On Algorithms for Discrete and Approximate Brouwer Fixed Points

[Extended Abstract]

Xi Chen [*]
Department of Computer Science
Tsinghua University
Beijing, P.R.China
xichen00@mails.tsinghua.edu.cn

Xiaotie Deng [†]
Department of Computer Science
City University of Hong Kong
Hong Kong SAR, P.R.China
deng@cs.cityu.edu.hk

ABSTRACT

We study the algorithmic complexity of the discrete fixed point problem and develop an asymptotic matching bound for a cube in any constantly bounded finite dimension. To obtain our upper bound, we derive a new fixed point theorem, based on a novel characterization of boundary conditions for the existence of fixed points.

In addition, exploring a linkage with the approximation problem of the continuous fixed point problem, we obtain asymptotic matching bounds for complexity of the approximate Brouwer fixed point problem in the continuous case for Lipschitz functions that close a previous exponential gap. It settles a fifteen years old open problem of Hirsch, Papadimitriou and Vavasis by improving both the upper and lower bounds.

Our new characterization for existence of a fixed point is also applicable to functions defined on non-convex domain and makes it a potentially useful tool for design and analysis of algorithms for fixed points in general domain.

Categories and Subject Descriptors

F.2.2 [**Analysis of Algorithms and Problem Complexity**]: Nonnumerical Algorithms and Problems—*Computations on discrete structures, Geometrical problems and computations, Sorting and searching*

[*]Work supported by National Natural Science Foundation of China (60135010, 60321002) and Chinese National Key Foundation Research & Development Plan (2004CB318108).

[†]The work described in this paper was supported by a grant from the Research Grants Council of Hong Kong Special Administrative Region, China (Project No. CityU 1156/04E).

STOC'05, May 22-24, 2005, Baltimore, Maryland, USA.

General Terms

Algorithms, Economics, Theory

Keywords

Fixed point theorem, Lipschitz function, Approximate fixed point, Sperner's lemma

1. INTRODUCTION

The Brouwer fixed point theorem and its variations have had a profound influence in mathematical sciences and applications, including approximation theory [26], dynamical systems [31], game theory [28], and the most popularly known of all, the theory of general equilibrium in Economics [2]. While fixed point theorems are applied to establish fundamental theories, fixed point algorithms are used to solve important application problems, especially in many recent works for network communication: TCP network calculus [1], by Altman, Avrachenkov, and Barakat; network edge pricing [7], by Cole, Dodis, and Roughgarden; multicast pricing [25], by Mehta, Shenker, and Vazirani; and TCP queue management [24], by Low. Fixed point theorems also have other important applications in computer science, e.g., the spectral analysis for numerical computation [41], by Spielman and Teng.

The Brouwer fixed point theorem [4] can be stated succinctly as follows: any continuous function $\mathcal{F}$ mapping $D = [0,1]^d$ to itself has a fixed point: $\mathcal{F}(x) = x$ $(x \in D)$. In various levels of generalities, it can be extended to different relaxed requirements on the function $\mathcal{F}$ and the domain D. A mathematical structural characterization of the Brouwer's theorem is the Sperner's lemma. It ensures that a certain labelling rule on vertices of a simplicial partitioning of a simplex S^n guarantees the existence of a sub-simplex with all vertices differently labelled. Naturally, it has been influential on design of combinatorial algorithms for the fixed point problem, started in the 60's with Scarf's seminal work [32], which finds an approximate fixed point by finding a completely labelled primitive set based on a structure lemma similar to, but not the same as, the Sperner's Lemma. Kuhn replaced the primitive sets by simplices and simplicial partition [23]. The simplicial approach has since been adopted by most general purpose fixed point algorithms developed later, such as the restart algorithm of Merrill [27] and the homotopy algorithm of Eaves [12].

Many combinatorial algorithms based on simplicial partitioning are in the worst case exponential in computation time. It was conjectured the performance of some such algorithms might be better. Hirsch, Papadimitriou and Vavasis played down such a hope by proving a general exponential lower bound [13]. For contractive Lipschitz functions, that is, $\mathcal{F}$ such that $|\mathcal{F}(x) - \mathcal{F}(y)| \leq c * |x - y|$ with $c < 1$, the fixed point problem can be solved much faster, for example, by the fixed point iteration algorithm of Banach [3], the Newton method (see the works of Ortega and Rheinbolt [30], Kellogg, Li, and Yorke [20], and Smale [39]), the interior ellipsoid algorithm of Huang, Khachiyan and Sikorski [14].

We study the fixed point algorithm for general Lipschitz functions. Therefore, the iteration algorithm approach for contractive functions does not apply here. Our study is motivated by a particularly interesting recent discrete version of the fixed point problem, introduced by Iimura [15, 16], stating that any direction-preserving function $\mathcal{F}$ (a discrete analogue to the continuous function) that maps $D = N^d$ ($N = \{0, 1, 2 ... n-1\}$) to itself, has a fixed point [15, 16].

Iimura's proof is non-constructive by extending the discrete direction-preserving function to a continuous function such that the latter has a fixed point if and only if the former has one. It is, therefore, not suitable to develop algorithms for finding a solution. The algorithmic results for approximate fixed point by Hirsch, Papadimitriou, and Vavasis, on the other hand, have a natural extension for the discrete version, leading to a lower bound of $\Omega(n^{d-2})$ and an upper bound of $O(n^d)$ for the discrete fixed point problem on the grid N^d. Noticeably, for $d = 2$, Hirsch, Papadimitriou and Vavasis have a matching bound of $\Theta(n)$ for a grid of N^2. Closer examination of the upper bound of Hirsch, Papadimitriou and Vavasis would reveal a boundary condition for a fixed point to exist: The winding number of the boundary is non-zero. Our upper bound relies on the establishment of such a boundary condition for higher dimensions. We exploit the grid structure and the direction-preserving condition for the discrete fixed point problem to develop a succinct combinatorial structure that leads to the design of our algorithm.

In addition, the combinatorial lemma derives an independent and a constructive proof for Iimura's fixed point theorem. In fact, our result derives a general characterization for a pair of function and domain $(\mathcal{F}, D)$ to have a fixed point which is also applicable to non-convex domains and thus improves the results of Iimura [15], Iimura, Murota, and Tamura [16].

The construction is based on a characterization, via a parity argument, of $\mathcal{F}(x) - x$ on a unit cube, and then builds on a collection of unit cubes to establish a global boundary condition, if no fixed point exists. It is as elementary as Sperner's Lemma [40]. In fact, the Brouwer fixed point theorem for the continuous case can also be derived from our characterization lemma.

The tight lower bound proof for the two dimension case by Hirsch, Papadimitriou and Vavasis is also very elegantly done. However, some weakness makes it not suitable to be extended to higher dimensions directly. Our lower bound proof fully utilizes the simplification brought in by the discrete version and introduces a game on a lattice graph to focus on the essence of the problem. Then we extend the result derived from the game played on the lattice graph to the matching lower bound for the discrete fixed point problem. Even though the final proof is deep and rather complicated, the approach is quite clear and accessible.

Finally, it is not hard to establish a linkage between direction-preserving functions for the discrete fixed point problem and Lipschitz functions for the approximate fixed point problem. That linkage makes our results for the discrete version extendable to the approximate fixed point problem for Lipschitz functions. In particular, our algorithmic results solved an open problem proposed by Hirsch, Papadimitriou and Vavasis [13].

For succinctness of the presentation here, we consider the function $f(x) = \mathcal{F}(x) - x$. The problem of finding a fixed point for $\mathcal{F}$ is equivalent to the problem of finding a root for f: $\mathcal{F}(x) = x$ if and only if $f(x) = 0$. We call such a point a zero point (or root) of f. As pointed out by Hirsch, Papadimitriou and Vavasis [13], the general zero point problem is algorithmically harder than the fixed point problem. However, our discussion only considers a special class of zero point problem which is equivalent to the discrete fixed point problem. We should in Section 2 present the necessary definitions, together with a proof that the above two problems are equivalent in computational complexity. The crucial combinatorial lemma is discussed in Section 3, followed by the algorithm and the upper bound proof. The lower bound construction and proof will be outlined in Section 4. In Section 5, we will discuss applications to the continuous case. We conclude in Section 6 with discussions and remarks on our approach and other related results as well as potential research directions.

2. DEFINITIONS

For any $x \neq 0 \in \mathbb{R}$, we define $\text{sgn}(x) = +1$ if $x > 0$ and $\text{sgn}(x) = -1$ if $x < 0$. For any $1 \leq k \leq d$, we use e^k to denote the kth unit vector of $\mathbb{Z}^d$. Here $e^k_k = 1$ and for any $1 \leq i \neq k \leq d$, $e^k_i = 0$. For any vector $v \in \mathbb{Z}^d$, $1 \leq k \leq d$ and $l \in \mathbb{Z}$, we define vector $v[k \leftarrow l] = v + (l - v_k)\, e^k$. For simplicity, we use v^- to denote vector $v[d \leftarrow (v_d - 1)]$ and v^+ to denote vector $v[d \leftarrow (v_d + 1)]$.

Definition 1. *For any $p < q \in \mathbb{Z}^d$, we define a rectangular set $A_{p,q} = \{ r \in \mathbb{Z}^d \mid p \leq r \leq q \} \subset \mathbb{Z}^d$. Its boundary is defined as $B_{p,q} = \{ r \in A_{p,q} \mid \exists\ 1 \leq i \leq d$ such that $r_i = p_i$ or $q_i \} \subset A_{p,q}$.*

Definition 2. *Map $\mathcal{F} : A_{p,q} \rightarrow \mathbb{R}^d$ is said to be direction-preserving if $\forall\ r^1, r^2 \in A_{p,q}$ such that $|r^1 - r^2|_\infty \leq 1$, we have $(\mathcal{F}_i(r^1) - r^1_i)(\mathcal{F}_i(r^2) - r^2_i) \geq 0$, for any $1 \leq i \leq d$.*

Definition 3. *Function $f : S \rightarrow \{ 0, \pm e^1, \pm e^2 ... \pm e^d \}$, where $S \subset \mathbb{Z}^d$, is said to be direction-preserving if $\forall\ r^1, r^2 \in S$ such that $|r^1 - r^2|_\infty \leq 1$, we have $|f(r^1) - f(r^2)|_\infty \leq 1$. We use $F[S]$ to denote all such functions on S. Function $f : A_{p,q} \rightarrow \{ 0, \pm e^1, ... \pm e^d \}$ is said to be bounded if $\mathcal{F}(r) = f(r) + r$ is a map from $A_{p,q}$ to itself.*

Using the discrete fixed point theorem in [15][16], we get the following simplified version. Actually, we will derive an independent constructive proof later.

Theorem 1. *For any direction-preserving map $\mathcal{F}$ from $A_{p,q}$ to itself, there exists $r^* \in A_{p,q}$ such that $\mathcal{F}(r^*) = r^*$. Such point r^* is called a fixed point of map $\mathcal{F}$.*

We are interested in the following discrete fixed point problem $\mathbf{DFP}^d$: given a direction-preserving map $\mathcal{F}$ from $A_{p,q}$ to itself, where $A_{p,q} \subset \mathbb{Z}^d$, compute a fixed point r^* of $\mathcal{F}$. Algorithms discussed in this paper will be restricted to those that are based on map evaluations. That is, map $\mathcal{F}$ looks like a black box to algorithm designers. It can only be accessed by giving a point $r \in A_{p,q}$ and evaluate $\mathcal{F}(r)$. We use $n = |p-q|_\infty + 1$ as the measure of input size, then two kinds of complexities will be considered, query complexity $Q(n,d)$ and time complexity $T(n,d)$.

Actually, we only need to focus on the following discrete zero point problem $\mathbf{DZP}^d$: given a bounded function $f \in F[A_{p,q}]$ where $A_{p,q} \subset \mathbb{Z}^d$, compute a zero point r^* such that $f(r^*) = 0$. The existence of such r^* is guaranteed by Theorem 1, as $(\mathbf{A})$ $\mathcal{F}(r) = f(r) + r$ is a direction-preserving map from $A_{p,q}$ to itself. Similarly, we use $Q'(n,d)$ and $T'(n,d)$ here to denote the query complexity and time complexity of $\mathbf{DZP}^d$ respectively. Our main results in section 3 and 4 are

THEOREM 2. *For any $d \geq 2$ and $n > 48d$, we have*

$$0.5\,(\,\lfloor (n-1)/2^{10} \rfloor\,)^{d-1} \;\leq\; Q'(n,d) \;\leq\; 7n^{d-1}$$
$$Q'(n,d) \leq T'(n,d) \;\leq\; O(d^2(2n)^{d-1})$$

There is a strong relationship between these two problems. $(\mathbf{A})$ shows that any algorithm for $\mathbf{DFP}^d$ can be used to solve $\mathbf{DZP}^d$. We simply call it on $\mathcal{F}$ to get a fixed point r^* of $\mathcal{F}$ and it must also be a zero point of f. If at some time, the algorithm wants to evaluate $\mathcal{F}(r)$, then we just query $f(r)$ and compute $\mathcal{F}(r) = f(r) + r$ using $O(d)$ steps. This shows $Q'(n,d) \leq Q(n,d)$ and $T'(n,d) \leq O(d)Q(n,d) + T(n,d)$.

On the other hand, for any direction-preserving map $\mathcal{F}$ from $A_{p,q}$ to itself, a bounded function $f \in F[A_{p,q}]$ can be constructed. For any $r \in A_{p,q}$, if $\mathcal{F}(r) = r$, then $f(r) = 0$. Otherwise, let $1 \leq i \leq d$ be the largest integer such that $\mathcal{F}_i(r) \neq r_i$, then $f(r) = \mathrm{sgn}(\mathcal{F}_i(r) - r_i)\, e^i$. Similarly we have $Q(n,d) \leq Q'(n,d)$ and $T(n,d) \leq O(d)Q'(n,d) + T'(n,d)$.

In conclusion, these two problems are equivalent in computational complexity and Theorem 2 is also true for both $Q(n,d)$ and $T(n,d)$. For any specific $d \geq 2$, it gives us matching bounds $\Theta(n^{d-1})$ for these two complexities.

3. AN ALGORITHM FOR DZP

In this section, we'll first prove a discrete zero point theorem. For any function $f \in F[A_{p,q}]$, it gives us a condition on $f(B_{p,q})$ which guarantees the existence of zero point in $A_{p,q}$. Then a recursive algorithm will be presented, which can be used to solve problem $\mathbf{DZP}^d$. Its main idea is really similar to binary search. In each turn, we divide the function domain into two parts and the zero point theorem is employed to decide which side to follow.

3.1 The Discrete Zero Point Theorem

First we define subsets of $\mathbb{Z}^d$ called t-cubes where $0 \leq t \leq d$. Lemma 1 and 2 about t-cubes are both easy to prove.

DEFINITION 4. *For any $r \in \mathbb{Z}^d$ and $S \subset \{1, 2 \ldots d\}$ with $|S| = d - t$, the t-cube $C^t \subset \mathbb{Z}^d$ which is centered at r and perpendicular to S is defined as $C^t = \{\, p \in \mathbb{Z}^d \mid \forall\, 1 \leq i \leq d$ if $i \in S$, then $p_i = r_i$. Otherwise, $p_i = r_i$ or $r_i + 1 \,\}$.*

DEFINITION 5. *For any rectangular set $A_{p,q} \subset \mathbb{Z}^d$, we define $V[p,q] = \{\, (d-1)$-cube $C \subset A_{p,q} \mid C$ is perpendicular to $S = \{k\}$ and centered at r such that $r_k = p_k$ or $q_k \,\}$.*

LEMMA 1. *Let $C^t \subset \mathbb{Z}^d$ be a t-cube where $t \geq 2$, then for any $(t-2)$-cube $C^{t-2} \subset C^t$, there are exactly two $(t-1)$-cubes in C^t that contain C^{t-2}.*

LEMMA 2. *Let $C^{d-1} \subset A_{p,q} \subset \mathbb{Z}^d$ be some $(d-1)$-cube. If $C^{d-1} \in V[p,q]$, then there is exactly one d-cube $\subset A_{p,q}$ that contains it. Otherwise, there are exactly two such d-cubes.*

Our zero point theorem is closely related to the number of bad $(d-1)$-cubes in set $V[p,q]$. Inductively, bad t-cubes in $\mathbb{Z}^d$ where $0 \leq t \leq d-1$ with respect to certain function f are defined as follows.

DEFINITION 6. *A 0-cube $C^0 \subset \mathbb{Z}^d$ is bad relative to f iff $f(C^0) = \{\, e^1 \,\}$. For $1 \leq t \leq d-1$, a t-cube $C^t \subset \mathbb{Z}^d$ is bad relative to f if $(\mathbf{B}_1)$ $f(C^t) = \{\, e^1, e^2 \ldots e^{t+1} \,\}$ and $(\mathbf{B}_2)$ the number of bad $(t-1)$-cubes in C^t is odd. For any function $f \in F[A_{p,q}]$, set $V_f[p,q] = \{$ bad $(d-1)$-cubes in $V[p,q] \,\}$.*

LEMMA 3. *For any d-cube $C^d \subset \mathbb{Z}^d$ and $f \in F[C^d]$ such that f has no zero point in C^d, the number of bad $(d-1)$-cubes in C^d must be even.*

Before presenting the proof of Lemma 3 (which is simply a corollary of Lemma 4), we note Lemma 2 and 3 together have already proved a zero point theorem for us.

THEOREM 3. *Any function $f \in F[A_{p,q}]$ satisfies $|V_f[p,q]|$ is odd must have a zero point in $A_{p,q}$.*

LEMMA 4. *For any t-cube $C^t \subset \mathbb{Z}^d$ where $1 \leq t \leq d$ and any $f \in F[C^t]$ such that $f(C^t) \subset \{\pm e^1, \pm e^2, \ldots \pm e^t\}$, the number of bad $(t-1)$-cubes in C^t must be even.*

PROOF. The base case for $t = 1$ is trivial. For $t \geq 2$, we assume that the claim is true for $t-1$. If there is no bad $(t-1)$-cube $\subset C^t$, then we are done. Otherwise, there exists at least one bad $(t-1)$-cube $C^{t-1} \subset C^t$. $(\mathbf{B}_1)$ shows that $f(C^{t-1}) = \{\, e^1, e^2 \ldots e^t \,\}$ and the direction-preserving property of f indicates that $f(C^t) = \{\, e^1, e^2 \ldots e^t \,\}$.

For any $(t-1)$-cube $C^{t-1} \subset C^t$, we'll prove that if it satisfies $(\mathbf{B}_2)$, then it must also satisfy $(\mathbf{B}_1)$. This shows that $C^{t-1} \subset C^t$ is bad iff the number of bad $(t-2)$-cubes in C^{t-1} is odd. As a consequence, the parity of bad $(t-1)$-cubes in C^t must be the same as the following summation $\sum_{C^{t-1} \subset C^t} |\,\{$ bad $(t-2)$-cubes in $C^{t-1} \,\}\,|$. But Lemma 1 asserts that it is even and the lemma will follow.

Therefore, we only need to prove that $(\mathbf{B}_2)$ implies $(\mathbf{B}_1)$ for any $C^{t-1} \subset C^t$. As there is at least one bad $(t-2)$-cube in C^{t-1} to ensure $(\mathbf{B}_2)$, we have $\{\, e^1, e^2 \ldots e^{t-1} \,\} \subset f(C^{t-1})$. If $f(C^{t-1}) = \{\, e^1, e^2 \ldots e^{t-1} \,\}$, then induction assumption for the claim of this lemma requires the number of bad $(t-2)$-cubes in C^{t-1} to be even, and hence can't satisfy $(\mathbf{B}_2)$. Therefore, $f(C^{t-1}) \subset f(C^t) = \{\, e^1, e^2 \ldots e^t \,\}$ must equal to $\{\, e^1, e^2 \ldots e^t \,\}$ and $(\mathbf{B}_1)$ is satisfied. □

3.2 The Recursive Algorithm

We're now ready to present our recursive algorithm called $\mathbf{FindZero}^d(g, V_g[p,q])$. Its input satisfies that $g \in F[A_{p,q}]$ where $A_{p,q} \subset \mathbb{Z}^d$ and $|V_g[p,q]|$ is odd. In each turn, this algorithm divides the function domain into two parts and specifies the one with a zero point, as guaranteed by Theorem 3 above, and proceeds recursively. Let $n = |p-q|_\infty + 1$, then it uses at most $6n^{d-1}$ queries and $O(d^2(2n)^{d-1})$ time to find a zero point of g, for any $d \geq 2$ and $n > 48d$.

Algorithm $\mathbf{Cut}^d(g, V_g[p,q], k)$

Ensure: $1 \le k \le d$, $q_k - p_k > 1$ and $|V_g[p,q]|$ is odd
1: set $l = \lfloor (p_k + q_k)/2 \rfloor$ and $p' = p[k \leftarrow l]$
set $q' = q[k \leftarrow l]$ and $V_g[p,q'] = V_g[p',q] = \emptyset$
2: **for any** $r \in S = \{ r \in \mathbb{Z}^d \mid \forall\ 1 \le i \le d,\ p'_i \le r_i \le q'_i \}$
query $g(r)$
3: **for any** $(d-1)$-cube $C \in V_g[p,q]$
4: **case** $C \in V[p,q']$ **:** $V_g[p,q'] = V_g[p,q'] \cup \{C\}$
case $C \in V[p',q]$ **:** $V_g[p',q] = V_g[p',q] \cup \{C\}$
5: compute $V = \{$ bad $(d-1)$-cubes in $S \}$
6: **for any** $(d-1)$-cube $C \in V$
add C into both $V_g[p,q']$ and $V_g[p',q]$
7: **if** $|V_g[p,q']|$ is odd, **then** output $(g, V_g[p,q'])$
else output $(g, V_g[p',q])$

Algorithm $\mathbf{FindZero}^d(g, V_g[p,q])$

Ensure: $p < q$ and $|V_g[p,q]|$ is odd
1: **if** $|p-q|_\infty = 1$ **then** query every point $r \in A_{p,q}$ and output a zero point of g
2: **for** $i = 1$ to d
3: **if** $q_i - p_i > 1$, set $(g, V_g[p,q]) = \mathbf{Cut}^d(g, V_g[p,q], i)$
4: output $\mathbf{FindZero}^d(g, V_g[p,q])$

Figure 1: The recursive algorithm $\mathbf{FindZero}^d$

But how can we use $\mathbf{FindZero}^d$ to solve $\mathbf{DZP}^d$? Given a bounded $f \in F[A_{p,q}]$, we extend f to be f' on $A_{p',q'}$, where $p' = p-1$ and $q' = q+1$ as follows: $f' = f$ on $A_{p,q}$. For any $r \in B_{p',q'}$, let i be the largest integer satisfies $r_i = p'_i$ or q'_i. If $r_i = p'_i$, then $f'(r) = +e^i$. Otherwise, we set $f'(r) = -e^i$. It's easy to check that f' is direction-preserving on $A_{p',q'}$. The proof of Lemma 5 can be found in the full version.

LEMMA 5. *For any bounded function $f \in F[A_{p,q}]$, the set $V_{f'}[p',q']$ contains exactly one $(d-1)$-cube, i.e., the $(d-1)$-cube centered at p' and perpendicular to $S = \{1\}$.*

Observation shows that any zero point of f' must also be a zero point of f, so we can call $\mathbf{FindZero}^d(f', V_{f'}[p',q'])$ to compute a zero point of f and

$$Q'(n,d) \le 6(n+2)^{d-1} \le 7n^{d-1} \quad T'(n,d) = O(d^2(2n)^{d-1})$$

for any $d \ge 2$ and $n > 48d$.

The algorithm is showed in figure 1. Here $\mathbf{Cut}^d$ uses set S which is perpendicular to the kth dimension to divide $A_{p,q}$ into two smaller rectangular sets of almost the same size. After querying all the points in S, it chooses one set, which still satisfies the condition of our zero point theorem, to return. The correctness of $\mathbf{FindZero}^d$ is easy to prove.

Let $n = |p-q|_\infty + 1$, then the number of queries used by the d calls to $\mathbf{Cut}^d$ in $\mathbf{FindZero}^d$ is at most

$$n^{d-1} + n^{d-2}(\lfloor n/2 \rfloor + 1) + \ldots + (\lfloor n/2 \rfloor + 1)^{d-1} < 3n^{d-1}$$

under the condition that $n > 48d$. Recurrence can be easily solved to get a $6n^{d-1}$ upper bound for the query complexity of $\mathbf{FindZero}^d$. An implementation of line 5 in $\mathbf{Cut}^d$, which is based on dynamic programming, can be found in the full version. It requires $O(d^2 2^d |S|)$ time, so the d calls to $\mathbf{Cut}^d$ in $\mathbf{FindZero}^d$ totally use $O(d^2 2^d n^{d-1})$ time and we get a $O(d^2(2n)^{d-1})$ upper bound for its time complexity.

3.3 A New Discrete Fixed Point Theorem

It's easy to check that Theorem 1 can be directly proved by Theorem 3 and Lemma 5. Actually, Theorem 3 implies a stronger fixed point theorem. For any direction-preserving map $\mathcal{F}$ from $A_{p,q}$ to $\mathbb{R}^d$, function $f \in F[A_{p,q}]$ can be constructed using the method in section 2 and we have

COROLLARY 1. *If $|V_f[p,q]|$ is odd, then there must exist some fixed point in map $\mathcal{F}$.*

Furthermore, the way we prove Theorem 3 suggests that both Theorem 3 and Corollary 1 can be easily generalized to non-convex domain $D \subset \mathbb{Z}^d$ which is union of d-cubes.

4. A LOWER BOUND FOR DZP

In this section, we define a class of graphs $G_{m,d} = (N_{m,d}, E_{m,d})$, then a game on $G_{m,d}$ between two players, Alice and Bob, is introduced and strategies for Alice are constructed. Finally, we show that these strategies can be used to prove a lower bound for the query complexity of $\mathbf{DZP}^d$.

4.1 Pipe Paths and Sparse Sets

First we define $G_{m,d}$ which looks like a lattice in $\mathbb{Z}^d$.

DEFINITION 7. *For any $m \ge 2$, we define rectangular set $N_{m,d} \subset \mathbb{Z}^d$ as $N_{m,d} = \{ r \in \mathbb{Z}^d \mid \forall\ 1 \le i \le d,\ 1 \le r_i \le m \}$. Let S be some subset of $\mathbb{Z}^d$, then for any $t \in \mathbb{Z}$, the layer t of set S is defined as $S^t = \{ r \in S \mid r_d = t \}$.*

DEFINITION 8. *For any $d \ge 1$ and $m \ge 2 \in \mathbb{Z}^+$ which is a multiple of 256, we define graph $G_{m,d} = (N_{m,d}, E_{m,d})$. Here for any $u, v \in N_{m,d}$, $uv \in E_{m,d}$ iff $\exists\ 1 \le i \le d$ such that $|u_i - v_i| = 1$ and $\forall\ 1 \le j \ne i \le d$, $u_j = v_j$. For any $1 \le t \le m$, the layer t of $G_{m,d}$ is the subgraph spanned by $N^t_{m,d}$. Obviously, it is isomorphic to $G_{m,d-1}$ under the mapping D, where $D(u) = (u_1, u_2 \ldots u_{d-1})$. Given a path $P = u \ldots w$ in $G_{m,d}$, u is called the start vertex and w is called the end vertex of P. If the end vertex of P_1 is same as the start vertex of P_2, then we use $P_1 \cup P_2$ to denote the path concatenated by P_1 and P_2.*

DEFINITION 9. *Path $P = v^1 v^2 \ldots v^k$ in $G_{m,d}$ is said to be monotone if $k = 1$ or*

$$v^1_d + 1 = v^2_d \le \ \ldots\ \le v^{k-1}_d = v^k_d - 1 \text{ or}$$
$$v^1_d - 1 = v^2_d \ge \ \ldots\ \ge v^{k-1}_d = v^k_d + 1$$

For any $v^1_d \le t \le v^k_d$, we use P^t to denote the part of P on layer t of $G_{m,d}$. It is clear that for any monotone path P, P^t is a subpath of P on layer t of graph $G_{m,d}$.

Now we define pipe paths and sparse sets in $G_{m,d}$.

DEFINITION 10. *For $d = 1$, any path P in $G_{m,1}$ is also called a pipe path. For $d \ge 2$, path P in $G_{m,d}$ is called a pipe path if it is monotone and for any $1 \le t \le m$, P^t is either empty or a pipe path in layer t of $G_{m,d}$ (means that $D(P^t)$ is a pipe path in graph $G_{m,d-1}$).*

DEFINITION 11. *Let S be some subset of graph $G_{m,d}$ and vertex $u \in G_{m,d}$. For $d = 1$, S is said to be sparse relative to u in $G_{m,1}$ iff $S = \emptyset$. For $d \ge 2$, S is said to be sparse relative to u iff (**1**) $u \notin S$ and $|S| < l_{m,d} = (m/256)^{d-1}$. (**2**) If $u_d < m$, then S^{u_d+1} is sparse relative to u^+ in layer*

u_d+1 *of* $G_{m,d}$. (**3**) *If* $u_d > 1$, *then* S^{u_d-1} *is sparse relative to* u^- *in layer* u_d-1 *of* $G_{m,d}$. *Here* (**2**) (*and* (**3**) *similarly*) *implies that set* $D(S^{u_d+1})$ ($D(S^{u_d-1})$) *is sparse relative to* $D(u^+)$ ($D(u^-)$) *in graph* $G_{m,d-1}$. *We use* $K_{m,d}$ *to denote the set of all such pairs* (S,u) *in graph* $G_{m,d}$.

Proofs of the following two lemmas can be found in the full version.

LEMMA 6. *For any* $S \subset G_{m,d}$, *we have* $\big|\{\, u \in G_{m,d} \text{ such that } (S,u) \notin K_{m,d} \,\}\big| \le 256^{d-1} m|S|$.

LEMMA 7. *For any given pair* $(S,u) \in K_{m,d}$ *where* $m \ge 12d$, *there exist at least* $m^d/2$ *pipe paths which all start at* u, *end at different vertices and contain no vertex in* S.

4.2 A Game on $G_{m,d}$ and Strategies for Alice

Now we can define a game on $G_{m,d}$ between two players. Assume Alice holds a pipe path P in $G_{m,d}$. At the beginning, she sends a pair $(S,u) \in K_{m,d}$ to Bob and tells him that u is the start vertex of P and $P \cap S = \emptyset$. Bob tries to find the end vertex w of P through the following protocol.

In each round, Bob sends a vertex $v \in G_{m,d}$ to Alice and asks if it is in P. Alice answers according to the following **Rules**: (**1**) If $v \notin P$, then the answer is simply false. (**2**) If $v \in P$ and $v = w$, then the answer is true and Bob wins the game. (**3**) If $v \in P$ but $v \neq w$, then the subpath of P which starts at u and ends at the successor of v is answered. Bob only has $l_{m,d} = (m/256)^{d-1}$ chances. After all these queries, he can ask Alice to show him P and check whether she acts according to the **Rules** above.

For any graph $G_{m,d}$ such that $m \ge 12d$, we'll construct a strategy $T[m,d]$ for Alice which consists of three parts: **Init**(S,u), **Query**(v) and **GetPaths**(). At the beginning, Alice calls **Init**(S,u) to initiate it. In each round, she uses the output of **Query**(v) to answer. Finally, she chooses any pipe path returned by **GetPaths**(), which is consistent with all the answers before, and shows it to Bob.

After the first $s \le l_{m,d}$ queries, we use U_s to denote the set of all those paths output by **Query**(v) before and H_s to denote the union of S and $\{$ vertex $v \mid v$ is queried by Bob before and **Query**$(v) =$ false $\}$.

THEOREM 4. *For any* $m \ge 12d$, *our strategy* $T[m,d]$ *has the following 2 properties:* ($\mathbf{C}_1$) *After initiated by any pair* $(S,u) \in K_{m,d}$, *no matter how Bob queries, the output of* **Query**(v) *is either false or a path that starts at* u, *contains* v *and ends at the successor of* v; ($\mathbf{C}_2$) *After* $l_{m,d}$ *queries,* **GetPaths**() *can output at least* $(m^d/8)$ *pipe paths in* $G_{m,d}$ *which all start at* u *and end at different vertices. Each of them contains all the paths in* $U_{l_{m,d}}$ *and has no vertex in set* $H_{l_{m,d}}$. *In another word, they are all consistent with Alice's answers before.*

Theorem 4 shows that Alice can always employ strategy $T[m,d]$ to beat Bob. Our strategy $T[m,d]$ for Alice is constructed inductively. For $d = 1$, strategy $T[m,1]$ is trivial since we require set S to be empty. For $d \ge 2$, to build $T[m,d]$, we assume strategy $T[m,d-1]$ has already been constructed as $m \ge 12d > 12(d-1)$. Furthermore, strategy $T[m,d-1]$ can be employed to work on any layer of $G_{m,d}$ using the isomorphic mapping D.

During the game, $T[m,d]$ always maintains a pipe path $Q \subset G_{m,d}$. It starts at vertex u and grows away from layer

Algorithm $T[m,d]$**.Init**(S,u)

1: set $Q = uu^+$ and $V = S$
2: create a new strategy $T[m,d-1]$ on the layer u_d+1 of graph $G_{m,d}$ and use $T[u_d+1]$ to denote it
3: call $T[u_d+1]$**.Init**(V^{u_d+1}, u^+)

Algorithm $T[m,d]$**.Query**(v)

1: Assume $Q = uv^1 \ldots v^k$ and $t = v_d^k$
2: **if** $v \in S$ **then** output false
3: **else if** $v \in V$ **then** { v must be queried at some time before } output the same answer
4: **else if** $v_d > t$ **then** output false
5: **else if** $v_d < t$ **then**
6: **if** $v \notin Q$ **then** output false
7: **else** output the subpath of Q that starts at u and ends at the successor of v
8: **else if** $|V^t| < (c_{m,d}-1)$ **then**
9: call $T[t]$**.Query**(v)
10: **if** the output is false **then** output false
11: **else** the output must be a path Q^*, output $Q \cup Q^*$
12: **else** { $|V^t| \ge (c_{m,d}-1)$ }
13: find the smallest l satisfies $l > t$ and $|V^l| < c_{m,d}$
14: call $T[t]$**.GetPaths**() to get a set R of pipe paths in the layer t of graph $G_{m,d}$
15: find a pipe path $Q^* \in R$, whose end w^* satisfies ($\mathbf{G}_1$). for any $v \in V$, $D(w^*) \neq D(v)$ ($\mathbf{G}_2$). V^l is sparse relative to $w^*[d \leftarrow l]$ in layer l
16: delete $T[t]$, create a new strategy $T[m,d-1]$ on the layer l of $G_{m,d}$ and use $T[l]$ to denote it
17: set $Q = Q \cup Q^* \cup (w^* w^*[d \leftarrow t+1] \ldots w^*[d \leftarrow l])$ and call $T[l]$**.Init**$(V^l, w^*[d \leftarrow l])$
18: **if** $v \notin Q$ **then** output false
19: **else** output the subpath of Q which starts at u and ends at the successor of v
20: set $V = V \cup \{v\}$

Algorithm $T[m,d]$**.GetPaths**()

1: Assume $Q = uv^1 \ldots v^k$ and $t = v_d^k$
2: find the smallest l such that $l > t$ and $|V^l| < c_{m,d}$
3: call $T[t]$**.GetPaths**() to get a set R of pipe paths in the layer t of graph $G_{m,d}$
4: find a pipe path $Q^* \in R$, whose end vertex w^* satisfies ($\mathbf{G}_1$). for any $v \in V$, $D(w^*) \neq D(v)$ ($\mathbf{G}_2$). V^l is sparse relative to $w^*[d \leftarrow l]$ in layer l
5: set $Q = Q \cup Q^* \cup (w^* w^*[d \leftarrow t+1] \ldots w^*[d \leftarrow l])$
6: find a set R' of pipe paths in the layer l of $G_{m,d}$ with different end vertices such that every pipe path in R' starts at $w^*[d \leftarrow l]$ and has no vertex in V^l
7: **for any** pipe path $Q' \in R'$ whose end vertex w' satisfies that for any $v \in V$, $D(w') \neq D(v)$ **do**
8: **for any** $l < i \le m$ **do**
output $Q \cup Q' \cup (w' w'[d \leftarrow l+1] \ldots w'[d \leftarrow i])$

Figure 2: Details of Strategy $T[m,d]$

u_d very slowly. If $u_d \le m/2$, then Q will grow from the bottom up. Otherwise, it will grow from the top down. These two cases are really similar, so only the one for $u_d \le m/2$ is described here. Details of strategy $T[m,d]$ are presented in figure 2. Here the constant $c_{m,d} \in \mathbb{R}^+$ is defined as $c_{m,d} = l_{m,d-1}/16 = (m/256)^{d-2}/16$. Proof of Theorem 4 can be found in the full version.

4.3 A Lower Bound for DZP

Theorem 4 can be applied to get a lower bound for problem $\mathbf{DZP}^d$. For any pipe path P in $G_{m,d}$ and $n = 4m+1$, a bounded function $f_P \in F[N_{n,d}]$ can be constructed. The method can be found in the full version, which is straight forward but tedious. It has the following 2 properties: ($\mathbf{D}_1$) function f_P has exactly one zero point. Once it is found, the end vertex of P can be directly decided, thus any algorithm for $\mathbf{DZP}^d$ can be employed to help Bob find the end vertex of P held by Alice. Assume at some time, the algorithm for $\mathbf{DZP}^d$ wants to evaluate $f_P(r)$, then ($\mathbf{D}_2$) at most two queries to Alice are enough. If one of the answers is true, then the game is over. Otherwise, $f_P(r)$ can be decided. Theorem 4 shows that for any $n = 4m+1$ such that $m \ge 12d$, $2Q'(n,d) \ge l_{m,d}$. Thus

$$Q'(n,d) \ge 0.5\,(\lfloor (n-1)/2^{10} \rfloor)^{d-1}$$

for any $d \ge 2$ and $n > 48d$.

5. APPLICATION TO THE APPROXIMATE FIXED POINT PROBLEM

In this section, we first define the approximate fixed point problem [13] with respect to Lipschitz functions. Then three bounds for it are proved, as corollaries of Theorem 2.

Definition 12. *Map $\mathcal{F} : E^d = [0,1]^d \to \mathbb{R}^d$ satisfies a Lipschitz condition with constant L if for any $x, y \in E^d$, $|\mathcal{F}(x) - \mathcal{F}(y)|_\infty \le L\,|x-y|_\infty$. We use $L_{M,d}$ to denote the set of all those maps $\mathcal{F} : E^d \to E^d$ such that $\mathcal{F}(x) - x$ is a map satisfies a Lipschitz condition with constant M.*

Brouwer's fixed point theorem shows that any $\mathcal{F} \in L_{M,d}$ has a fixed point $x^* \in E^d$ which satisfies $\mathcal{F}(x^*) = x^*$. The approximate fixed point problem $\mathbf{AFP}^{M,d,m}$ is defined as follows: given a map $\mathcal{F} \in L_{M,d}$, find an approximate fixed point $x^* \in E^d$ with error 2^{-m}, which actually means that $|\mathcal{F}(x^*) - x^*|_\infty \le 2^{-m}$. Similarly, $\mathcal{F}$ will be taken as an black box to algorithms, which can only be accessed by map evaluations. We use $Q(M,d,m)$ and $T(M,d,m)$ to denote the query complexity and time complexity of $\mathbf{AFP}^{M,d,m}$ respectively. For $d = 2$, [13] proves that $Q(M,d,m) = \Theta(2^m M)$. But for case $d > 2$, there is still a gap between their lower bound and upper bound.

Theorem 5. *For any $d \ge 2$, $2^m M > 192d^3$ and $2^m > 4d$*

$$0.25\,(\lfloor n_2/2^{11} \rfloor)^{d-1} - 1 \le Q(M,d,m) \le 8n_1^{d-1}$$
$$T(M,d,m) = O(d^2(2^{m+1}M)^{d-1})$$

where $n_1 = \lceil 2^m M \rceil$ and $n_2 = \lfloor 2^{m-2} M/d^2 \rfloor$.

For any specific d, it gives matching bounds $\Theta((2^m M)^{d-1})$ for both complexities, thus settles an open problem in [13].

5.1 Two Upper Bounds for AFP

First we prove the two upper bounds for $\mathbf{AFP}^{M,d,m}$. Let $\mathcal{F} \in L_{M,d}$ be the input map of $\mathbf{AFP}^{M,d,m}$, then we can construct $f \in F[A_{p,q}]$ where $p_i = 0$ and $q_i = n_1$, $1 \le i \le d$. For any $r \in A_{p,q}$, the value of $f(r)$ is decided by $\mathcal{F}(x)$ where $x = r/n_1$. If $|\mathcal{F}(x) - x|_\infty < 2^{-m}$, then $f(r) = 0$. Otherwise, let i be the largest integer satisfies $|\mathcal{F}_i(x) - x_i| \ge 2^{-m}$ and set $f(r) = \mathrm{sgn}(\mathcal{F}_i(x) - x_i)\,e^i$. It's not hard to check that the Lipschitz property of $\mathcal{F}$ guarantees that f is both bounded and direction-preserving. Hence, we can use any algorithm for $\mathbf{DZP}^d$ to compute a zero point r^* of f and $x^* = r^*/n_1$ must be an approximate fixed point of $\mathcal{F}$. Each time the algorithm for $\mathbf{DZP}^d$ queries $f(r)$, we use one query on $\mathcal{F}$ and $O(d)$ time to compute $f(r)$. Therefore, $Q(M,d,m) \le Q'(n_1+1,d)$, $T(M,d,m) \le Q'(n_1+1,d)O(d) + T'(n_1+1,d)$ Using Theorem 2, we get the upper bounds in Theorem 5.

5.2 A Lower Bound for AFP

For any d-cube $C \subset \mathbb{Z}^d$ which is centered at $\widetilde{r}$, we define set $V_C \subset \mathbb{R}^d$ as $[\widetilde{r}_1, \widetilde{r}_1 + 1] \times [\widetilde{r}_2, \widetilde{r}_2 + 1] \ldots \times [\widetilde{r}_d, \widetilde{r}_d + 1]$. For any map $\mathcal{F} : C \to \mathbb{R}^d$, the **Cartesian Interpolation** extends it to be a map $\mathcal{F}'$ from V_C to $\mathbb{R}^d$ as follows.

Definition 13. *For any $r \in C$, function $w_r : V_C \to [0,1]$ is defined as $w_r(x) = \prod_{i=1}^d (1 - |x_i - r_i|)$, then*

$$\mathcal{F}'(x) - x = \sum_{r \in C} w_r(x)\,(\mathcal{F}(r) - r) \qquad \forall\, x \in V_C$$

It's easy to check that for any $x \in V_C$, $\sum_{r \in C} w_r(x) = 1$.

Let $c = M/(2d)$, $l = \lfloor c \rfloor$, point p and q satisfy $p_i = 0$ and $q_i = n_2 + 2l$ for any $1 \le i \le d$. For any bounded function $f \in F[N_{n_2+1,d}]$, we construct a map $\mathcal{F}^* \in L_{M,d}$ in 4 steps. The proof of Lemma 8 below can be found in the full version.

($\mathbf{E}_1$) Construct a bounded function $f' \in F[A_{p,q}]$ as: for any $r \in A_{p,q}$ satisfies $l \le r \le l + n_2$, $f'(r) = f(r - l + 1)$. Otherwise, let $1 \le i \le d$ be the largest integer such that $r_i < l$ or $r_i > l + n_2$ and $f'(r) = \mathrm{sgn}(l - r_i)e^i$.

($\mathbf{E}_2$) Construct a map $\mathcal{F}$ on $A_{p,q}$ as $\mathcal{F}(r) = r + cf'(r)$.

($\mathbf{E}_3$) Use the **Cartesian Interpolation** on each d-cube in $A_{p,q}$. In this way, we extend $\mathcal{F}$ to be a map $\mathcal{F}'$ from set $[0, n_2 + 2l]^d$ to $\mathbb{R}^d$.

($\mathbf{E}_4$) For any $x \in E^d$, $\mathcal{F}^*(x) = \mathcal{F}'((n_2 + 2l)\,x)/(n_2 + 2l)$.

Lemma 8. *Map $\mathcal{F}^*$ constructed above must satisfy $\mathcal{F}^* \in L_{M,d}$. Let x^* be an approximate fixed point of map $\mathcal{F}^*$ with error 2^{-m} and $C \subset A_{p,q}$ be any d-cube which satisfies that $(n_2 + 2l)x^* \in V_C$, then there must exist $r^* \in C$ such that $f'(r^*) = 0$ and $f(r^* - l + 1) = 0$.*

For any bounded function $f \in F[N_{n_2+1,d}]$, lemma above shows that any algorithm for $\mathbf{AFP}^{M,d,m}$ can be used to compute a zero point of f and solve $\mathbf{DZP}^d$. We first use it to find an approximate fixed point x^* of $\mathcal{F}^*$. Each time it queries $\mathcal{F}^*(x)$, 2^d queries on f are sufficient to decide the value of $\mathcal{F}^*(x)$. Once we get x^*, 2^d more queries on f are enough to find a zero point of f, according to Lemma 8. As $2^m M > 192\,d^3$ and $n_2 + 1 > 48d$, result in section 4 shows that $0.5\,(\lfloor n_2/2^{10} \rfloor)^{d-1} \le 2^d Q(M,d,m) + 2^d$. This gives us the lower bound in Theorem 5.

6. CONCLUSION AND REMARKS

In establishing the algorithmic complexity for the discrete fixed point problem with the oracle function calls, we develop a deep lower bound proof and a succinct algorithm of an asymptotic matching upper bound. The powerfulness of our approach allows us to close an exponential gap between the upper bound and the lower bound for approximate fixed point algorithms for the classic Brouwer fixed point problem, therefore, settles this problem. The novelty of our upper bound proof may shed new light in algorithm design for other related problems, for example, the fixed point problem for non-convex regions.

The celebrated fixed point theorem of Brouwer followed from a concept of degree, which was also the main idea in many of his other contributions in topology. The idea can be traced back to the Kronecker Integral [22]. Brouwer derived it from a discretrization of the metric space under consideration, as in our definition of badness. Informally, it is the number of "positively" oriented simplices minus the number of "negatively" oriented simplices that maps into regions that covers a given point in the image in the limit as the simplices goes to infinitely small uniformly. Brouwer proved that the value is a constant independent of the choices of the "triangulation" of the domain space. In addition, he showed that the value is invariant under homotopy, if certain conditions are satisfied. The concept of degree with these properties can be applied to derive a series of fixed point theorems. Each of them describes an interesting class of function-domain pairs which guarantee the existence of fixed point.

Degree in the two dimensional case can be simplified to the winding number. If the winding number of a function f around the boundary of its domain is nonzero, it must have a fixed point. Moreover, this function-domain property defined by winding number is dividable. That is, after dividing the domain into two parts, this property still holds in one of them. Therefore, the existence of fixed point is ensured by its existence in the limit. Employing this idea, Hirsch, Papadimitriou and Vavasis got their matching algorithm bound for the two dimensional case. To generalize the winding number to higher dimensions, however, through the boundary conditions, is not easy and has been relied on the discretrization process of Brouwer's definition of degree, which is not suitable for a divide-&-conquer approach to narrow down the existence of fixed points. On the other hand, our discrete fixed point theorem describes a new class of function-domain pairs which is dividable for all dimensions. It allows us to obtain the matching algorithm bound for arbitrary dimensions.

Our oracle model for function evaluation is quite strong. In a more general approach, one may assume the function is presented as a Turing machine. It becomes undecidable if the domain contains all the integers since one may easily reduce the halting problem to it. Papadimitriou [29] and Ko [21] studied interesting properties of the fixed point problem for some interesting class of function-domain pairs, with function evaluations done by Turing machines.

Though our algorithmic matching bound concludes the study of the oracle model, it still leaves room for better algorithms to be designed for specific classes of functions. There have been extensive literatures in algorithms and complexity for finding fixed points of various classes of functions [36, 33, 34, 35, 37, 38, 42], by Sikorski and his co-authors, as well as others. The fixed point problem also has a strong connection with the economic market equilibrium problem which has been recently studied intensively on its algorithmic complexity issues [5, 6, 8, 9, 10, 11, 17, 18, 19]. Our results may contribute new ideas to such studies.

7. ACKNOWLEDGEMENT

The authors would like to thank Frances Yao for many research discussion sessions with us; to Frances and Andrew Chi-Chih Yao for patiently spending hours to listen to the proofs; to Yinyu Ye, Chuangyin Dang and Shanghua Teng for their suggestions and comments; and to Stephen Vavasis for references. We would also thank several anonymous referees of STOC 2005 for valuable comments, references and suggestions.

8. REFERENCES

[1] E. Altman, K. Avrachenkov, and C. Barakat. Tcp network calculus: The case of large delay-bandwidth product. In *Proceedings of INFOCOM 2002*.

[2] K. Arrow and G. Debreu. Existence of an equilibrium for a competitive economy. *Econometrica*, 22:265–290, 1954.

[3] S. Banach. Sur les opérations dans les ensembles abstraits et leur application aux Équations intégrales. *Fundamenta Mathematicae*, 3:133–181, 1922.

[4] L. Brouwer. Über abbildung von mannigfaltigkeiten. *Mathematische Annalen*, 71:97–115, 1910.

[5] N. Chen, X. Deng, X. Sun, and A. C.-C. Yao. Fisher equilibrium price with a class of concave utility functions. In *ESA 2004*, pages 169–179.

[6] B. Codenotti and K. Varadarajan. Efficient computation of equilibrium prices for market with leontief utilities. In *To appear in ICALP 2004*.

[7] R. Cole, Y. Dodis, and T. Roughgarden. Pricing network edges for heterogeneous selfish users. In *STOC 2003*, pages 521–530.

[8] X. Deng, C. Papadimitriou, and S. Safra. On the complexity of equilibria. In *STOC 2002*, pages 67–71.

[9] X. Deng, C. Papadimitriou, and S. Safra. On the complexity of price equilibrium. *Journal of Computer and System Sciences*, 67(2):311–324, 2003.

[10] N. Devanur. The spending constraint model for market equilibrium: algorithmic, existence and uniqueness results. In *STOC 2004*, pages 519–528.

[11] N. Devanur, C. Papadimitriou, A. Saberi, and V. Vazirani. Market equilibrium via a primal-dual-type algorithm. In *FOCS 2002*, pages 389–395.

[12] B. Eaves. Homotopies for computation of fixed points. *Mathematical Programming*, 3:1–22, 1972.

[13] M. Hirsch, C. Papadimitriou, and S. Vavasis. Exponential lower bounds for finding brouwer fixed points. *J.Complexity*, 5:379–416, 1989.

[14] Z. Huang, L. Khachiyan, and K. Sikorski. Approximating fixed points of weakly contracting mappings. *J.Complexity*, 15:200–213, 1999.

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

[16] T. Iimura, K. Murota, and A. Tamura. Discrete fixed point theorem reconsidered. *METR*, 9 2004.

[17] K. Jain. A polynomial time algorithm for computing the arrow-debreu market equilibrium for linear utilities. In *FOCS 2004*, pages 286–294.

[18] K. Jain, M. Mahdian, and A. Saberi. Approximating market equilibria. In *RANDOM-APPROX 2003*, pages 98–108.

[19] K. Jain, V. Vazirani, and Y. Ye. Market equilibria for homothetic, quasi-concave utilities and economies of scale in production. In *SODA 2005*.

[20] R. Kellogg, T. Li, and J. Yorke. Constructive proof of the brouwer fixed point theorem and computational results. *SIAM J. Numer. Anal.*, 13:473–483, 1976.

[21] K.-I. Ko. Computational complexity of fixed points and intersection points. *J.Complexity*, 11:265–292.

[22] L. Kronecker. Über systeme von funktionen mehrerer variabeln. *Montsh. Berl. Akad. Wiss.*, pages 159–163, 688–698, 1869.

[23] H. Kuhn. Simplicial approximation of fixed points. *Proceedings of National Academy of Science, USA*, 61:1238–1242, 1968.

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

[25] A. Mehta, S. Shenker, and V. Vazirani. Profit-maximizing multicast pricing by approximating fixed points. In *ACM Conference on Electronic Commerce*, pages 218–219, 2003.

[26] G. Meinardus. Invarianz bei linearen approximationen. *Arch. Rational. Mech. Anal.*, 14:301–303, 1963.

[27] O. Merrill. *Applications and Extensions of an Algorithm that Computes Fixed Points of Certain Upper Semi-Continuous Point-to-Set Mappings*. Phd thesis, University of Michigan, Annarbor, 1972.

[28] J. F. Nash. Equilibrium points in n-person games. In *Proceedings of NAS*, 1950.

[29] C. Papadimitriou. On graph-theoretic lemmata and complexity classes. In *STOC 1990*, pages 794–801.

[30] J. Ortega and W. Rheinbolt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, 1970.

[31] C. Robinson. *Dynamical Systems, Stability, Symbolic Dynamics, and Chaos*. CRC Press, 1999.

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

[33] S. Shellman and K. Sikorski. A two-dimensional bisection envelope algorithm for fixed points. *J. Complexity*, 18(2):641–659, 2002.

[34] S. Shellman and K. Sikorski. Algorithm 825: A deep-cut bisection envelope algorithm for fixed points. *ACM Trans. Math. Soft.*, 29(3):309–325, 2003.

[35] S. Shellman and K. Sikorski. A recursive algorithm for the infinity-norm fixed point problem. *J. Complexity*, 19(6):799–834, 2003.

[36] K. Sikorski. Fast algorithms for the computation of fixed points. In R. Milanese and A.Vicino, editors, *In Robustness in Identification and Control*, pages 49–59. Plenum Press, New York, 1989.

[37] K. Sikorski, C. Tsay, and H. Wozniakowski. An ellipsoid algorithm for the computation of fixed points. *J. Complexity*, 9:181–200, 1993.

[38] K. Sikorski and H. Wozniakowski. Complexity of fixed points, i. *J. Complexity*, 3(4):388–405, 1987.

[39] S. Smale. A convergent process of price adjustment process and global newton methods. *Journal on Mathematical Economics*, 3:107–120, 1976.

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

[41] D. Spielman and S.-H. Teng. Spectral partitioning works: Planar graphs and finite element meshes. In *FOCS 1996*, pages 96–105.

[42] Z. Yang. *Computing equilibria and fixed points: The solution of nonlinear inequalities*. Kluwer Academic Publishers, Dordrecht, 1999.