise, i.e., if the treewidth is greater than k or there is no equilibrium satisfying the constraints, it returns false.

For simplicity, we assume in this algorithm that every constraint c is an equality constraint, i.e., $op(c)$ is $=$. However, it is very easy to modify the algorithm in order to remove this simplifying assumption.

Roughly, **DecideNashExistence**$_k$ is based on a recursive Boolean function $findNash_k$ that at the generic step, receives as its inputs a combined strategy $\mathbf{x}$ for a set of players $P_{\mathbf{x}}$, a set of players P_{comp} and a value val_c for each constraint c of the game. Intuitively, we want to extend the strategy $\mathbf{x}$ to all players in P_{comp}, in such a way that the evaluation of each constraint c in this extension equals the assigned value val_c.

The key point for tractability here is the fact that we just look at local equilibrium properties – see Condition C2 above. This is only feasible if we can arrange players in a tree-like structure such that, at each step, we need a limited amount of information about players choices. In our case, we exploit a possible tree-decomposition of the dependency graph of the game, having width k at most. Note that such a decomposition is not fixed a-priori. Rather it is implicitly computed by the algorithm when we guess, at each step, the new set of players S we are going to deal with.

The proof of soundness and correctedness of this algorithm is quite involved, and will be omitted in this extended abstract, for space reasons. Moreover, we claim that, if the constraints are smooth and the number of global constraints is bounded by h, this algorithm may be implemented on a logspace alternating Turing machine (ATM) M with a polynomially-bounded proof-tree (see [18], for formal definition and further information on these issues). Thus, by exploiting a well known characterization of LOGCFL in terms of logspace alternating Turing machines with a polynomially bounded proof tree [18], we get the following tractability result, evidencing also that the algorithm is highly-parallelizable (recall that LOGCFL $\subseteq$ NC$_2$).

Theorem 7 *For acyclic and bounded treewidth weakly constrained games, deciding whether there exists a pure Nash equilibrium is* LOGCFL*-complete.*

Finally, let us observe that the algorithm in Figure 5 can be used for computing a Nash equilibrium as well, by exploiting the information in the proof tree of the ATM M, as described in [7]. In fact, a Nash equilibrium can be obtained by exploiting the strategies encoded in the configurations of M.

Let us now conclude with the study of tractable classes of optimization games. In principle, **DecideNashExistence**$_k$ can be used for computing the Nash equilibrium optimizing a function f, after several instantiations with the constraint $[f = h']$, for all possible values of h' within the range of f. Actually, in the case no constraints are imposed on the game, we can also define a more direct and efficient approach, which does not need a bound on the range of the optimization function.

Theorem 8 *Computing a Nash equilibrium which optimizes a given (possibly non-smooth) aggregation objective function is feasible in polynomial time for bounded treewidth games.*

REFERENCES

[1] Z. Chen and S. Toda. The complexity of selecting maximal solutions. *Information and Computation*, 119(2), pp. 231-239, 1995.
[2] V. Conitzer and T. Sandholm. Complexity Results about Nash Equilibria. *in Proc. of IJCAI'03*, pp. 765–771, 2003.
[3] A. Fabrikant, C.H. Papadimitriou, and K.Talwar, The Complexity of Pure Nash Equilibria, *to appear in Proc. of STOC'04*.
[4] M.R. Garey and D.S.Johnson, *Computers and Intractability. A Guide to the Theory of* NP*-completeness*, Freeman and Comp., NY, USA, 1979.
[5] I. Gilboa and E. Zemel, Nash and correlated equilibria: Some complexity consideration, *Games and Economic Behaviour*, 1, pp. 80-93, 1989.
[6] G. Gottlob, G. Greco, and F. Scarcello. Pure Nash Equilibria: Hard and Easy Games. *In Proc. of TARK'03*, pp. 215-230, 2003.
[7] G. Gottlob, N. Leone, F. Scarcello, The complexity of acyclic conjunctive queries, *Journal of the ACM*, 48(3), pp. 431-498, 2001.
[8] M. Kearns, M.L. Littman and S. Singh, An Efficient Exact Algorithm for Singly Connected Graphical Games, *in Proc. of NIPS'01*, pp. 817-823, 2001.
[9] M. Kearns, M.L. Littman and S. Singh, Graphical Models for Game Theory, *in Proc. of UAI'01)*, pp. 253-260, 2001.
[10] M. Kearns and Y. Mansour, Efficient Nash Computation in Large Population Games with Bounded Influence, *in Proc. of UAI'02*, pp. 259-266, 2002.
[11] M. Kearns and L. Ortiz, Nash Propagation for Loopy Graphical Games, *in Proc. of NIPS'02*, pp. XX-XX, 2002.
[12] D. Koller and B. Milch, Multi-Agent Influence Diagrams for Representing and Solving Games *in Proc. of IJCAI'01* pp. 1027-1034, 2001.
[13] E. Koutsoupias and C.H. Papadimitriou, Worst Case Equilibria, *In Proc. of STACS'99*, pp. 404-413, 1999.
[14] J.F. Nash, Non-cooperative Games, *Annals of Mathematics*, 54(2), pp. 286-295, 1951.
[15] G. Owen, *Game Theory*, Academic Press, New York, 1982.
[16] C.H. Papadimitriou, Algorithms, Games, and the Internet, *In Proc. of ICALP'01*, pp. 1-3, 2001.
[17] N. Robertson and P.D.Seymour, Graph Minors II. Algorithmic aspects of tree width, *Journal of Algorithms*, 7, pp. 309-322, 1986.
[18] W.L. Ruzzo. Tree-size bounded alternation. *Journal of Cumputer and System Sciences*, 21, pp. 218-235, 1980.
[19] D. Vickrey and D. Koller. Multi-Agent Algorithms for Solving Graphical Games. *In Proc. of AAAI/IAAI'02*, pp.345-351, 2002.