Finding Approximate Nash Equilibria of Bimatrix Games via Payoff Queries

John Fearnley and Rahul Savani

University of Liverpool

Abstract. We study the deterministic and randomized query complexity of finding approximate equilibria in a $k \times k$ bimatrix game. We show that the deterministic query complexity of finding an ϵ-Nash equilibrium when $\epsilon < \frac{1}{2}$ is $\Omega(k^2)$, even in zero-one constant-sum games. In combination with previous results [8], this provides a complete characterization of the deterministic query complexity of approximate Nash equilibria. We also study randomized querying algorithms. We give a randomized algorithm for finding a $(\frac{3-\sqrt{5}}{2} + \epsilon)$-Nash equilibrium using $O(\frac{k \cdot \log k}{\epsilon^2})$ payoff queries, which shows that the $\frac{1}{2}$ barrier for deterministic algorithms can be broken by randomization. For well-supported Nash equilibria (WSNE), we first give a randomized algorithm for finding an ϵ-WSNE of a zero-sum bimatrix game using $O(\frac{k \cdot \log k}{\epsilon^4})$ payoff queries, and we then use this to obtain a randomized algorithm for finding a $(\frac{2}{3} + \epsilon)$-WSNE in a general bimatrix game using $O(\frac{k \cdot \log k}{\epsilon^4})$ payoff queries. Finally, we initiate the study of lower bounds against randomized algorithms in the context of bimatrix games, by showing that randomized algorithms require $\Omega(k^2)$ payoff queries in order to find an ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$, even in zero-one constant-sum games. In particular, this rules out query-efficient randomized algorithms for finding exact Nash equilibria.

1 Introduction

The Nash equilibrium is the central solution concept in game theory, which makes algorithms for finding Nash equilibria an important topic. A recent strand of work [2, 3, 8, 11, 12] has studied this problem from the perspective of *payoff query complexity*.

The payoff query model is motivated by practical applications of game theory. In many practical applications it is often the case that we know that there is a game to be solved, but we do not know what the payoffs are. In order to discover the payoffs, we would have to play the game, and hence discover the payoffs through experimentation. This may be quite costly, so it is natural to ask whether we can find a Nash equilibrium of a game while minimising the number of experiments that we must perform.

Payoff queries model this situation. In the payoff query model we are told the structure of the game, ie. the strategy space, but we are not told the payoffs. We are permitted to make payoff queries, where we propose a pure strategy profile, and we are told the payoff to each player under that strategy profile. Our task is to compute an equilibrium of the game while minimising the number of payoff queries that we make.

An example of a practical application for payoff queries is *empirical game theoretic analysis*. Wellman's survey paper gives an overview of this [17], and the paper of Fearnley et al. gives a detailed description of how payoff query complexity is relevant [8].

This paper studies the payoff query complexity of finding equilibria in bimatrix games. Previous work has shown that the deterministic query complexity of finding an *exact* Nash equilibrium in a $k \times k$ bimatrix game is k^2, even for zero-one constant-sum games [8]. In other words, one cannot hope to find an exact Nash equilibrium without querying all pure strategy profiles. Since query-efficient algorithms for exact equilibria do not exist, this naturally raises the question: what is the payoff query complexity of finding *approximate* equilibria?

There are two competing notions of approximate equilibrium for bimatrix games. An exact Nash equilibrium requires that all players achieve their best-response payoff, and thus have no incentive to deviate. An *ϵ-Nash equilibrium* requires that each player receives a payoff that is within ϵ of their best-response payoff, and thus all players have only a small incentive to deviate. One problem with this definition is that an ϵ-Nash equilibrium permits a player to place a small amount of probability on a very bad strategy, and it is questionable whether a rational player would actually want to do this. An *ϵ-well supported Nash equilibrium* (ϵ-WSNE) rectifies this, by requiring that all players only place probability on strategies that are within ϵ of being a best-response.

Previous work by Fearnley, Gairing, Goldberg, and Savani [8] has shown that an algorithm of Daskalakis, Mehta and Papadimitriou [7] can be adapted to produce a deterministic algorithm that finds a $\frac{1}{2}$-Nash equilibrium using $2k-1$ payoff queries. The same paper also shows that, for all i in the range $2 \leq i < k$, the deterministic payoff query complexity of finding a $(1 - \frac{1}{i})$-Nash equilibrium is at least $k-i+1$. Note, in particular, that this implies that for every constant ϵ in the range $\frac{1}{2} \leq \epsilon < 1$, the deterministic query complexity of finding an ϵ-Nash equilibrium is $\Theta(k)$.

However, relatively little is known about the more interesting case of $\epsilon < \frac{1}{2}$. A lower bound of $\Omega(k \cdot \log k)$ has been shown against deterministic algorithms that find a $O(\frac{1}{\log k})$-Nash equilibrium [8]. For the special case of zero-sum games, Goldberg and Roth have shown that an ϵ-Nash equilibrium of a zero-sum bimatrix game can be found by a *randomized* algorithm that uses $O(\frac{k \cdot \log k}{\epsilon^2})$ many payoff queries [11].

Our contribution As we have mentioned, so far relatively little is known for the deterministic query complexity of finding an ϵ-Nash equilibrium when $\epsilon < \frac{1}{2}$. We address this with a lower bound: in Section 3 we show that, for every $\epsilon > 0$, the deterministic payoff query complexity of finding a $(\frac{1}{2} - \epsilon)$-Nash equilibrium in a $k \times k$ bimatrix game is $\Omega(k^2)$. Our lower bound holds even for zero-one constant-sum games. When combined with previous results, this provides a complete characterization of the deterministic query complexity of ϵ-Nash equilibria when ϵ is constant: it is $\Theta(k)$ for $\epsilon \geq \frac{1}{2}$, and $\Theta(k^2)$ for $\epsilon < \frac{1}{2}$.

Since our lower bound rules out deterministic query-efficient algorithms for finding $(\frac{1}{2} - \epsilon)$-Nash equilibria, it is natural to ask whether this threshold can be broken through the use of randomization. In Section 6 we give a positive answer to this question. Our approach is to take an algorithm of Bosse, Byrka, and Markakis [4], and to apply the randomized algorithm of Goldberg and Roth [11] in order to solve the zero-sum game used by the BBM algorithm. We show that this produces a randomized algorithm that uses $O(\frac{k \cdot \log k}{\epsilon^2})$ payoff queries and finds a $(\frac{3-\sqrt{5}}{2}+\epsilon)$-Nash equilibrium in a $k \times k$ bimatrix game. Since $\frac{3-\sqrt{5}}{2} \approx 0.382$, this shows that randomization can be used to defeat the barrier at 0.5 that exists for deterministic algorithms.

In Section 7, we turn our attention to approximate well-supported Nash equilibria, which is a topic that has not previously been studied from the payoff query perspective in the context of bimatrix games. We adapt the algorithm of Kontogiannis and Spirakis for finding a $\frac{2}{3}$-WSNE in a bimatrix game [14], and in doing so we obtain a randomized algorithm for finding a $(\frac{2}{3} + \epsilon)$-WSNE using $O(\frac{k \cdot \log k}{\epsilon^4})$ many payoff queries. Note that this almost matches the best known polynomial-time algorithm for finding approximate well-supported Nash equilibria: Fearnley, Goldberg, Savani, and Sørensen [9] have given a polynomial time algorithm that finds a $(\frac{2}{3} - 0.004735)$-WSNE.

As part of the proof of this result, we prove an interesting side result: there is a randomized algorithm that finds an ϵ-WSNE of a zero-sum game using $O(\frac{k \cdot \log k}{\epsilon^4})$ payoff queries. This complements the result of Goldberg and Roth for finding ϵ-Nash equilibria in zero-sum games, and potentially could find applications elsewhere.

Finally, we initiate the study of lower bounds against randomized algorithms in the context of bimatrix games. In Section 4 we show that the randomized query complexity of finding an ϵ-Nash equilibrium for $\epsilon < \frac{1}{4k}$ in a $k \times k$ bimatrix game is $\Omega(k^2)$. Again, our lower bound holds even for zero-one constant-sum games. Note, in particular, that our lower bound rules out the existence of query-efficient randomized algorithms for finding exact equilibria. This improves over earlier results, which only considered deterministic algorithms for finding exact Nash equilibria [8].

Related work Apart from the related work on bimatrix games that we have already mentioned, there have been a number of results on payoff query complexity in the context of n-player strategic form games. Recently, Babichenko has shown that there is a constant (but small) ϵ for which the randomized query complexity of finding an ϵ-well-supported Nash equilibrium in an n-player strategic form is exponential in n [2].

The query complexity of finding approximate-correlated equilibria has been studied in a pair of papers [3, 12], where it was shown that a randomized algorithm can find approximate-correlated equilibria in polynomial time, but that non-random algorithms require exponentially many queries, and that finding an exact equilibrium requires exponentially many queries.

Finally, Goldberg and Roth have given a randomized algorithm that uses logarithmically many payoff queries in order to find approximate correlated equilibria in binary-action n-player strategic form games, and they also give a matching lower bound [11]. The same paper shows a linear lower bound for finding well-supported approximate correlated Nash equilibria.

Of course, there has been much previous work studying approximate Nash equilibria from the computational complexity point of view [4, 6, 7, 13, 16], some of which we have drawn upon for our query complexity results. So far, the best polynomial-time algorithm for finding ϵ-Nash equilibria was given by Tsaknakis and Spirakis, who showed that a 0.3393-Nash equilibrium can be found in polynomial-time [16]. For well-supported approximate Nash equilibria, the first result on the subject gave an algorithm for finding a $\frac{5}{6}$-WSNE in polynomial time [7], but this only holds if a certain unproved graph-theoretic conjecture is true. The best result until recently was by Kontogiannis and Spirakis, who gave an algorithm for finding $\frac{2}{3}$-WSNE in polynomial time. The current best known algorithm was given by Fearnley, Goldberg, Savani, and Sørensen [9] who, in a slight improvement over previous work, produced a polynomial-time algorithm for finding a $(\frac{2}{3} - 0.004735)$-WSNE.

There has also been a line of work studying the support size requirements for approximate Nash equilibria. It has been shown that every game has a $\frac{1}{2}$-Nash equilibrium with support size 2 [7], but logarithmic support sizes are both necessary [10] and sufficient [15] for ϵ-Nash equilibria with $\epsilon < \frac{1}{2}$. The threshold of $\frac{1}{2}$, of course, also appears in our work on the deterministic query complexity of approximate Nash equilibria.

A similar support-size threshold may exist for well-supported Nash equilibria at $\frac{2}{3}$: it has been shown that ϵ-WSNE with $\epsilon < \frac{2}{3}$ require super-constant support sizes [1], whereas an as yet unproved graph theoretic conjecture would imply that every game has a $\frac{2}{3}$-WSNE with support size 3 [7].

2 Preliminaries

Games and Strategies A $k \times k$ *bimatrix game* is a pair (R, C) of two $k \times k$ matrices: R gives payoffs for the *row player*, and C gives payoffs for the *column player*. We make the standard assumption that all payoffs lie in the range $[0, 1]$. For each $n \in \mathbb{N}$, we use $[n]$ to denote the set $\{1, 2, \dots, n\}$. Each player has k *pure strategies*. To play the game, both players simultaneously select a pure strategy: the row player selects a row $i \in [k]$, and the column player selects a column $j \in [k]$. The row player then receives payoff $R_{i,j}$, and the column player receives payoff $C_{i,j}$. We say that a bimatrix game (R, C) is a *zero-one* game, if all entries of R and C are either 0 or 1. We say that (R, C) is *constant-sum* if there is a constant c such that $R_{i,j} + C_{i,j} = c$, for all $i, j \in [k]$.

A *mixed strategy* is a probability distribution over $[k]$. We denote a mixed strategy for the row player as a row vector $\mathbf{x}$ of length k, such that $\mathbf{x}_i$ is the probability that the row player assigns to pure strategy i. A mixed strategy of the column player is a column vector $\mathbf{y}$ of length k, with the same interpretation.

Given a mixed strategy $\mathbf{x}$ for either player, the *support* of $\mathbf{x}$, denoted $\text{Supp}(\mathbf{x})$, is the set of pure strategies i with $x_i > 0$. If $\mathbf{x}$ and $\mathbf{y}$ are mixed strategies for the row and column player, respectively, then we call $(\mathbf{x}, \mathbf{y})$ a *mixed strategy profile.*

Solution Concepts Let $(\mathbf{x}, \mathbf{y})$ be a mixed strategy profile in a $k \times k$ bimatrix game (R, C). We say that a row $i \in [k]$ is a *best response* for the row player if $R_i \cdot \mathbf{y} = \max_{l \in [k]} R_l \cdot \mathbf{y}$. We say that a column $j \in [k]$ is a best response for the column player if $(\mathbf{x} \cdot C)_j = \max_{l \in [k]} (\mathbf{x} \cdot C)_l$. We define the *regret* suffered by the row player to be the difference between the payoff that the row player obtains under $(\mathbf{x}, \mathbf{y})$, and the payoff of a best response. More formally, the row player's regret is $\max_{i \in [k]} (R_i \cdot \mathbf{y}) - \mathbf{x} \cdot R \cdot \mathbf{y}$. Similarly, the column player's regret is defined to be $\max_{j \in [k]} ((\mathbf{x} \cdot C)_j) - \mathbf{x} \cdot C \cdot \mathbf{y}$. The mixed strategy profile $(\mathbf{x}, \mathbf{y})$ is a *mixed Nash equilibrium* if both players have regret 0 under $(\mathbf{x}, \mathbf{y})$. Note that this is equivalent to saying that every pure strategy in $\text{Supp}(\mathbf{x})$ is a best response against $\mathbf{y}$, and every pure strategy in $\text{Supp}(\mathbf{y})$ is a best response against $\mathbf{x}$.

The two approximate solution concepts that we study in this paper both weaken the requirements of a mixed Nash equilibrium, but in different ways. An *ϵ-Nash equilibrium* is an approximate solution concept that weakens the regret based definition of a mixed Nash equilibrium. For every $\epsilon \in [0, 1]$, a mixed strategy profile $(\mathbf{x}, \mathbf{y})$ is an ϵ-Nash equilibrium if both player suffer regret at most ϵ under $(\mathbf{x}, \mathbf{y})$.

An *ϵ-well supported Nash equilibrium* (ϵ-WSNE) weakens the best-response definition of a mixed Nash equilibrium. A strategy $i \in [k]$ is an ϵ-best response against $\mathbf{y}$ if:

$$R_i \cdot \mathbf{y} \geq \max_{l \in [k]} (R_l \cdot \mathbf{y}) - \epsilon.$$

Similarly, a strategy $j \in [k]$ is an ϵ-best response against $\mathbf{x}$ if:

$$(\mathbf{x} \cdot C)_j \geq \max_{l \in [k]} ((\mathbf{x} \cdot C)_l) - \epsilon.$$

For every $\epsilon \in [0, 1]$, a mixed strategy profile $(\mathbf{x}, \mathbf{y})$ is an ϵ-WSNE if every pure strategy in $\text{Supp}(\mathbf{x})$ is an ϵ-best response against $\mathbf{y}$, and every pure strategy in $\text{Supp}(\mathbf{y})$ is an ϵ-best response against $\mathbf{x}$. Note that every ϵ-WSNE is an ϵ-Nash equilibrium, but that the converse does not hold.

Payoff Queries In the payoff query model, an algorithm initially only knows the size of the game, but does not know the payoffs. That is, the algorithm knows that (R, C) is a $k \times k$ bimatrix game, but it does not know any of the payoffs in R or C. In order to discover the payoffs, the algorithm must make payoff queries. A payoff query is a pair (i, j) where $i \in [k]$ is a pure strategy of the row player, and $j \in [k]$ is a pure strategy of the column player. When an algorithm makes a payoff query (i, j), it receives a pair (a, b) where $a = R_{i,j}$ is the row player payoff and $b = C_{i,j}$ is the column player payoff.

3 An $\Omega(k^2)$ lower bound against deterministic algorithms for finding $(\frac{1}{2} - \epsilon)$-Nash equilibria

In this section, we show that for every $\epsilon > 0$, no deterministic algorithm can find a $(\frac{1}{2} - \epsilon)$-Nash equilibrium in a $k \times k$ bimatrix with strictly less than $\frac{\epsilon}{2} \cdot k^2$ queries. Thus, we show that the deterministic query complexity of finding a $(\frac{1}{2} - \epsilon)$-Nash equilibrium is $\Omega(k^2)$ for all $\epsilon > 0$. This lower bound holds even for zero-one constant-sum games.

Our proof will take the form of an algorithm interacting with an adversary, who will respond to the payoff queries that are made by the algorithm. The fundamental idea behind our lower bound is that the adversary will hide a column c such that $C_{i,c} = 1$ for all $i \in [k]$. Thus, column c always has payoff 1 for the column player, no matter what strategy the row player is using. Our goal is to show that the column player must place a significant amount of probability on c in order to be in a $(\frac{1}{2} - \epsilon)$-Nash equilibrium.

We now define our adversary strategy. For each payoff query (i, j), we respond according to the following rules:

- If column j has received strictly less than $\epsilon \cdot k$ payoff queries, then we respond with $(0, 1)$.
- If column j has received at least $\epsilon \cdot k$ payoff queries, then we respond with $(1, 0)$.

The result of the interaction between the algorithm and the adversary is a *partial bimatrix game*, which is a bimatrix game (R, C) where $R_{i,j}$ and $C_{i,j}$ are defined only for the pairs (i, j) that have received a payoff query.

The idea behind this strategy is that, if an algorithm makes strictly less than $\epsilon \cdot k$ payoff queries in a column j, then it cannot rule out the possibility that j is the hidden column c. Crucially, if an algorithm makes strictly less than $\frac{\epsilon}{2} \cdot k^2$ payoff queries overall, then there will be strictly more than $\frac{k}{2}$ columns that receive strictly less than $\epsilon \cdot k$ payoff queries, and thus, there will be strictly more than $\frac{k}{2}$ possible candidates for the hidden column.

So, suppose that an algorithm made strictly less than $\frac{\epsilon}{2} \cdot k^2$ payoff queries and then produced a mixed strategy profile $(\mathbf{x}, \mathbf{y})$ which is purported to be a $(\frac{1}{2} - \epsilon)$-Nash equilibrium. We will construct a game that is consistent with every query made by the algorithm, such that $(\mathbf{x}, \mathbf{y})$ is not a $(\frac{1}{2} - \epsilon)$-Nash equilibrium of this game.

Let (R', C') be the partial bimatrix game that corresponds to the queries made by the algorithm. We will extend (R', C') to a fully defined bimatrix game (R, C) as follows:

- We first place the hidden column. To do so, we pick a column c with $\mathbf{y}_c < \frac{2}{k}$, such that c has received strictly less than $\epsilon \cdot k$ payoff queries. Since strictly more than $\frac{k}{2}$ columns received strictly less than $\epsilon \cdot k$ payoff queries, such a column is guaranteed to exist. We set $R_{i,c} = 0$ and $C_{i,c} = 1$ for all $i \in [k]$.

– For each column $j \neq c$, we set all unqueried elements of j to be $(1, 0)$. More formally, for each column $j \neq c$ and each $i \in [k]$, we set

$$R_{i,j} = \begin{cases} R'_{i,j} & \text{if } R'_{i,j} \text{ is defined.} \\ 1 & \text{otherwise,} \end{cases}$$

and we set $C_{i,j} = 1 - R_{i,j}$.

Note that (R, C) is a zero-one constant-sum bimatrix game. The mixed strategy profile $(\mathbf{x}, \mathbf{y})$ and the bimatrix game (R, C) will be fixed for the rest of this section. Observe that, by construction, we have ensured that $\mathbf{y}$ plays c with low probability. We will exploit this to show that $(\mathbf{x}, \mathbf{y})$ is not a $(\frac{1}{2} - \epsilon)$-Nash equilibrium in (R, C).

In our first lemma, we give a lower bound on the row player's best response payoff against $\mathbf{y}$.

Lemma 1. *The row player's best response payoff against* $\mathbf{y}$ *in* (R, C) *is at least* $1 - \epsilon - \frac{2}{k}$.

Proof. Consider a strategy $\mathbf{x}'$ that mixes uniformly over all rows. Let j be a column such that $j \neq c$. By definition, we have that there are at most $\epsilon \cdot k$ rows i in j that satisfy $C_{i,j} = 1$, and there are at least $k - \epsilon \cdot k$ rows i in j that satisfy $C_{i,j} = 0$. Therefore, since $\mathbf{x}'$ plays each row with probability $\frac{1}{k}$, we have $(\mathbf{x}' \cdot C)_j \leq \epsilon$. Note that $\mathbf{y}$ plays c with probability at most $\frac{2}{k}$. Therefore, we have the following bound on the payoff of $\mathbf{y}$ against $\mathbf{x}'$:

$$\begin{aligned} \mathbf{x}' \cdot C \cdot \mathbf{y} &= \mathbf{y}_c \cdot (\mathbf{x}' \cdot C)_c + \sum_{j \neq c} \mathbf{y}_j \cdot (\mathbf{x}' \cdot C)_j \\ &\leq \frac{2}{k} + \sum_{j \neq c} \mathbf{y}_j \cdot \epsilon \\ &\leq \frac{2}{k} + \epsilon. \end{aligned}$$

Since the game is constant sum, this implies that the payoff of $\mathbf{x}'$ against $\mathbf{y}$ is at least $1 - \epsilon - \frac{2}{k}$. Therefore, the row player's best response against $\mathbf{y}$ must also have payoff at least $1 - \epsilon - \frac{2}{k}$. □

We now use the previous lemma to show that, provided that $\frac{2}{k} < \epsilon$, the mixed strategy profile $(\mathbf{x}, \mathbf{y})$ cannot be a $(\frac{1}{2} - \epsilon)$-Nash equilibrium.

Lemma 2. *If* $\frac{2}{k} < \epsilon$, *then* $(\mathbf{x}, \mathbf{y})$ *is not a* $(\frac{1}{2} - \epsilon)$*-Nash equilibrium in* (R, C).

Proof. Suppose for the sake of contradiction that $(\mathbf{x}, \mathbf{y})$ is a $(\frac{1}{2} - \epsilon)$-Nash equilibrium in (R, C). By Lemma 1, the row player's best response payoff against $\mathbf{y}$ is at least $1 - \epsilon - \frac{2}{k}$. Since, by assumption, the regret of the row player is at most $\frac{1}{2} - \epsilon$, the payoff of $\mathbf{x}$ against $\mathbf{y}$ must be at least:

$$\left(1 - \epsilon - \frac{2}{k}\right) - \left(\frac{1}{2} - \epsilon\right) = \frac{1}{2} - \frac{2}{k}.$$

Since the game is constant sum, this then implies that the payoff of $\mathbf{y}$ against $\mathbf{x}$ is at most $\frac{1}{2} + \frac{2}{k}$. But, since the payoff of column c is always 1, the payoff of $\mathbf{y}$ against $\mathbf{x}$ must be at least $1 - (\frac{1}{2} - \epsilon) = \frac{1}{2} + \epsilon$. Thus, we have a contradiction whenever $\frac{2}{k} < \epsilon$. □

Note that the precondition of Lemma 2 is equivalent to $k > \frac{2}{\epsilon}$. Therefore this lemma shows that, for every $\epsilon > 0$, there exists a k' such that, for all $k \geq k'$ no deterministic algorithm can find a $(\frac{1}{2} - \epsilon)$-Nash equilibrium in a $k \times k$ bimatrix game with strictly less than $\frac{\epsilon}{2} \cdot k$ payoff queries. Thus, we have shown the following theorem, which is the main result of this section.

Theorem 1. *The deterministic query complexity of finding a $(\frac{1}{2} - \epsilon)$-Nash equilibrium is $\Omega(k^2)$, even in zero-one constant-sum games.*

4 An $\Omega(k^2)$ lower bound against randomized algorithms for finding ϵ-Nash equilibria for $\epsilon < \frac{1}{4k}$

In this section, we show that no randomized algorithm can find a ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$ while making $o(k^2)$ playoff queries, even in zero-one constant-sum games.

At a high level, our technique is similar to the one we used for our deterministic lower bound in in Section 3: we will hide a column c such that $C_{i,c} = 1$ for all $i \in [k]$. In this case however, instead of using an adversary, we use a probability distribution over games. For each column $j \neq c$, we will select a single row r_j uniformly at random, and set $R_{r_j,j} = 1$ and $C_{r_j,j} = 0$. For each row $i \neq r_j$ we will set $R_{i,j} = 0$ and $C_{i,j} = 1$. Thus, in order to distinguish between column c, and a column $j \neq c$, the algorithm must find the row r_j, and as we will show, even randomized algorithms cannot do this in a query efficient manner.

Formally, we define $\mathcal{G}^k$ to be a random distribution over zero-one constant-sum $k \times k$ bimatrix games, defined in the following way. To draw a game from $\mathcal{G}^k$, we first choose a column $c \in [k]$ uniformly at random, which will be referred to as the hidden column. Furthermore, we choose $r_1, r_2, \ldots, r_k$ to be k uniformly and independently chosen rows (which may include repeats). Then we construct the game (R, C), where for all $i, j \in [k]$ we have:

$$C_{i,j} = \begin{cases} 1 & \text{if } j = c, \\ 0 & \text{if } j \neq c \text{ and } r_j = i, \\ 1 & \text{otherwise.} \end{cases}$$

We define $R_{i,j} = 1 - C_{i,j}$ for all $i, j \in [k]$.

We show that if $(\mathbf{x}, \mathbf{y})$ is a strategy profile in which $\mathbf{y}$ does not assign strictly more than $\frac{1}{2}$ probability to the hidden column c, then $(\mathbf{x}, \mathbf{y})$ is not an ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$. Since $\mathbf{y}$ can assign strictly more than $\frac{1}{2}$ probability to at most one column, this implies that any algorithm that produces an ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$ must find the hidden column. The following lemma shows that this property holds for all games in the support of the distribution $\mathcal{G}^k$.

Lemma 3. *Let (R, C) be a game drawn from $\mathcal{G}^k$, where c is the hidden column. If $(\mathbf{x}, \mathbf{y})$ is an ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$ in (R, C), then $\mathbf{y}_c > \frac{1}{2}$.*

Proof. Suppose, for the sake of contradiction, that $(\mathbf{x}, \mathbf{y})$ is an ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$ in (R, C) and that $\mathbf{y}_c \leq \frac{1}{2}$. We begin by arguing that the row player's best response payoff against $\mathbf{y}$ is at least $\frac{1}{2k}$. To see this, let $\mathbf{x}'$ be a row-player strategy that mixes uniformly over all rows. Consider a column $j \neq c$, and observe that by construction, column j contains exactly one 0 entry for the column player. Therefore, we have $(C \cdot \mathbf{x}')_j = 1 - \frac{1}{k}$. Since $\mathbf{y}$ assigns at least $\frac{1}{2}$ probability to columns other than c, we have that the payoff of $\mathbf{y}$ against $\mathbf{x}'$ is at most:

$$\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot (1 - \frac{1}{k}) = 1 - \frac{1}{2k}.$$

Since the game is constant-sum, this then implies that the payoff of $\mathbf{x}'$ against $\mathbf{y}$ is at least $\frac{1}{2k}$.

Since the row player's best response payoff against $\mathbf{y}$ must be at least $\frac{1}{2k}$, and since $(\mathbf{x}, \mathbf{y})$ is an ϵ-Nash equilibrium with $\epsilon < \frac{1}{4k}$, we have that the payoff of $\mathbf{x}$ against $\mathbf{y}$ is at least $\frac{1}{2k} - \epsilon$. Since the game is constant-sum, this then implies that the payoff of $\mathbf{y}$ against $\mathbf{x}$ is at most $1 - \frac{1}{2k} + \epsilon$. However, since the column player can obtain a payoff of 1 by playing column c, we have shown that the regret of $\mathbf{y}$ is at least $\frac{1}{2k} - \epsilon > \frac{1}{4k} > \epsilon$. Therefore, we have a contradiction with the fact that $(\mathbf{x}, \mathbf{y})$ is an ϵ-Nash equilibrium. □

Recall that our goal is to show that randomized algorithms cannot determine if a column is the hidden column in a query efficient manner for the games in $\mathcal{G}^k$. In the next lemma, we will formalise this idea. Suppose that our algorithm makes a series of randomized queries. For each column j, we will use A_j to be an indicator variable for the event "the algorithm did not find a row i such that $C_{i,j} = 0$". Intuitively, if $A_j = 1$, then the algorithm cannot rule out the possibility that column j is the hidden column. The following lemma gives a simple bound on the probability of A_j.

Lemma 4. *Let (R, C) be a game drawn from $\mathcal{G}^k$. Suppose that an algorithm makes $f(k)$ queries in column j. We have $\Pr(A_j = 1) \geq 1 - \frac{f(k)}{k}$.*

Proof. If $j = c$, then $\Pr(A_j = 1) = 1$, and the lemma holds. Otherwise, since the row r_j was chosen uniformly at random, if we make $f(k)$ queries in column j, then we find r_j with probability $\frac{f(k)}{k}$. Therefore $\Pr(A_j = 1) = 1 - \frac{f(k)}{k}$. □

Note that, if $f(k) \in o(k)$, then $1 - \frac{f(k)}{k}$ tends to 1 as k tends to infinity. Thus we will have $A_j = 1$ almost always, and j will be indistinguishable from c. In the following lemma, we use this fact to show our lower bound. In particular, we show that no algorithm that makes $o(k^2)$ payoff queries can succeed with a positive constant probability.

Lemma 5. *Any algorithm that makes $o(k^2)$ payoff queries cannot find an ϵ-Nash equilibrium, where $\epsilon < \frac{1}{4k}$, with positive constant probability against the distribution $\mathcal{G}^k$ for all k.*

Proof. Suppose, to the contrary, that there does exist an algorithm that makes at most $g(k) \in o(k^2)$ payoff queries, and when it is run against the distribution $\mathcal{G}^k$, it finds an ϵ-Nash equilibrium, where $\epsilon < \frac{1}{4k}$, with probability at least $3 \cdot \delta$, for constant $\delta \in (0, \frac{1}{3}]$. We consider the outcome when the algorithm is presented with the distribution $\mathcal{G}^k$, for some k that will later be varied. Let $\epsilon < \frac{1}{4k}$, and let $(\mathbf{x}, \mathbf{y})$ be the purported ϵ-Nash equilibrium found by the algorithm when it is run against $\mathcal{G}^k$.

Note that at least $k - \delta \cdot k$ columns must receive $o(k)$ payoff queries, because otherwise $\delta \cdot k$ columns would receive $\Omega(k)$ payoff queries, giving $\Omega(k^2)$ payoff queries in total. Thus, there must exist a set S of columns, with $|S| = k - \delta \cdot k$, and a function $f(k) \in o(k)$, such that each column in S received at most $f(k)$ payoff queries.

Let $S' \subseteq S$ be the set of columns in S such that $A_j = 1$. Recall that, for each column j in S', the algorithm cannot distinguish between j and c. We now prove a lower bound on the size of S'. By Lemma 4, we have that if $j \in S$, then $E[A_j] \geq 1 - \frac{f(k)}{k}$. Define $A = \sum_{j \in S} A_j$, and note that by linearity of expectations we have:

$$E[A] \geq (k - \delta \cdot k) \cdot (1 - \frac{f(k)}{k})$$

Since each of the events corresponding to A_j are independent, we can apply Hoeffding's inequality to obtain:

$$\Pr(|A - E[A]| \geq \delta \cdot k) \leq 2 \cdot \exp(-\frac{2 \cdot (\delta \cdot k)^2}{k - \delta k})$$

Thus, with probability at least $1 - 2 \cdot \exp(-\frac{2 \cdot (\delta \cdot k)^2}{k - \delta k})$ we have that S' contains at least

$$(k - \delta \cdot k) \cdot (1 - \frac{f(k)}{k}) - \delta \cdot k$$

columns.

Let us focus on the case where S' contains at least this many columns. Note that $\mathbf{y}$ can assign strictly more than $\frac{1}{2}$ probability to at most one column in S', and therefore there are at least $|S'| - 1$ columns in S' that are not assigned strictly more than $\frac{1}{2}$ probability by $\mathbf{y}$. Since c is chosen uniformly at random, and since the hidden column could be any of the columns in S', we have that $\mathbf{y}_c \leq \frac{1}{2}$ with probability at least:

$$\begin{aligned}(|S'| - 1) \cdot \frac{1}{k} &= \frac{(k - \delta \cdot k) \cdot (1 - \frac{f(k)}{k}) - \delta \cdot k - 1}{k} \\ &= (1 - \delta) \cdot (1 - \frac{f(k)}{k}) - \delta - \frac{1}{k}.\end{aligned}$$

By Lemma 3, if $\mathbf{y}_c \leq \frac{1}{2}$, then $(\mathbf{x}, \mathbf{y})$ is not an ϵ-Nash equilibrium.

In summary, we have that $(\mathbf{x}, \mathbf{y})$ is not a ϵ-Nash equilibrium with probability at least

$$\left(1 - 2 \cdot \exp(-\frac{2 \cdot (\delta \cdot k)^2}{k - \delta k})\right) \cdot \left((1 - \delta) \cdot (1 - \frac{f(k)}{k}) - \delta - \frac{1}{k}\right),$$

where the first term is the probability that S' contains enough columns, and the second term is the probability that $\mathbf{y}_c \leq \frac{1}{2}$. As k tends to infinity, we have that $\exp(-\frac{2\cdot(\delta\cdot k)^2}{k-\delta k})$ tends to 0, and since $f(k) \in o(k)$, we have that $\frac{f(k)}{k}$ tends to 0. Thus, the entire expression tends to $1 - 2 \cdot \delta$. Thus, for large k, the algorithm will fail with probability strictly greater than $1 - 3 \cdot \delta$, which contradicts our assumption that the algorithm succeeds with probability at least $3 \cdot \delta$ for all k. □

Lemma 5 shows that every algorithm that makes $o(k^2)$ payoff queries on a game drawn from $\mathcal{G}^k$ will fail to find an ϵ-Nash equilibrium, where $\epsilon < \frac{1}{4k}$, with probability tending to 1 as k tends to infinity. Moreover, recall that all games in $\mathcal{G}^k$ are zero-one constant-sum games. Therefore, we have shown the following theorem, which is the main result of this section.

Theorem 2. *The randomized query complexity of finding an ϵ-Nash equilibrium for $\epsilon < \frac{1}{4k}$ is $\Omega(k^2)$, even in zero-one constant-sum games.*

5 Zero-sum Games

We now turn our attention to showing upper bounds. In Sections 6 and 7, we will give query efficient randomized algorithms for finding $(\frac{3-\sqrt{5}}{2} + \epsilon)$-Nash equilibria, and $(\frac{2}{3} + \epsilon)$-WSNE, respectively. In this section, we give some preliminary results on zero-sum games, which are required for our later results: the result in Section 6 requires us to find an ϵ-Nash equilibrium of a zero-sum game, and the result in Section 7 requires us to find an ϵ-WSNE of a zero-sum game. In Section 5.1, we present the previous work of Goldberg and Roth [11], which provides a randomized query-efficient algorithm for finding an ϵ-Nash equilibrium in a zero-sum game, and in Section 5.2, we show how this can be converted into a randomized query-efficient algorithm for finding an ϵ-WSNE in a zero-sum game.

Recall that we assumed that all bimatrix game payoffs lie in the range $[0, 1]$. The two algorithms that we adapt both solve games that meet this assumption. In doing so, both algorithms create and solve a derived zero-sum game with payoffs lying in the range $[-1, 1]$. Thus, for the sake of convenience, during this section on zero-sum games, we assume that all payoffs lie in the range $[-1, 1]$.

5.1 A randomized algorithm for finding an ϵ-Nash equilibrium of a zero-sum game

The following theorem, shown by Goldberg and Roth [11] using using multiplicative weights update no-regret algorithms, states that we have a randomized $O(\frac{k\cdot\log k}{\epsilon^2})$ payoff query algorithm that finds an ϵ-Nash equilibrium in a zero-sum game.

Theorem 3 ([11]). *An ϵ-Nash equilibrium in a $k \times k$ zero-sum bimatrix game can, with probability $1 - k^{-\frac{1}{8}}$, be computed using $O(\frac{k\cdot\log k}{\epsilon^2})$ payoff queries.*

When we apply this result, we will also need to know the payoff vectors for both players. We now give a randomized $O(\frac{k \cdot \log k}{\epsilon^2})$ algorithm for discovering *approximate* payoff vectors. Let $(\mathbf{x}, \mathbf{y})$ be a mixed strategy profile in a $k \times k$ bimatrix game (R, C). We say that a k-dimensional vector $\mathbf{r}$ is an *ϵ-approximate payoff vector* for the row player if, for each $i \in [k]$, we have $|\mathbf{r}_i - R_i \cdot \mathbf{y}| \leq \epsilon$. Similarly, we say that a k-dimensional vector $\mathbf{c}$ is an ϵ-approximate payoff vector for the column player if, for each $j \in [k]$, we have $|\mathbf{c}_j - (\mathbf{x} \cdot C)_j| \leq \epsilon$. The following lemma shows that we can use a randomized algorithm to find an ϵ-approximate payoff vector for the row player using $O(\frac{k \cdot \log k}{\epsilon^2})$ payoff queries.

Lemma 6. *Let $(\mathbf{x}, \mathbf{y})$ be a mixed strategy profile in a $k \times k$ bimatrix game (R, C). With probability at least $1 - \frac{2}{k}$ we can find an ϵ-approximate payoff vector for the row player using $O(\frac{k \cdot \log k}{\epsilon^2})$ payoff queries.*

Proof. We begin by giving a randomized method for finding the payoff of a fixed row $i \in [k]$ with probability $1 - \frac{2}{k^2}$ using $\frac{4 \cdot \ln k}{\epsilon^2}$ many payoff queries. We will make $T = \frac{4 \cdot \ln k}{\epsilon^2}$ many payoff queries along row i, chosen according to the probability distribution $\mathbf{y}$. Let $X_1, X_2, \ldots, X_T$ be the result of these queries, and define $\overline{X} = \frac{1}{T}(X_1 + X_2 + \cdots + X_T)$. Applying Hoeffding's inequality, and noting that all payoffs lie in the range $[-1, 1]$, gives:

$$\begin{aligned}\Pr(|\overline{X} - R_i \cdot \mathbf{y}| \geq \epsilon) &\leq 2 \cdot \exp(-\frac{2 \cdot T \cdot \epsilon^2}{(1+1)^2}) \\ &= 2 \cdot \exp(-2 \cdot \ln k) \\ &= \frac{2}{k^2}.\end{aligned}$$

Thus, with probability $1 - \frac{2}{k^2}$, we have that $\overline{X}$ is within ϵ of $(R \cdot \mathbf{y})_i$.

Now, to find an ϵ-approximation of the row player's payoffs, we simply apply the above method separately for each row $i \in [k]$. Clearly, this will use $O(\frac{k \cdot \ln k}{\epsilon^2})$ many payoff queries. The probability that we correctly compute an ϵ-approximation of the row player's payoffs is:

$$\begin{aligned}\left(1 - \frac{2}{k^2}\right)^k &\geq 1 - \binom{k}{1} \cdot \frac{2}{k^2} \\ &= 1 - \frac{2}{k}.\end{aligned}$$

This completes the proof. □

Note that, since we can swap the roles of the two players, Lemma 6 can also be used to find an ϵ-approximate payoff vector for the column player. Combining Lemma 6 with Theorem 3 gives the following corollary.

Corollary 1. *Given a $k \times k$ zero-sum bimatrix game, with probability at least $(1-k^{-\frac{1}{8}})(1-\frac{2}{k})^2$, we can compute an ϵ-Nash equilibrium $(\mathbf{x}, \mathbf{y})$, and ϵ-approximate payoff vectors for both players under $(\mathbf{x}, \mathbf{y})$, using $O(\frac{k \cdot \log k}{\epsilon^2})$ payoff queries.*

We now introduce further notation for working with approximate payoff vectors. Let $(\mathbf{x}, \mathbf{y})$ be a mixed strategy profile in a bimatrix game (R, C). Let $\mathbf{r}$ and $\mathbf{c}$ be ϵ-approximate payoff vectors for the row and column player, respectively. We say that a row i is a *best response according to* $\mathbf{r}$ if $\mathbf{r}_i = \max_{l \in [k]} \mathbf{r}_l$, and that a column j is a best response according to $\mathbf{c}$ if $\mathbf{c}_j = \max_{l \in [k]} \mathbf{c}_l$. We will frequently use the fact that, if i is a best response according to $\mathbf{r}$, and if i' is an actual best response against $\mathbf{y}$, then:

$$|\mathbf{r}_i - R_{i'} \cdot \mathbf{y}| \leq \epsilon. \tag{1}$$

This is because, if $\mathbf{r}_i > R_{i'} \cdot \mathbf{y} + \epsilon$, then $R_i \cdot \mathbf{y} > R_{i'} \cdot \mathbf{y}$, which would contradict the fact that i' is a best response, and if $\mathbf{r}_i < R_{i'} \cdot \mathbf{y} + \epsilon$ then $\mathbf{r}_i < \mathbf{r}_{i'}$, which would contradict the fact that i is a best response according to $\mathbf{r}$. If row i is a best response according to $\mathbf{r}$, then we define the row player's *regret according to* $\mathbf{r}$ to be $\mathbf{r}_i - \mathbf{x} \cdot \mathbf{r}$. Similarly, if column j is a best response according to $\mathbf{c}$, then we define the regret according to $\mathbf{c}$ to be $\mathbf{c}_j - \mathbf{c} \cdot \mathbf{y}$. Note that, if i is an actual best response against $\mathbf{y}$, then we have: $|(\mathbf{r} - \mathbf{x} \cdot \mathbf{r}) - (R_i \cdot \mathbf{y} - \mathbf{x} \cdot R \cdot \mathbf{y})| \leq 2\epsilon$. In other words, the regret according to $\mathbf{r}$ is within 2ϵ of the actual regret suffered by the row player under $(\mathbf{x}, \mathbf{y})$.

5.2 A randomized algorithm for finding an ϵ-WSNE of a zero-sum game

In this section, we give a randomized query efficient algorithm for finding approximate well-supported Nash equilibria in zero-sum games. Our approach is to first compute an approximate Nash equilibrium of the zero-sum game using Corollary 1, and to then convert this into an ϵ-WSNE for the zero-sum game. Chen, Deng, and Teng have given an algorithm (henceforth referred to as the CDT algorithm) that takes a $\frac{\epsilon^2}{8}$-Nash equilibrium of a game, and in polynomial-time finds an ϵ-WSNE for that game [5]. However, their method requires that we know the payoff of every pure strategy in the $\frac{\epsilon^2}{8}$-Nash equilibrium. In our setting, we only know approximate payoff vectors for both players, so the CDT algorithm cannot be directly applied. In the next lemma, we show a variant of their result, which can be applied when we only know approximate payoff vectors.

Lemma 7. *Let (R, C) be a bimatrix game, let $(\mathbf{x}, \mathbf{y})$ be an $\frac{\epsilon^2}{24}$-Nash equilibrium of (R, C), and let $\mathbf{r}$ and $\mathbf{c}$ be $\frac{\epsilon^2}{24}$-approximate payoff vectors for the row and column player under $(\mathbf{x}, \mathbf{y})$. Without making any payoff queries, we can construct an ϵ-WSNE of (R, C).*

Proof. Let i^* be an actual best response against $\mathbf{y}$, and let l^* be a best response according to $\mathbf{r}$. We define the set $B = \{l \,:\, \mathbf{r}_{l^*} > \mathbf{r}_l + \frac{\epsilon}{4}\}$, and our first task is to show that B contains every row l that is not a $\frac{\epsilon}{2}$-best response against $\mathbf{y}$.

Note that $|\mathbf{r}_{l^*} - R_{i^*} \cdot \mathbf{y}| \leq \frac{\epsilon^2}{24}$. Furthermore, note that for every row l we have:

$$R_{i^*} \cdot \mathbf{y} - R_l \cdot \mathbf{y} \leq \mathbf{r}_{l^*} - \mathbf{r}_l + \frac{\epsilon^2}{12}.$$

Therefore, if row l satisfies $\mathbf{r}_{l^*} \leq \mathbf{r}_l + \frac{\epsilon}{4}$ then:

$$\begin{aligned} R_{i^*} \cdot \mathbf{y} - R_l \cdot \mathbf{y} &\leq \frac{\epsilon}{4} + \frac{\epsilon^2}{12} \\ &\leq \frac{\epsilon}{2}. \end{aligned}$$

This implies that l is an $\frac{\epsilon}{2}$-best response against $\mathbf{y}$. So, the set B must contain every row l that is not an $\frac{\epsilon}{2}$-best response against $\mathbf{y}$.

Define the random variable $Y = R_{i^*} \cdot \mathbf{y} - \mathbf{x} \cdot R \cdot \mathbf{y}$ and the random variable $X = \mathbf{r}_{l^*} - \mathbf{x} \cdot \mathbf{r}_l$. Note that, since $\mathbf{x}$ is a $\frac{\epsilon^2}{24}$-Nash equilibrium, we know that the row player suffers regret at most $\frac{\epsilon^2}{24}$ under $(\mathbf{x}, \mathbf{y})$, and therefore we have $E[Y] \leq \frac{\epsilon^2}{24}$. Furthermore, since X is the regret according to $\mathbf{r}$, we know that:

$$\begin{aligned} E[X] &\leq E[Y] + \frac{\epsilon^2}{12} \\ &\leq \frac{\epsilon^2}{8}. \end{aligned}$$

By applying Markov's inequality, we obtain:

$$\begin{aligned} \Pr(X \geq \frac{\epsilon}{4}) &\leq \frac{\epsilon^2}{8} \Big/ \frac{\epsilon}{4} \\ &= \frac{\epsilon}{2}. \end{aligned}$$

Therefore, $\mathbf{x}$ must assign at most $\frac{\epsilon}{2}$ probability to rows in B. We define $\mathbf{x}'$ to be a modification of $\mathbf{x}$ where all probability assigned to rows in B is is shifted arbitrarily to rows that are not in B.

Now consider the column player. If j^* is a best response according to $\mathbf{c}$ and we define $B' = \{j \, : \, \mathbf{c}_{j^*} > \mathbf{c}_j + \frac{\epsilon}{4}\}$, then we can use the same argument as above to prove that B' contains every column that is not a $\frac{\epsilon}{2}$-best response against $\mathbf{x}$, and that the column player assigns at most $\frac{\epsilon}{2}$ probability to the columns in B'. Similarly, we define $\mathbf{y}'$ to be a modification of $\mathbf{y}$ where all probability assigned to columns in B' is shifted arbitrarily to columns not in B'.

Note that every row in $\text{Supp}(\mathbf{x}')$ is a $\frac{\epsilon}{2}$-best response against $\mathbf{y}$. Since $\mathbf{y}'$ differs from $\mathbf{y}$ by a shift of at most $\frac{\epsilon}{2}$ probability, we have that every row in $\text{Supp}(\mathbf{x}')$ is an ϵ-best response against $\mathbf{y}'$. Using the same technique, we can argue that every row in $\text{Supp}(\mathbf{y}')$ is an ϵ-best response against $\mathbf{x}'$, and therefore $(\mathbf{x}', \mathbf{y}')$ is an ϵ-WSNE. □

By using the reduction from Lemma 7, it is now easy to see that we can compute an ϵ-WSNE of a zero-sum bimatrix game. The following corollary is a combination of Corollary 1 and Lemma 7.

Corollary 2. *Given a $k \times k$ zero-sum bimatrix game, with probability at least $(1 - k^{-\frac{1}{8}})(1 - \frac{2}{k})^2$, we can compute an ϵ-WSNE $(\mathbf{x}, \mathbf{y})$ using $O(\frac{k \cdot \log k}{\epsilon^4})$ payoff queries.*

6 A randomized algorithm for finding a $(\frac{3-\sqrt{5}}{2} + \epsilon)$-Nash equilibrium

In this section, we present a randomized payoff-query efficient algorithm for finding a $(\frac{3-\sqrt{5}}{2}+\epsilon)$-Nash equilibrium in a bimatrix game, where $\frac{3-\sqrt{5}}{2} \approx 0.38197$. We adapt an algorithm of Bosse, Byrka, and Markakis [4] (henceforth referred to as the BBM algorithm) for finding a $(\frac{3-\sqrt{5}}{2} + \epsilon)$-Nash equilibrium in a bimatrix game. The BBM algorithm solves a zero-sum game, and then makes a decision based on the regret suffered by the players. We must adapt the algorithm to work with approximate payoff vectors.

Let (R, C) be a $k \times k$ bimatrix game, and define $D = R - C$. Let $\alpha \in [0, 1]$ be a parameter that will be fixed later. The algorithm is as follows.

1. Apply Theorem 3 for $\frac{\epsilon}{4}$ to the game $(D, -D)$ in order to obtain $(\mathbf{x}, \mathbf{y})$, which is a $\frac{\epsilon}{4}$-Nash equilibrium. Then apply Lemma 6 in order to find $\mathbf{r}$ and $\mathbf{c}$, which are $\frac{\epsilon}{4}$-approximate payoff vectors for when $(\mathbf{x}, \mathbf{y})$ is played in the game (R, C). This step succeeds with probability $(1 - k^{-\frac{1}{8}})(1 - \frac{2}{k})^2$ and uses $O(\frac{k \cdot \log k}{\epsilon^2})$ payoff queries.
2. We will assume, without loss of generality, that the regret according to $\mathbf{r}$ is larger than the regret according to $\mathbf{c}$. Let row b be a best response according to $\mathbf{r}$ in the game (R, C), and let g be the regret according to $\mathbf{r}$. Since b and g are determined by $\mathbf{r}$, this step requires no payoff queries.
3. Let d be a best response for the column player against row b in the game (R, C). This can be found using k payoff queries, by querying every column in row b.
4. If $g \leq \alpha$, then output $(\mathbf{x}, \mathbf{y})$. Otherwise, let $\delta = \frac{1-g}{2-g}$ and output the following strategy, denoted as $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$: the row player plays b as a pure strategy, and the column player plays $\mathbf{y}$ with probability $(1 - \delta)$ and d with probability δ. This step uses no payoff queries.

The following lemma shows that this algorithm is correct, in the case Theorem 3 succeeds in finding an approximate Nash equilibrium of $(D, -D)$, and Lemma 6 succeeds in finding the approximate payoff vectors in (R, C).

Lemma 8. *If $(\mathbf{x}, \mathbf{y})$ is a $\frac{\epsilon}{4}$-Nash equilibrium of $(D, -D)$, and $\mathbf{r}$ and $\mathbf{c}$ are $\frac{\epsilon}{4}$-approximate payoff vectors for $(\mathbf{x}, \mathbf{y})$ in (R, C), then the algorithm outputs a $(\max(\alpha, \frac{1-\alpha}{2-\alpha}) + \epsilon)$-Nash equilibrium of (R, C).*

Proof. First consider the case where $g \leq \alpha$. Let i be a best response for the row player against $\mathbf{y}$. By applying Equation (1), we can obtain the following bound on the row player's regret:

$$\begin{aligned} R_i \cdot \mathbf{y} - \mathbf{x} \cdot R \cdot \mathbf{y} &\leq \mathbf{r}_b - \mathbf{x} \cdot \mathbf{r} + \frac{\epsilon}{2} \\ &= g + \frac{\epsilon}{2} \\ &\leq \alpha + \frac{\epsilon}{2} \end{aligned}$$

Thus, the row player's regret at most $\alpha + \frac{\epsilon}{2}$ under $(\mathbf{x}, \mathbf{y})$. Since, by assumption, the regret according to $\mathbf{c}$ is at most g, we can use the same argument to prove that the column player's regret is at most $\alpha + \frac{\epsilon}{2}$. Therefore, $(\mathbf{x}, \mathbf{y})$ is a $(\alpha + \frac{\epsilon}{2})$-Nash equilibrium.

We now consider the case where $g > \alpha$. We first consider the row player. Let $\hat{b}$ be a best response against $\hat{\mathbf{y}}$. The row player's regret in $(\hat{\mathbf{x}}, \hat{\mathbf{y}})$ is:

$$\begin{aligned}
R_{\hat{b}} \cdot \hat{\mathbf{y}} - R_b \cdot \hat{\mathbf{y}} &= (1-\delta)(R_{\hat{b}} \cdot \mathbf{y} - R_b \cdot \mathbf{y}) + \delta \cdot (R_{\hat{b},d} - R_{b,d}) \\
&\leq (1-\delta)(\mathbf{r}_{\hat{b}} - \mathbf{r}_b + \frac{\epsilon}{2}) + \delta \cdot (R_{\hat{b},d} - R_{b,d}) \\
&\leq (1-\delta) \cdot \frac{\epsilon}{2} + \delta \cdot (R_{\hat{b},d} - R_{b,d}) \\
&\leq \frac{\epsilon}{2} + \delta.
\end{aligned}$$

The first inequality holds because $\mathbf{r}$ is is an $\frac{\epsilon}{4}$-approximate payoff vector against $\mathbf{y}$. The second inequality holds because $b = \max_{i \in [k]} \mathbf{r}_i$. The third inequality holds because $1 - \delta \leq 1$ and $R_{i,j} \in [0, 1]$ for all i and j.

So, we have shown that the row player's regret is at most

$$\begin{aligned}
\delta + \frac{\epsilon}{2} &= \frac{1-g}{2-g} + \frac{\epsilon}{2} \\
&\leq \frac{1-\alpha}{2-\alpha} + \frac{\epsilon}{2}
\end{aligned}$$

The inequality above holds because $\frac{1-g}{2-g}$ is a decreasing function when $g \leq 1$, and because $g > \alpha$.

We now consider the column player. Since $(\mathbf{x}, \mathbf{y})$ is an $\frac{\epsilon}{4}$-Nash equilibrium of $(D, -D)$, the row player's regret under $(\mathbf{x}, \mathbf{y})$ in $(D, -D)$ must be at most $\frac{\epsilon}{4}$. Thus, all rows i must satisfy $D_i \cdot \mathbf{y} \leq \mathbf{x} \cdot D \cdot \mathbf{y} + \frac{\epsilon}{4}$. Applying this inequality for row b gives the following:

$$\begin{aligned}
D_b \cdot \mathbf{y} &\leq \mathbf{x} \cdot D \cdot \mathbf{y} + \frac{\epsilon}{4} \\
(R - C)_b \cdot \mathbf{y} &\leq \mathbf{x} \cdot (R - C) \cdot \mathbf{y} + \frac{\epsilon}{4} \\
R_b \cdot \mathbf{y} - \mathbf{x} \cdot R \cdot \mathbf{y} + \mathbf{x} \cdot C \cdot \mathbf{y} - \frac{\epsilon}{4} &\leq C_b \cdot \mathbf{y} \qquad (2)
\end{aligned}$$

Since $\mathbf{r}$ is a $\frac{\epsilon}{4}$-approximate payoff vector we have:

$$\begin{aligned}
R_b \cdot \mathbf{y} - \mathbf{x} \cdot R \cdot \mathbf{y} &\geq \mathbf{r}_b - \mathbf{x} \cdot \mathbf{r} - \frac{\epsilon}{2} \\
&= g - \frac{\epsilon}{2}
\end{aligned}$$

Substituting this into Equation (2)

$$\begin{aligned}
C_b \cdot \mathbf{y} &\geq g + \mathbf{x} \cdot C \cdot \mathbf{y} - \frac{3}{4}\epsilon \\
&\geq g - \epsilon \qquad (3)
\end{aligned}$$

Recall that d is an actual best response against b for the column player. Since $\hat{\mathbf{x}}$ plays b as a pure strategy, we have therefore have that d is a best response against $\hat{\mathbf{x}}$. Moreover, observe that

$$\begin{aligned}(1-\delta)(1-g) &= \frac{(1-g)(2-g)+(1-g)^2}{2-g} \\ &= \frac{1-g}{2-g}. \qquad (4)\end{aligned}$$

Thus, the column player's regret when playing $\hat{\mathbf{y}}$ against $\hat{\mathbf{x}}$ is:

$$\begin{aligned}(\hat{\mathbf{x}}\cdot C)_d - \hat{\mathbf{x}}\cdot C\cdot\hat{\mathbf{y}} &= C_{b,d} - C_b\cdot\hat{\mathbf{y}} \\ &= C_{b,d} - ((1-\delta)\cdot C_b\cdot\mathbf{y} + \delta\cdot C_{b,d}) \\ &= (1-\delta)(C_{b,d} - C_b\cdot\mathbf{y}) \\ &\le (1-\delta)(1-g+\epsilon) && \text{[By Equation (3)]} \\ &\le \frac{1-g}{2-g} + \epsilon && \text{[By Equation (4)]} \\ &< \frac{1-\alpha}{2-\alpha} + \epsilon.\end{aligned}$$

We have now shown that, in the case where $g > \alpha$, both players have regret at most $\frac{1-\alpha}{2-\alpha} + \epsilon$, which implies that $(\hat{x}, \hat{y})$ is a $(\frac{1-\alpha}{2-\alpha} + \epsilon)$-Nash equilibrium. □

As shown by Bosse, Byrka, and Markakis, we have that the expression $\max(\alpha, \frac{1-\alpha}{2-\alpha})$ is minimized when $\alpha = \frac{3-\sqrt{5}}{2} \approx 0.38197$. Thus, we have the following theorem.

Theorem 4. *Given a $k\times k$ bimatrix game, with probability at least $(1-k^{-\frac{1}{8}})(1-\frac{2}{k})^2$, we can compute a $(\frac{3-\sqrt{5}}{2} + \epsilon)$-Nash equilibrium using $O(\frac{k\cdot\log k}{\epsilon^2})$ payoff queries.*

7 A randomized algorithm for finding finding a $(\frac{2}{3} + \epsilon)$-well-supported Nash equilibrium

In this section, we give a randomized $O(\frac{k\cdot\log k}{\epsilon^4})$ payoff query algorithm for finding a $(\frac{2}{3} + \epsilon)$-WSNE equilibrium in a bimatrix game, and we give a randomized $O(\frac{k\cdot\log k}{\epsilon^4})$ payoff query algorithm for finding a $(\frac{1}{2} + \epsilon)$-WSNE in a zero-one bimatrix game. We follow the algorithm of Kontogiannis and Spirakis (henceforth referred to as the KS algorithm) for finding a $\frac{2}{3}$-WSNE in a bimatrix game [14]. Their approach can be summarised as follows: they first perform a preprocessing step in which they check whether the game has a pure $\frac{2}{3}$-WSNE. If it does not, then they construct a zero-sum game and compute an exact Nash equilibrium for it. They show that, if the original game does not have a pure $\frac{2}{3}$-WSNE, then an exact Nash equilibrium in the zero-sum game is a $\frac{2}{3}$-WSNE in the original game.

There are two problems with this approach in the payoff query setting. Firstly, it has been shown that finding an exact Nash equilibrium of a $k \times k$ zero-sum game requires k^2 queries [8]. To solve this problem, we substitute an ϵ-WSNE of the zero-sum game in place of an exact Nash equilibrium, and we show that, after this substitution, the KS algorithm will still produce a well-supported Nash equilibrium of the original game. The second problem is that we are unable to perform the preprocessing step in a query-efficient manner. A naive algorithm would require k^2 payoff queries in order to verify whether there is a pure $\frac{2}{3}$-WSNE, and it is not clear how this can be improved. To solve this problem, we show that the preprocessing step does not need to be carried out before the zero-sum game has been solved. Instead, we find an ϵ-WSNE in the zero-sum game, and then check whether it is a $\frac{2}{3}$-WSNE in the original. If it is not, then we show how this information can be used to find a pure $(\frac{2}{3}+\epsilon)$-WSNE in the original game.

Let (R, C) be a bimatrix game, where R is the payoff matrix of the row player, and C is the payoff matrix of the column player. The KS algorithm uses the following definitions.

$$D := \frac{1}{2}(R - C) \qquad\qquad X := -\frac{1}{2}(R + C)$$

Observe that $D = R + X$. The KS algorithm finds an exact Nash equilibrium of the zero-sum game $(D, -D)$. In contrast to this, we do the following:

- we will apply Corollary 2 to $(D, -D)$ to obtain an $\frac{\epsilon}{10}$-WSNE for $(D, -D)$, which we denote as $(\mathbf{x}, \mathbf{y})$.
- We then obtain approximate payoff vectors for when $(\mathbf{x}, \mathbf{y})$ is played in (R, C): we apply Lemma 6 to obtain $\frac{\epsilon}{10}$-approximate payoff vectors $\mathbf{r}$, which approximates $R \cdot \mathbf{y}$, and $\mathbf{c}$, which approximates $\mathbf{x} \cdot C$.

These two steps succeed with probability at least $(1 - k^{-\frac{1}{8}})(1 - \frac{2}{k})^3$. We will fix $(\mathbf{x}, \mathbf{y})$, $\mathbf{r}$, and $\mathbf{c}$ for the rest of this section.

In the following lemma, we show how the preprocessing of the KS algorithm can be delayed until after the zero-sum game has been solved. In particular, we show that if there is a row in the support of $\mathbf{x}$ that is far from being an approximate best response, then, in a query-efficient manner, we can find a pure strategy profile $i, j \in [k]$ such that $(R+C)_{i,j}$ is large. The parameter z will allow us to apply this lemma for both the zero-one and general bimatrix games: in the zero-one case we will set $z = 0.5$, and for general bimatrix games we will set $z = \frac{2}{3}$.

Lemma 9. *Let $z \in [0, 1]$. If there is an $i' \in \text{Supp}(\mathbf{x})$ and $i \in [k]$ such that $\mathbf{r}_i > \mathbf{r}_{i'} + z - \frac{\epsilon}{5}$, then using k payoff queries, we can find a $j \in [k]$ such that $(R + C)_{i,j} > 2z - \epsilon$.*

Proof. Since $\mathbf{r}$ is an $\frac{\epsilon}{10}$-approximate payoff vector for the row player, and since $\mathbf{r}_i > \mathbf{r}_{i'} + z - \frac{\epsilon}{5}$ we have:

$$R_i \cdot \mathbf{y} > R_{i'} \cdot \mathbf{y} + z - \frac{2}{5}\epsilon. \tag{5}$$

Since $(\mathbf{x}, \mathbf{y})$ is an $\frac{\epsilon}{10}$-WSNE in $(D, -D)$, and since $i' \in \text{Supp}(\mathbf{x})$ by assumption, we have:

$$\begin{aligned} D_{i'} \cdot \mathbf{y} &\geq D_i \cdot \mathbf{y} - \frac{\epsilon}{10} \\ (R+X)_{i'} \cdot \mathbf{y} &\geq (R+X)_i \cdot \mathbf{y} - \frac{\epsilon}{10} \\ R_{i'} \cdot \mathbf{y} &\geq R_i \cdot \mathbf{y} - (X_{i'} - X_i) \cdot \mathbf{y} - \frac{\epsilon}{10}. \end{aligned}$$

Combining this with Equation (5) yields:

$$\begin{aligned} R_{i'} \cdot \mathbf{y} &> R_{i'} \cdot \mathbf{y} + z - \frac{2\epsilon}{5} - (X_{i'} - X_i) \cdot \mathbf{y} - \frac{\epsilon}{10} \\ (X_{i'} - X_i) \cdot \mathbf{y} &> z - \frac{\epsilon}{2}. \end{aligned}$$

Since $X = -\frac{1}{2}(R+C)$, we have that $X_{i'} \cdot \mathbf{y} \in [-1, 0]$ and hence $X_{i'} \cdot \mathbf{y} \leq 0$. Therefore, we have:

$$\begin{aligned} -X_i \cdot \mathbf{y} &> z - \frac{\epsilon}{2} \\ \frac{1}{2}(R+C)_i \cdot \mathbf{y} &> z - \frac{\epsilon}{2} \\ (R+C)_i \cdot \mathbf{y} &> 2z - \epsilon. \end{aligned}$$

In order for this inequality to hold, there must be at least one column j such that $(R+C)_{i,j} > 2z - \epsilon$. Since we know row i, we can find column j using k payoff queries. □

Note that, by swapping the roles of the two players, Lemma 9 can also be applied for the column player. We can now prove the main result of this section.

Theorem 5. *Let (R, C) be a $k \times k$ bimatrix game. With probability at least $(1 - k^{-\frac{1}{8}})(1 - \frac{2}{k})^3$ and using $O(\frac{k \cdot \log k}{\epsilon^4})$ we can compute a $(\frac{2}{3} + \epsilon)$-WSNE if (R, C) is a general bimatrix game, or a $(\frac{1}{2} + \epsilon)$-WSNE if (R, C) is a zero-one bimatrix game.*

Proof. As we have described, the algorithm spends $O(\frac{k \cdot \log k}{\epsilon^4})$ in order to compute $(\mathbf{x}, \mathbf{y})$, $\mathbf{r}$, and $\mathbf{c}$. We first prove the result for general bimatrix games. Observe that, since since $\mathbf{r}$ is an $\frac{\epsilon}{10}$-approximate payoff vector, if we have $\mathbf{r}_i \leq \mathbf{r}_{i'} + \frac{2}{3} - \frac{\epsilon}{5}$ for two rows $i, i' \in [k]$, then we have $R_i \cdot \mathbf{y} \leq R_{i'} + \frac{2}{3}$. So, our algorithm will check whether:

- $\mathbf{r}_i \leq \mathbf{r}_{i'} + \frac{2}{3} - \frac{\epsilon}{5}$ for every $i \in [k]$ and every $i' \in \text{Supp}(\mathbf{x})$, and
- $\mathbf{c}_j \leq \mathbf{c}_{j'} + \frac{2}{3} - \frac{\epsilon}{5}$ for every $j \in [k]$ and every $j' \in \text{Supp}(\mathbf{y})$.

If both of these checks succeed, then $(\mathbf{x}, \mathbf{y})$ is a $\frac{2}{3}$-WSNE in (R, C), and we are done. On the other hand, if either of the two checks fail, then we can apply Lemma 9 to obtain a pair $i, j \in [k]$ such that $(R+C)_{i,j} > \frac{4}{3} - \epsilon$. This implies that both $R_{i,j} > \frac{1}{3} - \epsilon$ and $C_{i,j} > \frac{1}{3} - \epsilon$, and therefore i and j form a pure $(\frac{2}{3} + \epsilon)$-WSNE.

The proof is similar for zero-one games. The algorithm checks whether $\mathbf{r}_i \leq \mathbf{r}_{i'} + (\frac{1}{2} + \epsilon) - \frac{\epsilon}{5}$ for every $i \in [k]$ and every $i' \in \text{Supp}(\mathbf{x})$, and whether $\mathbf{c}_j \leq \mathbf{c}_{j'} + (\frac{1}{2} + \epsilon) - \frac{\epsilon}{5}$ for every $j \in [k]$ and every $j' \in \text{Supp}(\mathbf{y})$. If both checks succeed, then we have that $(\mathbf{x}, \mathbf{y})$ is a $(\frac{1}{2} + \epsilon)$-WSNE. Otherwise, we can apply Lemma 9 to obtain a pair $i, j \in [k]$ such that $(R + C)_{i,j} > 1 + 2\epsilon - \epsilon > 1$. However, since this is a zero-one game, the only way to have $(R + C)_{i,j} > 1$ is if $R_{i,j} = 1$ and $C_{i,j} = 1$. Therefore (i, j) is a pure Nash equilibrium. □

8 A linear lower bound for finding ϵ-WSNE when $\epsilon < 1$

Our result in Section 3 completes the characterization of the deterministic query complexity for ϵ-Nash equilibria for constant ϵ, but we do not have a characterization for ϵ-WSNE. Of course, since every ϵ-WSNE is an ϵ-Nash equilibrium, the lower bound of Section 3 applies for $\epsilon < \frac{1}{2}$, but we do not know much about the deterministic query complexity of ϵ-WSNE with $\epsilon \geq \frac{1}{2}$.

In this section, we give one easy initial observation on this topic: for every $\epsilon < 1$, the deterministic query complexity of finding an ϵ-WSNE is $\Omega(k)$. It has been shown that we can always find a $(1 - \frac{1}{k})$-Nash equilibrium using no queries at all [8], so this lemma shows that we should expect ϵ-WSNE to behave differently to ϵ-Nash equilibria when $\epsilon > \frac{1}{2}$.

To prove our result, we will assume that all payoff queries return payoff 0 for both the row and column player. We will show that, when all queries are responded to in this way, all algorithms must make at least $k - 1$ payoff queries in order to correctly determine an ϵ-WSNE. We first show the following lemma.

Lemma 10. *Let $(\mathbf{x}, \mathbf{y})$ be a ϵ-WSNE. Let r be a row that is played with probability strictly less than 1 in $\mathbf{x}$. At most ϵ probability can be assigned to columns in r that have not been queried.*

Proof. Suppose, for the sake of contradiction, that $\mathbf{x}$ assigns strictly more than ϵ probability to unqueried columns in r. Let $U = \{c : (r, c) \text{ did not receive a payoff query}\}$. We construct a row player payoff matrix R as follows:

$$R_{i,j} = \begin{cases} 1 & \text{if } i = r \text{ and } j \in U, \\ 0 & \text{otherwise.} \end{cases}$$

This matrix is consistent with all payoff queries that have been made so far. Since $\mathbf{x}$ assigns strictly more than ϵ probability to the columns in U, the payoff of row r is strictly greater than ϵ. Moreover, the payoff of every row $i \neq r$ is 0. Since r is not played with probability 1 by $\mathbf{x}$, some probability must be assigned to a row r' with payoff 0. Therefore, we have:

$$R_{r'} \cdot \mathbf{y} - R_r \cdot \mathbf{y} > \epsilon - 0.$$

This proves that row r is not an ϵ-best response against $\mathbf{y}$, which provides our contradiction. □

Having shown Lemma 10, we can now provide the lower bound.

Lemma 11. *For every $\epsilon < 1$, the deterministic query complexity of finding an ϵ-WSNE in a $k \times k$ bimatrix game is at least $k-1$, even in zero-one games.*

Proof (of Lemma 11). Let $(\mathbf{x}, \mathbf{y})$ be an ϵ-WSNE. Let W be the set of rows that are not played with probability 1 by $\mathbf{x}$. Note that W contains at least $k-1$ rows. By Lemma 10, in each row in W, there must be at least $1-\epsilon$ probability assigned to queried columns. Since $\epsilon < 1$, this implies that each row in W must have at least one queried column, which implies that we must have made at least $k-1$ queries. □

9 Conclusion

In this paper, we have given a complete characterization of the deterministic query complexity of ϵ-Nash equilibria for constant ϵ, we have given randomized upper bounds for both ϵ-Nash equilibria and ϵ-WSNE, and we have initiated work on lower bounds against randomized algorithms for finding ϵ-Nash equilibria.

There are many open problems arising from this work. In addition to ruling out query-efficient randomized algorithms for finding exact Nash equilibria, our lower bound in Section 4 also rules out query efficient algorithms for finding ϵ-Nash equilibria with $\epsilon < \frac{1}{4k}$ in $k \times k$ bimatrix games. The most obvious open problem stemming from this is to prove a lower bound for randomized algorithms for constant approximations. At the very least, it would be desirable to have matching $\Omega(k \cdot \log k)$ lower bounds for our algorithms in Sections 6 and 7. Also, does there exist a constant $\epsilon < \frac{3-\sqrt{5}}{2}$ for which the randomized query complexity is $\omega(k \cdot \log k)$, or a constant $\epsilon < \frac{2}{3}$ for which the randomized query complexity of finding an ϵ-WSNE is $\omega(k \cdot \log k)$?

We adapted two approximation algorithms to give randomized query-efficient protocols, but we do not use the best possible polynomial-time approximations. There is a polynomial-time algorithm that finds a 0.3393-Nash equilibrium [16] using gradient descent, but it is not clear how this could be efficiently implemented using queries. For WSNE, there is a polynomial-time algorithm that finds a $(\frac{2}{3} - 0.004735)$-WSNE [9], which uses an *exhaustive* search of all 2×2 supports to improve the KS-algorithm. Even if this could be efficiently implemented using queries, it would only be slightly better than the result in Section 7.

Acknowledgements

We would like to thank Paul Goldberg and Noam Nisan for useful discussions on this topic. This work is supported by EPSRC grants EP/H046623/1 and EP/L011018/1, and by ESRC grant ESRC/BSB/09.

Bibliography

[1] Anbalagan, Y., Norin, S., Savani, R., and Vetta, A. 2013. Polylogarithmic supports are required for approximate well-supported Nash equilibria below 2/3. In *Proc. of WINE*.

[2] Babichenko, Y. 2014. Query complexity of approximate Nash equilibria. In *Proc. of STOC*. To appear (arXiv version: abs/1306.6686).

[3] Babichenko, Y. and Barman, S. 2013. Query complexity of correlated equilibrium. *CoRR abs/1306.2437*.

[4] Bosse, H., Byrka, J., and Markakis, E. 2010. New algorithms for approximate Nash equilibria in bimatrix games. *Theor. Comput. Sci. 411,* 1, 164–173.

[5] Chen, X., Deng, X., and Teng, S.-H. 2009. Settling the complexity of computing two-player Nash equilibria. *J. ACM 56,* 3.

[6] Daskalakis, C., Mehta, A., and Papadimitriou, C. H. 2007. Progress in approximate Nash equilibria. In *ACM Conference on Electronic Commerce*. 355–358.

[7] Daskalakis, C., Mehta, A., and Papadimitriou, C. H. 2009. A note on approximate Nash equilibria. *Theor. Comput. Sci. 410,* 17, 1581–1588.

[8] Fearnley, J., Gairing, M., Goldberg, P. W., and Savani, R. 2013. Learning equilibria of games via payoff queries. In *ACM Conference on Electronic Commerce*. 397–414.

[9] Fearnley, J., Goldberg, P. W., Savani, R., and Sørensen, T. B. 2012. Approximate well-supported Nash equilibria below two-thirds. In *SAGT*. 108–119.

[10] Feder, T., Nazerzadeh, H., and Saberi, A. 2007. Approximating Nash equilibria using small-support strategies. In *Proc. of 8th ACM EC*. 352–354.

[11] Goldberg, P. and Roth, A. 2013. Bounds for the query complexity of approximate equilibria. *Electronic Colloquium on Computational Complexity (ECCC) TR13,* 136.

[12] Hart, S. and Nisan, N. 2013. The query complexity of correlated equilibria. In *Proc. of SAGT*.

[13] Kontogiannis, S. C., Panagopoulou, P. N., and Spirakis, P. G. 2009. Polynomial algorithms for approximating Nash equilibria of bimatrix games. *Theor. Comput. Sci. 410,* 17, 1599–1606.

[14] Kontogiannis, S. C. and Spirakis, P. G. 2010. Well supported approximate equilibria in bimatrix games. *Algorithmica 57,* 4, 653–667.

[15] Lipton, R. J., Markakis, E., and Mehta, A. 2003. Playing large games using simple strategies. In *ACM Conference on Electronic Commerce*. 36–41.

[16] Tsaknakis, H. and Spirakis, P. G. 2008. An optimization approach for approximate Nash equilibria. *Internet Mathematics 5,* 4, 365–382.

[17] Wellman, M. 2006. Methods for empirical game-theoretic analysis. In *Proc. of AAAI*. 1552–1555.