Towards Polynomial Simplex-Like Algorithms for Market Equilibria

[Extended Abstract]

Jugal Garg* Ruta Mehta† Milind Sohoni‡ Nisheeth K. Vishnoi§

Abstract

In this paper we consider the problem of computing market equilibria in the Fisher setting for utility models such as spending constraint and perfect, price-discrimination. These models were inspired from modern e-commerce settings and attempt to bridge the gap between the computationally hard but realistic separable, piecewise-linear and concave utility model and, the tractable but less relevant linear utility case. While there are polynomial time algorithms known for these problems, the question of whether there exist polynomial time Simplex-like algorithms has remained elusive, even for linear markets. Such algorithms are desirable due to their conceptual simplicity, ease of implementation and practicality. This paper takes a significant step towards this goal by presenting the first Simplex-like algorithms for these markets assuming a positive resolution of an algebraic problem of Cucker, Koiran and Smale. Unconditionally, our algorithms are FPTASs; they compute prices and allocations such that each buyer derives at least a $\frac{1}{1+\varepsilon}$-fraction of the utility at a true market equilibrium, and their running times are polynomial in the input length and $1/\varepsilon$.

We start with convex programs which capture market equilibria in each setting and, in a systematic way, convert them into linear complementarity problem (LCP) formulations. Then, departing from previous approaches which try to pivot on a single polyhedron associated to the LCP obtained, we carefully construct a polynomial-length sequence of polyhedra, one containing the other, such that starting from an optimal solution to one allows us to obtain an optimal solution to the next in the sequence in a polynomial number of complementary pivot steps. Our framework to convert a convex program into an LCP and then come up with a Simplex-like algorithm that moves on a sequence of connected polyhedra may be of independent interest.

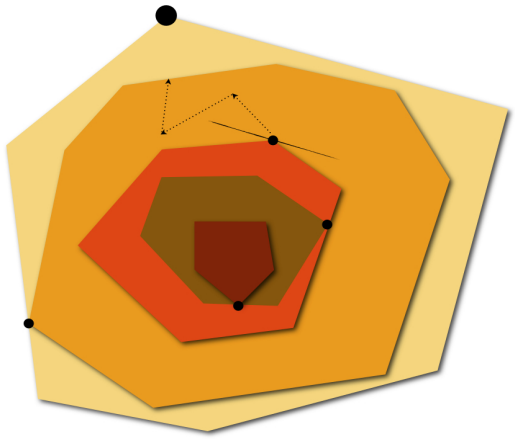

Figure 1: Pictorial description of the algorithm.

1 Introduction

1.1 Market equilibria in the age of e-commerce

The problem of finding an equilibrium price in a market consisting of buyers and goods has interested economists for more than a century due to its relevance in modeling real-world situations; mathematical economists have studied its connection with equilibrium theory and, more recently, computer scientists have focused on finding efficient algorithms for applications to e-commerce. The following market model was formulated in the late 1800's and is referred to as the Fisher model: Consider a market which comprises of a set $\mathcal{G}$ of divisible goods and a set $\mathcal{B}$ of buyers with $n \stackrel{\text{def}}{=} |\mathcal{G}|$ and $m \stackrel{\text{def}}{=} |\mathcal{B}|$. Buyer $i \in \mathcal{B}$ has a fixed amount of money, say M_i, which it can spend on buying goods, and its utility from a bundle of goods is determined by a non-negative, non-decreasing, concave function $U_i : \mathbb{R}^n_+ \to \mathbb{R}_+$. Given the prices of goods, each buyer acts selfishly and buys a (optimal) bundle that maximizes its utility to the extent allowed by its money. A setting of prices is referred to as *market clearing* if, after each buyer is given an optimal bundle, there is no deficiency or surplus of any good, and the money of all the buyers is exhausted, i.e., the market clears. The problem of market equilibria is to find such prices when they exist. It was only in 1954, that existence of market clearing prices, or *market equilibrium*, was proved by Arrow and Debreu [4], albeit, under mild

*Indian Institute of Technology, Bombay. jugal@cse.iitb.ac.in
†Indian Institute of Technology, Bombay. ruta@cse.iitb.ac.in
‡Indian Institute of Technology, Bombay. sohoni@cse.iitb.ac.in
§Microsoft Research India. nisheeth.vishnoi@gmail.com

conditions on the utilities.[1] Unfortunately, their proof of existence of market equilibrium relied on a fixed point theorem and did not yield any finite-time algorithm,[2] let alone an efficient one. Subsequent to the work of Arrow and Debreu, the problem of computing such equilibria received a great deal of attention from economists and mathematicians. Notably, Scarf [31] and Smale [33] gave algorithms for finding fixed points and, as an application, obtained algorithms for approximately computing market equilibria in general settings. Given their generality, it was not surprising that such algorithms were very slow and the search for more efficient algorithms continued. One approach to facilitate the search for efficient algorithms was to make assumptions, and consider utilities which model reality to a reasonable degree.[3] One such model was that the utilities are separable, piecewise-linear and concave (SPLC) functions. Separability means that buyer i's utility for a good j does not depend on any other good, while a PLC function can be used to approximate a concave function capturing the utility of the i-th buyer for the j-th good.[4] A special case, when there is exactly one linear piece in each utility is referred to as the linear model. While linear markets succumbed to efficient algorithms due to the existence of convex programming formulations for them [19, 28, 23], for SPLC, it was shown in a remarkable series of work [13, 9, 10] that efficient algorithms may be impossible. Even though the case of linear utilities turned out to be efficient and mathematically elegant, it was noted that it is not the most relevant model in practice. The reason being that at an equilibrium price, a buyer could spend all its money on a small number (even one) of goods, because its marginal utility remains the same for these goods even after procuring them in large quantities.

The last decade showed a renewed interest in algorithmic study of market equilibria due to the boom in e-commerce. With the advent of the Internet, new possibilities for markets emerged, such as with online advertising (Google, Bing) or online sales companies such as Amazon. Importantly, the scale brought on to the computational problems in these settings meant that efficiency and practicality would be key in evaluating an algorithm. Among the new market models introduced, the ones relevant to this paper are the spending constraint model [37] and the perfect, price-discrimination model [22]. In the spending constraint model, the utilities are SPLC and the buyer can spend at most a prescribed budget on each segment. Due to the utilities being SPLC, this model goes beyond the limitations imposed by linear markets and, unlike the SPLC case, it has a polynomial time algorithm. Further, as argued in [37], it captures the popular AdWords market to a high degree. In the perfect, price-discrimination model, apart from buyers and sellers, there is a middle-man who buys goods from the sellers according to the prices set by them and sells them to the buyers at prices they are willing and able to pay. The buyers still have SPLC utilities similar to the SPLC model. This model is aimed at capturing online display-ad marketplaces and, for this, a polynomial time algorithm to compute an equilibrium was given by [22].

1.2 A plethora of algorithms As is the case with many fundamental problems that are practically relevant, for instance matching, linear programming and graph partitioning, a large variety of algorithms to compute market equilibria have been developed for several market settings, each with its own virtues: While algorithms based on local dynamics [41, 14, 11, 12, 42] do not need any central authority for computing an equilibrium and also help us in understanding how the market might reach an equilibrium, combinatorial, flow-based algorithms [17, 22, 30, 37, 40, 39] reveal structural insights based on interesting market mechanisms; see Chapter 5 in [36] for an extensive survey. Complementary-pivoting algorithms or Simplex-like algorithms [18, 2, 21], on the other hand, are natural choices for several reasons; they are conceptually simpler, much easier to implement and run very fast in practice in spite of possible worst case exponential bounds. The situation here can also be compared to that for linear programming where, in spite of provably efficient algorithms and known exponential worst-case bounds for natural pivoting rules [20], the method of choice in practice has been Simplex-like pivoting algorithms.[5]

However, the problem of whether there are polynomial-time, Simplex-like algorithms for tractable market models, even for linear markets, has remained elusive. In fact, a peculiar feature of market equilibrium problems is that, even though there are no LP formulations known for them, there are strongly polynomial

[1] In fact these results were for a more general model where the money is not fixed and each buyer comes with an endowment of goods and the money is obtained by selling these goods. Such a setting is referred to as the Arrow-Debreu setting.

[2] Unless explicitly stated, by an algorithm, we do not necessarily mean an efficient (polynomial-time) algorithm; just a procedure which terminates in finite time.

[3] Note that the way we have defined the problem we have already assumed that the utility of buyer i depends only on the bundle of goods allocated to it and not on the allocation to others.

[4] Concavity of the utility function models the law of diminishing marginal returns.

[5] Theoretical results such as smoothed analysis of the Simplex algorithm [34] strongly justify this practice in hindsight.

time algorithms [30, 40] in many settings, and the solution to the convex programs for them can be proved to be *rational* if the input is rational [7, 22, 38]; a rarity in the world of convex programming. By an optimist, this could be taken as evidence that there may even be LP formulations for market equilibria settings and, hence, Simplex-like algorithms for these problems and, indeed, we consider this as an important open problem.

1.3 Our contribution We make significant progress towards the goal of finding Simplex-like, polynomial-time algorithms for market equilibria by providing (conditional) polynomial-time algorithms and unconditional FPTASs for the spending constraint and the perfect, price-discrimination models, both of which include linear markets in the Fisher model as a special case. We describe this informally in the following theorem.

THEOREM 1.1. (MAIN THEOREM- INFORMAL)
Assuming the positive resolution of an algebraic problem of Cucker, Koiran and Smale (see Problem 1.1), there are Simplex-like polynomial-time (in n, m, and the bit length required to encode the input) algorithms to compute the market equilibrium **exactly** *in the spending constraint and perfect, price-discrimination Fisher market models. The algorithms are Simplex-like in the sense that they pivot on the 1-skeleton of a sequence of polyhedrons and that there is a potential function for each polyhedra, which is monotone on the traced path. The total number of pivots is polynomial in the input size.*

Unconditionally, our algorithms can be proved to be FPTASs which compute strongly-approximate market equilibria in the sense that, given $\varepsilon > 0$, they compute prices and allocations such that each buyer derives at least a $\frac{1}{1+\varepsilon}$-fraction of the utility at a true market equilibrium. They run in time polynomial in the input size and $1/\varepsilon$.

We develop one set of techniques that works for both the models. We start with convex programming formulations, of [7] for the spending constraint model and, of [22] for the perfect, price-discrimination model. Subsequently, from the KKT conditions for these programs, we derive the first linear complementarity problem (LCP)-like formulations.[6] To an LCP-like formulation, one can associate a natural polyhedron by dropping the quadratic complementarity conditions. It can be shown that a solution of this LCP-like formulation occurs at a vertex of this polyhedron. One strategy then is to start at an arbitrary vertex of this polyhedron and pivot until an optimal vertex is reached. This Lemke [26] or Lemke-Howson [27] type approach has been attempted in the past by [18, 2] for linear markets but it is far from clear how to obtain a polynomial guarantee for these algorithms.

Our algorithm departs from previous approaches (for linear markets). Starting with an instance I of the market model at hand, our algorithm carefully constructs a sequence of instances $I^0, \ldots, I^t = I$ such that 1) I^l and I^{l+1} differ in the utility of exactly one segment for all l, and 2) given an optimal solution to the LCP corresponding to I^l, the algorithm obtains the optimal solution to I^{l+1} in a finite number of pivots. The way the sequence is constructed, an optimal solution of I^l violates exactly one complementarity condition for the instance I^{l+1}. Thus, the solution of I^l sits on the 1-skeleton of the polyhedron corresponding to I^{l+1}. What about polynomial number of pivots? The main technical result of the paper is to show that if all the segments in the utilities for all buyers are of the form $U_{ijk} = (1 + 1/q)^{n_{ijk}}$ for $q \in \mathbb{Z}_+$ then, employing a scaling technique, a polynomial-sized sequence of instances can be created such that the pivoting algorithm takes a polynomial number of steps on each. The question then is how to find a starting point. This can be done if we start with an instance where all utilities are set to the (same) value, which is the maximum of utilities over all segments. Overall the number of pivots and the number of polyhedrons turn out to be polynomial assuming all utilities are integer powers of the same $\alpha = 1 + 1/q$. The original utilities may not have such a form, but they can be approximated to a degree of accuracy which is enough to recover the exact market equilibrium. However, if we approximate the utility $U_{ijk} \sim \alpha^{n_{ijk}}$, the precision in this approximation that we require to compute the exact market equilibrium forces us to pick q and n_{ijk}s which can require polynomial number of bits to write down. Thus, each pivoting step in our algorithm, which requires to determine which hyperplane will become tight first, can be shown, in our case, to reduce to the following problem.

PROBLEM 1.1. *Given $\alpha = 1 + 1/q$ with $q \in \mathbb{Z}_{>0}$, positive integers $n_1, \ldots, n_l$, and $\sigma_i \in \{-1, 1\}$, the problem is to check the sign of the expression $\sum_{i=1}^{l} \sigma_i \alpha^{n_i}$ in polynomial time, i.e., in time $\mathrm{poly}(l, \log q, \log n_{max})$, where $n_{max} \stackrel{\text{def}}{=} \max_i n_i$.*

Hence, our algorithm will be truly polynomial time if there is a polynomial time procedure to above problem. In fact, Cucker, Koiran and Smale [15] gave an algorithm for this problem when α is integer in their

[6] It would be interesting to investigate if this approach can be generalized to derive LCPs for rational convex programs [38].

attempt to find integer roots of a lacunary polynomials. This problem appears as an open problem in their paper.

To obtain unconditional results, due to our current inability to solve Problem 1.1, we approximate the input utilities as powers of $1+\varepsilon/s$, where $\varepsilon > 0$ is the parameter in the FPTAS and s is the total number of segments in the instance. While the problem of determining signs and, hence pivoting, becomes easy, we still need to prove that an equilibrium for these approximate utilities is ε-strongly approximate. This requires us to do a sensitivity analysis of the corresponding LCP which seems daunting. We bypass this and, interestingly, use our *basic* algorithm as a proof technique to establish the claim about the utilities: The utilities derived by the buyers from the allocation and prices produced by our algorithm is not less than a $\frac{1}{1+\varepsilon}$ multiplicative factor of that each derives from a true market equilibrium.

Summarizing, we present the first Simplex-like algorithms which pivot polynomially many times for market models important in e-commerce settings. In fact, these are the first such results for any market model. Moreover, initial experiments suggest that these algorithms run very fast in practice on randomly generated instances. Conceptually, our framework to convert a convex program into an LCP and then come up with a Simplex-like algorithm that moves on a sequence of connected polyhedra may be of independent interest and should find more applications. Finally, our algorithm moves in the utility space and is an instantiation of the homotopy method which has been deployed in other powerful polynomial-time algorithms such as that for computing the permanent of a non-negative matrix [25]. Besides helping us prove a polynomial bound on the number of pivots, moving in utility space could be useful in other scenarios. Firstly, in practical situations, the utilities are the most conjectural of all data. Hence, such an algorithm may shed some light on how the buyers discover their utilities gradually. Secondly, it has been shown in [1] that buyers may actually benefit by reporting false utilities. Whence, a game-theoretic analysis would benefit from an algorithm which works in the utility space. Our algorithm may also be useful in a repeated market setting where buyers are indifferent among the goods to start with (may be the market is new and the buyers are not sure about the relative worth of the goods), and they learn their utilities gradually as they consume.

1.4 Organization of the paper In Section 2 we present an overview of our results. We first define the spending constraint model in Section 2.1 and derive an LCP-like formulation for it in Section 2.2. The algorithm description and main theorem appear in Section 2.3. The claims about the running time are presented in Section 2.4. A detailed overview of the proofs corresponding to Section 2.3 are presented in Section 4. In this version of the paper we omit all formal proofs. A precise description of the algorithms appear in Figures 2.1 and 2.2. The perfect, price-discrimination model, and its LCP-like formulation are described in Section 3. We present related work in Section 5. Section A presents a (general) technique to convert a convex programming formulation for the spending constraint model to an LCP-like formulation. For details on how to obtain a starting solution and degeneracy, we refer the reader to the full version.

2 Overview of the algorithm and results

In this section we present in detail the main technical ideas behind the algorithm mentioned in Theorem 1.1. As noted, when all the utilities in the spending constraint market or the perfect, price-discrimination are linear then both the settings reduce to linear markets. Hence, we will not present a separate algorithm and proof for this case. Further, to illustrate the key ideas, in this section, we focus on the spending constraint market. Towards the end we highlight what needs to be done additionally for the perfect, price-discrimination market. This section is organized as follows: First we start with a brief description of Fisher markets, market equilibrium conditions and show how to augment this model with spending constraints. Then, we present a characterization of market equilibrium in these markets and, based on this characterization, we derive an LCP-like formulation to capture market equilibria *exactly*. Subsequently, we describe the *basic* pivoting algorithm (see Figure 2.1) whose input consists of an optimal solution with respect to a set of utilities and a utility which has been reduced. This algorithm outputs an optimal solution for the new utilities. Then we present the *scaling* algorithm (see Figure 2.2) which combines the basic algorithm with a scaling technique and speeds it up. Finally, we present the main theorems regarding the algorithms. Our hope is that this section should be able to convey the key ideas in sufficient detail to convince the reader of their importance and correctness of our techniques. The important technical details are given in Section 4 and for the detailed proofs, we refer the reader to the full version.

2.1 The spending constraint model Recall that in the Fisher market model there is a set $\mathcal{G}$ of divisible goods and a set $\mathcal{B}$ of buyers with $n \stackrel{\text{def}}{=} |\mathcal{G}|$ and $m \stackrel{\text{def}}{=} |\mathcal{B}|$. Each buyer $i \in \mathcal{B}$ has money M_i which is

fixed. Its utility from a bundle of goods is defined by a non-negative, non-decreasing, concave function $U_i : \mathbb{R}_+^n \to \mathbb{R}_+$. At given prices, each buyer buys a utility maximizing (optimal) bundle, subject to the money constraint. A price vector is said to be *market clearing* if, after each buyer is given an optimal bundle, there is no deficiency or surplus of any good, and the money of all the buyers is exhausted, i.e., the market clears. The problem is to find such market clearing or *equilibrium prices* when they exist. Without loss of generality (w.l.o.g.) we assume that the quantity of every good is one. We will be concerned with utilities which are separable, i.e., for buyer i, its utility for good j only depends on the amount of good j it has been allocated. In the case when these utilities are linear, buyer i derives utility at a constant rate U_{ij} from good j up to any amount. Its total utility from a bundle $x = (x_1, \ldots, x_n)$ of goods is $\sum_{j\in\mathcal{G}} U_{ij}x_j$, a linear function. At prices $p = (p_1, \ldots, p_n)$, where p_j is the price of good j, buyer i buys a (optimal) bundle x that maximizes $\sum_{j\in\mathcal{G}} U_{ij}x_j$ subject to $\sum_{j\in\mathcal{G}} x_j p_j \leq M_i$ and $x_j \geq 0, \forall j \in \mathcal{G}$. It is known (and intuitively clear) that buyer i will buy only those goods for which U_{ij}/p_j, the utility from a unit amount of money, is maximized. U_{ij}/p_j is referred to as the *bang-per-buck* for (i, j).

In the *spending constraint* model, the utilities are a generalization of linear; separable, piecewise-linear and concave (SPLC). For a buyer i and good j, the utility function is piecewise-linear concave with s_{ij} linear segments; on segment $1 \leq k \leq s_{ij}$ buyer i derives utility at a constant rate U_{ijk}. The key difference between the SPLC model and the spending constraint model is that, here every segment has a budget bound of B_{ijk} instead of a bound on the amount of good: Buyer i is allowed to spend at most B_{ijk} money on the k-th segment corresponding to good j. Here B_{ijk}s are strictly positive and concavity implies that $U_{ijk} > U_{ij(k+1)}$ for $1 \leq k < s_{ij}$. If x_{ijk} is a variable that denotes the amount of good j allocated to buyer i for segment k, then $U_{ijk}x_{ijk}$ is the corresponding utility derived from this segment and the budget constraint implies that $x_{ijk}p_j \leq B_{ijk}$ for a fixed price vector $p = (p_1, \ldots, p_n)$. Therefore, at prices p, the budget-constrained, utility-maximization goal of buyer i is captured by the following optimization problem. A solution to this program is called its optimal bundle. We will assume, w.l.o.g., that all $U_{ijk}, B_{ijk} > 0$ and $M_i > 0$.

$$\begin{array}{l} \text{maximize } \sum_{j\in\mathcal{G}} U_{ijk}x_{ijk} \\ \sum_{j\in\mathcal{G}, 1\leq k\leq s_{ij}} x_{ijk}p_j \leq M_i \\ \forall j \in \mathcal{G},\ \ \forall 1 \leq k \leq s_{ij} : x_{ijk} \geq 0,\ x_{ijk}p_j \leq B_{ijk} \end{array} \tag{2.1}$$

2.2 Deriving an LCP formulation for the spending constraint model From the Karush-Kuhn-Tucker (KKT) conditions applied to the linear program (2.1), we can derive the following characterization of an optimal bundle for buyer i: Sort the segments $(i, j, k), \forall j, k$ by decreasing order of their bang-per-buck (U_{ijk}/p_j), and partition them into $P_1, P_2, \ldots$ such that each partition contains all segments with the same bang-per-buck and the bang-per-buck of $P_i > P_{i+1}$. Hence, at prices p, the segments in P_l make i strictly happier than those in $P_{l+1}, P_{l+2}, \ldots$. Therefore, a strategy for it to maximize its utility is to start buying partitions in increasing order, until all its money (M_i) is exhausted. Suppose it exhausts all its money at t-th partition. We refer to all partitions before P_t as *forced* (as they are bought fully), P_t is called *flexible* (as it be bought partially), and all the subsequent are referred to as *undesired* (as they can not be bought at all).

Let q_{ijk} denote the quantity of money spent by buyer i on the k-th segment for good j, i.e., $q_{ijk} = x_{ijk}p_j$, and let λ_i denote the inverse of the bang-per-buck of the flexible partition P_t for buyer i. Note that λ_i will be strictly positive assuming that prices are non-zero. The reason for using inverse is to get linear constraints, and will be clear shortly. First, for buyer i, we rewrite the conditions for forced, flexible and undesired partitions formally in-terms of λ_i. $\forall(j, k)$:

if	$\frac{U_{ijk}}{p_j} > \frac{1}{\lambda_i}$	then	$q_{ijk} = B_{ijk}$	(Forced)
else if	$\frac{U_{ijk}}{p_j} = \frac{1}{\lambda_i}$	then	$0 \leq q_{ijk} \leq B_{ijk}$	(Flexible)
else	$\frac{U_{ijk}}{p_j} < \frac{1}{\lambda_i}$	then	$q_{ijk} = 0$	(Undesired)

Clearly, at prices p, the above conditions are satisfied by an optimal bundle for buyer i. However, our goal is to capture (get sufficient conditions for) equilibrium prices. To capture the *Boolean-like* conditions above, we use the expressive power of LCPs. Now p_j-s will no longer be fixed and will act as variables. Hence, $\lambda_i (> 0)$ will also be an indeterminate and we would like it to act as inverse bang-per-buck of the flexible partition for buyer i. q_{ijk}s remain as money allocation variables. Note that the left hand side predicate in the above conditions can be linearized and re-written as $\text{sgn}(U_{ijk}\lambda_i - p_j)$; where "+" denotes forced, "0" flexible and "−" undesired. To capture this through a single linear constraint, we eliminate the "+" case by introducing slack variables γ_{ijk} which act as *price supplements* for forced segments so that their bang-per-buck can be equated to $1/\lambda_i$. Formally, the linear constraints we impose are: $\forall(i, j, k)$ $0 \leq q_{ijk} \leq B_{ijk}$ and $U_{ijk}\lambda_i - p_j - \gamma_{ijk} \leq 0$. Having done this the above three conditions can be rewritten as: $\forall(i, j, k),\ \ \gamma_{ijk} > 0 \Rightarrow q_{ijk} = B_{ijk}$ (forced), and

$U_{ijk}\lambda_i - p_j - \gamma_{ijk} < 0 \Rightarrow q_{ijk} = 0$ (undesired). Putting all these together, we get an LCP-like formulation given in Table 1, call it **LCP 1**.

As an example of how to use these complementarity conditions to derive the Boolean conditions above, note that if $\gamma_{ijk} > 0$ then $q_{ijk} = B_{ijk}$. Since $B_{ijk} > 0$ by assumption, $q_{ijk} > 0$. This implies that $U_{ijk}\lambda_i - p_j - \gamma_{ijk} = 0$ which implies that ${}^{U_{ijk}}/_{p_j} > {}^{1}/_{\lambda_i}$.

Conditions (2.5) and (2.6) of **LCP 1** are included to enforce market clearing. The non-negativity for p_js is implied by (2.5) and non-negativity of q_{ijk}s. Conditions (2.2), (2.3) and (2.4) of **LCP 1** have two parts. We refer to the left part by a and the right part by b. For example constraint $\lambda_i \geq 0$ is (2.2.a) and $q_{ijk} \geq 0$, $\gamma_{ijk} \geq 0$ by (2.2.b). Every inequality except (2.2.a) participates in some complementarity condition of (2.4). From the way we derived **LCP 1**, it is self-evident that a market equilibrium will satisfy it. The next theorem shows that the conditions of **LCP 1** are also sufficient and they capture all the market equilibria.

THEOREM 2.1. (LCP FOR SPENDING CONSTRAINT) *Any p, q, γ, λs that satisfy (2.2), (2.3), (2.4), (2.5) and (2.6) can be used to produce a market equilibrium for the corresponding spending constraint model.*

Finally, even though this may seem a specific way to derive an LCP-like formulation for the spending constraint model, such formulations can be derived in a systematic manner from convex programs such as that of [32, 7] and may be of independent interest, see Appendix A. Concretely, this technique allows us to derive an LCP for the perfect, price-discrimination market using a convex program of [22].

2.3 A Simplex-like algorithm We start by noticing that in the LCP in Table 1, the constraints (2.2), (2.3), (2.5) and (2.6) are linear and while the set of constraints (2.4) are quadratic (complementarity). It is not immediately clear how to solve **LCP 1** efficiently through a Simplex-like method due to its quadratic constraints. One possibility is to apply Lemke's scheme on **LCP 1**. However it is not guaranteed to converge and, even if it does, there does not seem to be a way to prove a bound on number of pivoting steps. Given their generality, complementary-pivoting algorithms are unlikely to be polynomial in general. However, can we exploit the structure of our problem to get polynomial algorithms? A step towards this appears in the works of [18, 2] which apply Lemke-type frameworks on an LCP for *linear* Fisher markets and show that it converges in finite time. However, even ignoring the fact that their algorithm makes sense only for linear markets, there seems to be little hope of proving a polynomial bound on the number of pivots.

We now present our algorithm for the spending constraint model. The algorithms formally appear in Figures 2.1 and 2.2. Our algorithm departs from previous approaches (for linear markets) that try to pivot on one polyhedron. Instead, starting with an instance (U, B, M) of the spending constraint model, we carefully construct a sequence of instances $(U^0, B, M), \ldots, (U^t, B, M) = (U, B, M)$ such that, roughly, the following properties hold. (Here U and B respectively are short for the utilities U_{ijk}s and budget restrictions B_{ijk}s of the segments, and M for the money M_is of buyers.)

1. Dropping the complementarity constraints from the **LCP 1** for (U^l, B, M), we obtain a polyhedron P^l which contains P^{l-1} and for which we are given a *very good* starting point S^{l-1}.

2. We give a complementary-pivoting scheme which, starting from S^{l-1} is able to obtain the optimal solution to the **LCP 1** for (U^l, B, M). This solution serves as the starting point for the next round and is denoted by S^l.

3. (U^0, B, M) are chosen such that we can produce the optimal solution S^0 easily.

4. Crucially, we prove that the number of pivots done on each polyhedron, and the number of polyhedrons in the sequence, are both (small growing) *polynomials* in the bits needed to describe the input (U, B, M).

The picture in Figure 1 is a graphic illustration of this algorithm. Hence, unlike other polynomial time algorithms, the algorithm is simple both conceptually and from the point of view of implementation. The difficulty is moved to proof of correctness. Now we describe the steps and the intuition behind them in more details.

Let $P(U, B, M)$ be the polyhedron in the (λ, p, q, γ)-space defined by the linear constraints (2.2), (2.3), (2.5) and (2.6), while the quadratic constraints of (2.4) are dropped.

Theorem 2.1 implies that it is sufficient to find a point in $P(U, B, M)$ which satisfies the complementarity conditions of (2.4); call such a point a *solution* of $P(U, B, M)$. Wishfully thinking, suppose we could choose U. Then the simplest choice would be to set all the U_{ijk}s to the same value and then a solution is recovered by setting all prices to be the same[7], from

[7] This is under the assumption that $\forall j \in \mathcal{G}$, $\sum_{i,k} B_{ijk} \geq \sum_i {}^{M_i}/_n$. Refer to full version for the general case.

$$\forall(i,j,k): \quad \lambda_i \geq 0 \quad \text{and} \quad q_{ijk} \geq 0,\ \gamma_{ijk} \geq 0 \tag{2.2}$$

$$\forall(i,j,k): \quad q_{ijk} \leq B_{ijk} \quad \text{and} \quad U_{ijk}\lambda_i - p_j - \gamma_{ijk} \leq 0 \tag{2.3}$$

$$\forall(i,j,k): \quad \gamma_{ijk}(q_{ijk} - B_{ijk}) = 0 \quad \text{and} \quad q_{ijk}(U_{ijk}\lambda_i - p_j - \gamma_{ijk}) = 0 \tag{2.4}$$

$$\forall j \in \mathcal{G}: \quad \textstyle\sum_{i,k} q_{ijk} = p_j \tag{2.5}$$

$$\forall i \in \mathcal{B}: \quad \textstyle\sum_{j,k} q_{ijk} = M_i \tag{2.6}$$

Table 1: **LCP 1** - An LCP-like formulation for the spending constraint Fisher market Model

which a feasible allocation can be computed. Now observe that if we change U_{ijk} for exactly one (i,j,k), then exactly one inequality of $P(U,B,M)$ is modified, namely $U_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} \leq 0$. Moreover, if we decrease U_{ijk}, then the polyhedron expands, containing all the previous feasible points. Let the solution of $P(U,B,M)$ be S and let the new utility for (i,j,k) be denoted by U'_{ijk}. Note that if q_{ijk} is zero at S, then S still remains a solution of $P(U',B,M)$ where U' is the same as U except at (i,j,k) where it is U'_{ijk}. However, if $q_{ijk} > 0$, then S violates the complementary constraint $q_{ijk}(U'_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk}) = 0$ in $P(U',B,M)$.

Assume that $P(U,B,M)$ and $P(U',B,M)$ are *non-degenerate* for the simplicity of ensuing arguments. At this point let us discuss some properties of a non-degenerate polyhedron, which are relevant to our complementary-pivoting algorithm. Consider a polyhedron P in an r-dimensional space, represented by a set of inequalities. It is called non-degenerate if at any point of this polyhedron the set of tight inequalities are linearly independent. In that case at a vertex (0-dimensional facet) of P exactly r inequalities are tight, on an edge (1-dimensional facet) $r-1$ are tight, and in general on i-dimensional facet $r-i$ are tight. The set of i or less dimensional facets of P is called an i-skeleton of P. As an edge of P is 1-dimensional, there are only two possible directions to move on it.

Let the total number of segments among all the utility functions be denoted by s. Hence, $P(U,B,M)$ is $2s$-dimensional as (2.5) and (2.6) imposes $m+n$ linearly independent equalities. Since, the complementarity conditions of (2.4) require $2s$ inequalities to be tight at S in $P(U,B,M)$, it forms a vertex. Changing U_{ijk} to U'_{ijk} relaxes exactly one inequality at S, hence S sits on an edge of $P(U',B,M)$ and now there are exactly two feasible directions to move. Recall that S violates only $q_{ijk}(U'_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk}) = 0$ and the goal is to establish it. Among these two directions, there is an obvious one, namely towards $U'_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} = 0$, as we want to reestablish the violated complementarity condition. The reason behind changing exactly one U_{ijk} is to ensure this uniqueness of the direction for movement (unambiguity). If we decrease more than one U_{ijk}s, then we end up on a higher dimensional facet which may have an infinite number of possible directions to move.

At the next vertex, say v, a new inequality, say I, becomes tight. If it corresponds to the constraint mentioned above (involving U'_{ijk}), then we are done. Otherwise, assuming it is not one of (2.2.a), it belongs to some complementarity condition, say C_I. Let I' be the other part of the complementarity condition C_I, i.e., C_I is $II' = 0$. Since, C_I was satisfied on the edge we took into the vertex v, I' was already tight on it. Therefore, at v, I and I' both are tight. For all other complementarity conditions, exactly one of the participating inequality is tight at v. Hence, in order to maintain all of (2.4), except for the violated constraint, we have only two choices - either relax I or relax I'. If we relax I then we go back on the previous edge. Therefore, in order to move forward there is no other choice than to relax I'. Such a pivoting is called **complementary pivoting**, where based on the new tight inequality at the vertex, the other inequality in its complementarity condition is relaxed in order to maintain all the complementarity constraints. Our algorithm traces a path by doing a complementary pivoting at every intermediate vertex until the violated constraint is satisfied. We will argue later that this path does not cycle and terminates at a solution vertex S' of $P(U',B,M)$.

A bit more formally, we start our algorithm with utilities U^0 where all U^0_{ijk}s are set to the maximum value among the input U_{ijk}s. For this input we can find a vertex of its polyhedron which is also an optimal solution of (U^0,B,M), say S^0. At S^0 all the prices are set to the same value $\sum_i M_i/n$. In that case all the segments give the same bang-per-buck with respect to U^0 and, hence, we can set γ_{ijk}s to zero and λ_is accordingly. q_{ijk}s can also be determined without much effort. Such an S^0 forms a vertex (though degenerate) of $P(U^0,B,M)$. Now, the key is to reduce the U^0_{ijk}s to their original values U_{ijk}s, one at a time, while keeping track of the optimal solutions. In this process, we construct a se-

quence $\{U^l\}_{l\geq 0}$ and their solutions $\{S^l\}_{l\geq 0}$ such that $U^0 \geq U^1 \geq \cdots \geq U^N = U$, where U^l and U^{l+1} differ only at a specific index (i,j,k) with $U^{l+1}_{ijk} < U^l_{ijk}$. As a consequence, $P(U^l, B, M) \subseteq P(U^{l+1}, B, M)$ and, thus, S^l is feasible in $P(U^{l+1}, B, M)$. If $q_{ijk} = 0$ at S^l, then S^{l+1} remains the same as S^l. Otherwise we move on the 1-skeleton of $P(U^{l+1}, B, M)$ using complementary pivoting as explained above to reach at S^{l+1}. Since (B, M) is always the same in all the iterations, for subsequent sections, we write the polyhedron $P(U, B, M)$ as $P(U)$. This completes the overview of our basic algorithm (see Figure 2.1).

Our algorithm is Simplex-like as it moves on 1-skeleton of an expanding polyhedron. In each iteration, the algorithm follows a Simplex-like path. Further, unlike all previous algorithms, our algorithm moves in *utility space*, while B and M are fixed and market clearing is enforced (2.5), (2.6). The advantages of these features are discussed in the introduction.

2.4 Running time While it is clear that the number of different polyhedra the algorithm, described in the previous section, will pivot on is finite, it is not clear why the algorithm does not get stuck or cycle in any iteration. The first main technical claim about our algorithm is that, in an iteration corresponding to segment (i,j,k), λ_i monotonically increases, and p_j and q_{ijk} monotonically decrease on the path followed by the algorithm, starting from S^l in $P(U^{l+1})$. Further, one of these three increase/decrease is strict in every pivoting step. This implies that during this iteration the algorithm can not cycle and no vertex is visited twice. Finally, one has to argue that our algorithm will never venture on an infinite ray and, hence, terminates at S^{l+1} in finitely many steps. Thus, apart from being Simplex-like our algorithm has a stronger property that it has monotone pivoting. We capture this in the following theorem.

THEOREM 2.2. (FINITE BOUND) *For every $l \geq 0$, the path traced by our algorithm starting from S^l in $P(U^{l+1})$ ends at a solution vertex S^{l+1} in a finite number of pivots.*

This gives us the first finite-time complementary-pivoting algorithm for the spending constraint model. Next, we show how to achieve the polynomial bound on the number of pivots in each iteration by extending our algorithm using a scaling based technique. Concretely, the next claim is that when the input utilities are of the form $U_{ijk} = \alpha^{n_{ijk}}$ for the same $\alpha > 1$, and n_{ijk}s are positive integers, the number of pivots needed to reduce U_{ijk} from $\alpha^{n_{ijk}}$ to $\alpha^{n_{ijk}-1}$ is linear.

THEOREM 2.3. (ONE STEP BOUND) *Suppose U is such that $U_{ijk} = \alpha^{n_{ijk}}$ where $\alpha > 1$, and n_{ijk}s are positive integers. Let U' be same as U except for one (i,j,k) where $U'_{ijk} = \alpha^{n_{ijk}-1}$, and let S and S' be the solution vertices of $P(U)$ and $P(U')$. The number of pivots our algorithm performs starting at S and ending at S' is at most $4(m+n)$.*

Using the above theorem, we immediately obtain a bound, on the total number of pivotings done by our algorithm across all polyhedra, of $\text{poly}(N, m, n, s)$, where $N \stackrel{\text{def}}{=} \max_{i,j,k} n_{ijk}$. However, this will not be sufficient. This is because in the case when the utilities are not powers of α, we will need to approximate them to a very high degree of accuracy and then n_{ijk} can be exponential in the bit-lengths involved in the description of the input U. The next idea is to reduce the dependency from N to $\log N$. Note that Theorem 2.3 says something stronger: If the utility we are changing within a factor of α, then the number of pivots needed is independent of α or n_{ijk}. We leverage this by combining our *basic* algorithm (see Figure 2.1) with a scaling technique to get a *scaling* algorithm (see Figure 2.2) whose dependence on N is $\log N$ instead of N in this bound above. Roughly, the idea is to divide the algorithm into phases, each taking $\text{poly}(m, n, s)$ many pivots, and to fix l-th most significant bit of n_{ijk}s in l-th phase, and we get the following theorem.

THEOREM 2.4. (POLYNOMIAL BOUND) *The scaling algorithm (Figure 2.2) finds an equilibrium of a spending constraint model in $O(s(m+n)\log N)$ many pivoting steps, where each U_{ijk} is of the form $\alpha^{n_{ijk}}$ for a fixed $\alpha > 1$, and n_{ijk}s are positive integers.*

Subsequently, we apply this scaling algorithm to traditional utilities, where all U_{ijk}s are integers. To do this, we first approximate each U_{ijk} by $U'_{ijk} = \alpha^{n_{ijk}}$ for a rational $\alpha > 0$ and integers n_{ijk}s such that the configuration of forced, flexible and undesired segments at the solutions of both $P(U)$ and $P(U')$ are the same. We show that such an approximation can be obtained where the size of α and n_{ijk}s are bounded by a polynomial in the number of bits needed to represent (U, B, M). The following theorem captures the result formally.

THEOREM 2.5. (TRADITIONAL UTILITIES) *The number of pivots needed by the scaling algorithm to compute an equilibrium of a spending constraint model is bounded by $\text{poly}(L)$, where L is the size of the input.*

We note that though the sizes of α and n_{ijk}s can be shown to be polynomially bounded individually, if

we want to store U' explicitly we need exponential sized space, which makes every pivoting exponential time. To avoid this we need to store α and n_{ijk}s carefully. We can represent the coordinates of the vertices through polynomials in α. Further, while moving from one vertex to another, we can even represent the direction vector through polynomials. As in the traditional Simplex or any pivoting algorithm, the problem that remains is to find the new tight inequality at the next vertex. This problem reduces to calculating the sign of a polynomial in α with poly(m, n, s) terms and exponential degree, in polynomial time. Formally we need to solve Problem 1.1 in polynomial time, in order to bound the time for each pivoting of the scaling algorithm. Hence, given a polynomial time algorithm for this problem, along with the fact that the number of pivots needed overall is a polynomial, we obtain a polynomial time Simplex-like algorithm for the spending constraint model in the Fisher setting.

Finally, we prove that our scaling algorithm is in fact an FPTAS unconditionally.

THEOREM 2.6. (FPTAS) *The scaling algorithm computes a strongly ε-approximate market equilibrium (see Definition 4.1) in time* poly$(L, 1/\varepsilon)$ *for Fisher markets with spending constraint utilities.*

To prove the above theorem, first, we approximate the input utilities only up to an accuracy of $(1 + \varepsilon/s)$, where s is the total number of segments in the instance. Now all the numbers involved are small and sign testing can be done in polynomial time. However, we have to compare the utilities derived by the buyers from the prices and allocations produced by the scaling algorithm on the approximate utilities. It can be shown that these utilities are at least a $1/1+\varepsilon$ fraction of the ones at the true market equilibrium. The proof is not straightforward and relies on the monotonicity claims of the basic algorithm. This concludes an overview of the algorithm and the steps in the proof of the polynomial bound required for the spending constraint model as in Theorem 1.1. These results can also be shown for the perfect, price-discrimination market model.

3 Perfect, Price-Discrimination Model

In this section we formally define the perfect, price-discrimination market model introduced in [22] and present its LCP-like formulation. First consider the Fisher market model with separable, piecewise-linear concave utilities (SPLC): The utility functions of buyers are separable across goods, for every buyer i and good j the utility function is piecewise-linear concave with s_{ij} segments and on segment $1 \leq k \leq s_{ij}$ buyer i derives utility at a constant rate U_{ijk}. Further, for every segment there is a constraint A_{ijk} that bounds the maximum quantity of the good that can be allocated to segment (i, j, k). Note that this is different than the spending constraint model where the maximum allocation on every segment is restricted by the budget and not by the quantity of good.

In the perfect, price-discrimination model we assume the buyers have SPLC utilities. Buyer i decides a rate r_i at which it wants to derive utility per unit of money it spends. Note that in the Fisher model with SPLC utilities, for a good j, buyer i derives utility at different rates since the price he pays in every segment of j is the same. In contrast, in the perfect, price-discrimination model, the rate is the same not only in every segment of a good, but also across goods. Additionally, there is a middle-man who buys all the goods from sellers at the price they set and sells the goods to the buyers charging according to the utility they derive. However, for any set of prices p_j, the middle-man never sells any good at a loss. This implies that if $r_i < U_{ijk}/p_j$ for a segment (i, j, k), then middle-man profits by selling this segment to buyer i. Similarly, if $r_i = U_{ijk}/p_j$, the middle-man is indifferent in between selling and not selling and if $r_i > U_{ijk}/p_j$, then the middle-man does not sell the segment. Therefore, if a buyer keeps its rate very high, it may not get any good and its budget is unused. As usual, the goal of every buyer is to maximize its utility subject to its budget constraint. Hence, the optimal rate of a buyer is the maximum rate at which it can spend its entire budget. Let $MaxU_i(r_i)$ be the maximum utility buyer i can receive at rate r_i, i.e., $MaxU_i(r_i) = \{\sum_{j,k} U_{ijk}A_{ijk} \mid r_i \leq U_{ijk}/p_j\}$. Buyer i needs to pay $MaxU_i(r_i)/r_i$ in order to buy all these segments. From this, we get the optimal rate $r_i^\star = \max_{r_i}\{MaxU_i(r_i)/r_i \geq M_i\}$ for buyer i; at any rate $r_i > r_i^\star$, it would not able to spend its entire budget and that might give it less utility. At $r_i^\star$, buyer i's utility is $r_i^\star M_i$, and it is indifferent between any bundle that gives it this much utility. At market equilibrium prices, every buyer, based on its optimal rate, gets a utility maximizing bundle and the market clears, i.e., all goods are sold to buyers and all of their money is spent.

We derive an LCP-like formulation for this model using the convex program by [22]. The input to the market is (U, A, M), where (U, A) and M specify the SPLC utility functions and money of the buyers respectively. The variables are $(p_j, q_{ijk}, \lambda_i, \boldsymbol{\gamma}_{ijk})$, where the price of good j is p_j, the amount of money spent by buyer i on the k-th segment of good j is q_{ijk}, the inverse of the rate of buyer i, i.e, $1/r_i$, is λ_i, and the extra price charged by the middle-man in selling good j to buyer i on k-th segment is $\boldsymbol{\gamma}_{ijk}$. The formulation is given in

Algorithm 2.1. (Basic Algorithm)
Input: *Spending constraint:* (U, B, M); *Perfect price discrimination:* (U, A, M)
/* $I = \begin{cases} (B, M) \text{ for spending constraint} \\ (A, M) \text{ for perfect, price-discrimination} \end{cases}$ */
/* $P(U, I) = \begin{cases} \text{Polyhedron associated with LCP 1 for spending constraint} \\ \text{Polyhedron associated with LCP 2 for perfect, price-discrimination} \end{cases}$ */
$U_{max} \leftarrow \max_{i,j,k} U_{ijk}$;
for each (i, j, k)
  $U^0_{ijk} \leftarrow U_{max}$;
$v \leftarrow$ solution vertex of $P(U^0, I)$;
return UpdateU(v, U^0, I, U); (See below)

UpdateU(v, U^c, I, U^f)
for each (i, j, k)
  $U^c_{ijk} \leftarrow U^f_{ijk}$;
  $H \leftarrow$ hyperplane $U^c_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} = 0$;
  if $q_{ijk} > 0$ at v **then**
    $v \leftarrow$ vertex obtained by moving towards H from v in $P(U^c, I)$;
    while $q_{ijk} > 0$ and $U^c_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} < 0$ at v **do**
      $v \leftarrow$ vertex obtained by complementary pivoting at v in $P(U^c, I)$; (See Section 2.3)
return v;

Algorithm 2.2. (Scaling Algorithm)
Input: *Spending constraint:* (U, B, M); *Perfect price discrimination:* (U, A, M)
/* $I = \begin{cases} (B, M) \text{ for spending constraint} \\ (A, M) \text{ for perfect, price-discrimination} \end{cases}$ */
/* $P(U, I) = \begin{cases} \text{Polyhedron associated with LCP 1 for spending constraint} \\ \text{Polyhedron associated with LCP 2 for perfect, price-discrimination} \end{cases}$ */
/* Every $U_{ijk} = \alpha^{n_{ijk}}$, where $\alpha > 1, n_{ijk} \in \mathbb{Z}_+$ */
$n_{max} \leftarrow \max_{i,j,k} n_{ijk}$; $N \leftarrow 2^{\lceil \log n_{max} \rceil}$
$\beta \leftarrow \alpha^N$;
for each (i, j, k)
  $U^c_{ijk} \leftarrow \beta$;
$v \leftarrow$ solution vertex of $P(U^c, I)$;
while $\beta > \alpha$ **do**
  $\beta \leftarrow \sqrt{\beta}$; $U^f \leftarrow U^c$;
  $S \leftarrow \{(i, j, k) \mid U_{ijk} \leq {}^{U^c_{ijk}}/_{\beta}\}$;
  for each $(i, j, k) \in S$
    $U^f_{ijk} \leftarrow {}^{U^c_{ijk}}/_{\beta}$;
  $v \leftarrow$ UpdateU(v, U^c, I, U^f); (See "Basic Algorithm" for UpdateU procedure)
  $U^c \leftarrow U^f$;
return v;

Table 2.

Theorem 3.1. *Any p, q, γ, λ's that satisfy the constraints in Table 2 can be used to produce a market equilibrium for the corresponding perfect, price-discrimination model.*

The basic algorithm in Figure 2.1 and scaling algorithm in Figure 2.2 compute an equilibrium for the perfect, price-discrimination model. There are minor differences

(3.7)	$\forall(i,j,k):$	$\lambda_i \geq 0$	and	$q_{ijk} \geq 0,\ \gamma_{ijk} \geq 0$
(3.8)	$\forall(i,j,k):$	$q_{ijk} \leq A_{ijk}(p_j + \boldsymbol{\gamma}_{ijk})$	and	$U_{ijk}\lambda_i - p_j - \gamma_{ijk} \leq 0$
(3.9)	$\forall(i,j,k):$	$\gamma_{ijk}(q_{ijk} - A_{ijk}(p_j + \boldsymbol{\gamma}_{ijk})) = 0$	and	$q_{ijk}(U_{ijk}\lambda_i - p_j - \gamma_{ijk}) = 0$
(3.10)	$\forall j \in \mathcal{G}:$	$\sum_{i,k} q_{ijk} = p_j + \sum_{i,k} A_{ijk}\boldsymbol{\gamma}_{ijk}$		
(3.11)	$\forall i \in \mathcal{B}:$	$\sum_{j,k} q_{ijk} = M_i$		
	$\forall j \in \mathcal{G}:$	$p_j > 0$		

Table 2: **LCP 2** - An LCP-like formulation for the perfect, price-discrimination market model.

in the proof of the finiteness lemma, for which we refer the reader to the full version.

4 Proof Overviews

4.1 The Finite Bound Theorem Now we present the technique underlying the proof of Theorem 2.2 and point out why such a theorem may not be true for the SPLC case without the budget constraints.[8] Also, it will be helpful to understand the approach in order to prove the Polynomial Bound Theorem.

Consider an iteration when the only utility that changes is for the segment (a, b, c) which is decreased from U_{abc} to U'_{abc}. Let U^l and U^{l+1} be the utilities at beginning and end of the iteration, and let S^l be the solution of $P(U^l)$. The only complementarity constraint that may no longer be satisfied by S^l in $P(U^{l+1})$ will be $(U^{l+1}_{abc}\lambda_a - p_b - \gamma_{abc})q_{abc} = 0$. In case $q_{abc} = 0$ at S^l, then this complementarity condition is satisfied and we can output $S^{l+1} = S^l$. Thus, we may assume that $q_{abc} > 0$ at S^l. Hence, S^l lies on a 1-face of $P(U^{l+1})$. The algorithm now starts to move from S^l. In the first step we have the choice to move in either direction until we hit a vertex of $P(U^{l+1})$. From then on we apply complementary pivoting. The main idea is to show that λ_a, p_a, q_{abc} change monotonically until S^{l+1} is found. Moreover at least one is strictly monotone on every pivot step. Hence, no vertex can appear twice in the path traced from S^l to S^{l+1}. The polyhedron, however, can be unbounded but we can show that this is not a problem: Even if we reach a vertex which has an unbounded ray, we will never take it. Since the number of vertices is finite, this will imply that we must find a solution to $P(U^{l+1})$ in a finite number of pivots.

We proceed to show how to prove monotonicity. There are several cases and one has to deal with degeneracy. We just show it for the first edge in the path and that also only for one case. Further, we assume non-degeneracy which needs to be handled separately. The omitted part of the argument is tedious but similar in flavor.

Given that we violate only the above mentioned complementarity condition, whenever we reach a vertex for which either $q_{abc} = 0$ or $U^{l+1}_{abc}\lambda_a - p_b - \gamma_{abc} = 0$, we have got S^{l+1} by Theorem 2.1. We refer to these two hyperplanes as H_1 and H_2 respectively. From S^l, we will attempt to move so that we come closer to H_2. (In subsequent pivotings, we will not have a choice, but a similar argument can be provided.) Let e_1 and $-e_1$ be the two directions at the point S^l on which we can move. We will pick the one for which the inner product is positive w.r.t. the normal vector to H_2; hence, in this direction, we move closer to H_2. The inner product can be zero in which case it can be shown that we have a degenerate configuration which we have to handle separately. Assume this does not happen. Let e_1 be the direction such that we move to $S^l + \delta e_1$ for a small enough $\delta > 0$ so that we are still in $P(U^{l+1})$. We will denote by λ, p, q, γ values at S^l and by $\lambda(\delta), p(\delta), q(\delta), \gamma(\delta)$ the values of the variables at $S^l + \delta e_1$. Eventually, we would like δ to increase until we hit a vertex of $P(U^{l+1})$. We call this path which starts at S^l and ends for the first time a vertex of $P(U^{l+1})$ is hit as Π, the other end-point not included.

There are two cases to consider: $\gamma_{abc} = 0$ or $\gamma_{abc} > 0$ at S^l. Let us show the idea of the proof for the case $\gamma_{abc} = 0$. In this case, at S^l, $U^{l+1}_{abc}\lambda_a - p_b < 0$, as $q_{abc} > 0$. The main claim is that on Π, $\lambda_a(\delta)$ is non-decreasing while $p_a(\delta)$ and $q_{abc}(\delta)$ are non-increasing and at least one of these changes strictly.

The proof is by contradiction. First note that along the path Π, all variables are monotone. This is because they are linear functions of δ and the increase/decrease is determined by the sign of that co-ordinate in e_1. Next, we try to interpret this movement combinatorially. Consider a bipartite graph between $\mathcal{B}$ and $\mathcal{G}$ with edges only for those $(i, j) \neq (a, b)$ such that there is a k

[8] In fact, we can construct an example where if we apply this pivoting on the LCP for the SPLC case, the algorithm can get stuck.

for which $\gamma_{ijk} = 0$ and $q_{ijk} > 0$ at S^l. This graph will not change on Π; it may change on the vertex we hit. Suppose the (connected) components for this graph are $C_1, \ldots, C_l$. Then, it can be shown that for every $1 \leq r \leq l$, for all $i \in \mathcal{B} \cap C_r$ and $j \in \mathcal{G} \cap C_r$, $\lambda_i(\delta)/\lambda_i = p_j(\delta)/p_j$ is a constant depending only on the component and δ, for all δ on Π. Roughly, this happens because if $q_{ijk} > 0$ and $\gamma_{ijk} = 0$ at S^l, then first $q_{ijk}(\delta) > 0$ and $\gamma_{ijk}(\delta) = 0$, as on the interior of an edge inequalities will not become tight from being non-tight and vice-versa. Hence, from (3.b), $U_{ijk}^{l+1}\lambda_i(\delta) = p_j(\delta)$. Let C_a and C_b be the components that contain a and b respectively. If $C_a = C_b$ a degeneracy can be detected. Hence, we will ignore this case for now. Another thing that can happen is that $C_a = \{a\}$. In this case, because at S^l, $U_{abc}^{l+1}\lambda_a - p_b - \gamma_{abc} < 0$, and a has no edge in the graph, this means that increasing λ_a by a little bit will bring us closer to H_2 and vice-versa without violating any other conditions. Hence, in this case λ_a is strictly monotonically increasing along Π. Similarly if $C_b = \{b\}$, it can be argued that p_b is strictly decreasing along Π. In both cases we have proved what we set out to. The case that remains is when $C_a \neq C_b$ and neither is a singleton.

In this case, if, on the contrary, $\lambda_a(\delta)$ decreases with δ, then $\lambda_a(\delta)/\lambda_a < 1$, which implies that $p_j(\delta)/p_j < 1$ for all $j \in C_a$. This in turn implies that $\sum_{j \in C_a} p_j(\delta) < \sum_{j \in C_a} p_j$. Since money with the buyers of C_a, $\sum_{i \in C_a} M_i$, is fixed, and for any (i, j) such that exactly one of them lies C_a, $q_{ijk} = B_{ijk}$ if $q_{ijk} > 0$ for every k (else that edge would be in the graph), in order to maintain their market clearing conditions, due to the decrease in the prices of the goods in C_a, C_a has to direct its money outwards. It follows from that (2.6), $q_{abc}(\delta) > q_{abc}$. This would imply that $p_b(\delta) > p_b$. Since $\lambda_a(\delta) < \lambda_a$ and $p_b(\delta) > p_b$, the path Π is moving away from H_2, contradicting our choice of movement and proving monotonicity all along. In this case $\lambda_a(\delta), p_b(\delta)$ $q_{abc}(\delta)$ turn out to be strictly monotone.

Since the money of each component is fixed, it is possible to argue that p_js and λ_is of all components other than C_a and C_b remain unchanged. The SPLC utilities are almost like spending constraint utilities except that the segment lengths are based on amount of good and not money- the constraint will look like $q_{ijk} \leq A_{ijk}p_j$ where A_{ijk} is the length of the segment. Thus, changing p_j will change q_{ijk}. This will cause the components to now exchange money through edges which are not present in the graph. This forces the prices in other components to change. This in turn breaks the monotonicity argument.

4.2 The One Step Bound Theorem Using the ideas in the proof of Finite Bound Theorem, we now sketch a proof of Theorem 2.3. In the proof of Theorem 2.2 we argued that in an iteration for segment (a, b, c), λ_a monotonically increases and p_b monotonically decreases. Moreover, for a buyer $i \neq a$, its λ_i increases by the same multiplicative factor as λ_a when $i \in C_a$ and it decreases by the same multiplicative factor as p_b when $i \in C_b$ and remains constant otherwise. Suppose, currently $i \in C_a$ and let $d \stackrel{\text{def}}{=} \lambda_a/\lambda_i$ and $d' \stackrel{\text{def}}{=} \lambda_a/p_b$. At the next vertex suppose i leaves C_a and after some time it again comes back, then clearly for the current values of λs we can say that $\lambda_a/\lambda_i > d$. In other words, the ratio λ_a/λ_i increases between the time when i leaves C_a and joins back. Further, it can be shown that the ratio λ_a/p_b is increasing the fastest, so we get $(\lambda_a/p_b)/d' > (\lambda_a/\lambda_i)/d$.

Next, we note that whenever there is a path between buyers i and $\tilde{i}$ through segments (i', j', k') such that $q_{i'j'k'} > 0$ and $\gamma_{i'j'k'} = 0$ (they are in the same component) we can write $\lambda_i/\lambda_{\tilde{i}}$ as *product of utilities/ product of utilities* using (2.3.a), by eliminating intermediate variables. This is because γ_{ijk}s are zero for all the segments forming the path. In case when U_{ijk}s are of the form $\alpha^{n_{ijk}}$, expression for this ratio $\lambda_i/\lambda_{\tilde{i}}$ evaluates to α^g where g is an integer (positive or negative).

Since $U'_{abc} = U_{abc}/\alpha$ and $\lambda_a/p_b = 1/U_{abc}$ at S, the ratio λ_a/p_b can increases at most by a multiplicative factor α on the path from S to S'. Putting the above two observations together, we can say that whenever buyer i leaves C_a and again joins back, the ratio λ_a/λ_i should have increased at least by a multiplicative factor α. In other words $\lambda_a/\lambda_i \geq \alpha d$. This gives $\lambda_a/p_b > \alpha d'$ using $(\lambda_a/p_b)/d' > (\lambda_a/\lambda_i)/d$, a contradiction. This implies that once i leaves C_a it can not join it again. A similar argument can be derived for C_b. Since, in every iteration some i joins/leaves either C_a or C_b, the number of pivotings between S and S' is bounded by $4(m + n)$.

4.3 The Polynomial Bound Theorem Using the power of Theorem 2.3 and the scaling technique, next we sketch a proof of Theorem 2.4. Recall that N is defined as $\max_{i,j,k} n_{ijk}$, and s as total number of segments in the instance. Let $\beta \stackrel{\text{def}}{=} \alpha^{2^{\lceil \log N \rceil}}$. The scaling algorithm starts with U^c where all the utilities are set to β. In i-th iteration it fixes i-th most significant bit of all the n_{ijk}s. In this algorithm (Figure 2.2), we reset β to $\sqrt{\beta}$ in every iteration. Suppose at the start of an iteration, U^c_{ijk} is β^l, where $l \in \mathbb{Z}_+$. Note that when β is reset with $\sqrt{\beta}$, the U^c_{ijk} becomes β^{2l}. Therefore, in each iteration U^c_{ijk}s remain integer powers of the current β.

In an iteration, utilities U^f are constructed such that U^c_{ijk}/U^f_{ijk} is either 1 or β for the current value of

β. Theorem 2.3 implies that the UpdateU procedure, while called from the scaling algorithm with starting and final utilities being U^c and U^f, takes $O(s(m+n))$ pivots, as we need to fix at most s segments in one call. It is clear that the scaling algorithm makes $O(\log N)$ calls to UpdateU procedure. Therefore, we can conclude that the scaling algorithm of Figure 2.2 finds an equilibrium of a Fisher market with spending constraint utilities in $O(s(m+n)\log N)$ many pivots. The bound follows trivially for linear Fisher markets, and may be worked out similarly for markets with perfect, price discrimination.

4.4 The Traditional Utilities Theorem In this section we sketch a proof of Theorem 2.5. Let us assume that all utilities U_{ijk}, money M_i and segment lengths B_{ijk} are integers, since scaling them by a positive number does not change equilibria and the input size only increases by a polynomial factor. Let $\Delta \stackrel{\text{def}}{=} hU_{max}^h$ and $M \stackrel{\text{def}}{=} \sum_{i\in\mathcal{B}} M_i$, where $h \stackrel{\text{def}}{=} \max\{m,n\}$, U_{max} is the maximum among tensor U.

Let $\alpha = (1 + 1/(m+n)\Delta M)$ and U' be the new utility matrix such that $U'_{ijk} = \alpha^{n_{ijk}}$ where $U'_{ijk}/\alpha < U_{ijk} \leq U'_{ijk}$. It is easy to show that the values of n_{ijk}s are bounded above by $2(m+n)\Delta M \log U_{ijk}$. Therefore, the number of pivots taken by the scaling algorithm on input (U', B, M) is bounded by $\text{poly}(m, n, L)$, where L is the size of the input (U, B, M). Note that $U' \leq U$. It may be shown that the number of pivots taken by $UpdateU(v, U', I, U)$ (Figure 2.1) is at most $(m+n)$, where v is a solution vertex of $P(U')$. This concludes that an equilibrium of a Fisher market with spending constraint utilities may be obtained in polynomially many pivots.

The Algebraic Problem. Note that the size of n_{ijk}s, and the numerator and denominator of α are polynomial in the input size, where the input is in binary form. Even if value of $\alpha^{n_{ijk}}$ is very near to U_{ijk}, to store it exactly we need $O(N \log \alpha)$ space, where $N = \max_{ijk} n_{ijk}$, which is exponential in the input size. Therefore, in order to do every step in $\text{poly}(m, n, \log N)$ time we need to store α and n_{ijk}s separately. In that case the problem of computing the new tight inequality at the next vertex in every pivoting step reduces to checking the sign of a polynomial, say $f(\alpha)$, with degree at most $(m+n)N$ and $m+n$ terms. The coefficients of the polynomial can be computed efficiently. Further, for our case of α it is easy to show that $f(\alpha)$ evaluates to zero if and only if all its coefficients are zero, *i.e.,* it is identically zero. The question is how to check if $f(\alpha)$ evaluates to positive or negative without evaluating it explicitly (formally stated in Problem 1.1).

4.5 Fully Polynomial Time Approximation Scheme Using the monotonicity properties shown for Theorem 2.2, we prove Theorem 2.6. We show that the scaling algorithm can compute strongly ε-approximate equilibrium for the Fisher market with spending constraints where the number of required pivots remains polynomial and, in addition, each pivot can be performed in time polynomial in the input size and $1/\varepsilon$. This result does not depend upon the resolution of Problem 1.1. Note that the market models we consider have unique equilibrium prices and utilities [7, 19, 22].

Definition 4.1. *(*Strongly Approximate Equilibrium*) For an $\varepsilon > 0$, we say that (x, p) is a strongly ε-approximate market equilibrium if $U_i(x_i) \geq U_i^\star/(1+\varepsilon)$, $\forall i \in \mathcal{B}$ and $p_j^\star/(1+\varepsilon) \leq p_j \leq p_j^\star(1+\varepsilon)$, $\forall j \in \mathcal{G}$, where $U^\star$ and $p^\star$ are respectively the (unique) equilibrium utilities and prices.*

The above is the strongest possible notion of approximate equilibrium; weaker notions have also been studied, for e.g., [24, 12]. Let L be the bit length of input.

First we approximate every U_{ijk} by $(1+\varepsilon)^{n_{ijk}}$ such that $(1+\varepsilon)^{n_{ijk}} \leq U_{ijk} < (1+\varepsilon)^{n_{ijk}+1}$, where $n_{ijk} \in \mathbb{Z}_+$. Let U' denotes the new utilities. Clearly, $U_{ijk}/(1+\varepsilon) < U'_{ijk} \leq U_{ijk}$. Next, we apply the scaling algorithm given in Figure 2.2 on U' and get the equilibrium solution (x', p'). From the discussion in Section 4.3 it is clear that the number of pivots are bounded by $\text{poly}(L)$. Since every $n_{ijk} \leq \log U_{max}/\log(1+\varepsilon) \leq \log U_{max}/\varepsilon$, the size of every number $(1+\varepsilon)^{n_{ijk}}$ is at most $O(1/\varepsilon \log U_{max} \log(1/\varepsilon))$. Hence we can store U' explicitly using space of size $\text{poly}(L, 1/\varepsilon)$, and every pivoting can be done in time $\text{poly}(L, 1/\varepsilon)$, thereby eliminating the necessity to handle Problem 1.1.

Next, we use the basic algorithm (Figure 2.1) to show that buyer i derives at least $U_i^\star/(1+\varepsilon)^s$ utility from allocation x', where $U_i^\star$ is its utility at the equilibrium of the original market and s is the total number of segments in U. Suppose the basic algorithm obtains a solution for U' starting with the solution of U. There are s iterations, one for fixing every segment. Let U_i^k be the utility of buyer i at the beginning of k-th iteration. Note that $U_i^1 = U_i^\star$. The following claim is key to our FPTAS result.

Claim 4.1. $U_i^{k+1} \geq U_i^k/(1+\varepsilon), 1 \leq k \leq s, \forall i.$

Since there are s iterations in the basic algorithm, the above claim implies that the utility of buyer i with respect to U' is at least $U_i^\star/(1+\varepsilon)^s$. By a similar analysis, we can show that equilibrium prices p' for input U' are such that $p_j^\star/(1+\varepsilon)^s \leq p'_j \leq p_j^\star(1+\varepsilon)^s, \forall j$, where $p^\star$ are the equilibrium prices for input U.

Given $\varepsilon > 0$, by choosing $\delta \stackrel{\text{def}}{=} \varepsilon/s$ and approximating U by $(1+\delta)^{n_{ijk}}$s, we get the ε-approximate solution in time poly$(L, 1/\varepsilon)$.

5 Comparison with Related Work

As mentioned in the introduction, the problem of computing market equilibria is an intensely studied problem. Rather than providing a comprehensive survey of the literature, we choose to compare our techniques with the ones we think are the most relevant. First we would like to clarify that, although we use convex programming techniques to derive our LCP-like formulations, we do not employ any method to solve convex programs. Our algorithm performs complementary pivoting on a sequence of carefully chosen polyhedra, where the polyhedra are obtained from dropping complementarity conditions from the LCPs. First, note that while LCP formulations were known for the SPLC model [21], prior to this work, there were no known LCP-like formulations for the spending constraint and the perfect, price-discrimination models. Our algorithmic technique is inspired by the algorithms of Lemke [26] and Lemke and Howson [27]. Previous such attempts to obtain complementary-pivoting or Simplex-like algorithms for linear markets were made by [18, 2]. As outlined before, while these algorithms pivoted on a single polyhedron, ours employs a scaling based technique and pivots on a sequence of connected polyhedra, each corresponding to a different set of utilities. Moreover, while ours does polynomially many pivots, no such bound is known for the other two. Also, our algorithm works for more general market models than linear.

Polynomial time algorithms for the spending constraint model and the perfect, price discrimination model were known due to [37] and [22] respectively. These algorithms built on the combinatorial techniques of [17]. These algorithms are primal-dual, flow-based and compute equilibrium by iteratively increasing prices.

Strongly polynomial time algorithms are also known, for instance for linear Fisher markets [30] used the scaling technique from [29] in a flow network. [40] considered a class of minimum-cost flow problems, where the cost function is convex and separable and he gave a strongly polynomial algorithm for these problems using a scaling based technique of [29]. As an application, he obtained a strongly polynomial time algorithms for the Fisher market model with spending constraint. In another paper, [39] considered the concave generalized flow problem, where the flow leaving an arc is an increasing concave function of the flow entering it, and he gave a polynomial time combinatorial algorithm for these problems. Since the convex programming formulation for perfect, price-discrimination model [22] can be formulated as such a problem, one obtains a polynomial time algorithm for this model.

There is also a large body of work on algorithms for computing market equilibrium via local-dynamics. Walras [41] was the first to consider such a dynamics in 1874, which he called tatonnement. Here the prices are changed based on the *excess demand* and equilibrium is attained when there is no excess demand. Later, it was shown by [5, 3, 6, 35] that the continuous tatonnement process converges locally for the markets satisfying weak gross substitutability (WGS)[9]. Recently, [11, 12] gave a discrete tatonnement process that converges to an approximate equilibrium for WGS markets. The running time of our FPTAS, which computes an approximate equilibrium for spending constraint and perfect, price-discrimination markets, can be compared to the running time of the result of [11]. While both these markets satisfy WGS condition, however, it is unclear how to obtain our FPTAS results from [11] as it is assumed in this paper that the demand is a single valued function of the prices which is not the case with these two markets.

References

[1] B. Adsul, C. S. Babu, J. Garg, R. Mehta, and M. Sohoni. Nash equilibria in Fisher market. In *Symposium on Algorithmic Game Theory*, pages 30–41, 2010.

[2] B. Adsul, C. S. Babu, J. Garg, R. Mehta, and M. Sohoni. A simplex-like algorithm for Fisher markets. In *Symposium on Algorithmic Game Theory*, pages 18–29, 2010.

[3] K. Arrow, H. Block, and L. Hurwicz. On the stability of the competitive equilibrium II. *Econometrica*, 27(1):82–109, 1959.

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

[5] K. Arrow and L. Hurwicz. On the stability of the competitive equilibrium. *Econometrica*, 26(4):522–552, 1958.

[6] K. Arrow and L. Hurwicz. Competitive stability and weak gross substitutability: the Euclidean distance approach. *International Economic Review*, 1(1):38–49, 1960.

[7] B. Birnbaum, N. R. Devanur, and L. Xiao. Distributed algorithms via gradient descent for Fisher markets. In *ACM Electronic Commerce*, pages 127–136, 2011.

[8] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[9] X. Chen, D. Dai, Y. Du, and S.-H. Teng. Settling the complexity of Arrow-Debreu equilibria in markets with

[9] An increase in the price of a good does not decrease the demand of other goods, whose prices are kept fixed.

additively separable utilities. In *IEEE Annual Symposium on Foundations of Computer Science*, pages 273–282, 2009.

[10] X. Chen and S.-H. Teng. Spending is not easier than trading: on the computational equivalence of Fisher and Arrow-Debreu equilibria. In *ISAAC: International Symposium on Algorithms and Complexity*, pages 647–656, 2009.

[11] B. Codenotti, B. McCune, and K. Varadarajan. Market equilibrium via the excess demand function. In *ACM Symposium on the Theory of Computing*, pages 74–83, 2005.

[12] B. Codenotti, S. Pemmaraju, and K. Varadarajan. On the polynomial time computation of equilibria for certain exchange economies. In *ACM-SIAM Annual Symposium on Discrete Algorithms*, pages 72–81, 2005.

[13] B. Codenotti, A. Saberi, K. Varadarajan, and Y. Ye. Leontief economies encode two-player zero-sum games. In *ACM-SIAM Annual Symposium on Discrete Algorithms*, 2006.

[14] R. Cole and L. Fleischer. Fast-converging tatonnement algorithms for one-time and ongoing market problems. In *STOC*, pages 315–324, 2008.

[15] F. Cucker, P. Koiran, and S. Smale. A polynomial-time algorithm for diophantine equations in one variable. *Journal of Symbolic Computation*, 27:21–29, 1999.

[16] N. R. Devanur. Fisher markets and convex programs. Working paper, `http://research.microsoft.com/en-us/um/people/nikdev/pubs/convex.pdf`.

[17] N. R. Devanur, C. H. Papadimitriou, A. Saberi, and V. V. Vazirani. Market equilibrium via a primal-dual algorithm for a convex program. *JACM*, 55(5), 2008.

[18] B. C. Eaves. A finite algorithm for the linear exchange model. *Journal of Mathematical Economics*, 3:197–203, 1976.

[19] E. Eisenberg and D. Gale. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics*, 30:165–168, 1959.

[20] O. Friedmann, T. D. Hansen, and U. Zwick. Subexponential lower bounds for randomized pivoting rules for the simplex algorithm. In *ACM Symposium on the Theory of Computing*, pages 283–292, 2011.

[21] J. Garg, R. Mehta, M. Sohoni, and V. V. Vazirani. A complementary pivot algorithm for market equilibrium under separable piecewise-linear concave utilities. In *ACM Symposium on the Theory of Computing*, 2012.

[22] G. Goel and V. V. Vazirani. A perfect price discrimination market model with production and a rational convex program for it. *Mathematics of Operations Research*, 36:762–782, 2011.

[23] K. Jain. A polynomial time algorithm for computing the Arrow-Debreu market equilibrium for linear utilities. *SIAM Journal on Computing*, 37(1):306–318, 2007.

[24] K. Jain, M. Mahdian, and A. Saberi. Approximating market equilibrium. In *Proceedings of 6th APPROX*, 2003.

[25] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with non-negative entries. *JACM*, 51(4):671–697, 2004.

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

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

[28] E. I. Nenakov and M. E. Primak. One algorithm for finding solutions of the Arrow-Debreu model. *Kibernetica*, 3:127–128, 1983.

[29] J. B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations Research*, 41(2):338–350, 1993.

[30] J. B. Orlin. Improved algorithms for computing Fisher's market clearing prices. In *ACM Symposium on the Theory of Computing*, pages 291–300, 2010.

[31] H. E. Scarf. The approximation of fixed points of a continuous mapping. *SIAM Journal on Applied Mathematics*, 15:1328–1343, 1967.

[32] V. I. Shmyrev. An algorithm for finding equilibrium in the linear exchange model with fixed budgets. *Journal of Applied and Industrial Mathematics*, 3(4):505–518, 2009.

[33] S. Smale. A convergent process of price adjustment and global Newton methods. *Journal of Mathematical Economics*, 3(2):107–120, 1976.

[34] D. A. Spielman and S.-H. Teng. Smoothed analysis: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004.

[35] H. Uzawa. Walras' tatonnement in the theory of exchange. *The Review of Economic Studies*, 27(3):182–194, 1960.

[36] V. V. Vazirani. Combinatorial algorithms for market equilibria. *Chapter 5, Algorithmic Game Theory, eds. N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani*, pages 53–78, 2007.

[37] V. V. Vazirani. Spending constraint utilities, with applications to the Adwords market. *Mathematics of Operations Research*, 35(2), 2010.

[38] V. V. Vazirani. The notion of a rational convex program, and an algorithm for the Arrow-Debreu Nash bargaining game. *Journal of ACM*, 59(2), 2012.

[39] L. A. Vegh. Concave generalized flows with applications to market equilibria. In *FOCS*, 2012.

[40] L. A. Vegh. Strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. In *STOC*, 2012.

[41] L. Walras. *Éléments d'économie politique pure ou théorie de la richesse sociale (Elements of Pure Economics, or the theory of social wealth)*. Lausanne, Paris, 1874. (1899, 4th ed.; 1926, rev ed., 1954, Engl. transl.).

[42] L. Zhang. Proportional response dynamics in the Fisher market. *Theor. Comput. Sci.*, 412(24):2691–2698, 2011.

A LCPs from Convex Programs

In this section we show how to derive the LCP for the spending constraint model starting from a convex programming formulation for it. The technique we present here is general enough to allow us to obtain an LCP for the perfect, price-discrimination model. Since linear Fisher markets arise as a special case of the spending constraint model, the LCP for them becomes simpler.

A.1 LCP formulation for Fisher markets with spending constraints Recall that the input to the spending constraint model is (U, B, M). Without loss of generality (w.l.o.g.), we can assume that for every good j, there is a segment (i, j, k) such that $U_{ijk} > 0$, otherwise no buyer is interested in buying this good and we can safely discard this good and set $p_j = 0$. Similarly, we assume that $M_i > 0$ for every buyer i, otherwise we can discard this buyer from the market. W.l.o.g., we also assume that the total quantity of every good is unit, otherwise we can get an equivalent market by scaling U appropriately. The following convex program by [7] is a generalization of [32, 16] and they proved that it captures the equilibrium prices and allocations for the spending constraint model. The variables are q_{ijk}s and p_js, where q_{ijk} denotes the amount of money spent by buyer i on segment (j, k) and p_j denotes the price of good j.

$$\text{maximize} \quad \sum_{i,j,k} q_{ijk} \log U_{ijk} - \sum_j p_j \log p_j$$

$$\text{(A.1)} \quad \forall i: \quad \sum_{j,k} q_{ijk} = M_i$$

$$\text{(A.2)} \quad \forall j: \quad \sum_{i,k} q_{ijk} = p_j$$

$$\text{(A.3)} \quad \forall (i,j,k): \quad q_{ijk} \geq 0$$

$$\text{(A.4)} \quad \forall (i,j,k): \quad q_{ijk} \leq B_{ijk}$$

Theorem A.1. *[7] An optimal solution of above convex program corresponds to an equilibrium for a Fisher market with spending constraints whose input is (U, B, M).*

Next, we write the KKT conditions for the above program, which are necessary conditions for the optimality [8]. Let $\alpha_i, \beta_j, \mu_{ijk}, \delta_{ijk}$ be the Lagrangian (dual) variables corresponding to equations (A.1)-(A.4). An optimal solution must satisfy the following KKT conditions:

$$\text{(A.5)} \quad \forall (i,j,k): \quad \log U_{ijk} = \alpha_i + \beta_j - \mu_{ijk} + \delta_{ijk}$$

$$\text{(A.6)} \quad \forall j: \quad 1 + \log p_j = \beta_j$$

$$\text{(A.7)} \quad \forall (i,j,k): \quad q_{ijk}\mu_{ijk} = 0$$

$$\text{(A.8)} \quad \forall (i,j,k): \quad (q_{ijk} - B_{ijk})\delta_{ijk} = 0$$

$$\text{(A.9)} \quad \forall (i,j,k): \quad \mu_{ijk} \geq 0, \delta_{ijk} \geq 0$$

We first show that at equilibrium, prices of all goods are strictly positive.

Lemma A.1. *At an optimal solution of above convex program, $p_j > 0, \forall j$.*

Proof. This proof is by contradiction. Suppose $p_j = 0$ for some j at an optimal solution. Since the optimal solution satisfies (A.1)-(A.9), we get that $q_{ijk} = 0, \forall (i,k)$ (from (A.2)). This gives us $\delta_{ijk} = 0, \forall (i,k)$ (from (A.8)). As per our assumption, there is a segment for good j, such that $U_{ijk} > 0$. For such a segment (i, j, k), putting $p_j = 0$ and $\delta_{ijk} = 0$ in (A.5)-(A.6), we get $\alpha_i = \infty$.

From (A.1), there is at least one segment (i, j', k') for buyer i, such that $q_{ij'k'} > 0$. For this segment, $\mu_{ij'k'} = 0$ (from (A.7)) and $p_{j'} > 0$ (from (A.3)). We get the contradiction by putting $\alpha_i = \infty$ and $\mu_{ij'k'} = 0$ in (A.5)-(A.6).

Hence, simplifying (A.5)-(A.6), we get

$$\text{(A.10)} \quad \forall (i,j,k): \quad \frac{U_{ijk}}{p_j} = e^{1+\alpha_i-\mu_{ijk}+\delta_{ijk}}$$

Replacing $e^{-(1+\alpha_i)}$ by λ_i, we get $\lambda_i \geq 0$ and the above equation becomes

$$\text{(A.11)} \quad \forall (i,j,k): \quad \frac{U_{ijk}}{p_j} = \frac{1}{\lambda_i} e^{-\mu_{ijk}+\delta_{ijk}}$$

Next, replacing $p_j e^{\delta_{ijk}}$ by $p_j + \boldsymbol{\gamma}_{ijk}$ and using $\delta_{ijk} \geq 0$, we get $\boldsymbol{\gamma}_{ijk} \geq 0$ and

$$\text{(A.12)} \quad \forall (i,j,k): \quad \frac{U_{ijk}}{p_j + \boldsymbol{\gamma}_{ijk}} = \frac{1}{\lambda_i} e^{-\mu_{ijk}}$$

Using (A.7), we get that (A.12) is equivalent to

$$\text{(A.13)} \; \forall (i,j,k): U_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} \leq 0; \quad q_{ijk} \geq 0; \quad q_{ijk}(U_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk}) = 0$$

Further, (A.8) is equivalent to

$$\text{(A.14)} \quad \forall (i,j,k): \quad q_{ijk} \leq B_{ijk}; \quad \boldsymbol{\gamma}_{ijk} \geq 0; \quad \boldsymbol{\gamma}_{ijk}(q_{ijk} - B_{ijk}) = 0$$

$$\text{(A.15)} \quad \forall j: \quad p_j > 0$$

Note that we need (A.15), because we want $\boldsymbol{\gamma}_{ijk} = 0$ iff $\delta_{ijk} = 0$, and that can be assumed due to Lemma A.1. Finally, we get that (A.1)-(A.9) are equivalent to the following LCP-like formulation.

$$
\begin{array}{rl}
\forall (i,j,k): & U_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} \leq 0; \quad q_{ijk} \geq 0; \\
& \qquad q_{ijk}(U_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk}) = 0 \\
\forall (i,j,k): & q_{ijk} \leq B_{ijk}; \quad \boldsymbol{\gamma}_{ijk} \geq 0; \\
& \qquad \boldsymbol{\gamma}_{ijk}(q_{ijk} - B_{ijk}) = 0 \\
\forall i: & \sum_{j,k} q_{ijk} = M_i \\
\forall j: & \sum_{i,k} q_{ijk} = p_j \\
\forall i: & \lambda_i \geq 0
\end{array}
\tag{A.16}
$$

Note that we have not put (A.15) in (A.16), because it is redundant as shown by the following lemma.

LEMMA A.2. *Every solution of* (A.16) *has* $p_j > 0, \forall j$.

Proof. The proof is by contradiction. Suppose there is a solution of (A.16) with $p_j = 0$ for some j. It implies that $q_{ijk} = 0, \forall (i,k)$ (from $\sum_{i,k} q_{ijk} = p_j$) and $\boldsymbol{\gamma}_{ijk} = 0, \forall (i,k)$ (from $\boldsymbol{\gamma}_{ijk}(q_{ijk} - B_{ijk}) = 0$). Consider a segment (i,j,k) for good j, such that $U_{ijk} > 0$. This implies that $\lambda_i = 0$ (from $U_{ijk}\lambda_i - p_j - \boldsymbol{\gamma}_{ijk} \leq 0$). There must exist some (i,j',k'), such that $q_{ij'k'} > 0$ (from $\sum_{j,k} q_{ijk} = M_i$) and $p_{j'} > 0$ (from $\sum_{i,k} q_{ij'k} = p_{j'}$), which is a contradiction to $q_{ij'k'}(U_{ij'k'}\lambda_i - p_{j'} - \boldsymbol{\gamma}_{ij'k'}) = 0$.

We call the formulation (A.16) LCP-like and not an LCP due to the following technicality: There are equalities in our formulation and all inequalities do not participate in the complementarity conditions. We ignore this distinction. Next, we prove the main theorem of this section, which is the same as Theorem 2.1.

THEOREM A.2. (LCP FOR SPENDING CONSTRAINT) *The p_js and q_{ijk}s give respectively market equilibrium prices and allocations for an instance of the spending constraint model if and only if they are solutions of* (A.16).

Proof. Recall that (A.16) is equivalent to (A.1)-(A.9). From Theorem A.1 and the fact that the above convex program satisfies strong duality [8], we get that every solution of (A.16) is a market equilibrium solution.

For the other direction, given a market equilibrium solution p_js and q_{ijk}s, we set $1/\lambda_i \stackrel{\text{def}}{=} \min_{i,j,k}\{U_{ijk}/p_j \mid q_{ijk} > 0\}$, $\forall i$. Further if $q_{ijk} < B_{ijk}$, then we set $\boldsymbol{\gamma}_{ijk} = 0$, otherwise $\boldsymbol{\gamma}_{ijk} = U_{ijk}\lambda_i - p_j$, $\forall (i,j,k)$. It is easy to check that these satisfy (A.16).