

Strongly Polynomial Algorithm for a Class of Minimum-Cost Flow Problems with Separable Convex Objectives

László A. Végh *

April 12, 2013

Abstract

A well-studied nonlinear extension of the minimum-cost flow problem is to minimize the objective $\sum_{ij \in E} C_{ij}(f_{ij})$ over feasible flows f , where on every arc ij of the network, C_{ij} is a convex function. We give a strongly polynomial algorithm for finding an exact optimal solution for a broad class of such problems. The most important characteristic of this class is that an optimal solution can be computed exactly provided its support.

This includes separable convex quadratic objectives and also certain market equilibria problems: Fisher's market with linear and with spending constraint utilities. We thereby give the first strongly polynomial algorithms for separable quadratic minimum-cost flows and for Fisher's market with spending constraint utilities, settling open questions posed e.g. in [16] and in [36], respectively. The running time is $O(m^4 \log m)$ for quadratic costs, $O(n^4 + n^2(m + n \log n) \log n)$ for Fisher's markets with linear utilities and $O(mn^3 + m^2(m + n \log n) \log m)$ for spending constraint utilities.

1 Introduction

Let us consider an optimization problem where the input is given by N numbers. An algorithm for such a problem is called *strongly polynomial* (see [13]), if (i) it uses only elementary arithmetic operations (addition, subtraction, multiplication, division, and comparison); (ii) the number of these operations is bounded by a polynomial of N (iii) if all numbers in the input are rational, then all numbers occurring in the computations are rational numbers of size polynomially bounded in N and the maximum size of the input numbers. Here, the size of a rational number p/q is defined as $\lceil \log_2(p + 1) \rceil + \lceil \log_2(q + 1) \rceil$.

The flow polyhedron is defined on a directed network $G = (V, E)$ by arc capacity and node demand constraints; throughout the paper, $n = |V|$ and $m = |E|$. We study the *minimum cost convex separable flow problem*: for feasible flows f , we aim to minimize $\sum_{ij \in E} C_{ij}(f_{ij})$, where on each arc $ij \in E$, C_{ij} is a differentiable convex function. We give a strongly polynomial algorithm for the case of convex quadratic functions, i.e. if $C_{ij}(\alpha) = c_{ij}\alpha^2 + d_{ij}\alpha$ with $c_{ij} \geq 0$ for every arc $ij \in E$. We also give strongly polynomial algorithms for Fisher's market with linear and with spending constraint utilities; these problems can be formulated as minimum cost convex separable flow problems, as shown respectively by Shmyrev [32] and by Devanur et al. [2].

These algorithms are obtained as special cases of an algorithm that works for the general problem setting under certain assumptions. We assume that the functions are represented by oracles (we

*Department of Management, London School of Economics. E-mail: L.Vegh@lse.ac.uk. This work was done at the College of Computing, Georgia Institute of Technology, supported by NSF Grant CCF-0914732. This is a revision of the work published in STOC'12, May 19 - 22 2012, New York, NY, USA Copyright 2012 ACM 978-1-4503-1245-5/12/05

provide the specific details later), and further we are provided two black-box subroutines. We give a strongly polynomial algorithm in the sense that it uses only basic arithmetic operations, oracle calls and subroutine calls; the total number of these operations is polynomial in n and m . We then verify our assumptions for convex quadratic objectives and the Fisher markets, and show that we can obtain strongly polynomial algorithms for these problems in the first sense.

Flows with separable convex objectives are natural convex extensions of minimum-cost flows with several applications as matrix balancing or traffic networks, see [1, Chapter 14] for further references. Polynomial-time combinatorial algorithms were given by Minoux [26] in 1986, by Hochbaum and Shantikumar [18] in 1990, and by Karzanov and McCormick [22] in 1997. The latter two approaches are able to solve even more general problems of minimizing a separable (not necessarily differentiable) convex objective over a polytope given by a matrix with a bound on its largest subdeterminant. Both approaches give polynomial, yet not strongly polynomial algorithms.

In contrast, for the same problems with linear objectives, Tardos [34, 35] gave strongly polynomial algorithms. One might wonder whether this could also be extended to the convex setting. This seems impossible for arbitrary convex objectives by the very nature of the problem: the optimal solution might be irrational, and thus the exact optimum cannot be achieved.

Beyond irrationality, the result of Hochbaum [16] shows that it is impossible to find an ε -accurate solution¹ in strongly polynomial time even for a network consisting of parallel arcs between a source and a sink node and the C_{ij} 's being polynomials of degree at least three. This is based on Renegar's result [31] showing the impossibility of finding ε -approximate roots of polynomials in strongly polynomial time.

The remaining class of polynomial objectives with hope of strongly polynomial algorithms is where every cost function is convex quadratic. If all coefficients are rational, then the existence of a rational optimal solution is guaranteed. Granot and Skorin-Kapov [12] extended Tardos's method [35] to solving separable convex quadratic optimization problems where the running time depends only on the entries of the constraint matrix and the coefficients of the quadratic terms in the objective. However, in a strongly polynomial algorithm, the running time should only depend on the matrix.

The existence of a strongly polynomial algorithm for the quadratic flow problem thus remained an important open question (mentioned e.g. in [16, 4, 17, 12, 33]). The survey paper [17] gives an overview of special cases solvable in strongly polynomial time. These include fixed number of suppliers (Cosares and Hochbaum, [4]), and series-parallel graphs (Tamir [33]). We resolve this question affirmatively, providing a strongly polynomial algorithm for the general problem in time $O(m^4 \log m)$.

There is an analogous situation for convex closure sets: [16] shows that no strongly polynomial algorithm may exist in general, but for quadratic cost functions, Hochbaum and Queyranne [15] gave a strongly polynomial algorithm.

An entirely different motivation of our study comes from the study of market equilibrium algorithms. Devanur et al. [5] developed a polynomial time combinatorial algorithm for a classical problem in economics, Fisher's market with linear utilities. This motivated a line of research to develop combinatorial algorithms for other market equilibrium problems. For a survey, see [28, Chapter 5] or [37]. All these problems are described by rational convex programs (see [37]). For the linear Fisher market problem, a strongly polynomial algorithm was given by Orlin [30].

To the extent of the author's knowledge, these rational convex problems have been considered so far as a new domain in combinatorial optimization. An explicit connection to classical flow problems was pointed out in the recent paper [38]. It turns out that the linear Fisher market, along

¹A solution x is called ε -accurate if there exists an optimal solution x^* with $\|x - x^*\|_\infty \leq \varepsilon$.

with several other problems, is captured by a concave extension of the classical generalized flow problem, solvable by a polynomial time combinatorial algorithm.

The paper [38] uses the convex programming formulation of linear Fisher markets by Eisenberg and Gale [7]. An alternative convex program for the same problem was given by Shmyrev [32]. This formulation turns out to be a convex separable minimum-cost flow problem. Consequently, equilibrium for linear Fisher market can be computed by the general algorithms [18, 22] (with a final transformation of a close enough approximate solution to an exact optimal one).

The class of convex flow problems solved in this paper also contains the formulation of Shmyrev, yielding an alternative strongly polynomial algorithm for linear Fisher market. Devanur et al. [2] gave an analogous formulation for Fisher’s market with spending constraint utilities, defined by Vazirani [36]. For this problem, we obtain the first strongly polynomial algorithm. Our running time bounds are $O(n^4 + n^2(m + n \log n) \log n)$ for linear and $O(mn^3 + m^2(m + n \log n) \log m)$ for spending constraint utilities, with m being the number of segments in the latter problem. For the linear case, Orlin [30] used the assumption $m = O(n^2)$ and achieved running time $O(n^4 \log n)$, the same as ours under this assumption. So far, no extensions of [30] are known for other market settings.

1.1 Prior work

For linear minimum-cost flows, the first polynomial time algorithm was the scaling method of Edmonds and Karp [6]. The current most efficient strongly polynomial algorithm, given by Orlin [29], is also based on this framework. On the other hand, Minoux extended [6] to the convex minimum-cost flow problem, first to convex quadratic flows ([25]), later to general convex objectives ([26]) Our algorithm is an enhanced version of the latter algorithm, in the spirit of Orlin’s technique [29]. However, there are important differences that make the nonlinear setting significantly harder. Let us remark that Orlin’s strongly polynomial algorithm for linear Fisher market [30] is also based on the ideas of [29]. In what follows, we give an informal overview of the key ideas of these algorithms that motivated our result. For more detailed references and proofs, we refer the reader to [1].

The algorithm of Edmonds and Karp consists of Δ -phases for a scaling parameter Δ . Initially, Δ is set to a large value, and decreases by at least a factor of two at the end of each phase. An optimal solution can be obtained for sufficiently small Δ . The elementary step of the Δ -phase transports Δ units of flow from a node with excess at least Δ to another node with demand at least Δ . This is done on a shortest path in the Δ -residual network, the graph of residual arcs with capacity at least Δ . An invariant property maintained in the Δ -phase is that the Δ -residual network does not contain any negative cost cycles. When moving to the next phase, the flow on the arcs has to be slightly modified to restore the invariant property.

Orlin’s algorithm [29] works on a problem instance with no upper capacities on the arcs (every minimum-cost flow problem can be easily transformed to this form). The basic idea is that if the algorithm runs for infinite number of phases, then the solution converges to an optimal solution; furthermore, the total change of the flow value in the Δ -phase and all subsequent phases is at most $4n\Delta$ on every arc. Consequently, if an arc ij has flow $> 4n\Delta$ in the Δ -phase, then the flow on ij must be positive in some optimal solution. Such an arc is called *abundant*. Using primal-dual slackness, this means that ij must be tight for an arbitrary dual optimal solution. It can be shown that within $O(\log n)$ scaling phases, an abundant arc ij appears.

Based on this observation, [29] obtains the following simple algorithm. Let us start running the Edmonds-Karp algorithm on the input graph. Once there is an abundant arc, it is contracted and the Edmonds-Karp algorithm is restarted on the smaller graph. The method is iterated until

the graph consists of a single node only. A dual optimal solution here can be easily extended to a dual optimal solution in the original graph by reversing the contraction operations. Provided a dual optimal solution, a primal optimal solution can be obtained by a single maximum flow computation. The paper [29] (see also [1, Chapter 10.7]) also contains a second, more efficient algorithm. When an abundant arc is found, instead of contracting and restarting, the arc is added to a special forest F . The scaling algorithm exploits properties of this forest and can thereby ensure that a new arc enters F in $O(\log n)$ phases. The running time can be bounded by $O(m \log n(m + n \log n))$, so far the most efficient minimum-cost flow algorithm known.

Let us now turn to the nonlinear setting. By the KKT conditions, a feasible solution is optimal if and only if the residual graph contains no negative cycles with respect to the cost function $C'_{ij}(f_{ij})$. The algorithm is a natural extension of the Edmonds-Karp scaling technique (see [25, 26], [1, Chapter 14.5]). In the Δ -phase it maintains the invariant that the Δ -residual graph contains no negative cycle with respect to the relaxed cost function $(C_{ij}(f_{ij} + \Delta) - C_{ij}(f_{ij}))/\Delta$. When transporting Δ -units of flow on a shortest path with respect to this cost function, this invariant is maintained. A key observation is that when moving to the $\Delta/2$ -phase, the invariant can be restored by changing the flow on each arc by at most $\Delta/2$. The role of the scaling factor Δ is twofold: besides being the quantity of the transported flow, it also approximates optimality in the following sense. As Δ approaches 0, the cost of ij converges to the derivative $C'_{ij}(f_{ij})$. Consequently, the solution converges to a feasible optimal solution. A variant of this algorithm is outlined in Section 3.

1.2 Strongly polynomial algorithm for convex quadratic flows

To formulate the exact assumptions needed for the general algorithm, several notions have to be introduced. Therefore we postpone the formulation of our main result Theorem 4.5 to Section 4.2. Now we exhibit the main ideas on the example of convex quadratic functions. We only give an informal overview here without providing all technical details; the precise definitions and descriptions are given in the later parts of the paper. Then in Section 6.1, we show how the general framework can be adapted to convex quadratic functions.

Let us assume that $C_{ij}(\alpha) = c_{ij}\alpha^2 + d_{ij}\alpha$ with $c_{ij} > 0$ for every arc $ij \in E$. The general algorithm also allows $c_{ij} = 0$. However, it is useful to make this restriction for the sake of our overview, as it guarantees that the optimal solution is unique, enabling simplifications to the general algorithm.

Let us further assume that the lower capacity is 0 on every arc and the upper capacity is ∞ . This is without loss of generality as an arbitrary instance can be transformed to such a form, by adding a corresponding new node for every arc (see Section 2). For node capacities b_i , our problem can be formulated as follows.

$$\begin{aligned} \min \quad & \sum_{ij \in E} c_{ij} f_{ij}^2 + d_{ij} f_{ij} \\ & \sum_{j:ji \in E} f_{ji} - \sum_{j:ij \in E} f_{ij} = b_i \quad \forall i \in V \\ & f \geq 0 \end{aligned}$$

Let f^* be the optimal solution; it is unique by the strict convexity of the objective. Let F^* denote the support of f^* . From the Karush-Kuhn-Tucker conditions, we can derive that f^* is unique solution of the following system of linear equations (note that the domain of f^* is restricted

to F^* ; π gives the optimal Lagrangian multipliers).

$$\begin{aligned} \pi_j - \pi_i &= 2c_{ij}f_{ij}^* + d_{ij} \quad \forall ij \in F^* \\ \sum_{j:ji \in F^*} f_{ji}^* - \sum_{j:ij \in F^*} f_{ij}^* &= b_i \quad \forall i \in V \\ f_{ij}^* &= 0 \quad \forall ij \in E \setminus F^* \end{aligned} \tag{1}$$

Consequently, provided F^* , we can obtain f^* by solving a system of linear equations (see Section 6.1 for details). Our subroutine $\text{TRIAL}(F, \hat{b})$ solves the above system of equations substituting F^* by an arbitrary arc set F and b by an arbitrary vector \hat{b} . Let us define the discrepancy $D_F(\hat{b})$ as the maximum value of $|\sum_{i \in K} \hat{b}_i|$, where K ranges over the undirected connected components of F . It can be verified that the system is solvable for F and \hat{b} if and only if $D_F(\hat{b}) = 0$ and a solution can be found in time $O(n^{2.37})$ (see Lemma 6.1).

Our starting point is a variant of Minoux's nonlinear scaling scheme as described above, with the only difference that the relaxed cost function is replaced by $C'_{ij}(f_{ij} + \Delta)$ (see Section 3).

Following Orlin [29], we can identify an arc carrying a ‘‘large’’ amount of flow in $O(\log n)$ steps. The required amount, $> (2n + m + 1)\Delta$ at the end of the Δ -phase, is large enough that even if we run the algorithm forever and thereby converge to the optimal solution f^* , this arc must remain positive. Consequently, it must be contained in F^* . However, we cannot simply contract such an arc as in [29]. The reason is that the KKT-conditions give $\pi_j - \pi_i = c_{ij}f_{ij}^* + d_{ij}$, a condition containing both primal and dual (more precisely, Lagrangian) variables simultaneously.

In every phase of the algorithm, we shall maintain a set $F \subseteq F^*$ of arcs, called *revealed arcs*. F will be extended by a new arc in every $O(\log n)$ phases; thus we find F^* in $O(m \log n)$ steps. Given a set $F \subseteq F^*$, we introduce some technical notions; the precise definitions and detailed discussions are given in Section 4.1. First, we waive the nonnegativity requirement on the arcs in F - a flow taking possibly negative values on the arcs in F but nonnegative in $E \setminus F$ is called an *F -pseudoflow*.

For an F -pseudoflow f and a scaling factor $\Delta > 0$, the (Δ, F) -residual graph contains all residual arcs where f can be increased by Δ so that it remains an F -pseudoflow (that is, all arcs in E , and all arcs ji where $ij \in F$ or $ij \in E \setminus F$ and $f_{ij} \geq \Delta$.) We require that the flow f in this phase satisfies the (Δ, F) -feasibility property: the (Δ, F) -residual graph contains no negative cycles with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$.

Let us now describe our algorithm. We start with $F = \emptyset$ and a sufficiently large Δ value so that the initial flow $f \equiv 0$ is (Δ, \emptyset) -feasible. We run the Minoux-type scaling algorithm sending flow on shortest paths in the (Δ, F) -residual graph from nodes with excess at least Δ to nodes with deficiency at least Δ . If there exist no more such paths, we move to the $\Delta/2$ -phase, after a simple modification step that transforms the flow to a $(\Delta/2, F)$ -feasible one, on the cost of increasing the total excess by at most $m\Delta/2$ (see subroutine ADJUST in Section 4.1). We include every edge ij into F with $f_{ij} > (2n + m + 1)\Delta$ at the end of the Δ -phase.

At the end of each phase when F is extended, we perform a special subroutine instead of simply moving to the $\Delta/2$ -phase. First, we compute the discrepancy $D_F(b)$ (as defined above). If this value is large, then it can be shown that F will be extended within $O(\log n)$ phases as in Orlin's algorithm.

If the discrepancy is small, the procedure TRIAL-AND-ERROR is performed, consisting of two subroutines. First, we run the subroutine $\text{TRIAL}(F, \hat{b})$, where \hat{b} is a small modification of b satisfying $D_F(\hat{b}) = 0$. This returns an F -pseudoflow \hat{f} , satisfying (1) with F in the place of F^* . (This step be seen as ‘‘pretending’’ that $F = F^*$ and trying to compute an optimal solution under this hypothesis.) The resulting \hat{f} is optimal if and only if $F = F^*$. Otherwise, we use a second subroutine $\text{ERROR}(\hat{f}, F)$, that returns the smallest value $\hat{\Delta} > 0$ such that \hat{f} is $(F, \hat{\Delta})$ -feasible.

This subroutine can be reduced to a minimum cost-to-time ratio cycle problem (also known as the tramp streamer problem), see [1, Chapter 5.7]; a strongly polynomial time algorithm was given by Megiddo [23].

If $\hat{\Delta} < \Delta/2$, then we set $\hat{\Delta}$ as our next scaling value and $f = \hat{f}$ as the next F -pseudoflow - we can proceed since \hat{f} is $(F, \hat{\Delta})$ -feasible. Otherwise, the standard transition to phase $\Delta/2$ is done with keeping the same flow f . The analysis shows that a new arc shall be revealed in every $O(\log n)$ phases. The key lemma is a proximity result between f and \hat{f} , which implies that TRIAL-AND-ERROR cannot return the same \hat{f} if performed again after $O(\log n)$ phases, implying that the set F cannot be the same, and has been therefore extended. Since $|F| \leq m$, this shows that the total number of scaling phases is $O(m \log n)$.

Besides the impossibility of contraction, an important difference as compared to Orlin's algorithm is that F cannot be assumed to be a forest (in the undirected sense). There are simple quadratic instances with the support of an optimal solution containing cycles. In Orlin's algorithm, progress is always made by connecting two components of F . This will also be an important event in our algorithm, but sometimes F shall be extended with arcs inside a component.

The paper is organized as follows. Section 2 contains the basic definitions and notation. Section 3 presents the simple adaptation of the Edmonds-Karp algorithm for convex cost functions, following Minoux [26]. Our algorithm in Section 4 is built on this algorithm with the addition of the subroutine TRIAL-AND-ERROR, that guarantees strongly polynomial running time. Analysis is given in Section 5. Section 6 adapts the general algorithm for quadratic utilities, and for Fisher's market with linear and with spending constraint utilities. Section 7 contains a final discussion of the results and some open questions.

2 Preliminaries

2.1 Problem definitions

Let $G = (V, E)$ be a directed graph possibly containing parallel arcs. Let $n = |V|$, $m = |E|$. We are given lower and upper capacities $\ell, u : E \rightarrow \mathbb{R} \cup \{\infty\}$ on the arcs, and node demands $b : V \rightarrow \mathbb{R}$ with $\sum_{i \in V} b_i = 0$. On each arc $ij \in E$, $C_{ij} : \mathbb{R} \rightarrow \mathbb{R} \cup \{\infty\}$ is a convex function. We allow two types of arcs ij :

- *Free arcs*: C_{ij} is differentiable everywhere on \mathbb{R} .
- *Restricted arcs*: $C_{ij}(\alpha) = \infty$ if $\alpha < \ell_{ij}$, C_{ij} is differentiable on (ℓ_{ij}, ∞) and has a right derivative in ℓ_{ij} , that equals $-\infty$; let $C'_{ij}(\ell_{ij}) = -\infty$ denote this right derivative. By convention, let us define $C'_{ij}(\alpha) = -\infty$ for $\alpha < \ell_{ij}$.

By convexity, C'_{ij} is continuous on \mathbb{R} for free and on $[\ell_{ij}, \infty)$ for restricted arcs. Restricted arcs will play a role in the Fisher market applications, where the function $C_{ij}(\alpha) = \alpha(\log \alpha - 1)$ will be used on certain arcs (with $C_{ij}(0) = 0$ and $C_{ij}(\alpha) = \infty$ if $\alpha < 0$.)

The minimum-cost flow problem with separable convex objective is defined as follows.

$$\begin{aligned}
 & \min \sum_{ij \in E} C_{ij}(f_{ij}) \\
 & \sum_{j:i \in E} f_{ji} - \sum_{j:i \in E} f_{ij} = b_i \quad \forall i \in V \\
 & \ell_{ij} \leq f_{ij} \leq u_{ij} \quad \forall ij \in E
 \end{aligned} \tag{P}$$

Throughout the paper, we shall work with the uncapacitated version, that is, $\ell \equiv 0$ and $u \equiv \infty$. With a standard method, every problem can be transformed to an equivalent uncapacitated form. Indeed, let us replace each arc $ij \in E$ by a new node k and two arcs ik and jk . Let us set $b_k = u_{ij} - \ell_{ij}$, $C_{ik}(\alpha) = C_{ij}(\alpha + \ell_{ij})$, $C_{jk} \equiv 0$. Furthermore, let us increase b_i by ℓ_{ij} and decrease b_j by u_{ij} . It is easy to see that this gives an equivalent optimization problem, and if the original graph had n' nodes and m' arcs, the transformed instance has $n = n' + m'$ nodes and $m = 2m'$ arcs.

Further, we may assume without loss of generality that $G = (V, E)$ is strongly connected and (P) is always feasible. Indeed, we can add a new node t with edges vt , tv for any $v \in V$, with extremely high (possibly linear) cost functions on the edges. This guarantees that an optimal solution shall not use such edges, whenever the problem is feasible. We will also assume $n \leq m$.

By a *pseudoflow* we mean a function $f : E \rightarrow \mathbb{R}$ satisfying the capacity constraints. For the uncapacitated problem, it simply means $f \geq 0$. Let

$$\rho_f(i) := \sum_{j:ji \in E} f_{ji} - \sum_{j:ij \in E} f_{ij}, \quad (2)$$

and let

$$Ex(f) = Ex_b(f) := \sum_{i \in V} \max\{\rho_f(i) - b_i, 0\}$$

denote the total positive excess. For an arc set F , let \overleftarrow{F} denote the set of backward arcs and let $\overleftrightarrow{F} = F \cup \overleftarrow{F}$. We shall use the vector norms $\|x\|_\infty = \max |x_i|$ and $\|x\|_1 = \sum |x_i|$.

Following [18] and [22], we do not require the functions C_{ij} to be given explicitly, but assume oracle access only.

Assumption 1. *We are given a differential oracle that satisfies either of the following properties.*

- (a) *For every arc $ij \in E$, the oracle returns the value $C'_{ij}(\alpha)$ in $O(1)$ time for every $\alpha \in \mathbb{R}$. If α is rational then $C'_{ij}(\alpha)$ is also rational.*
- (b) *For every arc $ij \in E$, the oracle returns the value $e^{C'_{ij}(\alpha)}$ in $O(1)$ time for every $\alpha \in \mathbb{R}$. If α is rational then $e^{C'_{ij}(\alpha)}$ is also rational.*

Assumption 1(a) holds for quadratic objectives, while (b) is valid for the Fisher markets, where the derivatives are $-\log U_{ij}$ and $\log \alpha$ on the different arcs. Note that we do not assume an evaluation oracle returning $C_{ij}(\alpha)$ or $e^{C_{ij}(\alpha)}$ - these values are not needed in the algorithm.

Our next assumption slightly restricts the class of functions C_{ij} for technical reasons.

Assumption 2. *Each cost function $C_{ij}(\alpha)$ is either linear or strictly convex, that is, $C'_{ij}(\alpha)$ is either constant or strictly monotone increasing.*

Arcs with $C_{ij}(\alpha)$ linear are called *linear arcs*, the rest is called *nonlinear arcs*. Let m_L and m_N denote their numbers, respectively. We use the terms linear and nonlinear for the corresponding reverse arcs as well. This assumption is only a mild restriction: if C_{ij} does not satisfy it, \mathbb{R} can be decomposed to intervals such that C'_{ij} is either constant or strictly monotone increasing on each interval. We can replace ij by a set of parallel arcs with appropriately chosen capacities and cost functions all of which satisfy the assumption (indeed, the piecewise linear utility functions in Fisher markets with spending constraint utilities will be handled in such a way). If the number of intervals is polynomial in n and m , this decreases the running time by a strongly polynomial factor only.

2.2 Optimality and Δ -feasibility

Given a pseudoflow f , let us define the residual graph E_f by $ij \in E_f$ if $ij \in E$, or $ji \in E$ and $f_{ji} > 0$. Arcs of the first type are called *forward*, those of the latter type *backward arcs*. We use the convention that on a backward arc ji , $f_{ji} = -f_{ij}$, $C_{ji}(\alpha) = C_{ij}(-\alpha)$, also convex and differentiable. The *residual capacity* is ∞ on forward arcs and f_{ij} on the backward arc ji .

The Karush-Kuhn-Tucker conditions assert that the solution f to (P) is optimal if and only if there exists a potential vector $\pi : V \rightarrow \mathbb{R}$ such that

$$\pi_j - \pi_i \leq C'_{ij}(f_{ij}) \quad \forall ij \in E_f. \quad (3)$$

This is equivalent to asserting that the residual graph contains no negative cost cycles with respect to the cost function $C'_{ij}(f_{ij})$.

For a value $\Delta > 0$, let $E_f(\Delta)$ denote the subset of arcs in E_f that have residual capacity at least Δ (in particular, it contains E). We say that the pseudoflow f is Δ -feasible, there exists a potential vector $\pi : V \rightarrow \mathbb{R}$ such that

$$\pi_j - \pi_i \leq C'_{ij}(f_{ij} + \Delta) \quad \forall ij \in E_f(\Delta). \quad (4)$$

Equivalently, f is Δ -feasible if and only if $E_f(\Delta)$ contains no negative cycles with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$. If ji is a reverse arc, then (4) gives $C'_{ij}(f_{ij} - \Delta) \leq \pi_j - \pi_i$.

We note that our notion is different (and weaker) than the analogous conditions in [26] and in [18], where $(C_{ij}(f_{ij} + \Delta) - C_{ij}(f_{ij}))/\Delta$ is used in the place of $C'_{ij}(f_{ij} + \Delta)$.

Subroutine ADJUST(Δ, f)

INPUT A 2Δ -feasible pseudoflow \bar{f} and a potential vector π satisfying (4) with \bar{f} and 2Δ .

OUTPUT A Δ -feasible pseudoflow f such that π satisfies (4) with f and Δ .

for all $ij \in E$ **do**

if $C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi_i$ **then** $f_{ij} \leftarrow \bar{f}_{ij} + \Delta$.

if $\bar{f}_{ji} \geq \Delta$ and $\pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta)$ **then** $f_{ij} \leftarrow \bar{f}_{ij} - \Delta$.

if neither of the above, then $f_{ij} \leftarrow \bar{f}_{ij}$.

Figure 1:

The subroutine ADJUST(Δ, f) (Figure 1) transforms a 2Δ -feasible pseudoflow to a Δ -feasible pseudoflow by possibly changing the value of every arc by $\pm\Delta$.

Lemma 2.1. *The subroutine ADJUST(Δ, f) is well-defined and correct: it returns a Δ -feasible pseudoflow with (f, π) satisfying (4). Further, $Ex(f) \leq Ex(\bar{f}) + m_N\Delta$ (recall that m_N is the number of nonlinear arcs).*

Proof. For well-definedness, we observe that the two alternatives cannot hold simultaneously: $C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta)$ would contradict the convexity of C_{ij} . Consider the potential vector π satisfying (4) with \bar{f} and 2Δ . We prove that π satisfies (4) with f and Δ as well.

First, take a forward arc $ij \in E$ with $C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi_i$. By 2Δ -feasibility we know $\pi_j - \pi_i \leq C'_{ij}(\bar{f}_{ij} + 2\Delta)$. These show that setting $f_{ij} = \bar{f}_{ij} + \Delta$ satisfies (4) for both ij and ji , using that

$$C'_{ij}(f_{ij} - \Delta) \leq C'_{ij}(f_{ij}) = C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi_i \leq C'_{ij}(\bar{f}_{ij} + 2\Delta) = C'_{ij}(f_{ij} + \Delta).$$

Next, assume $\bar{f}_{ji} \geq \Delta$ and $\pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta)$. Note that f_{ij} satisfies (4) by $\pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta) \leq C'_{ij}(\bar{f}_{ij}) = C'_{ij}(f_{ij} + \Delta)$.

If $ji \in E_{\bar{f}}(2\Delta)$ (that is, $\bar{f}_{ij} \geq 2\Delta$), then we have $C'_{ij}(f_{ij} - \Delta) = C'_{ij}(\bar{f}_{ij} - 2\Delta) \leq \pi_j - \pi_i$, and thus (4) also holds for ji . If $ji \in E_{\bar{f}}(\Delta) - E_{\bar{f}}(2\Delta)$, then $ji \notin E_f(\Delta)$.

Finally, consider the case when $f_{ij} = \bar{f}_{ij}$. The condition (4) holds for ij as we assume $\pi_j - \pi_i \leq C'_{ij}(f_{ij} + \Delta)$. Also, either $f_{ij} = \bar{f}_{ij} < \Delta$ and thus $ji \notin E_f(\Delta)$, or $f_{ij} = \bar{f}_{ij} \geq \Delta$ and (4) holds for ji by the assumption $C'_{ij}(\bar{f}_{ij} - \Delta) \leq \pi_j - \pi_i$.

To verify the last claim, observe that C'_{ij} is constant on every linear arc and therefore $\bar{f}_{ij} = f_{ij}$ will be set on every linear arc. The flow change is $\pm\Delta$ on every nonlinear arc; every such change may increase the excess of one of the endpoints of the arc by Δ . Consequently, $Ex(f) \leq Ex(\bar{f}) + m_N \Delta$ follows. \square

3 The basic algorithm

Figure 2 outlines a simple algorithm for minimum cost flows with separable convex objectives, to be referred as the ‘‘Basic algorithm’’. This is a modified version of Minoux’s algorithm [26].

```

Algorithm BASIC
 $f \leftarrow \mathbf{0}; \Delta \leftarrow \Delta_0;$ 
repeat // $\Delta$ -phase
  do //main part
     $S(\Delta) \leftarrow \{i \in V : \rho_f(i) - b_i \geq \Delta\};$ 
     $T(\Delta) \leftarrow \{i \in V : \rho_f(i) - b_i \leq -\Delta\};$ 
     $P \leftarrow$  shortest  $s - t$  path in  $E_f(\Delta)$  for the cost  $C'_{ij}(f_{ij} + \Delta)$  with  $s \in S(\Delta), t \in T(\Delta);$ 
    send  $\Delta$  units of flow on  $P$  from  $s$  to  $t;$ 
  while  $S(\Delta), T(\Delta) \neq \emptyset;$ 
  ADJUST( $\Delta/2, f$ );
   $\Delta \leftarrow \Delta/2;$ 

```

Figure 2:

We start with the pseudoflow $f \equiv \mathbf{0}$ and an initial value $\Delta = \Delta_0$. We assume that the value Δ_0 is provided in the input so that $\mathbf{0}$ is a Δ_0 -feasible and $Ex(\mathbf{0}) \leq (2n + m)\Delta_0$; in the enhanced algorithm we shall specify how such a Δ_0 value can be determined. The algorithm consists of Δ -phases, with Δ decreasing by a factor of two between two phases.

In the main part of phase Δ , let $S(\Delta) = \{i \in V : \rho_f(i) - b_i \geq \Delta\}$ and $T(\Delta) = \{i \in V : \rho_f(i) - b_i \leq -\Delta\}$, the set of nodes with excess and deficiency at least Δ . As long as $S(\Delta) \neq \emptyset$, $T(\Delta) \neq \emptyset$, send Δ units of flow from a node $s \in S(\Delta)$ to a node $t \in T(\Delta)$ on a shortest path in $E_f(\Delta)$ with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$. (Note that there must be a path connecting nodes in $S(\Delta)$ and $T(\Delta)$, due to our assumption that the graph $G = (V, E)$ is strongly connected, and $E \subseteq E_f(\Delta)$.)

The main part finishes once $S(\Delta) = \emptyset$ or $T(\Delta) = \emptyset$. The Δ -phase terminates by performing ADJUST($\Delta/2, f$) and proceeding to the next phase with scaling factor $\Delta/2$.

In the main part, we need to compute shortest paths in the graph $E_f(\Delta)$ for the cost function $C'_{ij}(f_{ij} + \Delta)$. This can be done only if there is no negative cost cycle. Δ -feasibility is exactly

this property and is maintained throughout (see Lemma 3.1 below). Details of the shortest path computation will be given in Section 5.1(ii), for the enhanced algorithm.

Note that the algorithm in this form does not terminate. If one requires an ε -accurate solution, it can be shown that for sufficiently small Δ we are able to obtain a ε -accurate solution. However, we do not include these details as the Basic algorithm is needed in the analysis of the enhanced algorithm exactly in the current infinite (nonterminating) version.

3.1 Analysis

We omit the proof of the following two simple lemmas; their analogous counterparts for the enhanced algorithm will be proved in Section 5. By the main part of the Δ -phase, we mean the sequence of flow augmentations.

Lemma 3.1. (i) *In the main part of the Δ -phase, the pseudoflow is an integer multiple of Δ on each arc, and consequently, $E_f(\Delta) = E_f$.*

(ii) *Δ -feasibility is maintained when augmenting on a shortest path.*

(iii) *At the beginning of the main part, $Ex(f) \leq (2n + m_N)\Delta$, and at the end, $Ex(f) \leq n\Delta$.*

(iv) *The main part consists of at most $2n + m_N$ flow augmentation steps.*

Lemma 3.2. *Let f be the pseudoflow at the end of the main part of the Δ -phase and f' in an arbitrary later phase. Then $\|f - f'\|_\infty \leq (2n + m + 1)\Delta$. If $f_{ij} > (2n + m + 1)\Delta$ at the end of the Δ -phase, then this property is maintained in all later phases, and there exists an optimal solution f^* with $f_{ij}^* > 0$.*

4 The enhanced algorithm

4.1 Revealed arc sets

Let F^* denote the set of arcs that are tight in every optimal solution (note that in general, we do not assume the uniqueness of the optimal solution). This arc set plays a key role in our algorithm. Formally,

$$F^* := \{ij \in E : \pi_j - \pi_i = C'_{ij}(f_{ij}) \text{ holds } \forall (f, \pi) \text{ satisfying (3)}\}.$$

The next lemma shows that F^* contains the support of every optimal solution.

Lemma 4.1. *Let f be an arbitrary optimal solution to (P), and $f_{ij} > 0$ for some $ij \in E$. Then $ij \in F^*$.*

The proof needs the following notion, also used later. Let $x, y : E \rightarrow \mathbb{R}$ be two vectors. Let us define the *difference graph* $D_{x,y} = (V, E_{x,y})$ with $ij \in E_{x,y}$ if $x_{ij} > y_{ij}$ or if $x_{ji} < y_{ji}$. Using the convention $x_{ji} = -x_{ij}$, $y_{ji} = -y_{ij}$ it follows that $x_{ij} > y_{ij}$ for every $ij \in E_{x,y}$. We will need the following simple claim.

Claim 4.2. *Assume that for two vectors $x, y : E \rightarrow \mathbb{R}$, $\rho_x = \rho_y$ holds (recall the definition of ρ in (2)). Then every arc ij with $x_{ij} > y_{ij}$ must be contained in a cycle in the difference graph $E_{x,y}$.*

Proof. Let us set $z_{ij} = x_{ij} - y_{ij}$ if $x_{ij} > y_{ij}$ and $z_{ij} = y_{ji} - x_{ji}$ if $x_{ji} < y_{ji}$. The assumption $\rho_x = \rho_y$ implies that z_{ij} is a circulation in $E_{x,y}$ with positive value on every arc. As such, it can be written as a nonnegative combination of incidence vectors of cycles. If $x_{ij} > y_{ij}$ then $z_{ij} > 0$ and therefore ij must be contained in a cycle in $E_{x,y}$. \square

Proof of Lemma 4.1. Let f^* be an arbitrary optimal solution, and consider potentials π and π^* with both (f, π) and (f^*, π^*) satisfying (3). We shall prove that $\pi_j^* - \pi_i^* = C'_{ij}(f_{ij}^*)$. Since (f^*, π^*) is chosen arbitrarily, this will imply $ij \in F^*$. If $f_{ij}^* > 0$, then $ji \in E_{f^*}$ and thus $\pi_j^* - \pi_i^* = C'_{ij}(f_{ij}^*)$ must hold.

Assume now $f_{ij}^* = 0$. Consider the difference graph D_{f, f^*} . Since $f_{ij} > f_{ij}^*$, it follows that $ij \in E_{f, f^*}$. Because of $\rho_{f^*} \equiv \rho_f \equiv b$, Claim 4.2 is applicable and provides a cycle C in E_{f, f^*} containing ij . For every arc $ab \in C$, $f_{ab} > f_{ab}^*$ and thus $ab \in E_{f^*}$ and $ba \in E_f$. By (3),

$$\begin{aligned} 0 &= \sum_{ab \in C} \pi_b^* - \pi_a^* \leq \sum_{ab \in C} C'_{ab}(f_{ab}^*) \quad \text{and} \\ 0 &= \sum_{ab \in C} \pi_a - \pi_b \leq \sum_{ab \in C} C'_{ba}(f_{ba}) = - \sum_{ab \in C} C'_{ab}(f_{ab}). \end{aligned}$$

The convexity of C_{ab} and $f_{ab} > f_{ab}^*$ give $C'_{ab}(f_{ab}) \geq C'_{ab}(f_{ab}^*)$. In the above inequalities, equality must hold everywhere, implying $\pi_j^* - \pi_i^* = C'_{ij}(f_{ij}^*)$ as desired. \square

We shall see that under Assumption 3 (to be formulated later), finding the set F^* enables us to compute an optimal solution in strongly polynomial time. In the Basic algorithm, $F = \{ij \in E : f_{ij} > (2n + m + 1)\Delta\}$ is always a subset of F^* according to Lemma 3.2. Furthermore, once an edge enters F , it stays there in all later phases. Yet there is no guarantee (and it is in fact not true) that in the Basic algorithm, F would be extended in some number of steps polynomially bounded in n and m . We shall modify the basic algorithm in order to guarantee that within $O(\log n)$ phases, a new arc is guaranteed to enter F .

In each step of the enhanced algorithm, there will be an arc set F , called the *revealed arc set*, which is guaranteed to be a subset of F^* . We remove the lower capacity 0 from arcs in F and allow also negative values here.

Formally, for an edge set $F \subseteq E$, a vector $f : E \rightarrow \mathbb{R}$ is an F -pseudoflow, if $f_{ij} \geq 0$ for $ij \in E \setminus F$ (but it is allowed to be negative on F). For such an f , let us define

$$E_f^F := E_f \cup \overleftarrow{F} = E \cup \overleftarrow{F} \cup \{ji : ij \in E, f_{ij} > 0\}.$$

If $ij \in F$, then the residual capacity of ji is ∞ . In every phase of the algorithm, we maintain an F -pseudoflow f for a revealed arc set $F \subseteq F^*$.

Provided the revealed arc set $F \subseteq F^*$, we will aim for F -optimal solutions as defined below; we prove that finding an F -optimal solution is essentially equivalent to finding an optimal one. We say that $f : E \rightarrow \mathbb{R}$ is F -optimal, if it is an F -pseudoflow with $Ex_b(f) = 0$ and there exists a potential vector $\pi : V \rightarrow \mathbb{R}$ with

$$\pi_j - \pi_i \leq C'_{ij}(f_{ij}) \quad \forall ij \in E_f^F. \quad (5)$$

This is stronger than the optimality condition (3) by requiring the inequality also on arcs in \overleftarrow{F} . On the other hand, it does not imply optimality as it allows $f_{ij} < 0$ for $ij \in F$. Nevertheless, it is easy to see that every optimal solution f^* is also F -optimal for every $F \subseteq F^*$. This is due to the definition of F^* as the set of arcs satisfying $\pi_j - \pi_i = C'_{ij}(f_{ij})$ whenever (f, π) satisfies (3). Conversely, we shall prove that provided an F -optimal solution, we can easily find an optimal solution by a single feasible circulation algorithm, a problem equivalent to maximum flows (see [1, Chapters 6.2, 7]).

Lemma 4.3. *Assume that for a subset $F \subseteq F^*$, an F -optimal solution f is provided. Then an optimal solution to (P) can be found by a feasible circulation algorithm. Further, $ij \in F^*$ whenever $f_{ij} > 0$.*

Proof. Assume (f, π) and $(\bar{f}, \bar{\pi})$ both satisfy (5). We prove that (i) $f_{ij} = \bar{f}_{ij}$ whenever ij is a nonlinear arc; and (ii) if ij is a linear arc with $f_{ij} \neq \bar{f}_{ij}$, then $\pi_j - \pi_i = C'_{ij}(f_{ij}) = C'_{ij}(\bar{f}_{ij}) = \bar{\pi}_j - \bar{\pi}_i$.

Note that (ii) immediately implies the second half of the claim as it can be applied for f and an arbitrary optimal (and consequently, F -optimal) solution \bar{f} .

The proof uses the same argument as for Lemma 4.1. W.l.o.g. assume $f_{ij} > \bar{f}_{ij}$ for an arc ij , and consider the difference graph $D_{f, \bar{f}}$. Since $\rho_f \equiv \rho_{\bar{f}} \equiv b$ and $f_{ij} > \bar{f}_{ij}$, Claim 4.2 is applicable and shows that ij must be contained on a cycle $C \subseteq E_{f, \bar{f}}$. For every arc $ab \in C$, $ab \in E_{\bar{f}}^F$ and $ba \in E_f^F$ follows (using $\overleftarrow{F} \subseteq E_{\bar{f}}^F \cap E_f^F$). By (5),

$$\begin{aligned} 0 &= \sum_{ab \in C} \bar{\pi}_b - \bar{\pi}_a \leq \sum_{ab \in C} C'_{ab}(\bar{f}_{ab}) \quad \text{and} \\ 0 &= \sum_{ab \in C} \pi_a - \pi_b \leq \sum_{ab \in C} C'_{ba}(f_{ba}) = - \sum_{ab \in C} C'_{ab}(f_{ab}). \end{aligned}$$

Now convexity yields $C'_{ab}(f_{ab}) = C'_{ab}(\bar{f}_{ab})$ for all $ab \in C$. Assumption 2 implies that all arcs in C are linear, in particular, ij is linear. This immediately proves (i). To verify (ii), observe that all above inequalities must hold with equality.

This suggests the following simple method to transform an F -optimal solution f to an optimal f^* . For every nonlinear arc ij , we must have $f_{ij}^* = f_{ij}$. Let $H \subseteq E$ be the set of linear arcs satisfying $\pi_j - \pi_i = C'_{ij}(f_{ij})$. Consider the solutions h of the following feasible circulation problem:

$$\begin{aligned} h_{ij} &= f_{ij} \quad \forall ij \in E \setminus H \\ \sum_{j:ji \in E} h_{ji} - \sum_{j:ij \in E} h_{ij} &= b_i \quad \forall i \in V \\ h &\geq 0 \end{aligned}$$

We claim that the feasible solutions to this circulation problem are precisely the optimal solutions to (P). Indeed, if f^* is an optimal solution, then (5) and (ii) imply that $f_{ij}^* = f_{ij}$ for all $ij \in E \setminus H$ and $ij \in H$ for every arc with $f_{ij} \neq f_{ij}^*$. The degree conditions are satisfied because of $\rho_{f^*} \equiv \rho_f \equiv b$. Conversely, every feasible circulation h is an optimal solution to (P), since (h, π) satisfies (3). \square

In every step of our algorithm we will have a scaling parameter $\Delta \geq 0$ and a revealed arc set $F \subseteq F^*$. The Basic algorithm used the notion of Δ -feasibility; it has to be modified according to F . Let $E_f^F(\Delta)$ denote the set of arcs in E_f^F with residual capacity at least Δ . That is,

$$E_f^F(\Delta) := E_f(\Delta) \cup \overleftarrow{F} = E \cup \overleftarrow{F} \cup \{ji : ij \in E, f_{ij} \geq \Delta\}. \quad (6)$$

We say that the F -pseudoflow f is (Δ, F) -feasible, if there exists a potential vector $\pi : V \rightarrow \mathbb{R}$ so that

$$\pi_j - \pi_i \leq C'_{ij}(f_{ij} + \Delta) \quad \forall ij \in E_f^F(\Delta). \quad (7)$$

This is equivalent to the property that $E_f^F(\Delta)$ contains no negative cycle with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$.

In accordance with (Δ, F) -feasibility, we have to modify the subroutine ADJUST. The modified subroutine, denoted by ADJUST(Δ, f, F) is shown on Figure 3. The only difference from the algorithm on Figure 1 is that the condition (4) is replaced by (7), and that in the second condition, " $\bar{f}_{ji} \geq \Delta$ " is replaced by " $\bar{f}_{ji} \geq \Delta$ or $ij \in F$ ". The following lemma can be proved by the same argument as Lemma 2.1.

Subroutine ADJUST(Δ, f, F)

INPUT A $(2\Delta, F)$ -feasible pseudoflow \bar{f} and a potential vector π satisfying (7) with \bar{f} and 2Δ .

OUTPUT A (Δ, F) -feasible pseudoflow f such that π satisfies (7) with f and Δ .

for all $ij \in E$ **do**

if $C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi$ **then** $f_{ij} \leftarrow \bar{f}_{ij} + \Delta$.

if $(\bar{f}_{ji} \geq \Delta$ or $ij \in F)$ and $\pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta)$ **then** $f_{ij} \leftarrow \bar{f}_{ij} - \Delta$.

if neither of the above, then $f_{ij} \leftarrow \bar{f}_{ij}$.

Figure 3:

Lemma 4.4. *The subroutine ADJUST(Δ, f, F) is well-defined and correct: it returns a (Δ, F) -feasible pseudoflow with (f, π) satisfying (7). Further, $Ex(f) \leq Ex(\bar{f}) + m_N \Delta$.*

Finally, we say that a set $F \subseteq E$ is *linear acyclic*, if F does not contain any undirected cycles of linear arcs (that is, no cycle in F may consist of linear arcs and their reverse arcs). We shall maintain that the set of revealed arcs, F is linear acyclic.

4.2 Subroutine assumptions

Given the set $F \subseteq F^*$ of revealed arcs, we will try to find out whether F already contains the support of an optimal solution. This motivates the following definition. We say that the (not necessarily nonnegative) vector $x : E \rightarrow \mathbb{R}$ is *F-tight*, if $x_{ij} = 0$ whenever $ij \notin F$ and there exists a potential vector $\pi : V \rightarrow \mathbb{R}$ with

$$\pi_j - \pi_i = C'_{ij}(x_{ij}) \quad \forall ij \in F. \quad (8)$$

For example, any optimal solution is F^* -tight by Lemma 4.1. Notice that an F -tight vector f is not necessarily F -optimal as (5) might be violated for edges in $E_f^F \setminus \overleftarrow{F}$ and also since $Ex_b(f) > 0$ is allowed. On the other hand, an F -optimal vector is not necessarily F -tight as it can be nonzero on $E \setminus F$.

Given F and some node demands $\hat{b} : V \rightarrow \mathbb{R}$, we would like to find an F -tight x with $Ex_{\hat{b}}(x) = 0$. This is equivalent to finding a feasible solution (x, π) to the following system:

$$\begin{aligned} \pi_j - \pi_i &= C'_{ij}(x_{ij}) \quad \forall ij \in F \\ \sum_{j:ji \in F} x_{ji} - \sum_{j:ij \in E} x_{ij} &= \hat{b}_i \quad \forall i \in V \\ x_{ij} &= 0 \quad \forall ij \in E \setminus F \end{aligned} \quad (9)$$

Let us define the *discrepancy* $D_{\hat{b}}(F)$ of F as the maximum of $|\sum_{i \in K} \hat{b}_i|$ over undirected connected components K of F . A trivial necessary condition for solvability is $D_{\hat{b}}(F) = 0$: indeed, summing up the second set of equalities for a component K we obtain $0 = \sum_{i \in K} \hat{b}_i$.

Assumption 3. *Assume we have a subroutine TRIAL(F, \hat{b}) so that for any linear acyclic $F \subseteq E$ and any vector $\hat{b} : V \rightarrow \mathbb{R}$ satisfying $D_{\hat{b}}(F) = 0$, it delivers an F -tight solution x to (9) with $Ex_{\hat{b}}(x) = 0$ in strongly polynomial running time $\rho_T(n, m)$.*

For quadratic cost functions and also for Fisher markets, this subroutine can be implemented by solving simple systems of equations (for quadratic, this was outlined in Section 1.2).

Consider now an F -tight vector f , and let

$$err_F(f) := \inf\{\Delta : f \text{ is } (\Delta, F)\text{-feasible}\}.$$

Recall the definition (6) of the edge set $E_f^F(\Delta)$. As f is assumed to be F -tight and therefore $f_{ij} > 0$ only if $ij \in F$, we get that $E_f^F(\Delta) = E \cup \overleftarrow{F}$. Consequently, $E_f^F(\Delta)$ is independent from the value of Δ . Because of continuity, this infimum is actually a minimum whenever the set is nonempty. If f is not (Δ, F) -feasible for any Δ , then let $err_F(f) = \infty$. f is F -optimal if and only if f is a feasible flow (that is, $Ex_b(f) = 0$) and $err_F(f) = 0$.

Assumption 4. *Assume a subroutine $\text{ERROR}(f, F)$ is provided, that returns $err_F(f)$ for any F -tight vector f in strongly polynomial running time $\rho_E(n, m)$. Further, if $err_\emptyset(\mathbf{0}) = \infty$, then (P) is unbounded.*

This subroutine seems significantly harder to implement for the applications: we need to solve a minimum cost-to-time ratio cycle problem for quadratic costs and all pairs shortest paths for the Fisher markets.

Having formulated all necessary assumptions, we are finally in the position to formulate the main result of the paper.

Theorem 4.5. *Let Assumptions 1-4 hold for the problem (P) in a network on n nodes and m arcs, m_N among them having nonlinear cost functions. Let $\rho_T(n, m)$ and $\rho_E(n, m)$ denote the running time of the subroutines defined in Assumptions 3 and 4, and let $\rho_S(n, m)$ be the running time needed for a single shortest path computation. Then an exact optimal solution can be found in $O((n + m_N)(\rho_T(n, m) + \rho_E(n, m)) + (n + m_N)^2 \rho_S(n, m) \log m)$ time.*

This gives an $O(m^4 \log m)$ algorithm for quadratic convex objectives. For Fisher markets, we obtain $O(n^4 + n^2(m + n \log n) \log n)$ running time for linear and $O(mn^3 + m^2(m + n \log n) \log m)$ for spending constraint utilities.

4.3 Description of the enhanced algorithm

The algorithm (see Figure 4) starts with the $f = \mathbf{0}$, $\Delta = \max\{err_\emptyset(\mathbf{0}), Ex_b(\mathbf{0})/(2n + m_N)\}$ and $F = \emptyset$. The algorithm consists of Δ -phases. In the Δ -phase, we shall maintain a linear acyclic revealed arc set $F \subseteq F^*$, and a (Δ, F) -feasible F -pseudoflow f .

The main part of the Δ -phase is the same as in the Basic algorithm. Let $S(\Delta) = \{i \in V : \rho_f(i) - b_i \geq \Delta\}$ and $T(\Delta) = \{i \in V : \rho_f(i) - b_i \leq -\Delta\}$. As long as $S(\Delta) \neq \emptyset$, $T(\Delta) \neq \emptyset$, send Δ units of flow from a node $s \in S(\Delta)$ to a node $t \in T(\Delta)$ on a shortest path in $E_f^F(\Delta)$ with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$. (The existence of such a path P is guaranteed by our assumption that the graph $G = (V, E)$ is strongly connected.)

After the main part (the sequence of path augmentations) is finished, the subroutine $\text{EXTEND}(\Delta, f, F)$ adds new arcs $ij \in E \setminus F$ with $f_{ij} > (2n + m + 1)\Delta$ to F maintaining the linear acyclic property. Note that all nonlinear such arcs will be included in F .

If no new arc enters F , then we perform $\text{ADJUST}(\Delta/2, f, F)$ and move to the next scaling phase with the same f and set the scaling factor to $\Delta/2$. This is done also if F is extended, but it has a high discrepancy: $D_b(F) > \Delta$.

Otherwise, the subroutine $\text{TRIAL-AND-ERROR}(F)$ determines the next f and Δ . Based on the arc set F , we find a new F -pseudoflow f and scaling factor at most $\Delta/2$. The subroutine may also

terminate with an F -optimal solution, which enables us to find an optimal solution to (P) by a maximum flow computation due to Lemma 4.3.

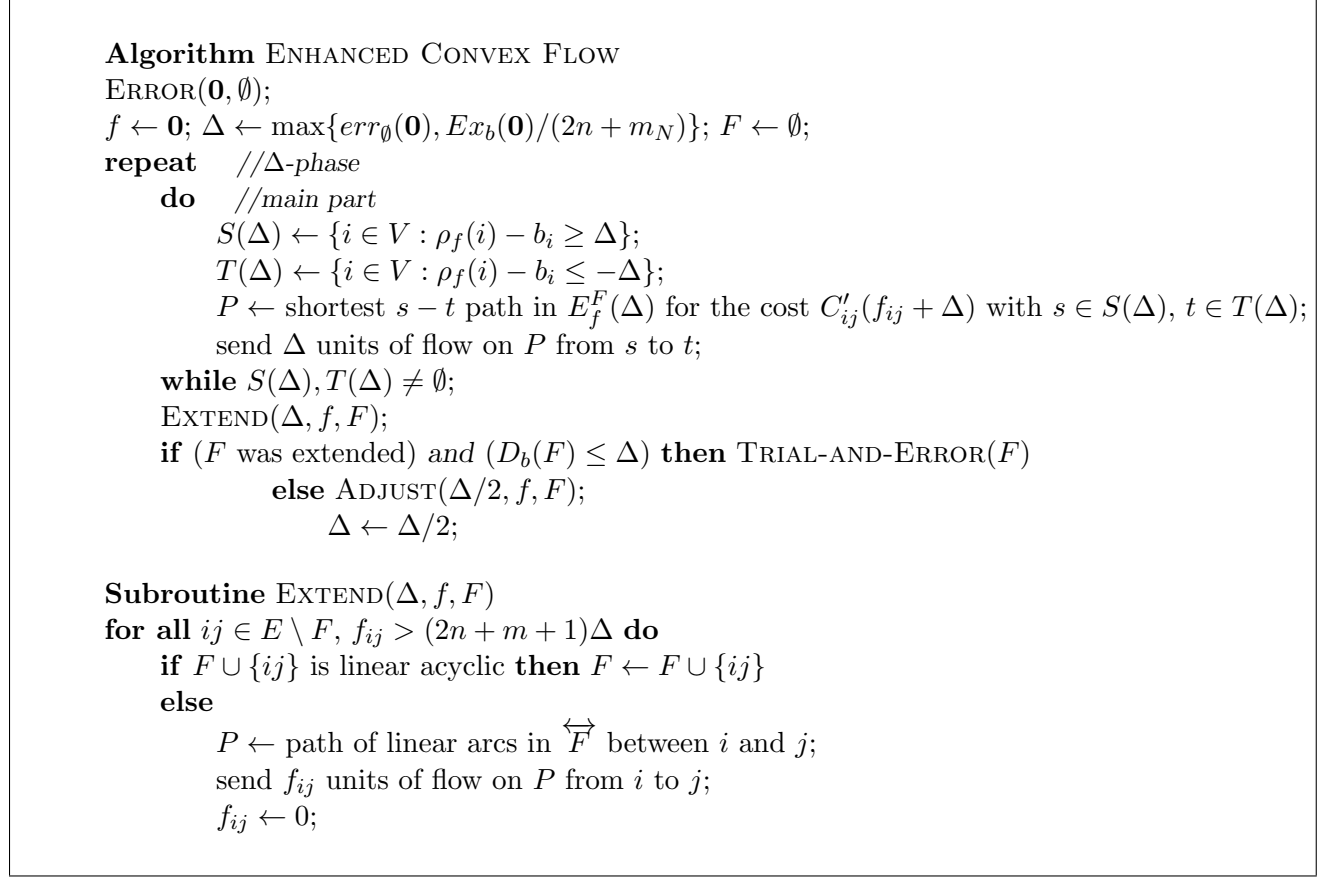


Figure 4:

The Trial-and-Error subroutine

The subroutine assumes that the discrepancy of F is small: $D_b(F) \leq \Delta$.

Step 1. First, modify b to \hat{b} : in each (undirected) component K of F , pick a node $j \in K$ and change b_j by $-\sum_{i \in K} b_i$; leave all other b_i values unchanged. Thus we get a \hat{b} with $D_{\hat{b}}(F) = 0$. **TRIAL**(F, \hat{b}) returns an F -tight vector \hat{f} .

Step 2. Call the subroutine **ERROR**(\hat{f}, F). If $b = \hat{b}$ and $err_F(\hat{f}) = 0$, then \hat{f} is F -optimal. An optimal solution to (P) can be found by a single maximum flow computation, as described in the proof of Lemma 4.3. In this case, the algorithm terminates. If $err_F(\hat{f}) \geq \Delta/2$, then keep the original f , perform **ADJUST**($\Delta/2, f, F$) and go to the next scaling phase with scaling factor $\Delta/2$. Otherwise, set $f = \hat{f}$ and define the next scaling factor as

$$\Delta_{next} = \max\{err_F(\hat{f}), Ex_b(\hat{f})/(2n + m_N)\}.$$

5 Analysis

The details how the shortest path computations are performed will be discussed in Section 5.1; in the following analysis, we assume it can be efficiently implemented. At the initialization, $err_{\emptyset}(\mathbf{0})$

must be finite or the problem is unbounded by the second part of Assumption 4.

TRIAL-AND-ERROR replaces f by \hat{f} if $\text{err}_F(\hat{f}) \leq \Delta/2$ and keeps the same f otherwise. The first case will be called a *successful* trial, the latter is *unsuccessful*. The following is (an almost identical) counterpart of Lemma 3.1.

Lemma 5.1. (i) *In the main part of the Δ -phase, the F -pseudoflow f is an integer multiple of Δ on each arc $ij \in E \setminus F$, and consequently, $E_f^F(\Delta) = E_f^F$.*

(ii) *(Δ, F) -feasibility is maintained in the main part and in subroutine $\text{EXTEND}(\Delta, f, F)$.*

(iii) *At the beginning of the main part, $\text{Ex}(f) \leq (2n + m_N)\Delta$, and at the end, $\text{Ex}(f) \leq n\Delta$.*

(iv) *The main part consists of at most $2n + m_N$ flow augmentation steps.*

Proof. For (i), f is zero on every arc in $E \setminus F$ at the beginning of the algorithm and after every successful trial. In every other case, the previous phase had scaling factor 2Δ , and thus by induction, the flow is an integer multiple of 2Δ at the end of the main part of the 2Δ -phase, a property also maintained by $\text{EXTEND}(2\Delta, f, F)$. The 2Δ -phase finishes with $\text{ADJUST}(\Delta, f, F)$, possibly modifying the flow on every arc by $\pm\Delta$. In the main part of the Δ -phase, the shortest path augmentations also change the flow by $\pm\Delta$. This implies $E_f^F(\Delta) = E_f^F$.

For (ii), P is a shortest path if there exists a potential π satisfying (7) with $\pi_j - \pi_i = C'_{ij}(f_{ij} + \Delta)$ on each arc $ij \in P$ (see also Section 5.1). We show that when augmenting on the shortest path P , (7) is maintained with the same π . If $ij, ji \notin P$, then it is trivial as the flow is left unchanged on ij . If $ij \in P$, then the new flow value will be $f_{ij} + \Delta$, hence we need $\pi_j - \pi_i \leq C'_{ij}(f_{ij} + 2\Delta)$, obvious as C'_{ij} is monotonely increasing. Finally, if $ji \in P$, then the new flow is $f_{ij} - \Delta$, and thus we need $\pi_j - \pi_i \leq C'_{ij}(f_{ij})$. By $ji \in P$ we had $\pi_i - \pi_j = C'_{ji}(f_{ji} + \Delta)$, which is equivalent to $\pi_j - \pi_i = C'_{ij}(f_{ij} - \Delta)$, implying again the claim.

In subroutine EXTEND , we reroute the flow f_{ij} from a linear arc ij if \overleftrightarrow{F} contains a directed path P from i to j . This cannot affect feasibility since the C'_{ij} 's are constant on linear arcs. Also note that arcs in \overleftrightarrow{F} have infinite residual capacities.

For (iii), $\text{Ex}(f) \leq n\Delta$ as the main part terminates with either $S(\Delta) = \emptyset$ or $T(\Delta) = \emptyset$. Lemma 4.4 shows that $\text{ADJUST}(\Delta/2, f, F)$ increases the excess by at most $m_N\Delta/2$. Consequently, $\text{Ex}(f) \leq (2n + m_N)(\Delta/2)$ at the beginning of the $\Delta/2$ -phase.

The other possible case is that a successful trial replaces Δ by Δ_{next} . By definition, the new excess is at most $(2n + m_N)\Delta_{\text{next}}$.

Finally, (iii) implies (iv), as each flow augmentation decreases $\text{Ex}(f)$ by Δ . \square

Lemma 5.2. *$F \subseteq F^*$ holds in each step of the algorithm.*

Proof. The proof is by induction. A new arc ij may enter F if $f_{ij} > (2n + m + 1)\Delta$ after the main part of the Δ -phase. We shall prove that $f_{ij}^* > 0$ for some F -optimal solution f^* , and thus Lemma 4.3 gives $ij \in F^*$.

After the phase when ij entered, let us continue with the following modified algorithm: do not extend F and do not perform TRIAL-AND-ERROR anymore, but always choose the next scaling factor as $\Delta/2$, and keep the algorithm running forever. (This is almost the same as the Basic algorithm, with the difference that we have a revealed arc set F .)

Let $\Delta_0 = \Delta$ and $\Delta_t = \Delta/2^t$ denote the scaling factor in the t 'th phase of this algorithm (with phase 0 corresponding to the Δ -phase). Consider any Δ_t -phase ($t \geq 1$). The flow is modified by at most $(2n + m_N)\Delta_t$ during the main part by Lemma 5.1(iv) and by $\Delta_t/2$ in $\text{ADJUST}(\Delta_t/2, f, F)$,

amounting to a total modification $\leq (2n + m_N + \frac{1}{2})\Delta_t$. Consequently, the total modification in the Δ_t phase and all later phases is bounded by $(2n + m_N + \frac{1}{2}) \sum_{k=t}^{\infty} \Delta_k \leq 2(2n + m + \frac{1}{2})\Delta_t$.

We may conclude that when running forever, the flow f converges to an F -optimal solution f^* . Indeed, let $f^{(t)}$ denote the F -pseudoflow at the end of the t 'th phase. By the above observation, $\|f^{(t)} - f^{(t')}\|_{\infty} \leq (2n + m + \frac{1}{2})\Delta_t$ for any $t' \geq t \geq 0$. Consequently, on every arc $ij \in E$, the sequence $f_{ij}^{(t)}$ converges; let f^* denote the limit. We claim the f^* is F -optimal.

Firstly, f^* is clearly an F -pseudoflow. Property (5) is equivalent to the property that E_f^F does not contain any negative cycle w.r.t. $C'_{ij}(f_{ij})$. This follows from the fact that $E_f^F(\Delta_t)$ does not contain any negative cycle w.r.t. $C'_{ij}(f_{ij}^{(t)})$ due to the (Δ_t, F) -feasibility of $f^{(t)}$. Finally, $Ex_b(f^*) = \lim_{t \rightarrow \infty} Ex_b(f^{(t)}) \leq \lim_{t \rightarrow \infty} n\Delta^t = 0$, and therefore $Ex_b(f^*) = 0$.

To finish the proof, we observe that $f_{ij}^* > 0$. Indeed, $f_{ij} > (2n + m + 1)\Delta$ after the main part of the Δ -phase, and hence $f_{ij} > (2n + m + \frac{1}{2})\Delta$ at the end of the Δ -phase (after performing $\text{ADJUST}(\Delta/2, f, F)$). By the above argument, the total change in all later phases is $\leq 2(2n + m + \frac{1}{2})\Delta_1 = (2n + m + \frac{1}{2})\Delta$, yielding the desired conclusion. \square

Recall that the objective function C_{ij} is called restricted if for some arc ij , $C_{ij}(\alpha) = \infty$ and $C'_{ij}(\alpha) = -\infty$ for $\alpha < 0$.

Claim 5.3. $f_{ij} \geq 0$ holds for every restricted arc ij during the entire algorithm, even if $ij \in F$.

Proof. $f_{ij} \geq 0$ holds at the initialization; consider the first Δ -phase when $f_{ij} < 0$ is attained. This can happen during a path augmentation or in the ADJUST subroutine (EXTEND may not modify f_{ij} as ij is a nonlinear arc). In case of a path augmentation, ji is contained on the shortest path P , and therefore $\pi_i - \pi_j = C'_{ij}(f_{ij} - \Delta)$ must hold for a potential π , a contradiction as $f_{ij} - \Delta < 0$ and thus $C'_{ij}(f_{ij} - \Delta) = -\infty$. A similar argument works for ADJUST . \square

The next lemma is of key importance.

Lemma 5.4. When $\text{TRIAL-AND-ERROR}(F)$ is performed in the Δ -phase, $\text{err}_F(\hat{f}) \leq 2(2n + m + 4)m\Delta$ holds.

Before proving the lemma, we show how it provides the strongly polynomial bound.

Theorem 5.5. The enhanced algorithm terminates in at most $O((n + m_N) \log m)$ scaling phases.

Proof. The set of revealed arcs can be extended at most $m_N + n - 1$ times, since there can be at most $(n - 1)$ linear arcs because of the linear acyclic property. We shall show that after any Δ -phase, a new arc is revealed within $2\lceil \log_2 T \rceil$ phases, for $T = 8(2n + m + 4)m$.

As Δ decreases by at least a factor of two between two phases, after $\lceil \log_2 T \rceil$ steps we have $\Delta_T \leq \Delta/T$. Assume that in the Δ_T phase, we still have the same revealed arc set F as in the Δ -phase.

Assume first $D_b(F) > \Delta$. At the end of the main part of the Δ_T -phase, $D_b(F) > (2n + m + 2)m\Delta_T$. Thus there is an undirected connected component K of F with $|\sum_{i \in K} b_i| > (2n + m + 2)m\Delta_T$. Let $\rho_f(K)$ denote the total f value on arcs entering K minus the value on arcs leaving K , that is,

$$\rho_f(K) := \sum_{ij \in E: i \notin K, j \in K} f_{ij} - \sum_{ij \in E: i \in K, j \notin K} f_{ij}.$$

We have

$$|\rho_f(K)| = \left| \sum_{i \in K} \rho_f(i) \right| = \left| \sum_{i \in K} (\rho_f(i) - b_i + b_i) \right| \geq \left| \sum_{i \in K} b_i \right| - Ex_b(f).$$

The last part is derived from the simple inequality $|\beta + \alpha^+ + \alpha^-| \geq |\beta| - \gamma$, whenever $\alpha^+, \alpha^-, \beta, \gamma \in \mathbb{R}$ with $-\gamma \leq \alpha^- \leq 0 \leq \alpha^+ \leq \gamma$. In our setting, $\beta = \sum_{i \in K} b_i$, $\alpha^+ = \sum_{i \in K} \max\{\rho_f(i) - b_i, 0\}$, $\alpha^- = \sum_{i \in K} \min\{\rho_f(i) - b_i, 0\}$, and $\gamma = Ex_b(f)$. The conditions hold since

$$\gamma = Ex_b(f) = \sum_{i \in V} \max\{\rho_f(i) - b_i, 0\} = - \sum_{i \in V} \min\{\rho_f(i) - b_i, 0\}.$$

Now we may conclude

$$|\rho_f(K)| \geq \left| \sum_{i \in K} b_i \right| - Ex_b(f) > (2n + m + 2)m\Delta_T - n\Delta_T \geq (2n + m + 1)m\Delta_T.$$

Consequently, there must be an arc ij entering or leaving K with $f_{ij} > (2n + m + 1)\Delta_T$, a contradiction as at least one such arc must have been added to F in $\text{EXTEND}(\Delta_T, f, F)$. Note that the first such arc examined during $\text{EXTEND}(\Delta_T, f, F)$ does keep the linear acyclic property as it connects two separate connected components of F .

Assume next $D_b(F) \leq \Delta$. We may assume that either we are at the very beginning of the algorithm with $F = \emptyset$, or in a phase when F just has been extended; otherwise, we could consider an earlier phase with this property. We can interpret the initial solution $\mathbf{0}$ and initial Δ as the output of $\text{TRIAL-AND-ERROR}(\emptyset)$.

If $D_b(F) > \Delta_T$, the above argument shows that within the next $\lceil \log_2 T \rceil$ steps, F shall be extended. Otherwise, we can apply the analysis of the TRIAL-AND-ERROR subroutine for the Δ_T -phase. (Even if the subroutine is not actually performed, its analysis is valid provided that $D_b(F) \leq \Delta_T$.)

Let \hat{f} be the arc set found by $\text{TRIAL}(F, \hat{b})$. This is the same in the Δ and the Δ_T -phase (we may assume that b is modified to \hat{b} always the same way for the same F). In the event of an unsuccessful trial in the Δ -phase, $\Delta/2 < \text{err}_F(\hat{f})$. Using Lemma 5.4 for the Δ_T -phase,

$$\text{err}_F(\hat{f}) \leq 2(2n + m + 4)m\Delta_T \leq \Delta/4 < \text{err}_F(\hat{f})/2,$$

a contradiction. On the other hand, if we had a successful trial in the Δ -phase, then $\Delta_T \leq 2\Delta_{\text{next}}/T$, as Δ_T is the scaling factor $T - 1$ phases after the Δ_{next} -phase. Lemma 5.4 and $Ex_b(\hat{f}) \leq nD_b(F) \leq n\Delta_T$ together yield

$$\Delta_{\text{next}} = \max\{\text{err}_F(\hat{f}), Ex_b(\hat{f})/(2n + m_N)\} \leq 2(2n + m + 4)m\Delta_T \leq \Delta_{\text{next}}/2,$$

a contradiction again. □

Some preparation is needed to prove Lemma 5.4.

Lemma 5.6. *For a linear acyclic arc set $F \subseteq E$, let x and y be two F -tight vectors. Then $\|x - y\|_\infty \leq \|\rho_x - \rho_y\|_1$ holds.*

Proof. First, we claim that the difference graph $D_{x,y} = (V, E_{x,y})$ is acyclic. Indeed, if there existed a cycle $C \subseteq E_{x,y}$, then we get $0 = \sum_{ab \in C} C'_{ab}(x_{ab}) = \sum_{ab \in C} C'_{ab}(y_{ab})$ as in the proof of Lemma 4.1. Since $x_{ab} > y_{ab}$ for every $ab \in C$, this is only possible if all arcs of C are linear (Assumption 2), contradicting the linear acyclic property of F . (Note that $E_{x,y} \subseteq \overleftarrow{F}$, since by definition, every F -tight vector is supported on F).

Define the function z by $z_{ij} = x_{ij} - y_{ij} > 0$ for $ij \in E_{x,y}$ (again with the convention $x_{ji} = -x_{ij}$, $y_{ji} = -y_{ij}$ if $ij \in E$). $\rho_z \equiv \rho_x - \rho_y$, therefore we have to prove $z_{ij} \leq \|\rho_z\|_1$ for $ij \in E_{x,y}$. This property indeed holds for every positive z with acyclic support.

Consider a reverse topological ordering v_1, \dots, v_n of V , where $v_a v_b \in E_{x,y}$ implies $a > b$. For the arc $ij \in E_{x,y}$, let $i = v_{t'}$ and $j = v_t$ ($t' > t$). Let $V_t = \{v_1, \dots, v_t\}$. V_t is a directed cut in $E_{x,y}$, thus

$$\sum_{a > t \geq b} z_{v_a v_b} = \sum_{p \leq t} \rho_z(v_p).$$

As z is positive on all arcs, this implies $z_{v_a v_b} \leq \sum_{p \leq t} \rho_z(v_p) \leq \|\rho_z\|_1$ for all such arcs, in particular, for ij . \square

Claim 5.7. *If f and \hat{f} are F -pseudoflows with $\hat{f}_{ij} = 0$ for $ij \in E \setminus F$, and f is (Δ, F) -feasible, then \hat{f} is $(\Delta + \|f - \hat{f}\|_\infty, F)$ -feasible.*

Proof. There is a potential π so that f and π satisfy (7), that is, $\pi_j - \pi_i \leq C'_{ij}(f_{ij} + \Delta)$ if $ij \in E_f^F(\Delta)$. For $\alpha = \|f - \hat{f}\|_\infty$, we have $f_{ij} + \Delta \leq \hat{f}_{ij} + \Delta + \alpha$. Consequently, (7) is satisfied for (\hat{f}_{ij}, π) and $\Delta + \alpha$ for every arc in $E_f^F(\Delta)$.

By the assumption that \hat{f} is zero outside F , we have $E_{\hat{f}}^F(\Delta + \alpha) = E \cup \overleftarrow{F} \subseteq E_f^F(\Delta)$ and thus the claim follows. \square

Proof of Lemma 5.4. When TRIAL-AND-ERROR is applied, f is (Δ, F) -feasible with some potential π and $Ex_b(f) \leq n\Delta$. We claim that there is an F -tight \bar{f} so that $|\bar{f}_{ij} - f_{ij}| \leq \Delta$ for every $ij \in F$, and $Ex_b(\bar{f}) \leq (2n + m + 2)m\Delta$.

Indeed, (Δ, F) -feasibility gives

$$C'_{ij}(f_{ij} - \Delta) \leq \pi_j - \pi_i \leq C'_{ij}(f_{ij} + \Delta) \quad \forall ij \in F.$$

If ij is a free arc (that is, differentiable on the entire \mathbb{R}), then C'_{ij} is continuous, so there must be a value $f_{ij} - \Delta \leq \beta \leq f_{ij} + \Delta$ with $C'_{ij}(\beta) = \pi_j - \pi_i$. This also holds if ij is a restricted arc, since by Claim 5.3, $f_{ij} \geq 0$ and C'_{ij} is continuous on $(\max\{0, f_{ij} - \Delta\}, f_{ij} + \Delta)$, and $C'_{ij}(0) = -\infty$. Let us set $\bar{f}_{ij} = \beta$. This increases $Ex_b(f)$ by at most $|F|\Delta$.

Let us set $\bar{f}_{ij} = 0$ for $ij \in E \setminus F$. Note that $f_{ij} \leq (2n + m + 1)\Delta$ if $ij \notin F$ (every arc with $f_{ij} > (2n + m + 1)\Delta$ is either added to F or is modified to $f_{ij} = 0$ in the subroutine EXTEND). Further, $Ex_b(f) \leq n\Delta$, and thus we obtain an F -tight \bar{f} with

$$\begin{aligned} Ex_b(\bar{f}) &\leq n\Delta + |F|\Delta + (2n + m + 1)(m - |F|)\Delta \\ &\leq (2n + m + 2)m\Delta. \end{aligned}$$

On the other hand, $Ex_b(\hat{f}) \leq nD_b(F) \leq n\Delta$, since $Ex_b(\hat{f}) = 0$ and \hat{b} is obtained from b by modifying certain values by $\leq D_b(F)$. Consequently,

$$\|\rho_{\bar{f}} - \rho_{\hat{f}}\|_1 \leq \|\rho_{\bar{f}} - b\|_1 + \|\rho_{\hat{f}} - b\|_1 = 2Ex_b(\bar{f}) + 2Ex_b(\hat{f}) \leq 2(2n + m + 3)m\Delta.$$

Applying Lemma 5.6 for $x = \bar{f}$ and $y = \hat{f}$ gives $\|\bar{f} - \hat{f}\|_\infty \leq 2(2n + m + 3)m\Delta$. Now \bar{f} is $2(2n + m + 4)m\Delta$ -feasible by Claim 5.7. \square

Theorem 5.8. *Let $\rho_S(n, m)$ be the running time needed for one shortest path computation. Then the running time of the algorithm is bounded by*

$$O((n + m_N)(\rho_T(n, m) + \rho_E(n, m)) + (n + m_N)^2 \rho_S(n, m) \log m).$$

Proof. By Theorem 5.5, there are at most $(n + m_N) \log m$ scaling phases, each dominated by $O(n + m_N)$ shortest path computations. The subroutine TRIAL-AND-ERROR is performed only when F is extended, that is, at most $n + m_N$ times, and performs the subroutines TRIAL and ERROR. \square

We may not use $O(\log n) = O(\log m)$ as the graph is allowed to contain parallel arcs.

5.1 Shortest path computations

For the sake of efficiency, we shall maintain a potential vector π during the entire algorithm such that (f, π) satisfies the condition (7) on (Δ, F) -feasibility.

For the initial Δ value, $\Delta \geq \text{err}_\emptyset(\mathbf{0})$, and the latter value is computed by $\text{ERROR}(\mathbf{0}, \emptyset)$. This means that $f = \mathbf{0}$ is (Δ, \emptyset) -feasible. Similarly, after every successful trial we have a new flow f and value $\Delta \geq \text{err}_F(f)$, computed by $\text{ERROR}(f, F)$. In the applications, this subroutine will also return a potential vector π such that (f, π) satisfies (7).

Alternatively, such a potential vector may be obtained by the standard label correcting algorithm (see [1, Chapter 5.5]), since it is a dual proof of the fact that the graph $E_f^F(\Delta)$ contains no negative cycles with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$; we have access to these values via the value oracle in Assumption 1.

In the main part of the Δ -phase, we may apply Dijkstra's algorithm (see [1, Chapter 4.5]) to compute shortest paths. This needs a nonnegative cost function, but instead of the original $C'_{ij}(f_{ij} + \Delta)$ that may take negative values, we shall use $C'_{ij}(f_{ij} + \Delta) - \pi_j + \pi_i$, a nonnegative function by (7); the set of shortest paths is identical for the two costs. This subroutine can be implemented by updating the potentials π such that finally we get $C'_{ij}(f_{ij} + \Delta) = \pi_j - \pi_i$ on every arc of every shortest path. For the sake of completeness, we describe this subroutine in the Appendix.

As shown in the proof of Lemma 5.1(ii), sending Δ -units of flow on a shortest path maintains (7) for (f, π) . It is also maintained in $\text{EXTEND}(\Delta, f, F)$ since flow values are modified only on arcs with C'_{ij} constant. Finally, $\text{ADJUST}(\Delta/2, f, F)$ modifies the flow so that (7) is maintained for the same π and $\Delta/2$ by Lemma 4.4.

Let us now explore the relation to Assumption 1. In both applications, we shall verify that the subroutine TRIAL-AND-ERROR returns a rational flow vector f and a rational value Δ . Since flow will always be modified in units of Δ in all other parts of the algorithm, we may conclude that a rational f will be maintained in all other parts. Under Assumption 1(a) (i.e., quadratic objectives), we shall maintain a rational potential vector π , while under Assumption 1(b) (i.e., Fisher markets), we shall maintain the rationality of the e^{π_i} values; during the computations, we shall use the representation of these values instead of the original π . For this aim, we will use a multiplicative version of Dijkstra's algorithm, also described in the Appendix. We shall also verify that in the corresponding applications, the subroutine $\text{ERROR}(f, F)$ returns a potential vector π so that (f, π) satisfies (7), with the π_i or the e^{π_i} values being rational, respectively.

Finally, it is easy to verify that whereas we are working on a transformed uncapacitated instance, we may use the complexity bound of the original instance, as summarized in the following remark.

Remark 5.9. *A shortest path computation can be performed in time $\rho_S(n, m) = O(m + n \log n)$ using Fibonacci heaps, see [9]. Recall that the original problem instance was on n' nodes and m' arcs, and it was transformed to an uncapacitated instance on $n = n' + m'$ nodes and $m = 2m'$ arcs. However, as in Orlin's algorithm [29], we can use the bound $O(m' + n' \log n')$ instead of $O(m' + m' \log n')$ because shortest path computations can be essentially performed on the original network.*

6 Applications

6.1 Quadratic convex costs

Assume that $C_{ij}(\alpha) = c_{ij}\alpha^2 + d_{ij}\alpha$ for each $ij \in E$, with $c_{ij} \geq 0$. This clearly satisfies Assumption 1(i) since $C'_{ij}(\alpha) = 2c_{ij}\alpha + d_{ij}$. Also, Assumption 2 is satisfied: ij is linear if $c_{ij} = 0$.

The subroutine $\text{TRIAL}(F, b)$ can be implemented by solving a system of linear equations.

$$\begin{aligned} \pi_j - \pi_i &= 2c_{ij}x_{ij} + d_{ij} \quad \forall ij \in F \\ \sum_{j:ji \in F} x_{ji} - \sum_{j:ij \in F} x_{ij} &= b_i \quad \forall i \in V \\ x_{ij} &= 0 \quad \forall ij \in E \setminus F \end{aligned} \tag{10}$$

Assumption 3 is verified by the next claim.

Lemma 6.1. *If F is linear acyclic and $D_b(F) = 0$, then (10) is feasible and a solution can be found in $\rho_T(n, m) = O(n^{2.37})$ time.*

Proof. Clearly, we can solve the system separately on different undirected connected components of F . In the sequel, let us focus on a single connected component; for simplicity of notation, assume this component is the entire V .

Consider first the case when all arcs are linear. Then we can solve the equalities corresponding to edges and nodes separately. As F is assumed to be linear acyclic, it forms a tree. If we fix one π_j value arbitrarily, it determines all other π_i values by moving along the edges in the tree. The x_{ij} 's can be found by solving a flow problem on the same tree with the demands b_i . This is clearly feasible by the assumption $D_b(F) = 0$, that is, $\sum_{i \in V} b_i = 0$ (note that we do not have nonnegativity constraints on the arcs). Both tasks can be performed in linear time.

Assume next both linear and nonlinear arcs are present, and let T be an undirected connected component of linear arcs. As above, all $\pi_j - \pi_i$ values for $i, j \in T$ are uniquely determined. If there is a nonlinear arc $ij \in F$ with $i, j \in T$, then $x_{ij} = (\pi_j - \pi_i - d_{ij})/(2c_{ij}) = \alpha$ is also uniquely determined. We can remove this edge by replacing b_i by $b_i + \alpha$ and b_j by $b_j - \alpha$. Hence we may assume that the components of linear arcs span no nonlinear arcs.

Next, we can contract each such component T to a single node t by setting $b_t = \sum_{i \in T} b_i$ and modifying the d_{ij} values on incident arcs appropriately. A solution to the contracted problem can be easily extended to the original instance.

For the rest, we can assume all arcs are nonlinear, that is, $c_{ij} > 0$ for all $ij \in F$. Let A be the node-arc incidence matrix of F : $A_{i,ij} = -1$, $A_{i,ji} = 1$ for all $ij \in F$, and all other entries are 0. Let C be the $|F| \times |F|$ diagonal matrix with $C_{ij,ij} = -2c_{ij}$. (10) can be written in the form

$$\begin{pmatrix} A^T & C \\ 0 & A \end{pmatrix} (\pi, x) = \begin{pmatrix} d \\ b \end{pmatrix}.$$

This can be transformed into

$$\begin{pmatrix} A^T & C \\ L & 0 \end{pmatrix} (\pi, x) = \begin{pmatrix} d \\ b' \end{pmatrix},$$

where L is the weighted Laplacian matrix with $L_{ii} = \sum_{j:ij \in F} \frac{1}{2c_{ij}}$, $L_{ij} = L_{ji} = -\frac{1}{2c_{ij}}$ if $ij \in F$ and $L_{ij} = 0$ otherwise, and b' is an appropriate vector with $\sum_{i \in V} b'_i = 0$.

The main task is to solve the system $L\pi = b'$. It is well-know (recall that V is assumed to be a single connected component) that L has rank $|V| - 1$ and the system is always feasible whenever

$\sum_{i \in V} b'_i = 0$. A solution can be found in $O(n^{2.37})$ time [3]. All previously described operations (eliminating nonlinear arcs spanned in components of linear arcs, contracting components of linear arcs) can be done in $O(m)$ time, hence the bound $\rho_T(n, m) = O(n^{2.37})$. \square

To implement $\text{ERROR}(f, F)$, we have to find the minimum Δ -value such that there exists a π potential with

$$\pi_j - \pi_i \leq (2c_{ij}f_{ij} + d_{ij}) + 2c_{ij}\Delta \quad \forall ij \in E \cup \overleftarrow{F}. \quad (11)$$

We show that this can be reduced to the minimum-cost-to-time ratio cycle problem, defined as follows (see [1, Chapter 5.7]). In a directed graph, there is a cost function p_{ij} and a time $\tau_{ij} \geq 0$ associated with each arc. The aim is to find a cycle C minimizing $(\sum_{ij \in C} p_{ij}) / (\sum_{ij \in C} \tau_{ij})$. A strongly polynomial algorithm was given by Megiddo [23, 24] that solves the problem in $\min\{O(n^3 \log^2 n), O(n \log n(n^2 + m \log \log n))\}$ time. The problem can be equivalently formulated as

$$\begin{aligned} \min \mu \text{ s. t. } & \text{there are no negative cycles} \\ & \text{for the cost function } p_{ij} + \mu\tau_{ij}. \end{aligned} \quad (12)$$

Our problem fits into this framework with $p_{ij} = 2c_{ij}f_{ij} + d_{ij}$ and $\tau_{ij} = 2c_{ij}$. In (12), the optimal μ value is $-\Delta$. However, [23] defines the minimum ratio cycle problem with $\tau_{ij} > 0$ for every $ij \in E$. This property is not essential for Megiddo's algorithm, which uses a parametric search method for μ to solve (12) under the only (implicit) restriction that the problem is feasible.

In our setting $\tau_{ij} > 0$ holds for nonlinear arcs, but $\tau_{ij} = 0$ for linear arcs. Also, there can be cycles C with $\sum_{ij \in C} \tau_{ij} = 0$. (This can happen even if F is linear acyclic, as C can be any cycle in $E \cup \overleftarrow{F}$.) If we have such a cycle C with $\sum_{ij \in C} p_{ij} < 0$, then (12) is infeasible. In every other case, the problem is feasible and thus Megiddo's algorithm can be applied.

For this reason, we first check whether there is a negative cycle with respect to the p_{ij} 's in the set of linear arcs in $E \cup \overleftarrow{F}$. This can be done via the label correcting algorithm in $O(nm)$ time ([1, Chapter 5.5]). If there exists one, then (11) is infeasible, thus $\text{err}_F(f) = \Delta = \infty$, and (P) is unbounded as we can send arbitrary flow around this cycle. Otherwise, we have $\sum_{ij \in C} \tau_{ij} > 0$ for every cycle with $\sum_{ij \in C} p_{ij} < 0$, and consequently, there exists a finite Δ satisfying (11).

Consequently, $\rho_T(n, m) = \min\{O(n^3 \log^2 n), O(n \log n(n^2 + m \log \log n))\}$. Theorem 5.8 gives the following running time bound.

Theorem 6.2. *For convex quadratic objectives on an uncapacitated instance on n nodes and m arcs, the algorithm finds an optimal solution in $O(m(n^3 \log^2 n + m \log m(m + n \log n)))$. For a capacitated instance, the running time can be bounded by $O(m^4 \log m)$.*

The bottleneck is clearly the m minimum-cost-to-time computations. As in Remark 5.9, it is likely that one can get the same running time $O(m(n^3 \log^2 n + m \log m(m + n \log n)))$ for capacitated instances via a deeper analysis of Megiddo's algorithm.

Let us verify that the algorithm is strongly polynomial. It uses elementary arithmetic operations only, and the running time is polynomial in n and m , according to the above theorem. It is left to verify property (iii): if all numbers in the input are rational, then every number occurring in the computations is rational and is of size polynomially bounded in the size of the input. At the initialization and in every successful trial, we compute a new flow f by solving (10) as described in Lemma 6.1, and compute the new Δ and π values by Megiddo's algorithm. These are strongly polynomial subroutines and return rational values of size polynomially bounded in the input. In phases between the initialization and the first successful trial, and between any two later successful trials, Δ is always decreased by a factor of 2 between two phases, and f is modified on every arc by

$\pm\Delta$ in a path augmentation. The π values are updated by the Dijkstra subroutine described in the Appendix. All these updates keep rationality if the input is rational. Since the number of phases between two successful trials is $O(\log m)$, the sizes of numbers remain polynomially bounded.

6.2 Fisher's market with linear utilities

In the *linear Fisher market model*, we are given a set B of buyers and a set G of goods. Buyer i has a budget m_i , and there is one divisible unit of each good to be sold. For each buyer $i \in B$ and good $j \in G$, $U_{ij} \geq 0$ is the utility accrued by buyer i for one unit of good j . Let $n = |B| + |G|$; let E be the set of pairs (i, j) with $U_{ij} > 0$ and let $m = |E|$. We assume that there is at least one edge in E incident to every buyer and to every good.

An equilibrium solution consist of prices p_i on the goods and an allocation x_{ij} , so that (i) all goods are sold, (ii) all money of the buyers is spent, and (iii) each buyer i buys a best bundle of goods, that is, goods j maximizing U_{ij}/p_j .

The classical convex programming formulation of this problem was given by Eisenberg and Gale [7]. Recently, Shmyrev [32] gave the following alternative formulation. The variable f_{ij} represents the money paid by buyer i for product j .

$$\begin{aligned} \min \sum_{i \in G} p_j (\log p_j - 1) - \sum_{ij \in E} f_{ij} \log U_{ij} \\ \sum_{j \in G} f_{ij} = m_i \quad \forall i \in B \\ \sum_{i \in B} f_{ij} = p_j \quad \forall j \in G \\ f_{ij} \geq 0 \quad \forall ij \in E \end{aligned}$$

Let us construct a network on node set $B \cup G \cup \{t\}$ as follows. Add an arc ij for every $ij \in E$, and an arc jt for every $j \in G$. Set $b_i = -m_i$ for $i \in B$, $b_j = 0$ for $j \in G$ and $b_t = \sum_{i \in B} m_i$. Let all lower arc capacities be 0 and upper arc capacities ∞ . With p_j representing the flow on arc jt , the above formulation is a minimum-cost flow problem with separable convex objective. (The arc jt is restricted, with extending the functions $p_j(\log p_j - 1)$ to take value 0 in 0 and ∞ on $(-\infty, 0)$. All other arcs are free; indeed, they are linear.) In this section, the convention $p_j = f_{jt}$ shall be used for some pseudoflow f in the above problem.

Let us justify that an optimal solution gives a market equilibrium. Let f be an optimal solution that satisfies (3) with $\pi : B \cup G \cup \{t\} \rightarrow \mathbb{R}$. We may assume $\pi_t = 0$. $C'_{jt}(\alpha) = \log \alpha$ implies $\pi_j = -\log p_j$. On each $ij \in E$ we have $\pi_j - \pi_i \leq -\log U_{ij}$ with equality if $f_{ij} > 0$. With $\beta_i = e^{\pi_i}$, this is equivalent to $U_{ij}/p_j \leq \beta_i$, verifying that every buyer receives a best bundle of goods.

Assumption 1(b) is valid, since the derivatives on arcs ij between buyers and goods are $-\log U_{ij}$, while on an arc jt it is $\log f_{jt}$. Assumption 2 is straightforward.

Let us turn to Assumption 3. When the subroutine TRIAL is called, we transform b to \hat{b} by changing the value at one node of each component K of F . For simplicity, let us always modify b_t if $t \in K$. We shall verify Assumption 3 only for such \hat{b} 's; the argument can easily be extended to arbitrary \hat{b} (although it is not necessary for the algorithm). Let us call the component K containing t the *large component*.

In TRIAL(F), we want to find a potential $\pi : B \cup G \cup \{t\} \rightarrow \mathbb{R} \cup \{\infty\}$, money allocations f_{ij} for

$ij \in F$, $i \in B$, $j \in G$, and prices $p_j = f_{jt}$ for $jt \in F$ such that

$$\begin{aligned}\pi_j - \pi_i &= -\log U_{ij} \quad \forall ij \in F, i \in B, j \in G \\ \pi_t - \pi_j &= \log p_j \quad \forall jt \in F \\ p_j &= \sum_{i \in B, ij \in F} f_{ij} \quad \forall jt \in F \\ p_j &= -\hat{b}_j + \sum_{i \in B, ij \in F} f_{ij} \quad \forall jt \in E \setminus F\end{aligned}$$

We may again assume $\pi_t = 0$. Let $P_j = e^{-\pi_j}$ for $j \in G$ and $R_i = e^{\pi_i}$ for $i \in B$. With this notation, $U_{ij}/P_j = R_i$ for $ij \in F$. If $jt \in F$, then $P_j = p_j$.

Finding f and π can be done independently on the different components of F . For any component different from the large one, all edges are linear. Therefore we only need to find a feasible flow on a tree, and independently, P_j and R_i values satisfying $U_{ij}/P_j = R_i$ on arcs ij in this component. Both of these can be performed in linear time in the number of edges in the tree. Note that multiplying each P_j by a constant $\alpha > 0$ and dividing each R_i by the same α yields another feasible solution.

Let T_1, \dots, T_k be the components of the large component after deleting t . If T_ℓ contains a single good j , then we set $p_j = P_j = 0$ ($\pi_j = 0$). If T_ℓ is nonsingular, then F restricted to T_ℓ forms a spanning tree. The equalities $U_{ij}/P_j = R_i$ uniquely define the ratio $P_j/P_{j'}$ for any $j, j' \in G \cap T_\ell$. Using that $p_j = P_j$ and $\sum_{i \in B \cap T_\ell} m_i = \sum_{j \in G \cap T_\ell} p_j$, this uniquely determines the prices in the component. Then the edges in F simply provide the allocations f_{ij} . All these computations can be performed in $\rho_T(n, m) = O(m)$ time.

For Assumption 4, we show that $\text{ERROR}(f, F)$ can be implemented based on the Floyd-Warshall algorithm (see [1, Chapter 5.6]). Let π be the potential witnessing that f is (Δ, F) -feasible. Assuming $\pi_t = 0$, and using again the notation $P_j = e^{-\pi_j}$ for $j \in G$ and $R_i = e^{\pi_i}$ for $i \in B$, we get

$$U_{ij}/P_j \leq R_i \text{ if } i \in B, j \in G, ij \in E, \text{ with equality if } ji \in E_f^F. \quad (13)$$

Furthermore, we have $p_j - \Delta \leq P_j \leq p_j + \Delta$ if $p_j > 0$ and $P_j \leq \Delta$ if $p_j = 0$.

Let us now define $\beta : G \times G \rightarrow \mathbb{R}$ as

$$\beta_{jj'} = \max \left\{ \frac{U_{ij'}}{U_{ij}} : i \in B, ji, ij' \in E_f^F \right\}.$$

If no such i exists, define $\beta_{jj'} = 0$; let $\beta_{jj} = 1$ for every $j \in G$.

Claim 6.3. *Assume we are given some P_j values, $j \in G$. There exists R_i values ($i \in B$) satisfying (13) if and only if $P_{j'} \geq P_j \beta_{jj'}$ holds for every $j, j' \in G$.*

Proof. The condition is clearly necessary by the definition of $\beta_{jj'}$. Conversely, if this condition holds, setting $R_i = \max_{j \in G} U_{ij}/P_j$ does satisfy (13). \square

If there is a directed cycle C with $\prod_{ab \in C} \beta_{ab} > 1$, then f cannot be (Δ, F) -feasible for any Δ . Otherwise, we may compute $\tilde{\beta}_{jj'}$ as the maximum of $\prod_{ab \in P} \beta_{ab}$ over all directed paths P in E_f^F from j to j' (setting the value 0 again if no such path exists). This can be done by the multiplicative version of the Floyd-Warshall algorithm in $O(n^3)$ time (note that this is equivalent to finding all-pair shortest paths for $-\log \beta_{ab}$).

For (Δ, F) -feasibility, we clearly need to satisfy

$$(p_j - \Delta)\tilde{\beta}_{jj'} \leq P_j \tilde{\beta}_{jj'} \leq P_{j'} \leq p_{j'} + \Delta.$$

Let us define Δ as the smallest value satisfying all these inequalities, that is,

$$\Delta = \max \left\{ 0, \max_{j, j' \in G} \frac{p_j \tilde{\beta}_{jj'} - p_{j'}}{\tilde{\beta}_{jj'} + 1} \right\}. \quad (14)$$

We claim that f is (Δ, F) -feasible with the above choice. For each $j \in G$, let $P_j = \max_{h \in G} \tilde{\beta}_{hj}(p_h - \Delta)$. It is easy to verify that these P values satisfy $P_{j'} \geq P_j \beta_{jj'}$, and $p_j - \Delta \leq P_j \leq p_j + \Delta$. The condition (13) follows by Claim 6.3.

The complexity of $\text{ERROR}(f, F)$ is dominated by the Floyd-Warshall algorithm, $O(n^3)$ [8]. The problem is defined on an uncapacitated network, with the number of nonlinear arcs $m_N = |G| < n$. Thus Theorem 5.8 gives the following.

Theorem 6.4. *For Fisher's market with linear utilities, the algorithm finds an optimal solution in $O(n^4 + n^2(m + n \log n) \log n)$.*

The algorithm of Orlin [30] runs in $O(n^4 \log n)$ time, assuming $m = O(n^2)$. Under this assumption, we get the same running time bound.

To prove that the algorithm is strongly polynomial, let us verify the nontrivial condition (iii). As discussed in Section 5.1, if the input is rational, we shall maintain that f , Δ and the e^{π_i} values are rational; the latter are used in the computations instead of the π_i 's. At the initialization and in every successful trial, the subroutines described above are strongly polynomial and therefore return rational f , Δ and e^{π_i} values, of size polynomially bounded in the input (note that the e^{π_i} values above are denoted by P_i for $i \in G$ and R_i for $i \in B$, and $e^{\pi_t} = 1$). Between two successful trials, Δ is always divided by two between two phases and the path augmentations change f by $\pm\Delta$; the multiplicative Dijkstra algorithm described in the Appendix also maintains rational e^{π_i} values. The sizes of the numbers remain polynomially bounded in n and m since the number of phases between two successful trials is $O(\log m)$.

6.3 Fisher's market with spending constraint utilities

The *spending constraint utility* extension of linear Fisher markets was defined by Vazirani [36]. In this model, the utility of a buyer decreases as the function of the money spent on the good. Formally, for each pair i and j there is a sequence $U_{ij}^1 > U_{ij}^2 > \dots > U_{ij}^{\ell_{ij}} > 0$ of utilities with numbers $L_{ij}^1, \dots, L_{ij}^{\ell_{ij}} > 0$. Buyer i accrues utility U_{ij}^1 for every unit of j he purchased by spending the first L_{ij}^1 dollars on good j , U_{ij}^2 for spending the next L_{ij}^2 dollars, etc. These ℓ_{ij} intervals corresponding to the pair ij are called *segments*. $\ell_{ij} = 0$ is allowed, but altogether at least one segment is required to be incident to each good i and to each buyer j . Let $n = |B| + |G|$ denote the total number of buyers and goods, and m denote the total number of segments. Note that $m > n^2$ is also possible.

No extension of the Eisenberg-Gale convex program is known to capture this problem. The existence of a convex programming formulation is left as an open question in [36]. This was settled by Devanur et al. [2], giving a convex program based on Shmyrev's formulation. Let f_{ij}^k represent

the money paid by buyer i for the k 'th segment of product j , $1 \leq k \leq \ell_{ij}$.

$$\begin{aligned} \min \sum_{i \in G} p_j (\log p_j - 1) - \sum_{i \in B, j \in G, 1 \leq k \leq \ell_{ij}} f_{ij}^k \log U_{ij}^k \\ \sum_{j \in G, 1 \leq k \leq \ell_{ij}} f_{ij}^k = m_i \quad \forall i \in B \\ \sum_{i \in B, 1 \leq k \leq \ell_{ij}} f_{ij}^k = p_j \quad \forall j \in G \\ 0 \leq f_{ij}^k \leq L_{ij}^k \quad \forall ij \in E. \end{aligned}$$

This gives a convex cost flow problem again on the node set $B \cup G \cup \{t\}$, by adding ℓ_{ij} parallel arcs from $i \in B$ to $j \in G$, and arcs jt for each $j \in G$. The upper capacity on the k 'th segment for the pair ij is L_{ij}^k . To apply our method, we first need to transform it to an equivalent problem without upper capacities. This is done by replacing the arc representing the k 'th segment of ij by a new node (ij, k) and two arcs $i(ij, k)$ and $j(ij, k)$. The node demand on the new node is set to L_{ij}^k , while on the good j , we replace the demand 0 by $-\sum_{i,k} L_{ij}^k$, the negative of the sum of capacities of all incident segments. The cost function on $i(ij, k)$ is $-\log U_{ij}^k \alpha$, while the cost of $j(ij, k)$ is 0. Let S denote the set of the new (ij, k) nodes. This modified graph has $n' = n + m + 1$ nodes and $m' = 2m + |G|$ arcs.

Assumptions 1 and 2 are satisfied the same way as for linear Fisher markets, using an oracle for the $e^{C'_{ij}(\alpha)}$ values.

In $\text{TRIAL}(F)$, we want to find an F -tight flow f' on the extended network, witnessed by the potential $\pi : B \cup S \cup G \cup \{t\} \rightarrow \mathbb{R}$. We may assume $\pi_t = 0$. Let $P_j = e^{-\pi_j}$ for $j \in G$ and $R_i = e^{\pi_i}$ for $i \in B$ and $S_{ij}^k = e^{-\pi(ij,k)}$. For the k 'th segment of ij , $U_{ij}^k/S_{ij}^k = R_i$ if $i(ij, k) \in F$ and $S_{ij}^k = P_j$ if $(ij, k)j \in F$.

As for linear Fisher markets, if a component of F does not contain t , we can simply compute all potentials and flows as F is a spanning tree of linear edges in this component.

For the component K with $t \in K$, let T_ℓ be a component of $K - t$. F is a spanning tree of linear edges in T_ℓ as well, therefore the ratio $P_j/P_{j'}$ is uniquely defined for any $j, j' \in G \cap T_\ell$. On the other hand, we must have $P_j = p_j$, and we know that $\sum_{j \in G \cap T_\ell} p_j = -\sum_{v \in T_\ell} b_v$ by flow conservation. These determine the $P_j = p_j$ values, and thus all other R_i and S_{ij}^k values in the component as well. The support of the flow f_{ij} is a tree and hence it can also easily be computed. The running time of TRIAL is again linear, $\rho_T(n', m') = O(m') = O(m)$.

$\text{ERROR}(f, F)$ can be implemented the same way as for the linear Fisher market. We shall define the values $\beta : G \times G \rightarrow \mathbb{R}$ so that $P_{j'} \geq P_j \beta_{jj'}$ must hold, and conversely, given P_j prices satisfying these conditions, we can define the R_i and S_{ij}^k values feasibly. Let

$$\beta_{jj'} = \max \left\{ \frac{U_{ij'}^{k'}}{U_{ij}^k} : i \in B, \right. \\ \left. j(ij, k), (ij, k)i, i(ij', k'), (ij', k')j' \in E_f^F \right\}.$$

Given these $\beta_{jj'}$ values, the $\tilde{\beta}_{jj'}$ values can be computed by the Floyd-Warshall algorithm and the optimal Δ obtained by (14) as for the linear case.

Finding the $\beta_{jj'}$ values can be done in $O(m')$ time, and the Floyd-Warshall algorithm runs in $O(|G|^3)$. This gives $\rho_E(n', m') = O(m' + |G|^3) = O(m + n^3)$. From Theorem 5.8, together with Remark 5.9, we obtain:

Theorem 6.5. *For an instance of Fisher’s market with spending constraint utilities with $n = |B| + |G|$ and m segments, the running time can be bounded by $O(mn^3 + m^2(m + n \log n) \log m)$.*

It can be verified that the algorithm is strongly polynomial the same way as for the linear case.

7 Discussion

We have given strongly polynomial algorithms for a class of minimum-cost flow problems with separable convex objectives. This gives the first strongly polynomial algorithms for quadratic convex cost functions and for Fisher’s market with spending constraint utilities. For Fisher’s market with linear utilities, we get the same complexity as in [30].

The bottleneck in complexity of all applications is the subroutine TRIAL. However, the exact value of $err_F(f)$ is not needed: a constant approximation would also yield the same complexity bounds. Unfortunately, no such algorithm is known for the minimum cost-to-time ratio cycle problem that would have significantly better, strongly polynomial running time. Finding such an algorithm would immediately improve the running time for quadratic costs.

A natural future direction could be to develop strongly polynomial algorithms for quadratic objectives and constraint matrices with bounded subdeterminants. This would be a counterpart of Tardos’s result [35] for linear programs. Such an extension could be possible by extending our techniques to the setting of Hochbaum and Shantikumar [18].

The recent paper [38] shows that linear Fisher market, along with several extension, can be captured by a concave extension of the generalized flow model. A natural question is if there is any direct connection between the concave generalized flow model and the convex minimum cost flow model studied in this paper. Despite certain similarities, no reduction is known in any direction. Indeed, no such reduction is known even between the linear special cases, that is, generalized flows and minimum-cost flows. The perfect price discrimination model [11], and the Arrow-Debreu Nash-bargaining problem [37], are instances of the concave generalized flow model, but they are not known to be reducible to convex cost flows. On the other hand, the spending constraint utility model investigated in this paper is not known to be reducible to concave generalized flows.

The algorithm in [38] is not strongly polynomial. Moreover, no strongly polynomial algorithm is known for linear generalized flows, despite the huge literature on polynomial time algorithms. Developing a strongly polynomial algorithm for generalized flows is a fundamental open question. Resolving it could lead to strongly polynomial algorithms for the market problems that fit into the concave generalized flow model.

A related problem is finding a strongly polynomial algorithm for minimizing a separable convex objective over a submodular polyhedron. Fujishige [10] showed that for separable convex quadratic costs, this is essentially equivalent to submodular function minimization. Submodular utility allocation markets by Jain and Vazirani [21] also fall into this class, and are solvable in strongly polynomial time; see also Nagano [27]. Other strongly polynomially solvable special cases are given by Hochbaum and Hong [14].

A common generalization of this problem and ours is minimizing a separable convex objective over a submodular flow polyhedron. Weakly polynomial algorithms were given by Iwata [19] and by Iwata, McCormick and Shigeno [20]. One might try to develop strongly polynomial algorithms for some class of separable convex objectives; in particular, for separable convex quadratic functions.

Acknowledgment

The author is grateful to an anonymous referee for several suggestions that helped to improve the presentation.

References

- [1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., Upper Saddle River, New Jersey, feb 1993.
- [2] B. Birnbaum, N. R. Devanur, and L. Xiao. Distributed algorithms via gradient descent for Fisher markets. In *Proceedings of ACM EC*, pages 127–136, 2011.
- [3] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.
- [4] S. Cosares and D. S. Hochbaum. Strongly polynomial algorithms for the quadratic transportation problem with a fixed number of sources. *Mathematics of Operations Research*, 19(1):94–111, 1994.
- [5] N. R. Devanur, C. H. Papadimitriou, A. Saberi, and V. V. Vazirani. Market equilibrium via a primal–dual algorithm for a convex program. *Journal of the ACM (JACM)*, 55(5):22, 2008.
- [6] J. Edmonds and R. M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of the ACM (JACM)*, 19(2):248–264, 1972.
- [7] E. Eisenberg and D. Gale. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics*, 30(1):165–168, 1959.
- [8] R. Floyd. Algorithm 97: shortest path. *Communications of the ACM*, 5(6):345, 1962.
- [9] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987.
- [10] S. Fujishige. Submodular systems and related topics. *Mathematical Programming at Oberwolfach II*, (22):113–131, 1984.
- [11] G. Goel and V. V. Vazirani. A perfect price discrimination market model with production, and a (rational) convex program for it. *Mathematics of Operations Research*, 36:762–782, 2011.
- [12] F. Granot and J. Skorin-Kapov. Towards a strongly polynomial algorithm for strictly convex quadratic programs: An extension of Tardos’ algorithm. *Mathematical Programming*, 46(1):225–236, 1990.
- [13] M. Grotschel, L. Lovász, and A. Schrijver. Geometric algorithms and combinatorial optimizations. 1993.
- [14] D. Hochbaum and S. Hong. About strongly polynomial time algorithms for quadratic optimization over submodular constraints. *Mathematical Programming*, 69(1):269–309, 1995.
- [15] D. Hochbaum and M. Queyranne. Minimizing a convex cost closure set. *SIAM Journal on Discrete Mathematics*, 16:192, 2003.

- [16] D. S. Hochbaum. Lower and upper bounds for the allocation problem and other nonlinear optimization problems. *Mathematics of Operations Research*, 19(2):390–409, 1994.
- [17] D. S. Hochbaum. Complexity and algorithms for nonlinear optimization problems. *Annals of Operations Research*, 153(1):257–296, 2007.
- [18] D. S. Hochbaum and J. G. Shanthikumar. Convex separable optimization is not much harder than linear optimization. *Journal of the ACM (JACM)*, 37(4):843–862, 1990.
- [19] S. Iwata. A capacity scaling algorithm for convex cost submodular flows. *Mathematical Programming*, 76(2):299–308, 1997.
- [20] S. Iwata, S. McCormick, and M. Shigeno. Fast cycle canceling algorithms for minimum cost submodular flow. *Combinatorica*, 23(3):503–525, 2003.
- [21] K. Jain and V. V. Vazirani. Eisenberg-Gale markets: Algorithms and game-theoretic properties. *Games and Economic Behavior*, 70(1):84–106, 2010.
- [22] A. V. Karzanov and S. T. McCormick. Polynomial methods for separable convex optimization in unimodular linear spaces with applications. *SIAM J. Comput.*, 26(4):1245–1275, 1997.
- [23] N. Megiddo. Combinatorial optimization with rational objective functions. *Mathematics of Operations Research*, 4(4):414–424, 1979.
- [24] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM (JACM)*, 30(4):852–865, 1983.
- [25] M. Minoux. A polynomial algorithm for minimum quadratic cost flow problems. *European Journal of Operational Research*, 18(3):377–387, 1984.
- [26] M. Minoux. Solving integer minimum cost flows with separable convex cost objective polynomially. *Mathematical Programming Study*, 25:237, 1985.
- [27] K. Nagano. On convex minimization over base polytopes. *Integer Programming and Combinatorial Optimization*, pages 252–266, 2007.
- [28] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press New York, NY, USA, 2007.
- [29] J. B. Orlin. A faster strongly polynomial minimum cost flow algorithm. *Operations Research*, 41(2):338–350, 1993.
- [30] J. B. Orlin. Improved algorithms for computing Fisher’s market clearing prices. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 291–300. ACM, 2010.
- [31] J. Renegar. On the worst-case arithmetic complexity of approximating zeros of polynomials. *Journal of Complexity*, 3(2):90–113, 1987.
- [32] V. I. Shmyrev. An algorithm for finding equilibrium in the linear exchange model with fixed budgets. *Journal of Applied and Industrial Mathematics*, 3(4):505–518, 2009.
- [33] A. Tamir. A strongly polynomial algorithm for minimum convex separable quadratic cost flow problems on series-parallel networks. *Mathematical Programming*, 59:117–132, 1993.

- [34] É. Tardos. A strongly polynomial minimum cost circulation algorithm. *Combinatorica*, 5(3):247–255, 1985.
- [35] É. Tardos. A strongly polynomial algorithm to solve combinatorial linear programs. *Operations Research*, 34(2):250–256, 1986.
- [36] V. V. Vazirani. Spending constraint utilities with applications to the adwords market. *Mathematics of Operations Research*, 35(2):458–478, 2010.
- [37] V. V. Vazirani. The notion of a rational convex program, and an algorithm for the Arrow-Debreu Nash bargaining game. *Journal of ACM (JACM)*, 59(2), 2012.
- [38] L. A. Végh. Concave generalized flows with applications to market equilibria. In *Proceedings of the 53rd Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 2012.

Appendix

In this Appendix we describe two versions of Dijkstra’s algorithm that are used for the shortest path computations in our algorithm. This is an equivalent description of the well-known algorithm, see e.g. [1, Chapter 4.5]. The first, standard version is shown on Figure 5. We start from a cost function c on a digraph $D = (V, A)$ and a potential vector π with $c_{ij} - \pi_j + \pi_i \geq 0$ for every arc, and two designated subsets S and T . The set R is initialized as $R = S$, and denotes in every iteration the set of nodes that can be reached from S on a tight path, that is, all arcs of the path satisfying $c_{ij} - \pi_j + \pi_i = 0$. Every iteration increases the potential on $V \setminus R$ until some new tight arcs enter. We terminate once R contains a node in T ; a shortest path between S and T can be recovered using the pointers $pred(i)$.

In our algorithm, this subroutine will be applied if Assumption 1(i) holds. In the Δ -phase, we apply it for the digraph $E_f^F(\Delta)$ and the cost function $c_{ij} = C'_{ij}(f_{ij} + \Delta)$, and the potential π as in the algorithm. Note that if the initial π is rational, and all c_{ij} values are rational, the algorithm terminates with a π that is also rational. Assumption 1(i) guarantees that if f_{ij} and Δ are rational numbers, then so is c_{ij} .

Figure 6 shows a multiplicative version of Dijkstra’s algorithm. It is the same as the original one after substituting $c_{ij} = \log \gamma_{ij}$ and $\pi_i = \log \mu_i$. This variant shall be applied under Assumption 1(ii). We shall assume that every e^{π_i} value is rational, and set $\mu_i = e^{\pi_i}$, and $\gamma_{ij} = e^{C'_{ij}(f_{ij} + \Delta)}$. The assumption guarantees that if f_{ij} and Δ are rational numbers, then so is γ_{ij} . Consequently, the rationality of the e^{π_i} values is maintained during the computations.

Subroutine DIJKSTRA

INPUT A digraph $D = (V, A)$, disjoint subsets $S, T \subseteq V$, a cost function $c : A \rightarrow \mathbb{R}$ and a potential vector $\pi : V \rightarrow \mathbb{R}$ with $c_{ij} - \pi_j + \pi_i \geq 0$ for every $ij \in A$.

OUTPUT A shortest path P between a node in S and a node in T and a $\pi' : V \rightarrow \mathbb{R}$ with $c_{ij} - \pi'_j + \pi'_i \geq 0$ for every $ij \in A$, and equality on every arc of P .

$R \leftarrow S$;

for $i \in S$ **do** $pred(i) \leftarrow NULL$;

while $R \cap T = \emptyset$ **do**

$\alpha \leftarrow \min\{c_{ij} - \pi_j + \pi_i : ij \in A, i \in R, j \in V \setminus R\}$;

for $j \in V \setminus R$ **do** $\pi_j \leftarrow \pi_j + \alpha$;

$Z \leftarrow \{j \in V \setminus R : \exists ij \in A, i \in R \text{ such that } c_{ij} - \pi_j + \pi_i = 0\}$;

for $j \in Z$ **do**

$pred(j) \leftarrow i \in R \text{ such that } \exists ij \in A : c_{ij} - \pi_j + \pi_i = 0$;

$R \leftarrow R \cup Z$;

$\pi' \leftarrow \pi$;

Figure 5:

Subroutine MULTIPLICATIVE DIJKSTRA

INPUT A digraph $D = (V, A)$, disjoint subsets $S, T \subseteq V$, a cost function $\gamma : A \rightarrow \mathbb{R}$ and a potential vector $\mu : V \rightarrow \mathbb{R}$ with $\gamma_{ij} \frac{\mu_i}{\mu_j} \geq 1$ for every $ij \in A$.

OUTPUT A shortest path P between a node in S and a node in T and a $\mu' : V \rightarrow \mathbb{R}$ with $\gamma_{ij} \frac{\mu'_i}{\mu'_j} \geq 1$ for every $ij \in A$, and equality on every arc of P .

$R \leftarrow S$;

for $i \in S$ **do** $pred(i) \leftarrow NULL$;

while $R \cap T = \emptyset$ **do**

$\alpha \leftarrow \min\{\gamma_{ij} \frac{\mu_i}{\mu_j} : ij \in A, i \in R, j \in V \setminus R\}$;

for $j \in V \setminus R$ **do** $\mu_j \leftarrow \alpha \mu_j$;

$Z \leftarrow \{j \in V \setminus R : \exists ij \in A, i \in R \text{ such that } \gamma_{ij} \frac{\mu_i}{\mu_j} = 1\}$;

for $j \in Z$ **do**

$pred(j) \leftarrow i \in R \text{ such that } \exists ij \in A : \gamma_{ij} \frac{\mu_i}{\mu_j} = 1$;

$R \leftarrow R \cup Z$;

$\mu' \leftarrow \mu$;

Figure 6: