Unique End of Potential Line

John Fearnley
University of Liverpool

Spencer Gordon
California Institute of Technology

Ruta Mehta
University of Illinois at Urbana-Champaign

Rahul Savani
University of Liverpool

Abstract

The complexity class CLS was proposed by Daskalakis and Papadimitriou in 2011 to understand the complexity of important NP search problems that admit both path following and potential optimizing algorithms. Here we identify a subclass of CLS – called UniqueEOPL – that applies a more specific combinatorial principle that guarantees unique solutions. We show that UniqueEOPL contains several important problems such as the P-matrix Linear Complementarity Problem, finding Fixed Point of Contraction Maps, and solving Unique Sink Orientations (USOs). UniqueEOPL seems to a proper subclass of CLS and looks more likely to be the right class for the problems of interest. We identify a problem – closely related to solving contraction maps and USOs – that is complete for UniqueEOPL. Our results also give the fastest randomised algorithm for P-matrix LCP.

2012 ACM Subject Classification Theory of computation → Problems, reductions and completeness

Keywords and phrases P-matrix Linear Complementarity Problem, Unique Sink Orientation, Contraction Map, TFNP, Total Search Problems, Continuous Local Search

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.51

Category Regular Paper Track A

Related Version This paper substantially revises and extends the work described in our previous preprint "End of Potential Line" `arXiv:1804.03450` [21]. The full version of the paper is available at `https://arxiv.org/abs/1811.03841`.

Funding *John Fearnley*: Research supported by EPSRC grant EP/P020909/1
Ruta Mehta: Research supported by NSF grant CCF-1750436

Acknowledgements We thank Aviad Rubinstein and Kousha Etessami for alerting us to the work in [51], and we thank Rasmus Ibsen Jensen for helpful comments on a preprint of this paper.

1 Introduction

The complexity class TFNP contains search problems that are guaranteed to have a solution, and whose solutions can be verified in polynomial time [44]. While it is a semantically defined complexity class and thus unlikely to contain complete problems, a number of syntactically defined subclasses of TFNP have proven very successful at capturing the complexity of total search problems. In this paper, we focus on two in particular, PPAD and PLS. The class PPAD was introduced in [49] to capture the difficulty of problems that are guaranteed total by a parity argument. It has attracted intense attention in the past decade, culminating in a series of papers showing that the problem of computing a Nash-equilibrium in two-player games is PPAD-complete [10, 13], and more recently a conditional lower bound that rules out a PTAS for the problem [52]. No polynomial-time algorithms for PPAD-complete problems are known, and recent work suggests that no such algorithms are likely to exist [4, 25]. PLS is the class of problems that can be solved by local search algorithms (in perhaps exponentially-many steps). It has also attracted much interest since it was introduced in [38], and looks similarly unlikely to have polynomial-time algorithms. Examples of PLS-complete problems include computing: a pure Nash equilibrium in a congestion game [19], a locally optimal max cut [53], or a stable outcome in a hedonic game [24].

If a problem lies in PPAD and PLS then it is unlikely to be complete for either class, since this would imply an extremely surprising containment of one class in the other. Daskalakis and Papadimitriou [14] observed that several prominent total function problems for which no polynomial-time algorithms are known lie in PPAD $\cap$ PLS. Motivated by this they introduced CLS, a syntactically defined subclass of PPAD $\cap$ PLS, that captures optimization problems over a continuous domain in which a continuous potential function is being minimized, with access to a polynomial-time continuous improvement function. They showed that many well-studied problems are in CLS, including the problem of solving a simple stochastic game, the more general problems of solving a P-matrix Linear Complementarity Problem, finding an approximate fixpoint to a contraction map, finding an approximate stationary point of a multivariate polynomial, and finding a mixed Nash equilibrium of a congestion game. In this paper we study an interesting subset of CLS consisting of problems with *unique* solutions.

Contraction. In this problem we are given a function $f : \mathbb{R}^d \to \mathbb{R}^d$ that is purported to be c-contracting, meaning that for all points $x, y \in [0, 1]^n$ we have $d(f(x), f(y)) \leq c \cdot d(x, y)$, where c is a constant satisfying $0 < c < 1$, and d is a distance metric. Banach's fixpoint theorem states that if f is contracting, then it has a unique *fixpoint* [3], meaning that there is a unique point $x \in \mathbb{R}^d$ such that $f(x) = x$.

P-LCP. The *P-matrix linear complementarity problem* (P-LCP) is a variant of the linear complementarity problem in which the input matrix is a P-matrix [12]. An interesting property of this problem is that, if the input matrix actually is a P-matrix, then the problem is guaranteed to have a unique solution [12]. Designing a polynomial-time algorithm for P-LCP has been open for decades, at least since the 1978 paper of Murty [47] that provided exponential-time examples for *Lemke's algorithm* [42] for P-LCPs.

USO. A *unique sink orientation* (USO) is an orientation of the edges of an n-dimensional hypercube such that every face of the cube has a unique sink. Since the entire cube is a face of itself, this means that there is a unique vertex of the cube that is a sink, meaning that all edges are oriented inwards. The USO problem is to find this *unique* sink.

All of these problems are most naturally stated as *promise* problems, since we have no way of efficiently verifying whether a function is contracting, whether a matrix is a P-matrix, or whether an orientation is a USO. Hence, it makes sense, for example, to study the contraction problem where it is promised that the function f is contracting, and likewise for the other two. However, each of these problems can be turned into non-promise problems that lie in TFNP. In the case of Contraction, if the function f is not contracting, then there exists a short certificate of this fact. Specifically, any pair of points $x, y \in \mathbb{R}^d$ such that $d(f(x), f(y)) > c \cdot d(x, y)$ give an explicit proof that the function f is not contracting. We call these *violations*, since they witness a violation of the promise inherent in the problem.

So, Contraction can be formulated as the non-promise problem of either finding a solution or finding a violation. This problem is in TFNP because in the case where there is not a unique solution, there must exist a violation of the promise. The P-LCP and USO problems also have violations that can be witnessed by short certificates, and so they can be turned into non-promise problems in the same way, and these problems also lie in TFNP. For

Contraction and P-LCP we actually know that both are in CLS [14]. Prior to this work USO was not known to lie in any non-trivial subclass of TFNP, and placing USO into a non-trivial subclass of TFNP was identified as an interesting open problem by Kalai [39, Problem 6].

We remark that not every problem in CLS has the uniqueness properties that we identify above. For example, the KKT problem [14] lies in CLS, but has no apparent notion of having a unique solution. The problems that we study share the special property that there is a natural promise version of the problem, and that promise problem has a unique solution.

Our contributions. We define the complexity classes PromiseUEOPL and UniqueEOPL to capture problems in CLS that have unique solutions. We argue that UniqueEOPL is likely to be a strict subset of CLS. We introduce the notion of *promise-preserving reductions*, which allow us to simultaneously obtain results for the promise and non-promise versions of problems. We show that all of our motivating problems – USO, P-LCP, and finding a fixpoint of a Piecewise-Linear Contraction under an ℓ_p-norm – are contained in UniqueEOPL (PromiseUEOPL for the promise versions) via promise-preserving reductions. Thus, we resolve the open problem of Kalai mentioned above, by showing that USO is in UniqueEOPL and thus also CLS, PPAD and PLS. Our results also imply that parity, mean-payoff, discounted, and simple-stochastic games lie in UniqueEOPL. We also provide a complete problem for UniqueEOPL, called One-Permutation Discrete Contraction (OPDC). It is motivated by a discretized version of contraction, but it is also closely related to USO, and we consider its hardness to be a substantial step towards showing hardness for contraction and USO.

The new techniques used in our reductions also lead to new algorithmic results. We obtain direct polynomial-time algorithms for finding fixpoints of contraction maps in fixed dimension for any ℓ_p norm, where previously such algorithms relied on a reduction to the Tarski fixpoint problem [51]. Our reduction for P-LCP allows a technique of Aldous [2] to be applied, which in turn gives the fastest-known randomized algorithm for P-LCP.

A main message of our paper is that several important problems lie in UniqueEOPL *and* that UniqueEOPL is likely to be a proper subset of CLS.

Related work. Hubáček and Yogev [36] proved lower bounds for CLS. They introduced a problem known as EndOfMeteredLine which they showed was in CLS, and for which they proved a query complexity lower bound of $\Omega(2^{n/2}/\sqrt{n})$ and hardness under the assumption that there were one-way permutations and indistinguishability obfuscators for problems in $\mathsf{P}_{/\mathsf{poly}}$. Recently, two variants of ContractionMap have been shown to be CLS-complete. Whereas in the original definition of ContractionMap it is assumed that an ℓ_p or ℓ_∞ norm is fixed, and the contraction property is measured w.r.t. the induced metric, in these two complete variants, a metric [15] and meta-metric [20] are given as input to the problem.

Papadimitriou showed that P-LCP, the problem of solving the LCP or returning a violation of the P-matrix property, is in PPAD [49] using Lemke's algorithm. The relationship between Lemke's algorithm and PPAD has been studied by Adler and Verma [1]. Later, Daskalakis and Papadimitrou showed that P-LCP is in CLS [14], using the potential reduction method in [41]. Many algorithms for P-LCP have been studied, e.g., [40,46,47]. However, no polynomial-time algorithms are known for P-LCP. The best-known algorithms for P-LCP are based on a reduction to Unique Sink Orientations (USOs) of cubes [59]. For an P-matrix LCP of size n, the USO algorithms of [60] apply, and give a deterministic algorithm that runs in time $O(1.61^n)$ and a randomized algorithm with expected running time $O(1.43^n)$. The application of Aldous' algorithm [2] to the UniqueEOPL instance that we produce from a P-matrix LCP takes expected time $2^{n/2} \cdot \text{poly}(n) = O(1.4143^n)$ in the worst case.

We study USOs of cubes, a problem that was first studied by Stickney and Watson [59] in the context of P-matrix LCPs. Motivated by Linear Programming, *acylic* USOs (AUSOs) have also been studied, both for cubes and general polytopes [28, 33]. Recently Gärtner and Thomas studied the computational complexity of recognizing USOs and AUSOs [29]. A series of papers provide upper and lower bounds for approaches for solving (A)USOs, including [22, 23, 26, 31, 43, 54, 55, 60, 61]. To the best of our knowledge, we are first to study the general problem of solving a USO from a complexity-theoretic point of view.

The problem of computing a fixpoint of a continuous map $f : \mathcal{D} \to \mathcal{D}$ with Lipschitz constant c has been extensively studied, in both continuous and discrete variants [8, 9, 16]. For arbitrary maps with $c > 1$, exponential bounds on the query complexity are known [7, 32]. In [6, 35, 58], algorithms for computing fixpoints of weakly ($c = 1$) and strictly ($c < 1$) contracting maps are studied.

A number of algorithms are known for contractions w.r.t. the ℓ_2 norm [34, 48, 57]. There is an exponential lower bound for absolute approximation with $c = 1$ [57]. For relative approximation ($||x - f(x)|| \leq \epsilon$) in dimension d, an $O(d \cdot \log 1/\epsilon)$ time algorithm is known [34]. For absolute approximation ($||x - x^*|| \leq \epsilon$ where x^* is an exact fixpoint) with $c < 1$, an ellipsoid-based algorithm with time complexity $O(d \cdot [\log(1/\epsilon) + \log(1/(1-c))])$ is known [34]. For the ℓ_∞ norm, [56] gave an algorithm to find an ϵ-relative approximation in time $O(\log(1/\epsilon)^d)$ which is polynomial for constant d. A polynomial time algorithm for finding an approximate fixpoint of a contraction map in constant dimension can be obtained through a reduction to the Tarski fixpoint problem [51].

2 Unique End of Potential Line

We define two new complexity classes called EOPL and UniqueEOPL. EOPL combines EndOfLine and SinkOfDag, the canonical complete problems for PPAD and PLS [49].

▶ **Definition 1** (EndOfPotentialLine). *Given Boolean circuits $S, P : \{0,1\}^n \to \{0,1\}^n$ such that $P(0^n) = 0^n \neq S(0^n)$ and a Boolean circuit $V : \{0,1\}^n \to \{0, 1, \dots, 2^m - 1\}$ such that $V(0^n) = 0$ find one of the following:*

(R1) A point $x \in \{0,1\}^n$ such that $S(P(x)) \neq x \neq 0^n$ or $P(S(x)) \neq x$.

(R2) A point $x \in \{0,1\}^n$ such that $x \neq S(x)$, $P(S(x)) = x$, and $V(S(x)) - V(x) \leq 0$.

This problem defines an exponentially large graph where each vertex has in-degree and out-degree at most one (as in EndOfLine) that is also a DAG (as in SinkOfDag). An edge exists from x to y if and only if $S(x) = y$, $P(y) = x$, and $V(x) < V(y)$. Only some bit-strings encode vertices. Specifically, if $S(x) = x$ for some bit-string x, then x does *not* encode a vertex. The problem consists of a single instance that is simultaneously an instance of EndOfLine and an instance of SinkOfDag. To solve the problem, it suffices to solve *either* of these problems. Solutions of type (R1) are ends of lines, and solutions of type (R2) are points where the potential does not increase along an edge.

We define the complexity class EOPL to consist of all problems that can be reduced in polynomial time to EndOfPotentialLine. We show the following containment.

▶ **Theorem 2.** EOPL $\subseteq$ CLS.

To prove this, we reduce EndOfPotentialLine to the EndOfMeteredLine, which was defined and shown to be in CLS by Hubáček and Yogev [36]. The difference between the two problems is that EndOfMeteredLine requires that the potential increases by *exactly* one along each edge. Our reduction inserts new vertices into the instance to satisfy this property.

2.1 Promise problems with unique solutions

Each of the problems that we study have a *promise*, and if the promise is satisfied the problem has a *unique* solution. For example, in the contraction problem, we are given a function as a circuit but cannot efficiently check whether the function is actually contracting. If the function is contracting, then Banach's fixpoint theorem states that it has a unique fixpoint [3]. If it is not contracting, there exist *violations* that can be witnessed by short certificates. We can use violations to formulate the problem as a non-promise problem that lies in TFNP: we ask for either a fixpoint or a violation of contraction.

When we place this type of problem in EOPL, we obtain an instance with extra properties. Specifically, if the original problem has no violations, i.e., the promise is satisfied, then the EndOfPotentialLine instance will contain a *single* line that starts at 0^n, and ends at the unique solution. So, if we ever find two distinct lines, we immediately know that the instance fails to satisfy the promise. We define the following problem to capture these properties.

▶ **Definition 3** (UniqueEOPL). *Given Boolean circuits $S, P : \{0,1\}^n \to \{0,1\}^n$ such that $P(0^n) = 0^n \neq S(0^n)$ and a Boolean circuit $V : \{0,1\}^n \to \{0, 1, \ldots, 2^m - 1\}$ such that $V(0^n) = 0$ find one of the following:*

(U1) A point $x \in \{0,1\}^n$ such that $P(S(x)) \neq x$.

(UV1) A point $x \in \{0,1\}^n$ such that $x \neq S(x)$, $P(S(x)) = x$, and $V(S(x)) - V(x) \leq 0$.

(UV2) A point $x \in \{0,1\}^n$ such that $S(P(x)) \neq x \neq 0^n$.

(UV3) Two points $x, y \in \{0,1\}^n$, such that $x \neq y$, $x \neq S(x)$, $y \neq S(y)$, and either $V(x) = V(y)$ or $V(x) < V(y) < V(S(x))$.

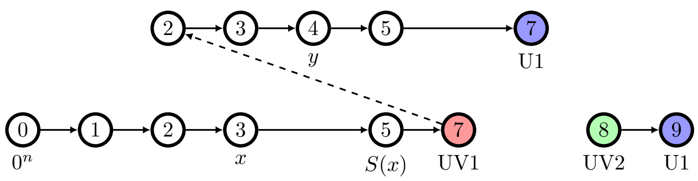

Figure 1 UniqueEOPL instance with 3 lines. The *main line* starts at 0^n and ends with a UV1 solution. There is a final line of length one to the bottom right, whose start vertex is a UV2 solution. The ranges of potential values for the main line and top line intersect, so they contribute many UV3 solutions. We highlight one on the diagram with x, $S(x)$, and y, such that $V(x) < V(y) < V(S(x))$.

We split solutions into two types: proper solutions and violations. Solutions of type (U1) encode the end of a line, which are the proper solutions. (UV1) violations are vertices at which the potential fails to increase. (UV2) violations are the start of any line other than 0^n. (UV3) violations are a different witness that there are more than one line, namely a pair of vertices x and y, with either $V(x) = V(y)$, or such that $V(y)$ lies between $V(x)$ and $V(S(x))$, so x and y cannot lie on the same line. See Figure 1 for an illustration.

We remark that (UV1) and (UV2) violations already capture the property that "there is a unique line", since if we exclude them, then a second line cannot exist. However, with only these two violations, we may find two vertices on two different lines, but both may be exponentially many steps away from the start of their respective lines. (UV3) violations make any pair of vertices that are provably on two different lines a violation. All of the problems in UniqueEOPL have the property that if we ever find a (UV3) violation, the problem can be solved immediately.

We define UniqueEOPL to be the class of problems that can be reduced in polynomial time to UNIQUEEOPL[1]. For each of our problems, it is also interesting to consider the promise variant, in which it is guaranteed via a promise that no violations exist. We define PROMISEUNIQUEEOPL to be the promise version of UNIQUEEOPL in which it is promised that 0^n is the only start of a line, and PromiseUEOPL to be the class of promise problems that can be reduced in polynomial time to PROMISEUNIQUEEOPL.

The problem UNIQUEEOPL has the interesting property that, if it is promised that there are no violation solutions, then there must be a unique solution. All of the problems that we study in this paper share this property, and indeed when we reduce them to UNIQUEEOPL, the resulting instance will have a unique line whenever the original problem has no violation solutions. We formalise this by defining the concept of a *promise-preserving* reduction. This is a reduction between two problems A and B, both of which have proper solutions and violation solutions. The reduction is promise-preserving if, when it is promised that A has no violations, then the resulting instance of B also has no violations. Hence, if we reduce a problem to UNIQUEEOPL via a chain of promise-preserving reductions, and we know that there are no violations in the original problem, then there is a unique line ending at the unique proper solution in the instance. So, if we show that a problem is in UniqueEOPL (or UniqueEOPL-complete) via a chain of promise-preserving reductions, then we automatically get that the promise version of that problem, where it is promised that there are no violations, lies in PromiseUEOPL (or PromiseUEOPL-complete).

3 One-Permutation Discrete Contraction (OPDC)

OPDC plays a crucial role in our results. We show that it lies in UniqueEOPL, and the we reduce both PL-CONTRACTION and UNIQUE-SINK-ORIENTATION to it, thereby showing that those problems also lie in UniqueEOPL. We also show that UNIQUEEOPL can be reduced to OPDC, making it the first example of a non-trivial UniqueEOPL-complete problem.

Direction functions. OPDC can be seen as a discrete variant of the continuous contraction problem. A contraction map is a function $f : [0,1]^n \to [0,1]^d$ that is contracting under a metric d, i.e., $d(f(x), f(y)) \leq c \cdot f(x,y)$ for all $x, y \in [0,1]^d$ and some constant c satisfying $0 < c < 1$. We discretize this by overlaying a grid of points on the $[0,1]^d$ cube. Let $[k]$ denote the set $\{0, 1, \dots, k\}$. Given a tuple of grid widths $(k_1, k_2, \dots, k_d)$, we define the set $P(k_1, k_2, \dots, k_d)$ as $[k_1] \times [k_2] \times \cdots \times [k_d]$. We sometimes refer to $P(k_1, k_2, \dots, k_d)$ simply as P. Note that each point $p \in P$ is a tuple $(p_1, p_2, \dots, p_d)$, where p_i is an integer between 0 and k_i, and this point maps onto the point $(p_1/k_1, p_2/k_2, \dots, p_d/k_d) \in [0,1]^d$.

Instead of a single function f, in the discretized problem we use a family of *direction functions* over the grid P. For each dimension $i \leq d$, we have function $D_i : P \to \{\mathsf{up}, \mathsf{down}, \mathsf{zero}\}$. Intuitively, the natural reduction from a contraction map f to a family of direction functions would, for each point $p \in P$ and each dimension $i \leq d$ set: $D_i(p) = \mathsf{up}$ whenever $f(p)_i > p_i$, $D_i(p) = \mathsf{down}$ whenever $f(p)_i < p_i$, and $D_i(p) = \mathsf{zero}$ whenever $f(p)_i = p_i$. In other words, the function D_i simply outputs whether $f(p)$ moves up, down, or not at all in dimension i. So a point $p \in P$ with $D_i(p) = \mathsf{zero}$ for all i would correspond to the fixpoint of f.

A 2d example. To illustrate this definition, consider the 2d instance given in the two leftmost parts of Figure 2, which we use as a running example. These are two direction functions: the left one shows a direction function for the up-down dimension, which we

[1] We remark that Hubáček and Yogev [36] mention that their lower bound results for CLS may also apply to such problems, but they did not investigate a corresponding complexity class.

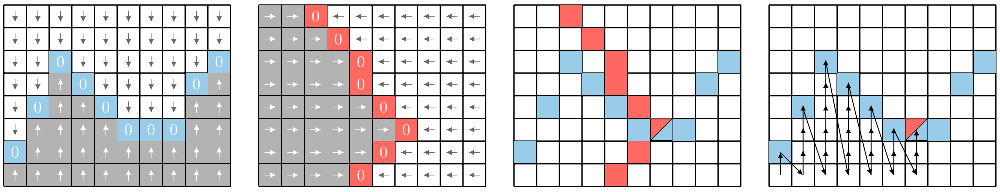

Figure 2 This figure should be viewed in color. From left to right: A direction function for the up/down dimension; a direction function for the left/right dimension; the red and blue surfaces; the path that we follow.

will call dimension 1 and illustrate in blue. The right one shows the left-right dimension, which we will call dimension 2 and illustrate in red. Each square represents a point in the discretized space, and the value of the direction function is shown inside the box. Note that there is exactly one point p where $D_1(p) = D_2(p) = \mathsf{zero}$, which is the fixpoint that we seek.

Slices. We will frequently refer to subsets of P in which some dimensions have been fixed. A *slice* is represented as a tuple $(s_1, s_2, \dots, s_d)$, where each s_i is either: a number in $[k_i]$, which indicates that dimension i should be fixed to s_i; or the special symbol $*$, which indicates that dimension i is free to vary. We define Slice_d to be the set of all possible slices in dimension d. Given a slice $s \in \mathsf{Slice}_d$, we define $P_s \subseteq P$ to be the set of points in that slice, i.e., P_s contains every point $p \in P$ such that $p_i = s_i$ whenever $s_i \neq *$. We say that a slice $s' \in \mathsf{Slice}_d$ is a sub-slice of a slice $s \in \mathsf{Slice}_d$ if $s_j \neq * \implies s'_j = s_j$ for all $j \in [d]$. An *i-slice* is a slice s for which $s_j = *$ for all $j \leq i$, and $s_j \neq *$ for all $j > i$. In other words, all dimensions up to and including dimension i are allowed to vary, while all other dimensions are fixed.

In our 2d example, there are three types of i-slices. There is one 2-slice: the slice $(*, *)$ that contains every point. For each x, there is a 1-slice $(*, x)$, which restricts the left/right dimension to the value x. For each pair x, y there is a 0-slice (y, x), which contains only the exact point corresponding to x and y.

The OPDC problem. Let P be a grid of points in dimension d and $\mathcal{D} = (D_i)_{i=1,\dots,d}$ a family of direction functions over P. We say that a point $p \in P_s$ in some slice s is a *fixpoint* of s if $D_i(p) = \mathsf{zero}$ for all dimensions i where $s_i = *$. The promise version of OPDC promises that for every i-slice s, the following conditions hold.

1. There is a unique fixpoint of s.
2. Let $s' \in \mathsf{Slice}_d$ be a sub-slice of s where some coordinate i for which $s_i = *$ has been fixed. If q is the unique fixpoint of s, and p is the unique fixpoint of s', then $p_i < q_i$ implies $D_i(p) = \mathsf{up}$, and $p_i > q_i$ implies $D_i(p) = \mathsf{down}$.

Since the slice $(*, *, \dots, *)$ is an i-slice, the first condition implies that all i-slices including the full problem have a unique fixpoint. Intuitively, the second condition just ensures that the D_i behave as direction functions. It says that if we have found the unique fixpoint p of the $(i+1)$-slice s', and if it is not the unique fixpoint of the i-slice s, then $D_i(p)$ tells us which way to walk to find the fixpoint of s. This is a crucial property used in our reduction from OPDC to UniqueEOPL, and in our algorithms for contraction maps.

In our 2d example, the first condition requires that every slice $(*, x)$ has a unique fixpoint, i.e., in every column there is a unique blue zero. The second condition says that, if we are at some blue zero, then the red direction function at that point tells us the direction of the overall fixpoint. Our example satisfies both conditions. We next define a total variant of OPDC that uses violations to cover the cases where $\mathcal{D}$ fails to satisfy these two conditions.

▶ **Definition 4** (OPDC). *Given a tuple $(k_1, k_2, \ldots, k_d)$ and circuits $(D_i(p))_{i=1,\ldots,d}$, where each circuit $D_i : P(k_1, k_2, \ldots, k_d) \to \{\mathsf{up}, \mathsf{down}, \mathsf{zero}\}$, find one of the following*

(O1) A point $p \in P$ such that $D_i(p) = \mathsf{zero}$ for all i.

(OV1) An i-slice s and points $p, q \in P_s$ with $p \neq q$ s.t. $D_j(p) = D_j(q) = \mathsf{zero}$ for all $j \leq i$.

(OV2) An i-slice s and points $p, q \in P_s$ s.t. $D_j(p) = D_j(q) = \mathsf{zero}$ for all $j < i$, $p_i = q_i + 1$, and $D_i(p) = \mathsf{down}$ and $D_i(q) = \mathsf{up}$.

(OV3) An i-slice s and a point $p \in P_s$ s.t. $D_j(p) = D_j(q) = \mathsf{zero}$ for all $j < i$, and either $p_i = 0$ and $D_i(p) = \mathsf{down}$, or $p_i = k_i$ and $D_i(p) = \mathsf{up}$.

Solution type (O1) encodes a fixpoint, which is the proper solution of OPDC. Solution type (OV1) witnesses a violation of the fact that each i-slice should have a unique fixpoint, by giving two different points p and q that are both fixpoints of the same i-slice. Solutions of type (OV2) witness violations of the first and second properties. In these solutions we have two points p and q that are both fixpoints of their respective $(i-1)$-slices and are directly adjacent in an i-slice s. If there is a fixpoint r of the slice s, then this witnesses a violation of the fact that $D_i(p)$ and $D_i(q)$ should both point towards r, since clearly one of them does not. On the other hand, if slice s has no fixpoint, then p and q also witness this fact, since the fixpoint should be in-between p and q, which is not possible. Solutions of type (OV3) consist of a point p that is a fixpoint of its $(i-1)$-slice but $D_i(p)$ points outside the boundary of the grid. These are violations because $D_i(p)$ should point towards the fixpoint of the i-slice containing p, but that fixpoint cannot be outside the grid.

It is perhaps not immediately obvious that OPDC is a total problem. Our promise-preserving reduction from OPDC to UniqueEOPL proves totality, and shows that if the OPDC instance has no violations then it has a unique solution. The prefix *One-Permutation* was chosen to emphasize that our solution conditions only consider i-slices. In the continuous contraction map problem with an ℓ_p metric, *every* slice has a unique fixpoint, and our reduction from contraction maps to OPDC works for any permutation of the dimensions.

3.1 One-Permutation Discrete Contraction is UniqueEOPL-complete

To show that OPDC lies in UniqueEOPL under promise-preserving reductions, we make use of an intermediate problem that we call UniqueForwardEOPL, which is a version of UniqueEOPL in which we only have a successor circuit S, meaning that no predecessor circuit P is given. Without this circuit, there is no way to tell if a vertex is the start of a line, so the only solutions to this problem are the end of a line, a vertex at which the potential fails to increase, or the analogue of (UV3). Although we no longer have a predecessor circuit, it is still the case that if the problem is promised to have no violations, then it must contain a single line ending at the unique proper solution. We reduce OPDC to UniqueForwardEOPL, and then reduce UniqueForwardEOPL to UniqueEOPL.

An illustration of the reduction. We illustrate using the 2d example shown in Figure 2. The reduction uses the notion of a *surface*. The surface of a direction function D_i is exactly the set of points $p \in P$ such that $D_i(p) = \mathsf{zero}$. In the third part of Figure 2, we have overlaid the surfaces of the two direction functions from Figure 2. The fixpoint p that we seek has $D_i(p) = \mathsf{zero}$ for all dimensions i, and so it lies at the intersection of these surfaces.

To reach the overall fixpoint, we walk along a path starting from the bottom-left corner, which is shown on the rightmost part of Figure 2. The path begins by walking upwards until it finds the blue surface. Once it has found the blue surface, it then there are two possibilities: either we have found the overall fixpoint, in which case the line ends, or we have not found the overall fixpoint, and the red direction function tells us that the direction of the overall fixpoint is to the right. If we have not found the overall fixpoint, then we move one step to the right, go back to the bottom of the diagram, and start walking upwards again. We keep repeating this until we find the overall fixpoint.

How do we define a potential? Observe that the dimension-two coordinates of the points on the line are weakly monotone, i.e., the line never moves to the left. Furthermore, for any dimension-two slice (any slice in which the left/right coordinate is fixed), the dimension-one coordinate is increasing. So, if $p = (p_1, p_2)$ denotes any point on the line, if k denotes the maximum coordinate in either dimension, then the function $V(p_1, p_2) = k \cdot p_2 + p_1$ is a function that monotonically increases along the line, which we can use as a potential function.

Uniqueness. For a promise-preserving reduction, the line must be unique whenever the OPDC instance has no violations. To do this we must be careful that only points that are to the left of the fixpoint are actually on the line, and that no "false" line exists to the right of the fixpoint. Here we rely on the following fact: if the line visits a point with coordinate x in dimension 2, then it must have visited the point p on the blue surface in the slice defined by $x - 1$. Moreover, for that point p we must have $D_2(p) = \mathsf{up}$, which means that it is to the left of the overall fixpoint. Using this fact, each vertex on our line will be a pair (p, q), where p is the current point that we are visiting, and q is either the symbol $-$, indicating that we are still in the first column of points, and we have never visited a point on the blue surface, or a point q that is on the blue surface that satisfies $q_2 = p_2 - 1$ and $D_2(q) = \mathsf{up}$. Hence q is always the last point that we visited on the blue surface, which provides a witness that we have not yet walked past the overall fixpoint. When we finish walking up a column, and find the point on the blue surface, we overwrite q with the new point. This step is not easily reversible, since to determine the predecessor of a vertex we would need to recover the value that was overwritten. So we create a UniqueForwardEOPL instance, and our onwards reduction to UniqueEOPL will produce a predecessor circuit.

Violations. Our 2d example does not contain any violations, but our reduction can still handle all possible violations in the OPDC instance. At a high level, there are two possible ways in which the reduction can go wrong if there are violations.

1. It is possible, that as we walk upwards in some column, we do not find a fixpoint, and our line will get stuck. In our 2d example, this case corresponds to a column of points in which there is no point on the blue surface. However, if there is no point on the blue surface, then we will either: find two adjacent points p and q in that column with $D_1(p) = \mathsf{up}$ and $D_2(p) = \mathsf{down}$, which is a solution of type (OV2), or find a point p at the top of the column with $D_1(p) = \mathsf{up}$, or a point q at the bottom of the column with $D_1(q) = \mathsf{down}$. Both of these are solutions of type (OV3). There is also the similar case where we walk all the way to the right without finding an overall fixpoint, in which case we will find a point p on the right-hand boundary that satisfies $D_1(p) = \mathsf{zero}$ and $D_2(p) = \mathsf{up}$, which is a solution of type (OV3).
2. The other possibility is that there may be more than one point on the blue surface in some columns. This inevitably leads to multiple lines, since if q and q' are both on the blue surface in some column, and p is in the column to the right of p and q, then (p, q) and (p, q') will both be valid vertices on two different lines. We map these violations back to solutions of type (OV1). Specifically, the points p and q, which are given as part of the two vertices, are both fixpoints of the same slice, which is exactly what (OV1) asks for.

Our reduction is promise-preserving because violations in the UFEOPL instance are never mapped back to proper solutions of the OPDC instance.

Generalizing to d dimensions. The full reduction from OPDC to UniqueForwardEOPL generalizes the approach given above to d dimensions. We say that a point $p \in P$ is on the *i-surface* if $D_j(p) = \mathsf{zero}$ for all $j \leq i$. In our 2d example we followed a line of points on the 1-surface, in order to find a point on the 2-surface. In between any two points on the 1-surface, we followed a line of points on the 0-surface (every point is trivially on the 0-surface). Our line will visit a sequence of points on the $(d-1)$-surface in order to find the point on the d-surface, which is the fixpoint. Between any two points on the $(d-1)$-surface the line visits a sequence of points on the $(d-2)$-surface, between any two points on the $(d-2)$-surface the line visits a sequence of points on the $(d-3)$-surface, and so on. Every time we find a point on the i-surface, we remember it, increment our position in dimension i by 1, and reset our coordinates back to 0 for all dimensions $j < i$. Hence, a vertex will be a tuple $(p_0, p_1, \dots, p_d)$, where each p_i is either the symbol $-$, indicating that we have not yet encountered the i-surface, or the most recent point on the i-surface that we have visited.

The potential is likewise generalized so that the potential of a point p is proportional to $\sum_{i=1}^{d} k^i p_i$, where k is some constant that is larger than the grid size. Thus progress in dimension i dominates progress in dimension j whenever $j < i$, and the potential monotonically increase along the line. We are also able to deal with all possible violations.

Completing the reduction to UniqueEOPL. The final step of the reduction uses SinkOfVerifiableLine, a problem introduced by Bitansky et al [4]. SinkOfVerifiableLine is intuitively similar to UniqueForwardEOPL. It was shown by Hubáček and Yogev [36] that SinkOfVerifiableLine can be reduced to an EndOfMeteredLine instance with a unique line, and hence to UniqueEOPL. However, [36] only deals with the promise problem. Our contribution is to deal with violations. In doing so, we complete our chain of promise-preserving reductions from OPDC to UniqueEOPL. It is worth pointing out that this step of the reduction implies that the cryptographic hardness results shown by Bitansky et al. for SinkOfVerifiableLine [5] also apply to UniqueEOPL.

Hardness of OPDC. We show that OPDC is UniqueEOPL-hard by giving a polynomial-time promise-preserving reduction from UniqueEOPL to OPDC. Our reduction produces an OPDC instances where the set of points P is the Boolean hypercube $\{0,1\}^n$. In the case where the UniqueEOPL instance has no violations, meaning that it contains a single line, the reduction embeds this line into the hypercube. To do this, it splits the line in half. The second half is embedded into a particular sub-cube, while the first half is embedded into all other sub-cubes. This process is recursive, so each half of the line is again split in half, and further embedded into sub-cubes. The reduction ensures that the only fixpoint of the instance corresponds to the end of the line. If the UniqueEOPL instance does have violations, this embedding may fail, but we are then able to produce a violation for the original UniqueEOPL instance. We remark that this reduction makes significant progress towards showing hardness for Contraction and USO, since OPDC is a discrete variant of Contraction, and when the set of points is a hypercube, the problem is also very similar to USO. The key difference is that OPDC insists that only i-slices have a unique fixpoint, whereas Contraction and USO insist that *all* slices have unique fixpoints.

▶ **Theorem 5.** OPDC *is* UniqueEOPL*-complete under promise-preserving reductions, even when the set of points P is a hypercube.*

4 UniqueEOPL containment results

We show that USO, P-LCP, and a variant of Contraction all lie in UniqueEOPL. For each of these three problems, we provide a reduction to OPDC, shown to be in UniqueEOPL in the previous section.

A USO instance naturally gives rise to an OPDC instance where the underlying grid of points is actually a cube, and there is an easy reduction that shows the following.

▶ **Theorem 6.** *USO is in* UniqueEOPL *under promise-preserving reductions.*

This result substantially advances our knowledge about USO, since prior to this work, it was only known to lie in TFNP, and Kalai [39, Problem 6] had posed the challenge to place it in some non-trivial subclass of TFNP. We place it in *all* of the standard subclasses of TFNP[2].

We provide two reductions from P-LCP to UniqueEOPL. One is via the known reduction from P-LCP to USO [59]. The other uses Lemke's algorithm and produces a UEOPL instance with size linear in that of the P-LCP, which we require for our algorithmic result in Section 5. Lemke's algorithm is a PPAD-style path-following algorithm. The potential comes from a parameter used in Lemke's algorithm that changes monotonically on an P-matrix LCP. The complication is to deal with violations when the input matrix is not a P-matrix.

The reason for two reductions is that each produces different types of violation. We emphasize that all violations used are well-known and natural, and perhaps one can convert between them in polynomial time. Moreover, for the promise problem the choice of violations is irrelevant: each reduction independently shows that promise P-LCP lies in PromiseUEOPL.

▶ **Theorem 7.** *P-LCP* $\in$ UniqueEOPL *under promise-preserving reductions.*

For contraction, we study maps specified by *piecewise linear* functions. This differs from [14], where the map is given by an arbitrary arithmetic circuit. Although every contraction map has a unique fixpoint, for an arbitrary arithmetic circuit, the unique *exact* fixpoint may be irrational, and finding it is not known to be in FNP. Prior work instead asked for an *approximate* fixpoint [14]. However, given our interest in uniqueness of solutions we need to consider exact fixpoints, and thus study the problem with LinearFIXP arithmetic circuits [18], where multiplication of two variables is disallowed, and when the function is contracting, there is a unique rational fixpoint. This is still an interesting class of contraction maps, since it is powerful enough to represent the well-studied simple-stochastic games [11, 18]. We place this problem in UniqueEOPL via a promise-preserving reduction to OPDC. When the promise is not satisfied, the reduction either produces the standard violation, a pair of points at which the function is not contracting, or a different, more technical, violation.

OPDC was inspired by the continuous contraction problem, and our reduction from contraction is to OPDC. The most complicated part of the reduction is picking a suitable set of points for the OPDC instance that is small enough, but also is guaranteed to contain the unique fixed point of the contraction instance. To do this, we formulate the fixpoint problem for a LinearFIXP circuit as an LCP and reason about the bit-length of solutions to this LCP.

▶ **Theorem 8.** *Finding the fixpoint of a piecewise linear contraction map in the ℓ_p norm is in* UniqueEOPL *under promise-preserving reductions for any $p \in \mathbb{N} \cup \{\infty\}$.*

Finally, we note that our results imply that several other problems lie in UniqueEOPL. The simple-stochastic game (SSG) problem is known to reduce to piecewise-linear Contraction [18] and P-LCP [30]. Discounted games are known to reduce to SSGs [62], mean-payoff games to discounted games [62], and parity games to mean-payoff games [50]. So all these problems lie in UniqueEOPL too. [27] noted that ARRIVAL [17] lies in EOPL; since their EndOfPotentialLine instance contains only one line, ARRIVAL also lies in UniqueEOPL. However, none

[2] However, we do not place the problem in the recently defined class PWPP [37]

of these are promise-problems. Each can be formulated so as to *unconditionally* have a unique solution. Hence, they seem to be easier than other problems captured by UniqueEOPL.

▶ **Theorem 9.** *The following problems are in* UniqueEOPL*: Solving a parity game; mean-payoff game; discounted game; simple-stochastic game; the ARRIVAL problem.*

5 New algorithms

The insights provided by our containment results give two algorithmic results. Firstly, we obtain simple polynomial-time algorithms for finding the fixpoint of a contraction map in fixed dimension for any ℓ_p norm. This result was already known via a reduction to the problem of finding a Tarski fixpoint [51], but our algorithm utilises the structural properties of contraction that arise from our reduction to OPDC, and is arguably simpler.

Secondly, as noted in [27], one of our reductions for P-LCP allows a technique of Aldous [2] to be applied, giving the fastest known randomized algorithm for P-LCP.

6 Conjectures and conclusions

▷ Conjecture 1. USO is hard for UniqueEOPL.

Among our three motivating problems, USO seems the most likely to be UniqueEOPL-complete. Our hardness proof for OPDC already goes some way towards proving this, since it applies even on a hypercube. Going further, could we even show the stronger result of hardness for P-LCP, which would imply hardness of USO? The complexity of these two problems has been open for decades.

▷ Conjecture 2. Piecewise-Linear Contraction in an ℓ_p norm is hard for UniqueEOPL.

For this result, in addition to the i-slice vs. all slice issue, we would also need to convert the discrete OPDC problem to the continuous contraction problem. Converting discrete problems to continuous fixpoint problems has been well-studied in the context of PPAD-hardness reductions [13, 45], but here we must additionally maintain the contraction property.

Aside from hardness, we also think that the relationship between Contraction and USO should be explored further, since OPDC exposes significant, previously unrecognised, similarities between the two problems.

▷ Conjecture 3. UniqueEOPL $\subset$ EOPL $=$ CLS.

The question of EOPL vs CLS is unresolved, and we actually think it could go either way. One could show that EOPL $=$ CLS by placing either of the two known CLS-complete Contraction variants into EOPL [15, 20]. If the two classes are actually distinct, then it is interesting to ask which of the problems in CLS are also in EOPL.

On the other hand, we believe that UniqueEOPL is a strict subset of EOPL. The evidence for this is that the extra violation in UniqueEOPL that does not appear in EndOfPotentialLine changes the problem significantly. It introduces many new solutions whenever there are multiple lines in the instance, and so it is unlikely, in our view, that one could reduce EndOfPotentialLine to UniqueEOPL. We also believe it very unlikely that other problems in CLS, such as the KKT problem of finding an approximate stationary point of a multivariate polynomial, are in UniqueEOPL. Of course, there is little hope to unconditionally prove that UniqueEOPL $\subset$ EOPL, but we can ask for further evidence, such as oracle separations, to support the idea.

References

1. Ilan Adler and Sushil Verma. The linear complementarity problem, Lemke algorithm, perturbation, and the complexity class PPAD. Technical report, Manuscript, Depatrment of IEOR, University of California, Berkeley, CA 94720, 2011.
2. David Aldous. Minimization algorithms and random walk on the d-cube. *The Annals of Probability*, pages 403–413, 1983.
3. Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 3(1):133–181, 1922.
4. Nir Bitansky, Omer Paneth, and Alon Rosen. On the cryptographic hardness of finding a Nash equilibrium. In *Proc. of FOCS*, pages 1480–1498, 2015.
5. Nir Bitansky, Omer Paneth, and Alon Rosen. On the cryptographic hardness of finding a Nash equilibrium. In *Foundations of Computer Science (FOCS), 2015 IEEE 56th Annual Symposium on*, pages 1480–1498. IEEE, 2015.
6. Ch. Boonyasiriwat, Kris Sikorski, and Ch. Xiong. A note on two fixed point problems. *J. Complexity*, 23(4-6):952–961, 2007.
7. Xi Chen and Xiaotie Deng. On algorithms for discrete and approximate Brouwer fixed points. In *Proc. of STOC*, pages 323–330, 2005.
8. Xi Chen and Xiaotie Deng. Matching algorithmic bounds for finding a Brouwer fixed point. *J. ACM*, 55(3):13:1–13:26, 2008.
9. Xi Chen and Xiaotie Deng. On the complexity of 2d discrete fixed point problem. *Theor. Comput. Sci.*, 410(44):4448–4456, 2009.
10. Xi Chen, Xiaotie Deng, and Shang-Hua Teng. Settling the complexity of computing two-player Nash equilibria. *J. ACM*, 56(3):14, 2009.
11. Anne Condon. The complexity of stochastic games. *Information and Computation*, 96(2):203–224, 1992.
12. Richard W Cottle, Jong-Shi Pang, and Richard E Stone. *The Linear Complementarity Problem*. SIAM, 2009.
13. Constantinos Daskalakis, Paul W Goldberg, and Christos H Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM Journal on Computing*, 39(1):195–259, 2009.
14. Constantinos Daskalakis and Christos Papadimitriou. Continuous Local Search. In *Proc. of SODA*, pages 790–804, 2011.
15. Constantinos Daskalakis, Christos Tzamos, and Manolis Zampetakis. A Converse to Banach's Fixed Point Theorem and its CLS Completeness. In *Proc. of STOC*, 2018.
16. Xiaotie Deng, Qi Qi, Amin Saberi, and Jie Zhang. Discrete fixed points: Models, complexities, and applications. *Math. Oper. Res.*, 36(4):636–652, 2011.
17. Jérôme Dohrau, Bernd Gärtner, Manuel Kohler, Jiří Matoušek, and Emo Welzl. ARRIVAL: a zero-player graph game in $\mathsf{NP} \cap \mathsf{coNP}$. In *A journey through discrete mathematics*, pages 367–374. Springer, Cham, 2017.
18. Kousha Etessami and Mihalis Yannakakis. On the complexity of nash equilibria and other fixed points. *SIAM J. Comput.*, 39(6):2531–2597, 2010.
19. Alex Fabrikant, Christos Papadimitriou, and Kunal Talwar. The complexity of pure Nash equilibria. In *Proc. of STOC*, pages 604–612. ACM, 2004.
20. John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. CLS: new problems and completeness. *CoRR*, abs/1702.06017, 2017. URL: http://arxiv.org/abs/1702.06017, arXiv:1702.06017.
21. John Fearnley, Spencer Gordon, Ruta Mehta, and Rahul Savani. End of potential line. *CoRR*, abs/1804.03450, 2018. URL: http://arxiv.org/abs/1804.03450, arXiv:1804.03450.
22. John Fearnley and Rahul Savani. The complexity of all-switches strategy improvement. In *Proc. of SODA*, pages 130–139, 2016.
23. Oliver Friedmann, Thomas Dueholm Hansen, and Uri Zwick. A subexponential lower bound for the random facet algorithm for parity games. In *Proc. of SODA*, pages 202–216, 2011.

24 Martin Gairing and Rahul Savani. Computing stable outcomes in hedonic games. In *Proc. of SAGT*, pages 174–185, 2010.
25 Sanjam Garg, Omkant Pandey, and Akshayaram Srinivasan. Revisiting the cryptographic hardness of finding a Nash equilibrium. In *Annual Cryptology Conference*, pages 579–604. Springer, 2016.
26 Bernd Gärtner. The random-facet simplex algorithm on combinatorial cubes. *Random Struct. Algorithms*, 20(3):353–381, 2002.
27 Bernd Gärtner, Thomas Dueholm Hansen, Pavel Hubáček, Karel Král, Hagar Mosaad, and Veronika Slívová. ARRIVAL: next stop in CLS. In *Proc. of ICALP*, pages 60:1–60:13, 2018.
28 Bernd Gärtner and Ingo Schurr. Linear programming and unique sink orientations. In *Proc. of SODA*, pages 749–757, 2006. URL: http://dl.acm.org/citation.cfm?id=1109557.1109639.
29 Bernd Gärtner and Antonis Thomas. The complexity of recognizing unique sink orientations. In *Proc. of STACS*, pages 341–353, 2015.
30 Thomas Dueholm Hansen and Rasmus Ibsen-Jensen. The complexity of interior point methods for solving discounted turn-based stochastic games. In *Conference on Computability in Europe*, pages 252–262, 2013.
31 Thomas Dueholm Hansen, Mike Paterson, and Uri Zwick. Improved upper bounds for random-edge and random-jump on abstract cubes. In *Proc. of SODA*, pages 874–881, 2014.
32 Michael D. Hirsch, Christos H. Papadimitriou, and Stephen A. Vavasis. Exponential lower bounds for finding Brouwer fix points. *J. Complexity*, 5(4):379–416, 1989.
33 Kathy Williamson Hoke. Completely unimodal numberings of a simple polytope. *Discrete Applied Mathematics*, 20(1):69–81, 1988.
34 Z. Huang, Leonid Khachiyan, and Krzysztof Sikorski. Approximating fixed points of weakly contracting mappings. *J. Complexity*, 15:200–213, 1999.
35 Z. Huang, Leonid G. Khachiyan, and Christopher (Krzysztof) Sikorski. Approximating fixed points of weakly contracting mappings. *J. Complexity*, 15(2):200–213, 1999.
36 Pavel Hubáček and Eylon Yogev. Hardness of continuous local search: Query complexity and cryptographic lower bounds. In *Proc. of SODA*, pages 1352–1371, 2017.
37 Emil Jeřábek. Integer factoring and modular square roots. *Journal of Computer and System Sciences*, 82(2):380–394, 2016.
38 David S Johnson, Christos H Papadimitriou, and Mihalis Yannakakis. How easy is local search? *Journal of Computer and System Sciences*, 37(1):79–100, 1988.
39 Gil Kalai. Three puzzles on mathematics, computation, and games. *CoRR*, abs/1801.02602, 2018. URL: http://arxiv.org/abs/1801.02602, arXiv:1801.02602.
40 Masakazu Kojima, Nimrod Megiddo, Toshihito Noma, and Akiko Yoshise. *A unified approach to interior point algorithms for linear complementarity problems*, volume 538. Springer Science & Business Media, 1991.
41 Masakazu Kojima, Nimrod Megiddo, and Yinyu Ye. An interior point potential reduction algorithm for the linear complementarity problem. *Mathematical Programming*, 54(1-3):267–279, 1992.
42 Carlton E Lemke. Bimatrix equilibrium points and mathematical programming. *Management science*, 11(7):681–689, 1965.
43 Jiří Matoušek and Tibor Szabó. Random edge can be exponential on abstract cubes. In *Proc. of FOCS*, pages 92–100, 2004.
44 Nimrod Megiddo and Christos H Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2):317–324, 1991.
45 Ruta Mehta. Constant rank bimatrix games are PPAD-hard. In *Proc. of STOC*, pages 545–554, 2014.
46 Walter D Morris Jr. Randomized pivot algorithms for P-matrix linear complementarity problems. *Mathematical programming*, 92(2):285–296, 2002.
47 Katta G Murty. Computational complexity of complementary pivot methods. In *Complementarity and fixed point problems*, pages 61–73. Springer, 1978.

48 A. Nemirovsky and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization.* Wiley, New York, 1983.

49 Christos H Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *Journal of Computer and System Sciences*, 48(3):498–532, 1994.

50 Anuj Puri. Theory of hybrid systems and discrete event systems. 1996.

51 Qi Qi. *Computational efficiency in internet economics and resource allocation.* PhD thesis, Stanford University, 2012.

52 Aviad Rubinstein. Settling the complexity of computing approximate two-player Nash equilibria. In *Proc. of FOCS*, pages 258–265, 2016.

53 Alejandro A Schäffer and Mihalis Yannakakis. Simple local search problems that are hard to solve. *SIAM journal on Computing*, 20(1):56–87, 1991.

54 Ingo Schurr and Tibor Szabó. Finding the sink takes some time: An almost quadratic lower bound for finding the sink of unique sink oriented cubes. *Discrete & Computational Geometry*, 31(4):627–642, 2004.

55 Ingo Schurr and Tibor Szabó. Jumping doesn't help in abstract cubes. In *Proc. of IPCO*, pages 225–235, 2005.

56 Spencer Shellman and Krzysztof Sikorski. A recursive algorithm for the infinity-norm fixed point problem. *Journal of Complexity*, 19(6):799 – 834, 2003.

57 Krzysztof Sikorski. *Optimal solution of Nonlinear Equations.* Oxford Press, New York, 200.

58 Krzysztof Sikorski. Computational complexity of fixed points. *Journal of Fixed Point Theory and Applications*, 6(2):249–283, 2009.

59 Alan Stickney and Layne Watson. Digraph models of bard-type algorithms for the linear complementarity problem. *Mathematics of Operations Research*, 3(4):322–333, 1978.

60 Tibor Szabó and Emo Welzl. Unique sink orientations of cubes. In *Proc. of FOCS*, pages 547–555, 2001.

61 Antonis Thomas. Exponential lower bounds for history-based simplex pivot rules on abstract cubes. In *Proc. of ESA*, pages 69:1–69:14, 2017.

62 Uri Zwick and Mike Paterson. The complexity of mean payoff games on graphs. *Theoretical Computer Science*, 158(1-2):343–359, 1996.