Computing Constrained Approximate Equilibria in Polymatrix Games*

Argyrios Deligkas[1], John Fearnley[2], and Rahul Savani[2]

[1] Technion, Israel
[2] University of Liverpool, UK

Abstract. This paper studies *constrained* approximate Nash equilibria in polymatrix games. We show that is NP-hard to decide if a polymatrix game has a constrained approximate equilibrium for 9 natural constraints and *any* non-trivial ϵ. We then provide a QPTAS for polymatrix games with bounded treewidth and logarithmically many actions per player that finds constrained approximate equilibria for a wide family of constraints.

1 Introduction

In this paper we study *polymatrix games*, which provide a succinct representation of a many-player game. In these games, each player is a vertex in a graph, and each edge of the graph is a bimatrix game. Every player chooses a single strategy and plays it in *all* of the bimatrix games that he is involved in, and his payoff is the *sum* of the payoffs that he obtains from each individual edge game.

A fundamental problem in algorithmic game theory is to design efficient algorithms for computing *Nash equilibria*. Unfortunately, even in bimatrix games, this is PPAD-complete [12,17], which probably rules out efficient algorithms. Thus, attention has shifted to *approximate* equilibria. There are two natural notions of an approximate equilibrium. An *ϵ-Nash equilibrium* (ϵ-NE) requires that each player has an expected payoff that is within ϵ of their best response payoff. An *ϵ-well-supported Nash equilibrium* (ϵ-WSNE) requires that all players only play pure strategies whose payoff is within ϵ of the best response payoff.

Constrained approximate equilibria. Sometimes, it is not enough to find an approximate NE, but instead we want to find one that satisfies certain constraints, such as having high social welfare. For bimatrix games, the algorithm of Lipton, Markakis, and Mehta (henceforth LMM) can be adapted to provide a quasi-polynomial time approximation scheme (QPTAS) for this task [31]: we can find in $m^{O(\frac{\ln m}{\epsilon^2})}$ time an ϵ-NE whose social welfare is at least as good as any ϵ'-NE where $\epsilon' < \epsilon$.

A sequence of papers [1,11,21,29] has shown that polynomial time algorithms for finding ϵ-NEs with good social welfare are unlikely to exist, subject to various hardness assumptions such as ETH. These hardness results carry over to a range of other properties, and apply for all $\epsilon < \frac{1}{8}$ [21].

* This work was supported by ISF grant 2021296 and EPSRC grant EP/P020909/1.

Our contribution. We show that deciding whether there is an ϵ-NE with good social welfare in a polymatrix game is NP-complete for all $\epsilon \in [0, 1]$. We then study a variety of further constraints (Table 1). For each one, we show that deciding whether there is an ϵ-WSNE that satisfies the constraint is NP-complete for all $\epsilon \in (0, 1)$. Our results hold even when the game is a planar bipartite graph with degree at most 3, and each player has at most 7 actions.

To put these results into context, let us contrast them with the known lower bounds for bimatrix games, which also apply directly to polymatrix games. Those results [1,11,21,29] imply that one cannot hope to find an algorithm that is better than a QPTAS for polymatrix games when $\epsilon < \frac{1}{8}$. In comparison, our results show a stronger, NP-hardness, result, apply to all ϵ in the range $(0, 1)$, and hold even when the players have constantly many actions.

We then study the problem of computing constrained approximate equilibria in polymatrix games with restricted graphs. Although our hardness results apply to a broad class of graphs, bounded treewidth graphs do not fall within their scope. A recent result of Ortiz and Irfan [33, 34] provides a QPTAS for finding ϵ-NEs in polymatrix games with bounded treewidth where every player has at most logarithmically many actions. We devise a dynamic programming algorithm for finding approximate equilibria in polymatrix games with bounded treewidth. Much like the algorithm in [33], we discretize both the strategy and payoff spaces, and obtain a complexity result that matches theirs. However, our algorithm works directly on the game, avoiding the reduction to a CSP used in their result.

The main benefit is that this algorithm can be adapted to provide a QPTAS for constrained approximate Nash equilibria. We introduce *one variable decomposable (OVD)* constraints, which are a broad class of optimization constraints, covering many of the problems listed in Table 1. We show that our algorithm can be adapted to find good approximate equilibria relative to an OVD constraint. Initially, we do this for the restricted class of *k-uniform* strategies: we can find a k-uniform 1.5ϵ-NE whose value is better than any k-uniform $\epsilon/4$-NE. Note that this is similar to the guarantee given by the LMM technique in bimatrix games. We extend this beyond the class of k-uniform strategies for constraints that are defined by a linear combination of the payoffs, such as social welfare. In this case, we find a 1.5ϵ-NE whose value is within $O(\epsilon)$ of *any* $\epsilon/8$-NE.

Related work. Barman, Ligett and Piliouras [4] have provided a randomised QPTAS for polymatrix games played on trees. Their algorithm is also a dynamic programming algorithm that discretizes the strategy space using the notion of a k-uniform strategy. Their algorithm is a QPTAS for general polymatrix games on trees and when the number of pure strategies for every player is bounded by a constant they get an expected polynomial-time algorithm (EPTAS).

The work of Ortiz and Irfan [33] applies to a much wider class of games that they call graphical multi-hypermatrix games. They provide a QPTAS for the case where the interaction hypergraph has bounded hypertreewidth. This class includes polymatrix games that have bounded treewdith and logarithmically many actions per player. For the special cases of tree polymatrix games and

tree graphical games they go further and provide explicit dynamic programming algorithms that work directly on the game, and avoid the need to solve a CSP.

Gilboa and Zemel [27] showed that it is NP-complete to decide whether there exist Nash equilibria in bimatrix games with certain properties, such as high social welfare. Conitzer and Sandholm [13] extended the list of NP-complete problems of [27]. Bilo and Mavronicolas [5] extended these results to win-lose bimatrix games. Bonifaci, Di Iorio and Laura [9] showed that it is NP-complete to decide whether a win-lose bimatrix game possesses a Nash equilibrium where every player plays a uniform strategy over their support. Recently, Garg et al. [26] and Bilo and Mavronicolas [6, 7] extended these results to many-player games and provided analogous ETR-completeness results.

Elkind, Goldberg, and Goldberg have given a polynomial time algorithm for finding exact Nash equilibria in two-action path graphical games [23]. They have also extended this to find good constrained exact equilibria in certain two-action tree graphical games [24]. Greco and Scarcello provide further hardness results for constrained equilibria in graphical games [28].

Computing approximate equilibria in bimatrix games has been well studied [10,14,18,19,25,30,36], but there has been less work for polymatrix games [3, 20,22]. Rubinstein [35] has shown that there is a small constant ϵ such that finding an ϵ-NE of a polymatrix game is PPAD-complete. For constrained ϵ-NE, the only positive results were for bimatrix games and gave algorithms for finding ϵ-NE with constraints on payoffs [15, 16].

2 Preliminaries

We start by fixing some notation. We use $[k]$ to denote the set of integers $\{1, 2, \ldots, k\}$, and when a universe $[k]$ is clear, we will use $\bar{S} = \{i \in [k] \ : \ i \notin S\}$ to denote the complement of $S \subseteq [k]$. For a k-dimensional vector x, we use x_{-S} to denote the elements of x with indices $\bar{S}$, and in the case where $S = \{i\}$ has only one element, we simply write x_{-i} for x_{-S}.

An n-player polymatrix game is defined by an undirected graph $G = (V, E)$ with n vertices, where each vertex is a player. The edges of the graph specify which players interact with each other. For each $i \in [n]$, we use $N(i) = \{j \ : \ (i, j) \in E\}$ to denote the neighbors of player i. Each edge $(i, j) \in E$ specifies a bimatrix game to be played between players i and j. Each player $i \in [n]$ has a fixed number of pure strategies m, so the bimatrix game on edge $(i, j) \in E$ is specified by an $m \times m$ matrix A_{ij}, which gives the payoffs for player i, and an $m \times m$ matrix A_{ji}, which gives the payoffs for player j. We allow the individual payoffs in each matrix to be an arbitrary rational number. We make the standard normalisation assumption that the maximum payoff each player can obtain under any strategy profile is 1 and the minimum is zero, unless specified otherwise. This can be achieved for example by using the procedure described in [22]. A *subgame* of a polymatrix game is obtained by ignoring edges that are not contained within a given subgraph of the game's interaction graph G.

A *mixed strategy* for player i is a probability distribution over player i's pure strategies. A *strategy profile* specifies a mixed strategy for every player. Given a strategy profile $\mathbf{s} = (s_1, \ldots, s_n)$, the pure strategy payoffs, or the payoff vector,

of player i under $\mathbf{s}$, where only $\mathbf{s}_{-i}$ is relevant, is the sum of the pure strategy payoffs that he obtains in each of the bimatrix games that he plays. Formally, we define: $\mathbf{p}_i(\mathbf{s}) := \sum_{j \in N(i)} A_{ij} s_j$. The *expected* payoff of player i under the strategy profile $\mathbf{s}$ is defined as $s_i \cdot \mathbf{p}_i(\mathbf{s})$. The *regret* of player i under $\mathbf{s}$ the is difference between i's best response payoff against $\mathbf{s}_{-i}$ and between i's payoff under $\mathbf{s}$. If a strategy has regret ϵ, we say that the strategy is an ϵ-best response. A strategy profile $\mathbf{s}$ is an ϵ-Nash equilibrium, or ϵ-NE, if no player can increase his utility more than ϵ by unilaterally switching from $\mathbf{s}$, i.e., if the regret of every player is at most ϵ. Formally, $\mathbf{s}$ is an ϵ-NE if for every player $i \in [n]$ it holds that $s_i \cdot \mathbf{p}_i(\mathbf{s}) \geq \max \mathbf{p}_i(\mathbf{s}) - \epsilon$. A strategy profile $\mathbf{s}$ is an ϵ-well-supported Nash equilibrium, or ϵ-WSNE, if if the regret of every pure strategy played with positive probability is at most ϵ. Formally, $\mathbf{s}$ is an ϵ-WSNE if for every player $i \in [n]$ it holds that for all $j \in \text{supp}(s_i) = \{k \in [m] \mid (s_i)_k > 0\}$ we have $(\mathbf{p}_i(\mathbf{s}))_j \geq \max \mathbf{p}_i(\mathbf{s}) - \epsilon$.

3 Decision problems for approximate equilibria

In this section, we show NP-completeness for nine decision problems related to constrained approximate Nash equilibria in polymatrix games. Table 1 contains the list of the problems that we study[3]. For Problem 1, we show hardness for all $\epsilon \in [0, 1]$. For the remaining problems, we show hardness for all $\epsilon \in (0, 1)$, i.e., for all approximate equilibria except exact equilibria ($\epsilon = 0$), and trivial approximations ($\epsilon = 1$). All of these problems are contained in NP because a "Yes" instance can be witnessed by a suitable approximate equilibrium (or two in the case of Problem 5). The starting point for all of our hardness results is the NP-complete problem Monotone 1-in-3 SAT.

Definition 1 (Monotone 1-in-3 SAT). *Given a monotone boolean CNF formula ϕ with exactly 3 distinct variables per clause, decide if there exists a satisfying assignment in which exactly one variable in each clause is true. We call such an assignment a 1-in-3 satisfying assignment.*

Every formula ϕ, with variables $V = \{x_1, \ldots, x_n\}$ and clauses $C = \{y_1, \ldots, y_m\}$, can be represented as a bipartite graph between V and C, with an edge between x_i and y_j if and only if x_i appears in clause y_j. We assume, without loss of generality, that this graph is connected. We say that ϕ is *planar* if the corresponding graph is planar. Recall that a graph is called *cubic* if the degree of every vertex is exactly three. We use the following result of Moore and Robson [32].

Theorem 2 (Section 3.1 [32]). *Monotone 1-in-3 SAT is NP-complete even when the formula corresponds to a planar cubic graph.*

From now on, we assume that ϕ is a monotone planar cubic formula. We say that ϕ is a "Yes" instance if ϕ admits a 1-in-3 satisfying assignment.

[3] Given probability distributions $\mathbf{x}$ and $\mathbf{x}'$, the *TV* distance between them is $\max_i\{|\mathbf{x}_i - \mathbf{x}'_i|\}$. The TV distance between strategy profiles $\mathbf{s} = (s_1, \ldots, s_n)$ and $\mathbf{s}' = (s'_1, \ldots, s'_n)$ is the maximum TV distance of s_i and s'_i over all i.

Problem description	**Problem definition**
Problem 1: Large total payoff $u \in (0, n]$	Is there an ϵ-NE $\mathbf{s}$ such that $\sum_{i\in[n]} \mathbf{p}_i(\mathbf{s}) \geq u$?
Problem 2: Small total payoff $u \in [0, n)$	Is there an ϵ-WSNE $\mathbf{s}$ such that $\sum_{i\in[n]} \mathbf{p}_i(\mathbf{s}) \leq u$?
Problem 3: Small payoff $u \in [0, 1)$	Is there an ϵ-WSNE $\mathbf{s}$ such that $\min_i \mathbf{p}_i(\mathbf{s}) \leq u$?
Problem 4: Restricted support $S \subset [n]$	Is there an ϵ-WSNE $\mathbf{s}$ with $\text{supp}(s_1) \subseteq S$?
Problem 5: Two ϵ-WSNE $d \in (0, 1]$ apart in Total Variation (TV) distance	Are there two ϵ-WSNE with TV distance $\geq d$?
Problem 6: Small largest probability $p \in (0, 1)$	Is there an ϵ-WSNE $\mathbf{s}$ with $\max_j s_1(j) \leq p$?
Problem 7: Large total support size $k \in [n \cdot m]$	Is there an ϵ-WSNE $\mathbf{s}$ such that $\sum_{i\in[n]} \lvert\text{supp}(s_i)\rvert \geq k$?
Problem 8: Large smallest support size $k \in [n]$	Is there an ϵ-WSNE $\mathbf{s}$ such that $\min_i \lvert\text{supp}(s_i)\rvert \geq k$?
Problem 9: Large support size $k \in [n]$	Is there an ϵ-WSNE $\mathbf{s}$ such that $\lvert\text{supp}(s_1)\rvert \geq k$?

Table 1: The decision problems that we consider. All problems take as input an n-player polymatrix game with m actions for each player and an $\epsilon \in [0, 1]$.

Large total payoff for ϵ-NEs. We show that Problem 1 is NP-complete for every $\epsilon \in [0, 1]$, even when the interaction graph for the polymatrix game is planar, bipartite, cubic, and each player has at most three pure strategies.

Construction. Given a formula ϕ, we construct a polymatrix game G with $m + n$ players as follows. For each variable x_i we create a player v_i and for each clause y_j we create a player c_j. We now use V to denote the set of variable players and C to denote the clause players. The interaction graph is the bipartite graph between V and C described above. Each edge game has the same structure. Every player in V has two pure strategies called True and False, while every player in C has three pure strategies that depend on the three variables in the clause. If clause y_j contains variables x_i, x_k, x_l, then player c_j has pure strategies i, k and l. The game played between v_i and c_j is shown on the left in Fig. 1. The bimatrix games for v_k and v_l are defined analogously.

Correctness. The constructed game is not normalised. We prove our result for all ϵ, and thus in the normalised game the result will hold for all $\epsilon \in [0, 1]$. We show that for every ϵ, there is an ϵ-NE with social welfare m if and only if ϕ is a "Yes" instance. We begin by showing that if there is a solution for ϕ, then there is an *exact* NE for G with social welfare m, and therefore there is also an ϵ-NE for all ϵ with social welfare m. We start with a simple observation about the maximum and minimum payoffs that players can obtain in G.

Lemma 3. *In G, the total payoff for every variable player is at most 0, and the total payoff for every clause player c_j is at most 1. Moreover, if c_j gets payoff 1, then c_j and the variable players connected to c_j play pure strategies.*

Player c_j \ Player v_i	True	False
i	0 1	-1 0
k	0 −1	0 0
l	0 −1	0 0

Player c_j \ Player v_i	True	False	Out
i	$\frac{1}{3}-\frac{\epsilon}{3}$ κ	0 $c\cdot\kappa$	$\frac{1}{3}$ 0
k	0 0	$\frac{1}{3}-\frac{\epsilon}{3}$ $c\cdot\kappa$	$\frac{1}{3}$ 0
l	0 0	$\frac{1}{3}-\frac{\epsilon}{3}$ $c\cdot\kappa$	$\frac{1}{3}$ 0
Out	0 $\frac{1}{3}$	0 $\frac{1}{3}$	$\frac{1}{3}$ $\frac{1}{3}$

Fig. 1: Left: The game between clause player c_j and variable player v_i for Problem 1. Right: The game between c_j and v_i for Problems 2–9.

Lemma 4. *If ϕ is a "Yes" instance, there is an NE for G with social welfare m.*

Lemma 5. *If there is a strategy profile for G with social welfare m, then ϕ is a "Yes" instance.*

Together, Lemma 4 and Lemma 5 show that for all possible values of ϵ, it is NP-complete to decide whether there exists an ϵ-NE for G with social welfare m. When we normalise the payoffs in $[0, 1]$, this holds for all $\epsilon \in [0, 1]$.

Theorem 6. *Problem 1 is NP-complete for all $\epsilon \in [0, 1]$, even for degree-3 bipartite planar polymatrix games in which each player has at most 3 pure strategies.*

Hardness of Problems 2–9. To show the hardness Problems 2–9, we modify the game constructed in the previous section. We use G' to denote the new polymatrix game. The interaction graph for G' is exactly the same as for the game G. The bimatrix games are extended by an extra pure strategy for each player, the strategy Out, and the payoffs are adapted. If variable x_i is in clause y_j, then the bimatrix game between clause player c_j and v_i is shown on the right in Fig. 1. To fix the constants, given $\epsilon \in (0, 1)$, we choose c to be in the range $(\max(1 - \frac{3\epsilon}{2}, 0), 1)$, and we set $\kappa = \frac{1-\epsilon}{1+2c}$. Observe that $0 < c < 1$, and that $\kappa + 2c \cdot \kappa = 1 - \epsilon$. Furthermore, since $c > 1 - \frac{3\epsilon}{2}$ we have $0 < \kappa < \frac{1}{3}$.

Lemma 7. *If ϕ is a "Yes" instance, then G' possesses an ϵ-WSNE such that no player uses strategy Out.*

Lemma 8. *If ϕ is a "No" instance, then G' possesses a unique ϵ-WSNE where every player plays Out.*

The combination of these two properties allows us to show that Problems 2–5 are NP-complete. For example, for Problem 4, we can ask whether there is an ϵ-WSNE of the game in which player one does not player Out.

Theorem 9. *Problems 2–5 are* NP*-complete for all* $\epsilon \in (0, 1)$*, even on degree-3 planar bipartite polymatrix games where each player has at most 4 pure strategies.*

Duplicating strategies. To show hardness for Problems 6–9, we slightly modify the game G' by duplicating every pure strategy except Out for all of the players. Since each player $c_j \in C$ has the pure strategies i, k, l and Out, we give player c_j pure strategies i', k' and l', which each have identical payoffs as the original strategies. Similarly for each player $v_i \in V$ we add the pure strategies True$'$ and False$'$. Let us denote the game with the duplicated strategies by $\widetilde{G}$. Then, if ϕ is a "Yes" instance, we can construct an ϵ-WSNE in which no player plays Out, where each player places at most 0.5 probability on each pure strategy, and where each player uses a support of size 2. These properties are sufficient to show that Problems 6–9 are NP-complete.

Theorem 10. *Problems 6–9 are* NP*-complete for all* $\epsilon \in (0, 1)$*, even on degree-3 planar bipartite polymatrix games where each player has at most 7 pure strategies.*

4 Constrained equilibria in bounded treewidth games

In this section, we show that some constrained equilibrium problems can be solved in quasi-polynomial time if the input game has bounded treewidth and at most logarithmically many actions per player. We first present a dynamic programming algorithm for finding approximate Nash equilibria in these games, and then show that it can be modified to find constrained equilibria.

Tree Decompositions. A tree decomposition of a graph $G = (V, E)$ is a pair $(\mathcal{X}, T)$, where $T = (I, F)$ is a tree and $\mathcal{X} = \{X_i | i \in I\}$ is a family of subsets of V such that (1) $\bigcup_{i \in I} X_i = V$, (2) for every edge $(u, v) \in E$ there exists an $i \in I$ such that $\{u, v\} \in X_i$, and (3) for all $i, j, k \in I$ if j is on the path from i to k in T, then $X_i \cap X_k \subseteq X_j$. The width of a tree decomposition $(\mathcal{X}, T)$ is $\max_i |X_i| - 1$. The treewidth of a graph is the minimum width over all possible tree decompositions of the graph. In general, computing the treewidth of a graph is NP-hard, but there are fixed parameter tractable algorithms for the problem. In particular Bodlaender [8] has given an algorithm that runs in $O(f(w) \cdot n)$ time, where w is the treewidth of the graph, and n is the number of nodes.

4.1 An algorithm to find approximate Nash equilibria

Let G be a polymatrix game and let $(\mathcal{X}, T)$ be a tree decomposition of G's interaction graph. We assume that an arbitrary node of T has been chosen as the root. Then, given some node v in T, we define $G(X_v)$ to be the subgame that is obtained when we only consider the players in the subtree of v. More formally, this means that we only include players i that are contained in some set X_u where u is in the subtree of v in the tree decomposition. Furthermore, we will use $\widetilde{G}(X_v)$ to denote the players in $G(X_v) \setminus X_v$. For every player $i \in X_v$, we will use $N_i(X_v)$ to denote the neighbours of i in $\widetilde{G}(X_v)$.

k-uniform strategies. A strategy s is said to be k-uniform if there exists a multi-set S of k pure strategies such that s plays uniformly over the pure strategies in S. These strategies naturally arise when we sample, with replacement, k

pure strategies from a distribution, and play the sampled strategies uniformly. The following is a theorem of [2].

Theorem 11. *Every n-player m-action game has a k-uniform ϵ-NE whenever* $k \geq 8 \cdot \frac{\ln m + \ln n - \ln \epsilon + \ln 8}{\epsilon^2}$.

Candidates and witnesses. For each node v in the tree decomposition, we compute a set of *witnesses*, where each witness corresponds to an ϵ-NE in $G(X_v)$. Our witnesses have two components: $\mathbf{s}$ provides a k-uniform strategy profile for the players in X_v, while $\mathbf{p}$ contains information about the payoff that the players in X_v obtain from the players in $\widetilde{G}(X_v)$. By summarising the information about the players in $\widetilde{G}(X_v)$, we are able to keep the number of witnesses small.

There is one extra complication, however, which is that the number of possible payoff vectors that can be stored in $\mathbf{p}$ depends on the number of different strategies for the players in $\widetilde{G}(X_v)$, which is exponential, and will cause our dynamic programming table to be too large. To resolve this, we *round* the entries of $\mathbf{p}$ to a suitably small set of rounded payoffs.

Formally, we first define $P = \{x \in [0,1] \ : \ x = \frac{\epsilon}{2n} \cdot k \text{ for some } k \in \mathbb{N}\}$, to be the set of rounded payoffs. Then, given a node v in the tree decomposition, we say that a tuple $(\mathbf{s}, \mathbf{p})$ is a *k-candidate* if:

- $\mathbf{s}$ is a set of strategies of size $|X_v|$, with one strategy for each player in X_v.
- Every strategy in $\mathbf{s}$ is k-uniform.
- $\mathbf{p}$ is a set of payoff vectors of size $|X_v|$. Each element $\mathbf{p}_i \in \mathbf{p}$ is of the form P^m, and assigns a rounded payoff to each pure strategy of player i.

The set of candidates gives the set of possible entries that can appear in our dynamic programming table. Every witness is a candidate, but not every candidate is a witness. The total number of k-candidates for each tree decomposition node v can be derived as follows. Each player has m^k possible k-uniform strategies, and so there are m^{kw} possibilities for $\mathbf{s}$. We have that $|P| = \frac{2n}{\epsilon}$, and that $\mathbf{p}$ contains $m \cdot w$ elements of P, so the total number of possibilities for $\mathbf{p}$ is $(2 \cdot \frac{n}{\epsilon})^{mw}$. Hence, the total number of candidates for v is $m^{kw} \cdot (2 \cdot \frac{n}{\epsilon})^{mw}$.

Next, we define what it means for a candidate to be a witness. We say that a k-candidate is an *ϵ, k, r-witness* if there exists a profile $\mathbf{s}'$ for $G(X_v)$ where

- $\mathbf{s}'$ agrees with $\mathbf{s}$ for the players in X_v.
- Every player in $\widetilde{G}(X_v)$ is *ϵ-happy*, which means that no player in $\widetilde{G}(X_v)$ can increase their payoff by more than ϵ by unilaterally deviating from $\mathbf{s}'$. Note that this does not apply to the players in X_v.
- Each payoff vector $p \in \mathbf{p}$ is within r of the payoff that player i obtains from the players in $\widetilde{G}(X_v)$. More accurately, for every pure strategy l of player i we have that: $|p_l - \sum_{j \in \widetilde{G}(X_v)} (A_{ij} \cdot \mathbf{s}'_j)_l| \leq r$. Note that $\mathbf{p}$ does not capture the payoff obtained from players in X_v, only those in the subtree of v.

The algorithm. Our algorithm computes a set of witnesses for each tree decomposition node by dynamic programming. At every leaf, the algorithm checks every possible candidate to check whether it is a witness. At internal nodes in

the tree decomposition, if a vertex is *forgotten*, that is, if it appears in a child of a node, but not in the node itself, then we use the set of witnesses computed for the child to check whether the forgotten node is ϵ-happy. If this is the case, then we create a corresponding witness for the parent node. The complication here is that, since we use rounded payoff vectors, this check may declare that a player is ϵ-happy erroneously due to rounding errors. So, during the analysis we must be careful to track the total amount of rounding error that can be introduced.

Once a set of witnesses has been computed for every tree decomposition node, a second phase is then used to find an ϵ-NE of the game. This phase picks an arbitrary witness in the root node, and then unrolls it by walking down the tree decomposition and finding the witnesses that were used to generate it. These witnesses collectively assign a k-uniform strategy profile to each player, and this strategy profile will be the ϵ-NE that we are looking for.

Lemma 12. *There is a dynamic programming algorithm that runs in time $O(n \cdot m^{2kw} \cdot (\frac{n}{\epsilon})^{2mw})$ that, for each tree decomposition node v, computes a set of candidates $C(v)$ such that: (1) Every candidate $(\mathbf{s}, \mathbf{p}) \in C(v)$ is an ϵ_v, k, r_v-witness for v for some $\epsilon_v \leq 1.5\epsilon$ and $r_v \leq \frac{\epsilon}{4}$. (2) If $\mathbf{s}$ is a k-uniform $\epsilon/4$-NE then $C(v)$ will contain a witness $(\mathbf{s}', \mathbf{p})$ such that $\mathbf{s}'$ agrees with $\mathbf{s}$ for all players in X_v.*

The running time bound arises from the total number of possible candidates for each tree decomposition node. The first property ensures that the algorithm always produces a 1.5ϵ-NE of the game, provided that the root node contains a witness. The second property ensures that the root node will contain a witness provided that game has a k-uniform $\epsilon/4$-NE. Theorem 11 tells us how large k needs to be for this to be the case. These facts yields the following theorem.

Theorem 13. *Let $\epsilon > 0$, G be a polymatrix game with treewidth w, and $k = 128 \cdot \frac{\ln m + \ln n - \ln \epsilon + \ln 8}{\epsilon^2}$. There is an algorithm that finds a 1.5ϵ-NE of G in in $O(n \cdot m^{2kw} + (\frac{n}{\epsilon})^{2mw})$ time.*

Note that if $m \leq \ln n$ (and in particular if m is constant), this is a QPTAS.

Corollary 14. *Let $\epsilon > 0$, and G be a polymatrix game with treewidth w, and $m \leq \ln n$. There is an algorithm that finds a 1.5ϵ-NE of G in $(\frac{n}{\epsilon})^{O(\frac{w \cdot \ln n}{\epsilon^2})}$ time.*

4.2 Constrained approximate Nash equilibria

One variable decomposable constraints. We now adapt the algorithm to find a certain class of constrained approximate Nash equilibria. As a motivating example, consider Problem 1, which asks us to find an approximate NE with high social welfare. Formally, this constraint assigns a single rational number (the social welfare) to each strategy profile, and asks us to maximize this number. This constraint also satisfies a *decomposability* property: if a game G consists of two subgames G_1 and G_2, and if there are no edges between these two subgames, then we can maximize social welfare in G by maximizing social welfare in G_1 and G_2 independently. We formalise this by defining a constraint to be *one variable decomposable (OVD)* if the following conditions hold.

– There is a polynomial-time computable function g such that maps every strategy profile in G to a rational number.
– Let $\mathbf{s}$ be a strategy for game G, and suppose that we want to add vertex v to G. Let s be a strategy choice for v, and $\mathbf{s}'$ be an extension of $\mathbf{s}$ that assigns s to v. There is a polynomial-time computable function add such that $g(\mathbf{s}') = \text{add}(G, v, s, g(\mathbf{s}))$.
– Let G_1 and G_2 be two subgames that partition G, and suppose that there are no edges between G_1 and G_2. Let $\mathbf{s}_1$ be a strategy profile in G_1 and $\mathbf{s}_2$ be a strategy profile in G_2. If $\mathbf{s}$ is the strategy profile for G that corresponds to merging $\mathbf{s}_1$ and $\mathbf{s}_2$, then there is a polynomial-time computable function merge such that $g(\mathbf{s}) = \text{merge}(G_1, G_2, g(\mathbf{s}_1), g(\mathbf{s}_2))$.

Intuitively, the second condition allows us to add a new vertex to a subgame, and the third condition allows us to merge two disconnected subgames. Moreover, observe that the functions add and merge depend only on the value that g assigns to strategies, and not the strategies themselves. This allows our algorithm to only store the value assigned by g, and forget the strategies themselves.

Examples of OVD constraints. Many of the problems in Table 1 are OVD constraints. Problems 1 and 2 refer to the total payoff of the strategy profile, and so g is defined to be the total payoff of all players, while the functions add and merge simply add the total payoff of the two strategy profiles. Problems 3 and 6 both deal with minimizing a quantity associated with a strategy profile, so for these problems the functions add and merge use the min function to minimize the relevant quantities. Likewise, Problems 7, 8, and 9 seek to maximize a quantity, and so the functions add and merge use the max function. In all cases, proving the required properties for the functions is straightforward.

Finding OVD k-uniform constrained equilibria. We now show that, for every OVD constraint, the algorithm presented in Section 4.1 can be modified to find a k-uniform 1.5ϵ-NE that also has a high value with respect to the constraint. More formally, we show that the value assigned by g to the 1.5ϵ-NE is greater than the value assigned to g to all k-uniform $\epsilon/4$-NE in the game.

Given an OVD constraint defined by g, add, and merge, we add an extra element to each candidate to track the variable from the constraint: each candidate has the form $(\mathbf{s}, \mathbf{p}, x)$, where $\mathbf{s}$ and $\mathbf{p}$ are as before, and x is a rational number. The definition of an ϵ, k, r, g-witness is extended by adding the condition:

– Recall that $\mathbf{s}'$ is a strategy profile for $G(X_v)$ whose existence is asserted by the witness. Let $\mathbf{s}''$ be the restriction of $\mathbf{s}'$ to $\widetilde{G}(X_v)$. We have $x = g(\mathbf{s}'')$.

We then modify the algorithm to account for this new element in the witness. At each stage we track the correct value for x. At the leaves, we use g to compute the correct value. At internal nodes, we use add and merge to compute the correct value using the values stored in the witnesses of the children.

If at any point two witnesses are created that agree on $\mathbf{s}$ and $\mathbf{p}$, but disagree on x, then we only keep the witness whose x value is higher. This ensures that we only keep witnesses corresponding to strategy profiles that maximize the

constraint. When we reach the root, we choose the strategy profile with maximal value for x to be unrolled in phase 2. The fact that we only keep one witness for each pair $\mathbf{s}$ and $\mathbf{p}$ means that the running time of the algorithm is unchanged.

Theorem 15. *For every $\epsilon > 0$ let $k = 128 \cdot \frac{\ln m + \ln n - \ln \epsilon + \ln 8}{\epsilon^2}$. If G is a polymatrix game with treewidth w, then there is an algorithm that runs in $O(n \cdot m^{2kw} + (\frac{n}{\epsilon})^{2mw})$ time and finds a k-uniform 1.5ϵ-NE $\mathbf{s}$ such that $g(\mathbf{s}) \geq g(\mathbf{s}')$ for every strategy profile $\mathbf{s}'$ that is an $\epsilon/4$-NE.*

Results for non-k-uniform strategies. The guarantee given by Theorem 15 is given relative to the best value achievable by a k-uniform $\epsilon/4$-NE. It is also interesting to ask whether we can drop the k-uniform constraint. In the following theorem, we show that if g is defined to be a linear function of the payoffs in the game, then a guarantee can be given relative to *every* $\epsilon/8$-NE of the game. Note that this covers Problems 1, 2, and 3.

Theorem 16. *Suppose that, for a given a OVD constraint, the function g is a linear function of the payoffs. Let $\mathbf{s}$ be the 1.5ϵ-NE found by our algorithm when For every $\epsilon/8$-NE $\mathbf{s}'$ we have that $g(\mathbf{s}) \geq g(\mathbf{s}') - O(\epsilon)$.*

References

1. P. Austrin, M. Braverman, and E. Chlamtac. Inapproximability of NP-complete variants of Nash equilibrium. *Theory of Computing*, 9:117–142, 2013.
2. Y. Babichenko, S. Barman, and R. Peretz. Simple approximate equilibria in large games. In *Proc. of EC*, pages 753–770, 2014.
3. S. Barman and K. Ligett. Finding any nontrivial coarse correlated equilibrium is hard. In *Proc. of EC*, pages 815–816, 2015.
4. S. Barman, K. Ligett, and G. Piliouras. Approximating Nash equilibria in tree polymatrix games. In *Proc. of SAGT*, pages 285–296, 2015.
5. V. Bilò and M. Mavronicolas. The complexity of decision problems about Nash equilibria in win-lose games. In *Proc. of SAGT*, pages 37–48, 2012.
6. V. Bilò and M. Mavronicolas. A catalog of $\exists\mathbb{R}$-complete decision problems about Nash equilibria in multi-player games. In *Proc. of STACS*, pages 17:1–17:13, 2016.
7. V. Bilò and M. Mavronicolas. $\exists\mathbb{R}$-complete decision problems about symmetric Nash equilibria in symmetric multi-player games. In *Proc. of STACS*, pages 13:1–13:14, 2017.
8. H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.
9. V. Bonifaci, U. D. Iorio, and L. Laura. The complexity of uniform Nash equilibria and related regular subgraph problems. *Theor. Comput. Sci.*, 401(1-3):144–152, 2008.
10. H. Bosse, J. Byrka, and E. Markakis. New algorithms for approximate Nash equilibria in bimatrix games. *Theor. Comput. Sci.*, 411(1):164–173, 2010.
11. M. Braverman, Y. Kun-Ko, and O. Weinstein. Approximating the best Nash equilibrium in $n^{o(\log n)}$-time breaks the exponential time hypothesis. In *Proc. of SODA*, pages 970–982, 2015.
12. X. Chen, X. Deng, and S.-H. Teng. Settling the complexity of computing two-player Nash equilibria. *Journal of the ACM*, 56(3):article 14, 57 pages, 2009.
13. V. Conitzer and T. Sandholm. New complexity results about Nash equilibria. *Games and Economic Behavior*, 63(2):621 – 641, 2008.

14. A. Czumaj, A. Deligkas, M. Fasoulakis, J. Fearnley, M. Jurdziński, and R. Savani. Distributed methods for computing approximate equilibria. In *Proc. of WINE*, pages 15–28, 2016.
15. A. Czumaj, M. Fasoulakis, and M. Jurdziński. Approximate Nash equilibria with near optimal social welfare. In *Proc. of IJCAI*, pages 504–510, 2015.
16. A. Czumaj, M. Fasoulakis, and M. Jurdziński. Approximate plutocratic and egalitarian Nash equilibria. In *Proc. of AAMAS*, pages 1409–1410, 2016.
17. C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009.
18. C. Daskalakis, A. Mehta, and C. H. Papadimitriou. Progress in approximate Nash equilibria. In *Proc. of EC*, pages 355–358, 2007.
19. C. Daskalakis, A. Mehta, and C. H. Papadimitriou. A note on approximate Nash equilibria. *Theor. Comput. Sci.*, 410(17):1581–1588, 2009.
20. A. Deligkas, J. Fearnley, T. P. Igwe, and R. Savani. An empirical study on computing equilibria in polymatrix games. In *Proc. of AAMAS*, pages 186–195, 2016.
21. A. Deligkas, J. Fearnley, and R. Savani. Inapproximability results for approximate Nash equilibria. In *Proc. of WINE*, pages 29–43, 2016.
22. A. Deligkas, J. Fearnley, R. Savani, and P. G. Spirakis. Computing approximate Nash equilibria in polymatrix games. *Algorithmica*, 77(2):487–514, 2017.
23. E. Elkind, L. A. Goldberg, and P. W. Goldberg. Nash equilibria in graphical games on trees revisited. In *Proc. of EC*, pages 100–109, 2006.
24. E. Elkind, L. A. Goldberg, and P. W. Goldberg. Computing good Nash equilibria in graphical games. In *Proc. of EC*, pages 162–171, 2007.
25. J. Fearnley, P. W. Goldberg, R. Savani, and T. B. Sørensen. Approximate well-supported Nash equilibria below two-thirds. In *Proc. of SAGT*, pages 108–119, 2012.
26. J. Garg, R. Mehta, V. V. Vazirani, and S. Yazdanbod. Etr-completeness for decision versions of multi-player (symmetric) Nash equilibria. In *Proc. of ICALP*, pages 554–566, 2015.
27. I. Gilboa and E. Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1(1):80 – 93, 1989.
28. G. Greco and F. Scarcello. On the complexity of constrained Nash equilibria in graphical games. *Theor. Comput. Sci.*, 410(38-40):3901–3924, 2009.
29. E. Hazan and R. Krauthgamer. How hard is it to approximate the best Nash equilibrium? *SIAM J. Comput.*, 40(1):79–91, 2011.
30. S. C. Kontogiannis and P. G. Spirakis. Well supported approximate equilibria in bimatrix games. *Algorithmica*, 57(4):653–667, 2010.
31. R. J. Lipton, E. Markakis, and A. Mehta. Playing large games using simple strategies. In *Proc. of EC*, pages 36–41, 2003.
32. C. Moore and J. M. Robson. Hard tiling problems with simple tiles. *Discrete & Computational Geometry*, 26(4):573–590, 2001.
33. L. E. Ortiz and M. T. Irfan. FPTAS for mixed-strategy Nash equilibria in tree graphical games and their generalizations. *CoRR*, abs/1602.05237, 2016.
34. L. E. Ortiz and M. T. Irfan. Tractable algorithms for approximate Nash equilibria in generalized graphical games with tree structure. In *Proc. of AAAI*, pages 635–641, 2017.
35. A. Rubinstein. Inapproximability of Nash equilibrium. In *Proc. of STOC*, pages 409–418, 2015.
36. H. Tsaknakis and P. G. Spirakis. An optimization approach for approximate Nash equilibria. *Internet Mathematics*, 5(4):365–382, 2008.