MATHEMATICS OF OPERATIONS RESEARCH
Vol. 00, No. 0, Xxxxxx 20xx, pp. xxx–xxx
ISSN 0364-765X | EISSN 1526-5471 |xx|0000|0xxx

DOI 10.1287/moor.xxxx.xxxx

Discrete Fixed Points: Models, Complexities and Applications

Xiaotie Deng[1]
Department of Computer Science, University of Liverpool, Liverpool, UK
email: xiaotie@liv.ac.uk http://www.csc.liv.ac.uk/~deng/

Qi Qi
Department of Management Science and Engineering, Stanford University, Stanford, California, USA
email: kaylaqi@stanford.edu

Amin Saberi[2]
Department of Management Science and Engineering, Stanford University, Stanford, California, USA
email: saberi@stanford.edu http://www.stanford.edu/~saberi/

Jie Zhang
Department of Computer Science, City University of Hong Kong, Hong Kong SAR, P. R. China
email: csjiezhang@gmail.com

In this work, we study three discrete fixed point concepts (SPERNER,DPZP,BROUWER) under two different models: the polynomial-time function model and the oracle function model. We fully characterize the computational complexities of these three problems. The computational complexity unification of the above problems gives us more choices in the study of different applications. As an example, by a reduction from DPZP, we derive asymptotically equal lower and upper bounds for TUCKER in the oracle model. The same reduction also allows us to derive a single proof for the *PPAD*-completeness of TUCKER in any constant dimension d, which is significantly simpler than the recent proof for the 2D case by Pálvölgyi, and for the 3D case and higher dimensions by Papadimitriou.

Key words: fixed point; algorithm; Sperner's lemma; Tucker's theorem.

MSC2000 Subject Classification: Primary: 91A99, 91-08; secondary: 68Q25, 55M20.

OR/MS subject classification: Primary: Games, Analysis of algorithms; secondary: Mathematics/Fixed points.

History: Received: July 13, 2010; Revised: June 02, 2011.

1. Introduction. Brouwer's fixed point theorem [4], together with its several variations, has played an important role in the development of applied mathematical sciences such as approximation theory [39], dynamical systems [44], equilibrium theory [41, 1], numerical computation [49], and most recently, computer networks [14, 37, 35]. Furthermore, speedup in the fixed point computation problems has been very important in applications such as signal processing [26] and Internet search [3, 29].

Even though fixed point theorems are mainly concerned with continuous functions, they are strongly tied to central problems in combinatorics. Sperner's lemma [48] identifies probably the simplest combinatorial fact based on which the fixed point theorems are obtained. It is a parity claim stating that, for any labeling of points on a circle with $\{0, 1\}$, there must be an even number of intervals with one end labeled with 0 and another with 1. (Such intervals are called Sperner's fully-colored simplices in one dimension). The lemma naturally extends to higher dimensions.

[1]Xiaotie Deng is supported by a grant from School of Electrical Engineering, Electronics and Computer Science, University of Liverpool, by the Research Grants Council of Hong Kong under Project No. RGC CityU 112909. The author is on leave from the department of computer science, City University of Hong Kong, Kowloon, Hong Kong, where some of preliminary work was conducted.
[2]Amin Saberi is supported by the National Science Foundation and a Sloan Research Fellowship.

We define SPERNER to be the problem of finding a fully-colored simplex in a triangulated simplex satisfying certain boundary conditions. Besides SPERNER, there are several other discrete versions of the fixed point problem, each defined with respect to different applications. Here we consider two more of them: DPZP and BROUWER. DPZP is defined for the computation of a discrete zero point of direction preserving functions satisfying the standard boundary conditions. It was first introduced by Chen and Deng [5] following Iimura [27]. Another version, called BROUWER, introduced by Daskalakis et al. [16], was used to settle the computational complexity of two player Nash equilibrium by Chen and Deng [8].

Fixed point theorems have a wide range of applications. For example, Meunier [40] provided a constructive proof to the necklace problem using a version of the fixed point theorem, the cubical version of Tucker's lemma found by Ky Fan. Eaves [18] used Brouwer's fixed point theorem to extend the existence theorem of Lemke in linear complementarity theory to the nonlinear case. Freund [21, 22] explored the path-following properties of the variable dimension complex to prove several combinatorial topology lemmas, including Sperner's lemma, KKM lemma, Tucker's lemma, etc. Herings et al. [24] developed a simplicial algorithm to compute constrained equilibria of an exchange economy. Su [50] showed that the Sperner's lemma approach can be used to treat chore division and rent-partitioning. More applications can be found in [2, 23, 51, 53, 55, 56].

The computational complexity of such fixed point problems is interesting only when the problem size is exponentially large with respect to input parameters. Otherwise, there are trivial polynomial time solutions that use exhaustive search. We will focus on the problem of size 2^{C*n} specified by an input parameter n and a constant C.

We need to represent such input data succinctly. We consider two models: *the oracle model* and *the polynomial function model*.

For the oracle model, we treat the function as a black box that outputs the function value for every domain variable once a request is sent in to the oracle. For example, given a node in the simplicial triangulation in an input of SPERNER, the oracle outputs the color of that node. The number of times the oracle is asked for the function value is defined as the query complexity of the problem. For the polynomial function model, on the other hand, the input is an algorithm that gives the answer for the function value on the input data in time polynomial in the input parameter n. Alternatively, the polynomial time algorithm can be replaced, for our problems, by a polynomial size logical circuits consisting of gates $\{AND, NOT, OR\}$ of Boolean variables.

Both the oracle and polynomial function models arise naturally in the applications of fixed point theorems in game theory and economics. Consider the problem of computing price equilibria in a market. For this problem, you need to represent the utility functions of the individuals. You can represent the utilities with explicit functions (linear, concave, etc..) or in general with a function that can be computed in polynomial time. Another option is to represent the utilities by a black box, where the individuals express their utility value for a bundle once the bundle is presented to them. The former gives rise to the polynomial function model and the latter to the oracle function model. The same issue also arises in the computation of Nash equilibria in games, where the utility function of each player is a function of his own actions as well as the actions of the other players.

Closer examination of the proofs of existence for SPERNER, BROUWER, DPZP reveals that they are all based on an embedded parity argument. In exploring the characterization of those parity properties and their computation, Papadimitriou [43] introduced the computational class *PPAD* (standing for Polynomial Parity Argument, Directed version) for the polynomial function model. The first *PPAD*-complete problem is the **End of Line** problem. It is based on an exponential size graph, consisting of directed paths and cycles. One starting vertex of some direction path is given as one of the input. The problem is to find any end vertex or another starting vertex different from the given one. A crucial component to the input of this problem under the polynomial-time function model is a polynomial time algorithm which given any vertex in the exponential graph, outputs its incoming edge and outgoing edge.

The restriction of a polynomial function in the definition of the class *PPAD* limits the number of functions under consideration. Because of that, it is desirable to consider computational models that admit more general functions, such as the **oracle function** model. The path following algorithm, initiated in the seminal work by Lemke and Howson [33] and found its usefulness in many related problems, is known to find the path's end point in an exponential number of steps even in the best starting direction

	PPAD-Completeness	Oracle Model Complexity
SPERNER	3D: Papadimitriou'94 [43] 2D: Chen & Deng'06 [6]	2D Lower bound: Crescenzi and Silvestri'98 [15] 2D Upper bound: Friedl, et al.'05 [20] Other dimensions: this paper
BROUWER	3D: Daskalakis, etc'06 [16] 2D: Chen & Deng'06 [6]	this paper
DPZP	this paper	Chen & Deng'05 [5]

Table 1: Previous Results

for some instances of the bimatrix game problem [45]. Our concern here is to determine a lower bound for any algorithm under the oracle model in terms of the worst case complexity.

In summary, our study explores the computational complexity of finding the three discrete versions of the fixed points SPERNER, DPZP and BROUWER under both models.

1.1 Previous Results. In introducing the class *PPAD*, Papadimitriou found many *PPAD*-complete problems including 3D Sperner's lemma, Brouwer and Kakutani's fixed point problems, as well as exchange economy [43]. In the last five years, there have been intensive interests in proving *PPAD*-completeness of important problems, such as 2D SPERNER [6], Nash equilibrium [16, 8], approximate-NASH [9], the Tucker's problem [42], envy-free cake cutting [17], Scarf's lemma and core of balanced games [30]. The previous results on the computational complexities of the three discrete versions of the fixed points are shown in Table 1.

Hirsch et al. [25] gave the first non-trivial bound for the oracle model on the time complexity for finding a fixed point. Modifying Iimura's direction preserving function concept, Chen and Deng [5] closed the exponential gap left in the work of Hirsch, et al., to derive a matching bound (that is asymptotically equal lower and upper bounds) in the oracle model for the problem DPZP. However, the study of the oracle model has been mainly focused on the DPZP problem, or other closely related models. In that line of research, Chen and Teng [12] recently proved a randomized matching bound for DPZP, and together with Sun [11], an almost matching quantum algorithmic bound under the oracle function model.

For SPERNER, Friedl et al. [20] developed an $O(\sqrt{n})$ deterministic query algorithm for a regular triangulation, that matches a deterministic lower bound by Crescenzi and Silvestri [15]. Before that, SPERNER was shown to be *PPAD*-complete by Papadimitriou [43]. BROUWER was proved to be *PPAD*-complete by Daskalakis, et al. [16] both for 3D and above. Chen and Deng [6] improved both results to show their *PPAD*-completeness for the 2D case, concluding this line of work for these problems. For DPZP, its complexity in the polynomial function model was not known previously.

Our current work fills in the unknown cases to complete the picture for the computational complexity of SPERNER, BROUWER, and DPZP. We believe that our analysis gives a better understanding of these problems and paves the way for using them for a wider range of applications. As an example, we build on our reductions to give a matching algorithmic bound for TUCKER in the oracle model and prove its *PPAD*-completeness in all dimensions. Our proof for *PPAD*-completeness is significantly simpler than the recent proof for the 2D case [42], and higher dimensions [43].

1.2 Our Contributions and Applications. Our main contribution is to develop matching upper and lower bounds for several important problems related to the discrete fixed point problem. Our first step in obtaining these bounds is to prove *PPAD*-completeness for DPZP by establishing a one-to-one correspondence of the solutions to BROUWER. Then, we use this technique to derive the oracle complexity of BROUWER, SPERNER, and TUCKER.

- *The PPAD-Completeness proof of DPZP.*
  We find a reduction between DPZP and BROUWER that shows their computational complexities are within a constant factor of each other. By the *PPAD*-completeness of BROUWER, DPZP is also *PPAD*-complete.
- *The matching oracle complexity of BROUWER for any constant dimension* $d \geq 2$.
  This result follows immediately from the reduction structures we find and the already known

matching oracle algorithmic bound for DPZP.

- *The matching oracle complexity of SPERNER for any constant dimension $d > 2$.*

  The upper bound is based on an interesting property of indices used for fixed point computation. For the lower bound, we find a complexity preserving reduction from BROUWER to SPERNER.

  All the reductions among the three above problems increase the problem size by a constant factor. Therefore, the results for randomized and quantum oracle function complexity immediately apply to all three problems.

- a) *The matching oracle complexity of TUCKER for any constant dimension $d \geq 2$.*

  b) *A unified proof for the PPAD-Completeness of TUCKER for any constant dimension $d \geq 2$.*

  As an example of an application, we design a reduction from DPZP to TUCKER. Tucker's theorem states that for any triangulation of a hypercube T that is antipodal preserving on the boundary, there exists a 1-simplex (an edge) in T that is complementary, i.e., its two vertices are labeled by opposite numbers.

  Our reduction gives a single proof for all constant dimensions. The connection also allows us to derive a lower bound in oracle complexity for the computation of TUCKER, for all constant dimensions. The upper bound requires a combinatorial parity lemma that relates the existence of the complementary edges on the boundary with certain parity property of the fully-colored simplices on the boundary of the hypergrid.

  The *PPAD*-completeness of TUCKER was shown for 3D by Papadimitriou [43], who also noticed that there was a similarity in difficulties to attempt *PPAD*-complete proofs for 2D SPERNER and 2D TUCKER. For 2D SPERNER, the difficulty was overcome later by Chen and Deng [6]. The recent proof of *PPAD*-completeness for 2D TUCKER by Pálvölgyi [42] exploits the techniques developed by Chen and Deng for 2D SPERNER to 2D TUCKER. Our reduction from DPZP is much simpler, and can also be applied to study the oracle complexity of TUCKER.

Traditionally, the algorithmic approach for fixed point computation and related problems such as Nash equilibrium computation has been based on the path following paradigm, established since Lemke-Howson's algorithm [33] for 2-player Nash equilibrium computation. On the other hand, our approach, especially for the oracle model, has been mainly a binary search approach utilizing the parity.

1.3 Structure of the Presentation. In Section 2, we formally introduce the complexity classes, as well as the definitions of the three discrete versions of fixed points. In Section 3, we develop reductions from DPZP to BROUWER, and back, to establish *PPAD*-completeness of DPZP and the oracle matching algorithmic bound for BROUWER. In Section 4, we prove the oracle matching algorithmic bound for SPERNER.

To demonstrate their applications, we discuss TUCKER in Section 5. In Section 6, we conclude our work with remarks on our results and discussion on related results and open problems.

2. Preliminaries. In this section, we formally introduce the necessary complexity models, the concept index (to be defined later) that is crucially important for the study of fixed points, as well as three different versions of discrete fixed points: SPERNER, BROUWER and DPZP.

We will use the terminologies of base hypercube and base simplex to refer to a smallest sub-hypercube and sub-simplex in the partition of a d-dimensional object into sub-hypercubes and sub-simplices.

2.1 Computational Complexity Models. Under the given boundary conditions, the SPERNER, DPZP and BROUWER are guaranteed to have a solution. Problems with such properties for the associated functions are called search problems, which are placed in the class $TFNP$ of the computational complexity theory, by Megiddo and Papadimitriou [38]. More generally, a binary relation $P(\cdot, \cdot)$ is in the class FNP iff, given x, if there exists some y no longer than a polynomial in $|x|$, such that $P(x, y)$ holds, then it can be verified in polynomial time. It is in $TFNP$ iff, for every x, there is always some y of length at most polynomial in $|x|$ such that $P(x, y)$ holds and can be verified in polynomial time.

One of the subclasses in $TFNP$, the class *PPAD* is based on an exponential size graph, consisting of directed paths and cycles. The input graph is not given explicitly. Instead, the problem input is a polynomial function that given any vertex in the graph, can compute the incoming edge or the outgoing

edge. Another input of the problem is one starting vertex in a directed path. The required output is an end vertex of any directed path or a starting vertex of another directed path different from the given one. Since the function for computing the incoming or the outgoing edge is given explicitly (e.g. with a polynomial size logical circuit consisting of $\{AND, NOT, OR\}$ gates), this model is called the polynomial function model.

Another form of input is the oracle function model. In this model, the input is given via a black box or an oracle. At each step, one can submit a query to an oracle. For example, a value x is given to the oracle to obtain the function value $f(x)$. Each return of the function value is counted as an oracle step. If the oracle is queried for the second time for the same variable value x, it must return the same function value $f(x)$. Even though an oracle enables us to quickly access the function values, it can still make the problem harder than under the polynomial function model. The main reason is that the oracle function model can admit a larger class of functions.

2.2 Triangulation and Index. All the fixed point concepts mentioned above are originated from Sperner's lemma and its generalizations (see, e.g., [52]). We will review the concept of index for its important and fundamental properties in discrete fixed point computations (see, e.g., [46, 52]). Then we state the main results that will be used in the subsequent discussion.

In the two dimensional space, a triangular grid of scale 1 is an ordinary triangle Δ which has three vertices and one base cell. A triangular grid of scale N places $N-1$ equally spaced line segments parallel to each of the three edges of Δ and divides the triangle into N^2 base cells.

We refer to the three vertices of Δ as *corner vertices*, and denote them by D_0, D_1, and D_2. The vertices along the edges of Δ are referred to as *boundary vertices*. Other vertices are referred to as *internal vertices*. Edges of a base cell are referred to as base edges. Each vertex $x = (i \times D_0 + j \times D_1 + k \times D_2)/N$ is represented by (i, j, k) where $i, j, k \geq 0$, $i + j + k = N$. We call it the *barycentric coordinates* of the vertex.

With barycentric coordinates, D_0 is represented by $(N, 0, 0)$; D_1 by $(0, N, 0)$ and D_2 by $(0, 0, N)$. Boundary vertices along D_0 and D_1 are in the form $(i, j, 0)$ with $i, j > 0, i + j = N$. Other boundary vertices are defined similarly. For any interior point represented by (i, j, k) we have $i, j, k > 0, i+j+k = N$. Let $V = \{(i, j, k) : i, j, k \geq 0, i + j + k = N\}$.

Base cells in the triangulation are oriented in the clockwise order of their vertices and base edges are oriented according to the clockwise order of their base cells. See **Fig. 1** below.

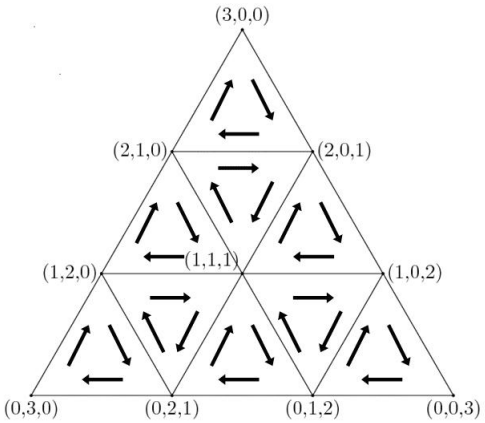

Figure 1: Base triangle with edge orientation

A coloring $\phi : V \to \{0, 1, 2\}$ is a Sperner coloring if and only if for any vertex $x = (x_0, x_1, x_2)$, $\phi(x_0, x_1, x_2) = j \in \{0, 1, 2\}$ implies $x_j > 0$. Sperner's lemma states that a triangulated triangle with a

valid Sperner coloring has a base cell such that its three vertices have different colors.

Given a Sperner coloring $\phi : V \to \{0, 1, 2\}$ of all vertices in V, let $sign(\delta, \phi)$ and $sign(e, \delta, \phi)$ denote the sign of a base cell δ and the sign of a base edge e in δ respectively. The sign of $e = (u, v)$ is 1 (or -1) if the colors of its two vertices are 0 and 1 and the orientation of e in δ is from color 0 vertex to color 1 vertex (or from 1 to 0). We denote the sign of a base edge by $sign(e, \phi)$ if there is no ambiguity in its orientation. In all other cases, $sign(e, \phi) = 0$. The sign of a base cell δ is defined to be the sum of the signs of its three base edges. Therefore, $sign(\delta, \phi) = sign(e_1, \delta, \phi) + sign(e_2, \delta, \phi) + sign(e_3, \delta, \phi)$.

We may verify the following by a simple case analysis.

PROPOSITION 2.1 *For any base cell, its sign is* 1 *(or* -1*) if and only if its three vertices are colored with* $0, 1, 2$ *in the clockwise (counterclockwise) order. In all other cases, its sign is zero.*

The index of a connected set Δ of base cells, with respect to the coloring ϕ is defined as:

$$index(\Delta, \phi) = \sum\{sign(\delta, \phi) : \delta \text{ is a base triangle} \in \Delta\}$$

LEMMA 2.1 *[52] For a triangulated triangle* Δ *with colors* $\phi : V \to \{0, 1, 2\}$*, there are at least* $|index(\Delta, \phi)|$ *fully-colored base cells. The index can be calculated by summing the signs over its boundary base edges.*

PROOF. By Proposition 2.1 and the definition of $index(\Delta, \phi)$, the statement that there are at least $|index(\Delta, \phi)|$ fully-colored base triangles is obviously true.

Note that every internal base edge belongs to two base triangles and has different signs with respect to the two base triangles. Therefore, they cancel each other and derive the following:

$$index(\Delta, \phi) = \sum\{sign(e, \phi) : e \text{ is a boundary base edge}\},$$

which completes the proof. Note that in the above sum, the orientation of the boundary base edges are in the clockwise order around Δ. □

The result also holds for general polygons in 2D. For a higher dimensional polyhedron P, we consider a simpler version of index that is defined in $GF(2)$ and was used in [7]. For a d-dimensional simplex with vertices assigned $d+1$ different colors $\{0, 1, \cdots, d\}$, we define its index as 1. Otherwise, it is defined to be zero. Let $V(P)$ be the vertices of its triangulation. With respect to a coloring $\phi : V(P) \to \{0, 1, \cdots, d\}$, its index is defined as $index(P, \phi) \equiv \sum_{\delta \in P} index(\delta, \phi) \mod 2$, where δ's are d-dimensional base simplices in the triangulation of P.

Denote by ∂P the boundaries of P. Note that the triangulation of P induces a triangulation of ∂P into $(d-1)$-dimensional simplices. We define $index_{d-1}(\partial P, \phi) \equiv \sum_{\delta_{d-1} \in \partial P} index_{d-1}(\delta_{d-1}, \phi) \mod 2$.

We need the standard result for the index defined here (see, e.g., [52, 5]). For completeness, we sketch a proof here.

PROPOSITION 2.2 $index(P, \phi) \equiv index_{d-1}(\partial P, \phi) \mod 2$.

PROOF. Let's start by proving the claim when P is a d-dimensional simplex δ with no further triangulation. In that case, $index_d(\delta, \phi)$ is 1 if and only if the set of colors of its $d+1$ vertices are all distinct and is the same as $\{0, 1, \cdots, d\}$. To verify the equality note that δ has $d+1$ faces of dimension $d-1$ which are simplices of dimension $d-1$. A $(d-1)$-dimensional simplex has index 1 if and only if the set of colors of all its vertices is the same as $\{0, 1, \cdots, d-1\}$.

First, assume $index_d(\delta, \phi)$ is 1. Then, exactly one of its faces has colors $\{0, 1, \cdots, d-1\}$ whose index is 1. Hence, $index_{d-1}(\partial\delta, \phi) = 1$. The claim follows.

Second, assume P has at least one color missing and its index is zero. In this case, the claim holds if all its faces have index zero. If there exists a face with all the colors $\{0, 1, \cdots, d-1\}$. The only vertex

not in this face must be colored by i: $i < d$. Then we will have exactly one more face with index 1. Summing up the indices of all the faces, we obtain a sum of 2 which is 0 mod 2.

In general, $index(P, \phi) \equiv \sum_{\delta \in P} index(\delta, \phi) \mod 2$. We can replace $index(\delta, \phi)$ by the sum of indices of the boundaries of δ. However, each $(d-1)$-dimensional simplex in the triangulation appears in exactly two d-dimensional simplices in the triangulations, unless it is at the boundary of P where it appears only once. Since we consider the sum mod 2, all the terms cancel out except those on the boundaries of P. The claim follows. □

2.3 SPERNER. We start with one of the *PPAD*-complete problems, SPERNER, defined as follows. SPERNER is based on a triangulation of a d-dimensional simplex with a coloring scheme $c(i)$ on its vertices satisfying the following conditions:

(i) $c(i) \in \{0, 1, 2, \cdots, d\}$

(ii) For any vertex with barycentric coordinates $(x_0, x_1, \cdots, x_d)$, if $x_i = 0$, then $c(x_i)$ is not i. In general, this condition can be replaced by one that the boundary has an odd index.

A base simplex is called a Sperner simplex if the labels of its vertices contain all colors. Finding a Sperner simplex is a search problem that is in *PPAD* [43].

In addition, we define the concept of balanced triangulations:

DEFINITION 2.1 *A simplex P is triangulated into balanced simplices of granularity $g = \frac{1}{N}$ if*

(i) P is fully contained in the unit cube $[0, 1]^d$;

(ii) every parallel plane along the coordinates $x_i = g \times j$ $(1 \le i \le d, 0 \le j \le N)$ cuts through P along the facets of the base cells of the triangulation, i.e., the parallel plane will not cut through the base cells;

(iii) the number of the d-dimensional simplices of the triangulation within any base hypercube of side length g is constant.

It is important to note that one can find a balanced triangulation of a simplex in all dimensions. One example is Kuhn's triangulation [31], which will be introduced later. We should call this problem balanced SPERNER. Friedl, et al., considered a special case of balanced SPERNER for the 2-dimensional case, where base triangles are in the form of $\{(x, y), (x + \epsilon, y), (x, y + \epsilon)\}$ where $\epsilon = \pm 1$. They called it regular SPERNER [20].

(a) Regular & Balanced (b) Balanced

Figure 2: Regular and Balanced Triangulation

In **Fig. 2**, the triangulation on the left is regular, and hence balanced; the one on the right is balanced but not regular.

2.4 BROUWER. Another problem of *PPAD*-complete class, BROUWER, is defined as follows.

For simplicity, we use "d-hypercube" to stand for a d dimensional hypercube. Starting with a d-hypercube of side length N, we place $N - 1$ equally spaced hyperplanes of dimension $d - 1$ parallel to each of the boundary faces of the large d-hypercube and divide it into N^d base hypercubes, each of unit

side length. We call the resulting hypergrid a d-hypergrid of side length N (or scale N). We denote its node set by:

$$V_N^d = \{\mathbf{p} = (p_1, p_2, \cdots, p_d) \in \mathcal{Z}^d : 0 \leq p_i \leq N \text{ for all } i\}$$

Its boundary B_N^d consists of points $\mathbf{p} \in V_N^d$ with $p_i \in \{0, N\}$ for some i:

$$B_N^d = \{\mathbf{p} \in V_N^d : p_i \in \{0, N\} \text{ for some } i\}.$$

For each $\mathbf{p} \in \mathcal{Z}^d$, let

$$\mathcal{K}_{\mathbf{p}} = \{\mathbf{q} : q_i \in \{p_i, p_i + 1\}\}.$$

We define $m(\mathbf{p}) = 0$ if $\forall i, p_i > 0$ and $m(\mathbf{p}) = \min\{i : p_i = 0\}$ otherwise. A coloring function $g : V_N^d \Rightarrow \{0, 1, \cdots, d\}$ is valid if $g(\mathbf{p}) = m(\mathbf{p})$, $\forall \mathbf{p} \in B_N^d$.

DEFINITION 2.2 *The input of BROUWER is a pair (G, V_N^d) where G generates a valid d-coloring g on V_N^d. The output of BROUWER is a point $\mathbf{p} \in V_N^d$ such that its corresponding cube $\mathcal{K}_{\mathbf{p}}$ is fully-colored, that is, $\mathcal{K}_{\mathbf{p}}$ has all $d + 1$ colors.*

Note that given any $\mathbf{p} \in V_N^d$, G computes the color $g(\mathbf{p})$ of a point $\mathbf{p}$. If G takes polynomial time to generate the color of a node, the problem is called the polynomial version of BROUWER. If G is a functional oracle, we call it the oracle version of BROUWER. In the former case, we sometimes simply state it as BROUWER, where there is no ambiguity.

It is known that BROUWER is *PPAD*-complete [43, 6] when G is a polynomial time algorithm.

2.5 Direction Preserving Zero Point. We consider a class of functions known as the direction preserving functions originally introduced by Iimura [27]. Our focus is on a discrete version as follows:

We should denote by $\mathbf{e}_i$ a unit vector having one on its i-th entry. And $\mathbf{0}$ is the vector of all zeros.

DEFINITION 2.3 *[5, 27] A function $f : V_N^d \Rightarrow \{\mathbf{0}, \pm\mathbf{e}_i, i = 1, 2, \cdots, d\}$ is direction preserving if for any $\mathbf{p} \in V_N^d$, $\mathbf{s}, \mathbf{t} \in \mathcal{K}_{\mathbf{p}}$ and $f(\mathbf{s}) \neq \mathbf{0}$, $f(\mathbf{s}) + f(\mathbf{t}) \neq \mathbf{0}$.*

A direction preserving function f on V_N^d is *bounded* if $\forall \mathbf{p} \in V_N^d$, $\mathbf{p} + f(\mathbf{p}) \in V_N^d$. We define the DPZP (Direction Preserving Zero Point) problem, following Chen and Deng [5], as follows:

DEFINITION 2.4 *[5] The input of DPZP is a pair (F, V_N^d) where F generates a bounded direction preserving function f on V_N^d. The output of DPZP is a point $\mathbf{p} \in V_N^d$ such that $f(\mathbf{p}) = \mathbf{0}$, which is called a zero point.*

Again, for the polynomial version, given any $\mathbf{p} \in V_N^d$, F takes time polynomial in N to compute the function value $f(\mathbf{p})$ of a point $\mathbf{p}$. We drop "polynomial" where there is no ambiguity. In the oracle version F is a functional oracle.

It is known that a bounded direction preserving function f on V_N^d always has a zero point [27, 28], and there is a matching algorithmic bound [5] for finding it in the oracle model.

3. *PPAD*-completeness of Direction Preserving Zero Point. In this section, we assume F is a polynomial time algorithm in the DPZP problem. We prove that DPZP is *PPAD*-complete and give a matching bound for the oracle version of BROUWER.

We will first introduce a reduction from DPZP to BROUWER. We then introduce a reduction from BROUWER to DPZP. As both reductions are polynomial time with respect to the algorithms F and G, they show that DPZP is *PPAD*-complete when F is a polynomial time algorithm. Also, since the oracle version of DPZP has a matching algorithmic bound of $\theta(N^{d-1})$ [5], the oracle version of BROUWER has the same matching algorithmic bound of $\theta(N^{d-1})$.

LEMMA 3.1 *DPZP is in* PPAD.

PROOF. We prove it in two steps. First we add one more layer to any given input of DPZP to make sure of a specific boundary condition. Then we reduce the expanded DPZP to BROUWER.

Given an input of DPZP (F, V_N^d), first we add one more layer to each face to expand V_N^d to V_{N+2}^d. Then, we define a polynomial time algorithm H which generates a function h on V_{N+2}^d as follows: (i) for each inner vertex $\mathbf{p} \in V_N^d$, let $h(\mathbf{p}) = f(\mathbf{p})$. (ii) for a boundary point $\mathbf{p}$ for which $p_i = 0$, find the smallest i with that property and let $h(\mathbf{p}) = \mathbf{e}_i$. (iii) if all coordinates of $\mathbf{p}$ are non-zero, then there is an i such that $p_i = N+2$ and $\forall j < i : p_j < N+2$. In that case, $h(\mathbf{p}) = -\mathbf{e}_i$. It is easy to verify that (H, V_{N+2}^d) is still a DPZP.

Next, we reduce DPZP (H, V_{N+2}^d) to BROUWER (G, V_{N+2}^d) as follows: define $g(\mathbf{p}) = i$ if $h(\mathbf{p}) = \mathbf{e}_i$; $g(\mathbf{p}) = 0$ otherwise.

We illustrate the processes in **Fig. 3**.

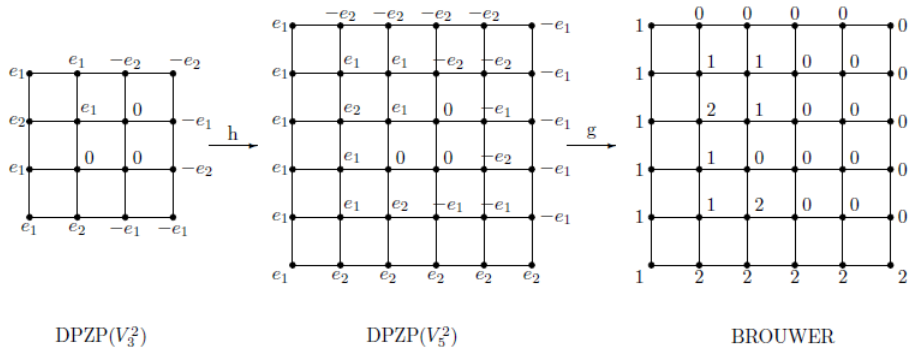

Figure 3: Proof of Lemma 3.1

Clearly, a direction preserving function h satisfying a bounded specific boundary condition for DPZP translates into a valid coloring for BROUWER. For a solution $\mathcal{K}_\mathbf{p}$ to BROUWER, we should have $\mathbf{q} \in \mathcal{K}_\mathbf{p}$ such that $g(\mathbf{q}) = 0$. Then $h(\mathbf{q}) \in \{\mathbf{0}, -\mathbf{e}_1, -\mathbf{e}_2, \cdots, -\mathbf{e}_d\}$ by the process of our reduction. For $\mathbf{t} \in \mathcal{K}_\mathbf{p}$ and $g(\mathbf{t}) = i \neq 0$, we have $h(\mathbf{t}) = \mathbf{e}_i$. Since all colors appear in $\mathcal{K}_\mathbf{p}$, by direction preserving property of h, we must have $h(\mathbf{q}) = \mathbf{0}$, and hence $\mathbf{q}$ is a solution to DPZP. □

LEMMA 3.2 *DPZP is* PPAD*-hard.*

PROOF. Given an input of BROUWER (G, V_N^d), we define an input for DPZP on an expanded hypergrid. The expanded hypergrid is obtained by placing $2N$ more hyperplanes parallel to each of the boundary faces to refine each base hypercube in V_N^d into 3^d smaller hypercubes.

The main idea is to assign values to grid points to maintain the following properties:

(I) A valid coloring g in BROUWER is reduced into a bounded direction preserving function f for DPZP;

(II) After finding a zero point in DPZP, we can get a corresponding fixed point set in BROUWER.

With a slight abuse of notation, we denote the input of the DPZP as $(F, \frac{1}{3}V_{3N}^d)$ in that each point $\mathbf{q}$ in the input for DPZP is of the form $\mathbf{q} = \frac{\mathbf{p}}{3}$ for some $\mathbf{p} \in V_{3N}^d$.

To provide an intuitive understanding, we first illustrate the reduction process for the two dimensional case ($d = 2$). The reduction process for any dimension $d > 2$ can be generalized with ease, which will be discussed afterwards with focus on the differences.

A $\mathcal{K}_\mathbf{p}$ in the original space of V_N^2 generates sixteen points in the expanded space $\frac{1}{3}V_{3N}^2$: $\mathcal{K}'_\mathbf{p} = \{(p_1 + \frac{i}{3}, p_2 + \frac{j}{3}) : i, j = 0, 1, 2, 3\}$ (See **Fig. 4**). For any point $\mathbf{q} \in \mathcal{K}'_\mathbf{p}$, we denote by $cube(\mathbf{q})$ the smallest integral hypercube that contains it. That is

$$cube(\mathbf{q}) = \{(r_1, r_2)^T : r_i \in [\lfloor q_i \rfloor, \lceil q_i \rceil], i = 1, 2\}$$

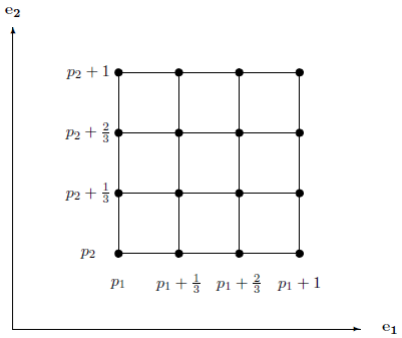

Figure 4: $\mathcal{K}'_p$

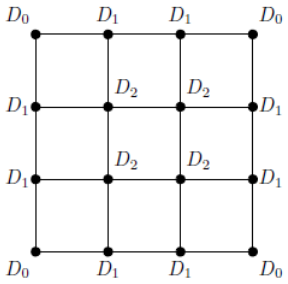

Figure 5: Classification of Vertices

We denote by $dim(\mathbf{q})$ the dimension of $cube(\mathbf{q})$. Obviously, $dim(\mathbf{q})$ is the number of coordinates of $\mathbf{q}$ that are not integers. Let $\mathcal{K}(\mathbf{q})$ denote the $2^{dim(\mathbf{q})}$ corner points of $cube(\mathbf{q})$.

We classify the vertices $\mathbf{q}$ in $\mathcal{K}'_{\mathbf{p}}$ to three types: (i) use D_0 to denote the vertex that all of its coordinates are integers; (ii) use D_1 to denote the vertex that exactly one of its coordinates is integer; (iii) use D_2 to denote the vertex that none of its coordinates are integers. Clearly, $cube(\mathbf{q}) = \{\mathbf{q}\}$ if $\mathbf{q} \in D_0$; $cube(\mathbf{q})$ contains four points in $\mathcal{K}'_{\mathbf{p}}$ on a line if $\mathbf{q} \in D_1$; and $cube(\mathbf{q})$ contains all sixteen points in $\mathcal{K}'_{\mathbf{p}}$ if $\mathbf{q} \in D_2$ (See **Fig. 5**).

We should derive a bounded direction preserving function $f : \frac{1}{3}V^2_{3N} \to \{\mathbf{0}, \pm\mathbf{e}_1, \pm\mathbf{e}_2\}$, from the valid coloring g for the input instance of BROUWER. Let $closest(\mathbf{q})$ be the closest point to $\mathbf{q}$ among those in $\mathcal{K}(\mathbf{q})$. Note that $closest(\mathbf{q}) = \mathbf{q}$ if $\mathbf{q} \in D_0$.

For $\mathbf{q} \in \mathcal{K}'_{\mathbf{p}}$, we define $colorset(\mathbf{q}) = \{i : i = g(\mathbf{r}) \text{ for all } \mathbf{r} \in \mathcal{K}(\mathbf{q})\}$. Next we define the function f by considering the following three cases:

(i) If $colorset(\mathbf{q}) = \{0, 1, 2\}$, we set $f(\mathbf{q}) = \mathbf{0}$ (See **Fig. 6**);

(ii) If $0 \notin colorset(\mathbf{q})$, set $f(\mathbf{q}) = \mathbf{e}_{g(\mathbf{p})}$, where $\mathbf{p} = closest(\mathbf{q})$ (See **Fig. 7**);

(iii) Otherwise($0 \in colorset(\mathbf{q})$, there exists some $i \leq 2$ such that $i \notin colorset(\mathbf{q})$). Let $j = \min\{i : i \notin colorset(\mathbf{q})\}$ and set $f(\mathbf{q}) = -\mathbf{e}_j$ (See **Fig. 8**).

To verify the correctness of the reduction, we need to prove **(I)** and **(II)**.

Proof of (I): For a vertex $\{\mathbf{r} \in D_0 : r_1 = 0\}$, we must have $g(\mathbf{r}) = 1$, so $f(\mathbf{r}) = \mathbf{e}_1$. By the definition of f, a vertex $\{\mathbf{u} \in D_1 : u_1 = 0\}$, $f(\mathbf{u}) = \mathbf{e}_1$. Therefore, $f(\mathbf{v}) = \mathbf{e}_1$ if $v_1 = 0$. Similarly, we have $f(\mathbf{v}) = \mathbf{e}_2$ if $v_2 = 0$ and $v_1 \neq 0$. For any other boundary vertex $\mathbf{q}$, since $0 \in colorset(\mathbf{q})$, we have $f(\mathbf{v}) = -\mathbf{e}_i$ for some i. Therefore f is bounded. Next, we prove f is direction preserving. Consider an integer point $\mathbf{p}$ and its corresponding $\mathcal{K}'_{\mathbf{p}}$ in DPZP in the following three cases:

(i) If for any integer vertex $\mathbf{q} \in \mathcal{K}'_{\mathbf{p}}$, $g(\mathbf{q}) \neq 0$, then for any vertex $\mathbf{r} \in \mathcal{K}'_{\mathbf{p}}$, $f(\mathbf{r}) \geq \mathbf{0}$. So $\mathcal{K}'_{\mathbf{p}}$ satisfies the direction preserving condition.

(ii) If $\mathcal{K}'_{\mathbf{p}}$ is fully-colored, then for any type D_2 vertex $\mathbf{r}$, $f(\mathbf{r}) = \mathbf{0}$; any type D_1 vertices on the

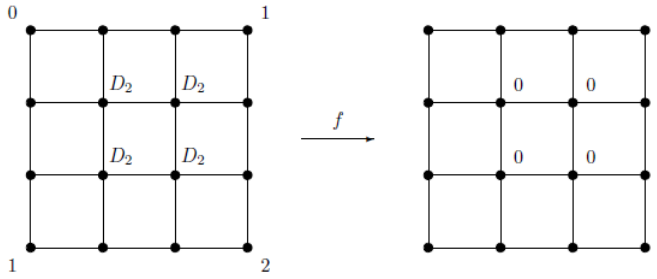

Figure 6: Case 1 in the proof of Lemma 3.2

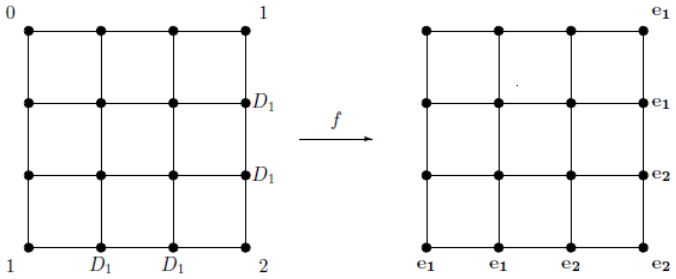

Figure 7: Case 2 in the proof of Lemma 3.2

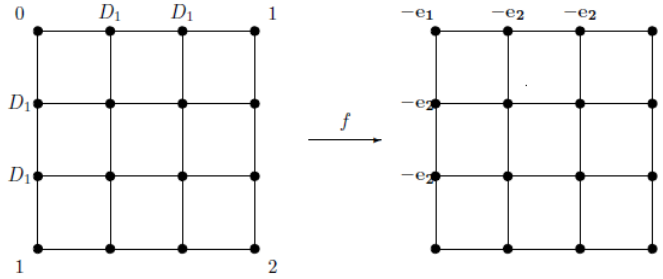

Figure 8: Case 3 in the proof of Lemma 3.2

same line have the same direction; any vertex adjacent to $\mathbf{p} : g(\mathbf{p}) = 0$ must have the negative direction. It follows that $\mathcal{K}'_{\mathbf{p}}$ satisfies the direction preserving condition.

(iii) Otherwise, let $j = \min\{i : i \notin colorset(\mathbf{q})\}$, where $\mathbf{q}$ is a type D_2 vertex. Then for any vertex $\mathbf{r} \in \mathcal{K}'_{\mathbf{p}}$, we must have $f(\mathbf{r}) > 0 \neq \mathbf{e}_j$ or $f(\mathbf{r}) = -\mathbf{e}_j$. Hence, the direction preserving condition holds.

Proof of (II): For a vertex $\mathbf{q}$, $f(\mathbf{q}) = 0$ if and only if $colorset(\mathbf{q}) = \{0, 1, 2\}$. Therefore, $\mathbf{q}$ is a zero point in DPZP only if $\mathcal{K}(\mathbf{q})$ is fully-colored in BROUWER.

To generalize the proof to any dimension d, we can use an analogous notation. For $\mathcal{K}_{\mathbf{p}}$ in the original space of V_N^d, its corresponding space $\frac{1}{3}V_{3N}^d$ is: $\mathcal{K}'_{\mathbf{p}} = \{(p_1 + \frac{v_1}{3}, \cdots, p_i + \frac{v_i}{3}, \cdots, p_d + \frac{v_d}{3}) : i = 0, 1, \cdots, d\}$. For any point $\mathbf{q} \in \mathcal{K}'_{\mathbf{p}}$, define

$$cube(\mathbf{q}) = \{(r_1, r_2, \cdots, r_d)^T : r_i \in [\lfloor q_i \rfloor, \lceil q_i \rceil]\}.$$

We use D_i, $i = 0, 1, \cdots, d$ to denote vertices that $d - i$ of their coordinates are integral, i.e., we classify all vertices $\mathbf{q}$ in $\mathcal{K}'_{\mathbf{p}}$ to $d + 1$ types.

We should derive a bounded direction preserving function $f : \frac{1}{3}V_{3N}^d \to \{\mathbf{0}, \pm\mathbf{e}_i : i = 1, 2, \cdots, d\}$. Let $closest(\mathbf{q})$ be the closest point to $\mathbf{q}$ among those in $\mathcal{K}(\mathbf{q})$. Let $\mathbf{q} \in \mathcal{K}'_{\mathbf{p}}$: $colorset(\mathbf{q}) = \{i : i = g(\mathbf{r}) \text{ for all } \mathbf{r} \in \mathcal{K}(\mathbf{q})\}$. The bounded direction preserving function f is defined as follows:

- If $colorset(\mathbf{q}) = \{0, 1, \cdots, d\}$, we set $f(\mathbf{q}) = 0$;
- If $0 \notin colorset(\mathbf{q})$, then set $f(\mathbf{q}) = \mathbf{e}_{g(\mathbf{p})}$, where $\mathbf{p} = closest(\mathbf{q})$;
- Otherwise($0 \in colorset(\mathbf{q})$, there exist some $i \leq d$ such that $i \notin colorset(\mathbf{q})$), let $j = \min\{i : i \notin colorset(\mathbf{q})\}$ and set $f(\mathbf{q}) = -\mathbf{e}_j$.

The correctness of the reduction can be proved in the same way as in the 2-D case. □

Combining the above two lemmas, we have

THEOREM 3.1 *Direction Preserving Zero Point(DPZP) is* PPAD*-complete.*

Since DPZP has a matching algorithmic bound of $\theta(N^{d-1})$ [5], by the above reduction processes from DPZP to BROUWER and from BROUWER to DPZP which can be done sufficiently efficient (within time $O(N^{d-1})$), we obtain the following corollary.

COROLLARY 3.1 *BROUWER has a matching algorithmic bound of* $\theta(N^{d-1})$.

4. Finding a Sperner Simplex under the Oracle Function Model. We will prove that the oracle complexity of finding a Sperner simplex in d dimensions under the oracle function model is $\theta(N^{d-1})$. Such a matching bound was known only for finding a Sperner simplex in a two dimensional $N \times N$ grid [15, 20]. Our extensions into higher dimensions make use of binary search for the upper bound, and of reduction from BROUWER for the lower bound. First, we derive the upper bound.

LEMMA 4.1 *For any balanced triangulation, there is an algorithm that finds a Sperner simplex in time* $O(N^{d-1})$.

PROOF. We fit the balanced triangulated simplex P into the unit cube $[0, 1]^d$ which is guaranteed by condition 1 of Definition 2.1. By condition 2, we use parallel planes along the coordinates to cut the cube, by the binary search method as follows.

First, consider the d cuts that divide the hypercube into 2^d sub-hypercubes of almost half the original side length. Compute the index of those 2^d sub-hypercubes and choose one of odd parity. Continue on this sub-hypercube and ignore the rest.

The correctness follows from Proposition 2.2 as the boundary conditions give an odd index for the initial simplex because of the Sperner coloring. The procedure can continue in this way until reaching the last base cube. Then, we can simply examine up to C remaining simplices contained in this base cube, where C is a constant by condition 3 of Definition 2.1.

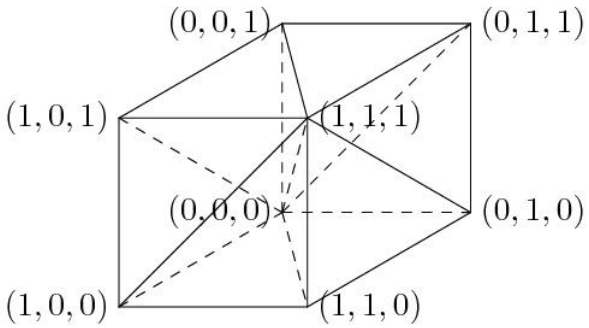

Figure 9: Kuhn's triangulation in 3 dimension

For query complexity, we have $q(N) \leq q(N/2)+dN^{d-1}$, resulting in $q(N) \leq 2dN^{d-1}$. For computational complexity, we have $t(N) \leq t(N/2) + 2^d \times 2d(\frac{N}{2})^{d-1} = t(N/2) + 4dN^{d-1}$, resulting in $t(N) \leq 8dN^{d-1}$. In $q(N)$ queries and $t(N)$ computational time we reduce the unit hypercube into a base cube of side length g. The computational upper bound follows. □

We also derive a similar lower bound:

LEMMA 4.2 *For any algorithm that finds a Sperner simplex for any triangulation of a d-dimensional simplex with a Sperner coloring, there exists some input triangulation such that the algorithm takes* $\Omega(N^{d-1})$ *time.*

To prove it, we will apply Kuhn's triangulation as defined below:

DEFINITION 4.1 *(Kuhn's Triangulation) Let a d-hypercube of side length N be located in the first quadrant such that one of its corner points is at the origin. Denote this vertex as* $\mathbf{v}_0 = (0, 0, \cdots, 0)_{1\times d}$ *and its diagonal vertex as* $(N, N, \cdots, N)_{1\times d}$*. Apply hyperplanes to cut the big hypercube into* N^d *unit d-hypercubes. Let* $\mathbf{e}_i$ *be the d-dimensional vector whose i-th coordinate is 1 and its other coordinates are 0. Let* $\pi = (\pi(1), \pi(2), \cdots, \pi(d))$ *be any permutation of integers* $1, 2, \cdots, d$*. Triangulate each unit hypercube in the following way: let the vertex which is the closest to* $\mathbf{v}_0$ *in each unit hypercube be the base point. Use each permutation* π *to create one base simplex whose vertices are given by* $\mathbf{v}_\pi^i = \mathbf{v}_\pi^{i-1} + \mathbf{e}_{\pi(i)}$*, where* $\mathbf{v}_\pi^0$ *is the base point of that unit hypercube.*

We illustrate Kuhn's triangulation for a 3-dimension hypercube with side length 1 in **Fig. 9**. It is easy to see that all vertices of the base simplices are the vertices of the hypercubes. We claim that these simplices all have disjoint interiors and their union is the d-hypercube. In fact, Kuhn's triangulation can be acquired by an equivalent cutting approach, that is, first using the $d!$ permutations π to triangulate the big hypercube to $d!$ simplices, and then triangulate each big simplex into N^d base simplices.

PROOF OF LEMMA 4.2. We establish a reduction from BROUWER to the Sperner problem of dimension d to derive a similar lower bound within a constant factor for any constant dimension d.

Given an input of BROUWER (G, V_N^d), we triangulate each base hypercube with side length 1 into $d!$ base simplices. Since Kuhn's triangulation doesn't generate any new vertices, the vertex set of the triangulated hypercube V_N^d is the same as the original.

Note that there are a total of $d!$ big simplices in such a triangulation, one for each permutation of the d coordinates. As the index for each input of BROUWER is one, there is at least one permutation for which the generated Kuhn's simplex has an odd index. Let T be the worst case complexity for an algorithm A to find a fully-colored base simplex for any odd indexed triangulated simplex. We may run A in series on all $d!$ simplices. Therefore, in time $d!T$, we will find a fully-colored base simplex. Since each base simplex is within a base hypercube of the original input, that gives a solution for BROUWER. By the lower bound of BROUWER, we have $d!T \geq cN^{d-1}$ for some constant $c > 0$. Since d is a constant, we obtain the lower bound $\Omega(N^{d-1})$ for Sperner (F, Δ_N^d).

□

Lemma 4.1 and Lemma 4.2 result in a matching bound as follows.

THEOREM 4.1 *(matching bound) Given a balanced triangulation where all vertices are colored by a Sperner coloring, finding a Sperner simplex takes time $\theta(N^{d-1})$.*

5. The Complexity of TUCKER. In this section, we prove that d-D TUCKER is *PPAD*-complete for any constant dimension d, and hence extend the results for 2-D TUCKER by [42] and 3-D TUCKER by [43]. Furthermore, we develop a matching algorithmic bound under the oracle function model.

5.1 A reduction of DPZP to TUCKER. First, we give the definition of TUCKER.

DEFINITION 5.1 *[54] The input of TUCKER is a pair (G, V_N^d) where V_N^d is a triangulated hypergrid centered at the origin $\mathbf{0}$ and G is a coloring function $g : V_N^d \Rightarrow \{\pm 1, \cdots, \pm d\}$. Let g be antipodal preserving for boundary vertices, that is, $g(-\mathbf{p}) = -g(\mathbf{p})$ for any boundary vertex $\mathbf{p}$. The output of TUCKER is a complementary 1-simplex in the hypergrid, i.e., an edge $(\mathbf{p}, \mathbf{q})$ in the triangulated hypercube so that $g(\mathbf{p}) = -g(\mathbf{q})$.*

The existence of the solution for this problem is guaranteed by a combinatorial topology theorem, Tucker's lemma [54]. The original problem is built on a d dimensional ball, here we cast it into a d-hypercube to simplify our discussion.

Tucker's lemma characterizes the fundamental combinatorial property underlying important mathematical problems such as the Borsuk-Ulam theorem, and Lovász's theorem on Kneser Graph (by Matoušek [36]). It also has application to a type of the fair division problem called consensus-halving problem (Simmons and Su [47]). It has also been applied to the necklace problem and consensus-$\frac{1}{k}$-division problem (Longueville and Živaljević [34]).

It is known that TUCKER is in *PPAD*.

LEMMA 5.1 *[43] d-D TUCKER is in* PPAD.

We prove TUCKER is *PPAD*-hard for any dimension d. The reduction is based on DPZP and Kuhn's triangulation [31].

LEMMA 5.2 *d-D TUCKER is PPAD-hard.*

PROOF. To prove TUCKER is *PPAD*-hard, we reduce DPZP to it. Similar to our previous reduction, for any input of DPZP (F, V_N^d), we first add one more layer to make DPZP (H, V_{N+2}^d) satisfy the antipodal constraint for the function values. The function h is defined slightly different from before as follows: (i) for each inner vertex $\mathbf{p} \in V_N^d$, let $h(\mathbf{p}) = f(\mathbf{p})$. (ii) for a boundary point $\mathbf{p}$, if there is an i such that $p_i = 0$ and $\forall j < i : N + 2 > p_j > 0$, then let $h(\mathbf{p}) = \mathbf{e}_i$. (iii) otherwise, there is an i such that $p_i = N + 2$ and $\forall j < i : 0 < p_j < N + 2$. In that case, $h(\mathbf{p}) = -\mathbf{e}_i$. It is easy to verify that (H, V_{N+2}^d) is a DPZP.

Next, we reduce DPZP (H, V_{N+2}^d) to TUCKER (G, V_{N+2}^d) by defining function g as: $g(\mathbf{p}) = i$ if $h(\mathbf{p}) = \mathbf{e}_i$; $g(\mathbf{p}) = -i$ if $h(\mathbf{p}) = -\mathbf{e}_i$; $g(\mathbf{p}) = -1$ if $h(\mathbf{p}) = \mathbf{0}$. Obviously, the function values of g satisfy TUCKER's antipodal boundary condition.

The last step is to triangulate each small hypercube by Kuhn's triangulation. Since Kuhn's triangulation will not add any extra vertices, the output of Kuhn's triangulation combining with the function g is an instance of TUCKER (G, V_{N+2}^d). By Tucker's lemma, it has a complementary edge. Because of the direction preserving property of the original function h, the complementary edge has to be $(1, -1)$. Again because of the direction preserving property, that node $\mathbf{p}$ with value -1 in the complementary $(1, -1)$ must correspond to $h(\mathbf{p}) = \mathbf{0}$ in the original problem. The result follows. We illustrate the reduction process in **Fig. 10**.

□

Combining Lemma 5.1 and Lemma 5.2, we have

THEOREM 5.1 *d-D TUCKER is* PPAD*-complete.*

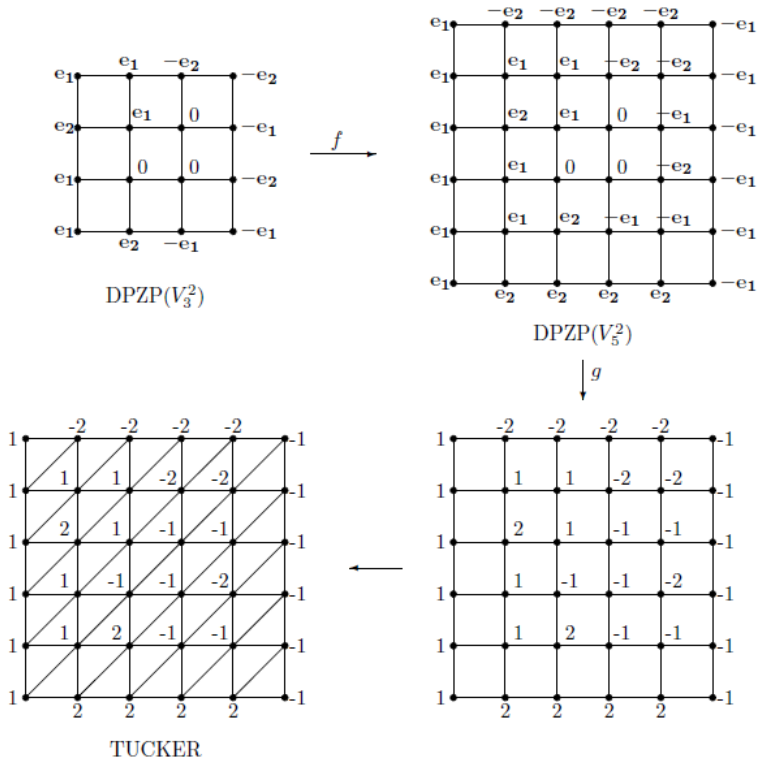

Figure 10: The Proof of Lemma 5.2

5.2 Matching Bound for TUCKER in the Oracle Model. In this section, we derive the oracle matching algorithmic bound for TUCKER.

THEOREM 5.2 *(Lower Bound) For any instance of d-D TUCKER, finding a complementary 1-simplex takes* $\Omega(N^{d-1})$ *time.*

PROOF. Using the same reduction as in *PPAD*-hardness proof of TUCKER, we notice that if there is an algorithm that solves problems of size N^d for TUCKER, it solves problems of size N^d for DPZP. Since there is a lower bound of $\Omega(N^{d-1})$ for DPZP [5], we have the same lower bound for TUCKER. □

The upper bound can be derived in two steps. First, we check if there is a complementary edge on the boundary. This takes $O(N^{d-1})$ steps. If the answer is no, we prove that the number of $(d-1)$-simplices on the boundary which are fully-colored must be odd. Then by using binary search on the parity, one can find a complementary edge in time $O(N^{d-1})$. Hence, the total time complexity is $O(N^{d-1})$.

For $d = 2$, the following boundary lemma is easy to check.

LEMMA 5.3 *For any instance of 2-D TUCKER, if there is no complementary edge on the boundary, then for any pair* (a_1, a_2) *with* $|a_i| = i$, $i = 1, 2$, *the number of* $\{a_1, a_2\}$ *edges on the boundary must be odd.*

PROOF. For simplicity, suppose $a_1 = 1$ and $a_2 = 2$. The other cases follow similarly. We classify the edges to three types:
$A = \{-1,-1\},\{-2,-2\},\{1,1\},\{2,2\}$, $B = \{-1,-2\},\{1,2\}$, and $C = \{-1,2\},\{1,-2\}$.

Choose an arbitrary pair of antipodal vertices; for example, 1 and -1. Let us regard 1 as the source and -1 as the sink of a path. We now consider the configuration of this boundary path.

The source 1 and sink -1 separate the boundary into two parts. Suppose one is denoted as S, the other part is S'. Then the number of $\{-1,-2\}$ edges in S is equal to the number of $\{1,2\}$ edges in S', so the

total number of $\{1,2\}$ edges on the boundary is equal to the number of $\{1,2\}$ edges plus the number of $\{-1,-2\}$ edges in S.

Consider the number of sign changes and absolute value changes on each edge. It is even for types of A and C edges, and odd for type B edges. From 1 to -1, there are an odd number of sign changes, and an even number of absolute value changes (from some $|a_1|=1$ to some $|a_2|=2$ or some $|a_2|=2$ to some $|a_1|=1$). The sum of sign changes and value changes from source to sink along S is odd. Therefore, we must have an odd number of B-type edges.

To prove the number of C-type edges is odd, we note that the total number of $\{-1,2\}$ edges on the boundary is equal to the number of $\{-1,2\}$ edges plus the number of $\{-2,1\}$ edges in S. Consider the number of sign changes on each edge. It is even for types of A and B edges, and odd for type C edges. From 1 to -1, there are an odd number of sign changes. Hence, the number of C-type edges must be odd. □

For $d=3$, we employ the parity argument in Cohen's proof [13] for Tucker's lemma to present the following proof.

DEFINITION 5.2 *A vector $(a_1,a_2,\cdots,a_d)$ is called a pseudo-full tuple if $|a_i|=i$. A pseudo-fully colored $\{a_1,\cdots,a_d\}$-simplex is a $(d-1)$-dimensional simplex with its d-corners colored with a pseudo-full tuple $(a_1,a_2,\cdots,a_d)$.*

LEMMA 5.4 *For any instance of 3-D TUCKER, partitioned by the Kuhn's triangulation, if there is no complementary edge on the boundary, then there exists a pseudo-full tuple (a_1,a_2,a_3) such that the number of $\{a_1,a_2,a_3\}$-simplices on the boundary is odd.*

PROOF.

Suppose not, i.e, assume that for some 3-tuple $\{a_i,a_2,a_2\}$, the number of $\{a_1,a_2,a_3\}$-simplices on the boundary is even. Let A and A' be the center of two opposite faces. Let P be a path on the boundary from A to A', and P' be the antipodal path of P. Then P and P' separate the boundary of the cube into two parts. Let us denote one part as S and the other part as S'.

Let $S(a_1,a_2,a_3)$ be the number of $\{a_1,a_2,a_3\}$-simplices on S, and $S'(a_1,a_2,a_3)$ be the number of $\{a_1,a_2,a_3\}$-simplices on S'. By the assumption, we have

$$S(a_1,a_2,a_3)+S'(a_1,a_2,a_3)\equiv 0 \mod 2.$$

Since the number of $\{a_1,a_2,a_3\}$-simplices on S' is equal to the number of $\{-a_1,-a_2,-a_3\}$-simplices on S, i.e.,

$$S'(a_1,a_2,a_3)=S(-a_1,-a_2,-a_3)$$

we get

$$S(a_1,a_2,a_3)+S(-a_1,-a_2,-a_3)\equiv 0 \mod 2.$$

On the other hand, let $P(a_1,a_2)$ denote the number of $\{a_1,a_2\}$ edges on P. As A and A' are of different signs, P must have an odd number of $\{a_1,a_2\}$ edges where a_1 and a_2 have different signs. Since there is no complementary edge on the boundary, there are six possible edges of such types, i.e., $\{a_1,-a_2\}$,$\{a_1,-a_3\}$,$\{a_2,-a_1\}$,$\{a_2,-a_3\}$,$\{a_3,-a_1\}$ and $\{a_3,-a_2\}$. So, we have

$$P(a_1,-a_2)+P(a_1,-a_3)+P(a_2,-a_1)+P(a_2,-a_3)+P(a_3,-a_1)+P(a_3,-a_2)\equiv 1 \mod 2$$

Now, we are prepared to derive a contradiction (**Fig. 11**).

Take S into consideration, and consider the 2-simplices of type $\{a_1,-a_2\}$ as doors and the 2-simplices of all other types as walls on the boundary. Starting from a $\{a_1,-a_2\}$ edge lying on $P\bigcup P'$, since there is no complementary edge on S, it will either reach another $\{a_1,-a_2\}$ edge on $P\bigcup P'$ (**Fig. 11(a)**), or will terminate in a $\{a_1,-a_2,\pm a_3\}$-simplex on S (**Fig. 11(b)**). Starting from $\{a_1,-a_2\}$ edges lying on $S\setminus(P\bigcup P')$, they will be paired in two $\{a_1,-a_2,\pm a_3\}$ simplices (**Fig. 11(c)**), or two $\{a_1,-a_2\}$ edges on $P\bigcup P'$ (**Fig. 11(d)**), or one in a $\{a_1,-a_2,\pm a_3\}$ simplex while another on $P\bigcup P'$ (**Fig. 11(e)**). This derives a parity equation that

$$S(a_1,-a_2,a_3)+S(a_1,-a_2,-a_3)\equiv P(a_1,-a_2)+P'(a_1,-a_2) \mod 2$$

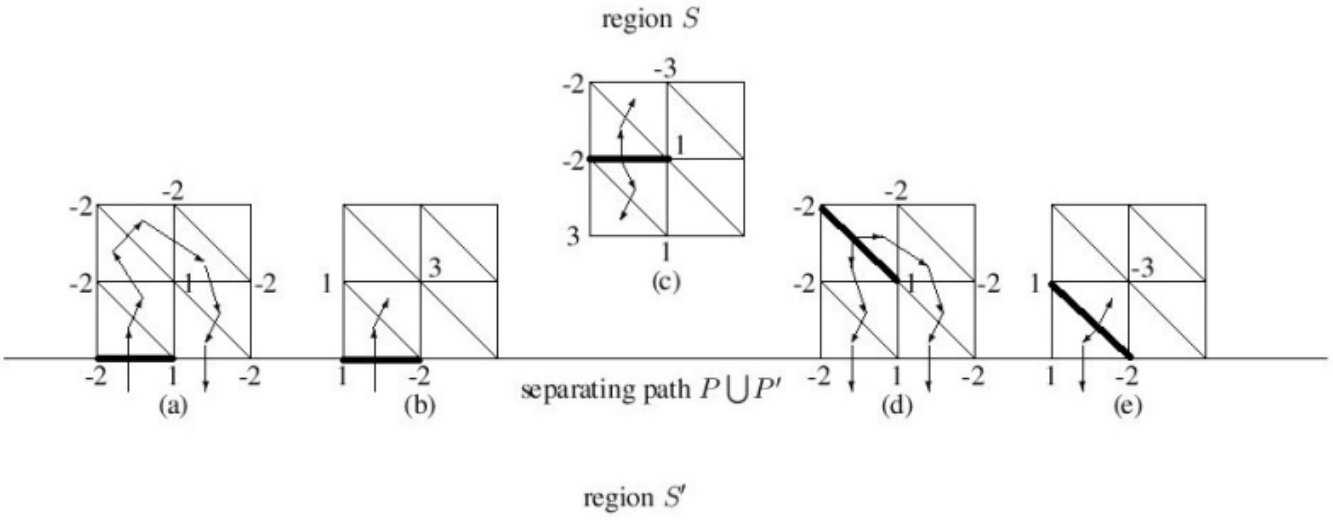

Figure 11: The proof of Lemma 5.4

which implies

$$S(a_1, -a_2, a_3) + S(a_1, -a_2, -a_3) \equiv P(a_1, -a_2) + P(-a_1, a_2) \mod 2$$

Similarly, we have

$$S(a_1, a_2, -a_3) + S(-a_1, a_2, -a_3) \equiv P(a_2, -a_3) + P(-a_2, a_3) \mod 2$$

and

$$S(-a_1, a_2, a_3) + S(-a_1, -a_2, a_3) \equiv P(-a_1, a_3) + P(a_1, -a_3) \mod 2$$

Summing up these three equations we reach a contradiction, which proves the lemma.

□

This lemma can be generalized to any constant dimension.

LEMMA 5.5 *For any instance of d-D TUCKER, partitioned by the Kuhn's triangulation, if there is no complementary edge on the boundary, there must exist a pseudo-full tuple* $(a_1, a_2, \cdots, a_d)$ *such that the number of* $\{a_1, a_2, \cdots, a_d\}$*-simplices on the boundary is odd.*

The result follows from Fan's work [19] on the generalized Tucker's lemma applied to Kuhn's triangulation.

We are now ready to state the proposition that allows us to take the divide-&-conquer approach for a matching bound.

PROPOSITION 5.1 *Given a Kuhn's triangulated hypergrid* H*, for any pseudo-full tuple* $(a_1, a_2, \cdots, a_d)$*, the number of* $\{a_1, a_2, \cdots, a_d\}$*-simplices on the boundary* ∂H *has the same parity as the total number of* $\{-a_i, a_1, a_2, \cdots, a_d\}$*-simplices in* H*, summing over* $i = 1, 2, \cdots, d$*. I.e.,*

$$\textit{no. of } (\{a_1, a_2, \cdots, a_d\}\textit{-simplices on } \partial H) \equiv \sum_{i=1}^{d} \textit{ no. of } (\{-a_i, a_1, a_2, \cdots, a_d\}\textit{-simplices on } H) \mod 2.$$

PROOF. We change the labels of all nodes with label $-a_i$ to a_0. Then consider the subscripts of $a_0, a_1, \cdots, a_d$. from standard index theorems (see, e.g., [5]), The parity of $(a_1, a_2, \cdots, a_d)$ on the boundary ∂H is the same as the parity of $(a_0, a_1, \cdots, a_d)$ on ∂H. The claim follows when we replace a_0 back to their original label $-a_i$, $i = 1, 2, \cdots, d$. □

THEOREM 5.3 *(Upper Bound) There exists an* $O(N^{d-1})$ *algorithm for d-D TUCKER under Kuhn's triangulation, when the function g is given by an oracle.*

PROOF.

First, by checking the boundary, we can find out whether there are complementary edges on the boundary. If there are any, finding one of them can be completed in time $O(N^{d-1})$. If not, by Lemma 5.5, there must be an $\{a_1, a_2, \cdots, a_d\}$-simplex on the boundary that has appeared an odd number of times. Within the same $O(N^{d-1})$ time, we find this pseudo-full tuple $(a_1, a_2, \cdots, a_d)$. Suppose we replace the label $-a_i$,$\forall i = 1, 2, \cdots, d$ with label a_0. For the purpose of this algorithm, we need to replace the label only for nodes queried about their colors. By Proposition 5.1, the number of $\{a_0, a_1, \cdots, a_d\}$-simplices is odd. Use a divide-&-conquer approach applies to the subscripts of $(a_0, a_1, \cdots, a_d)$ to find a simplex of labels $\{a_0, a_1, \cdots, a_d\}$. a_0 is representing some negative labeled node $-a_i$, for some i. Thus, the $(a_i, -a_i)$ edge in this simplex is the complementary edge we wanted to find.

Since the size of the hypercube decreases geometrically, the complexity of checking which case occurred will not exceed $O(N^{d-1})$.

□

THEOREM 5.4 *For any d-hypercube which is an instance of TUCKER and it is triangulated by Kuhn's triangulation, the computational time for finding a complementary 1-simplex is* $\theta(N^{d-1})$ *under the oracle model.*

6. Conclusion. Our study builds up a computational complexity connection among three discrete versions of the fixed point problem. Three new results characterizing computational complexity of those discrete fixed points are derived: a *PPAD*-completeness proof for computing a direction preserving zero point, a matching oracle complexity bound for BROUWER, as well as a matching oracle complexity bound for SPERNER.

We argued that these problems can be used for different applications. For example, utilizing SPERNER, Deng et al. [17], derive the computational complexity of the envy-free cake cutting problem, both in the polynomial function model and in the oracle function model.

In another direction, in this paper, we build up a connection between DPZP and TUCKER which allows for a simple proof that TUCKER is *PPAD*-complete for all constant dimensions. In comparison, both the results of Pálvölgyi for the 2D case [42] and Papadimitriou for 3D case [43] have been based on the first principles. It follows that both the deterministic lower bound [5] and the randomized lower bound [12] for DPZP immediately applies to TUCKER because our reduction only expands DPZP by a constant factor in size. Furthermore, our construction is suitable for discussion of randomized oracle complexity and quantum oracle complexity.

Acknowledgments. The authors wish to express their thanks to the referees, Associate Editor and Area Editor for their suggestions and comments.

References

[1] K. Arrow and G. Debreu. *Existence of an equilibrium for a competetive economy*, Econometrica **22** (1954), 265–290.

[2] R.B. Bapat, *A constructive proof of a permutation-based generalization of Sperner's lemma*, Mathematical Programming **44** (1989),1–3.

[3] S. Brin and L. Page, (1998). *The anatomy of a large-scale hypertextual Web search engine*, Proceedings of the 7th World Wide Web Conference, Brisbane, Australia (1998), 107–117.

[4] L.E.J. Brouwer, *Ueber eineindeutige, stetige Transformationen von Flächen in sich*, Mathematische Annalen **69** (1910), 176–180.

[5] X. Chen and X. Deng, *Matching algorithmic bounds for finding a Brouwer fixed point*, Journal of the ACM **55**(3) (2008), 13:1–13:26.

[6] X. Chen and X. Deng, *On the Complexity of 2D Discrete Fixed Point Problem*, Theoretical Computer Science **410**(44) (2009), 4448–4456.

[7] X. Chen and X. Deng, *A Simplicial Approach for Discrete Fixed Point Theorems* Algorithmica **53**(2) (2009), 3–12.

[8] X. Chen and X. Deng, *Settling the Complexity of Two-Player Nash Equilibrium*, Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science, Berkeley, CA, USA (2006), 261–272.

[9] X. Chen, X. Deng and S.-H. Teng, *Computing nash equilibria: approximation and smoothed complexity*, the 47th annual IEEE Symposium on Foundations of Computer Science, Berkeley, CA, USA (2006), 603–612.

[10] X. Chen, X. Deng, S.-H. Teng, *Settling the complexity of computing two-player Nash equilibria*, Journal of the ACM **56**(3) (2009), 14:1–14:57.

[11] X. Chen, X. Sun and S.-H. Teng, *Quantum separation of local search and fixed point Computation*, Proceedings of the 14th Annual International Computing and Combinatorics Conference, Dalian, China (2008), 170–179.

[12] X. Chen and S.-H. Teng, *Paths beyond local search: a tight bound for randomized fixed-point computation*, 48th annual IEEE Symposium on Foundations of Computer Science. Providence, RI, USA (2007) 124–134.

[13] D.I.A. Cohen, *On the combinatorial antipodal-point lemmas*, Journal of Combinatorial Theory (Series B) **27** (1979) 87–91.

[14] R. Cole, Y. Dodis, and T. Roughgarden, *Pricing network edges for heterogeneous selfish users*, Proceedings of the Thirty-Fifth Annual ACM Symposium on Theory of Computing, San Diego, CA, USA (2003), 521–530.

[15] P. Crescenzi and R. Silvestri, *Sperner's Lemma and Robust Machines*, Computational Complexity **7(2)** (1998), 163–173.

[16] C. Daskalakis, P. W. Goldberg and C. H. Papadimitriou, *The complexity of computing a Nash equilibrium*, the 38th ACM Symposium on Theory of Computing, Seattle, WA, USA (2006), 71–78.

[17] X. Deng, Q. Qi and A. Saberi, *On the complexity of envy-free cake cutting*, (2009), http://arxiv.org/abs/0907.1334.

[18] B.C. Eaves, *On the basic theorem of complementarity*, Mathematical Programming, **1**(1) (1971), 68–75.

[19] K. Fan, *A generalization of Tucker's combinatorial lemma with topological applications*, The Annals of Mathematics (Second Series) **56** (3) (1952), 431–437.

[20] K. Friedl, G. Ivanyos, M. Santha and F. Verhoeven. *On the Black-box Complexity of Sperner's Lemma*, Proceedings of the 15th International Symposium on Fundamentals of Computation Theory, Lübeck, Germany (2005), 245–257.

[21] R. M. Freund, *Variable dimension complexes Part I: Basic Theory*, Mathematics of Operations Research **9**(4) (1984), 479–497.

[22] R.M. Freund, *Variable dimension complexes Part II: A unified approach to some combinatorial lemmas in topology*, Mathematics of Operations Research, **9**(4) (1984), 498–509.

[23] T. Fujimoto, *An extension of Tarski's fixed point theorem and its application to isotone complementarity problems*, Mathematical Programming **28** (1) (1984), 116–118.

[24] J.J. Herings, D. Talman and Z. Yang. The Computation of a Continuum of Constrained Equilibria, *Mathematics of Operations Research*, Volume 21, Issue 3, 1996.

[25] M.D. Hirsch, C.H. Papadimitriou and S. Vavasis. Exponential Lower Bounds for Finding Brouwer Fixed Points, Journal of Complexity, Vol. 5 (1989), pp.379–416.

[26] A. Hyvärinen and E. Oja, *Fast and robust fixed-point algorithms for independent component analysis*, Neural Computation **9** (1997), 1483–1492.

[27] T. Iimura, *A discrete fixed point theorem and its applications*, Journal of Mathematical Economics **39** (2003), 725–742.

[28] T. Iimura, K. Murota, and A. Tamura, *Discrete fixed point theorem reconsidered*, Journal of Mathematical Economics **41** (2005), 1030-1036.

[29] J. Kleinberg, *Authoritative sources in a hyperlinked environment*, Journal of the ACM **46**(5) (1998), 604–632.

[30] S. Kintali, L.J. Poplawski, R. Rajaraman, R. Sundaram and S.-H. Teng, *Reducibility Among Fractional Stability Problems*, Proceedings, of the 50th Annual Symposium on Foundations of Computer Science, Atlanta, GA, USA (2009), 283–292.

[31] K.W. Kuhn, *Some combinatorial lemmas in topology*, IBM Journal of Research and Development **4**(5) (1960), 518–524.

[32] G.V.D. Laan, A.J.J. Talman and Z. Yang, *Solving discrete zero point problems*, Mathematical Programming (2006), 108(1), 127–134.

[33] C. E. Lemke and J. T. Howson, *Equilibrium Points of Bimatrix Games*, Journal of the Society for Industrial and Applied Mathematics **12** (1964), 413–423.

[34] M. Longueville and R.T. Živaljević, *The Borsuk-Ulam-property, Tucker-property and constructive proofs in combinatorics*, Journal of Combinatorial Theory (Series A) **113** (2006), 839–850.

[35] S. Low, *A duality model of tcp and queue management algorithms*, IEEE/ACM Transactions on Networking **11**(4) (2003), 525–536.

[36] J. Matoušek, *A combinatorial proof of Kneser's conjecture*, Combinatorica **24** (2004): 163–170.

[37] A. Mehta, S. Shenker, and V. Vazirani, *Profit-maximizing multicast pricing by approximating fixed points*, Proceedings of ACM Conference on Electronic Commerce, San Diego, CA, USA (2003), 218–219.

[38] N. Megiddo and C.H. Papadimitriou, *A Note on total functions, existence theorems and computational complexity*, Theoretical Computer Science **81** (1991), 317–324.

[39] G. Meinardus, *Invarianz bei linearen approximationen*, Archive for Rational Mechanics and Analysis **14** (1963), 301–303.

[40] F. Meunier, *Discrete splittings of the necklace*, Mathematics of Operations Research **33**(3) (2008), 678–688.

[41] J. Nash, *Equilibrium points in n-person games*, Proceedings of the National Academy of the USA, **36**(1) (1950),48–49.

[42] D. Pálvölgyi, *2D-TUCKER is PPAD-complete*, Proceedings of the 5th International Workshop on Internet and Network Economics Rome, Italy (2009), 569–574.

[43] C.H. Papadimitriou, *On the complexity of the parity argument and other inefficient proofs of existence*, Journal of Computer and System Sciences **48** (1994), 498–532.

[44] C. Robinson, *Dynamical systems, stability, symbolic dynamics, and chaos*, CRC Press, 1999.

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

[46] H.E. Scarf, *The approximation of fixed points of a continuous mapping*, SIAM Journal on Applied Mathematics **15** (1967), 997–1007.

[47] F.W. Simmons and F.E. Su, *Consensus-halving via theorems of Borsuk-Ulam and Tucker*, Mathematical Social Science **45** (2003), 15–25.

[48] E. Sperner, *Neuer Beweis für die Invarianz der Dimensionszahl und des Gebietes*, Abhandlungen aus dem Mathematischen Seminar der Universitt Hamburg **6** (1928), 265–272.

[49] D. Spielman and S.-H. Teng, *Spectral partitioning works: Planar graphs and finite element meshes*, Proceedings of the 37th Annual Symposium on Foundations of Computer Science (1996), Burlington, VT, USA, 96–105.

[50] F.E. Su, *Rental harmony: Sperner's lemma in fair division*, The American Mathematical Monthly **106** (1999), 930–942.

[51] A.J.J. Talman and Z. Yang, *A constructive proof of Ky Fan's coincidence theorem*, Mathematical Programming (Ser. A) **118** (2009), 317–325.

[52] M.J. Todd, *The computation of fixed points and applications*, Lecture Notes in Economics and Mathematical Systems, Springer-Verlag, New York, 1976.

[53] M.J. Todd, *A quadratically-convergent fixed-point algorithm for economic equilibria and linearly constrained optimization*, Mathematical Programming **18**(1) (1980), 111-126.

[54] A.W. Tucker, *Some topological properties of disk and sphere*, Proceedings of the First Canadian Mathematical Congress, Montréal, Canada(1945), 285–309.

[55] Y. Yamamoto, *A new variable dimension algorithm for the fixed point problem*, Mathematical Programming **25**(3) (1983), 329–342.

[56] Y. Ye. *A path to the Arrow-Debreu competitive market equilibrium*, Mathematical Programming **111**(1-2) (2007), 315–348 .