A Strongly Polynomial Algorithm for Linear Exchange Markets

Jugal Garg*
University of Illinois at Urbana-Champaign
Urbana, IL, USA
jugal@illinois.edu

László A. Végh†
London School of Economics and Political Science
London, UK
l.vegh@lse.ac.uk

ABSTRACT

We present a strongly polynomial algorithm for computing an equilibrium in Arrow-Debreu exchange markets with linear utilities. Our algorithm is based on a variant of the weakly-polynomial Duan–Mehlhorn (DM) algorithm. We use the DM algorithm as a subroutine to identify revealed edges, i.e. pairs of agents and goods that must correspond to best bang-per-buck transactions in every equilibrium solution. Every time a new revealed edge is found, we use another subroutine that decides if there is an optimal solution using the current set of revealed edges, or if none exists, finds the solution that approximately minimizes the violation of the demand and supply constraints. This task can be reduced to solving a linear program (LP). Even though we are unable to solve this LP in strongly polynomial time, we show that it can be approximated by a simpler LP with two variables per inequality that is solvable in strongly polynomial time.

CCS CONCEPTS

• **Theory of computation** → **Market equilibria**; *Linear programming*.

KEYWORDS

Market Equilibria, Linear Exchange Markets, Strongly Polynomial Algorithm, Z_+-Matrix

ACM Reference Format:
Jugal Garg and László A. Végh. 2019. A Strongly Polynomial Algorithm for Linear Exchange Markets. In *Proceedings of the 51st Annual ACM SIGACT Symposium on the Theory of Computing (STOC '19), June 23–26, 2019, Phoenix, AZ, USA.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3313276.3316340

1 INTRODUCTION

The exchange market model has been introduced by Walras in 1874 [42]. In this model, a set of agents arrive at a market with an initial endowment of divisible goods and have a utility function over allocations of goods. Agents can use their revenue from selling their initial endowment to purchase their preferred bundle of goods. In a market equilibrium, the prices are such that each agent can spend their entire revenue on a bundle of goods that maximizes her utility at the given prices, and all goods are fully sold.

The celebrated result of Arrow and Debreu [2] shows the existence of an equilibrium for a broad class of utility functions. Computational aspects have been already addressed since the 19th century, see e.g. [3], and polynomial time algorithms have been investigated in the theoretical computer science community over the last twenty years; see the survey [4] for early work, and the references in [18] for more recent developments.

In this paper we study the case where all utility functions are *linear*. Linear market models have been extensively studied since 1950s; see [9] for an overview of earlier work. These models are also appealing from a combinatorial optimization perspective due to their connection to classical network flow models and their rich combinatorial structure. A well-studied special case of the exchange market is the *Fisher market* setting, where every buyer arrives with a fixed budget instead of an endowment of goods. Using network flow techniques, Devanur et al. [10] gave a polynomial-time combinatorial algorithm that was followed by a series of further such algorithms [21, 37], including strongly polynomial ones [32, 40]. A combinatorial algorithm for the general exchange market was developed much later by Duan and Mehlhorn [13], and no strongly polynomial algorithm has been known thus far.

Strongly polynomial algorithms and rational convex programs. Assume that a problem is given by an input of N rational numbers given in binary description. An algorithm for such a problem is *strongly polynomial* (see [23, Section 1.3]), if it only uses elementary arithmetic operations (addition, comparison, multiplication, and division), and the total number of such operations is bounded by poly(N). Further, the algorithm is required to run in *polynomial space*: that is, the size of the numbers occurring throughout the algorithm remain polynomially bounded in the size of the input. Here, the size of a rational number p/q with integers p and q is defined as $\lceil \log_2(|p|+1) \rceil + \lceil \log_2(|q|+1) \rceil$.

It is a major open question to find a strongly polynomial algorithm for linear programming. Such algorithms are known for special classes of linear optimization problems. We do not present a comprehensive overview here but only highlight some examples: systems of linear equations with at most two nonzero entries per inequality [1, 5, 28]; minimum cost circulations e.g. [22, 31, 34]; LPs with bounded entries in the constraint matrix [35, 36]; generalized flow maximization [30, 41], and variants of Markov Decision Processes [43, 45].

For nonlinear convex optimization, only sporadic results are known. The relevance of certain market equilibrium problems in

*Supported by the NSF CRII Award 1755619.
†Supported by the ERC Starting Grant ScaleOpt.

STOC '19, June 23–26, 2019, Phoenix, AZ, USA

ACM ISBN 978-1-4503-6705-9/19/06…$15.00
https://doi.org/10.1145/3313276.3316340

this context is that they can be described by *rational convex programs*, where a rational optimal solution exists with encoding size bounded in the input size (see [38]). This property gives hope for finding strongly polynomial algorithms.

The linear Fisher market equilibrium can be captured by two different convex programs, one by Eisenberg and Gale [16], and one by Shmyrev [33]. These are special cases of natural convex extensions of classical network flow models [39, 40]. In particular, the second model is a network flow problem with a separable convex cost function; [40] provides a strongly polynomial algorithm for the linear Fisher market using this general perspective.

The exchange market model is not known to be described by such simple convex programs. A rational convex program was given in [9], but the objective is not separable and hence the result in [40] cannot be applied. Previous convex programs [6, 24, 29] included nonlinear constraints and did not appear amenable for a combinatorial approach (see [9] for an overview).

Model. Let A be the set of n agents. Without loss of generality, we can assume that there is one unit in total of each divisible good, and that there is a one-to-one correspondence between agents and goods: agent i brings the entire unit of the i-th good, g_i to the market. If agent i buys x_{ij} units of good g_j, her utility is $\sum_j u_{ij}x_{ij}$, where u_{ij} is the utility of agent i for a unit amount of good g_j. Given prices $p = (p_i)_{i\in A}$, the bundle that maximizes the utility of agent i is any choice of *maximum bang-per-buck goods*, that is, goods that maximize the ratio u_{ij}/p_j. The prices p and allocations $(x_{ij})_{i,j\in A}$ form a market equilibrium, if *(i)* $\sum_{i\in A} x_{ij} = 1$ for all j, that is, every good is fully sold; *(ii)* $p_i = \sum_{j\in A} p_j x_{ij}$ for all i, that is, every agent spends her entire revenue; and *(iii)* $x_{ij} > 0$ implies that $u_{ij}/p_j = \max_k u_{ik}/p_k$, that is, all purchases maximize bang-per-buck.

Algorithms for the linear exchange market. A finite time algorithm based on Lemke's scheme [27] was obtained by Eaves [14]. A necessary and sufficient condition for the existence of equilibrium was described by Gale [17]. The first polynomial-time algorithms for the problem were given by Jain [24] using the ellipsoid method and by Ye [44] using an interior point method. The combinatorial algorithm of Duan and Mehlhorn [13] builds on the algorithm [10] for linear Fisher markets. An improved variant was given in [12] with the current best weakly polynomial running time $O(n^7 \log^3(nU))$, assuming all u_{ij} values are integers between 0 and U. A main measure of progress in these algorithms is the increase in prices. However, the upper bound on the prices depends on the u_{ij} values; therefore, such an analysis can only provide a weakly polynomial bound. The existence of a strongly polynomial algorithm is described in [13] as a major open question. There is a number of simple algorithms for computing an approximate equilibrium [11, 19, 20, 25], but these do not give rise to polynomial-time exact algorithms.

Our result. We provide a strongly polynomial algorithm for linear exchange markets with running time $O(n^{10}\log^2 n)$. Let us give an overview of the main ideas and techniques. Let F^* denote the set of edges (pairs of agents and goods) that correspond to a best bang-per-buck transaction in every equilibrium. In the algorithm, we maintain a set $F \subseteq F^*$ called *revealed edges*, and the main progress is adding a new edge in strongly polynomial time. At a high level, this approach resembles that of [40], which extends Orlin's approach for minimum-cost circulations [31].

In a money allocation (p, f), $(p_i)_{i\in A}$ is a set of prices and $(f_{ij})_{i,j\in A}$ represent the amount of money paid for good g_j by agent i; f_{ij} may only be positive for maximum bang-per-buck pairs. In the algorithm we work with a money allocation where goods may not be fully sold and agents may have leftover money; we let $\|s(p,f)\|_1$ denote the total surplus money left, and $\|s(p,f)\|_\infty$ the maximum surplus of any good. It can be shown that if $f_{ij} > \|s(p,f)\|_1$ in a money allocation, then $(i, g_j) \in F^*$. This is analogous to the method of identifying *abundant arcs* for minimum cost flows by Orlin [31].

We use a variant of the Duan-Mehlhorn (DM) algorithm to identify abundant arcs. We show that $\phi(p,f) = \|s(p,f)\|_\infty/(\prod_j p_j)^{1/n}$ decreases geometrically in the algorithm; from this fact, it is not difficult to see that an arc with $f_{ij} > \|s(p,f)\|_1$ appears in a strongly polynomial number of iterations, yielding the first revealed arc. We need to modify the DM algorithm [13] so that, among other reasons, the potential decreases geometrically when we run the algorithm starting with any arbitrary price vector p; see Remark 4.1 for all the differences.

To identify subsequent revealed arcs, we need a more flexible framework and a second subroutine. We work with the more general concept of F-allocations, where the money amount f_{ij} could be negative if $(i, g_j) \in F$. This is a viable relaxation since an F-equilibrium (namely, a market equilibrium with possibly negative allocations in F) can be efficiently turned into a proper market equilibrium, provided that $F \subseteq F^*$. Given a set F of revealed arcs, our *Price Boost* subroutine finds an approximately optimal F-allocation using only edges in F. Namely, the subroutine finds an F-equilibrium if there exists one; otherwise, it finds an F-allocation that is zero outside F, and subject to this, it approximately minimizes $\phi(p,f)$. This will provide the initial prices for the next iteration of the DM subroutine. Since DM decreases $\phi(p,f)$ geometrically, after a strongly polynomial number iterations it will need to send a substantial amount of flow on an edge outside F, providing the next revealed edge.

Let us now discuss the Price Boost subroutine. The analogous subproblem for Fisher markets in [40] reduces to a simple variant of the Floyd-Warshall algorithm. For exchange markets, we show that optimizing $\phi(p,f)$ can be captured by a linear program. A strongly polynomial LP algorithm would therefore immediately provide the desired subroutine. Alas, this LP is not contained in any special class of LP where strongly polynomial algorithms are currently known.

We will exploit the following special structure of the LP. We can eliminate the f_{ij} variables and only work with price variables. The objective is to maximize the sum of all variables over a feasible set of the form $P \cap P'$. The first polyhedron P is defined by inequalities with one positive and one nonnegative variable per inequality. The constraint matrix defining the second polyhedron P' is what we call a Z_+-matrix: all off-diagonal elements are nonpositive but all column sums are nonnegative. This corresponds to a submatrix of the transposed of a weighted Laplacian matrix. In case we only had constraints of the form P, classical results [1, 5, 28] would provide a strongly polynomial running time. To deal with the constraints

defining P', we approximate our LP by a second LP that can be solved in strongly polynomial time.

More precisely, we replace the second polyhedron P' by Q such that $\frac{1}{n^2}Q \subseteq P' \subseteq Q$, and that Q is also a system with one positive and one nonnegative variable per inequality. Thus, the algorithms [1, 5, 28] are applicable to maximize the sum of the variables over $P \cap Q$ in strongly polynomial time. For an optimal solution $\bar{p}$, the vector $\bar{p}/n^2$ is feasible to the original LP and the objective value is within a factor n^2 of the optimum. For the purposes of identifying a new revealed arc in the algorithm, such an approximation of the optimal $\phi(p, f)$ value already suffices.

The construction of the approximating polyhedron Q is obtained via a general method applicable for systems given by Z_+-matrices. We show that for such systems, Gaussian elimination can be used to generate valid constraints with at most two nonzero variables per row. Moreover, we show that the intersection of all relevant such constraints provides a good approximation of the original polyhedron.

The full version of the paper, including all proofs, is available at https://arxiv.org/abs/1809.06266. The rest of this extended abstract is structured as follows. Section 2 introduces basic definitions and notation. Section 3 describes the overall algorithm by introducing the notion of F-allocations, the main potential, and the two necessary subroutines. Section 4 presents the first of these two subroutines, a variant of the Duan-Mehlhorn algorithm. Section 5 shows how the second subroutine, Price Boost, can be reduced to solving an LP. Section 6 gives a sketch of the polyhedral approximation result for Z_+-matrices. Section 7 concludes with some open questions.

2 PRELIMINARIES

For a positive integer t, we let $[t] := \{1, 2, \dots, t\}$, and for $k < t$, we let $[k, t] := \{k, k+1, \dots, t\}$. For a vector $a \in \mathbb{R}^n$, we let

$$\|a\|_1 := \sum_{i=1}^{n} |a_i|, \quad \|a\|_2 := \sqrt{\sum_{i=1}^{n} a_i^2}, \quad \text{and} \quad \|a\|_\infty := \max_{i \in [n]} |a_i|$$

denote the ℓ_1, ℓ_2, and ℓ_∞-norms, respectively. Further, for a vector $a \in \mathbb{R}^n$, we let $\mathrm{supp}(a) \subseteq [n]$ denote its support, that is, $\mathrm{supp}(a) := \{i \in [n] \mid a_i \neq 0\}$. For a subset $S \subseteq [n]$, we let $a(S) := \sum_{i \in S} a_i$.

The linear exchange market. We let $A := [n]$ denote the set of agents, $G := \{g_1, g_2, \dots, g_n\}$ denote the set of goods, and $u_{ij} \geq 0$ denote the utility of agent i for a unit amount of good g_j. Let $E \subseteq A \times G$ denote the set of pairs (i, g_j) such that $u_{ij} > 0$; let $m := |E|$. We will assume that for each $i \in A$ there exists a $g_j \in G$ such that $u_{ij} > 0$, and for each $g_j \in G$ there exists an $i \in A$ such that $u_{ij} > 0$. Hence, $m \geq n$.

The goods are divisible and there is one unit of each in total. Agent i arrives to the market with her initial endowment comprising exactly the unit of good g_i. As mentioned in the introduction, the general case with an arbitrary set of goods and arbitrary initial endowments can be easily reduced to this setting; see [9, 24].

Let $p = (p_j)_{g_j \in G}$ denote the prices where p_j is the price of good g_j. Given prices p, we let

$$\alpha_i := \max_{g_k \in G} \frac{u_{ik}}{p_k}, \quad \forall i \in A.$$

For each agent $i \in A$, the goods satisfying equality here are called *maximum bang-per-buck* (MBB) goods; for such a good g_j, (i, g_j) is called an MBB edge. We let $\mathrm{MBB}(p) \subseteq E$ denote the set of MBB edges at prices p.

Definition 2.1. *Let $f = (f_{ij})_{i \in A, g_j \in G}$ denote the money flow where f_{ij} is the money spent by agent i on good g_j. We say that (p, f) is a* money allocation *if*

(i) $p > 0$, *and* $f \geq 0$;
(ii) $\mathrm{supp}(f) \subseteq \mathrm{MBB}(p)$;
(iii) $\sum_{g_j \in G} f_{ij} \leq p_i$ *for every agent* $i \in A$;
(iv) $\sum_{i \in A} f_{ij} \leq p_j$ *for every good* $g_j \in G$.

For the money allocation (p, f), the *surplus* of agent i is defined as

$$c_i(p, f) := p_i - \sum_{g_j \in G} f_{ij},$$

and the surplus of good $g_j \in G$ is defined as

$$s_j(p, f) := p_j - \sum_{i \in A} f_{ij}.$$

Parts (iii) and (iv) in the definition of the money allocation require that the surplus of all agents and goods are nonnegative. We let $c(p, f) := (c_i(p, f))_{i \in A}$ and $s(p, f) := (s_j(p, f))_{g_j \in G}$ denote the surplus vectors of the agents and the goods, respectively. Clearly, $\|s(p, f)\|_1 = \|c(p, f)\|_1$, and $\|s(p, f)\|_\infty \leq \|s(p, f)\|_1 \leq n\|s(p, f)\|_\infty$.

Definition 2.2. *A money allocation (p, f) is called a* market equilibrium *if $\|s(p, f)\|_1 = 0$.*

Existence of an equilibrium. A necessary and sufficient condition for the existence of an equilibrium can be given as follows. Let us define the directed graph $(A, \bar{E})$, where $(i, j) \in \bar{E}$ if and only if $u_{ij} > 0$ (that is, if $(i, g_j) \in E$).

Lemma 2.1 ([9, 17])**.** *There exists a market equilibrium if and only if for every strongly connected component $S \subseteq A$ of the digraph $(A, \bar{E})$, if $|S| = 1$, then there is a loop incident to the node in S.*

This condition can be easily checked in strongly polynomial time. Further, it is easy to see that if the above condition holds, then finding an equilibrium in an arbitrary input can be reduced to finding an equilibrium in an input where the digraph $(A, \bar{E})$ is strongly connected. Thus, we will assume the following throughout the paper:

$$\text{The graph } (A, \bar{E}) \text{ is strongly connected.} \qquad (\star)$$

3 THE OVERALL ALGORITHM

In this section, we describe the overall algorithm. We formulate the statements that are needed to prove our main theorem:

Theorem 3.1. *There exists a strongly polynomial algorithm that computes a market equilibrium in linear exchange markets in time $O(n^{10} \log^2 n)$.*

We start by introducing the concepts of revealed edges, F-money allocations, and balanced F-flows.

3.1 Revealed Edges

Throughout the paper, we let $F^* \subseteq E$ denote the set of edges $(i, g_j) \in E$ that are MBB edges for every market equilibrium (p, f). In the algorithm, we will maintain a subset of edges $F \subseteq F^*$ that will be called the *revealed edge set*. This is initialized as $F = \emptyset$.

Definition 3.2. *For an edge set $F \subseteq E$, (p, f) is an* F-money allocation, *or* F-allocation *in short, if*

(i) $p > 0$, *and* $f_{ij} \geq 0$ *for* $(i, g_j) \notin F$;
(ii) $\mathrm{supp}(f) \cup F \subseteq \mathrm{MBB}(p)$;
(iii) $\sum_{g_j \in G} f_{ij} \leq p_i$ *for every agent* $i \in A$;
(iv) $\sum_{i \in A} f_{ij} \leq p_j$, *for every good* $g_j \in G$.

An F-allocation is called an F-equilibrium *if* $\|s(p, f)\|_1 = 0$.

Note that f_{ij} could be negative for $(i, g_j) \in F$. A $\emptyset$-allocation simply corresponds to a money allocation.

The main progress step in the algorithm will be adding new edges to F. This will be enabled by the following lemma.

Lemma 3.1. *Let $F \subseteq F^*$, and let (p, f) be an F-allocation. If $f_{k\ell} > \|s(p, f)\|_1$ for an edge $(k, g_\ell) \in E$, then $(k, g_\ell) \in F^*$.* □

The proof is given in the full version. It is a proof by contradiction, comparing the sets of MBB goods at prices p and at an equilibrium where (k, g_ℓ) is not an MBB good.

Our algorithm will obtain an F-equilibrium. Whereas an F-equilibrium is not necessarily an equilibrium, the following holds true:

Lemma 3.2. *Let $F \subseteq F^*$, and assume we are given an F-equilibrium (p, f). Then a market equilibrium (p, f') can be obtained in $O(nm)$ time.* □

The proof is via a maximum flow computation in an auxiliary network. We let Final-Flow(p) denote the algorithm as in the Lemma.

3.2 Balanced Flows

Balanced flows play a key role in the Duan-Mehlhorn algorithm [13], as well as in previous algorithms for Fisher market models [10, 21, 37]. We now introduce the natural extension for F-allocations.

Definition 3.3. *Given an edge set $F \subseteq E$ and prices p, we say that (p, f) is a* balanced F-flow, *if (p, f) is an F-allocation that minimizes $\|s(p, f)\|_1$, and subject to that, it minimizes $\|c(p, f)\|_2$.*

Lemma 3.3. *[8, 26] Given $F \subseteq E$ and prices p such that $F \subseteq MBB(p)$, a balanced F-flow can be computed in $O(nm \log(n^2/m))$ time.* □

We let Balanced(F, p) denote the subroutine guaranteed by the Lemma.

3.3 The Algorithm

The overall algorithm is presented in Algorithm 1. The main progress is gradually expanding a revealed edge set $F \subseteq F^*$, initialized as $F = \emptyset$. Every cycle of the algorithm performs the subroutines Boost(F) and DM$(F, \hat{p})$, and at least one new edge is added to F at every such cycle. Once an F-equilibrium is obtained for the current F, we use the subroutine Final-Flow(p) as in Lemma 3.2 to compute a market equilibrium.

Algorithm 1: Arrow-Debreu Equilibrium

Input : Set A of agents, set G of goods, and utilities $(u_{ij})_{i \in A, g_j \in G}$
Output: Market equilibrium (p, f)

$F \leftarrow \emptyset$;
repeat
  $(\hat{p}, \hat{f}) \leftarrow$ Boost(F) // Theorem 3.4
  $(p, f) \leftarrow$ DM$(F, \hat{p})$ // Theorem 3.5
  $F \leftarrow F \cup \{(i, g_j) \mid f_{ij} > \|s(p, f)\|_1\}$ // Lemma 3.1
until $\|s(p, f)\|_1 = 0$
$f \leftarrow$ Final-Flow(p) // Lemma 3.2
return (p, f)

We now introduce the key potential measures used in analysis. For an F-allocation (p, f), we define

$$\phi(p, f) := \frac{\|s(p, f)\|_\infty}{(\prod_{j=1}^n p_j)^{1/n}}.$$

Note that this is invariant under scaling, i.e. $\phi(p, f) = \phi(\alpha p, \alpha f)$ for any $\alpha > 0$. Further, (p, f) is an F-equilibrium if and only if $\phi(p, f) = 0$. For a given $F \subseteq F^*$, we define

$$\Psi(F) := \min\{\phi(p, f) : (p, f) \text{ is an } F\text{-allocation, } \mathrm{supp}(f) \subseteq F\}. \quad (1)$$

Theorem 3.4. *There exists a strongly polynomial time algorithm that for any input $F \subseteq E$, returns in time $O(n^4 \log^2 n)$ an F-allocation $(\hat{p}, \hat{f})$ with $\mathrm{supp}(\hat{f}) \subseteq F$ such that $\Psi(F) \leq \phi(\hat{p}, \hat{f}) \leq (n-1)^2 \Psi(F)$.*

The algorithm in the theorem will be denoted as Boost(F), and is described in Section 5. In particular, if $\Psi(F) = 0$, then Boost(F) returns an F-equilibrium.

The second main subroutine DM$(F, \hat{p})$, is a variant of the Duan-Mehlhorn algorithm [13], described in Section 4. As the input, it uses the prices $\hat{p}$ obtained in the F-allocation $(\hat{p}, \hat{f})$ returned by Boost(F), and outputs an F-allocation (p, f) such that either $\|s(p, f)\|_1 = 0$, that is, an F-equilibrium, or it is guaranteed that $f_{ij} > \|s(p, f)\|_1$ for some $(i, g_j) \in E \setminus F$ connecting two different connected components of F. Such an edge (i, g_j) can be added to F by Lemma 3.1. The following simple lemma asserts the existence of such an edge.

Lemma 3.4. *Let (p, f) be an F-allocation with $\phi(p, f) < \Psi(F)/(n(m+1))$. Then, $f_{ij} > \|s(p, f)\|_1$ for at least one edge $(i, g_j) \in E \setminus F$ such that i and g_j are in two different undirected connected components of F.* □

Theorem 3.5. *There exists a strongly polynomial $O(n^9 \log^2 n)$ time algorithm, that, for a given $F \subseteq E$ and prices $\hat{p}$, computes an F-allocation (p, f) such that*

$$\phi(p, f) \leq \frac{\phi(\hat{p}, \tilde{f})}{n^4(m+1)},$$

where $\tilde{f}$ is the balanced flow computed by Balanced$(F, \hat{p})$.

The algorithm DM$(F, \hat{p})$ given in Section 4 will satisfy the assertion of this theorem. Using these claims, we are ready to prove Theorem 3.1.

Proof of Theorem 3.1. By Lemma 3.4, the number of connected components of F decreases after every cycle of Algorithm 1; thus, the total number of cycles is $\leq 2n-1$. Consider any cycle. Let $(\hat{p}, \hat{f})$ denote that F-allocation returned by Boost(F) with $\phi(\hat{p}, \hat{f}) \leq (n-1)^2\Psi(F)$, and let $(\hat{p}, \tilde{f})$ denote the balanced F-flow at prices $\hat{p}$. Then,

$$\|s(\hat{p}, \tilde{f})\|_\infty \leq \|s(\hat{p}, \tilde{f})\|_1 \leq \|s(\hat{p}, \hat{f})\|_1 \leq n\|s(\hat{p}, \hat{f})\|_\infty,$$

since $(\hat{p}, \tilde{f})$ minimizes $\|s(\hat{p}, \tilde{f})\|_1$ among all F-allocations. Therefore $\phi(\hat{p}, \tilde{f}) < n^3\Psi(F)$. Theorem 3.5 guarantees that DM$(F, \hat{p})$ finds an F-allocation (p, f) with $\phi(p, f) < \Psi(F)/(n(m+1))$. Lemma 3.4 guarantees that F is extended by at least one new edge in this cycle. The overall running time estimation is dominated by the running time estimation of the calls to DM. □

4 THE DUAN-MEHLHORN (DM) SUBROUTINE

In this section, we present a variant of the Duan-Mehlhorn (DM) algorithm [13] as a subroutine DM$(F, \hat{p})$ in Algorithm 2. The input is a revealed edge set F and prices $\hat{p}$ such that $F \subseteq$ MBB$(\hat{p})$, and the output is either an F-equilibrium, or an F-allocation (p, f) where $f_{ij} > \|s(p, f)\|_1$ for some $(i, g_j) \in E \setminus F$ connecting two different components of F. The modifications compared to the original DM algorithm are listed in Remark 4.1. We now provide a description where the subroutine terminates once an arc with $f_{ij} > \|s(p, f)\|_1$ is identified. The variant as required in Theorem 3.5 can be obtained by simply by removing the termination condition, and letting the algorithm run for $O(n^6 \log^2 n)$ iterations of the outer loop.

We call one execution of the outer loop a *phase*, and one execution of the inner loop an *iteration*. Algorithm 2 first computes a balanced flow f using the subroutine Balanced(F, p) as in Lemma 3.3. Then, the agents are sorted in decreasing order of surplus. Without loss of generality, we assume that $c_1(p, f) \geq \cdots \geq c_n(p, f)$. Then, we find the smallest ℓ for which the ratio $c_\ell(p, f)/c_{\ell+1}(p, f)$ is more than $1 + 1/n$. If there is no such ℓ then we let $\ell := n$. Let S be the set of first ℓ agents, and let $\Gamma(S)$ be the set of goods for which there is a non-zero flow from agents in S. Since f is balanced, the agents outside S have zero flow to goods in $\Gamma(S)$, i.e., $f_{ij} = 0, \forall i \notin S, g_j \in \Gamma(S)$ and the surplus of every good in $\Gamma(S)$ is zero. We set γ to 1 before we go into the inner loop.

Next, the algorithm runs the inner loop where it increases the prices of goods in $\Gamma(S)$ and the flow between agents in S and goods in $\Gamma(S)$ by a multiplicative factor $x \geq 1$ until one of the three events occurs. Observe that except for the MBB edges (i, g_j) where $i \notin S, g_j \in \Gamma(S)$, all MBB edges remain MBB with this price change, and the surplus of every good in $\Gamma(S)$ remains zero. When prices of goods in $\Gamma(S)$ increase, an edge (i, g_j) from $i \in S$ and $g_j \notin \Gamma(S)$ can become MBB. We need to stop when such an event occurs in order to maintain an F-allocation. This is captured by Event 1. In Event 2, we stop when the surplus of an agent $i \in S$ becomes equal to either the surplus of an agent $i' \notin S$ or zero. Let us note that $c_i(p, f) \geq 0$ is maintained throughout; we use the expression $\max\{\max_{i \notin S} c_i(p, f), 0\}$ to also cover the possible case $S = [n]$. In Event 3, we stop when γx becomes $1 + 1/(56e^2n^3)$.

If Event 1 occurs, then we have a new MBB edge (a, g_b) from $a \in S$ to $g_b \notin \Gamma(S)$. Using this new edge, it is now possible to decrease the surplus of agent a and increase the surpluses of agents $i \notin S$ by increasing f_{ab} and decreasing f_{ib}. We next check if this can lead to making the surplus of an agent $i \in S$ and $i' \notin S$ equal. Observe that it is always possible if there exists an edge $(i', g_b) \in F$. If yes, then we break from the inner loop, otherwise we update flow so that agent a buys the entire good g_b, add g_b to $\Gamma(S)$, update γ to γx, and go for another iteration.

Algorithm 2: DM$(F, \hat{p})$

Input : Utilities $(u_{ij})_{i \in A, g_j \in G}$, an edge set $F \subseteq E$, and prices $\hat{p}$ with $F \subseteq$ MBB$(\hat{p})$.

Output : An F-equilibrium (p, f) or an F-allocation (p, f) such that $f_{ij} > \|s(p, f)\|_1$ for an $(i, g_j) \in E \setminus F$, where i and g_j are in different connected components of F.

$p \leftarrow \hat{p}$; $f \leftarrow$ Balanced(F, p) // Lemma 3.3
repeat
  Sort the agents in decreasing order of surplus, i.e., $c_1(p, f) \geq c_2(p, f) \geq \ldots \geq c_n(p, f)$
  Find the smallest ℓ for which $c_\ell(p, f)/c_{\ell+1}(p, f) > 1 + 1/n$, and let $\ell = n$ when there is no such ℓ.
  $S \leftarrow [\ell]$; $\Gamma(S) = \{g_j \in G \mid \exists i \in S : f_{ij} \neq 0\}$
  $\gamma \leftarrow 1$
  repeat
    $x \leftarrow 1$; Define $p_j \leftarrow xp_j, \forall g_j \in \Gamma(S), f_{ij} \leftarrow xf_{ij}, \forall i \in S, \forall g_j \in \Gamma(S)$
    // $c_i(p, f)$ and $s_j(p, f)$ change accordingly
    Increase x continuously up from 1 until one of the following events occurs
      Event 1: A new edge, say (a, g_b), becomes MBB // $a \in S$, $g_b \notin \Gamma(S)$
      Event 2: $\min_{i \in S} c_i(p, f) = \max\{\max_{i \notin S} c_i(p, f), 0\}$ // Balancing
      Event 3: $\gamma x = 1 + 1/(56e^2n^3)$ // Price-rise
    if *Event 1 occurs* **then**
      $\tilde{c}_i(p, f) \leftarrow c_i(p, f), \ \forall i \in S \setminus \{a\}$
      $\tilde{c}_a(p, f) \leftarrow c_a(p, f) - p_b$
      $\tilde{c}_i(p, f) \leftarrow c_i(p, f) + f_{ib}, \ \forall i \notin S$
      if $\exists i \in A \setminus S$ *s.t.* $(i, g_b) \in F$ *or* $\min_{i \in S} \tilde{c}_i(p, f) \leq \max\{\max_{i \notin S} \tilde{c}_i(p, f), 0\}$ **then**
        break // break from the inner loop
      $f_{ib} \leftarrow 0, \forall i \in A$; $f_{ab} = p_b$; $\Gamma(S) \leftarrow \Gamma(S) \cup \{g_b\}$; $\gamma \leftarrow \gamma x$
  until *Event 2 or 3 occurs*
  $f \leftarrow$ Balanced(F, p)
until *either* $f_{ij} > \|s(p, f)\|_1$ *for an edge* $(i, g_j) \in E \setminus F$ *with* i *and* g_j *in different components of* F, *or* $\|s(p, f)\|_1 = 0$
return (p, f)

Lemma 4.1. *The number of iterations in a phase is at most n.*

Proof. Consider the iterations of a phase. At the beginning of every iteration, the size of $\Gamma(S)$ grows by 1, and hence there cannot be more than n iterations in a phase. □

When we break from the inner loop, we recompute a balanced flow and then check if either $\|s(p,f)\|_1$ is zero or there is an edge $(i,g_j) \notin F$ with $f_{ij} > \|s(p,f)\|_1$ connecting two different components of F. If yes, then we return the current (p,f), otherwise we go for another phase. Next, we show that (p,f) remains an F-allocation throughout the algorithm, which implies that the algorithm returns an F-allocation.

Lemma 4.2. *The output (p,f) of Algorithm 2 is an F-allocation.*

Proof. We only need to show that $F \subseteq \mathrm{MBB}(p)$ throughout the algorithm. Observe that an MBB edge (i,g_j) becomes non-MBB only if $i \notin S$ and $g_j \in \Gamma(S)$, where S and $\Gamma(S)$ are obtained with respect to a balanced flow f. If an edge $(i,g_j) \in F$ is such that $i \notin S$ and $g_j \in \Gamma(S)$ then it contradicts that f is a balanced flow because the edges in F are allowed to carry negative flow. □

The running time analysis of Algorithm 2 is based on the evolution of the norm $\|c(p,f)\|_2$ and prices p. If a phase terminates due to Event 3, then we call it *price-rise*, otherwise *balancing*. The next two lemmas are crucial that eventually imply that the potential function $\phi(p,f)$ decreases substantially within a strongly polynomial number of phases.

Lemma 4.3. *In Algorithm 2, the price of every good monotonically increases and the total surplus, i.e., $\|s(p,f)\|_1$, monotonically decreases.*

Proof. Clearly, the price of every good monotonically increases in Algorithm 2. During a price increase step, $s_j(p,f) = 0$ is maintained for every $g_j \in \Gamma(S)$, and $s_j(p,f)$ does not change for $g_j \in G \setminus \Gamma(S)$. If the allocation changes during Event 1, then $s_b(p,f)$ decreases to 0, and the other surpluses remain unchanged. When a balanced flow is recomputed at the end of a phase, then $\|s(p,f)\|_1$ can only decrease. □

The proof of the next lemma is an adaptation of the proof in [12], and is given in the full version, along with the definition of the auxiliary network $N(p,F)$.

Lemma 4.4. *Let f be a balanced flow in $N(p,F)$ at the beginning of a phase, and let (p',f') be the prices and flow at the end of the phase. Then*

(i) $\prod_{j=1}^n p'_j \geq \left(1+\frac{1}{Cn^3}\right)\prod_{j=1}^n p_j$ *in a price-rise phase, and*

(ii) $\|c(p',f')\|_2 \leq \|c(p,f)\|_2 / \left(1+\frac{1}{Cn^3}\right)$ *in a balancing phase,*

where $C = 56e^2$. □

Lemma 4.5. *The number of arithmetic operations in a phase of Algorithm 2 is $O(n^3)$.* □

In the next lemma, we show that the potential function $\phi(p,f)$ decreases by a large factor within a strongly polynomial number of phases. This together with Lemmas 3.4 and 3.1 imply that every major cycle terminates in strongly polynomial time.

Lemma 4.6. *The potential function $\phi(p,f)$ decreases by a factor of at least $1/n^\gamma$ in $4(2+\gamma)^2C^2n^6\ln^2 n$ phases of Algorithm 2 for any $\gamma > 0$, where $C = 56e^2$.* □

Proof of Theorem 3.5. According to the above lemma, if we do not terminate Algorithm 2 in the first iteration when an arc $(i,g_j) \in E \setminus F$ with $f_{ij} > \|s(p,f)\|_1$ is found, then the potential $\phi(p,f)$ decreases by a factor $n^4(m+1)$ within $O(n^6\log^2 n)$ phases.

For a strongly polynomial algorithm, we also need to keep all intermediate numbers polynomial bit length. For this, we can use the Duan-Mehlhorn [13] technique by restricting the prices and update factor x to powers of $(1+1/L)$ where L has polynomial bit length. This guarantees that all arithmetic is performed on rational numbers of polynomial bit length. As shown in [13] this does not change the number of iterations of the DM subroutine. □

Remark 4.1. Compared to the original DM algorithm in [13], Algorithm 2 differs in the following.

(1) We handle Event 1 (in line 10) differently than the other two events and this gives rise to two nested loops, unlike [13] where every event is handled similarly and there is only one loop.
(2) The edges in F are allowed to carry negative flow, unlike [13] where flow is always non-negative.
(3) We initialize prices to p, unlike [13] where every price is initialized to 1. And, we stop when a new edge is revealed.

5 A LINEAR PROGRAM FOR $\Psi(F)$

In this section, we first formulate an LP to compute $\Psi(F)$. Then, we introduce the class of Z_+-matrices, and formulate a general statement (Theorem 5.3) that shows how certain LPs with a Z_+ constraint matrix can be approximated by a two variable per inequality system. We use this to prove Theorem 3.4. The proof of Theorem 5.3 will be given in Section 6.

Given $F \subseteq E$, we consider the bipartite graph $(A \cup G, F)$. Let $C_1, C_2, \ldots, C_t$ denote the connected components that have a non-empty intersection with G. (In particular, we include all isolated vertices in G, but not those in A.) Let $\gamma_i := |C_i \cap G|$. Let us fix an arbitrary good in each of these components; for simplicity of notation, let us assume that the fixed good in C_i is g_i.

If all edges in F are forced to be MBB edges, then fixing the price p_i of g_i uniquely determines the prices of all goods in $C_i \cap G$. Indeed, for any buyer $k \in C_i \cap A$, and any goods $g_\ell, g_{\ell'} \in C_i \cap G$ with $k\ell, k\ell' \in F$, we have that $p_\ell/p_{\ell'} = u_{k\ell}/u_{k\ell'}$. Consequently, for any $i \in [t]$, and for any $g_\ell \in C_i \cap G$, we can compute the multiplier $\theta_{i\ell} > 0$ such that $p_\ell = \theta_{i\ell}p_i$ whenever all edges in F are MBB. For an agent $\ell \in A$, let $\rho(\ell) \in [t]$ denote the index of the component containing the good g_ℓ of this agent: that is, $g_\ell \in C_{\rho(\ell)} \cap G$, and $p_\ell = \theta_{\rho(\ell)\ell}p_{\rho(\ell)}$. Let $\Theta_i := \sum_{g_\ell \in C_i \cap G} \theta_{i\ell}$; the total price of the goods in C_i is $\Theta_i p_i$.

5.1 Constructing the LP

The variables $(p_i)_{i\in[t]}$ uniquely determine the price of every good. We can formulate the problem of computing $\Psi(F)$ in terms of these variables. To differentiate between this t-dimensional price vector and the n-dimensional price vector of all goods, we say that for a price vector $\bar{p} \in \mathbb{R}^t$, the vector $p \in \mathbb{R}^n$ is the *extension* of $\bar{p}$, if $p_\ell = \theta_{\rho(\ell)\ell}\bar{p}_{\rho(\ell)}$ for all $\ell \in [n]$ (in particular, $p_\ell = \bar{p}_\ell$ for $\ell \in [t]$). We also say that the F-allocation (p,f) is an extension of $\bar{p}$, if p is the extension of $\bar{p}$.

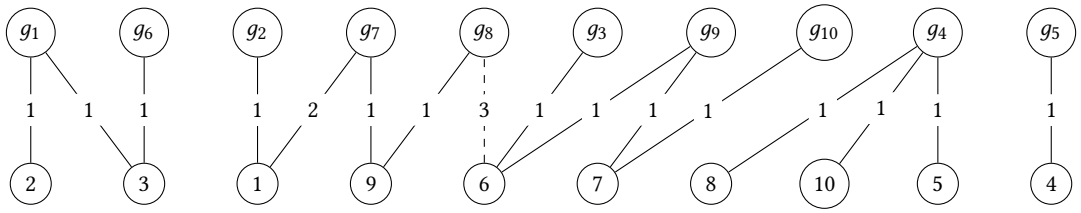


Figure 1: Example problem setting.

Example. Throughout this and the next section, we illustrate the argument with the example in Figure 1. There are 10 agents and 15 edges in F. The edges in F are depicted by solid edges with the u_{ij} values indicated; all these are 1 except for $u_{17} = 2$. The picture does not include the edges in $E \setminus F$ except for one example: the dashed line for $(6, g_8)$ with $u_{68} = 3$. There are 5 connected components, containing goods $\{g_1, g_6\}$, $\{g_2, g_7, g_8\}$, $\{g_3, g_9, g_{10}\}$, $\{g_4\}$, and $\{g_5\}$, with $p_6 = p_1$, $p_7 = p_8 = 2p_2$, and $p_3 = p_9 = p_{10}$. Thus, $\Theta_1 = 2$, $\Theta_2 = 5$, $\Theta_3 = 3$, $\Theta_4 = 1$, $\Theta_5 = 1$, and $\gamma_1 = 2$, $\gamma_2 = 3$, $\gamma_3 = 3$, $\gamma_4 = 1$, $\gamma_5 = 1$.

We now formulate linear constraints that ensure that a vector $\bar{p} \in \mathbb{R}^t$ can be extended to an F-allocation (p, f) with $\|s(p, f)\|_\infty \le 1$, and $\mathrm{supp}(f) \subseteq F$. The first set of constraints will enforce that all edges in F are MBB, and the second set will guarantee the existence of a desired money flow f with the surplus bounds.

First, the edges in F are MBB if and only if $u_{kj}/p_j \le u_{kj'}/p_{j'}$ for any $k \in A$, and any $g_j, g_{j'} \in G$ such that $(k, g_j) \in E$, $(k, g_{j'}) \in F$. The $\theta_{i\ell}$ coefficients already capture that equality holds if $(k, g_j), (k, g_{j'}) \in F$. For the rest of the pairs, we can express this constraint in terms of the $\bar{p}$ variables as

$$u_{kj}\theta_{\rho(j')j'}\bar{p}_{\rho(j')} - u_{kj'}\theta_{\rho(j)j}\bar{p}_{\rho(j)} \le 0 \qquad \forall k, j, j' \in A,\ (k,g_j) \in E \setminus F,\ (k, g_{j'}) \in F. \tag{2}$$

We add a second set of constraints for $\|s(p, f)\|_\infty \le 1$. Since f is supported on F and is allowed to be negative, this can be guaranteed if and only if for any component C_i, $i \in [t]$, the total price of the goods in $C_i \cap G$ exceeds the total budget of the agents in $C_i \cap A$ by at most $\gamma_i = |C_i \cap G|$. Recall that given the prices $\bar{p}$ of the fixed goods, the total price of goods in $C_i \cap G$ is $\Theta_i \bar{p}_i$. We obtain the constraints

$$\Theta_i \bar{p}_i - \sum_{k \in C_i \cap A} \theta_{\rho(k)k}\bar{p}_{\rho(k)} \le \gamma_i \quad \forall i \in [t]. \tag{3}$$

Let us now define the following LP:

$$\begin{aligned} &\max \sum_{i=1}^{t} \bar{p}_i \\ &\text{s. t. constraint sets (2) and (3),} \\ &\bar{p} \ge 0. \end{aligned} \tag{P_F}$$

Note that $\bar{p} = 0$ is a feasible solution. Using LP duality, the above program is unbounded if and only if the next LP has a feasible solution $\bar{p} \neq 0$.

$$\begin{aligned} &\text{constraint set (2),} \\ &\Theta_i \bar{p}_i - \sum_{k \in C_i \cap A} \theta_{\rho(k)k}\bar{p}_{\rho(k)} \le 0 \quad \forall i \in [t] \\ &\bar{p} \ge 0. \end{aligned} \tag{P_F^0}$$

Example. Let us show the formulation for the example in Figure 1. The variables are $\bar{p}_1, \bar{p}_2, \bar{p}_3, \bar{p}_4$, and $\bar{p}_5$. From the constraint set (2), we only show the example of $k = 6$, $j = 8$, and $j' = 3$:

$$3\bar{p}_3 - 2\bar{p}_2 \le 0.$$

For the components, we have

$$\begin{aligned} 2\bar{p}_1 - \bar{p}_2 - \bar{p}_3 &\le 2 \\ 5\bar{p}_2 - \bar{p}_1 - \bar{p}_3 &\le 3 \\ 3\bar{p}_3 - \bar{p}_1 - 2\bar{p}_2 &\le 3 \\ \bar{p}_4 - 2\bar{p}_2 - \bar{p}_3 - \bar{p}_5 &\le 1 \\ \bar{p}_5 - \bar{p}_4 &\le 1. \end{aligned}$$

Lemma 5.1. *(i) Any solution $\bar{p} \in \mathbb{R}^n$ to (P_F) can be extended to an F-allocation (p, f) with $\|s(p, f)\|_\infty \le 1$, and $\mathrm{supp}(f) \subseteq F$.*

(ii) If (P_F) is bounded, then there exists a pointwise maximal solution $\bar{p}^* \in \mathbb{R}^t$, that is, $\bar{p} \le \bar{p}^*$ for any solution $\bar{p} \in \mathbb{R}^t$ to (P_F). Let (p^*, f^*) denote the extension of these prices to an F-allocation with $\|s(p^*, f^*)\|_\infty \le 1$, and $\mathrm{supp}(f^*) \subseteq F$. Then, $\Psi(F) = \phi(p^*, f^*)$.

(iii) Under assumption ($\star$), every nonzero solution to (P_F^0) is strictly positive. Such a solution can be extended to an F-equilibrium. □

5.2 Monotone Two Variable Per Inequality Systems

Let $\mathcal{M}_2(m, n)$ denote the set of $m \times n$ rational matrices such that every matrix contains at most one positive and at most one negative entry per row. For a matrix $A \in \mathcal{M}_2(m, n)$, and an arbitrary vector $b \in \mathbb{Q}^m$, the LP $Ax \le b$ is called a *monotone two variable per inequality system*, abbreviated as *M2VPI*. In every such system, whenever the objective function $\max \sum_i x_i$ is bounded, there exists a *pointwise maximal feasible solution*, that is, a feasible x^* such that for every feasible solution x, $x \le x^*$.

This property holds more generally. Namely, a matrix is called *pre-Leontief* if every column contains at most one positive element. If $A^\top$ is pre-Leontief, then the system $A^\top x \le c$ has a pointwise maximal feasible solution whenever $\max \sum_i x_i$ is bounded [7]. Whereas

M2VPI systems are strongly polynomially solvable, as stated in the next theorem, no such algorithm is known for the general pre-Leontief setting.

Theorem 5.1 ([1, 5, 28]). *Consider an M2VPI system $Ax \le b$ with $A \in \mathcal{M}_2(m,n)$. Then there exists a strongly polynomial $O(mn \log m + mn^2 \log^2 n)$ time algorithm that finds a feasible solution or concludes infeasibility. Further, if there exists a pointwise maximal feasible solution, the algorithm also finds that one.*

Note that this theorem is not directly applicable to (P_F). Whereas the constraints (2) are of the required form, the constraints (3) have only one positive coefficient but possibly multiple negative ones. In what follows, we show that finding an approximate solution to (P_F) can be reduced to an M2VPI system.

5.3 Z_+-Matrices

Let $M \in \mathbb{R}^{t\times t}$ be the matrix representing the left hand side of the constraints in (3). That is, for all $i,j \in [t]$, we let

$$M_{ij} := \begin{cases} \Theta_i - \sum_{k\in C_i \cap A:\rho(k)=i} \theta_{ik}, & \text{if } i=j, \\ -\sum_{k\in C_i\cap A:\rho(k)=j} \theta_{jk}, & \text{if } i\neq j. \end{cases} \tag{4}$$

Thus, (3) can be written as $M\bar{p} \le \gamma$, where $\gamma^\top = (\gamma_1,\dots,\gamma_t)$.

Definition 5.2. *A matrix $M \in \mathbb{Q}^{k\times t}$ is a Z_+-matrix, if all off-diagonal entries are nonpositive,[1] and all column sums are nonnegative. We let $\mathcal{Z}_+(k,t)$ denote the set of $k\times t$ Z_+-matrices.*

Clearly, the matrix M defined by (4) is in $\mathcal{Z}_+(t,t)$. Recall that a matrix is called a Z-matrix if all off-diagonal entries are nonpositive; the notation reflects the additional requirement on the columns. Further, note that a matrix is a $Z_+(t,t)$-matrix if and only if it is the transposed of a weighted Laplacian of a directed graph on t vertices, or if it can be obtained by deleting a row and a column of the transposed of a weighted Laplacian of a directed graph on $t+1$ vertices. We will prove the following theorem on LPs with Z_+-matrices as constraint matrices.

Theorem 5.3. *Given a matrix $M \in \mathcal{Z}_+(k,t)$ with ℓ nonzero entries, and $b \in \mathbb{Q}^k$, with $b > 0$, we let*

$$P_M = \{x \in \mathbb{R}^t : Mx \le b, x \ge 0\}.$$

Then, in time $O(\ell t^3)$, we can construct a matrix $\bar{M} \in \mathcal{M}_2(\ell',t)$ and $\bar{b} \in \mathbb{Q}^{l'}$ for $\ell' \le \ell$ such that

$$P_M \subseteq \{x \in \mathbb{R}^t : \bar{M}x \le \bar{b}, x \ge 0\} \subseteq B^2 P_M,$$

where $B = \sum_{j=1}^k b_j / \min_{i\in[k]} b_i$. Further, the size of the entries in $\bar{M}$ and $\bar{b}$ will be polynomially bounded in the encoding size of the input.

Here, we use the notation $\alpha P = \{\alpha x : x \in P\}$ for a set P and a constant $\alpha > 0$. The proof of Theorem 5.3 will be given in Section 6; we now use it to derive Theorem 3.4.

Proof of Theorem 3.4. Lemma 5.1 establishes that computing $\Psi(F)$ is equivalent to solving the LP (P_F). We construct a second LP Q_F as follows. For the constraint set (3) in the form $M\bar{p} \le \gamma$, we apply Theorem 5.3 to obtain $\bar{M}\bar{p} \le \bar{\gamma}$. Note that $B \le n$, since $\sum_{i=1}^t \gamma_i = n$, and $\gamma_i \ge 1$ for $i \in [t]$. Then, we let

$$Q_F := \{\bar{p} \in \mathbb{R}^t : \bar{p} \text{ satisfies (2) and } \bar{M}\bar{p} \le \bar{\gamma}\}.$$

Let P_F denote the feasible region of (P_F). Using that all right hand sides in (2) are 0, we see that

$$P_F \subseteq Q_F \subseteq n^2 P_F.$$

Since Q_F is an M2VPI system, Theorem 5.1 provides a strongly polynomial algorithm to obtain the prices $\bar{p}$ maximizing $\sum_{i=1}^t \bar{p}_i$ over Q_F, or concludes that this objective is unbounded on Q_F. In case a finite optimum exists, $\bar{p}/n^2$ is feasible to (P_F) and is within a factor n^2 from an optimal solution.

If the objective is unbounded on Q_F, then we claim that we can get a nonzero solution to (P_F^0). Using LP duality, the objective is unbounded on Q_F if and only if there is a feasible solution $\bar{p} \neq 0$ to

$$Q_F^0 = \{\bar{p} \in \mathbb{R}^t : \bar{p} \text{ satisfies (2) and } \bar{M}\bar{p} \le 0\}.$$

Again, Theorem 5.1 is applicable to find a nonzero solution q. Suppose $q \neq 0$ is a solution to Q_F^0. This implies that αq is a feasible solution to Q_F for all $\alpha \ge 0$. Since for every feasible solution $\bar{p}$ to Q_F, $\bar{p}/n^2$ is a feasible solution to (P_F), this further implies that αq is also a feasible solution to (P_F) for all $\alpha \ge 0$. Therefore, q must be a solution to (P_F^0).

The number of nonzero entries in M is $\le 2n$. Thus, constructing $\bar{M}$ and $\bar{\gamma}$ takes $O(n^4)$ time. We obtain an M2VPI system with $\le m+2n$ constraints and $\le n$ variables, and $m = O(n^2)$, thus the running time for solving the M2VPI system is $O(n^4 \log^2 n)$ that dominates the total running time.

Finally, for a strongly polynomial algorithm we also have to provide polynomial bounds on the encoding lengths of the numbers during the algorithm. The entries of M are simple expressions of the input parameters u_{ij}. Then, Theorem 5.3 guarantees that $\bar{M}$ and vector $\bar{\gamma}$ also have bounded encoding length. Thus, the strongly polynomial M2VPI algorithm takes a polynomial size input and therefore the overall algorithm will be strongly polynomial. □

6 APPROXIMATING SYSTEMS WITH Z_+-MATRICES

This section is dedicated to the proof of Theorem 5.3. For consistency with the market terminology, we use $\bar{p} \in \mathbb{R}^t$ as the variables. Recall that we need to show that given a system $P_M = \{\bar{p} \in \mathbb{R}^t : M\bar{p} \le \gamma,\ \bar{p} \ge 0\}$ with $M \in \mathcal{Z}_+(k,t)$ with ℓ nonzero entries and $\gamma \in \mathbb{Q}^k, \gamma > 0$, we can construct a matrix $\bar{M} \in \mathcal{M}_2(\ell',t)$ and $\bar{\gamma} \in \mathbb{Q}^{l'}$ for $\ell' \le \ell$ in $O(\ell t^3)$ time such that $P_M \subseteq \{\bar{p} \in \mathbb{R}^t : \bar{M}\bar{p} \le \bar{\gamma},\ \bar{p} \ge 0\} \subseteq B^2 P_M$, where $B = \sum_{j=1}^k \gamma_j / \min_{i\in[k]} \gamma_i$.

Let $M_i \in \mathbb{R}^t$ denote the i-th row of the matrix M for $i \in [t]$. We will assume that $k = t$, that is, M is a square matrix. Indeed, if $k > t$, then the last $k-t$ rows only contain nonpositive coefficients. Therefore, for $i > t$, $M_i\bar{p} \le 0$ holds for every $\bar{p} \ge 0$. If $t > k$, then by the Z_+-property, all entries of the last $t-k$ columns must be 0, and thus, we can delete these columns. We further assume that all diagonal entries are strictly positive; if $M_{ii} = 0$ then we can remove the i-th row and i-th column similarly.

Let us also note that every matrix in $\mathcal{Z}_+(t,t)$ can be obtained in the form (4), corresponding to a market problem with t components. We start by showing a lower bound on $M\bar{p}$.

[1] For a non-square matrix, by diagonal entries we mean all entries z_{ii} for $1 \le i \le \min\{k,t\}$.

Lower bounding the vector $M\bar{p}$. For $i \in [t]$, we let $\lambda_i := \sum_{j \neq i} \gamma_j$, and we let $\lambda := (\lambda_1, \ldots, \lambda_t)^\top$.

Lemma 6.1. *Let $M \in \mathcal{Z}_+(t,t)$. Assume that $\bar{p} \in \mathbb{R}^t_+$ satisfies $M\bar{p} \leq \gamma$. Then, we also have*

$$M\bar{p} \geq -\lambda \geq -(B-1)\gamma. \qquad \square$$

6.1 Gaussian Elimination for Z_+-Matrices

We will use Gaussian elimination to generate new constraints. For this purpose, we show that Gaussian elimination on Z_+-matrices will only add nonnegative multiples of rows to other rows.

Lemma 6.2. *Let $T \in \mathcal{Z}_+(\ell, t)$. Then, using Gaussian elimination, we can obtain a matrix $T' = YT$, where $T' \in \mathbb{R}^{\ell \times t}$ is an upper triangular matrix with diagonal entries 0 or 1, and all off-diagonal entries are nonpositive; further, all entries of $Y \in \mathbb{R}^{\ell \times \ell}$ are nonnegative. If $T_{ik} < 0$ for some $k \in [t]$, $i \in [k+1, \ell]$, then $T'_{kk} = 1$.*

Proof. Let $T^{(k)} = Y^{(k)}T$ be the matrix after k steps in the Gaussian elimination with $T^{(0)} = T$ and $Y^{(0)} = I_\ell$. By induction on k, we simultaneously show the following:

- $Y^{(k)}$ is a nonnegative matrix;
- $\sum_{i=k+1}^{\ell} T^{(k)}_{ij} \geq 0$ for $j \in [k+1, t]$;
- $T^{(k)}_{ij} \leq 0$ for $i \neq j$;
- $T^{(k)}_{ii} \geq 0$ for all $i \in [k]$.

Note that the last three properties imply that the lower right $(\ell - k) \times (t - k)$ submatrix of $T^{(k)}$ is a Z_+-matrix.

The properties clearly hold for $k = 0$; assume we have proved these for $k-1$. Consider the k-th iteration. If $T^{(k-1)}_{kk} = 0$, then no row operation is performed. In this case, we set $T^{(k)} := T^{(k-1)}$ and $Y^{(k)} := Y^{(k-1)}$. We only need to verify that $\sum_{i=k+1}^{\ell} T^{(k)}_{ij} \geq 0$ for $j \in [k+1, t]$. This follows from the induction hypotheses: $\sum_{i=k}^{\ell} T^{(k-1)}_{ij} \geq 0$, and $T^{(k-1)}_{kj} \leq 0$ for $j \in [k+1, t]$.

If $T^{(k-1)}_{kk} > 0$, then we multiply the k-th row by $1/T^{(k-1)}_{kk}$, and add $-T^{(k-1)}_{ik}/T^{(k-1)}_{kk}$ times the k-th row to the i-th row for all $i \in [k+1, \ell]$. By induction, these coefficients are all nonnegative. We update the transformation matrix $Y^{(k)}$ accordingly, and thus it remains a nonnegative matrix. Consider now the j-th column of $T^{(k)}$ for $j \in [k+1, t]$. We have

$$\sum_{i=k+1}^{\ell} T^{(k)}_{ij} = \sum_{i=k+1}^{\ell} T^{(k-1)}_{ij} - T^{(k-1)}_{kj} \sum_{i=k+1}^{\ell} T^{(k-1)}_{ik}/T^{(k-1)}_{kk}. \quad (5)$$

The second induction hypothesis for $j = k$ gives

$$T^{(k-1)}_{kk} + \sum_{i=k+1}^{\ell} T^{(k-1)}_{ik} \geq 0.$$

Rearranging, and using that $T^{(k-1)}_{kk} > 0$, we obtain

$$-\sum_{i=k+1}^{\ell} T^{(k-1)}_{ik}/T^{(k-1)}_{kk} \leq 1.$$

If we multiply this by $T^{(k-1)}_{kj} \leq 0$, we get

$$-T^{(k-1)}_{kj} \sum_{i=k+1}^{\ell} T^{(k-1)}_{ik}/T^{(k-1)}_{kk} \geq T^{(k-1)}_{kj}.$$

Substituting into (5), this yields

$$\sum_{i=k+1}^{\ell} T^{(k)}_{ij} \geq \sum_{i=k+1}^{\ell} T^{(k-1)}_{ij} + T^{(k-1)}_{kj} \geq 0,$$

again by the induction hypothesis.

For the last part, let $T_{ik} < 0$ for $k \in [t]$, $i \in [k+1, \ell]$. Note that $T^{(k-1)}_{ik} \leq T^{(k-2)}_{ik} \leq \ldots \leq T^{(0)}_{ik} < 0$. The induction hypothesis gives $\sum_{j=k}^{\ell} T^{(k-1)}_{jk} \geq 0$, and therefore $T^{(k-1)}_{kk} > 0$. We set $T^{(k)}_{kk} = 1$, and this entry does not change in any later steps of the algorithm. $\square$

6.2 Constructing the Approximate System

Let us now describe the construction of the M2VPI system $\bar{M}p \leq \bar{\gamma}$ as in Theorem 5.3. We define a digraph $([t], H)$ by adding an arc $ij \in H$ if $M_{ij} < 0$. For each $i \in [t]$, we let $D_i \subseteq [t]$ be the set of vertices reachable from i in the digraph $([t], H)$, and let $d_i := |D_i|$. We let $M^{(i)}$ denote the $d_i \times t$ submatrix of M comprising the rows M_j for $j \in D_i$. We partition $[t]$ into three groups:

$$T_1 := \{i \in [t] : |\{j : ij \in H\}| \leq 1\},$$
$$T_2 := \{i \in [t] \setminus T_1 : \mathrm{rk}(M^{(i)}) = d_i\},$$
$$T_3 := \{i \in [t] \setminus T_1 : \mathrm{rk}(M^{(i)}) < d_i\}.$$

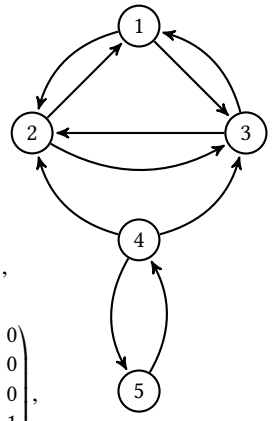
If $i \in T_1$, then M_i has at most one positive and at most one negative entry; thus, we can keep the constraint $M_i\bar{p} \leq \gamma_i$ unchanged. For $i \in T_2 \cup T_3$, for every outgoing arc $ij \in H$, we shall define a constraint in the form $v^{(ij)\top}\bar{p} \geq \delta^{(ij)}$. Further, for every $i \in T_2$, we shall add an additional constraint $\bar{p}_i \leq \kappa_i$.

The construction is somewhat technical, even though the underlying idea is relatively simple. For each $ij \in H$, we wish to obtain the constraint $v^{(ij)\top}\bar{p} \geq \delta^{(ij)}$ such that $\bar{p}_j$ has a positive coefficient, $\bar{p}_i$ has a nonpositive coefficient, and all other coefficients are 0. We wish to derive a valid constraint for (P_F) by taking a nonnegative combination of constraints from $M\bar{p} \geq -\lambda$; recall from Lemma 6.1 that these are valid for (P_F). Lemma 6.2 shows that when we apply Gaussian elimination to a Z_+-matrix, then we only add rows with nonnegative coefficients. Hence, if we apply Gaussian elimination to the matrix M, and apply the same operations to the right hand side $-\lambda$, then we can derive valid constraints from $M\bar{p} \geq -\lambda$. In the construction that follows, we apply a permutation to a submatrix of M where in the penultimate step of Gaussian elimination produces a constraint of the desired form.

Let $i \in T_2 \cup T_3$, and $d := d_i$. For every $ij \in H$, let us define a permutation $\sigma^{(ij)}$ of the set $[t]$ as follows. We set $\sigma(d-1) = j$, $\sigma(d) = i$, and fill the first $d-2$ positions with the elements of $D_i \setminus \{i, j\}$ in such a way that for any $\ell \in D_i \setminus \{i\}$, there is an edge $\ell'\ell \in H$ such that $\sigma(\ell) < \sigma(\ell') \leq d$. The final $t-d$ positions contain the elements of $[t] \setminus D_i$ in an arbitrary order. Let $M^{(ij)} \in \mathbb{R}^{d \times t}$ denote the matrix obtained from $M^{(i)}$ by applying the permutation $\sigma^{(ij)}$ to the rows and the columns, and deleting the last $t-d$ rows.

It is easy to see that $M^{(ij)}$ is a Z_+-matrix.

Example. The picture shows the graph $([t], H)$ for the system obtained from Figure 1. We have $D_1 = D_2 = D_3 = \{1,2,3\}$, and $D_4 = D_5 = \{1,2,3,4,5\}$. Now,



$$M^{(1)} = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 5 & -1 & 0 & 0 \\ -1 & -2 & 3 & 0 & 0 \end{pmatrix},$$

$$M^{(4)} = \begin{pmatrix} 2 & -1 & -1 & 0 & 0 \\ -1 & 5 & -1 & 0 & 0 \\ -1 & -2 & 3 & 0 & 0 \\ 0 & -2 & -1 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix},$$

and $M^{(2)} = M^{(3)} = M^{(1)}$, $M^{(5)} = M^{(4)}$. We get $T_1 = \{5\}$, $T_2 = \{1,2,3\}$, and $T_3 = \{4\}$.

Let us apply Gaussian elimination as in Lemma 6.2 to $M^{(ij)}$ to obtain an upper triangular matrix $N^{(ij)} = Y^{(ij)}M^{(ij)}$. Let $\gamma^{(ij)}, \lambda^{(ij)} \in \mathbb{R}^d$ be the vectors obtained by permuting the components of γ and λ with $\sigma^{(ij)}$, and removing the last $t-d$ entries.

Let us set $v^{(ij)}$ to be the $(d-1)$-st row $N^{(ij)}_{d-1}$ with the inverse of the permutation $\sigma^{(ij)}$ applied to its elements. (So that its i-th coordinate corresponds to $\bar{p}_i$). Let

$$\delta^{(ij)} := -Y^{(ij)}_{d-1}\lambda^{(ij)}. \tag{6}$$

For $i \in T_2$, we add an additional constraint $\bar{p}_i \le \kappa^{(i)}$. Let us pick an arbitrary $ij \in H$, and let

$$\kappa^{(i)} := Y^{(ij)}_d \gamma^{(ij)}. \tag{7}$$

It will be shown in Lemma 6.3 that this value is independent of the choice of the arc ij. The LP $\bar{M}\bar{p} \le \bar{\gamma}$ will be the following system:

$$\begin{aligned} M_i\bar{p} &\le \gamma_i && \forall i \in T_1, \\ \bar{p}_i &\le \kappa^{(i)} && \forall i \in T_2, \\ {v^{(ij)}}^\top \bar{p} &\ge \delta^{(ij)} && \forall ij \in H, i \in T_2 \cup T_3, \\ \bar{p} &\ge 0. \end{aligned} \tag{8}$$

6.3 Proof of Correctness

We need one more claim before proving Theorem 5.3.

Claim 6.1. *Let $i \in [t]$ and let $d := d_i$.*

(i) For any $ij \in H$, $N^{(ij)}_{kk} = 1$ for all $k \in [d-1]$, and $N^{(ij)}_{k\ell} = 0$ for all $k \in [d]$, $\ell \in [d+1, t]$.

(ii) If $i \in T_2$, then $N^{(ij)}_{dd} = 1$, and if $i \in T_3$, then $N^{(ij)}_{dd} = 0$.

(iii) If $i \in T_3$, then M_i can be written as a linear combination of the vectors $\{M_h : h \in D_i \setminus \{i\}\}$. □

Proof of Theorem 5.3. **Form of the constraints.** First, let us show that the system $\bar{M}\bar{p} \le \bar{\gamma}$ given in (8) is an M2VPI system. This is clearly true for the constraints for $i \in T_1$ and for $i \in T_2$. Consider now the constraints ${v^{(ij)}}^\top \bar{p} \ge \delta^{(ij)}$. The vector $v^{(ij)}$ was obtained as the appropriate permutation of the $(d-1)$-st row of the matrix $N^{(ij)}$. According to Claim 6.1, this row may contain nonzero entries

Example. Continuing with the example, we have $\gamma^\top = (2,3,3,1,1)$, and $\lambda^\top = (8,7,7,9,9)$. Let us consider $i = 1$, $j = 2$. We use the permutation $\sigma^{(12)} = (32145)$, yielding

$$M^{(12)} = \begin{pmatrix} 3 & -2 & -1 & 0 & 0 \\ -1 & 5 & -1 & 0 & 0 \\ -1 & -1 & 2 & 0 & 0 \end{pmatrix},$$

$$\gamma^{(12)} = \begin{pmatrix} 3 \\ 3 \\ 2 \end{pmatrix}, \quad \lambda^{(12)} = \begin{pmatrix} 7 \\ 7 \\ 8 \end{pmatrix}.$$

From Gaussian elimination, we get

$$N^{(12)} = \begin{pmatrix} 1 & -\frac{2}{3} & -\frac{1}{3} & 0 & 0 \\ 0 & 1 & -\frac{4}{13} & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix},$$

$$Y^{(12)}\gamma^{(12)} = \begin{pmatrix} 1 \\ \frac{12}{13} \\ \frac{59}{15} \end{pmatrix}, \; Y^{(12)}\lambda^{(12)} = \begin{pmatrix} \frac{7}{3} \\ \frac{28}{13} \\ \frac{181}{15} \end{pmatrix}.$$

This yields the constraint ${v^{(12)}}^\top \bar{p} \ge \delta^{(12)}$ for ${v^{(12)}}^\top = \left(-\frac{4}{13}, 1, 0, 0, \right)$, and $\delta^{(12)} = -\frac{28}{13}$, that is,

$$-\frac{4}{13}\bar{p}_1 + \bar{p}_2 \ge -\frac{28}{13}.$$

Further, we can also use this to obtain $\bar{p}_1 \le \kappa^{(1)}$ for $\kappa^{(1)} = \frac{59}{15}$, that is,

$$\bar{p}_1 \le \frac{59}{15}.$$

The system Q_F comprises the constraint set (2), and the following constraints:

$$\begin{aligned} \bar{p}_2 - \frac{4}{13}\bar{p}_1 &\ge -\frac{28}{13}, & \bar{p}_1 &\le \frac{59}{15}, & (i=1, j=2) \\ \bar{p}_1 - \bar{p}_2 &\ge -\frac{31}{5}, & \bar{p}_2 &\le \frac{32}{15}, & (i=2, j=1) \\ \bar{p}_1 - \frac{2}{3}\bar{p}_3 &\ge -\frac{47}{9}, & \bar{p}_3 &\le \frac{61}{12}, & (i=3, j=1) \\ \bar{p}_3 - \frac{7}{13}\bar{p}_1 &\ge -\frac{49}{13}, & & & (i=1, j=3) \\ \bar{p}_3 - \bar{p}_2 &\ge -\frac{22}{5}, & & & (i=2, j=3) \\ \bar{p}_2 - \frac{1}{3}\bar{p}_3 &\ge -\frac{22}{9}, & & & (i=3, j=2) \\ & & \bar{p}_2 &\ge -\frac{88}{15}, & (i=4, j=2) \\ & & \bar{p}_3 &\ge -\frac{154}{15}, & (i=4, j=3) \\ & & \bar{p}_5 - \bar{p}_4 &\ge -9, & (i=4, j=5) \\ & & \bar{p}_4 - \bar{p}_5 &\ge -31. & (i=5, j=4). \end{aligned}$$

only in positions $d-1$ and d. Further, $v_j^{(ij)} = N^{(ij)}_{(d-1)(d-1)} = 1$, and $v_i^{(ij)} = N^{(ij)}_{(d-1)d} \leq 0$ as it is an off-diagonal entry.

Encoding length. We need to show that the encoding size of $\bar{M}$ and $\bar{b}$ are polynomially bounded in the encoding size of M and b. This easily follows since the constraints are obtained by Gaussian elimination; we refer to [15] for strong polynomiality of Gaussian elimination.

Containment of P_M. We show that every $\bar{p}$ satisfying $M\bar{p} \leq \gamma$ is also feasible to (8). For $i \in T_1$, the constraint $M_i\bar{p} \leq \gamma_i$ is identical to the i-th constraint in P_M.

For $i \in T_2$, let $ij \in H$ be the edge used in the definition of $\kappa^{(i)}$. According to Claim 6.1, the row $N_d^{(ij)}$ has a single nonzero entry $N^{(ij)}_{dd} = 1$. Lemma 6.2 guarantees that the coefficient matrix $Y^{(ij)}$ is nonnegative. Therefore, the constraint $\bar{p}_i \leq \kappa^{(i)}$ can be obtained as a nonnegative combination of the constraint set $M\bar{p} \leq \gamma$, by multiplying $M_h\bar{p} \leq \gamma_h$ for $h \in D_i$ by $Y^{(ij)}_{d\sigma^{(ij)}(h)}$.

The validity of the constraints ${v^{(ij)}}^\top \bar{p} \geq \delta^{(ij)}$ follows similarly. Recall from Lemma 6.1 that $M\bar{p} \geq -\lambda$ is valid for $\bar{p}$. The constraint ${v^{(ij)}}^\top \bar{p} \geq \delta^{(ij)}$ is obtained by taking a nonnegative combination of the inequalities $M\bar{p} \geq -\lambda$ combining $M_h\bar{p} \geq -\lambda_h$ for $h \in D_i$ with the nonnegative coefficient $Y^{(ij)}_{(d-1)\sigma^{(ij)}(h)}$. Hence, all these inequalities are valid for $\bar{p}$.

Approximate reverse containment. We next show that if $\bar{p}$ is feasible to (8), then $\bar{p}$ is feasible to B^2P_M, that is, $M\bar{p} \leq B^2\gamma$. Clearly, for $i \in T_1$, $M_i\bar{p} \leq \gamma_i \leq B^2\gamma_i$. The more difficult part is to show the validity of $M_i\bar{p} \leq B^2\gamma_i$ for $i \in T_2 \cup T_3$.

For $i \in T_3$, we show that the constraints

$$ {v^{(ij)}}^\top \bar{p} \geq \delta^{(ij)} \quad \forall j : ij \in H $$

together imply $M_i\bar{p} \leq B^2\gamma_i$. For $i \in T_2$, we will also make use of the additional constraint $\bar{p}_i \leq \kappa^{(i)}$ to derive $M_i\bar{p} \leq B^2\gamma_i$. The following technical lemma will be needed.

Lemma 6.3. *Consider any $i \in T_2 \cup T_3$.*

(i) There is a unique vector $q^{(i)} \in \mathbb{R}^t_+$ such that $M_\ell q^{(i)} = 0$ for all $\ell \in D_i \setminus \{i\}$, $q_i^{(i)} = 1$, and $q_\ell^{(i)} = 0$ for $\ell \in [t] \setminus D_i$.

(ii) For any $ij \in H$, $v_j^{(ij)} = 1$ and $v_i^{(ij)} = -q_j^{(i)}$.

(iii) If $i \in T_2$, then there exists a vector $r^{(i)} \in \mathbb{R}^t_+$ with $Mr^{(i)} \leq \gamma$, $M_\ell r^{(i)} = \gamma_\ell$ for all $\ell \in D_i$, and $r_i^{(i)} = \kappa^{(i)}$.

(iv) If $i \in T_3$, then there exists a vector $r^{(i)} \in \mathbb{R}^t_+$ with $Mr^{(i)} \leq \gamma$, and $M_\ell r^{(i)} = \gamma_\ell$ for all $\ell \in D_i \setminus \{i\}$. □

Assume now that $i \in T_2 \cup T_3$. We will show that $M_i\bar{p} \leq B^2\gamma_i$. Let $q := q^{(i)}$ as in Lemma 6.3(i). By part (ii) of the same lemma, and substituting the definition (6) of $\delta^{(ij)}$, the constraints can be written as

$$ \bar{p}_j - q_j\bar{p}_i \geq -Y^{(ij)}_{d-1}\lambda^{(ij)} \quad \forall j : ij \in H. $$

Note that $\lambda_\ell^{(ij)} \leq (B-1)\gamma_\ell^{(ij)}$ for all $\ell \in [d]$ by the definition of B, and $Y^{(ij)}_{d-1} \geq 0$. Therefore, these constraints imply

$$ \bar{p}_j - q_j\bar{p}_i \geq -(B-1)Y^{(ij)}_{d-1}\gamma^{(ij)} \quad \forall j : ij \in H. $$

Recall that $M_{ij} < 0$ if and only if $ij \in H$. Let us multiply the inequality for every $j \neq i$ by $M_{ij} \leq 0$, and add up these inequalities. We obtain

$$ \sum_{j:ij\in H} M_{ij}\bar{p}_j - \left(\sum_{j:ij\in H} M_{ij}q_j\right)\bar{p}_i \leq -(B-1)\sum_{j:ij\in H} M_{ij}Y^{(ij)}_{d-1}\gamma^{(ij)}. \tag{9} $$

For the rest of the proof, we distinguish the cases $i \in T_3$ and $i \in T_2$.

Case $i \in T_3$. Since $M_hq = 0$ for all $h \in D_i \setminus \{i\}$, Claim 6.1(iii) implies $M_iq = 0$. Substituting $q_i = 1$, we see that $M_{ii} = -\sum_{j:ij\in H} M_{ij}q_j$. With $\eta_i := -\sum_{j:ij\in H} M_{ij}Y^{(ij)}_{d-1}\gamma^{(ij)}$, (9) can be written as

$$ M_{ii}\bar{p}_i + \sum_{j:ij\in H} M_{ij}\bar{p}_j \leq (B-1)\eta_i. \tag{10} $$

The left hand side is $M_i\bar{p}$. We next show $\eta_i \leq \lambda_i$, which together with $\lambda_i \leq (B-1)\gamma_i$ yields $M_i\bar{p} \leq (B-1)^2\gamma_i$.

To see $\eta_i \leq \lambda_i$, we make use of the vector $r = r^{(i)}$ as in Lemma 6.3(iv). Let $\hat{r}$ denote the permutation $\sigma^{(ij)}$ applied to r. Since $M_\ell^{(ij)}\hat{r} = \gamma_\ell$ is valid for all $\ell \in [d-1]$, we have $N^{(ij)}_{d-1}\hat{r} = Y^{(ij)}_{d-1}\gamma^{(ij)}$. This can be written as

$$ r_j - q_jr_i = Y^{(ij)}_{d-1}\gamma^{(ij)}. $$

Summing up these equalities after multiplying the j-th one by $M_{ij} < 0$, we see as above that

$$ M_ir = -\eta_i. $$

Since $Mr \leq \gamma$, from Lemma 6.1 we have $-\eta_i = M_ir \geq -\lambda_i$, and therefore $\eta_i \leq \lambda_i$ as needed.

Case $i \in T_2$. The coefficient of $\bar{p}_i$ in (9) equals $M_{ii} - M_iq$. In contrast with the previous case, M_iq is not necessarily 0. We claim that $M_iq \geq 0$. To see this, note that $\sum_{h\in D_i} M_hq \geq 0$ since $\sum_{h\in D_i} M_h \geq 0$ from the Z_+-property and $q \geq 0$; further, $M_hq = 0$ for $h \in D_i \setminus \{i\}$. Let us further add to (9) M_iq times the inequality $\bar{p}_i \leq \kappa^{(i)}$. Thus, we obtain

$$ M_i\bar{p} \leq (B-1)\eta_i + \kappa^{(i)}M_iq. \tag{11} $$

Let $r = r^{(i)}$ as in Lemma (6.3)(iii). As for $i \in T_3$, we must have

$$ r_j - q_jr_i = Y^{(ij)}_{d-1}\gamma^{(ij)} \quad \forall ij \in H. $$

Adding up these equations multiplied by M_{ij}, and further adding M_iq times the equality $r_i = \kappa^{(i)}$, we obtain

$$ M_ir = -\eta_i + \kappa^{(i)}M_iq. $$

On the other hand, we know that $M_ir = \gamma_i$. Thus, $\gamma_i = -\eta_i + \kappa^{(i)}M_iq$. Consequently, from (11) we obtain

$$ M_i\bar{p} \leq B\kappa^{(i)}M_iq. \tag{12} $$

The next claim completes the proof of $M_i\bar{p} \leq B^2\gamma_i$.

Claim 6.2. $\kappa^{(i)}M_iq \leq B\gamma_i$. □

□

7 CONCLUSIONS

We have given a strongly polynomial algorithm for computing an equilibrium in linear exchange markets. We use the Duan-Mehlhorn algorithm as a subroutine in a framework that repeatedly identifies revealed arcs. Before each iteration of this subroutine, we use another method to find a good starting solution for the current set of revealed arcs. The best solution here corresponds to the optimal solution of a linear program. Whereas no strongly polynomial algorithm is known for an LP of this form, we presented a strongly polynomial approximation by constructing a second LP.

It could be worth exploring whether this approach extends further. An immediate question is to see if one can use such an approach to obtain a ε-approximation of the LP in strongly polynomial time for every $\varepsilon > 0$. Further, such a method could be potentially useful for a broader class of LPs; a natural candidate would be systems of the form $A^\top x \leq c$ for a pre-Leontief matrix A [7], a class where a pointwise maximal solution exists, but no strongly polynomial algorithm is known.

Our approach was specific to the market equilibrium problem. The method of identifying revealed arc sets originates from [40]. This result was applicable not only for the linear Fisher market model, but more generally, for minimum-cost flow problems with separable convex objectives satisfying certain assumptions. It would be desirable to extend the current approach to a broader class of convex programs that include the formulation in [9].

Acknowledgements. The authors are grateful to Kurt Mehlhorn, Vijay Vazirani, and Richard Cole for many interesting discussions on this problem.

REFERENCES

[1] I Adler and S Cosares. 1991. A strongly polynomial algorithm for a special class of linear programs. *Operations Research* 39, 6 (1991), 955–960.
[2] Kenneth J Arrow and Gerard Debreu. 1954. Existence of an equilibrium for a competitive economy. *Econometrica: Journal of the Econometric Society* (1954), 265–290.
[3] William Brainard and Herbert Scarf. 2000. How to Compute Equilibrium Prices in 1891. *Cowles Foundation Discussion Paper* 1270 (2000).
[4] Bruno Codenotti, Sriram Pemmaraju, and Kasturi Varadarajan. 2004. The computation of market equilibria. *ACM SIGACT News* 35, 4 (2004), 23–37.
[5] Edith Cohen and Nimrod Megiddo. 1994. Improved algorithms for linear inequalities with two variables per inequality. *SIAM J. Comput.* 23, 6 (1994), 1313–1347.
[6] Bernard Cornet. 1989. *Linear exchange economies.* Technical Report. Cahier Eco-Math, Université de Paris.
[7] Richard W Cottle and Arthur F Veinott. 1972. Polyhedral sets having a least element. *Mathematical Programming* 3, 1 (1972), 238–249.
[8] Omar Darwish and Kurt Mehlhorn. 2016. Improved balanced flow computation using parametric flow. *Inf. Process. Lett.* 116, 9 (2016), 560–563.
[9] Nikhil R Devanur, Jugal Garg, and László A Végh. 2016. A rational convex program for linear Arrow-Debreu markets. *ACM Transactions on Economics and Computation (TEAC)* 5, 1 (2016), 6.
[10] Nikhil R Devanur, Christos H Papadimitriou, Amin Saberi, and Vijay V Vazirani. 2008. Market equilibrium via a primal–dual algorithm for a convex program. *Journal of the ACM (JACM)* 55, 5 (2008), 22.
[11] Nikhil R. Devanur and Vijay V. Vazirani. 2003. An Improved Approximation Scheme for Computing Arrow-Debreu Prices for the Linear Case. In *Proceedings of FSTTCS*. 149–155.
[12] Ran Duan, Jugal Garg, and Kurt Mehlhorn. 2016. An Improved Combinatorial Polynomial Algorithm for the Linear Arrow-Debreu Market. In *Proceedings of SODA*. 90–106.
[13] Ran Duan and Kurt Mehlhorn. 2015. A combinatorial polynomial algorithm for the linear Arrow–Debreu market. *Information and Computation* 243 (2015), 112–132.
[14] B. Curtis Eaves. 1976. A Finite Algorithm for the Linear Exchange Model. *Journal of Mathematical Economics* 3 (1976), 197–203.
[15] Jack Edmonds. 1967. Systems of distinct representatives and linear algebra. *Journal of Research of the National Bureau of Standards B* 71 (1967), 241–245.
[16] Edmund Eisenberg and David Gale. 1959. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics* 30, 1 (1959), 165–168.
[17] David Gale. 1976. The linear exchange model. *Journal of Mathematical Economics* 3, 2 (1976), 205–209.
[18] Jugal Garg, Ruta Mehta, Vijay V. Vazirani, and Sadra Yazdanbod. 2017. Settling the complexity of Leontief and PLC exchange markets under exact and approximate equilibria. In *Proceedings of STOC*. 890–901.
[19] Rahul Garg and Sanjiv Kapoor. 2006. Auction Algorithms for Market Equilibrium. *Mathematics of Operations Research* 31, 4 (2006), 714–729.
[20] Mehdi Ghiyasvand and James B. Orlin. 2012. A Simple Approximation Algorithm for Computing Arrow-Debreu Prices. *Operations Research* 60, 5 (2012), 1245–1248.
[21] Gagan Goel and Vijay Vazirani. 2011. A Perfect Price Discrimination Market Model with Production, and a Rational Convex Program for It. *Mathematics of Operations Research* 36, 4 (2011), 762–782.
[22] A. V. Goldberg and R. E. Tarjan. 1989. Finding minimum-cost circulations by canceling negative cycles. *Journal of the ACM (JACM)* 36, 4 (1989), 873–886.
[23] Martin Grötschel, László Lovász, and Alexander Schrijver. 1988. *Geometric algorithms and combinatorial optimization.* Springer Verlag.
[24] Kamal Jain. 2007. A polynomial time algorithm for computing an Arrow-Debreu market equilibrium for linear utilities. *SIAM J. Comput.* 37, 1 (2007), 303–318.
[25] Kamal Jain, Mohammad Mahdian, and Amin Saberi. 2003. Approximating Market Equilibria. In *Proceedings of APPROX-RANDOM*. 98–108.
[26] Naoyuki Kamiyama. 2019. A note on balanced flows in equality networks. *Inf. Process. Lett.* 145 (2019), 74–76.
[27] C. E. Lemke. 1965. Bimatrix equilibrium points and mathematical programming. *Management Science* 11, 7 (1965), 681–689.
[28] N. Megiddo. 1983. Towards a genuinely polynomial algorithm for linear programming. *SIAM J. Comput.* 12, 2 (1983), 347–353.
[29] E I Nenakov and M E Primak. 1983. One algorithm for finding solutions of the Arrow-Debreu model. *Kibernetica* 3 (1983), 127–128.
[30] N. Olver and L. A. Végh. 2017. A Simpler and Faster Strongly Polynomial Algorithm for Generalized Flow Maximization. In *Proceedings of STOC*. ACM, 100–111.
[31] J. B. Orlin. 1993. A faster strongly polynomial minimum cost flow algorithm. *Operations Research* 41, 2 (1993), 338–350.
[32] James B Orlin. 2010. Improved algorithms for computing Fisher's market clearing prices. In *Proceedings of STOC*. ACM, 291–300.
[33] Vadim I. Shmyrev. 2009. An algorithm for finding equilibrium in the linear exchange model with fixed budgets. *Journal of Applied and Industrial Mathematics* 3, 4 (2009), 505–518.
[34] É. Tardos. 1985. A strongly polynomial minimum cost circulation algorithm. *Combinatorica* 5, 3 (1985), 247–255.
[35] É. Tardos. 1986. A strongly polynomial algorithm to solve combinatorial linear programs. *Operations Research* (1986), 250–256.
[36] Stephen A Vavasis and Yinyu Ye. 1996. A primal-dual interior point method whose running time depends only on the constraint matrix. *Mathematical Programming* 74, 1 (1996), 79–120.
[37] Vijay Vazirani. 2010. Spending Constraint Utilities with Applications to the Adwords Market. *Mathematics of Operations Research* 35, 2 (2010), 458–478.
[38] Vijay V Vazirani. 2012. The notion of a rational convex program, and an algorithm for the Arrow-Debreu Nash bargaining game. *Journal of the ACM (JACM)* 59, 2 (2012), 7.
[39] László A Végh. 2013. Concave generalized flows with applications to market equilibria. *Mathematics of Operations Research* 39, 2 (2013), 573–596.
[40] László A. Végh. 2016. A strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. *SIAM J. Comput.* 45, 5 (2016), 1729–1761.
[41] L. A. Végh. 2017. A Strongly Polynomial Algorithm for Generalized Flow Maximization. *Mathematics of Operations Research* 42, 2 (2017), 179–211.
[42] Léon Walras. 1874. Eléments d'économie politique pure, ou théorie de la richesse sociale (in French). English translation: Elements of pure economics; or, the theory of social wealth. American Economic Association and the Royal Economic Society, 1954.
[43] Yinyu Ye. 2005. A new complexity result on solving the Markov decision problem. *Mathematics of Operations Research* 30, 3 (2005), 733–749.
[44] Yinyu Ye. 2008. A path to the Arrow–Debreu competitive market equilibrium. *Mathematical Programming* 111, 1-2 (2008), 315–348.
[45] Yinyu Ye. 2011. The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate. *Mathematics of Operations Research* 36, 4 (2011), 593–603.