

Market Equilibrium via a Primal-Dual-Type Algorithm

Nikhil R. Devanur*, Christos H. Papadimitriou[†], Amin Saberi*, Vijay V. Vazirani*

Abstract

Although the study of market equilibria has occupied center stage within Mathematical Economics for over a century, polynomial time algorithms for such questions have so far evaded researchers. We provide the first such algorithm for the linear version of a problem defined by Irving Fisher in 1891. Our algorithm is modeled after Kuhn’s primal-dual algorithm for bipartite matching.

1 Introduction

We present the first polynomial time algorithm for the linear version of an old problem, first defined in 1891 by Irving Fisher [3]: 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 statement 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, i.e. the market clears.

Besides defining the problem, Fisher considered the issue of computability of equilibrium prices and built a remarkable hydraulic apparatus for this purpose (see [3] for a description)¹. Fisher’s work was done contemporarily and independently of Walras’ pioneering work [16] on modeling market equilibria. Through the ensuing years, the study of market equilibria has occupied center stage within Mathematical Economics. Its crowning achievement came with the work of Arrow and Debreu [2] which established the existence of equilibrium prices in a very general setting, through the use of Kakutani’s fixed point theorem. The First Welfare Theorem, showing Pareto optimality of allo-

cations obtained at equilibrium prices, provides important social justification for this theory.

The highly non-constructive nature of Arrow and Debreu’s proof naturally raised questions of efficient computability of equilibrium prices. Despite impressive progress on this issue, e.g., Scarf’s work [13], which has been useful in many applications [4], polynomial time algorithms have evaded researchers, even for the case of linear utility functions. The latter question was raised recently in [5], which gives polynomial time algorithms if the number of goods or agents is bounded. Complexity-theoretic aspects of these issues were studied in [12].

For the case of linear utilities, it is natural to seek an algorithmic answer in the theory of linear programming. However, there does not seem to be any linear programming formulation for this problem. The main contribution of this paper is to point out that a suitable adaptation of the primal-dual schema yields a solution to Fisher’s problem.

Our algorithm is modeled after Kuhn’s primal-dual algorithm for the bipartite matching problem [11]. At the heart of the primal-dual schema lies the following powerful paradigm: the algorithm starts with trivial solutions to the primal and dual LP’s corresponding to the given problem and alternately improves these solutions until a termination criterion is met (see [15] for a detailed discussion); the current primal suggests how to improve the dual, and vice versa. We identify two processes: the “primal process” updates the amount of each good sold to each buyer and the “dual process” updates prices of goods. Throughout the algorithm the prices are such that buyers have surplus money left over. Each update decreases this surplus, and when it vanishes, the prices are right for the market to clear exactly. Our proof of correctness involves new combinatorial facts: understanding how the min-cut changes in a network derived from a bipartite graph as the capacities of its source edges are increased.

Prior to this work, [8] had used the above-stated paradigm, of two processes making improvements relative to each other, outside of the setting of linear programming. This naturally raises the question of whether there is a formal mathematical framework in which such “primal-dual-type” algorithms can be set. The analogous setting for primal-dual algorithms is of course the theory of LP-duality.

*College of Computing, Georgia Institute of Technology. This work was done while visiting U.C. Berkeley and ICSI Berkeley. Email: {nikhil, saberi, vazirani}@cc.gatech.edu.

[†]Computer Science Department, U. C. Berkeley. Email: christos@cs.berkeley.edu.

¹In a sense, Fisher was not entirely off mark – our algorithm reduces the problem to flow computations!

2 Problem

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 w.l.o.g. 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 [7, 6]. We assume that we are in the latter case.

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_i 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 $N(\mathbf{p})$, i.e., max-flow in $N(\mathbf{p})$ equals the total amount of money possessed by the buyers.

Remark 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.*

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, i.e., has an edge incident at it in the equality subgraph. Compute α_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 $m(S) = \sum_{i \in B} e_i$. W.r.t. 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 2 *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.

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* w.r.t. 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 (w.r.t. 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 3 *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.

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 build an algorithm for Event 1.

4 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 w.r.t. prices \mathbf{p} . W.l.o.g. assume that w.r.t. 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 4 *W.r.t. 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 2, w.r.t. 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 $x \cdot m(S^*) > x^* \cdot m(S^*) = m(\Gamma(S^*))$, w.r.t. 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 $S^* \subseteq A_1$.

Remark 5 *A more complete statement for the first part of Lemma 4, 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 *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 ∞ 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 4. Hence, A_1 is a proper subset of A .

Lemma 7 *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 4 $x^* = x$. 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 4 and 6, $S^* \subseteq A_1 \subset A$. Therefore, it is sufficient to recurse on the smaller graph $(A_1, \Gamma(A_1))$.

5 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 8 *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 w.l.o.g. 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.

Lemma 9 *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.

Lemma 10 *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 8, $q \leq \Delta$ and $r \leq \Delta$. Therefore the increase in price of j ,

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

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') ;

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 ;

$\epsilon = M/ne$, where e is the base of the natural logs;

while $\epsilon \geq 1/(n\Delta^2)$ **do**

$(F, F') \leftarrow (\emptyset, \emptyset)$ (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**

 Add ϵ to prices of goods in (H, H') and find the maximal min-cut;

$S' \leftarrow$ the s-side of this min-cut; Move S' from (H, H') to (F, F') ;

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 ;

 Add ϵ to prices of goods in (F, F') and find the maximal min-cut;

$S'' \leftarrow$ the t-side of this min-cut; Move S'' from F to H ;

 Remove all edges from F' to H ;

$\epsilon = \epsilon/e$;

Algorithm 2: Polynomial Time Algorithm

Lemma 11 *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 10, this is $\geq 1/\Delta^2$. In this manner, each phase can be charged $1/\Delta^2$. The lemma follows.

Corollary 12 *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 13 *The upper bound given above is quite loose, e.g., it is easy to shave off a factor of n by giving a tighter version of Lemma 9.*

6 Establishing polynomial running time

The algorithm is speeded up by ensuring that in a phase, prices increase substantially. The key idea is to preemptively freeze sets that are ‘‘almost tight’’. Let $\epsilon > 0$ be fixed. The actions taken in case of the two events are modified as follows.

Let S be the newly tight set at the end of a phase and let \mathbf{p} be the current prices. The algorithm executes the following extra step: Add ϵ to the price of each good in the active subgraph and find a min-cut in the network that maximizes the s -side (i.e., the maximal min-cut). Let $S' \subseteq A$ be the subset of A on the s -side of the min-cut. Freeze $(S', \Gamma(S'))$. Clearly, $S \subseteq S'$ and

$$m(\Gamma(S)) \leq m(S) + |S|\epsilon,$$

where prices \mathbf{p} are used for computing the function m . Hence, the surplus in this subgraph,

$$m(\Gamma(S)) - m(S) \leq |S|\epsilon.$$

The algorithm will continue the next phase with prices \mathbf{p} (i.e., ϵ was added only for the purpose of finding S'). Hence, the algorithm still maintains the Invariant.

Next suppose that a new edge (i, j) enters the equality subgraph, i.e., begins satisfying $\alpha_i = u_{ij}/p_j$. The subgraph to be unfrozen is determined as follows. Add ϵ to the prices of all goods in the frozen part and compute a maximal min-cut (with edge (i, j) added, of course). Move the part of this subgraph that is on the t side to the active subgraph.

Let \mathbf{p} be the prices at the beginning of the current phase and let S be a set of goods in the active subgraph at any point in the current phase. Say that S is ϵ -loose if $m(\Gamma(S)) \geq m(S) + \epsilon$, where the money is computed w.r.t. prices \mathbf{p} .

Lemma 14 *At any point of the current phase, every subset of goods in the active subgraph is ϵ -loose.*

Proof : Consider set S of goods in the active subgraph. Partition S into subsets depending on the time at which they were added to the active subgraph in the current phase (or if they were in the active subgraph at the start of the phase). Let S_1 be the part that was came earliest. Then,

$$m(\Gamma(S_1)) \geq m(S) + |S_1|\epsilon.$$

On the other hand, because of the Invariant, $m(\Gamma(S) - \Gamma(S_1)) \geq m(S - S_1)$. The lemma follows.

As a consequence of Lemma 14, when the current phase ends with a new tight set, flow must increase by at least ϵ .

Consider the situation when all goods are frozen. Let us call the running of the algorithm til this stage an *epoch*. Since the surplus at the start of the epoch was M , the number of phases executed in the epoch is $\leq M/\epsilon$. By the observation made above about the surplus in each frozen subgraph, the total surplus at the end of the epoch (w.r.t. current prices) is $\leq n\epsilon$. At this stage, the algorithm reduces ϵ by a constant factor, c , and executes the next epoch, starting with the current prices. It turns out that $c = e$ (base of natural logs) is a good choice.

In the first epoch, $\epsilon = M/nc$. Consider the first epoch when ϵ drops to $< 1/(n\Delta^2)$. At the end of this epoch, the surplus is $< 1/\Delta^2$. The next, and final, epoch is run with $\epsilon = 0$. By Lemma 10, during this epoch there cannot be a good j that appears in the newly tight set of two different phases. Hence, this epoch can have at most n phases and terminates with market clearing prices. It is easy to see that the number of epochs is (assuming $c = e$):

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

Hence we get

Theorem 15 *Algorithm 2 executes*

$$O(n^2(n \log U + \log Mn^2))$$

max-flow computations and finds market clearing prices.

7 Discussion

By definition, for each buyer i and good j , $\alpha_i \geq u_{ij}/p_j$. Buyer i is allowed to buy good j only if edge (i, j) is in the

equality subgraph, i.e., if the above inequality is satisfied with equality. This condition is analogous to the dual complementary slackness condition that states that if a primal variable is non-zero, then the corresponding dual constraint must be satisfied with equality.

Does Algorithm 1 have a polynomial running time, i.e., without the modifications of Section 6? We leave this as an open problem. Another question is whether Algorithm 1 has, in fact, a strongly polynomial running time. Algorithm 1 has performed well in preliminary experiments conducted on randomly chosen instances with up to 150 buyers and 50 goods.

Considering the importance of efficiently computing equilibria, especially in the context of new applications arising from the Internet, developing an algorithmic theory of market equilibria, via polynomial time exact and approximation algorithms, would be a worthwhile goal. Can Fisher's original problem, i.e., with concave utility functions, be solved in polynomial time? Such utility functions incorporate the fact that a buyer's utility for a good decreases as she gets satiated by it. In this vein, [14] defines the following generalization of the linear case and extends ideas of this paper to it: the rate at which buyer i derives utility for good j depends, in a piecewise-linear and concave manner, on the fraction of i 's budget that is spent on j . Another generalization worth studying, raised as an open question recently in [5], is the linear case of the Arrow-Debreu setting, in which there is no dichotomy between buyers and sellers. A third, and somewhat unexpected, generalization arises by viewing Kelly's charging and rate control problem for networks [9] as a market equilibrium question [10]. A different avenue for progress may arise from Adler's [1] observation that the linear version of Fisher's problem can be stated as a nonlinear complementarity problem.

8 Acknowledgments

We wish to thank Ilan Adler, Dick Karp, Noam Nisan, Herb Scarf, and Pete Veinott for valuable discussions and pointers into the literature. Thanks also to Tejas Iyer for conducting experiments.

References

- [1] I. Adler. Personal communication.
- [2] K. Arrow and G. Debreu. Existence of an equilibrium for a competitive economy. *Econometrica*, 22:265–290, 1954.
- [3] W. C. Brainard and H. E. Scarf. How to compute equilibrium prices in 1891. *Cowles Foundation Discussion Paper*, (1270), 2000.
- [4] G. Debreu. Economic theory in a mathematical mode: the Nobel Lecture, 1984. AER.
- [5] X. Deng, C. Papadimitriou, and S. Safra. On the complexity of equilibria. In *Proceedings of ACM Symposium on Theory of Computing*, 2002.
- [6] E. Eisenberg and D. Gale. Consensus of subjective probabilities: The pari-mutuel method. *Annals Of Mathematical Statistics*, 30:165–168, 1959.
- [7] D. Gale. *Theory of Linear Economic Models*. McGraw Hill, N.Y., 1960.
- [8] K. Jain and V. V. Vazirani. Equitable cost allocations via primal-dual-type algorithms. In *Proceedings of 33rd ACM Symposium on Theory of Computing*, 2002.
- [9] F. P. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997.
- [10] F. P. Kelly and V. V. Vazirani. Rate control as a market equilibrium. In preparation.
- [11] H. W. Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1995.
- [12] C. H. Papadimitriou. On the complexity of the parity argument and other inefficient proofs of existence. *JCSS*, 48(3):498–532.
- [13] H. Scarf. *The Computation of Economic Equilibria (with collaboration of T. Hansen)*. Cowles Foundation Monograph No. 24., New Haven: Yale University Press, 1973.
- [14] V. V. Vazirani. Market equilibrium under stepwise-linear utility functions. In preparation.
- [15] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag, Berlin, 2001.
- [16] L. Walras. *Éléments d'économie politique pure ou théorie de la richesse sociale (Elements of Pure Economics, or the theory of social wealth)*. Lausanne, Paris, 1874. (1899, 4th ed.; 1926, rev ed., 1954, Engl. transl.).