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

László A. Végh*
College of Computing
Georgia Institute of Technology
Atlanta, GA
veghal@cs.elte.hu

## 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 key 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 [15] and in [35], 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.

## Categories and Subject Descriptors

G.2.1 [**Combinatorics**]: [Combinatorial algorithms]; F.2 [**Analysis of Algorithms and Problem Complexity**]: [General]

## General Terms

Theory, Algorithms, Economics

## Keywords

network flow algorithms, convex optimization, strongly polynomial algorithms, market equilibrium

*Supported by NSF Grant CCF-0914732.



## 1. INTRODUCTION

The flow polyhedron is defined on a directed network $G=(V,E)$ by arc capacity and node demand constraints. For feasible solutions $f$, 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. Assume that each $C_{ij}$ can be described by some numerical parameters (for example, if $C_{ij}(\alpha) = c_{ij}\alpha^2 + d_{ij}\alpha$, then it is described by the values $c_{ij}, d_{ij}$). By a strongly polynomial algorithm for such a problem, we mean the following: (i) the algorithm uses only elementary arithmetic operations and comparisons; (ii) the number of these operations is bounded as a polynomial of the number of nodes ($n$) and arcs of the network ($m$); (iii) if the input consists of rational numbers, then the size of all numbers during the computations is polynomially bounded in $n$, $m$, and the size of the input numbers.

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 [25] in 1986, by Hochbaum and Shantikumar [17] in 1990, and by Karzanov and McCormick [21] 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 [33, 34] 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 [15] shows that it is impossible to find an $\varepsilon$-approximate solution 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 [30] result showing the impossibility of finding $\varepsilon$-approximate roots of polynomials in strongly polynomial time.

The remaining class of polynomial objectives with hope of strongly polynomial algorithms is convex quadratic. For these functions, the existence of a rational optimal solution is always guaranteed. Granot and Skorin-Kapov [12] extended Tardos's method [34] 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 [15, 4, 16, 12, 32]). The survey paper [16] gives an overview of special cases solvable in strongly polynomial time. This includes fixed number of suppliers (Cosares and Hochbaum, [4]), and series-parallel graphs (Tamir [32]). 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: [15] shows that no strongly polynomial algorithm may exist in general, but for quadratic cost functions, Hochbaum and Queyranne gave a strongly polynomial algorithm [14].

An entirely different motivation of our study comes from the field 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 [27, Chapter 5] or [36]. All these problems are described by rational convex programs. For the linear Fisher market problem, a strongly polynomial algorithm was given by Orlin [29].

To the extent of the author's knowledge, this field has been considered so far as an entirely new domain in combinatorial optimization. An explicit connection to classical flow problems was pointed out in the recent paper [37]. It turns out that the linear Fisher market, along 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 [37] 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 [31]. 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 [17] or [21] (with a final transformation of a close enough approximate solution to the 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. This formulation can be extended to spending constraint utilities, a market defined by Vazirani [35]. 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. Orlin [29] 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 [29] are known for other market settings. Using the general framework enables a simpler treatment as in [29].

## 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 [28], is also based on this framework. On the other hand, the algorithm of [6] has also been extended to the convex minimum-cost flow problem, first by Minoux [24] to convex quadratic flows, later to general convex objectives (Minoux [25], Hochbaum and Shantikumar [17]). Our algorithm is an enhanced version of these later algorithms, in the spirit of Orlin's technique [28]. 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 [29] is also based on the ideas of [28]. To get an overview on linear and nonlinear minimum-cost flow algorithms, we refer the reader to [1].

The algorithm of Edmonds and Karp consists of $\Delta$-phases, and the scaling factor $\Delta$ decreases at least by a factor of two for the next phase. The elementary step of the $\Delta$-phase transports $\Delta$ units of flow from a node with excess at least $\Delta$ to another with demand at least $\Delta$. This is done on a shortest path in the $\Delta$-residual network, the graph of residual arcs with capacity at least $\Delta$. An invariant property maintained in the $\Delta$-phase is that the $\Delta$-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 [28] 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 once an arc $ij$ has flow at least $4n\Delta$ in the $\Delta$-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, one can obtain the following simple algorithm. Let us contract $ij$, and restart the Edmonds-Karp algorithm in the smaller graph. Iterating this method, we can find an optimal dual solution, which easily enables to compute an optimal primal solution. Instead of explicitly contracting and restarting, [28] (see also [1, Chapter 10.7]) continues the scaling method after finding an abundant arc, and maintains the forest $F$ of such arcs. A new arc enters $F$ in $O(\log n)$ phases, and 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. The Edmonds-Karp algorithm naturally extends here (see [24, 25, 17]). In the $\Delta$-phase, the invariant is that the $\Delta$-residual graph contains no negative cycle for the linearization of the cost function into $\Delta$-chunks. That is, the cost of the arc $ij$ is defined as $(C_{ij}(f_{ij} + \Delta) - C_{ij}(f_{ij}))/\Delta$. When transporting $\Delta$-units of flow on a shortest path with respect to this cost function, the 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 $\Delta/2$. As $\Delta$ approaches 0, the cost of $ij$ converges to the derivative $C'_{ij}(f_{ij})$. By the KKT conditions, a flow $f$ is optimal if and only if the residual graph contains no negative cycles for the derivatives. A variant of this algorithm is outlined in Section 3.

## 1.2 Our results

The class of problems where we give a strongly polynomial algorithm is defined by four assumptions. We need to have an oracle access to the derivatives of the $C_{ij}$'s by Assumption 1. The technical Assumption 2 restricts the problem instances where each cost functions is either strictly convex or linear.

Two main ingredients of the algorithm are black box subroutines provided by Assumptions 3 and 4. The first subroutine returns a solution that satisfies optimally the KKT-conditions on a subset of edges. In particular, it returns an optimal solution if the input is the support of an optimal solution. For quadratic cost functions and for the Fisher markets, this can be implemented by solving simple systems of equations. The second is a technical subroutine measuring the quality of a candidate solution, returning its distance from optimality in a certain metric. This subroutine appears to be the bottleneck in complexity: for quadratic costs, it needs a minimum cost-to-time ratio cycle computation (also known as the tramp steamer problem). For the Fisher markets, an all pairs shortest paths computation is performed. The main result of the paper is the following.

THEOREM 1.1. *Let Assumptions 1-4 hold for the problem of minimizing $\sum_{ij\in E} C_{ij}(f_{ij})$ over feasible flows $f$ 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.*

We shall now outline the basic ideas of our strongly polynomial algorithm. The problem is first transformed to an instance with no upper bounds on the arcs. All cost functions $C_{ij}$ are assumed to be differentiable. From a complexity perspective, cost functions are provided via oracles (Assumption 1), as in [17] and [21].

For simplicity of presentation, let us now assume that there exists a unique optimal solution $f^*$. We shall assume that $f^*$ can be computed exactly (in particular, it is rational). Let $F^*$ denote the set of arcs $ij$ with $f^*_{ij} > 0$. By the KKT conditions, $\pi_j - \pi_i = C'_{ij}(f^*_{ij})$ for each $ij \in F^*$ and for the Lagrangian multipliers $\pi$. Assumption 3 guarantees that if we can somehow guess the set $F^*$, then we can obtain the optimal $f^*$ by solving the above equality system.

Our starting point is the nonlinear scaling scheme as described above, with the only difference that the linearized cost function is replaced by $C'_{ij}(f_{ij}+\Delta)$. This has similar properties but is easier to handle. As in [28], we can identify an abundant arc $ij$ in $O(\log n)$ steps, which must be contained in $F^*$. However, contraction does not work. The reason is that the KKT-conditions give $\pi_j - \pi_i = C'_{ij}(f^*_{ij})$, a condition containing both primal and dual (more precisely, Lagrangian) variables simultaneously.

In each 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. We change the terminology from abundant as they will not necessarily carry a huge amount of flow. Indeed, for revealed arcs, we delete the lower capacity and even allow the flow value to become negative. Accordingly, we keep both $ij$ and $ji$ for $ij \in F$ in the set of residual arcs.

The second important difference is that in contrast to Orlin's algorithm, we cannot assume that $F$ is acyclic: 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 $F$ must also be extended with arcs inside a component.

At the end of each phase when $F$ is extended, we compute the discrepancy of $F$, the maximum absolute value of the sum of node demands in a component of $F$. 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. We pretend that $F = F^*$ and try to compute an optimal solution $\hat{f}$ under this hypothesis, which can be done by a subroutine guaranteed by Assumption 3. If $F \subsetneq F^*$, then $\hat{f}$ is not optimal. Another subroutine, described in Assumption 4, enables us to compute the smallest $\hat{\Delta}$ value for which $\hat{f}$ is $\hat{\Delta}$-feasible (that is, it satisfies the necessary conditions for a $\hat{\Delta}$-phase). If $\hat{\Delta} < \Delta/2$, then we set $\hat{\Delta}$ as our next scaling value and $f = \hat{f}$ as the next pseudoflow. Otherwise, the usual 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.

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 [25]. 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. 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 $\ell, u : E \to \mathbb{R}\cup\{\infty\}$ on the arcs, and node demands $b : V \to \mathbb{R}$ with $\sum_{i\in V} b_i = 0$. On each arc $ij \in E$, $C_{ij} : (-\infty,\infty) \to \mathbb{R}\cup\{\infty\}$ is a convex function. Either it is differentiable everywhere, or it is $\infty$ on $(-\infty,\ell_{ij})$ and is differentiable on $(\ell_{ij},\infty)$. The minimum-cost flow problem with separable convex objective is defined as follows.

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

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 path $ikj$ of length 3, by introducing a new node $k$. Let us set $b_k = u_{ij} - \ell_{ij}$, $C_{ik}(\alpha) = C_{ij}(\alpha + \ell_{ij})$, $C_{kj} \equiv 0$. Furthermore, let us increase $b_i$ by $\ell_{ij}$ 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.

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

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

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 [17] and [21], we do not require the functions $C_{ij}$ to be given explicitly, but assume oracle access only.

Assumption 1. *For each arc $ij$, we are given a differential oracle that returns the value $C'_{ij}(\alpha)$ in $O(1)$ time for any $\alpha \in \mathbb{R}$.*

Note that we do not assume an evaluation oracle returning $C_{ij}(\alpha)$. The main algorithm needs only access to a differential oracle; however, the subroutines described in Assumptions 4 and 3 may need stronger oracles. For quadratic objectives, the derivatives are linear functions, however, for the market applications, we will have logarithmic derivatives. As we shall argue in Section 6.2, in those settings, access is provided to the exact values $e^{C'_{ij}(\alpha)}$, which is also sufficient for all applications of the oracle, most importantly, shortest path computations. Assumptions 2, 3, and 4 shall be given in Section 4.

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 $\infty$ 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 $\pi : V \to \mathbb{R}$ such that

$$\pi_j - \pi_i \leq C'_{ij}(f_{ij}) \quad \forall ij \in E_f. \tag{1}$$

For a value $\Delta > 0$, let $E_f(\Delta)$ denote the subset of arcs in $E_f$ that have residual capacity at least $\Delta$ (in particular, it contains $E$). We say that the pseudoflow $f$ is $\Delta$-feasible, if $E_f(\Delta)$ contains no negative cycles with respect to the cost function $C'_{ij}(f_{ij} + \Delta)$. This is equivalent to the existence of a potential $\pi : V \to \mathbb{R}$ so that

$$\pi_j - \pi_i \leq C'_{ij}(f_{ij} + \Delta) \quad \forall ij \in E_f(\Delta). \tag{2}$$

If $ji$ is a reverse arc, then this condition gives $C'_{ij}(f_{ij} - \Delta) \leq \pi_j - \pi_i$. This is different (and weaker) than the condition in [25] and [17], where $(C_{ij}(f_{ij} + \Delta) - C_{ij}(f_{ij}))/\Delta$ is used in the place of $C'_{ij}(f_{ij} + \Delta)$.

The following lemma shows that a $2\Delta$-feasible pseudoflow can be transformed to a $\Delta$-feasible pseudoflow by small changes on the arcs.

Lemma 2.1. *If the pseudoflow $\bar{f}$ is $2\Delta$-feasible, then there exists a $\Delta$-feasible pseudoflow $f$ with $\bar{f}_{ij} - f_{ij} \in \{0, \pm\Delta\}$ on each $ij \in E$. Consequently, $Ex(f) \leq Ex(\bar{f}) + m\Delta$.*

Proof. Consider a potential $\pi$ satisfying (2) with $\bar{f}$ and $2\Delta$. We want to prove that modifying $\bar{f}$ on each arc by $\pm\Delta$ or 0 will give a solution $f$ satisfying (2) with $\Delta$ and the same $\pi$.

Let $ij \in E$ be a forward arc. If the condition is violated on $ij$, then $C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi_i$. However, by $2\Delta$-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 (2) for both $ij$ and $ji$, using that $C'_{ij}(f_{ij} - \Delta) \leq C'_{ij}(f_{ij}) = C'_{ij}(\bar{f}_{ij} + \Delta) < \pi_j - \pi_i$.

In the sequel, assume that (2) holds for $ij \in E$. If $ji \notin E_{\bar{f}}(\Delta)$ or (2) holds for $ji$, then we set $f_{ij} = \bar{f}_{ij}$. Assume now $ji \in E_{\bar{f}}(\Delta)$ but (2) is violated for $ji$, meaning $\pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta)$. Let us set $f_{ij} = \bar{f}_{ij} - \Delta$. $ij$ satisfies (2) by $\pi_j - \pi_i < C'_{ij}(\bar{f}_{ij} - \Delta) \leq C'_{ij}(\bar{f}_{ij}) = C'_{ij}(f_{ij} + \Delta)$.

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

The subroutine Adjust($\Delta$) performs the simple steps of the proof.

Finally, we may assume without loss of generality that $G = (V, E)$ is strongly connected. 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 $n \leq m$.

# 3. THE BASIC ALGORITHM

Let us now outline the (weakly) polynomial algorithm by Minoux [25], a simple extension of the Edmonds-Karp algorithm.

We start with the $\Delta_0$-feasible solution $f^0 \equiv \mathbf{0}$ with $Ex(f^0) \leq (2n+m)\Delta_0$. The algorithm consists of $\Delta$-phases, starting with $\Delta = \Delta_0$, with $\Delta$ decreasing by exactly a factor of two between two phases.

Each $\Delta$-phase consists of a preprocessing part and a main part. In the first, $\Delta_0$-phase, no preprocessing is needed. In each later phase, we start with a $2\Delta$-feasible pseudoflow $f'$. We perform Adjust($\Delta$) to obtain a $\Delta$-feasible pseudoflow $f$.

In the main part of phase $\Delta$, 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 $\Delta$ 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 $C'_{ij}(f_{ij} + \Delta)$. If $S(\Delta) = \emptyset$ or $T(\Delta) = \emptyset$, we proceed to the next phase with scaling factor $\Delta/2$.

## 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.

LEMMA 3.1. *(i) In the $\Delta$-phase, the pseudoflow is an integer multiple of $\Delta$ on each arc, and consequently, $E_f^F(\Delta) = E_f^F$.*

*(ii) $\Delta$-feasibility is maintained when augmenting on a shortest path.*

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

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

LEMMA 3.2. *Let $f$ be the pseudoflow at the end of the $\Delta$-phase and $f'$ in an arbitrary later phase. Then $||f-f'||_\infty \le (2n+m+1)\Delta$. If $f_{ij} > (2n+m+1)\Delta$ at the end of the $\Delta$-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

We investigate a slightly restricted class of functions $C_{ij}$.

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 reverse arcs as well.

Let $F^*$ denote the set of arcs that are tight in every optimal solution. Formally, $ij \in F^*$ if for all pairs $(f,\pi)$ satisfying (1), $\pi_j - \pi_i = C'_{ij}(f_{ij})$ holds. 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 $x, y : E \to \mathbb{R}$ 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 $x_{ji} < y_{ji}$.

PROOF OF LEMMA 4.1. Let $f^*$ be a different optimal solution, and consider potentials $\pi$ and $\pi^*$ with both $(f,\pi)$ and $(f^*,\pi^*)$ satisfying (1). 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 $\rho_{f^*} \equiv \rho_f$, $E_{f,f^*}$ must contain a cycle $C$ containing $ij$. For every arc $ab \in C$, $f_{ab} > f^*_{ab}$ and thus $ab \in E_{f^*}$ and $ba \in E_f$. By (1),

$$0 = \sum_{ab \in C} \pi^*_b - \pi^*_a \quad \le \sum_{ab \in C} C'_{ab}(f^*_{ab}) \text{ and}$$
$$0 = \sum_{ab \in C} \pi_a - \pi_b \quad \le \sum_{ab \in C} C'_{ba}(f_{ba}) = -\sum_{ab \in C} C'_{ab}(f_{ab}).$$

Since each function $C_{ab}$ is convex, $f_{ab} > f^*_{ab}$ gives $C'_{ab}(f_{ab}) \ge C'_{ab}(f^*_{ab})$. 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, finding the set $F^*$ enables us to compute an optimal solution. In the basic algorithm, $F = \{ij \in E : f_{ij} > (2n+m+1)\Delta\}$ is always a subset of $F^*$ by 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, $F$ is extended in some number of steps polynomially bounded in $n$ and $m$. We shall modify the 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, a vector $f : E \to \mathbb{R}$ is an *$F$-pseudoflow*, if $f_{ij} \ge 0$ for $ij \in E - F$ (but it is allowed to be negative on $F$). For such an $f$, let us define $E_f^F = E_f \cup \overleftarrow{F}$. If $ij \in F$, then the residual capacity of $ji$ is $\infty$. We shall maintain an $F$-pseudoflow when $F \subseteq F^*$ is the set of revealed arcs in some phase of the algorithm.

We say that $f : E \to \mathbb{R}$ is *$F$-optimal*, if it is an $F$-pseudoflow with $Ex_b(f) = 0$ and there exists $\pi : V \to \mathbb{R}$ with

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

The definition of $F^*$ implies that any optimal $f^*$ is also $F$-optimal if $F \subseteq F^*$. We shall prove that given an $F$-optimal solution, we can easily find an optimal solution as well.

LEMMA 4.2. *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 maximum flow computation. Further, $ij \in F^*$ whenever $f_{ij} > 0$.*

PROOF. Assume $(f,\pi)$ and $(\bar f, \bar\pi)$ both satisfy (3). We prove that *(i)* $f_{ij} = \bar f_{ij}$ whenever $ij$ is a nonlinear arc; and *(ii)* if $ij$ is a linear arc with $f_{ij} \ne \bar f_{ij}$, then $\pi_j - \pi_i = C'_{ij}(f_{ij}) = C'_{ij}(\bar f_{ij}) = \bar\pi_j - \bar\pi_i$.

*(ii)* immediately implies the second half of the claim as it can be applied for an arbitrary optimal $f^*$. 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 $f_{ij} > \bar f_{ij}$, $ij$ must be contained on a cycle $C \subseteq E_{f,\bar f}$. For every arc $ab \in C$, $ab \in E^F_{\bar f}$ and $ba \in E^F_f$ follows (using $\overleftrightarrow{F} \subseteq E^F_{\bar f} \cap E^F_f$). By (3),

$$0 = \sum_{ab \in C} \bar\pi_b - \bar\pi_a \quad \le \sum_{ab \in C} C'_{ab}(\bar f_{ab}) \text{ and}$$
$$0 = \sum_{ab \in C} \pi_a - \pi_b \quad \le \sum_{ab \in C} C'_{ba}(f_{ba}) = -\sum_{ab \in C} C_{ab}(f_{ab}).$$

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, $ij$. This immediately proofs *(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 $f^*_{ij} = f_{ij}$. On the set of linear arcs satisfying $\pi_j - \pi_i = C'_{ij}(f_{ij})$, we can solve a feasible circulation problem with node demands being the same as the $\rho_f(i)$ values, lower capacities 0 and upper capacities $\infty$. The feasible solutions are precisely the optimal solutions. Indeed, if $f^*$ is an optimal solution, then (3) and *(ii)* imply $\pi_j - \pi_i = C'_{ij}(f_{ij})$ for all $ij$ with $f^*_{ij} > 0$. $\square$

Let $F$ be the set of revealed arcs in the $\Delta$-phase. We maintain an $F$-pseudoflow $f$, but instead of $\Delta$-feasibility, we require the following slightly stronger property. Let $E_f^F(\Delta)$

denote the set of arcs in $E_f^F$ with residual capacity at least $\Delta$. We say that the $F$-pseudoflow $F$ is $(\Delta, F)$-*feasible*, if $E_f^F(\Delta)$ contains no negative cycle with respect to the cost function $C'_{ij}(f_{ij}+\Delta)$. Equivalently, there exists a potential $\pi : V \to \mathbb{R}$ so that

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

We shall also maintain that the set of revealed arcs, $F$ is *linear acylic*, meaning that $F$ does not contain any cycle of linear arcs.

## 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 $x : E \to \mathbb{R}$ is $F$-*tight*, if $x_{ij} = 0$ whenever $ij \notin F$ and there exists a potential $\pi : V \to \mathbb{R}$ with

$$\pi_j - \pi_i = C'_{ij}(x_{ij}) \quad \forall ij \in F. \tag{5}$$

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 (3) might be violated for edges in $E - F$ and also since $Ex_b(f) > 0$ is allowed. On the other hand, an $F$-optimal set is not necessarily $F$-tight as it can be nonzero on $E - F$.

Given $F$ and some node demands $\hat{b} : V \to \mathbb{R}$, we would like to find an $F$-tight $x$ with $Ex_{\hat{b}}(x) = 0$. 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 $D_{\hat{b}}(F) = 0$.

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

Consider now an $F$-tight $f$, and let

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

Because of continuity, this infimum is actually a minimum whenever the set is nonempty. If $f$ is not $(\Delta, F)$-feasible for any $\Delta$, 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 $err_F(f) = 0$. As $f$ is assumed to be $F$-tight, $E_f^F(\Delta) = E \cup \overleftarrow{F}$, the same arc set in (4) for any value of $\Delta$.

ASSUMPTION 4. *Assume a subroutine* ERROR$(f, F)$ *is provided, that returns $err_F(f)$ for any $F$-tight pseudoflow $f$ in 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.

## 4.3 Description of the enhanced algorithm

The algorithm starts with the $(\Delta_0, \emptyset)$-feasible solution $\mathbf{0}$ with $Ex(\mathbf{0}) \le (2n + m_N)\Delta_0$. The appropriate value can be chosen as $\Delta_0 = \max\{err_\emptyset(\mathbf{0}), Ex_b(\mathbf{0})/(2n + m_N)\}$. By the second part of Assumption 4, $err_\emptyset(\mathbf{0})$ must be finite or the problem is unbounded. We initialize the revealed arc set $F = \emptyset$.

The algorithm consists of $\Delta$-phases, starting with $\Delta = \Delta_0$. In the $\Delta$-phase, we shall maintain a linear acyclic revealed arc set $F \subseteq F^*$, and a $(\Delta, F)$-feasible $F$-pseudoflow $f$. Besides preprocessing and main part, the phases have a third part, TRIAL-AND-ERROR, where the next value of $\Delta$ is also determined.

No preprocessing is done in the $\Delta_0$-phase. In a later phase, either we have a $\Delta$-feasible pseudoflow from the previous phase, or the scaling factor in the previous phase was $2\Delta$, and thus we have a $2\Delta$-feasible $f$. In the latter case, ADJUST$(\Delta)$ provides a $\Delta$-feasible pseudoflow.

The main part of the $\Delta$-phase is the same as in the basic algorithm. Let $S(\Delta) = \{i \in V : \rho_f(i) - b_i \ge \Delta\}$ and $T(\Delta) = \{i \in V : \rho_f(i) - b_i \le -\Delta\}$. As long as $S(\Delta) \ne \emptyset$, $T(\Delta) \ne \emptyset$, send $\Delta$ 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 $C'_{ij}(f_{ij}+\Delta)$.

After the main part is finished, EXTEND$(F, f, \Delta)$ adds some arcs $ij \in E - F$ with $f_{ij} > (2n+m+1)\Delta$ to $F$. We add all nonlinear such $ij$'s to $F$, and keep adding linear arcs as long as the linear acyclic property is maintained. Consider a linear arc $ij$, which is not admitted to $F$ because there exists a path $P \subseteq \overleftrightarrow{F}$ between $i$ and $j$. Then we reroute the entire amount $f_{ij}$ of flow from $ij$ to $P$.

If no new arc enters $F$, then we 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 still has a high discrepancy: $D_b(F) > \Delta$.

Otherwise, the subroutine TRIAL-AND-ERROR$(F, f, \Delta)$ determines the next $f$ and $\Delta$. 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 4.2.

---

**Algorithm** ENHANCED CONVEX FLOW
$f \leftarrow \mathbf{0}$; $\Delta \leftarrow \max\{err_\emptyset(f), Ex_b(f)/(2n + m_N)\}$;
$F \leftarrow \emptyset$;
**repeat** //$\Delta$-*phase*
  **if** $f$ is not $\Delta$-feasible **then** ADJUST$(\Delta)$;
  **do** //*main part*
    $S(\Delta) \leftarrow \{i \in V : \rho_f(i) - b_i \ge \Delta\}$;
    $T(\Delta) \leftarrow \{i \in V : \rho_f(i) - b_i \le -\Delta\}$;
    $P \leftarrow$ shortest $s - t$ path in $E_f^F$
      with $s \in S(\Delta)$, $t \in T(\Delta)$;
    send $\Delta$ units on $P$ from $s$ to $t$;
  **while** $S(\Delta), T(\Delta) \ne \emptyset$;
  EXTEND$(F, f, \Delta)$;
  **if** $F$ was extended *and* $(D_b(F) \le \Delta)$
      **then** TRIAL-AND-ERROR$(F, f, \Delta)$
      **else** $\Delta \leftarrow \Delta/2$;

---

### *The Trial-and-Error subroutine*

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

**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. By Lemma 4.2, an optimal solution to (P) can be found by a single maximum flow computation. In this case, the algorithm terminates. If $err_F(\hat{f}) \ge \Delta/2$, then keep the original $f$, 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

TRIAL-AND-ERROR replaces $f$ by $\hat{f}$ if $err_F(\hat{f}) \le \Delta/2$ 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 $\Delta$-phase, the $F$-pseudoflow $f$ is an integer multiple of $\Delta$ on each arc $ij \in E - F$, and consequently, $E_f^F(\Delta) = E_f^F$.*

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

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

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

PROOF. For *(i)*, $f$ is zero everywhere in $E - F$ at the beginning of the algorithm and after every successful trial. In every other case, the previous phase had scaling factor $2\Delta$, and thus by induction, the flow is an integer multiple of $2\Delta$ at the beginning of the $\Delta$-phase. This is maintained in the preprocessing, as ADJUST may only modify by $\Delta$. The shortest path augmentations also change the flow by $\Delta$. This implies $E_f^F(\Delta) = E_f^F$.

For *(ii)*, $P$ is a shortest path if there exists potentials $\pi$ verifying (4) with $\pi_j - \pi_i = C'_{ij}(f_{ij} + \Delta)$ on each arc $ij \in P$. We show that when augmenting on the shortest path $P$, (4) is maintained with the same $\pi$. If neither of $ij, ji$ is in $P$, then it is trivial as the flow is not changed on $ij$. If $ij \in P$, then the new flow value will be $f_{ij} + \Delta$, hence we need $\pi_j - \pi_i \le C'_{ij}(f_{ij} + 2\Delta)$, obvious as $C'_{ij}$ is monotonely increasing. Finally, if $ji \in P$, then the new flow is $f_{ij} - \Delta$, and thus we need $\pi_j - \pi_i \le C'_{ij}(f_{ij})$. 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)*, $Ex(f) \le n\Delta$ as the main part terminates with either $S(\Delta) = \emptyset$ or $T(\Delta) = \emptyset$. If the original flow $f$ is kept and the next scaling factor is $\Delta/2$, then $Ex(f) \le 2n(\Delta/2)$ at the beginning of the next phase. ADJUST$(\Delta/2)$ increases the excess by at most $\Delta/2$ on each nonlinear arc, and it does not change values on linear arcs. If a successful trial replaced $\Delta$ by $\Delta_{next}$, then by definition, the new excess is at most $(2n + m_N)\Delta_{next}$, and ADJUST$(\Delta_{next})$ does not change anything as the flow is already $\Delta_{next}$-feasible. *(iii)* immediately implies *(iv)*, as each flow augmentation decreases $Ex(f)$ by $\Delta$. □

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 $f_{ij} > (2n + m + 1)\Delta$ for a $(\Delta, F)$-feasible $f$. We shall prove that $f_{ij} > 0$ for some $F$-optimal solution, and thus Lemma 4.2 gives $ij \in F^*$.

After the phase when $ij$ entered, let us continue with running the basic algorithm in all later phases: we do not extend $F$ and do not perform TRIAL-AND-ERROR in any of the later phases, and always choose the next scaling factor as $\Delta/2$. In a $\Delta'$-phase, the flow is modified by at most $\Delta'$ on $ij$ during preprocessing and $(2n+m_N)\Delta'$ during the main part by Lemma 5.1(v). Consequently, in all phases after $\Delta$, the total modification is bounded by $(2n+m+1)(\Delta/2+\Delta/4+...) \le (2n + m + 1)$.

If we leave the algorithm running forever, it converges to the $F$-optimal solution $f^*$. By the above observation, $f^*_{ij} > 0$. □

The next lemma is of key importance.

LEMMA 5.3. *In* TRIAL-AND-ERROR$(F, f, \Delta)$, $err_F(\hat{f}) \le 2(2n + m + 4)m\Delta$.

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

THEOREM 5.4. *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 $\Delta$-phase, a new arc is revealed within $2\lceil\log_2 T\rceil$ phases, for $T = 8(2n+m+4)m$. As $\Delta$ decreases by at least a factor of two between two phases, after $\lceil\log_2 T\rceil$ steps we have $\Delta_T \le \Delta/T$. Assume that in the $\Delta_T$ phase, we still have the same revealed arc set $F$ as in the $\Delta$-phase.

Assume first $D_b(F) > \Delta$. At the end of the main part of the $\Delta_T$-phase, $D_b(F) > (2n + m + 2)m\Delta_T$. Thus there is a connected component $K$ of $F$ with $|\sum_{i\in K} b_i| > (2n + m + 2)m\Delta_T$. We have

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

There must be an arc $ij$ entering or leaving $K$ with $f_{ij} > 2n + m + 1$, a contradiction as at least one such arc must be added to $F$ in EXTEND$(F, f, \Delta_T)$.

Assume next $D_b(F) \le \Delta$. 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 $\Delta_0$ as the output of TRIAL-AND-ERROR$(\mathbf{0})$.

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 $\Delta_T$-phase. (Even if the subroutine is not actually performed, its analysis is valid provided that $D_b(F) \le \Delta_T$.)

Let $\hat{f}$ be the arc set found by TRIAL$(F, \hat{b})$. This is the same in the $\Delta$ and the $\Delta_T$-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 $\Delta$-phase, $\Delta/2 < err_F(\hat{f})$. Using Lemma 5.3 for the $\Delta_T$-phase,

$$err_F(\hat{f}) \le 2(2n+m+4)m\Delta_T \le \Delta/4 < err_F(\hat{f})/2,$$

a contradiction. On the other hand, if we had a successful trial in the $\Delta$-phase, then $\Delta_T \le 2\Delta_{next}/T$. Also, $Ex_b(\hat{f}) \le nD_b(F) \le n\Delta_T$. Thus

$$\Delta_{next} = \max\{err_F(\hat{f}), Ex_b(\hat{f})/(2n+m_N)\} \le$$
$$\le 2(2n+m+4)m\Delta_T \le \Delta_{next}/2,$$

a contradiction again. $\Box$

Some preparation is needed to prove Lemma 5.3.

LEMMA 5.5. *For a linear acylic arc set $F \subseteq E$, let $x$ and $y$ be two $F$-tight vectors. Then $||x-y||_\infty \le ||\rho_x - \rho_y||_1$ 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 $x_{ab} > y_{ab}$ for every $ab \in C$, this is only possible if all arcs of $C$ are linear, contradicting the linear acyclic property of $F$. (Note that $E_{x,y} \subseteq \overleftrightarrow{F}$).

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

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

$$\sum_{p > t \ge q} g_{v_p v_q} = \sum_{p \le t} \rho_g(v_p).$$

As $g$ is positive on all arcs, this implies $g_{v_a v_b} \le \sum_{p \le t} \rho_g(v_p) \le ||\rho_g||_1$ for all such arcs, in particular, for $ij$. $\Box$

CLAIM 5.6. *If $f$ and $\hat{f}$ are $F$-pseudoflows with $\hat{f}_{ij} = 0$ for $ij \in E - F$, and $f$ is $(\Delta, F)$-feasible, then $\hat{f}$ is $\Delta + ||f - \hat{f}||_\infty$ feasible.*

PROOF. There is a potential $\pi$ so that $f$ and $\pi$ satisfy (4), that is, $\pi_j - \pi_i \le C'_{ij}(f_{ij} + \Delta)$ if $ij \in E_f^F(\Delta)$. For $\alpha = ||f - \hat{f}||_\infty$, we have $f_{ij} + \Delta \le \hat{f}_{ij} + \Delta + \alpha$. Consequently, (4) is satisfied for $\hat{f}_{ij}$, $\pi$ 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. $\Box$

PROOF OF LEMMA 5.3. $f$ is $(\Delta, F)$-feasible with some potential $\pi$. We claim that there is an $F$-tight $\bar{f}$ so that $|\bar{f}_{ij} - f_{ij}| \le \Delta$ for every $ij \in F$, and $Ex_b(f) \le (2n+m+2)m\Delta$.

Indeed, $(\Delta, F)$-feasibility gives

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

As $C'_{ij}$ is continuous, there must be a value $f_{ij} - \Delta \le \beta \le f_{ij} + \Delta$ with $C'_{ij}(\beta) = \pi_j - \pi_i$. 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 $ij \in E - F$. Using that $f_{ij} \le (2n+m+1)\Delta$ if $ij \notin F$ and $Ex_b(f) \le n\Delta$, we obtain an $F$-tight $\bar{f}$ with

$$Ex_b(\bar{f}) \le n\Delta + |F|\Delta + (2n+m+1)(m - |F|)\Delta$$
$$\le (2n+m+2)m\Delta.$$

On the other hand, $Ex_b(\hat{f}) \le nD_b(F) \le n\Delta$. Consequently,

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

Applying Lemma 5.5 for $x = \bar{f}$ and $y = \hat{f}$ gives $||\bar{f} - \hat{f}||_\infty \le 2(2n+m+3)m\Delta$. Now $\bar{f}$ is $2(2n+m+4)m\Delta$-feasible by Claim 5.6. $\Box$

THEOREM 5.7. *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 5.4, 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. $\Box$

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

REMARK 5.8. *A shortest path computation can be performed in time $\rho_S(n,m) = O(m + n\log n)$, 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 [28] algorithm, 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 $c_{ij} \ge 0$. This clearly satisfies Assumption 1 since $C'_{ij}(\alpha) = 2c_{ij}\alpha + d_{ij}$. Also, Assumption 2 is satisfied.

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

$$\pi_j - \pi_i = 2c_{ij}x_{ij} + d_{ij} \qquad \forall ij \in F \tag{6}$$
$$\sum_{j:ji \in F} x_{ji} - \sum_{j:ij \in F} x_{ij} = b_i \qquad \forall i \in V$$

To verify Assumption 3, we show that this system is solvable if $F$ is linear acyclic and $D_b(F) = 0$. Clearly, we can solve the system separately on different connected components of $F$. In the sequel, let us focus on a connected component $K$.

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 $\pi_j$ value arbitrarily, it determines all other $\pi_i$ 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 $b_i$. Both tasks can be performed in linear time.

Assume next both linear and nonlinear arcs are present, and let $T$ be a 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 $x_{ij} = \alpha$ 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 edges span no nonlinear edges.

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 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 $F$ on component $K$: $A_{i,ij} = -1$, $A_{i,ji} = 1$ for all $ij \in F$, and all other entries are 0. Let $C$ be the $|F| \times |F|$ diagonal matrix with $C_{ij,ij} = -2c_{ij}$. (6) 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 $|K| \times |K|$ Laplacian matrix with $L_{ii} = \sum_{j: ij \in \overleftrightarrow{F}} \frac{1}{2c_{ij}}$, $L_{ij} = L_{ji} = -\frac{1}{2c_{ij}}$ if $ij \in F$ and $L_{ij} = 0$ otherwise, and $b'$ is an appropriate vector with $\sum_{i \in K} b'_i = 0$.

The main task is to solve the system $L\pi = b'$. It is well-know that in the component $K$ of $F$, $L$ has rank $|K| - 1$ and the system is always solvable whenever $\sum_{i \in K} b'_i = 0$. A solution can be found in $O(n^{2.37})$ time [3]. All previously described operations can be done in $O(m)$ time, hence we obtain $\rho_E(n, m) = O(n^{2.37})$.

To implement $\textsc{Error}(f, F)$, we have to find the minimum $\Delta$-value such that there exists a $\pi$ potential with

$$\pi_j - \pi_i \le (2c_{ij}x_{ij} + d_{ij}) + 2c_{ij}\Delta \quad \forall ij \in E \cup \overleftarrow{F}. \tag{7}$$

We show that this can be reduced to the minimum-cost-to-time ratio cycle problem (see [1, Chapter 5.7]). In a directed graph, there is a cost function $p_{ij}$ and a time $\tau_{ij} \ge 0$ 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 [22, 23] 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

$$\begin{aligned} &\min \mu \text{ s. t. there are no negative cycles} \\ &\qquad \text{for the cost function } p_{ij} - \mu\tau_{ij}. \end{aligned} \tag{8}$$

Our problem fits into this framework with $p_{ij} = 2c_{ij}x_{ij} + d_{ij}$ and $\tau_{ij} = 2c_{ij}$. In (8), the optimal $\mu$ value is $-\Delta$. However, [22] 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 $\mu$ to solve (8) 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 (8) 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 $E \cup \overleftarrow{F}$. If there exists one, then (7) is infeasible, thus $\Delta = \infty$, and (P) is unbounded as we can send arbitrary flow around this cycle. Otherwise, we have $\sum_{ij \in C} \tau_{ij} > 0$ for any cycle with $\sum_{ij \in C} p_{ij} < 0$, and consequently, there exists a finite $\Delta$ satisfying (7).

Consequently, $\rho_T(n, m) = \min\{O(n^3 \log^2 n), O(n \log n(n^2 + m \log\log n))\}$. Theorem 5.7 gives the following running time bound.

Theorem 6.1. *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 5.8, 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 as well analyzing Megiddo's algorithm.

## 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 $j \in G$, $U_{ij} \ge 0$ 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 incident to each $i$ and to each $j$.

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 relaxation of this problem was given by Eisenberg and Gale [7] in 1959. Recently, Shmyrev [31] gave the following alternative relaxation. The variable $f_{ij}$ represents the money payed by buyer $i$ for product $j$.

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

Let us construct a network on node set $B \cup G \cup \{t\}$ as follows. Add an arc $ij$ for every $i \in B$, $j \in G$ with $U_{ij} > 0$, 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 $\infty$. With $p_j$ representing the flow on arc $jt$, the above formulation is a minimum-cost flow problem with separable convex objective. 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 (1) with $\pi : B \cup G \cup \{t\} \to \mathbb{R}$. We may assume $\pi_t = 0$. $C'_{jt}(\alpha) = \log \alpha$ implies $\pi_j = -\log p_j$. On each $ij \in E$ we have $\pi_j - \pi_i \le -\log U_{ij}$ with equality if $f_{ij} > 0$. With $\beta_i = e^{\pi_i}$, this is equivalent to $U_{ij}/p_j \le \beta_i$, verifying that every buyer buys a best bundle of goods.

Assumption 2 is clearly satisfied, however, Assumption 1 needs oracle access to the derivatives. On the arcs $ij$ between buyers and goods, this is $-\log U_{ij}$, while on an arc $jt$ it is $\log f_{jt}$, hence we cannot determine the exact values.

The derivative oracle is used in two parts of the algorithm: for computing shortest paths between $S(\Delta)$ and $T(\Delta)$ in the main part, and for the subroutine $\textsc{Adjust}(\Delta)$ in the preprocessing part. We can solve both these problems provided the rational values $e^{C'_{ij}(\alpha)}$. For shortest path computations, we use a multiplicative version of Dijkstra's algorithm, computing products instead of sums on the arcs. For $\textsc{Adjust}(\Delta)$, we need a potential $\pi$ verifying that $f$ is $(\Delta, F)$-feasible. Instead of the $\pi_j$ values we can use the $e^{\pi_j}$ values, that can be chosen as rational numbers.

Let us turn to Assumption 3. When the subroutine $\textsc{Trial}$ 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 $\textsc{Trial}(F)$, we want to find a potential $\pi : B \cup G \cup \{t\} \to \mathbb{R} \cup \{\infty\}$, 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

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

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 $\pi$ 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 the 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 $\alpha$ yields another feasible solution.

Let $T_1, \dots, T_k$ be the components of the large component after deleting $t$. If $T_\ell$ contains a single good $j$, then we shall set $p_j = P_j = 0$ ($\pi_j = 0$). If $T_\ell$ is nonsingular, then $F$ restricted to $T_\ell$ 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 $\textsc{Error}(f, F)$ can be implemented based on the Floyd-Warshall algorithm. Let $\pi$ be the potential witnessing that $f$ is $(\Delta, F)$-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 $U_{ij}/P_j \le R_i$ if $i \in B$, $j \in G$, $ij \in E$, with equality if $ji \in E_f^F$. Furthermore, we have $p_j - \Delta \le P_j \le p_j + \Delta$ if $p_j > 0$ and $P_j \le \Delta$ if $p_j = 0$.

Let us now define $\beta : G \times G \to \mathbb{R}$ 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 $j \in G$.

Clearly, $P_{j'} \ge P_j \beta_{jj'}$ must hold to guarantee that edges $ij$ with $ji \in E_f^F$ have the best bang-per-buck values. Conversely, if the prices $P_j$ satisfy this property, then we can set the $R_i$ values as best bang-per-buck ratios, thus satisfying the necessary inequalities on all edges.

If there is a directed cycle $C$ with $\Pi_{ab \in C} \beta_{ab} > 1$, then $f$ cannot be $(\Delta, F)$-feasible for any $\Delta$. Otherwise, we may compute $\tilde{\beta}_{jj'}$ as the maximum of $\Pi_{ab \in P} \beta_{ab}$ over all directed paths $P$ 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 $(\Delta, F)$-feasibility, we clearly need to satisfy

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

Let us define $\Delta$ as the smallest value satisfying all these inequalities, that is,

$$\Delta = \max \left\{ 0, \max_{j, j' \in G} \frac{p_j \tilde{\beta}_{jj'} - p_{j'}}{\tilde{\beta}_{jj'} + 1} \right\}. \tag{9}$$

We claim that $f$ is $(\Delta, F)$-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 $P_{j'} \ge P_j \beta_{jj'}$, and $p_j - \Delta \le P_j \le p_j + \Delta$.

The complexity of $\textsc{Error}(f, F)$ 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 5.7 gives the following.

THEOREM 6.2. *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 by Orlin [29] runs in $O(n^4 \log n)$ time, assuming $m = O(n^2)$. Under this assumption, we get the same running time bound.

## 6.3 Fisher's market with spending constraint utilities

The *spending constraint utility* extension of linear Fisher markets was defined by Vazirani [35]. 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 $U^1_{ij} > U^2_{ij} > \ldots > U^{\ell_{ij}}_{ij} > 0$ of utilities with numbers $L^1_{ij}, \ldots, L^{\ell_j}_{ij} > 0$. Buyer $i$ accrues utility $U^1_{ij}$ for every unit of $j$ he purchased by spending the first $L^1_{ij}$ dollars on good $j$, $U^2_{ij}$ for spending the next $L^2_{ij}$ dollars, etc. These $\ell_{ij}$ 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 [35]. This was settled by Devanur et al. [2], showing that Shmyrev's formulation naturally extends here. Let $f_{ij}^k$ represent the money payed by buyer $i$ for the $k$'th segment of product $j$, $1 \le k \le \ell_{ij}$.

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

This gives a convex cost flow problem again on the node set $B \cup G \cup \{t\}$, by adding $\ell_{ij}$ 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 Trial($F$), we want to find an $F$-tight flow $f'$ on the extended network, witnessed by the potential $\pi : B \cup S \cup G \cup \{t\} \to \mathbb{R}$. 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 $ij$, $U_{ij}^k / S_{ij}^k \le R_i$ and $S_{ij}^k \ge P_j$. If $i(ij,k) \in F$ then $U_{ij}^k / P_j \ge R_i$ and if $j(ij,k) \in F$ then $U_{ij}^k / P_j \le R_i$ .

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 $T_\ell$ be a component of $K - t$. $F$ is a spanning tree of linear edges in $T_\ell$ 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 computed. The running time of Trial is again linear, $\rho_T(n', m') = O(m') = O(m)$.

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

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

Given these $\beta_{jj'}$ values, the $\tilde{\beta}_{jj'}$ values can be computed by the Floyd-Warshall algorithm and the optimal $\Delta$ obtained by (9) 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 5.7, together with Remark 5.8, we obtain:

Theorem 6.3. *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 $O(mn^3 + m^2(m + n \log n) \log m)$.*

# 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 Orlin [29].

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' [34] for linear programs. Such an extension could be possible by extending our techniques to the setting of Hochbaum and Shantikumar [17].

The recent paper [37] 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 by Goel and Vazirani [11], and the Arrow-Debreu Nash-bargaining problem by Vazirani [36], 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 [37] 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 [20] also fall into this class, and are solvable in strongly polynomial time; see also Nagano [26]. Other strongly polynomially solvable special cases are given by Hochbaum and Hong [13].

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 [18] and by Iwata, McCormick and Shigeno [19]. One might try to develop strongly polynomial algorithms for some class of separable convex objectives; in particular, for separable convex quadratic functions.

## 8. REFERENCES


[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications.* Prentice-Hall, Inc., 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*, pages 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*, pages 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] D. Hochbaum and S. Hong. About strongly polynomial time algorithms for quadratic optimization over submodular constraints. *Mathematical programming*, 69(1):269–309, 1995.

[14] D. Hochbaum and M. Queyranne. Minimizing a convex cost closure set. *SIAM Journal on Discrete Mathematics*, 16:192, 2003.

[15] 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.

[16] D. S. Hochbaum. Complexity and algorithms for nonlinear optimization problems. *Annals of Operations Research*, 153(1):257–296, 2007.

[17] 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.

[18] S. Iwata. A capacity scaling algorithm for convex cost submodular flows. *Mathematical Programming*, 76(2):299–308, 1997.

[19] S. Iwata, S. Mccormick, and M. Shigeno. Fast cycle canceling algorithms for minimum cost submodular flow. *Combinatorica*, 23(3):503–525, 2003.

[20] K. Jain and V. V. Vazirani. Eisenberg-Gale markets: Algorithms and game-theoretic properties. *Games and Economic Behavior*, 70(1):84–106, 2010.

[21] 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.

[22] N. Megiddo. Combinatorial optimization with rational objective functions. *Mathematics of Operations Research*, 4(4):414–424, 1979.

[23] N. Megiddo. Applying parallel computation algorithms in the design of serial algorithms. *Journal of the ACM (JACM)*, 30(4):852–865, 1983.

[24] M. Minoux. A polynomial algorithm for minimum quadratic cost flow problems. *European Journal of Operational Research*, 18(3):377–387, 1984.

[25] M. Minoux. Solving integer minimum cost flows with separable convex cost objective polynomially. *Mathematical Programming Study*, 25:237, 1985.

[26] K. Nagano. On convex minimization over base polytopes. *Integer Programming and Combinatorial Optimization*, pages 252–266, 2007.

[27] N. Nisan, T. Roughgarden, E. Tardos, and V. Vazirani. *Algorithmic Game Theory.* Cambridge University Press New York, NY, USA, 2007.

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

[29] 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.

[30] J. Renegar. On the worst-case arithmetic complexity of approximating zeros of polynomials. *Journal of Complexity*, 3(2):90–113, 1987.

[31] 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.

[32] A. Tamir. A strongly polynomial algorithm for minimum convex separable quadratic cost flow problems on series-parallel networks. *Mathematical Programming*, 59:117–132, 1993.

[33] É. Tardos. A strongly polynomial minimum cost

circulation algorithm. *Combinatorica*, 5(3):247–255, 1985.

[34] É. Tardos. A strongly polynomial algorithm to solve combinatorial linear programs. *Operations Research*, 34(2):250–256, 1986.

[35] V. V. Vazirani. Spending constraint utilities with applications to the adwords market. *Mathematics of Operations Research*, 35(2):458–478, 2010.

[36] 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.

[37] L. A. Végh. Concave generalized flows with applications to market equilibria. *Arxiv preprint arXiv:1109.3893*, 2011.