Constant Rank Bimatrix Games are PPAD-Hard

[Extended Abstract] *

Ruta Mehta†
College of Computing
Georgia Institute of Technology
Atlanta, USA
rmehta@cc.gatech.edu

ABSTRACT

The rank of a bimatrix game (A, B) is defined as $rank(A+B)$. Computing a Nash equilibrium (NE) of a rank-0, i.e., zero-sum game is equivalent to linear programming (von Neumann'28, Dantzig'51). In 2005, Kannan and Theobald gave an FPTAS for constant rank games, and asked if there exists a polynomial time algorithm to compute an exact NE. Adsul et. al. (2011) answered this question affirmatively for rank-1 games, leaving rank-2 and beyond unresolved.

In this paper we show that NE computation in games with rank ≥ 3, is PPAD-hard, settling a decade long open problem. Interestingly, this is the first instance that a problem with an FPTAS turns out to be PPAD-hard. Our reduction bypasses graphical games and game gadgets, and provides a simpler proof of PPAD-hardness for NE computation in bimatrix games. In addition, we get:

- An equivalence between $2D$-Linear-FIXP and PPAD, improving a result by Etessami and Yannakakis (2007) on equivalence between Linear-FIXP and PPAD.
- NE computation in a bimatrix game with convex set of Nash equilibria is as hard as solving a simple stochastic game [12].
- Computing a symmetric NE of a symmetric bimatrix game with rank ≥ 6 is PPAD-hard.
- Computing a $\frac{1}{poly(n)}$-approximate fixed-point of a (Linear-FIXP) piecewise-linear function is PPAD-hard.

The status of rank-2 games remains unresolved.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

*A full version of this paper is available at `arxiv.org/abs/1402.3350`

†Supported by NSF Grant CCF-1216019.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.
STOC '14, May 31 - June 03 2014, New York, NY USA
Copyright 2014 ACM 978-1-4503-2710-7/14/05 $15.00.
http://dx.doi.org/10.1145/2591796.2591835

General Terms

Theory

Keywords

Constant rank, PPAD-hard, 2D-Brouwer, Linear-FIXP, LCP

1. INTRODUCTION

Two player, finite, non-cooperative games constitute the most simple and fundamental model within game theory [25], and have been studied extensively for their computational and structural properties. Such a game can be represented by two payoff matrices (A, B), one for each player, and therefore are also known as bimatrix games. Von Neumann (1928) showed that in games where one player's loss is the other player's gain ($B = -A$, zero-sum), the min-max strategies are stable [32]. This turned out to be equivalent to linear programming (LP) [15, 2] and therefore polynomial-time computable. In 1950, John Nash [26] extended this notion to formulate an equilibrium concept, and proved its existence for finite multi-player games. It has since been named Nash equilibrium (NE) and is perhaps the most important and well-studied solution concept in game theory.

The classical Lemke-Howson algorithm (1964) [22], to compute Nash equilibrium in general bimatrix games, performs very well in practice. However it may take exponential time in the worst case [29]. Other methods that followed [21, 31] are also similar in nature [6, 19], and a complexity theoretic study of the problem was called for. Henceforth, by 2-Nash we mean computing a Nash equilibrium of a bimatrix game.

The complexity class NP is not applicable for 2-Nash, because an equilibrium is guaranteed to exist [26]. However, computing a special kind of NE, for numerous special properties, has been shown to be NP-complete [18, 14]. In 1994 Papadimitriou introduced complexity class PPAD [27], *Polynomial Parity Argument for Directed* graph, for problems with path following argument for existence, like Sperner's lemma [30]. He showed that 2-Nash, among many other problems, is in PPAD. After more than a decade, the problem was shown to be PPAD-hard in a remarkable series of works [16, 11]. Chen et. al. [11] showed that even $\frac{1}{poly(n)}$-approximation of 2-Nash is PPAD-hard, *i.e.*, if there is a fully polynomial-time approximation scheme (FPTAS) for 2-Nash then PPAD=P. This was followed by PPAD-hardness results for special classes of bimatrix games, like sparse games [10] and win-lose games [1], and their approximation were also shown to be PPAD-hard.

On the positive side, polynomial-time algorithms were developed for many special classes of games. Among these, one of the most significant is the class of constant rank games defined by Kannan and Theobald (2005) - rank of game (A, B) is defined as $rank(A+B)$. They gave an FPTAS for constant rank games,[1] and asked if there is an efficient algorithm to compute an exact NE in these games. Note that, rank-0 are zero-sum games, and therefore are polynomial-time solvable. For rank-1 games, Adsul et. al. [4] gave a polynomial time algorithm, by reducing the problem to 1-dimensional fixed-point, however rank-2 and beyond remained unresolved.

In this paper we show that NE computation in games with rank ≥ 3 is PPAD-hard, settling a decade long open problem. Since there is an FPTAS for constant rank games, this result comes as a surprise, because until now whenever a problem, in games or markets, was shown to be PPAD-hard, so was its approximation (i.e., no FPTAS unless PPAD=P) [11, 10, 1, 8, 20].

To obtain the result, we reduce $2D$-Brouwer, a two dimensional discrete fixed point problem which is known to be PPAD-hard [7], to a rank-3 game. The reduction is done in two steps. First we reduce $2D$-Brouwer to $2D$-Linear-FIXP; Linear-FIXP [17] is a class of fixed-point problems with polynomial piecewise-linear functions, and kD-Linear-FIXP is its subclass consisting of k-dimensional fixed-point problems. In the second step, we reduce an instance of $2D$-Linear-FIXP to a rank-3 bimatrix game, such that a linear function of Nash equilibrium strategies of the resulting game gives fixed-points of the $2D$-Linear-FIXP instance.

Our reduction completely bypasses the machinery of graphical games and game gadgets, central to the previous approaches, and instead exploits relations between LPs, linear complementarity problems (LCPs) and bimatrix games. Such a conceptual leap seems to be necessary to show hardness of constant rank games, since the game gadgets used previously inherently give rise to higher rank games. Our approach also provides a simpler proof for PPAD-hardness of 2-Nash, and may be of independent interest to show hardness for other problems, and to understand connections between parameterized LPs and bimatrix games. We can achieve further simplification by avoiding even the parameterized LP, but the resulting game turns out to be of high rank (see [23] for details).

Apart from the hardness of constant rank games, a number of results follow as corollaries from our reduction. The first step shows PPAD-hardness of $2D$-Linear-FIXP and thereby improves the equivalence result Linear-FIXP = PPAD of Etessami and Yannakakis to $2D$-Linear-FIXP = PPAD. This also implies $2D$-Linear-FIXP = Linear-FIXP; in other words the class of Linear-FIXP remains unchanged even when functions are restricted to two dimensions. Since, an instance of $1D$-Linear-FIXP can be solved in polynomial time using binary search, our result establishes a *dichotomy* between $1D$ and kD, $k \geq 2$ Linear-FIXP problems; the former are in P and the latter are PPAD-complete.

Our approach can be extended to reduce kD-Brouwer to kD-Linear-FIXP to rank-$(k + 1)$ games, where the reduction from kD-Linear-FIXP to rank-$(k + 1)$ games (almost) preserves the number of solutions. Using this, together with a result from [17], we show that bimatrix games with convex set of NE are no easier. In fact they are as hard as solving simple stochastic games, which are known to be in NP $\cap$ coNP [12], however despite significant efforts its exact complexity remains open [13, 5]. Further, we can show that computing weak[2] $\frac{1}{poly(n)}$-approximate fixed-point of a function in Linear-FIXP is also PPAD-hard (see [23] for details). It will be interesting to see if this can be extended to show hardness of approximation in 2-Nash.

Since NE computation in a rank-k game can be reduced to computing symmetric NE of a symmetric game with rank-$2k$ [28], we get that computing symmetric Nash equilibria in symmetric games with rank ≥ 6 is PPAD-hard. Again computing symmetric NE in symmetric rank-0 games can be solved using LP, and for rank-1 games recently Mehta et. al. [24] gave a polynomial-time algorithm. This leaves the status of symmetric games with rank between 2 and 5 unresolved. Also the status of rank-2 bimatrix games remains unresolved.

1.1 Overview of the Reduction

In this section we explain the main ideas behind the reductions: from $2D$-Brouwer to $2D$-Linear-FIXP, and then to rank-3 game. We start with a brief description of $2D$-Brouwer and Linear-FIXP problems.

$2D$-Brouwer is a class of 2-dimensional discrete fixed-point problems, known to be PPAD-hard [9]. An instance of $2D$-Brouwer consists of a grid $G_n = \{0, \dots, 2^n-1\} \times \{0, \dots, 2^n-1\}$ and a valid coloring function $g : G_n \rightarrow \{0, 1, 2\}$ which satisfies some boundary conditions, and thereby ensures existence of a trichromatic unit square in the grid.[3] The problem is to find one such trichromatic square (see Section 2.2 for details). Function g is specified by a Boolean circuit C^b with $2n$ input bits; n bits to represent each of the two co-ordinate of a grid point.[4]

Linear-FIXP [17] is a class of fixed-point problems with polynomial piecewise-linear functions. A function $F : [0,\ 1]^n \rightarrow [0,\ 1]^n$ in Linear-FIXP is defined by a circuit, say C, with n real inputs and outputs, and $\{\max, +, *\zeta\}$ operations, where $*\zeta$ is multiplication by a rational constant (see Section 2.3 for details). Such a function has rational fixed-points of size polynomial in the input size [17]. We denote the class of k-dimensional fixed-point problems in Linear-FIXP by kD-Linear-FIXP.

Given circuit C^b of a $2D$-Brouwer instance, in Section 3 we construct a $2D$-Linear-FIXP circuit C such that all the fixed-points of the function F defined by C are in trichromatic unit squares of the grid G_n. It is easy to simulate C^b in C by replacing $\wedge$, $\vee$ and $\neg$ with $\min, \max$ and $(1 - x)$ respectively, if input to this simulation is guaranteed to be Boolean. To guarantee this, we need to extract bit representation of $\lfloor \boldsymbol{p} \rfloor$, for a $\boldsymbol{p} \in [0,\ 2^n - 1]^2$. Since, *floor* is a discontinuous function it can not be simulated using Linear-FIXP circuit, whose operations can generate only continuous functions. However, we design a bit extraction gadget which does the job for almost all the points, except those that are close to the boundary of unit squares of G_n. Finally, using a sampling lemma similar to that of [11] we ensure that the fixed-points of the function defined by the resulting circuit

[1]$O(L/\epsilon)^k poly(n)$ time algorithm to compute an ϵ-approximate Nash equilibrium in a rank-k $n \times n$ game of bit size L.

[2]Vector $\boldsymbol{x}$ is a *weak* ϵ-approximate fixed-point of function f if $\|\boldsymbol{x} - f(\boldsymbol{x})\|_\infty \leq \epsilon$

[3]This is similar to the Sperner's lemma

[4]We use super-script b to differentiate Boolean circuits from Linear-FIXP circuits that will follow.

C are always in trichromatic unit squares of the grid G_n, and we get,

THEOREM 1 (INFORMAL). *Computing a fixed-point of a Linear-FIXP instance with k inputs and k outputs, with $k > 1$, is PPAD-hard. In other words, $2D$-Linear-FIXP=PPAD.*

Etessami and Yannakakis [17] showed that Linear-FIXP = PPAD. Theorem 2 improves this to $2D$-Linear-FIXP = PPAD, and in turn we get Linear-FIXP = $2D$-Linear-FIXP, *i.e.,* fixed-point problems with polynomial piecewise-linear functions in constant (two) dimension are as hard as those in n-dimension.

Next, we reduce the fixed-point computation of a kD-Linear-FIXP instance to Nash equilibrium computation in a rank-$(k+1)$ game (see Section 4). Let $\boldsymbol{\lambda} = (\lambda_1, \dots, \lambda_k)$ denote the k inputs of circuit C of the given kD-Linear-FIXP instance. First we replace circuit C by a parameterized linear program $LP(\boldsymbol{\lambda})$, so that circuit evaluation for a given input is same as solving the LP.

This is done as follows: There is an ordering among max gates since C forms a DAG. Suppose x_i captures the output of the i^{th} max gate. Since the $+$ and $*\zeta$ operations of circuit C generates only linear expressions, for $x_j = \max\{L, R\}$, L and R both are linear expression in $x_1, \dots, x_{j-1}$ and $\boldsymbol{\lambda}$. Further, this max operation is equivalent to $x_j \geq L, x_j \geq R, (x_j - L)(x_j - R) = 0$. The first two linear conditions define the feasible region of LP, where x_js are variables and λ_is are parameters. Note that, r.h.s. of the constraints of LP is parameterized by $(\lambda_1, \dots, \lambda_k)$, and the constraint matrix is lower triangular. Using this property we show that $\exists \boldsymbol{c}$ such that for all $\boldsymbol{\lambda}$, min : $\boldsymbol{c}^T\boldsymbol{x}$ over this feasible region will ensure the quadratic constraints as well for each max gate. This gives the $LP(\boldsymbol{\lambda})$ which can replace circuit C.

Since, primal-dual feasibility, and complementary slackness characterizes solutions of an LP, LP is a special case of linear complementarity problem (LCP). Using this connection for $LP(\boldsymbol{\lambda})$, we construct an LCP$_C$ whose solutions exactly capture the fixed point of the given kD-Linear-FIXP instance (Section 4.2). Further, the matrix of the LCP turns out to be off-block-diagonal, with the two blocks in off-diagonal adding up to a rank-k matrix. Finally, using the fact that the LCP capturing Nash equilibria of a bimatrix game also has a off-block-diagonal matrix, we construct a bimatrix game, whose Nash equilibria are in one-to-one correspondence with the solutions of LCP$_C$. The rank of the resulting game turns out to be $(k+1)$, and one of its payoff matrix is upper-triangular.

THEOREM 2 (INFORMAL). *Nash equilibrium computation in bimatrix games with rank ≥ 3 is PPAD-hard, even when one of the payoff matrix is lower/upper triangular.*

Theorem 2, together with the reduction from 2-Nash to symmetric2-Nash [28], implies that computing symmetric NE of a symmetric game with rank ≥ 6 is PPAD-hard.

2. PRELIMINARIES

To show the hardness of rank-3 games, we start with $2D$-Brouwer, reduce it to Linear-FIXP and then to a bimatrix game. In this section we discuss each of these problems separately. First we describe a characterization of Nash equilibria in bimatrix games, and the class of $2D$-Brouwer problems. Both the problems are known to be PPAD-complete [11, 9]. Next, we describe Linear-FIXP [17], and define a subclass called kD-Linear-FIXP.

Notations: All the vectors are in bold-face letters, and are considered as column vectors. To denote a row vector we use $\boldsymbol{x}^T$. The i^{th} coordinate of the vector $\boldsymbol{x}$ is denoted by x_i. $\boldsymbol{1}$ and $\boldsymbol{0}$ represent all ones and all zeros vectors. We use $[n]$ to denote the set $\{1, \dots, n\}$.

2.1 Bimatrix games and Nash equilibrium

A bimatrix game is a two player game, where each player has finitely many pure strategies (moves). Let $S_i,\ i = 1, 2$ be the set of strategies of player i, and let $m \overset{\text{def}}{=} |S_1|$ and $n \overset{\text{def}}{=} |S_2|$. Then such a game can be represented by two payoff matrices A and B, each of $m \times n$ dimension.

Players may randomize among their strategies; a randomized play is called a *mixed strategy.* Let the set of mixed strategies of the first player be $X = \{\boldsymbol{x} = (x_1, \dots, x_m) \mid \boldsymbol{x} \geq 0, \sum_{i=1}^m x_i = 1\}$, and that of the second player be $Y = \{\boldsymbol{y} = (y_1, \dots, y_n) \mid \boldsymbol{y} \geq 0, \sum_{j=1}^n y_j = 1\}$. At profile $(\boldsymbol{x}, \boldsymbol{y}) \in X \times Y$, the expected payoffs of the first-player and second-player are, respectively

$$\sum_{i,j} A_{ij} x_i y_j = \boldsymbol{x}^T A \boldsymbol{y} \qquad \text{and} \qquad \sum_{i,j} B_{ij} x_i y_j = \boldsymbol{x}^T B \boldsymbol{y}$$

DEFINITION 3. *(Nash Equilibrium [33]) A strategy profile $(\boldsymbol{x}, \boldsymbol{y}) \in X \times Y$ is said to be a at Nash equilibrium iff $\forall \boldsymbol{x}' \in X$, $\boldsymbol{x}^T A \boldsymbol{y} \geq \boldsymbol{x}'^T A \boldsymbol{y}$ and $\forall \boldsymbol{y}' \in Y$, $\boldsymbol{x}^T B \boldsymbol{y} \geq \boldsymbol{x}^T B \boldsymbol{y}'$.*

It is easy to obtain the following characterization for NE.

$$\begin{array}{lll} \forall i \in S_1,\ x_i > 0 & \Rightarrow & (A\boldsymbol{y})_i = \max_{k \in S_1} (A\boldsymbol{y})_k \\ \forall j \in S_2,\ y_j > 0 & \Rightarrow & (\boldsymbol{x}^T B)_j = \max_{k \in S_2} (\boldsymbol{x}^T B)_k \end{array}$$

The next lemma follows from the above discussion.

LEMMA 4. *Strategy profile $(\boldsymbol{x}, \boldsymbol{y}) \in X \times Y$ is a NE of game (A, B) if and only if the following holds, where π_1 and π_2 are scalars capturing respective payoffs at $(\boldsymbol{x}, \boldsymbol{y})$.*

$$\begin{array}{ll} \forall i \in S_1, (A\boldsymbol{y})_i \leq \pi_1; & x_i((A\boldsymbol{y})_i - \pi_1) = 0 \\ \forall j \in S_2, (\boldsymbol{x}^T B)_j \leq \pi_2; & y_j((\boldsymbol{x}^T B)_j - \pi_2) = 0 \end{array}$$

2.2 2D-Brouwer

Let G_n denote the two dimensional grid $\{0, \dots, 2^n - 1\} \times \{0, \dots, 2^n - 1\}$. A 3-coloring of G_n is a function g from the vertices of G_n to $\{0, 1, 2\}$. Function g is said to be valid if for every vertex (p_1, p_2) on the boundary of G_n, we have

$$\begin{array}{l} \text{If } \ p_2 = 0 \ \text{ then } \ g(\boldsymbol{p}) = 2, \\ \quad \text{else if } \ p_2 > 0 \ \&\ p_1 = 0 \ \text{ then } g(\boldsymbol{p}) = 1, \quad \text{else } g(\boldsymbol{p}) = 0 \end{array}$$

Let $K\boldsymbol{p}$ denote the unit square with $\boldsymbol{p}$ at the bottom left corner. Due to Sperner's Lemma it is known that for any valid coloring of g of G_n there exists a vertex $\boldsymbol{p} \in G_n$ such that vertices of $K\boldsymbol{p}$ have all the three colors - trichromatic.

2D-Brouwer Mapping Circuit: Consider a Boolean circuit C^b generating valid coloring on grid G_n.[5] The circuit has $2n$ input bits, n bits for each of the two integers representing a grid point, and 4 output bits $\Delta_1^+, \Delta_1^-, \Delta_2^+, \Delta_2^-$. It is a *valid Brouwer-mapping circuit* if the following is true:

[5]We use the definitions and terminology of [11] to remain consistent.

- For every $\boldsymbol{p} \in G_n$, the 4 output bits of C^b satisfies one of the following 3 cases:
  - Case 0: $i = 1, 2$, $\Delta_i^- = 1$ and $\Delta_i^+ = 0$.
  - Case i, $i = 1, 2$: $\Delta_i^+ = 1$ and all the other 3 bits are zero.
- For every $\boldsymbol{p}$ on the boundary of G_n, if $p_2 = 0$ then Case 2 is satisfied, if $p_1 = 0$ and $p_2 \neq 0$ then Case 1 is satisfied, and for the rest Case 0 is satisfied.

Such a circuit C^b defines a valid color assignment $g_{C^b} : G_n \to \{0, 1, 2\}$ by setting $g_{C^b}(\boldsymbol{p}) = i$, if the output bits of C^b evaluated at $\boldsymbol{p}$ satisfy Case i.

DEFINITION 5 (2D-BROUWER [9]). *The input to the 2D-Brouwer consists of a valid Brouwer-mapping circuit C^b that produces a valid coloring on G_n. The problem is to find a point $\boldsymbol{p} \in G_k$ such that $K\boldsymbol{p}$ is trichromatic.*

The size of the given 2D-Brouwer problem is $size[C^b]$, which is #input nodes + #output nodes + # gates.

The outputs of the circuit (defining a color), can also be mapped to incremental vector $(\Delta_1^+ - \Delta_1^-, \Delta_2^+ - \Delta_2^-)$. Let $\boldsymbol{e}^i$ be the incremental vector corresponding to Case (color) i, then clearly, $\boldsymbol{e}^0 = (-1, -1), \boldsymbol{e}^1 = (1, 0)$ and $\boldsymbol{e}^2 = (0, 1)$. Define a discrete function H, such that $H(\boldsymbol{p}) = \boldsymbol{p} + \boldsymbol{e}^{g_{C^b}(\boldsymbol{p})}$. It is easy to see that if C^b is a valid Brouwer-mapping circuit, then H is $G_n \to G_n$, and vertices of a trichromatic square $K\boldsymbol{p}$ goes in each of the $\boldsymbol{e}^i$ direction under H. Chen and Deng showed finding such a square is PPAD-hard [9].

2.3 Linear-FIXP

Etessami and Yannakakis [17] defined the class FIXP to capture complexity of the exact fixed point problems with algebraic solutions. An instance I of FIXP consists of an algebraic circuit C_I defining a function $F_I : [0,\ 1]^d \to [0,\ 1]^d$, and the problem is to compute a fixed-point of F_I. The circuit is a finite representation of function F_I (like a formula), consisting of $\{\max, +, *\}$ operations, and rational constants.

The circuit C_I is a sequence of gates $g_1, \dots, g_m$, where for $i \in [d]$, $g_i := \lambda_i$ is an input variable. For $d < i \leq d + r$, $g_i := c_i \in \mathbb{Q}$ is a rational constant, with numerator and denominator encoded in binary. For $i > d + r$ we have $g_i = g_j \circ g_k$, where $j, k < i$ and the binary operator $\circ \in \{\max, +, *\}$. The last d gates are the output gates. Note that the circuit forms a directed acyclic graph (DAG), when gates are considered as nodes, and there is an edge from g_j and g_k to g_i if $g_i = g_j \circ g_k$. Since, circuit C_I represents function F_I, for an input $\boldsymbol{\lambda} \in [0,\ 1]^d$, all the gates of C_I are well defined and $C_I(\boldsymbol{\lambda}) = F_I(\boldsymbol{\lambda}) \in [0,\ 1]^d$. We note that a circuit representing a problem in FIXP operates on real numbers, but the underlying model of computation is still the standard discrete Turing machine. In other words, an algorithm for FIXP problems is not allowed to do computation on reals.

Let $*\zeta$ denote multiplication by a rational constant $\zeta \in \mathbb{Q}$. The Linear-FIXP is a subclass of FIXP where the operations are restricted to $\circ \in \{max, +, *\zeta\}$. A function defined by a Linear-FIXP circuit is polynomial piecewise-linear, and all its fixed points are rational numbers of size $poly(L)$ [17], where L is the total size of the circuit which is #inputs + #gates + total size of the constants used in the circuit. Etessami and Yannakakis showed that PPAD = Linear-FIXP. Next, we define a subclass of Linear-FIXP based on the number of inputs and outputs.

DEFINITION 6. *For a $k \geq 1$, an instance I is in kD-Linear-FIXP if $F_I : [0,\ 1]^k \to [0,\ 1]^k$. i.e., F_I is defined by a circuit with k inputs and k outputs.*

Since fixed-point of a 1-dimensional piecewise-linear function can be computed in polynomial time using a binary search, kD-Linear-FIXP is in P for $k = 1$. But for any constant $k > 1$ it is not clear if the problem is in P or it is hard. In the next section, we show that the problem is PPAD-hard even for $k = 2$.

3. PPAD-HARDNESS OF 2D-LINEAR-FIXP

In this section we describe the construction of a Linear-FIXP circuit with two inputs and two outputs, from an instance of $2D$-Brouwer defined by a Boolean Brouwer-mapping circuit. We show that the function defined by the resulting $2D$-Linear-FIXP circuit is such that all its fixed-points are in trichromatic squares of the $2D$-Brouwer instance, thereby proving PPAD-hardness of $2D$-Liner-FIXP using [9].

Let C^b be the valid Brouwer-mapping circuit of a given 2D-Brouwer instance on grid G_n, and H be the discrete function defined by circuit C^b. We construct a Linear-FIXP circuit C that computes a function $F : [0,\ 2^n - 1]^2 \to [0,\ 2^n - 1]^2$, an extension of the discrete function H.

Recall that given a bit representation of a grid point $\boldsymbol{p} \in G_n$, circuit C^b outputs four bits $\Delta_1^+, \Delta_1^-, \Delta_2^+, \Delta_2^-$, so that for $I = (\Delta_1^+ - \Delta_1^-, \Delta_2^+ - \Delta_2^-)$, $H(\boldsymbol{p}) = \boldsymbol{p} + I$. Similarly, for every non-grid point $\boldsymbol{p} = (p_1, p_2) \in K\boldsymbol{q}$, we need to compute an incremental vector based on the incremental vectors of the vertices of $K\boldsymbol{q}$. For this we need to extract the integer parts of p_1 and p_2, i.e., compute $\lfloor p_1 \rfloor$ and $\lfloor p_2 \rfloor$, and then its bit representation. Since, *floor* is a discontinuous function, it can not be computed using Linear-FIXP operations, which are inherently continuous. However, next we achieve this for the points not very near to the boundary of any cell.

Recall that the operations allowed in a Linear-FIXP circuit are $\{max, +, *\zeta\}$. Clearly, $\{min, -\}$ can be simulated using the allowed operations. Let $L > 16$ be a large integer with value being a power of 2, and at most polynomial in $size[C^b]$, *i.e.*, $L = 2^l \leq poly(size[C^b])$. Consider the *ExtractBits* procedure of Table 1.

```
ExtractBits(a)
$x \leftarrow a$
**for** i=n-1 **to** 0 **do**
    $b_i \leftarrow \min\{\max\{((x - 2^i) * L^2) + 1, 0\}, 1\}$
    $x \leftarrow x - 2^i b_i$
**endfor**
Output the bit vector $\boldsymbol{b} = (b_{n-1}, \dots, b_0)$.
```

Table 1: Extract Bits of the Integer Part

DEFINITION 7. *We say that $a \in \mathbb{R}_+$ is* poorly positioned *if for some integer $t \in \mathbb{Z}_+$, $a = t + \epsilon$, where $1 - \frac{1}{L^2} < \epsilon < 1$. A point $\boldsymbol{p} \in \mathbb{R}_+^2$ is said to be* poorly-positioned*, if any of its coordinates is poorly positioned, otherwise it is called* well-positioned.

LEMMA 8. *Given a well-positioned number $a \in [0,\ 2^n)$, the vector $\boldsymbol{b} = ExtractBits(a)$ is a bit representation of $\lfloor a \rfloor$.*

PROOF. Let $a = a' + \epsilon$, where $a' \in \mathbb{Z}_+$ and $0 \leq \epsilon \leq 1 - \frac{1}{L^2}$. We show that every b_i is either 0 or 1, and is set correctly. Proof is by induction. If $a' \geq 2^{n-1}$, then clearly, $(a - 2^{n-1}) * L^2 + 1 \geq 1$ and b_{n-1} will be one. If $a' < 2^{n-1}$, then $(a - 2^{n-1}) * L^2 + 1 \leq (-1 + \epsilon) * L^2 + 1 \leq (-1 + 1 - \frac{1}{L^2})L^2 + 1 \leq 0$ and hence b_{n-1} will be zero. In either case $x = a - b_{n-1}2^{n-1}$ will satisfy the hypothesis, and we can apply the same argument for bit b_{n-2}. $\square$

Given a well positioned point $\boldsymbol{p} \in K\boldsymbol{q}$, we can extract bit representations of each of the coordinates of $\boldsymbol{q}$ due to Lemma 8, and hence of all the vertices of $K\boldsymbol{q}$. Next task is to obtain each of their incremental vectors by simulating circuit C^b in Linear-FIXP. Circuit C^b is a Boolean circuit with operations $\wedge, \vee$ and $\neg$ and takes only Boolean input. These operations are easy to simulate in Linear-FIXP: If $a, b \in \{0, 1\}$, then clearly $a \wedge b = \min\{a, b\}$, $a \vee b = \max\{a, b\}$ and $\neg a = (1 - a)$.

Thus, if $\boldsymbol{p}$ is well positioned, then incremental vectors of the vertices of $K\boldsymbol{q}$ can be computed using a Linear-FIXP circuit. However, if $\boldsymbol{p}$ is poorly-positioned, then Lemma 8 provides no guarantees and indeed the *ExtractBits* procedure may produce vector $\boldsymbol{b}$ with the value b_is being anything in [0, 1]. This is expected due to continuity property of Linear-FIXP operations. Similar difficulty arises in the approaches of Daskalakis et al. [16] and Chen et al. [11]. Both resort to a sampling argument, first proposed in [16], and later improved in [11]. Next we describe a version of [11] argument.

Given a set of points $S = \{\boldsymbol{p}^1, \ldots, \boldsymbol{p}^l\}$, let $I_w(S)$ and $I_p(S)$ denote the set of indices of the well and poorly positioned points of S respectively. Given $\boldsymbol{p} \in \mathbb{R}^2_+$, let $\pi(\boldsymbol{p}) = \{\boldsymbol{q} \mid q_1, q_2$ are the largest integers from $\{0, \ldots, 2^n - 1\}$ s.t. $q_1 \leq p_1$ and $q_2 \leq p_2\}$. For $\boldsymbol{e}^1 = (1, 0), \boldsymbol{e}^2 = (0, 1)$ and $\boldsymbol{e}^0 = (-1, -1)$, let $\zeta(\boldsymbol{p}) = \boldsymbol{e}^i$, where $i = g_{C^b}(\pi(\boldsymbol{p}))$.

LEMMA 9. *Given $\boldsymbol{p} \in [0,\ 2^n - 1]^2$, consider the set $S = \{\boldsymbol{p}^1, \ldots, \boldsymbol{p}^{16}\}$ such that*

$$\boldsymbol{p}^j = \boldsymbol{p} + (j - 1)(\frac{1}{L}, \frac{1}{L}), \qquad j \in [16]$$

For each $j \in I_p(S)$, let $\boldsymbol{r}^j \in \mathbb{R}^2$ be a vector with $\|\boldsymbol{r}^j\|_\infty \leq 1$. And for each $j \in I_w(S)$, let $\boldsymbol{r}^j = \zeta(\boldsymbol{p}^j)$. If $\|\sum_{j=1}^{16} \boldsymbol{r}^j\|_\infty = 0$ then $K_{\pi(\boldsymbol{p})}$ is trichromatic.

REMARK 10. *Note that, in Lemma 9, it suffices to assume $\|\sum_{j=1}^{16} \boldsymbol{r}^j\|_\infty < 1$ for $\boldsymbol{p}$ to be in a trichromatic square. We use this fact to derive inapproximability results in [23].*

Lemma 9 implies that even if point $\boldsymbol{p}$ is poorly positioned, we can make sure that it forms a fixed-point only when it is in a trichromatic square by sampling 16 carefully chosen points near it. Next we describe a complete construction of the Linear-FIXP circuit C, and then show its correctness using Lemmas 8 and 9.

S_1. Let p_1 and p_2 denote the two inputs of the Linear-FIXP circuit. These are any real number from $[0,\ 2^n - 1]$. Compute 16 points using the + gates and rational constants:

$$\boldsymbol{p}^i = \boldsymbol{p} + (j - 1)(\frac{1}{L}, \frac{1}{L}), \qquad j \in [16]$$

S_2. Call ExtractBits($\boldsymbol{p}^j_t$), $t = 1, 2$ and $j \in [16]$, and let the output vector be $\boldsymbol{b}^{j,t}$.

S_3. For each $j \leq [16]$, feed $b_0^{j,1}, \ldots, b_{n-1}^{j,1}, b_0^{j,2}, \ldots b_{n-1}^{j,2}$ to a simulation of circuit C^b, where $\vee$, $\wedge$ and $\neg x$ are replaced with max, min and $1 - x$ respectively. Note that, there are total of 16 simulations of circuit C^b. Let $\Delta_1^{j+}, \Delta_1^{j-}, \Delta_2^{j+}, \Delta_2^{j-}$ be the output values of these.

S_4. For each $j \in [16]$, compute $r_1^j = \min\{\max\{\Delta_1^{j+} - \Delta_1^{j-}, -1\}, 1\}$ and $r_2^j = \min\{\max\{\Delta_2^{j+} - \Delta_2^{j-}, -1\}, 1\}$.

S_5. Compute $r_1 = \frac{1}{16}\sum_{j \in [16]} r_1^j$, and $r_2 = \frac{1}{16}\sum_{j \in [16]} r_2^j$.

S_6. Output $p_1' = \max\{\min\{p_1 + r_1, 2^n - 1\}, 0\}$ and $p_2' = \max\{\min\{p_2 + r_2, 2^n - 1\}, 0\}$.

The number of gates used in steps S_1, S_4, S_5 and S_6 of the above procedure are constant. We used $O(n)$ gates in step S_2, and 16 times as many as the number of gates in C^b in step S_3. Further, since value of L is polynomial in $size[C^b]$, the constants used in steps S_1, S_2 and S_5 are polynomial sized. Thus, the total size of the Linear-FIXP circuit C constructed by the above procedure is polynomial in $size[C^b]$. Next we show that each of the fixed-points of function F represented by circuit C are in trichromatic squares of the grid G_n.

LEMMA 11. *Every fixed point of F is inside a trichromatic square of G_n.*

It is easy to shrink the range of F from $[0,\ 2^n - 1]$ to $[0,\ 1]$. Consider a function $F' : [0,\ 1]^2 \to [0,\ 1]^2$, such that $F'(\lambda_1, \lambda_2) = \frac{1}{2^n - 1} F((2^n - 1)\lambda_1, (2^n - 1)\lambda_2)$, then clearly, (λ_1, λ_2) is a fixed-point of F' if and only if $((2^n - 1)\lambda_1, (2^n - 1)\lambda_2))$ is a fixed-point of F. Thus we get the following theorem using Lemma 11 and the fact that $size[C] = poly(size[C^b])$.

THEOREM 12. *The class of kD-Linear-FIXP with $k > 1$ is PPAD-hard.*

4. REDUCTION: KD-LINEAR-FIXP TO RANK-(K+1) GAME

Given a kD-Linear-FIXP instance, with function $F : [0,\ 1]^k \to [0,\ 1]^k$ represented by circuit C, in this section we construct a rank-$(k + 1)$ bimatrix game whose Nash equilibria are almost[6] in one-to-one correspondence with the fixed points of F. We do this in two steps. First we replace the circuit C, by a parametric linear program (LP) with k-parameters, where inputs of circuit C become parameters of the LP. Given values of the k inputs, we show that the k outputs of the circuit C are linear function of a solution of the LP. This defines a function F^{lp} from $\mathbb{R}^k$ to $\mathbb{R}^k$, and we show that the fixed points of F^{lp} are in one to one correspondence with the fixed-points of F. Later, we construct a rank-$(k + 1)$ game using the LP and its dual, whose Nash equilibria exactly captures the fixed points of F^{lp}.

REMARK 13. *Recall that linear programs are equivalent to zero-sum games [15, 2]. However, the reductions from LP to zero-sum games constructs a symmetric game, and require to compute a symmetric Nash equilibrium. There are no such restrictions in our construction, however our reduction is not general enough and uses the fact that the parametric LP has been constructed from a Linear-FIXP circuit. It will be interesting to reduce an LP to a non-symmetric zero-sum game, and also a fixed-point problem with parametric LP to a constant rank game in general.*

[6]Essentially, Nash equilibrium strategies of the first player are in one-to-one correspondence with the fixed points

4.1 Replacing Linear-FIXP circuit with a linear program

In this section we construct a parameterized linear program with k parameters, which can replace a kD-Linear-FIXP circuit. Let C be a kD-Linear-FIXP circuit representing the function $F : [0,\ 1]^k \rightarrow [0,\ 1]^k$. Circuit C being a Linear-FIXP circuit, it allows only three operations, namely $\max, +$ and $*\zeta$ where ζ is a rational number, and it forms a DAG. The $size[C]$ is # inputs +# gates +# total bit lengths of the constants in the circuit.

If C is considered as a function from $\mathbb{R}^k$ to $\mathbb{R}^k$, then it is same as function F on $[0,\ 1]^k$, but can be anything outside this range and hence may have fixed-points outside $[0,\ 1]^k$ as well. To prevent this, we add two max gates for every output of the circuit, as follows: Let $\tau_1, \ldots, \tau_k$ be the k outputs of circuit C. Without loss of generality (wlog), we will add two max gates for each $l \in [k]$ to ensure that each output value is in $[0,\ 1]$:

$$\max\{0, \min\{1, \tau_l\}\} = \max\{0, -1 * \max\{-1, -1 * \tau_l\}\} \quad (1)$$

The above transformation ensures that the output vector of C is always in $[0,\ 1]^k$, and hence fixed-points of C are exactly the fixed-points of F. Next, we show that it is wlog to assume that one of the inputs of every max gate is zero.

Lemma 14. *Given a circuit C, it can be transformed to an equivalent polynomial sized circuit where one of the inputs of every* max *gate is zero.*

Proof. Consider a max gate, and let a and b be the inputs and c be the output, then we have $c = \max\{a, b\}$ which is equivalent to $c = \max\{0, b - a\} + a$. Therefore, we can transform circuit C such that one input of every max gate is 0. This transformation requires 3 extra gates per max gate, two $+$ and one $*\zeta$ where $\zeta = -1$. Clearly, the increase in the size of the circuit is polynomial. $\square$

Wlog we assume that every max gate of circuit C has exactly one non-trivial input, and the other input is always zero (due to Lemma 14). Let m be the number of max gates in C. Since C is a DAG, there is an ordering among the max gates, say $g_1, \ldots, g_m$, such that if there is a path from g_i to g_j in C then $i < j$; ties are broken arbitrarily. Let $n = m - 2k$ be the number of max gate in the original circuit, before the addition of (1) per output. Let these be the first n max gates. In (1) let g_{n+2l-1} denote the inner max gate and g_{n+2l} denote the outer one, then k outputs of circuit are the outputs of gates $g_{n+2l},\ l \in [k]$.

Let the k inputs of circuit C be denoted by $\boldsymbol{\lambda} = (\lambda_1, \ldots \lambda_k)$, and let x_i capture the output of the i^{th} max gate. Note that, $x_{n+2l},\ l \in [k]$ are the output of the circuit. Except for the max, rest of the two operations give rise to linear expressions in the $\boldsymbol{\lambda}$ and x_is of the previous max gates. We use this observation crucially in the rest of the construction.

Note that, for each $i \in [m]$, the input of g_i is a linear expression in $x_1, \ldots, x_{i-1}, \lambda_1, \ldots, \lambda_k$, with a constant term. We denote this expression by $L_i(x_1, \ldots, x_{i-1}, \boldsymbol{\lambda})$, then the following conditions exactly capture the operation of g_i.

$$\forall i \in [m], \quad x_i \geq 0, \qquad x_i \geq L_i(x_1, \ldots, x_{i-1}, \boldsymbol{\lambda}) \qquad (2)$$

$$\forall i \in [m], \quad x_i(x_i - L_i(x_1, \ldots, x_{i-1}, \boldsymbol{\lambda})) = 0 \qquad (3)$$

The next lemma follows by construction.

Lemma 15. *Given $\boldsymbol{\lambda} \in \mathbb{R}^k$, $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies (2) and (3) iff when $\boldsymbol{\lambda}$ is given as the input to circuit C, the i^{th}* max *gate evaluates to x_i for all $i \in [m]$.*

Proof. Reverse direction follows just by construction. For the forward direction we will argue by induction. Suppose, $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies (2) and (3). Then, for $\boldsymbol{\lambda}$ as input to C, clearly L_1 evaluates to exactly the input of the (first max) gate g_1. In that case, (2) forces that x_1 is at least as large as inputs of g_1, and (3) forces that it equals one of the input. Thus, x_1 captures output of g_1. Now, suppose this is true for first $k \geq 1$ max gates. Then for $(k+1)^{th}$ max gate, again L_{k+1} is exactly the input of g_{k+1}, and the lemma follows by the same argument. $\square$

Constraints of (2) gives a system of linear inequalities,

$$A\boldsymbol{x} \geq \sum_{l=1}^{k} \lambda_l \boldsymbol{u}^l + \boldsymbol{b}, \qquad \boldsymbol{x} \geq 0 \qquad (4)$$

where, $\boldsymbol{b}$ and $\boldsymbol{u}^l,\ l \in [k]$ are m-dimensional rational vectors, and A is an $m \times m$ lower-triangular rational matrix with ones on the diagonal. Once we plugin some values for $\lambda_1, \ldots, \lambda_k$, (4) becomes a polyhedron in $\boldsymbol{x}$. Let it be denoted by $\mathcal{P}(\boldsymbol{\lambda})$. For any $\boldsymbol{\lambda} \in \mathbb{R}^k$ and $\boldsymbol{x} \in \mathcal{P}(\boldsymbol{\lambda})$, vector $(\boldsymbol{x}, \boldsymbol{\lambda})$, satisfies (2).

Remark 16. *Enforcing (3) requires quadratic complementarity-type constraints. Using this fact, in [23] we give a simplified proof of PPAD-harness of* 2*-Nash and symmetric* 2*-Nash (bypassing even the parameterized LPs).*

Next, we construct a cost vector $\boldsymbol{c} \in \mathbb{R}^m$, such that minimizing $\boldsymbol{x}^T \cdot \boldsymbol{c}$ over $\mathcal{P}(\boldsymbol{\lambda})$ will give a solution that, together with $\boldsymbol{\lambda}$, satisfies (3) as well.

```
ConstructCost($A$)
$c_m \leftarrow 1$, $\beta_m \leftarrow 1$
for $i = m - 1$ to 1 do
    $c_i \leftarrow \sum_{j>i} |a_{ji}|\beta_j + 1$,    $\beta_i \leftarrow c_i + \sum_{j>i} |a_{ji}|\beta_j$
endfor;
Output $\boldsymbol{c}$
```

Table 2: Construction of the Cost Vector

For $\boldsymbol{c}$ =ConstructCost(A), consider the following parameterized LP and its dual.

$$\begin{array}{llll} LP(\boldsymbol{\lambda}) & & DLP(\boldsymbol{\lambda}) & \\ \min: & \boldsymbol{c}^T \cdot \boldsymbol{x} & \max: & (\sum_{l\in[k]} \lambda_l \boldsymbol{u}^l + \boldsymbol{b})^T \boldsymbol{y} \\ s.t., & A\boldsymbol{x} \geq \sum_{l\in[k]} \lambda_l \boldsymbol{u}^l + \boldsymbol{b} & s.t., & A^T \boldsymbol{y} \leq \boldsymbol{c} \\ & \boldsymbol{x} \geq 0 & & \boldsymbol{y} \geq 0 \end{array} \qquad (5)$$

The complementary slackness requires that solutions of $LP(\boldsymbol{\lambda})$ and $DLP(\boldsymbol{\lambda})$ satisfy (KKT conditions),

$$\forall i \in [m],\ y_i(A\boldsymbol{x} - \sum_{l\in[k]} \lambda_l \boldsymbol{u}^l - \boldsymbol{b})_i = 0,\ x_i(A^T\boldsymbol{y} - \boldsymbol{c})_i = 0 \quad (6)$$

Lemma 17. *Given $\boldsymbol{\lambda} \in \mathbb{R}^k$, $\boldsymbol{x}$ is a solution of $LP(\boldsymbol{\lambda})$ iff $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies (2) and (3).*

Proof. ($\Rightarrow$) Let $\boldsymbol{y}$ be the dual solution corresponding to $\boldsymbol{x}$, i.e., $(\boldsymbol{x}, \boldsymbol{y})$ satisfies (6). Since $\boldsymbol{x}$ is a feasible point of $LP(\boldsymbol{\lambda})$, clearly, $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies (2). For (3), it suffices to show that $\forall i \in [m], x_i > 0 \Rightarrow y_i > 0$, then the proof follows using (6).

Let $\boldsymbol{\beta} \in \mathbb{R}^m$ be the vector calculated in ConstructCost(A) of Table 2. We do the proof by induction, where we show that $\forall i \in [m]$, $y_i \leq \beta_i$, and $x_i > 0 \Rightarrow y_i > 0$. Recall that A is lower-triangular with ones on the diagonal. Therefore, A^T is upper-triangular with ones on the diagonal.

Our base case is when $i = m$: If $x_m > 0$, then due to (6) we have $y_m = (A^T\boldsymbol{y})_m = c_m = 1 > 0$. Further, $(A^T\boldsymbol{y})_m \leq c_m \Rightarrow y_m \leq 1$. Since $\beta_m = 1$ we get $y_m \leq \beta_m$.

Now, let the hypothesis be true for $j > r$. For r if $x_r > 0$ then $(A^T\boldsymbol{y})_r = c_r \Rightarrow (A^T\boldsymbol{y})_r = y_r + \sum_{j>r} a_{jr}y_j = c_r$ (due to (6)). Since, $\forall j > r, 0 \leq y_j \leq \beta_j$ and $c_r = \sum_{j>r} |a_{jr}|\beta_j + 1$, we have $\sum_{j>r} a_{jr}y_j < c_r$. Therefore, for the equality to hold we must have $y_r > 0$. Further, $(A^T\boldsymbol{y})_r = y_r + \sum_{j>r} a_{jr}y_j \leq c_r \Rightarrow y_r \leq c_r - \sum_{j>r} a_{jr}y_j \leq c_r + \sum_{j>r} |a_{jr}|y_j \leq c_r + \sum_{j>r} |a_{jr}|\beta_j = \beta_r$.

($\Leftarrow$) If $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies (2) and (3) then clearly $\boldsymbol{x}$ is feasible in $LP(\boldsymbol{\lambda})$. Construct $\boldsymbol{y}$, from y_m to y_1 as follows: if $x_r = 0$ then set $y_r = 0$, else set $y_r = c_r - \sum_{j>r} a_{jr}y_j$. It is easy to see that $\boldsymbol{y}$ is feasible in $DLP(\boldsymbol{\lambda})$, and it, together with $(\boldsymbol{x}, \boldsymbol{\lambda})$ satisfies (6). $\square$

Lemmas 15 and 17 imply that $LP(\boldsymbol{\lambda})$ simulates the circuit. Next, we show that the circuit can be replaced by $LP(\boldsymbol{\lambda})$ without affecting the fixed-points of F. Consider function $F^{lp} : \mathbb{R}^k \rightarrow [0,\ 1]^k$, such that,

$$\boldsymbol{\lambda} \in \mathbb{R}^k, \quad F^{lp}(\boldsymbol{\lambda}) = (x_{n+2l})_{l\in[k]}, \quad \text{where } \boldsymbol{x} = LP(\boldsymbol{\lambda}) \quad (7)$$

We show that function F^{lp} is well-defined, and its fixed-points are exactly the fixed-points of F.

LEMMA 18. *F^{lp} is well defined, and $\boldsymbol{\lambda} \in \mathbb{R}^k$ is a fixed-point of F^{lp} iff it is a fixed point of F.*

PROOF. For any given $\boldsymbol{\lambda} \in \mathbb{R}^k$, using Lemmas 15 and 17, it follows that $LP(\boldsymbol{\lambda})$ has a unique solution, and if it is $\boldsymbol{x}$ then $0 \leq x_{n+2l} \leq 1,\ \forall l \in [k]$. Thus, F^{lp} is well defined.

For the second part, we know that $F^{lp}(\boldsymbol{\lambda}) \in [0,\ 1]^k$. Since, $\boldsymbol{\lambda}$ is a fixed point, this also forces $\boldsymbol{\lambda} \in [0,\ 1]^k$. Further, since circuit C represents the function F in range $[0,\ 1]^k$, it suffices to show that $F^{lp}(\boldsymbol{\lambda}) = C(\boldsymbol{\lambda})$.

In other words, when vector $\boldsymbol{\lambda}$ is the input to circuit C, then i^{th} max gate evaluates to x_i, $\forall i \in [m]$, where $\boldsymbol{x} = LP(\boldsymbol{\lambda})$. This follows using Lemmas 15 and 17. $\square$

LEMMA 19. *Size of matrix A, and vectors $\boldsymbol{c}$, $\boldsymbol{b}$ and $\boldsymbol{u}^l$, $\forall l \in [k]$ are polynomial in size$[C]$.*

PROOF. By construction A, $\boldsymbol{b}$ and $\boldsymbol{u}^l$, $\forall l \in [k]$, are formed by the coefficients of the linear expressions L_is of (2). These linear expressions are constructed due to the $+$ and $*\zeta$ gates of the circuit C, therefore, the absolute value of any of its co-efficient is at most ζ_{max}^v, where v is the number of $*\zeta$ gates in C, and ζ_{max} is the maximum absolute rational constant used in C. For rational constants ζ_1, ζ_2, since $size(\zeta_1 * \zeta_2) = size(\zeta_1) + size(\zeta_2)$, we have that the size of every co-efficient of L_is is at most $size[C]$. Thus, sizes of A, $\boldsymbol{b}$ and $\boldsymbol{u}^l$, $\forall l \in [k]$ are at most polynomial in $size[C]$. Let $A_{max} = \max_{i,j\in[m]} A_{ij}$, then by construction $c_1 = \max_{j\in[m]} c_j \leq (2A_{max}+1)^n$ (see Table 2). Therefore, the size of $\boldsymbol{c}$ is also bounded by a polynomial in $size[C]$. $\square$

From Lemmas 18 and 19 we can conclude that finding a fixed point of F is equivalent to finding one for function F^{lp}, which can be represented using polynomially many bits in the $size[C]$. Next we reduce the fixed-point computation in F^{lp} to Nash equilibrium computation in a rank-$(k+1)$ game, such that the size of the game is polynomial in size of the parameters of function F^{lp}.

4.2 Constructing Rank-(k+1) Game

Since, feasibility and complementary slackness are necessary and sufficient conditions for the solutions of an LP, it is well known that an LP can be formulated as a linear complementarity problem (LCP). Using this, next we construct an LCP whose solutions are exactly the fixed points of F^{lp}. Before we do this, note that since all the co-ordinates of the cost vector $\boldsymbol{c}$ are strictly positive, we can make it all ones vector by dividing j^{th} column of A by c_j. Let H be the transformed matrix, *i.e.,*

$$H_{ij} = {}^{A_{ij}}/_{c_j}$$

To reflect this transformation in LPs of (5) define,

$$\begin{aligned} LP'(\boldsymbol{\lambda})&: \min\{\textstyle\sum_j x_j \mid H\boldsymbol{x} \geq \boldsymbol{b} + \sum_l \boldsymbol{u}^l\lambda_l;\ \boldsymbol{x} \geq 0\} \\ DLP'(\boldsymbol{\lambda})&: \max\{(\textstyle\sum_{l\in[k]} \lambda_l\boldsymbol{u}^l + \boldsymbol{b})^T\boldsymbol{y} \mid H^T\boldsymbol{y} \leq 1, ;\ \boldsymbol{y} \geq 0\} \end{aligned} \quad (8)$$

The next lemma follows by construction.

LEMMA 20. *Given $\boldsymbol{\lambda} \in \mathbb{R}^k$, $\boldsymbol{x}$ and $\boldsymbol{y}$ are solutions of $LP(\boldsymbol{\lambda})$ and $DLP(\boldsymbol{\lambda})$ respectively iff, $\boldsymbol{x}'$, where $x'_j = x_jc_j$, $\forall j \in [m]$, and $\boldsymbol{y}$ are solutions of $LP'(\boldsymbol{\lambda})$ and $DLP'(\boldsymbol{\lambda})$ respectively.*

For $l \in [k]$, let $\boldsymbol{v}^l$ be an m-dimensional vector with $n+2l^{th}$ co-ordinate set to ${}^1/_{c_{n+2l}}$ and rest all set to zero, then ${\boldsymbol{v}^l}^T \cdot \boldsymbol{x}' = {}^{x'_{n+2l}}/_{c_{n+2l}}$. Lemma 20 implies that, at a fixed-point of F^{lp} we have $\lambda_l = x_{n+2l} = {}^{x'_{n+2l}}/_{c_{n+2l}} = {\boldsymbol{v}^l}^T \cdot \boldsymbol{x}'$, $\forall l \in [k]$, where $\boldsymbol{x} = LP(\boldsymbol{\lambda})$ and $\boldsymbol{x}' = LP'(\boldsymbol{\lambda})$. Using this as a motivation, we replace λ_l with $({\boldsymbol{v}^l}^T \cdot \boldsymbol{x})$ in the constraints of $LP'(\boldsymbol{\lambda})$. The resulting matrix will be

$$H' = H - \sum_{l=1}^{k} \boldsymbol{u}^l \cdot {\boldsymbol{v}^l}^T,\ \forall(i,j)$$

Using the above observation, and feasibility and complementary slackness conditions for (8), we construct the following LCP, called LCP$_C$,

$$\begin{array}{rll} & H'\boldsymbol{x} \geq \boldsymbol{b}; & H^T\boldsymbol{y} \leq \boldsymbol{1} \\ & \boldsymbol{x} \geq \boldsymbol{0}; & \boldsymbol{y} \geq \boldsymbol{0} \\ \forall i \in [m], & y_i(H'\boldsymbol{x} - b)_i = 0; & x_i(H^T\boldsymbol{y} - \boldsymbol{1})_i = 0 \end{array} \quad (9)$$

Before we connect the solutions of LCP$_C$ with the fixed-points of F^{lp}, we need to establish a few properties about H, H', $\boldsymbol{b}$ and $\boldsymbol{u}^l$s. For this we need to understand (2) for the last $2k$ max gates that we added in (1) to ensure that the outcome of the circuit is in $[0,\ 1]^k$. Due to Lemma 14 we have assumed that one of the inputs of every max gate is zero. For this to be the case, the (1) has to be transformed as follows,

$$\forall l \in [k], \quad \max\{0, -1 * (\max\{0, -\tau_l + 1\} - 1)\}$$

Here, $\forall l \in [k]$, τ_l is a linear expression in $x_1, \ldots, x_n, \boldsymbol{\lambda}$, and in turn so is $L_{n+2l-1} = 1 - \tau_l$. Recall that x_{n+2l-1} captures the output of the inner max gate and x_{n+2l} captures the output of the outer max gate. Therefore, we have

$$\begin{array}{lll} \forall l \in [k], & x_{n+2l-1} \geq 0, & x_{n+2l-1} \geq L_{n+2l-1}(x_1,\dots,x_n,\boldsymbol{\lambda}) \\ \forall l \in [k], & x_{n+2l} \geq 0, & x_{n+2l} \geq 1 - x_{n+2l-1} \\ & & \Rightarrow x_{n+2l-1} + x_{n+2l} \geq 1 \end{array} \quad (10)$$

The following properties are easy to obtain using (10).

P_1. $\forall l \in [k]$, $(A\boldsymbol{x})_{n+2l} = x_{n+2l-1} + x_{n+2l}$, $b_{n+2l} = 1$, and $u^{l'}_{n+2l} = 0$, $\forall l' \in [k]$.

P_2. $\forall l \in [k]$, note that x_{n+2l} appears only in one constraint. Thus $n + 2l^{th}$ column of A is a unit vector with $n + 2l^{th}$ co-ordinate set to one, and hence $(A^T\boldsymbol{y})_{n+2l} = y_{n+2l}$, $\forall l \in [k]$.

P_3. From (P_2) and the ConstructCost procedure it follows that $c_{n+2l} = 1$, $\forall l \in [k]$. Therefore, the non-zero coordinate of $\boldsymbol{v}^l$, namely $1/c_{n+2l}$, is 1, for all $l \in [k]$.

P_4. Since $H_{ij} = A_{ij}/c_j$, we get that $\forall l \in [k]$, $(H^T\boldsymbol{y})_{n+2l} = y_{n+2l}$ (using (P_2) and (P_3)), and $(H'\boldsymbol{x})_{n+2l} \geq b_{n+2l} \equiv x_{n+2l-1}/c_{n+2l-1} + x_{n+2l} \geq 1$ (using (P_1) and (P_3)).

The above properties are crucial to the over all reduction.

Lemma 21. *Vector $(\boldsymbol{x}', \boldsymbol{y}')$ is a solution of LCP_C of (9) if and only if $\boldsymbol{\lambda} \in [0,\ 1]^k$, where $\lambda_l = x'_{n+2l}$, $\forall l \in [k]$, is a fixed-point of F^{lp}.*

Proof. $(\Rightarrow)$ Let $(\boldsymbol{x}', \boldsymbol{y}')$ be a solution of LCP$_C$. Then by construction of LCP$_C$, clearly $\boldsymbol{x}'$ and $\boldsymbol{y}'$ are solutions of $LP'(\boldsymbol{\lambda})$ and $DLP'(\boldsymbol{\lambda})$ respectively, where $\lambda_l = \boldsymbol{v}^{l^T}\boldsymbol{x}' = x'_{n+2l}$, $\forall l \in [k]$ (using P_3). Set $\boldsymbol{y} = \boldsymbol{y}'$, and $\boldsymbol{x}$ be such that $x_j = x'_j/c_j$, then using Lemma 20 we get that, $\boldsymbol{x}$ and $\boldsymbol{y}$ are solutions of $LP(\boldsymbol{\lambda})$ and $DLP(\boldsymbol{\lambda})$. Further, property P_3 ensures that $x_{n+2l} = x'_{n+2l} = \lambda_l, \forall l \in [k]$. Thus, $\boldsymbol{\lambda}$ is a fixed-point of F^{lp}.

$(\Leftarrow)$ Let $\boldsymbol{\lambda}$ be a fixed-point of F^{lp} and let $\boldsymbol{x}$ and $\boldsymbol{y}$ be the solutions of $LP(\boldsymbol{\lambda})$ and $DLP(\boldsymbol{\lambda})$. Let $\boldsymbol{y}' = \boldsymbol{y}$ and $x'_j = c_j x_j, \forall j \in [m]$, then using Lemma 20 we get that $\boldsymbol{x}'$ and $\boldsymbol{y}'$ are solutions of $LP'(\boldsymbol{\lambda})$ and $DLP'(\boldsymbol{\lambda})$ respectively. Using the fact that $\boldsymbol{\lambda}$ is a fixed-point of F^{lp} and property P_3, we get $\boldsymbol{v}^{l^T} \cdot \boldsymbol{x}' = x'_{n+2l} = x_{n+2l} = \lambda_l$, $\forall l \in [k]$. In that case, feasibility and complementary slackness of $LP'(\boldsymbol{\lambda})$ and $DLP'(\boldsymbol{\lambda})$, ensures that $(\boldsymbol{x}', \boldsymbol{y}')$ is a solution of LCP$_C$. □

Next, we capture solutions of LCP$_C$ as Nash equilibria of a bimatrix game. Consider the following game:

$$\tilde{A} = \begin{bmatrix} H^T & \boldsymbol{0} \\ \boldsymbol{0}^T & 1 \end{bmatrix}, \qquad \tilde{B} = \begin{bmatrix} -H'^T & \boldsymbol{0} \\ \boldsymbol{b}^T + \boldsymbol{1}^T & 1 \end{bmatrix} \quad (11)$$

where $\boldsymbol{1}$ and $\boldsymbol{0}$ are m-dimensional vectors of 1s and 0s respectively. Number of strategies of both the players is $m + 1$. Let $(\tilde{\boldsymbol{x}}, s)$ and $(\tilde{\boldsymbol{y}}, t)$ denote mixed-strategy vectors of the first player and the second player, then we have,

$$(\tilde{\boldsymbol{x}}, s) \geq 0; \quad (\tilde{\boldsymbol{y}}, t) \geq 0; \quad s + \sum_{i=1}^{m} \tilde{x}_i = 1; \quad t + \sum_{j=1}^{m} \tilde{y}_j = 1 \quad (12)$$

Remark 22. *Adler and Verma [3], used this idea of adding an extra column/row to handle the r.h.s., in their reduction from 'solving' some special LCPs to symmetric game.*

The property that matrix of LCP$_C$ is semi-monotone, shown in the next lemma, is important to derive equivalence between the NE of $(\tilde{A}, \tilde{B})$ and the solutions of LCP$_C$.

Lemma 23. *Let $M = \begin{bmatrix} \boldsymbol{0} & H^T \\ -H' & \boldsymbol{0} \end{bmatrix}$ be the matrix of LCP_C. For any $\boldsymbol{q} \in \mathbb{R}^{2m}$ with $\boldsymbol{q} > 0$, the only solution of LCP $\{M\boldsymbol{z} \leq \boldsymbol{q};\ \boldsymbol{z} \geq 0;\ \boldsymbol{z}^T(M\boldsymbol{z} - \boldsymbol{q}) = 0\}$ is $\boldsymbol{z} = 0$.*

Proof. It suffices to show that for any $\boldsymbol{z} \geq 0, \boldsymbol{z} \neq 0$, there is a $d \in [2m]$ such that $z_d > 0$ and $(M\boldsymbol{z})_d \leq 0$. Partition $\boldsymbol{z}$ as $(\boldsymbol{x}, \boldsymbol{y})$. If $\forall l \in [k]$, $z_{n+2l} = x_{n+2l} = 0$, then $H'\boldsymbol{x} = H\boldsymbol{x}$. Therefore, $\boldsymbol{z}^T M \boldsymbol{z} = \boldsymbol{x}^T H^T \boldsymbol{y} - \boldsymbol{y}^T H \boldsymbol{x} = 0$. For all $d \in [m]$, if we have, $z_d > 0 \Rightarrow (M\boldsymbol{z})_d > 0$ then $\boldsymbol{z}^T M \boldsymbol{z} > 0$, a contradiction.

On the other hand, $\exists l \in [k]$ with $z_{n+2l} > 0$ and $(M\boldsymbol{z})_{n+2l} \leq 0$ then done. Otherwise, we have $z_{n+2l} > 0$ and $(M\boldsymbol{z})_{n+2l} > 0$. This gives $(M\boldsymbol{z})_{n+2l} = y_{n+2l} = z_{m+n+2l} > 0$ and $(M\boldsymbol{z})_{m+n+2l} = -(H'\boldsymbol{x})_{n+2l} = -x_{n+2l-1}/c_{n+2l-1} - x_{n+2l} = x_{n+2l-1}/c_{n+2l-1} - z_{n+2l} < 0$ (Using (P_4)). □

If $((\tilde{\boldsymbol{x}}, s), (\tilde{\boldsymbol{y}}, t))$ is a Nash equilibrium of game $(\tilde{A}, \tilde{B})$, the following have to be satisfied (see Lemma 4 for the NE characterization), where π_1 and π_2 are the scalars capturing payoffs of the first and the second player respectively.

$$\begin{array}{lll} & t \leq \pi_1; & s(t - \pi_1) = 0 \\ & s \leq \pi_2; & t(s - \pi_2) = 0 \\ \forall i \in [m], & (H^T\tilde{\boldsymbol{y}})_i \leq \pi_1; & \tilde{x}_i((H^T\tilde{\boldsymbol{y}})_i - \pi_1) = 0 \\ \forall j \in [m], & (-\tilde{\boldsymbol{x}}^T H')_j & \tilde{y}_j((-\tilde{\boldsymbol{x}}^T H')_j + b_j s \\ & \quad + b_j s + s \leq \pi_2; & \quad + s - \pi_2) = 0 \end{array} \quad (13)$$

Lemma 24. *If $((\tilde{x}, s), (\tilde{y}, t))$ is a Nash equilibrium of game $(\tilde{A}, \tilde{B})$ with $s > 0$ and $t > 0$, then $(\frac{\tilde{\boldsymbol{x}}}{s}, \frac{\tilde{\boldsymbol{y}}}{t})$ is a solution of LCP_C. Further, if $(\boldsymbol{x}, \boldsymbol{y})$ is a solution of LCP_C then $(\frac{(\boldsymbol{x},1)}{1+\sum_i x_i}, \frac{(\boldsymbol{y},1)}{1+\sum_i y_i})$ is a NE of game $(\tilde{A}, \tilde{B})$.*

Proof. Since, $s > 0$ and $t > 0$, we have $\pi_1 = t$ and $\pi_2 = s$ respectively (using (13)). Replacing π_1 and π_2 accordingly in the inequalities of (13), we get

$$\begin{array}{lll} \forall i \in [m], & (H^T\tilde{\boldsymbol{y}})_i \leq t; & \tilde{x}_i((H^T\tilde{\boldsymbol{y}})_i - t) = 0 \\ \forall j \in [m], & (H'\tilde{\boldsymbol{x}})_j \geq b_j s; & \tilde{y}_j((H'\tilde{\boldsymbol{x}})_j - b_j s) = 0 \end{array}$$

Dividing the first expression of first line by t and of second line by s, and the second expression in both lines by $s * t$, we get constraints of LCP$_C$.Thus $(\frac{\tilde{\boldsymbol{x}}}{s}, \frac{\tilde{\boldsymbol{y}}}{t})$ is a solution of the LCP. The second part is easy to verify using the formulation of LCP$_C$ (9) and NE conditions (12) and (13). □

Lemma 24 shows that NE of game $(\tilde{A}, \tilde{B})$ with $s > 0, t > 0$ exactly capture the solutions of LCP$_C$. Next lemma shows that these are the only NE of this game.

Lemma 25. *If $((\tilde{\boldsymbol{x}}, s), (\tilde{\boldsymbol{y}}, t))$ is a Nash equilibrium of game $(\tilde{A}, \tilde{B})$ then $s > 0$ and $t > 0$.*

Proof. We will derive a contradiction for each of the three cases separately.

Case 1: $s > 0$ and $t = 0$
Then, we have $\pi_1 = t = 0$ and therefore, $H^T\tilde{\boldsymbol{y}} \leq 0$. Since H^T is upper-triangular with strictly positive values on the diagonal, the only solution of $H^T\tilde{\boldsymbol{y}} \leq 0$ is $\tilde{\boldsymbol{y}} = 0$, which contradicts the fact that co-ordinates of vector $(\tilde{\boldsymbol{y}}, t)$ sums to one (see (12)).

Case 2: $s = 0$ and $t > 0$
Then, we have $\pi_2 = s = 0$ and therefore, $-H'\tilde{\boldsymbol{x}} \leq 0$. Recall

that $H' = H - \sum_{l=1}^{k} \boldsymbol{u}^l \cdot \boldsymbol{v}^{l^T}$ and $\boldsymbol{v}^{l^T} \cdot \tilde{\boldsymbol{x}} = \tilde{x}_{n+2l}$. Further, due to property (P_4), $\forall l \in [k]$, $(H'\tilde{\boldsymbol{x}})_{n+2l} = \tilde{x}_{n+2l-1}/c_{n+2l-1} + \tilde{x}_{n+2l}$. And, due to (P_2) we have $(H^T\tilde{\boldsymbol{y}})_{n+2l} = \tilde{y}_{n+2l}$.

Now, for an $l \in [k]$ if $\tilde{x}_{2+nl} > 0$, then $(H^T\tilde{\boldsymbol{y}})_{n+2l} = \pi_1 \Rightarrow \tilde{y}_{n+2l} = \pi_1 > 0$ (using (13) and $t > 0$). However, the $n+2l^{th}$ strategy of the second player is not fetching the maximum payoff, because $(-H'\tilde{\boldsymbol{x}})_{n+2l} \le -\tilde{x}_{n+2l} < 0$, a contradiction.

Thus, we have $\tilde{x}_{2+nl} = 0, \forall l \in [k]$. Then $H'\tilde{\boldsymbol{x}} = H\tilde{\boldsymbol{x}}$. Further, the best response condition of the first player gives $(\tilde{\boldsymbol{x}}, s)^T \tilde{A}(\tilde{\boldsymbol{y}}, t) = \pi_1 \Rightarrow \tilde{\boldsymbol{x}}^T H^T \tilde{\boldsymbol{y}} = \pi_1 > 0$, and the best response condition of the second player gives $(\tilde{\boldsymbol{x}}, s)^T \tilde{B}(\tilde{\boldsymbol{y}}, t) = \pi_2 \Rightarrow \tilde{\boldsymbol{x}}^T H^T \tilde{\boldsymbol{y}} = 0$ a contradiction.

Case 3: $s = 0$ and $t = 0$
If $\pi_1 > 0$ and $\pi_2 > 0$, then due to conditions (12) and (13), vector $\tilde{\boldsymbol{z}} = (\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}) \neq 0$ is a solution of LCP $M\boldsymbol{z} \le \boldsymbol{q}, \boldsymbol{z} \ge 0, \boldsymbol{z}^T(M\boldsymbol{z} - \boldsymbol{q}) = 0$, where $M = \begin{bmatrix} \boldsymbol{0} & H^T \\ -H' & \boldsymbol{0} \end{bmatrix}$ and $\boldsymbol{q} = (\pi_1 * \boldsymbol{1}, \pi_2 * \boldsymbol{1}) > 0$. This contradicts Lemma 23.

If $\pi_1 = 0$, then the argument is similar to *Case 1*. If $\pi_1 > 0$ and $\pi_2 = 0$, then it is similar to *Case 2*. $\square$

Now, we have established all the required facts to obtain the main theorems. Using Lemmas 25, 24, 21, 18, and 19, we show the next theorem.

THEOREM 26. *Given a kD-Linear-FIXP function F defined by circuit C, there exists a bimatrix game* $(\tilde{A}, \tilde{B})$ *with* $rank(\tilde{A} + \tilde{B}) \le (k + 1)$*, and* $\tilde{A}$ *upper-triangular, such that the Nash equilibrium strategies of the first player in game* $(\tilde{A}, \tilde{B})$ *are in one-to-one correspondence with the fixed-points of function F, where* $size[\tilde{A}] + size[\tilde{B}] \le poly(size[C])$.

PROOF. From circuit C of F construct F^{lp} of (7), then LCP$_C$ of (9) from F^{lp}, and finally game $(\tilde{A}, \tilde{B})$ of (11) from the LCP. Using Lemmas 18 and 21 it follows that solution vectors $\boldsymbol{x}$ of LCP$_C$ are in one-to-one correspondence with the fixed point of F^{lp}, which are exactly the fixed points of function F in Linear-FIXP that we started with.

Further, the Nash equilibrium $(\tilde{\boldsymbol{x}}, s), (\tilde{\boldsymbol{y}}, t))$ of game $(\tilde{A}, \tilde{B})$ maps to a solution $(\frac{\tilde{\boldsymbol{x}}}{s}, \frac{\tilde{\boldsymbol{y}}}{t})$ of LCP$_C$ (due to Lemmas 24 and 25). And, two NE with distinct first players strategies $(\tilde{\boldsymbol{x}}, s) \neq (\tilde{\boldsymbol{x}}', s')$ can not map to the same $\boldsymbol{x}$ in a solution of LCP$_C$. If they do, then we have $\frac{\tilde{\boldsymbol{x}}}{s} = \frac{\tilde{\boldsymbol{x}}'}{s'} \Rightarrow s' \sum_i \tilde{x}_i = s \sum_i \tilde{x}'_i \Rightarrow s'(1 - s) = s(1 - s') \Rightarrow s' = s \Rightarrow \tilde{\boldsymbol{x}} = \tilde{\boldsymbol{x}}'$, a contradiction.

Thus we get a game $(\tilde{A}, \tilde{B})$ whose Nash equilibrium strategies of the first player are in one-to-one correspondence with the fixed points of F. Since $H' = H - \sum_{l=1}^{k} \boldsymbol{u}^l \boldsymbol{v}^{l^T}$, $rank(\tilde{A} + \tilde{B}) \le k + 1$, and since H is upper-triangular, $\tilde{A}$ is also upper-triangular. The size of matrices $\tilde{A}$ and $\tilde{B}$ is bounded by polynomial in size of A, $\boldsymbol{b}$, $\boldsymbol{c}$ and $\boldsymbol{u}^l$, $\forall l \in [k]$, and hence the theorem follows using Lemma 19. $\square$

Using Theorems 12 and 26, we get the next theorem.

THEOREM 27. *Nash equilibrium computation in bimatrix games with rank-k,* $k > 2$ *is PPAD-hard.*

Since, matrix $\tilde{A}$ is upper-triangular, we get the following corollary,

COROLLARY 28. *Nash equilibrium computation in constant rank bimatrix games with one of the matrix being lower/upper-triangular is PPAD-hard.*

NE computation in a bimatrix game (A, B) can be reduced to computing a symmetric NE of a symmetric bimatrix game (S, S^T) where $S = \begin{bmatrix} 0 & A \\ B^T & 0 \end{bmatrix}$ [28]. Note that if, $rank(A + B)$ is k then $rank(S + S^T)$ is $2k$, and therefore using Theorem 27 we get,

COROLLARY 29. *Computing a symmetric Nash equilibrium of a symmetric game with rank-k,* $k > 5$*, is PPAD-hard.*

Etessami and Yannakakis [17] showed that solving a simple stochastic games reduces to computing a unique fixed-point of a Linear-FIXP problem. Note that if the Linear-FIXP instance that we start with has a unique fixed-point then the resulting game in Theorem 26 will have a unique Nash equilibrium strategy of the first player. In that case, the NE strategies of the second player should form a convex set because they are essentially solutions of a feasibility lp (follows Lemma 4). Using this together with the result of [17], we get the following.

COROLLARY 30. *Nash equilibrium computation in bimatrix games with a convex set of Nash equilibria is as hard as solving a simple stochastic game.*

Chen et. al. [11] showed PPAD-hardness for NE computation in bimatrix games (2-Nash), which also implies that symmetric NE computation in symmetric bimatrix game is PPAD-hard (symmetric 2-Nash) as the former reduces to the latter (discussed in Section 2). Theorem 27 gives an alternative proof of these facts. The Chen et. al. reduction goes through generalized circuit (similar to Linear-FIXP circuit) with fuzzy gates, graphical games, and game gadgets to simulate each gate of the generalized circuit separately. Our reduction bypasses all of these completely, and provides a simpler reduction using the connections between LPs, LCPs and bimatrix games. In [23] we give further simplified proof for PPAD-hardness of 2-Nash and symmetric 2-Nash, bypassing even the parameterized LP.

5. DISCUSSION

In this paper we show that Nash equilibrium computation in bimatrix games with rank$\geq$ 3 is PPAD-hard by reducing $2D$-Brouwer to rank-3 games. Given an instance of $2D$-Brouwer first we reduce it to $2D$-Linear-FIXP, a 2-dimensional fixed-point problem defined by a Linear-FIXP circuit with two inputs. Next we replace the circuit by a parameterized linear program with two parameters, and finally using the connections between LPs and LCPs, and LCPs and bimatrix games, we construct a rank-3 game. This, last step of the reduction uses the fact that the parameterized linear program was constructed from a Linear-FIXP circuit. It will be interesting to reduce a fixed-point problem, defined by a parameterized LP, to a bimatrix game in general. This will extend the classical construction of zero-sum games from linear programs by Dantzig [15]. If fixed-point problem with k-parameter LP can be reduced to a rank-k game, then it will imply that rank-2 games are also PPAD-hard, settling the only unresolved case.

As corollaries of our reduction, we get that $2D$-Linear-FIXP = PPAD = Linear-FIXP, and in turn a sharp dichotomy on complexity of Linear-FIXP problems; $1D$-Linear-FIXP is in P, while for $k \ge 2$, kD-Linear-FIXP is PPAD-complete. We also give an explicit construction of a rank-$(k+1)$ game from a kD-Linear-FIXP problem. This construction (almost) preserves the number of solutions (in terms of NE strategies of the first player), and is different from all the previous approaches. This should be of useful to understand the connections between problems that reduces to Linear-FIXP, and bimatrix games. One such example is Corollary 30 which shows that even if Nash equilibrium set of a bimatrix game is guaranteed to be convex, finding one is as hard as solving a simple stochastic game.

For the case of symmetric games, the PPAD-hardness of rank-3 games imply that computing a symmetric Nash equilibrium in a symmetric rank-6 games is PPAD-hard. The polynomial time algorithm for computing symmetric NE of a rank-1 symmetric games by Mehta et. al. [24] leaves the status of symmetric games with rank-2 to rank-5 unresolved.

Acknowledgments. I wish to thank Vijay V. Vazirani, Milind Sohoni, Bernhard von Stengel, Mihalis Yannakakis, and Jugal Garg for valuable discussions.

6. REFERENCES

[1] T. Abbott, D. Kane, and P. Valiant. On the complexity of two-player win-lose games. In *FOCS*, pages 113–122, 2005.

[2] I. Adler. The equivalence of linear programs and zero-sum games. *International Journal of Game Theory*, 42(1):165–177, 2013.

[3] I. Adler and S. Verma. A direct reduction of PPAD Lemke-verified linear complementarity problems to bimatrix games. *Manuscript*, 2013.

[4] B. Adsul, J. Garg, R. Mehta, and M. Sohoni. Rank-1 bimatrix games: A homeomorphism and a polynomial time algorithm. In *STOC*, pages 195–204, 2011.

[5] D. Andersson and P. B. Miltersen. The complexity of solving stochastic games on graphs. In *Algorithms and Computation*, pages 112–121, 2009.

[6] A. Balthasar. Equilibrium tracing in strategic-form games. *Economic Theory*, 42(1):39–54, 2010.

[7] D. Chakrabarty and N. Devanur. On competitiveness in uniform utility allocation markets. Unpublished manuscript, 2006.

[8] X. Chen, D. Dai, Y. Du, and S.-H. Teng. Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *FOCS*, 2009.

[9] X. Chen and X. Deng. On the complexity of 2D discrete fixed point problem. In *ICALP*, pages 489–500, 2006.

[10] X. Chen, X. Deng, and S.-H. Teng. Sparse games are hard. In *WINE*, pages 262–273, 2006.

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

[12] A. Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992.

[13] A. Condon. On algorithms for simple stochastic games. *Advances in computational complexity theory*, 13:51–73, 1993.

[14] V. Conitzer and T. Sandholm. New complexity results about Nash equilibria. *Games and Economic Behavior*, 63(2):621–641, 2008.

[15] G. B. Dantzig. A proof of the equivalence of the programming problem and the game problem. *In: Koopmans TC (ed) Activity analysis of production and allocation*, pages 330–335, 1951.

[16] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing, Special issue for STOC 2006*, 39(1):195–259, 2009.

[17] K. Etessami and M. Yannakakis. On the complexity of Nash equilibria and other fixed points. *SIAM Journal on Computing*, 39(6):2531–2597, 2010. Preliminary version appeared in STOC'07.

[18] I. Gilboa and E. Zemel. Nash and correlated equilibria: Some complexity considerations. *Games Econ. Behav.*, 1:80–93, 1989.

[19] P. W. Goldberg, C. H. Papadimitriou, and R. Savani. The complexity of the homotopy method, equilibrium selection, and Lemke-Howson solutions. *ACM Transactions on Economics and Computation*, 1(2):9:1–9:25, 2013.

[20] S. Kintali, L. J. Poplawski, R. Rajaraman, R. Sundaram, and S.-H. Teng. Reducibility among fractional stability problems. In *FOCS*, pages 283–292, 2009.

[21] C. E. Lemke. Bimatrix equilibrium points and mathematical programming. *Management Science*, 11(7):681–689, 1965.

[22] C. E. Lemke and J. J. T. Howson. Equilibrium points of bimatrix games. *SIAM J. on Applied Mathematics*, 12(2):413–423, 1964.

[23] R. Mehta. Constant rank bimatrix games are ppad-hard. 2014. arXiv preprint arXiv:1402.3350.

[24] R. Mehta, V. V. Vazirani, and S. Yazdanbod. Settling some open problems on symmetric Nash equilibria. Manuscript, 2013.

[25] R. B. Myerson. *Game Theory: Analysis of Conflicts*. Harward Univ. Press, 1991.

[26] J. Nash. Non-cooperative games. *Annals of Mathematics*, 54(2):289–295, September 1951.

[27] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *JCSS*, 48(3):498–532, 1994.

[28] C. H. Papadimitriou. The complexity of finding Nash equilibria. *Chapter 2, Algorithmic Game Theory, eds. N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani*, pages 29–50, 2007.

[29] R. Savani and B. von Stengel. Hard-to-solve bimatrix games. *Econometrica*, 74(2):397–429, 2006.

[30] E. Sperner. Neuer beweis fur die invarianz der dimensionszahl und des gebietes. *Abhandlungen aus dem Mathematischen Seminar Universitat Hamburg*, 6:265–272, 1928.

[31] A. H. van den Elzen and A. J. J. Talman. A procedure for finding Nash equilibria in bi-matrix games. *ZOR Methods Models Oper Res*, 35:27–43, 1991.

[32] J. von Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100:295–320, 1928.

[33] B. von Stengel. Equilibrium computation for two-player games in strategic and extensive form. *Chapter 3, Algorithmic Game Theory, eds. N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani*, 2007.