

# Market Equilibrium via a Primal–Dual Algorithm for a Convex Program

NIKHIL R. DEVANUR

*Georgia Institute of Technology*

CHRISTOS H. PAPADIMITRIOU

*University of California at Berkeley, Berkeley, California*

AMIN SABERI

*Stanford University, Stanford, California*

AND

VIJAY V. VAZIRANI

*Georgia Institute of Technology*

**Abstract.** We give the first polynomial time algorithm for exactly computing an equilibrium for the linear utilities case of the market model defined by Fisher. Our algorithm uses the primal–dual paradigm in the enhanced setting of KKT conditions and convex programs. We pinpoint the added difficulty raised by this setting and the manner in which our algorithm circumvents it.

**Categories and Subject Descriptors:** F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumeric Algorithms and Problems

**General Terms:** Algorithms, Economics

**Additional Key Words and Phrases:** Market equilibria, primal–dual algorithms

---

A. Saberi was supported by National Science Foundation (NSF) Grant CCF-0546889 and a gift from Google.

V. Vazirani was supported by NSF Grant CCF-0728640.

Authors' addresses: N. R. Devanur and V. V. Vazirani, Computer Science & Systems, Georgia Institute of Technology, Atlanta, GA 30332, e-mail: {vijay.vazirani,Nikhil}@cc.gatech.edu; C. H. Papadimitriou, Computer Science Division, University of California at Berkeley, Soda Hall 689, EECS Department, Berkeley, CA 94720, e-mail: chrisots@cs.berkeley.edu; A. Saberi, Terman Engineering Building, Room 317, Stanford, CA 94305, e-mail: saberi@stanford.edu.

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or direct commercial advantage and that copies show this notice on the first page or initial screen of a display along with the full citation. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers, to redistribute to lists, or to use any component of this work in other works requires prior specific permission and/or a fee. Permissions may be requested from Publications Dept., ACM, Inc., 2 Penn Plaza, Suite 701, New York, NY 10121-0701 USA, fax +1 (212) 869-0481, or [permissions@acm.org](mailto:permissions@acm.org).  
© 2008 ACM 0004-5411/2008/10-ART22 \$5.00 DOI 10.1145/1411509.1411512 <http://doi.acm.org/10.1145/1411509.1411512>

**ACM Reference Format:**

Devanur, N. R., Papadimitriou, C. H., Saberi, A., and Vazirani, V. V. 2008. Market equilibrium via a primal-dual algorithm for a convex program. *J. ACM* 55, 5, Article 22 (October 2008), 18 pages. DOI = 10.1145/1411509.1411512 <http://doi.acm.org/10.1145/1411509.1411512>

## 1. Introduction

We present the first polynomial time algorithm for the linear version of an old problem, defined in 1891 by Irving Fisher [Brainard and Scarf 2000]: Consider a market consisting of buyers and divisible goods. The money possessed by buyers and the amount of each good are specified. Also specified are utility functions of buyers, which are assumed to be linear (Fisher's original definition assumed concave utility functions). The problem is to compute prices for the goods such that even if each buyer is made optimally happy, relative to these prices, there is no deficiency or surplus of any of the goods, that is, the market clears.

Fisher's work was done contemporarily and independently of Walras' pioneering work [Walras 1874] on modeling market equilibria. Through the ensuing years, the study of market equilibria occupied center stage within mathematical economics. Its crowning achievement came with the work of Arrow and Debreu [1954] which established the existence of equilibrium prices in a very general setting, through the use of Kakutani's fixed point theorem.

Fisher's and Arrow and Debreu's market equilibrium models are considered the two most fundamental models within mathematical economics. The latter can be seen as a generalization of the former—it consists of agents who come to the market with initial endowments of goods, and at any set prices, want to sell all their goods and buy optimal bundles at these prices. The problem again is to find market clearing prices.

**1.1. PRIOR ALGORITHMIC RESULTS.** General equilibrium theory has long enjoyed the status of the crown jewel within mathematical economics. However, other than a few isolated results, it is essentially a non-algorithmic theory. Among its algorithmic results are Scarf's work on approximately computing fixed points [Scarf 1973] and some very impressive nonlinear convex programs that capture, as their optimal solutions, equilibrium allocations for the case of linear utility functions: the Eisenberg–Gale program for Fisher's model [Eisenberg and Gale 1959] and the Nenakov–Primak program [Nenakov and Primak 1983] for the Arrow–Debreu model; see Codenotti et al. [2004] for a survey of these works. The ellipsoid algorithm can be used to find approximate solutions to these programs. Subsequent to our work, exact algorithms for solving these programs have also been found (see Section 1.3).

Within theoretical computer science, the question of polynomial time solvability of equilibria, market equilibria as well as Nash equilibria, was first considered by Megiddo and Papadimitriou [1991]. Subsequently, a complexity-theoretic framework for establishing evidence of intractability for such issues was given by Papadimitriou [1994]. Our work follows Deng et al. [2002] who gave polynomial time algorithms for the Arrow–Debreu model for the cases that the utility functions are linear and either the number of goods or the number of agents is bounded.

In retrospect, all necessary ingredients for obtaining an exact, though not combinatorial, polynomial time algorithm for Fisher's linear case were present even before

our work. The fact that equilibrium prices and allocations for this case are rational numbers that can be expressed using only polynomially many bits, which is shown in this paper as a consequence of our combinatorial algorithm (see Lemma 7.1), can also be shown directly using the Eisenberg–Gale program (for a proof, see for example Theorem 5.1 in Vazirani [2007]). This fact, together with the work of Newman and Primak [1992] yields the desired algorithm.

1.2. ALGORITHMIC CONTRIBUTIONS OF OUR WORK. For the linear case of Fisher’s model, it is natural to seek an algorithmic answer in the theory of linear programming. However, there does not seem to be any natural linear programming formulation for this problem. Instead, a remarkable nonlinear convex program, given by Eisenberg and Gale [1959], captures, as its optimal solutions, equilibrium allocations for this case.

Our algorithm uses the primal-dual paradigm—not in its usual setting of LP-duality theory, but in the enhanced setting of convex programming and the Karush-Kuhn-Tucker (KKT) conditions. After introducing some definitions and notation, in Section 3, we pinpoint in Section 4 the added difficulty of working in this enhanced setting and the manner in which our algorithm circumvents this difficulty.

Our algorithm is not strongly polynomial. Indeed, obtaining such an algorithm is an important open question remaining. It will require a qualitatively different approach, perhaps one which satisfies KKT conditions in discrete steps, as is the rule with all other primal-dual algorithms known today (as pointed out in Section 4, we start by suitably relaxing the KKT conditions and our algorithm satisfies these conditions continuously rather than in discrete steps).

The usual advantages of combinatorial algorithms apply to our work as well, namely such algorithms are easier to adapt, certainly heuristically and sometimes even formally, to related problems and fine tuned for use in special circumstances; Section 1.3 offers specific examples.

Our first exposition [Devanur et al. 2002] of this algorithm suffered from a major shortcoming. Although the high level algorithmic idea given in Devanur et al. [2002] was the same as the one given in the current version (see Sections 5 and 7), the exact implementation (using the notion of “pre-emptive freezing”) contained a subtle though fatal flaw. Fixing this flaw involved introducing the notion of balanced flows, a nontrivial idea that is likely to find future applications (see Section 8).

We explain briefly the role played by this new notion. The primal variables in the Eisenberg–Gale program are allocations to buyers and the “dual” variables are Lagrangian variables corresponding to the packing constraints occurring in the program; these are interpreted as prices of goods. As is usual in primal–dual algorithms, our algorithm alternates between primal and dual update steps. Throughout the algorithm, the prices are such that buyers have surplus money left over. Each update attempts to decrease this surplus, and when it vanishes, the prices are right for the market to clear exactly.

Clearly, the number of update steps executed needs to be bounded by a polynomial. Devanur et al. [2002] attempted to do this by adjusting the high level algorithm to ensure that in each iteration, the decrease in the total surplus money is at least an inverse polynomial fraction of the total. However, despite numerous attempts, no implementation of this idea has yet been found.

The main new idea is to measure progress with respect to the  $l_2$ -norm of the vector of surplus money of buyers, rather than the  $l_1$ -norm; the latter of course

is the total surplus money. Unlike the  $l_1$ -norm, the  $l_2$ -norm of the surplus vector depends on the particular allocation chosen. The special allocation we choose is the one that minimizes the  $l_2$ -norm of the surplus vector. In turn, this allocation corresponds to a *balanced flow* in the network  $N(\mathbf{p})$  defined in Section 3.

The following observation may shed additional light. The special allocation mentioned above (and the notion of balanced flow in network  $N(\mathbf{p})$ ), has an alternative definition. Let us compare the vector of surplus money with respect to two allocations lexicographically, after sorting the vectors in decreasing order. The special allocation that minimizes the  $l_2$ -norm of the surplus vector is also the one that yields the lexicographically smallest surplus vector.

This alternative definition can be used for stating an algorithm that is identical to ours. Can a polynomial running time be established for the algorithm using the alternative definition, thereby dispensing with  $l_2$ -norm altogether? At present we see no way of doing this—our proof of the fact that guarantees progress, namely Corollary 8.6, crucially uses  $l_2$ -norm.

Another ingredient for ensuring polynomial running time is new combinatorial facts in parametric bipartite networks (see Section 6).

**1.3. SUBSEQUENT ALGORITHMIC DEVELOPMENTS.** The conference version of this article [Devanur et al. 2002] spawned off new algorithmic work along several different directions. [Jain et al. 2003; Devanur and Vazirani 2003] used this algorithm to give an approximate market clearing algorithm for the linear case of the Arrow–Debreu model. [Vazirani 2006] gave the notion of spending constraint utility functions for Fisher’s model, a polynomial time algorithm for the case of step functions and showed that these utilities are particularly expressive in Google’s AdWords market. [Devanur and Vazirani 2004] extended spending constraint utilities to the Arrow–Debreu model and established many nice properties of these utilities. Garg and Kapoor [2004] gave some very interesting approximate equilibrium algorithms for the linear case of both models using an auction based approach. These algorithms have much better running times than ours.

Another exciting development came from a simple observation in Kelly and Vazirani [2002] that Fisher’s linear case can be viewed as a special case of the resource allocation framework given by Kelly [1997] for modeling and understanding TCP congestion control. Kelly and Vazirani [2002] observed that although continuous time algorithms, not having polynomial running times, had been developed for Kelly’s problem, finding discrete time algorithms would be interesting.

Jain and Vazirani [2008] explored this issue by defining the class of *Eisenberg–Gale markets*—markets whose equilibrium allocations can be captured via convex programs having the same form as the Eisenberg–Gale program—and studying algorithmic solvability and game-theoretic properties of these markets. This line of work was extended further in Chakrabarty et al. [2006]—they study algorithmic solvability of Eisenberg–Gale markets with two agents, thereby positively settling some of the open problems of Jain and Vazirani [2008].

The above stated works provide combinatorial algorithms for computing equilibria. An advantage of this approach is illustrated in Vazirani [2006]. As stated above, some of the basic properties of equilibria for linear Fisher markets can be easily established using the Eisenberg–Gale convex program (e.g., see Theorem 5.1 in Vazirani [2007]). Interestingly enough, all these properties also hold for the generalization to spending constraint utility functions. They are established in

Vazirani [2006] via a generalization of our combinatorial algorithm to this case; at present we do not know of a convex program that captures equilibrium allocations for this case.

Yet another application of the combinatorial structure of markets was to determining continuity properties of equilibrium prices and allocations for linear Fisher markets and some of its generalizations [Megiddo and Vazirani 2007; Vazirani and Wang 2008] (the proofs in Megiddo and Vazirani [2007] for linear Fisher markets are based on the Eisenberg–Gale program; however, it was the combinatorial structure that made these properties apparent).

Progress has also been made on obtaining convex programs that capture equilibria for various utility functions for the two fundamental market models, see Codenotti et al. [2004], as well as on the question of finding exact equilibria by solving convex programs using either the ellipsoid method [Jain 2004] or interior point algorithms [Ye 2008].

## 2. Fisher’s Linear Case and the Eisenberg–Gale Convex Program

Fisher’s linear case is the following. Consider a market consisting of a set  $B$  of buyers and a set  $A$  of divisible goods. Assume  $|A| = n$  and  $|B| = n'$ . We are given for each buyer  $i$  the amount  $e_i$  of money she possesses and for each good  $j$  the amount  $b_j$  of this good. In addition, we are given the utility functions of the buyers. Our critical assumption is that these functions are linear. Let  $u_{ij}$  denote the utility derived by  $i$  on obtaining a unit amount of good  $j$ . Given prices  $p_1, \dots, p_n$  of the goods, it is easy to compute baskets of goods (there could be many) that make buyer  $i$  happiest. We will say that  $p_1, \dots, p_n$  are *market clearing* prices if after each buyer is assigned such a basket, there is no surplus or deficiency of any of the goods. Our problem is to compute such prices in polynomial time.

First, observe that, without loss of generality, we may assume that each  $b_j$  is unit—by scaling the  $u_{ij}$ ’s appropriately. The  $u_{ij}$ ’s and  $e_i$ ’s are in general rational; by scaling appropriately, they may be assumed to be integral. Now, it turns out that there is a market clearing price iff each good has a potential buyer (one who derives nonzero utility from this good). Moreover, if there is a solution, it is unique [Gale 1960; Eisenberg and Gale 1959]. We assume that we are in the latter case.

The Eisenberg–Gale convex program is the following:

$$\begin{array}{ll}
 \text{maximize} & \sum_{i=1}^{n'} e_i \log u_i \\
 \text{subject to} & u_i = \sum_{j=1}^n u_{ij} x_{ij} \quad \forall i \in B \\
 & \sum_{i=1}^{n'} x_{ij} \leq 1 \quad \forall j \in A \\
 & x_{ij} \geq 0 \quad \forall i \in B, \forall j \in A,
 \end{array} \tag{1}$$

where  $x_{ij}$  is the amount of good  $j$  allocated to buyer  $i$ . The price of good  $j$  in the equilibrium is equal to the optimum value of the Lagrangean variable corresponding to the second constraint in the above program.

By the KKT conditions, optimal solutions to  $x_{ij}$ ’s and  $p_j$ ’s must satisfy the following conditions:

- (1)  $\forall j \in A : p_j \geq 0$ .
- (2)  $\forall j \in A : p_j > 0 \Rightarrow \sum_{i \in A} x_{ij} = 1$ .

$$(3) \forall i \in B, \forall j \in A : \frac{u_{ij}}{p_j} \leq \frac{\sum_{j \in A} u_{ij} x_{ij}}{e_i}.$$

$$(4) \forall i \in B, \forall j \in A : x_{ij} > 0 \Rightarrow \frac{u_{ij}}{p_j} = \frac{\sum_{j \in A} u_{ij} x_{ij}}{e_i}.$$

Via these conditions, it is easy to see that an optimal solution to the Eisenberg and Gale program gives equilibrium allocations for Fisher's linear case, and the corresponding dual variables give equilibrium prices of goods. The Eisenberg and Gale program also helps prove, in a very simple manner, basic properties of the set of equilibria: Equilibrium exists under certain conditions (the mild conditions stated above), the set of equilibria is convex, equilibrium utilities and prices are unique, and if the program has all rational entries then equilibrium allocations and prices are also rational.

### 3. High Level Idea of the Algorithm

Let  $\mathbf{p} = (p_1, \dots, p_n)$  denote a vector of prices. If at these prices buyer  $i$  is given good  $j$ , she derives  $u_{ij}/p_j$  amount of utility per unit amount of money spent. Clearly, she will be happiest with goods that maximize this ratio. Define her *bang per buck* to be  $\alpha_i = \max_j \{u_{ij}/p_j\}$ ; clearly, for each  $i \in B, j \in A, \alpha_i \geq u_{ij}/p_j$ . If there are several goods maximizing this ratio, she is equally happy with any combination of these goods. This motivates defining the following bipartite graph,  $G$ . Its bipartition is  $(A, B)$  and for  $i \in B, j \in A$   $(i, j)$  is an edge in  $G$  iff  $\alpha_i = u_{ij}/p_j$ . We will call this graph the *equality subgraph* and its edges the *equality edges*.

Any goods sold along the edges of the equality subgraph will make buyers happiest, relative to the current prices. Computing the largest amount of goods that can be sold in this manner, without exceeding the budgets of buyers or the amount of goods available (assumed unit for each good), can be accomplished by computing max-flow in the following network: Direct edges of  $G$  from  $A$  to  $B$  and assign a capacity of infinity to all these edges. Introduce source vertex  $s$  and a directed edge from  $s$  to each vertex  $j \in A$  with a capacity of  $p_j$ . Introduce sink vertex  $t$  and a directed edge from each vertex  $i \in B$  to  $t$  with a capacity of  $e_i$ . The network is clearly a function of the current prices  $\mathbf{p}$  and will be denoted  $N(\mathbf{p})$ . The algorithm maintains the following throughout:

*Invariant.* The prices  $\mathbf{p}$  are such that  $(s, A \cup B \cup t)$  is a min-cut in  $N(\mathbf{p})$ .

The Invariant ensures that, at current prices, all goods can be sold. The only eventuality is that buyers may be left with surplus money. The algorithm raises prices systematically, always maintaining the Invariant, so that surplus money with buyers keeps decreasing. When the surplus vanishes, market clearing prices have been attained. This is equivalent to the condition that  $(s \cup A \cup B, t)$  is also a min-cut in  $N(\mathbf{p})$ , that is, max-flow in  $N(\mathbf{p})$  equals the total amount of money possessed by the buyers.

*Remark 3.1.* With this setup, we can define our market equilibrium problem as an optimization problem: find prices  $\mathbf{p}$  under which network  $N(\mathbf{p})$  supports maximum flow.

### 4. The Enhanced Setting and How to Deal with It

We will use the notation set up in the previous section to pinpoint the difficulties involved in solving the Eisenberg–Gale program combinatorially and the manner in which these difficulties are circumvented.

As is well known, the primal–dual schema has yielded combinatorial algorithms for obtaining, either optimal or near-optimal, integral solutions to numerous linear programming relaxations. Other than one exception, namely Edmonds’ algorithm for maximum weight matching in general graphs [Edmonds 1965], all other algorithms raise dual variables via a greedy process.

The disadvantage of a greedy dual growth process is obvious—the fact that a raised dual is “bad”, in the sense that it “obstructs” other duals that could have led to a larger overall dual solution, may become clear only later in the run of the algorithm. In view of this, the issue of using more sophisticated dual growth processes has received a lot of attention, especially in the context of approximation algorithms. Indeed, Edmonds’ algorithm is able to find an optimal dual for matching by a process that increases and decreases duals.

The problem with such a process is that it will make primal objects go tight and loose and the number of such reversals will have to be upper bounded in the running time analysis. The impeccable combinatorial structure of matching supports such an accounting and in fact this leads to a strongly polynomial algorithm. However, thus far, all attempts at making such a scheme work out for other problems have failed.

The fundamental difference between complimentary slackness conditions for linear programs and KKT conditions for nonlinear convex programs is that whereas the former do not involve both primal and dual variables simultaneously in an equality constraint (obtained by assuming that one of the variables takes a non-zero value), the latter do.

Now, our dual growth process is greedy—prices of goods are never decreased. Yet, because of the more complex nature of KKT conditions, edges in the equality subgraph appear and disappear as the algorithm proceeds. Hence, we are forced to carry out the difficult accounting process alluded to above for bounding the running time.

We next point out which KKT conditions our algorithm enforces and which ones it relaxes, as well as the exact mechanism by which it satisfies the latter. Throughout our algorithm, we enforce the first two conditions listed in Section 2. As mentioned in Section 3, at any point in the algorithm, via a max-flow in the network  $N(\mathbf{p})$ , all goods can be sold; however, buyers may have surplus money left over. With respect to a balanced flow in network  $N(\mathbf{p})$  (see Section 8 for a definition of such a flow), let  $m_i$  be the money spent by buyer  $i$ . Thus, buyer  $i$ ’s surplus money is  $\gamma_i = e_i - m_i$ . We will relax the third and fourth KKT conditions to the following:

$$\begin{aligned} & \text{—} \forall i \in B, \forall j \in A : \frac{u_{ij}}{p_j} \leq \frac{\sum_{j \in A} u_{ij} x_{ij}}{m_i}. \\ & \text{—} \forall i \in B, \forall j \in A : x_{ij} > 0 \Rightarrow \frac{u_{ij}}{p_j} = \frac{\sum_{j \in A} u_{ij} x_{ij}}{m_i}. \end{aligned}$$

We consider the following potential function:

$$\Phi = \gamma_1^2 + \gamma_2^2 + \cdots + \gamma_n^2,$$

and we give a process by which this potential function decreases by an inverse polynomial fraction in polynomial time (in each phase, as detailed in Lemma 8.10). When  $\Phi$  drops all the way to zero, all KKT conditions are exactly satisfied.

There is a marked difference between the way we satisfy KKT conditions and the way primal-dual algorithms for LP’s do. The latter satisfy complimentary conditions

in *discrete steps*, that is, in each iteration, the algorithm satisfies at least one new condition. So, if each iteration can be implemented in strongly polynomial time, the entire algorithm has a similar running time. On the other hand, we satisfy KKT conditions *continuously*—as the algorithm proceeds, the KKT conditions corresponding to each buyer get satisfied to a greater extent.

Next, let us consider the special case of Fisher’s market in which all  $u_{ij}$ ’s are 0/1. There is no known LP that captures equilibrium allocations in this case as well and the only recourse seems to be the special case of the Eisenberg–Gale program in which all  $u_{ij}$ ’s are restricted to 0/1. Although this is a nonlinear convex program, it is easy to derive a strongly polynomial combinatorial algorithm for solving it. Of course, in this case as well, the KKT conditions involve both primal and dual variables simultaneously. However, the setting is so easy that this difficulty never manifests itself. The algorithm satisfies KKT conditions in discrete steps, much the same way that a primal-dual algorithm for solving an LP does.

In retrospect, Megiddo [1974] (and perhaps other papers in the past) have implicitly given strongly polynomial primal-dual algorithms for solving nonlinear convex programs. Some very recent papers have also done so explicitly, for example, Jain and Vazirani [2008]. However, the problems considered in these papers are so simple (e.g., multicommodity flow in which there is only one source), that the enhanced difficulty of satisfying KKT conditions is mitigated and the primal-dual algorithms are not much different than those for solving LP’s.

### 5. A Simple Algorithm

In this section, we give a simple algorithm, without the use of balanced flows. Although we do not know how to establish polynomial running time for it, it still provides valuable insights into the problem and shows clearly exactly where the idea of balanced flows fits in. We pick up the exposition from the end of Section 3.

How do we pick prices so the Invariant holds at the start of the algorithm? The following two conditions guarantee this:

- The initial prices are low enough prices that each buyer can afford all the goods. Fixing prices at  $1/n$  suffices, since the goods together cost one unit and all  $e_i$ ’s are integral.
- Each good  $j$  has an interested buyer, has an edge incident at it in the equality subgraph. Compute  $\alpha_i$  for each buyer  $i$  at the prices fixed in the previous step and compute the equality subgraph. If good  $j$  has no edge incident, reduce its price to

$$p_j = \max_i \left\{ \frac{u_{ij}}{\alpha_i} \right\}.$$

The iterative improvement steps follow the spirit of the primal-dual schema: The “primal” variables are the flows in the edges of  $N(\mathbf{p})$  and the “dual” variables are the current prices. The current flow suggests how to improve the prices and vice-versa.

For  $S \subseteq B$ , define its money  $m(S) = \sum_{i \in B} e_i$ . With respect to prices  $\mathbf{p}$ , for set  $S \subseteq A$ , define its money  $m(S) = \sum_{j \in A} p_j$ ; the context will clarify the price vector  $\mathbf{p}$ . For  $S \subseteq A$ , define its *neighborhood in  $N(\mathbf{p})$*

$$\Gamma(S) = \{j \in B \mid \exists i \in S \text{ with } (i, j) \in G\}.$$

By the assumption that each good has a potential buyer,  $\Gamma(A) = B$ . The Invariant can now be more clearly stated.

LEMMA 5.1. *For given prices  $\mathbf{p}$  network,  $N(\mathbf{p})$  satisfies the Invariant iff*

$$\forall S \subseteq A : m(S) \leq m(\Gamma(S)).$$

PROOF. The forward direction is trivial, since under max-flow (of value  $m(A)$ ) every set  $S \subseteq A$  must be sending  $m(S)$  amount of flow to its neighborhood.

Let's prove the reverse direction. Assume  $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$  is a min-cut in  $N(\mathbf{p})$ , with  $A_1, A_2 \subseteq A$  and  $B_1, B_2 \subseteq B$ . The capacity of this cut is  $m(A_2) + m(B_1)$ . Now,  $\Gamma(A_1) \subseteq B_1$ , since otherwise the cut will have infinite capacity. Moving  $A_1$  and  $\Gamma(A_1)$  to the  $t$  side also results in a cut. By the condition stated in the Lemma, the capacity of this cut is no larger than the previous one. Therefore, this is also a min-cut in  $N(\mathbf{p})$ . Hence, the Invariant holds.  $\square$

If the Invariant holds, it is easy to see that there is a unique maximal set  $S \subseteq A$  such that  $m(S) = m(\Gamma(S))$ . Say that this is the *tight set* with respect to prices  $\mathbf{p}$ . Clearly the prices of goods in the tight set cannot be increased without violating the Invariant. Hence our algorithm only raises prices of goods in the *active subgraph* consisting of the bipartition  $(A - S, B - \Gamma(S))$ . We will say that the algorithm *freezes* the subgraph  $(S, \Gamma(S))$ . Observe that in general, the bipartite graph  $(S, \Gamma(S))$  may consist of several connected components (with respect to equality edges). Let these be  $(S_1, T_1), \dots, (S_k, T_k)$ .

Clearly, as soon as prices of goods in  $A - S$  are raised, edges  $(i, j)$  with  $i \in \Gamma(S)$  and  $j \in (A - S)$  will not remain in the equality subgraph anymore. We will assume that these edges are dropped. Before proceeding further, we must be sure that these changes do not violate the Invariant. This follows from:

LEMMA 5.2. *If the Invariant holds and  $S \subseteq A$  is the tight set, then each good  $j \in (A - S)$  has an edge, in the equality subgraph, to some buyer  $i \in (B - \Gamma(S))$ .*

PROOF. Since the Invariant holds,  $j \in (A - S)$  must have an equality graph edge incident at it. If all such edges are incidents at buyers in  $\Gamma(S)$ , then  $\Gamma(S \cup j) = \Gamma(S)$  and therefore

$$m(S \cup j) > m(S) = m(\Gamma(S)) = m(\Gamma(S \cup j)).$$

This contradicts the fact that the Invariant holds.  $\square$

We would like to raise prices of goods in the active subgraph in such a way that the equality edges in it are retained. This is ensured by multiplying prices of all these goods by  $x$  and gradually increasing  $x$ , starting with  $x = 1$ . To see that this has the desired effect, observe that  $(i, j)$  and  $(i, l)$  are both equality edges iff

$$\frac{p_j}{p_l} = \frac{u_{ij}}{u_{il}}.$$

The algorithm raises  $x$ , starting with  $x = 1$ , until one of the following happens:

- Event 1.* A set  $R \neq \emptyset$  goes tight in the active subgraph.
- Event 2.* An edge  $(i, j)$  with  $i \in (B - \Gamma(S))$  and  $j \in S$  becomes an equality edge. (Observe that as prices of goods in  $A - S$  are increasing, goods in  $S$  are becoming more and more desirable to buyers in  $B - \Gamma(S)$ , which is the reason for this event.)

If Event 1 happens, we redefine the active subgraph to be  $(A - (S \cup R), B - \Gamma(S \cup R))$ , and proceed with the next iteration. Suppose Event 2 happens and that  $j \in S_l$ . Because of the new equality edge  $(i, j)$ ,  $\Gamma(S_l) = T_l \cup i$ . Therefore,  $S_l$  is not tight anymore. Hence, we move  $(S_l, T_l)$  into the active subgraph.

To complete the algorithm, we simply need to compute the smallest values of  $x$  at which Event 1 and Event 2 happen, and consider only the smaller of these. For Event 2, this is straightforward. Below we give an algorithm for Event 1.

## 6. Finding Tight Sets

Let  $\mathbf{p}$  denote the current price vector (i.e., at  $x = 1$ ). We first present a lemma that describes how the min-cut changes in  $N(x \cdot \mathbf{p})$  as  $x$  increases. Throughout this section, we will use the function  $m$  to denote money with respect to prices  $\mathbf{p}$ . Without loss of generality, assume that with respect to prices  $\mathbf{p}$  the tight set in  $G$  is empty (since we can always restrict attention to the active subgraph, for the purposes of finding the next tight set). Define

$$x^* = \min_{\emptyset \neq S \subseteq A} \frac{m(\Gamma(S))}{m(S)},$$

the value of  $x$  at which a nonempty set goes tight. Let  $S^*$  denote the tight set at prices  $x^* \cdot \mathbf{p}$ . If  $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$  is a cut in the network, we will assume that  $A_1, A_2 \subseteq A$  and  $B_1, B_2 \subseteq B$ .

LEMMA 6.1. *With respect to prices  $x \cdot \mathbf{p}$ :*

- if  $x \leq x^*$ , then  $(s, A \cup B \cup t)$  is a min-cut.
- if  $x > x^*$ , then  $(s, A \cup B \cup t)$  is not a min-cut. Moreover, if  $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$  is a min-cut in  $N(x \cdot \mathbf{p})$  then  $S^* \subseteq A_1$ .

PROOF. Suppose  $x \leq x^*$ . By definition of  $x^*$ ,

$$\forall S \subseteq A : x \cdot m(S) \leq m(\Gamma(S)).$$

Therefore, by Lemma 5.1, with respect to prices  $x \cdot \mathbf{p}$ , the Invariant holds. Hence,  $(s, A \cup B \cup t)$  is a min-cut.

Next suppose that  $x > x^*$ . Since  $x \cdot m(S^*) > x^* \cdot m(S^*) = m(\Gamma(S^*))$ , with respect to prices  $x \cdot \mathbf{p}$ , the cut  $(s \cup S^* \cup \Gamma(S^*), t)$  has strictly smaller capacity than the cut  $(s \cup A \cup B, t)$ . Therefore the latter cannot be a min-cut.

Let  $S^* \cap A_2 = S_2$  and  $S^* - S_2 = S_1$ . Suppose  $S_2 \neq \emptyset$ . Clearly,  $\Gamma(S_1) \subseteq B_1$  (otherwise the cut will have infinite capacity). If  $m(\Gamma(S_2) \cap B_2) < x \cdot m(S_2)$ , then by moving  $S_2$  and  $\Gamma(S_2)$  to the  $s$  side, we can get a smaller cut, contradicting the minimality of the cut picked. In particular, if  $S_2 = S^*$ , then this inequality must hold, leading to a contradiction. Hence,  $S_1 \neq \emptyset$ . Furthermore,

$$m(\Gamma(S_2) \cap B_2) \geq x \cdot m(S_2) > x^* m(S_2).$$

On the other hand,

$$m(\Gamma(S_2) \cap B_2) + m(\Gamma(S_1)) \leq x^* (m(S_2) + m(S_1)).$$

The two imply that

$$\frac{m(\Gamma(S_1))}{m(S_1)} < x^*,$$

contradicting the definition of  $x^*$ . Hence,  $S_2 = \emptyset$  and  $S^* \subseteq A_1$ .  $\square$

*Remark 6.2.* A more complete statement for the first part of Lemma 6.1, which is not essential for our purposes, is: If  $x < x^*$ , then  $(s, A \cup B \cup t)$  is the unique min-cut in  $N(x \cdot \mathbf{p})$ . If  $x = x^*$ , then the min-cuts are obtained by moving a bunch of connected components of  $(S^*, \Gamma(S^*))$  to the  $s$ -side of the cut  $(s, A \cup B \cup t)$ .

LEMMA 6.3. *Let  $x = m(B)/m(A)$  and suppose that  $x > x^*$ . If  $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$  be a min-cut in  $N(x \cdot \mathbf{p})$ , then  $A_1$  must be a proper subset of  $A$ .*

PROOF. If  $A_1 = A$ , then  $B_1 = B$  (otherwise this cut has  $\infty$  capacity), and  $(s \cup A \cup B, t)$  is a min-cut. But for the chosen value of  $x$ , this cut has the same capacity as  $(s, A \cup B \cup t)$ . Since  $x > x^*$ , the latter is not a min-cut by Lemma 6.1. Hence,  $A_1$  is a proper subset of  $A$ .  $\square$

LEMMA 6.4.  *$x^*$  and  $S^*$  can be found using  $n$  max-flow computations.*

PROOF. Let  $x = m(B)/m(A)$ . Clearly,  $x \geq x^*$ . If  $(s, A \cup B \cup t)$  is a min-cut in  $N(x \cdot \mathbf{p})$ , then by Lemma 6.1  $x^* = x$ . If so,  $S^* = A$ .

Otherwise, let  $(s \cup A_1 \cup B_1, A_2 \cup B_2 \cup t)$  be a min-cut in  $N(x \cdot \mathbf{p})$ . By Lemmas 6.1 and 6.3,  $S^* \subseteq A_1 \subset A$ . Therefore, it is sufficient to recurse on the smaller graph  $(A_1, \Gamma(A_1))$ .  $\square$

## 7. Termination with Market Clearing Prices

Let  $M$  be the total money possessed by the buyers and let  $f$  be the max-flow computed in network  $N(\mathbf{p})$  at current prices  $\mathbf{p}$ . Thus  $M - f$  is the *surplus money* with the buyers. Let us partition the running of the algorithm into *phases*, each phase terminates with the occurrence of Event 1. Each phase is partitioned into *iterations* which conclude with a new edge entering the equality subgraph. We will show that  $f$  must be proportional to the number of phases executed so far, hence showing that the surplus must vanish in bounded time.

Let  $U = \max_{i \in B, j \in A} \{u_{ij}\}$  and let  $\Delta = nU^n$ .

LEMMA 7.1. *At the termination of a phase, the prices of goods in the newly tight set must be rational numbers with denominator  $\leq \Delta$ .*

PROOF. Let  $S$  be the newly tight set and consider the equality subgraph induced on the bipartition  $(S, \Gamma(S))$ . Assume without loss of generality that this graph is connected (otherwise we prove the lemma for each connected component of this graph). Let  $j \in S$ . Pick a subgraph in which  $j$  can reach all other vertices  $j' \in S$ . Clearly, at most  $2|S| \leq 2n$  edges suffice. If  $j$  reaches  $j'$  with a path of length  $2l$ , then  $p_{j'} = ap_j/b$  where  $a$  and  $b$  are products of  $l$  utility parameters ( $u_{ik}$ 's) each. Since alternate edges of this path contribute to  $a$  and  $b$ , we can partition the  $u_{ik}$ 's in this subgraph into two sets such that  $a$  and  $b$  use  $u_{ik}$ 's from distinct sets. These considerations lead easily to showing that  $m(S) = p_j c/d$  where  $c \leq \Delta$ . Now,

$$p_j = m(\Gamma(S))d/c,$$

hence proving the lemma.  $\square$

LEMMA 7.2. *Each phase consists of at most  $n$  iterations.*

PROOF. Each iteration brings goods from the tight set to the active subgraph. Clearly, this cannot happen more than  $n$  times without a set going tight.  $\square$

LEMMA 7.3. *Consider two phases  $P$  and  $P'$ , not necessarily consecutive, such that good  $j$  lies in the newly tight sets at the end of  $P$  as well as  $P'$ . Then the increase in the price of  $j$ , going from  $P$  to  $P'$ , is  $\geq 1/\Delta^2$ .*

PROOF. Let the prices of  $j$  at the end of  $P$  and  $P'$  be  $p/q$  and  $r/s$ , respectively. Clearly,  $r/s > p/q$ . By Lemma 7.1,  $q \leq \Delta$  and  $r \leq \Delta$ . Therefore, the increase in price of  $j$ ,

$$\frac{r}{s} - \frac{p}{q} \geq \frac{1}{\Delta^2}. \quad \square$$

LEMMA 7.4. *After  $k$  phases,  $f \geq k/\Delta^2$ .*

PROOF. Consider phase  $P$  and let  $j$  be a good that lies in the newly tight set at the end of this phase. Let  $P'$  be the last phase, earlier than  $P$ , such that  $j$  lies in the newly tight set at the end of  $P'$  as well. If there is no such phase (because  $P$  is the first phase in which  $j$  appears in a tight set), then let  $P'$  be the start of the algorithm. Let us charge to  $P$  the entire increase in the price of  $j$ , going from  $P'$  to  $P$  (even though this increase takes place gradually over all the intermediate phases). By Lemma 7.3, this is  $\geq 1/\Delta^2$ . In this manner, each phase can be charged  $1/\Delta^2$ . The lemma follows.  $\square$

COROLLARY 7.5. *Algorithm 1 terminates with market clearing prices in at most  $M\Delta^2$  phases, and executes  $O(Mn^2\Delta^2)$  max-flow computations.*

Remark 7.6. The upper bound given above is quite loose, for example, it is easy to shave off a factor of  $n$  by giving a tighter version of Lemma 7.2.

**Algorithm 1.** The Basic Algorithm

---

**Initialization:**

$\forall j \in A, p_j \leftarrow 1/n; \quad \forall i \in B, \alpha_i \leftarrow \min_j u_{ij}/p_j;$   
 Compute equality subgraph  $G$ ;  
 $\forall j \in A$  **if**  $\text{degree}_G(j) = 0$  **then**  $p_j \leftarrow \max_i u_{ij}/\alpha_i;$   
 Recompute  $G$ ;  
 $(F, F') \leftarrow (\emptyset, \emptyset)$  (The frozen subgraph);  $(H, H') \leftarrow (A, B)$  (The active subgraph);

**While**  $H \neq \emptyset$  **do**

$x \leftarrow 1;$   
 Define  $\forall j \in H$ , price of  $j$  to be  $p_j x$ ;  
 Raise  $x$  continuously until one of two events happens:  
**if**  $S \subseteq H$  becomes tight **then**  
   Move  $(S, \Gamma(S))$  from  $(H, H')$  to  $(F, F')$ ;  
   Remove all edges from  $F'$  to  $H$ ;  
**if** an edge  $(i, j), i \in H', j \in F$  attains equality,  $\alpha_i = u_{ij}/p_j$ , **then**  
   Add  $(i, j)$  to  $G$ ;  
   Move connected component of  $j$  from  $(F, F')$  to  $(H, H')$ ;

---

### 8. Establishing Polynomial Running Time

For a given flow  $f$  in the network  $N(\mathbf{p})$ , define the *surplus* of buyer  $i$ ,  $\gamma_i(\mathbf{p}, f)$ , to be the residual capacity of the edge  $(i, t)$  with respect to  $f$ , which is equal to  $e_i$  minus the flow sent through the edge  $(i, t)$ .

In this section, we are trying to speed up Algorithm 1 by increasing the prices of goods adjacent only to “high-surplus” buyers. However, the surplus of a buyer might be different for two different maximum flows in the same graph. Therefore, we will restrict ourselves to a specific flow so that the surplus of a buyer is well defined. The following definition serves this purpose:

Define the surplus vector  $\gamma(\mathbf{p}, f) := (\gamma_1(\mathbf{p}, f), \gamma_2(\mathbf{p}, f), \dots, \gamma_n(\mathbf{p}, f))$ . Let  $\|v\|$  denote the  $l_2$  norm of vector  $v$ .

**Definition 8.1 (Balanced flow).** For any given  $\mathbf{p}$ , a maximum flow that minimizes  $\|\gamma(\mathbf{p}, f)\|$  over all choices of  $f$  is called a *balanced flow*.

If  $\|\gamma(\mathbf{p}, f)\| < \|\gamma(\mathbf{p}, f')\|$ , then we say  $f$  is *more balanced* than  $f'$ .

For a given  $\mathbf{p}$  and a flow  $f$  in  $N(\mathbf{p})$ , let  $R(\mathbf{p}, f)$  be the residual network of  $N(\mathbf{p})$  with respect to the flow  $f$ . We will give a characterization of balanced flow via  $R(\mathbf{p}, f)$

**LEMMA 8.2.** *Let  $f$  and  $f'$  be any two maximum flows in  $N(\mathbf{p})$ . If  $\gamma_i(\mathbf{p}, f') < \gamma_i(\mathbf{p}, f)$  for some  $i \in B$ , then there exist a  $j \in B$  such that  $\gamma_j(\mathbf{p}, f) < \gamma_j(\mathbf{p}, f')$  and*

- (1) *There is a path from  $j$  to  $i$  in  $R(\mathbf{p}, f) \setminus \{s, t\}$ .*
- (2) *There is a path from  $i$  to  $j$  in  $R(\mathbf{p}, f') \setminus \{s, t\}$ .*

**PROOF.** Consider the flow  $f' - f$ . It defines a feasible circulation in the network  $R(\mathbf{p}, f)$ . Since  $\gamma_i(\mathbf{p}, f') < \gamma_i(\mathbf{p}, f)$ , there is a positive flow along the edge  $(i, t)$  in  $f' - f$ . By following this flow all the way back to  $t$  in the circulation, one can find a node  $j$ , such that there is a positive flow from  $t$  to  $j$  and then to  $i$  in  $f' - f$ . Since both flows are maximum,  $s$  is an isolated vertex in  $f' - f$  and this flow does not go through  $s$ . Now,  $f' - f$  is a valid flow in  $R(\mathbf{p}, f)$  and therefore there exists a path from  $j$  to  $i$  in  $R(\mathbf{p}, f) \setminus \{s, t\}$ . Moreover, having a positive flow from  $t$  to  $j$  implies that  $\gamma_j(\mathbf{p}, f) < \gamma_j(\mathbf{p}, f')$ . A similar argument shows that there is also a path from  $i$  to  $j$  in  $R(\mathbf{p}, f') \setminus \{s, t\}$ .  $\square$

**LEMMA 8.3.** *If  $a \geq b_i \geq 0, i = 1, 2, \dots, n$  and  $\delta \geq \sum_{j=1}^n \delta_j$  where  $\delta, \delta_j \geq 0, j = 1, 2, \dots, n$ , then  $\|(a, b_1, b_2, \dots, b_n)\|^2 \leq \|(a + \delta, b_1 - \delta_1, b_2 - \delta_2, \dots, b_n - \delta_n)\|^2 - \delta^2$ .*

**PROOF**

$$(a + \delta)^2 + \sum_{i=1}^n (b_i - \delta_i)^2 - a^2 - \sum_{i=1}^n b_i^2 \geq \delta^2 + 2a(\delta - \sum_{i=1}^n \delta_i) \geq 0. \quad \square$$

The following property characterizes all balanced flows. It defines the flows for which there is no path from a low-surplus node to a high-surplus node in the residual network.

**Property 1.** There is no path from node  $i \in B$  to node  $j \in B$  in  $R(\mathbf{p}, f)$  if surplus of  $i$  is more than surplus of  $j$  in  $N(\mathbf{p}, f)$ .

**Algorithm 2.**


---

**Initialization:**  
 $\forall j \in A, p_j \leftarrow 1/n$ ;  
 $\forall i \in B, \alpha_i \leftarrow \min_j u_{ij}/p_j$ ;  
 Define  $G(A, B, E)$  with  $(i, j) \in E$  iff  $\alpha_i = u_{ij}/p_j$ ;  
 $\forall j \in A$  **if**  $\text{degree}_G(j) = 0$  **then**  $p_j \leftarrow \max_i u_{ij}/\alpha_i$ ;  
 Recompute  $G$ ;  $\delta = M$ ;  
**repeat**  
   Compute a balanced flow  $f$  in  $G$ ;  
   Define  $\delta$  to be the maximum surplus in  $B$ ;  
   Define  $H$  to be the set of buyers with surplus  $\delta$  ;  
   **repeat**  
     Let  $H'$  be the set of neighbors of  $H$  in  $A$ ;  
     Remove all edges from  $B \setminus H$  to  $H'$ ;  
      $x \leftarrow 1$ ; Define  $\forall j \in H'$ , price of  $j$  to be  $p_j x$ ;  
     Raise  $x$  continuously until one of the two events happens:  
     **Event 1:** an edge  $(i, j), i \in H, j \in A \setminus H'$  attains equality,  $\alpha_i = u_{ij}/p_j$ ;  
       Add  $(i, j)$  to  $G$ ;  
       Recompute  $f$ ;  
       In the residual network corresponding to  $f$  in  $G$ , define  $I$  to be the set of buyers that  
       can reach  $H$ ;  $H \leftarrow H \cup I$ ;  
     **Event 2:**  $S \subseteq H$  becomes tight;  
   **until** some subset  $S \subseteq H$  is tight;  
**Until**  $A$  is tight;

---

**THEOREM 8.4.** *A maximum-flow  $f$  is balanced iff it has Property 1.*

**PROOF.** Suppose  $f$  is a balanced flow. Let  $\gamma_i(\mathbf{p}, f) > \gamma_j(\mathbf{p}, f)$  for some  $i$  and  $j$ , and suppose for the sake of contradiction, that there is a path from  $j$  to  $i$  in  $R(\mathbf{p}, f) \setminus \{s, t\}$ . Then one can send a circulation of positive value along  $t \rightarrow j \rightarrow i \rightarrow t$  in  $R(\mathbf{p}, f)$ , decreasing  $\gamma_i$  and increasing  $\gamma_j$ . From Lemma 8.3, the resulting flow is more balanced than  $f$ , contradicting the fact that  $f$  is a balanced flow.

To prove the other direction, suppose that  $f$  is not a balanced maximum flow. Let  $f'$  be a balanced flow. Since  $\|\gamma(\mathbf{p}, f')\| < \|\gamma(\mathbf{p}, f)\|$ , there exists  $i \in B$  such that  $\gamma_i(\mathbf{p}, f') < \gamma_i(\mathbf{p}, f)$ .

By Lemma 8.2, there exists  $j \in B$  such that  $\gamma_j(\mathbf{p}, f) < \gamma_j(\mathbf{p}, f')$  and there is a path from  $j$  to  $i$  in  $R(\mathbf{p}, f) \setminus \{s, t\}$ . Since  $f$  has Property 1,  $\gamma_i(\mathbf{p}, f) \leq \gamma_j(\mathbf{p}, f)$ . The above three inequalities imply  $\gamma_i(\mathbf{p}, f') < \gamma_j(\mathbf{p}, f)$ . But again by Lemma 8.2, there is a path from  $i$  to  $j$  in  $R(\mathbf{p}, f') \setminus \{s, t\}$  so  $f'$  doesn't have Property 1. This contradicts the assumption that  $f'$  is a balanced flow by what we proved in the first half the theorem.  $\square$

The following lemma provides our main tool for proving polynomial running time of Algorithm 2. We will use it to prove an upper bound on the  $l_2$ -norm of the surplus vector of buyers at the end of every phase.

**LEMMA 8.5.** *If  $f$  and  $f^*$  are respectively a feasible and a balanced flow in  $N(\mathbf{p})$  and for some  $i \in B$  and  $\delta > 0$   $\gamma_i(f) = \gamma_i(f^*) + \delta$ , then there is a flow  $f'$  and for some  $k$  there is a set of vertices  $i_1, i_2, \dots, i_k$  and values  $\delta_1, \delta_2, \dots, \delta_k$  such that*

$$\begin{aligned} &-\sum_{i=1}^k \delta_i \leq \delta \\ &-\gamma_i(f') = \gamma_i(f) - \delta \end{aligned}$$

$$\begin{aligned} -\gamma_i(f') &= \gamma_i(f) + \delta_i \\ -\gamma_i(f') &\geq \gamma_i(f'). \end{aligned}$$

PROOF. Consider  $f^* - f$  in  $R(\mathbf{p}, f)$  and in a similar fashion as in Lemma 8.2 follow the incoming flow of node  $i$  until you reach  $s$  or the node  $i$  itself. Let  $f'$  be the flow augmented from  $f$  by sending back the flow through all these circulations and paths. We will have  $\gamma_i(f') = \gamma_i(f) - \delta$  and for a set of vertices  $i_1, i_2, \dots, i_k$  and values  $\delta_1, \delta_2, \dots, \delta_k$  such that  $\sum_{l=1}^k \delta_l \leq \delta$ , we have  $\gamma_{i_l}(f') = \gamma_{i_l}(f) + \delta_l$ . Moreover, since  $f^*$  is balanced,  $\gamma_i(f') = \gamma_i(f^*) \geq \gamma_i(f^*) \geq \gamma_i(f')$ .  $\square$

COROLLARY 8.6.  $\|\gamma(\mathbf{p}, f)\|^2 \geq \|\gamma(\mathbf{p}, f^*)\|^2 + \delta^2$ .

PROOF. By Lemma 8.3,  $\|\gamma(f, \mathbf{p})\|^2 \geq \|\gamma(f', \mathbf{p})\|^2 + \delta^2$  and since  $f^*$  is a balanced flow in  $N(\mathbf{p})$ ,  $\|\gamma(f', \mathbf{p})\|^2 \geq \|\gamma(f^*, \mathbf{p})\|^2$ .  $\square$

COROLLARY 8.7. *For any given  $\mathbf{p}$ , all balanced flows in  $N(\mathbf{p})$  have the same surplus vector.*

As a result, one can define the surplus vector for a given price as  $\gamma(\mathbf{p}) := \gamma(\mathbf{p}, f)$  where  $f$  is the balanced flow in  $N(\mathbf{p})$ . This vector can be found by computing a balanced flow in the equality subgraph in the following way:

COROLLARY 8.8. *For a given price vector  $\mathbf{p}$  the balanced flow can be computed by at most  $n$  max-flow computation.*

PROOF. We will use the divide and conquer method. Let  $m_{\text{avg}} := (\sum_{i=1}^{n'} e_i - \sum_{j=1}^n p_j)/n'$ . Compute the maximum flow in the equality subgraph after subtracting  $m_{\text{avg}}$  from the capacity of each edge adjacent  $t$ . Let  $(S, T)$  be the maximal min-cut in that network.  $s \in S, t \in T$ . If  $A \subset S$ , then the current maximum flow is balanced. Otherwise, let  $N_1$  and  $N_2$  be the networks induced by  $T \cup \{s\}$  and  $S \cup \{t\}$  respectively. Claim that the union of balanced flows in  $N_1$  and  $N_2$  is a balanced flow in  $N$ .

In order to prove the claim, it is enough (from Theorem 8.4) to show that the surplus of all buyers in  $N_1$  (in a balanced flow) is at least  $m_{\text{avg}}$  and that of all buyers in  $N_2$  is at most  $m_{\text{avg}}$ . We will prove the former; the proof of the latter is similar. Let  $L$  be the set of all buyers in  $N_1$  with the lowest surplus, say  $s$ . Suppose  $s < m_{\text{avg}}$ . Let  $K$  be the set of goods reachable by  $L$  in the residual network of  $N_1$  w.r.t a balanced flow. By Theorem 8.4, no other buyers are reachable from  $L$  in this network. Hence,  $\Gamma_{N_1}(K) \subseteq L$ . Since the surplus of all buyers in  $L$  is  $s$ ,  $m(K) = m(L) - s|L| > m(L) - m_{\text{avg}}|L|$ . This is a contradiction to the fact that  $(S, T)$  was a min-cut.  $\square$

In a set of feasible vectors, a vector  $v$  is called *min-max fair* iff for every feasible vector  $u$  and an index  $i$  such that  $u_i < v_i$  there is a  $j$  for which  $u_j < v_j$  and  $v_j < v_i$ . Similarly,  $v$  is *max-min fair* iff  $u_i > v_i$  implies that there is a  $j$  for which  $u_j < v_j$  and  $v_j > v_i$ .

*Remark.* The surplus vector of a balanced flow is both min-max and max-min fair.

8.1. THE POLYNOMIAL-TIME ALGORITHM. The main idea of Algorithm 2 is that it tries to reduce  $\|\gamma(\mathbf{p}, f)\|$  in every phase. Intuitively, this goal is achieved by finding a set of high-surplus buyers in the balanced flow and increasing the

prices of goods in which they are interested. If a subset becomes tight as a result of this increase, we have reduced  $\|\gamma(\mathbf{p}, f)\|$  because the surplus of a formerly high-surplus buyer is dropped to zero. The other event that can happen is that a new edge is added to the equality subgraph. In that case, this edge will help us to make the surplus vector more balanced: we can reduce the surplus of high-surplus buyers and increase the surplus of low-surplus ones. This operation will result in the reduction of  $\|\gamma(\mathbf{p}, f)\|$ .

The algorithm starts with finding a price vector that does not violate the invariant. The rest of the algorithm is partitioned into *phases*. In each phase, we have an active graph  $(H, H')$  with  $H \subset B$  and  $H' \subset A$  and we increase the prices of goods in  $H'$  like Algorithm 1. Let  $\delta$  be the maximum surplus in  $B$ . The subset  $H$  is initially the set of buyers whose surplus is equal to  $\delta$ .  $H'$  is the set of goods adjacent to buyers in  $H$ .

Each phase is divided into *iterations*. In each iteration, we increase the prices of goods in  $H'$  until either a new edge joins the equality subgraph or a subset becomes tight. If a new edge is added to the equality subgraph, we recompute the balanced flow  $f$ . Then, we add to  $H$  all vertices that can reach a member of  $H$  in  $R(\mathbf{p}, f) \setminus \{s, t\}$ . If a subset becomes tight as a result of increase of the prices, then the phase terminates.

Consider a phase in the execution of Algorithm 2. Define  $\mathbf{p}_i$  and  $H_i$  to be the price vector and the set of nodes in  $H$  after executing the  $i$ 'th iteration in that phase. Let  $H_0$  denote the set of nodes in  $H$  before the first iteration.

**LEMMA 8.9.** *The number of iterations executed in a phase is at most  $n$ . Moreover, in every phase, there is an iteration in which surplus of at least one of the vertices is reduced by at least  $\frac{\delta}{n}$ .*

**PROOF.** Let  $k$  denote the number of iterations in the phase. Every time an edge is added to the equality subgraph,  $|H'|$  is increased by at least one. Therefore,  $k$  is at most  $n$ .

Define  $\delta_i = \min_{j \in H_i} (\gamma_j(\mathbf{p}_i))$ , for  $0 \leq i \leq k$ .  $\delta_0 = \delta$  and the phase ends when the surplus of one buyer in  $H$  becomes zero so  $\delta_k = 0$ . So there is an iteration  $t$  in which  $\delta_t - \delta_{t-1} \geq \frac{\delta}{n}$ .

Consider the residual network corresponding to the balanced flow computed at iteration  $t$ . In that network, every vertex in  $H_t \setminus H_{t-1}$  can reach a vertex in  $H_{t-1}$  and therefore, by Theorem 8.4, its surplus is greater than or equal to the surplus of that vertex. This means that minimum surplus  $\delta_t$  is achieved by a vertex  $i$  in  $H_{t-1}$ . Hence, the surplus of vertex  $i$  is decreased by at least  $\delta_{t-1} - \delta_t$  during iteration  $t$ .  $\square$

**LEMMA 8.10.** *If  $\mathbf{p}_0$  and  $\mathbf{p}^*$  are price vectors before and after a phase,  $\|\gamma(\mathbf{p}^*)\|^2 \leq \|\gamma(\mathbf{p}_0)\|^2 (1 - \frac{1}{n^2})$ .*

**PROOF.** In every iteration, we increase prices of goods in  $H$  or add new edges to the equality subgraph. Moreover, all the edges of the network that are deleted in the beginning of a phase have zero flow. Therefore, the balanced flow computed at iteration  $i$  is a feasible flow for  $N(\mathbf{p}_{i+1})$ . Therefore, by Lemma 8.6,  $\|\gamma(\mathbf{p}_0)\| \geq \|\gamma(\mathbf{p}_1)\| \geq \|\gamma(\mathbf{p}_2)\| \geq \dots \geq \|\gamma(\mathbf{p}_k)\|$ . Furthermore, by the previous lemma there is an iteration  $t$  and node  $i$  such that  $\gamma_i(\mathbf{p}_{t-1}) - \gamma_i(\mathbf{p}_t) \geq \frac{\delta}{n}$ . So we have:

$\|\gamma(\mathbf{p}_t)\|^2 \leq \|\gamma(\mathbf{p}_{t-1})\|^2 - \left(\frac{\delta}{n}\right)^2$ , which means that

$$\|\gamma(\mathbf{p}^*)\|^2 \leq \|\gamma(\mathbf{p}_t)\|^2 \leq \|\gamma(\mathbf{p}_{t-1})\|^2 - \left(\frac{\delta}{n}\right)^2 \leq \|\gamma(\mathbf{p}_0)\|^2 - \left(\frac{\delta}{n}\right)^2.$$

Now  $\|\gamma(\mathbf{p}_0)\|^2 \leq \delta^2 n$  so

$$\|\gamma(\mathbf{p}^*)\|^2 \leq \|\gamma(\mathbf{p}_0)\|^2 \left(1 - \frac{1}{n^3}\right). \quad \square$$

*Remark 8.11.* The upper bound given above is quite loose, for example, one can reduce the upper bound to  $(1 - \frac{1}{n^2})$  by considering all iterations  $t$  in which  $\delta_{t-1} - \delta_t > 0$ .

By the bound given in the above, it is easy to see that after  $O(n^2)$  phases,  $\|\gamma(p)\|^2$  is reduced to at most half of its previous value. In the beginning,  $\|\gamma(p)\|^2 \leq M^2$ . Once the value of  $\|\gamma(p)\|^2 \leq \frac{1}{\Delta^4}$ , the algorithm takes at most one more step. This is because Lemma 7.1, and consequently, Lemma 7.3 holds for Algorithm 2 as well. Hence, the number of phases is at most

$$O(n^2 \log(\Delta^4 M^2)) = O(n^2(\log n + n \log U + \log M)).$$

As noted before, the number of iterations in each phase is at most  $n$ . Each iteration requires at most  $O(n)$  max-flow computations.

Hence we get:

**THEOREM 8.12.** *Algorithm 2 executes at most*

$$O(n^4(\log n + n \log U + \log M))$$

*max-flow computations and finds market clearing prices.*

## 9. Discussion

An important question remaining is whether there is a strongly polynomial algorithm for computing equilibrium for Fisher’s linear case and solving the Eisenberg–Gale program. Another issue is whether the machinery developed in Section 8 is necessary for obtaining a polynomial time algorithm, that is, does the algorithm given in Sections 5 and 6 have a polynomial running time? If not, it would be nice to find a family of instances on which it takes super-polynomial time.

**ACKNOWLEDGMENTS.** We wish to thank Lisa Fleischer and Mohammad Mahdian for pointing out a subtle though fatal bug in the “pre-emptive freezing” part in Devanur et al. [2002], which has been corrected in this article.

## REFERENCES

- ARROW, K., AND DEBREU, G. 1954. Existence of an equilibrium for a competitive economy. *Econometrica* 22, 265–290.
- BRAINARD, W. C., AND SCARF, H. E. 2000. How to compute equilibrium prices in 1891. *Cowles Foundation Discussion Paper* 1270.
- CHAKRABARTY, D., DEVANUR, N., AND VAZIRANI, V. 2006. New results on rationality and strongly polynomial solvability in Eisenberg–Gale markets. In *Proceedings of 2nd Workshop on Internet and Network Economics*.

- CODENOTTI, B., PEMMARAJU, S., AND VARADARAJAN, K. 2004. Algorithms column: The computation of market equilibria. *ACM SIGACT News* 35, 4.
- DENG, X., PAPADIMITRIOU, C., AND SAFRA, S. 2002. On the complexity of equilibria. In *Proceedings of ACM Symposium on Theory of Computing*. ACM, New York.
- DEVANUR, N., PAPADIMITRIOU, C., SABERI, A., AND VAZIRANI, V. V. 2002. Market equilibrium via a primal-dual-type algorithm. In *Proceedings of the 43rd Annual IEEE Symposium on Foundations of Computer Science*. IEEE Computer Society Press, Los Alamitos, CA.
- DEVANUR, N. R., AND VAZIRANI, V. V. 2003. An improved approximation scheme for computing Arrow-Debreu prices for the linear case. In *Proceedings of 23rd FSTTCS*.
- DEVANUR, N. R., AND VAZIRANI, V. V. 2004. The spending constraint model for market equilibrium: Algorithmic, existence and uniqueness results. In *Proceedings of 36th Annual ACM Symposium on Theory of Computing (STOC)*. ACM, New York.
- EDMONDS, J. 1965. Maximum matching and a polyhedron with 0,1-vertices. *J. Res. NBS, Section B* 69, 125–130.
- EISENBERG, E., AND GALE, D. 1959. Consensus of subjective probabilities: The pari-mutuel method. *Ann. Math. Stat.* 30, 165–168.
- GALE, D. 1960. *Theory of Linear Economic Models*. McGraw Hill, New York.
- GARG, R., AND KAPOOR, S. 2004. Auction algorithms for market equilibrium. In *Proceedings of 36th Annual ACM Symposium on Theory of Computing (STOC)*. ACM, New York.
- JAIN, K. 2004. A polynomial time algorithm for computing the Arrow-Debreu market equilibrium for linear utilities. In *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*. IEEE Computer Society Press, Los Alamitos, CA.
- JAIN, K., MAHDIAN, M., AND SABERI, A. 2003. Approximating market equilibria. In *Proceedings of the International Workshop on Approximation Algorithms for Combinatorial Optimization Problems*.
- JAIN, K., AND VAZIRANI, V. V. 2008. Eisenberg-Gale markets: Algorithms and game-theoretic properties. *Games Econ. Behav.*
- KELLY, F. P. 1997. Charging and rate control for elastic traffic. *Europ. Trans. Telecomm.* 8, 33–37.
- KELLY, F. P., AND VAZIRANI, V. V. 2002. Rate control as a market equilibrium. Unpublished manuscript 2002. (Available at: <http://www-static.cc.gatech.edu/vazirani/KV.pdf>).
- MEGIDDO, N. 1974. Optimal flows in networks with multiple sources and sinks. *Math. Prog.* 7, 97–107.
- MEGIDDO, N., AND PAPADIMITRIOU, C. H. 1991. On total functions, existence theorems and computational complexity. *Theor. Comput. Sci.* 81, 317–324.
- MEGIDDO, N., AND VAZIRANI, V. V. 2007. Continuity properties of equilibrium prices and allocations in linear Fisher markets. In *Proceedings of the 3rd Workshop on Internet and Network Economics (WINE2007)*. (San Diego, CA, Dec. 12–14). Lecture Notes in Computer Science, vol. 4858, Springer-Verlag, New York, 362–367.
- NENAKOV, E. I., AND PRIMAK, M. E. 1983. One algorithm for finding solutions of the Arrow-Debreu model. *Kibernetika* 3, 127–128.
- NEWMAN, D., AND PRIMAK, M. 1992. Complexity of circumscribed and inscribed ellipsoid methods for solving equilibrium economical models. *Appl. Math. Comput.* 52, 223–231.
- PAPADIMITRIOU, C. H. 1994. On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. Syst. Sci.* 48, 3, 498–532.
- SCARF, H. 1973. *The Computation of Economic Equilibria (with Collaboration of T. Hansen)*. Cowles Foundation Monograph No. 24., Yale University Press, New Haven, CI
- VAZIRANI, V. V. 2006. Spending constraint utilities, with applications to the AdWords market. *Math. Oper. Res.* (Submitted). Available at <http://www.cc.gatech.edu/vijay.vazirani/spending.pdf>.
- VAZIRANI, V. V. 2007. Combinatorial algorithms for market equilibria. In *Algorithmic Game Theory*, N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Eds. Cambridge University Press, Cambridge, UK, 103–134.
- VAZIRANI, V. V., AND WANG, L. 2008. Continuity properties of equilibria in some Fisher and Arrow-Debreu market models. Available online at *The Electronic Colloquium on Computational Complexity (ECCC)*, Report TR08-047.
- WALRAS, L. 1874. *Éléments d'économie politique pure ou théorie de la richesse sociale (Elements of Pure Economics, or the theory of social wealth)*. Lausanne, Paris. (1899, 4th ed.; 1926, rev ed., 1954, Engl. transl.).
- YE, Y. 2008. A path to the Arrow-Debreu competitive market equilibrium. *Math. Prog.* 111, 315–348.

RECEIVED JULY 2005; REVISED APRIL 2006; ACCEPTED JULY 2008