Laboratoire ARTEMIS
BP 53x - 38041 - Grenoble cedex



Photo Focale Studio Grenoble

RAPPORT DE RECHERCHE

Polynomial Methods for Separable Convex Optimization in Unimodular Linear Spaces with Applications to Circulations and Co-circulations in Networks

Alexander V. Karzanov **S. Thomas McCormick**

RR 927 -M- **Janvier 1994**

Centre national de la recherche scientifique
Ecole normale supérieure de Lyon

Institut IMAG

Institut national polytechnique de Grenoble
Université Joseph Fourier Grenoble I

Polynomial Methods for Separable Convex Optimization in Unimodular Linear Spaces with Applications to Circulations and Co-circulations in Networks *

Alexander V. Karzanov
Institute for System Analysis
9 Prospect 60 Let Oktyabrya
117312 Moscow Russia

S. Thomas McCormick
Faculty of Commerce and Business Administration
University of British Columbia
Vancouver, BC V6T 1Z2 Canada

December 1993

Abstract

We consider the problem of minimizing a separable convex objective function over the linear space given by system $Mx = 0$ with a totally unimodular matrix M. In particular, this generalizes the usual minimum linear cost circulation and co-circulation problems in a network, and the problem of determining the Euclidean distance to certain polyhedra (e.g. the perfect bipartite matching polytope and the feasible flows polyhedron).

We first show that the idea of minimum mean cycle canceling originally worked out for linear cost circulations by Goldberg and Tarjan [9] and extended to some other problems [3, 6, 14] can be generalized to give a combinatorial method with geometric convergence for our problem. We also generalize the computationally more efficient Cancel and Tighten method.

We then consider specialized objective functions, such as piecewise linear, pure and piecewise quadratic, and piecewise mixed linear and quadratic, and show how both methods can be implemented to find exact solutions in polynomial time (strongly polynomial in the piecewise linear case). These implementations are then further specialized for finding circulations and co-circulations in a network.

We finish by showing how to extend our methods to find optimal integer solutions, to linear spaces of larger fractionality, and to the case when the objective functions are given by approximate oracles.

*The research of the first author was supported by "Chaire municipale", Mairie de Grenoble; the second author was partially supported by an NSERC Operating Grant, an NSERC Grant for Research Abroad, and a UBC Killam Faculty Study Leave Fellowship. This research was done while both authors were visiting Laboratoire ARTEMIS IMAG at Université Joseph Fourier de Grenoble, France.

1 Introduction

A *circulation* in a digraph $G = (V, E)$ is a vector $x \in \mathbf{R}^E$ satisfying the *conservation* condition:

$$\sum_{k:(k,i)\in E} x_{ki} - \sum_{j:(i,j)\in E} x_{ij} = 0 \qquad \text{for all } i \in V.$$

A vector $\pi \in \mathbf{R}^E$ of *node potentials* induces a *co-circulation* (potential difference) $\Delta\pi \in \mathbf{R}^E$ in G defined by

$$\Delta\pi(i,j) = \pi_j - \pi_i \qquad \text{for } (i,j) \in E.$$

Suppose that for each $e \in E$ we are given a *weight* function $w_e : \mathbf{R} \to \mathbf{R}$ satisfying the following condition:

(C) w_e is convex.

Consider the problem of minimizing the separable *total weight* function defined as

$$w(x) = \sum_{e\in E} w_e(x_e) \tag{1}$$

over the set C of circulations in G, and consider a similar problem for the set $C^\perp$ of co-circulations in G. In this framework one can describe many special problems on circulations and co-circulations. The most popular is the classical *minimum linear cost circulation problem* (briefly, the *linear circulation problem*): given vectors $l, u \in \mathbf{R}^E$ (of lower and upper capacities of arcs) with $l \le u$, and $c \in \mathbf{R}^E$ (of costs of arcs), find $x \in C$ that minimizes $\sum_{e\in E} c_e x_e$ provided that x is within the bounds $l_e \le x_e \le u_e$, $e \in E$. This is reduced to the above "unconstrained" problem by putting, for $e \in E$,

$$w_e(r) = \begin{cases} c_e u_e + \alpha(r - u_e) & \text{if } r > u_e, \\ c_e r & \text{if } l_e \le r \le u_e, \\ c_e l_e + \alpha(l_e - r) & \text{if } r < l_e; \end{cases} \tag{2}$$

where α is an arbitrarily large positive number. (We could also just set $w_e(x_e) = +\infty$ when x_e is outside the bounds.) Thus we obtain a special case of separable piecewise linear objective function w. A similar reduction applies for the *minimum linear cost co-circulation problem* (briefly, *linear co-circulation problem*).

It is tempting to further generalize this model by allowing demands at nodes, or by putting convex penalties on violations of conservation, or by putting convex costs on the node potentials, etc. However, all of these generalizations can be handled by adjoining a new node 0 to the network, along with arcs from 0 to every other node. These new arcs can then carry demands, conservation violations, or node potentials. Thus we do not really lose any generality by considering only the simpler model.

Among the many approaches to solving the linear circulation problem (see, e.g., Ahuja, Magnanti and Orlin [2] for a survey), one strongly polynomial method is distinguished by its simplicity, namely, the minimum mean cycle canceling method due to Goldberg and Tarjan [9]. Its ideas inspired a number of polynomial and strongly polynomial algorithms for other problems, such as the linear co-circulation problem (Ervolina and McCormick [3]), and determining the Euclidean distances to certain polyhedra [6, 14]. We briefly outline some of these results below.

The first purpose of the present paper is to show that the "minimum mean canceling method" can be unified and generalized to solve a wide class of problems. We generalize it in two directions at once: to deal with separable convex objectives as in (1) and (C), and to arbitrary unimodular linear spaces L (i.e., the solution set of system $Mx = 0$ with a totally unimodular matrix M). The

sets of circulations and co-circulations of a digraph are special cases of such an L. The method we develop is combinatorial and it has geometrical dual convergence in terms of the ℓ_1-distance of the current vector of partial (left and right) derivatives to an optimal dual vector.

There is a faster version of the minimum mean cycle canceling method for the linear circulation problem, the so-called "cancel and tighten method" [9, 3]. We show that this also can be generalized to our generic case to give a method with geometrical dual convergence at the same rate but much more efficient than our first method.

The generic Minimum Mean Canceling and Cancel and Tighten methods are described in Sections 2–3 and Section 4, respectively. In Sections 5–7 we analyze how these methods work for special cases of objective function w and of unimodular space L. First we explain that if the functions w_e are "fast growing" (e.g., quadratic functions), geometric dual convergence ensures geometric primal convergence, and estimates the rate of this convergence. We also distinguish three important special cases of w: (i) piecewise linear, (ii) piecewise quadratic, and (iii) piecewise mixed linear and quadratic. We show that in case (i) the number of iterations of the methods to get an optimal solution is strongly polynomial (for the linear circulation and co-circulation problems the implementations of the methods are, in fact, equivalent to algorithms in [9, 3]). For cases (ii) and (iii) we establish that a polynomial number of iterations is sufficient to find an approximate solution that can be rounded to an exact solution on quadratic pieces (using standard continued fractions techniques) in polynomial time; then case (iii) needs to solve a remaining piecewise linear problem.

It should be noted that to implement an iteration in our generic methods we require that the functions w_e be given by a simple oracle (defined in Section 2) and that certain linear programs with unimodular matrices be solved efficiently (we discuss this matter in Section 8). Section 9 extends the methods to linear spaces of larger fractionality, showing that the methods remain efficient if the fractionality is "bounded". Section 10 shows that the methods can solve problems where the variables are restricted to integer values just as easily as continuous problems, and Section 11 considers the case when the w_e's are given by approximate oracles. Finally, in Section 12 we discuss how practical computations can take advantage of the symmetric duality present in our problem: since we can freely choose to solve either a primal or a dual problem, in some cases we may be better off solving the dual of the "natural" form of a problem.

For a survey of relevant earlier results on circulations and co-circulations we refer the reader to [2, Chapter 14]; many of the applications below are taken from that chapter. Polynomial algorithms for the circulation problem with non-linear costs have been designed by Florian [4] and Minoux [27, 28]. These algorithms are based on cost scaling techniques. In particular, [28] gives a polynomial algorithm for finding an optimal integer-valued circulation in a network with quadratic costs; this algorithm was speeded-up and extended to general convex costs in [2, Chapter 14]. Our methods do not require any scaling at all, and work with the problem in a very natural and combinatorial way. Also, we shall see in Section 10 that our algorithm is able to solve the same problem faster in most cases.

Hochbaum and Shanthikumar [13] give a polynomial method for separable convex minimization over a linear space which is also based on cost scaling. However, their method is rather expensive as it involves solving linear programs where each original variable gets replicated $4n$ times (n is the number of original variables). By contrast, our methods work in the space of original variables at all times, and appear to be quite practical. Some computational testing has been done by McCormick and Liu [24] for an algorithm which is effectively a specialization of our methods, implemented for computing classic linear co-circulations.

There are many applications for convex separable cost circulations. As we shall see below, many of these have objective functions with rather simple forms. However our method should be useful also for more general non-linear problems. An example of this comes from applications to queueing networks where modelers are interested in the arc congestion function $w_e = a_e x_e/(u_e - x_e)$ on the interval $[0, u_e)$ (see, e.g., Magnanti [20]).

Applications for co-circulations are rarer, but they do arise. The best known example is the dual to the minimum cost network flow problem, which has a piecewise linear convex objective (see below). Chapter 7F of [35] contains several other more general applications of co-circulations. A more non-linear application is to traffic flow equilibrium in congested cities, where the co-circulation represents differences in timings of traffic lights, see Hassin [12]. This application is not convex, but perhaps our methods could be used as a heuristic for quickly finding local optima. Our co-circulation methods should be useful also in PERT networks, where there are often non-linear penalties for the elapsed time (time difference) for the task represented by an arc.

An example of a problem that is not unimodular is the well-known fractional b-matching problem. Such problems have the node-edge incidence matrix of an undirected graph as the essential part of their coefficient matrix. This problem has fractionality two, which is certainly bounded, so the extension of our methods in Section 9 will give strongly polynomial algorithms for this problem.

A more significant application arises when we add a new set of linear constraints $d^i x = 0$, $i = 1, \ldots, p$, to the unimodular system $Mx = 0$, where all entries d^i_j are integers and the sum of their absolute values is at most a fixed number k. For example, in a network we could add the equalities $x_e = x_{e'}$ for $k/2$ pairs of arcs $\{e, e'\}$ in the network. Then 2^k (which is constant because k is fixed) is an upper bound on any subdeterminant of the resulting matrix, and so also on the fractionality of the problem. Thus our methods are again polynomial.

1.1 The Linear and Piecewise Linear Cases

We first dispose of the purely linear objective. Recall that in our framework, including bounds would make the objective piecewise linear. Thus "purely linear" here means a circulation problem with no lower or upper bounds. Such a problem is trivial: we can check if there are any non-zero cycles. If such exist, the problem is unbounded; if not, any circulation is optimal.

In the piecewise linear case the circulation problem includes the above-mentioned linear circulation problem, whose arc objective functions look like Figure 1. Note that its dual is an instance of the piecewise linear co-circulation problem. Indeed, the dual objective function can be written as

$$\max_{\pi} \{ l^T (\bar{c}^\pi)^+ - u^T (\bar{c}^\pi)^- \} \tag{3}$$

(see Ford and Fulkerson [5] or Hassin [11]), where $\bar{c}^\pi_e = c_e - \Delta\pi_e$ is the vector of reduced costs, and $(\cdot)^+$, $(\cdot)^-$ are the positive and negative parts. Hence, we can re-write (3) in terms of the co-circulation $\Delta\pi$ as minimizing the sum over arcs of the functions w_e defined by

$$w_e(\Delta\pi_e) = \begin{cases} u_e(\Delta\pi_e - c_e) & \text{if } \Delta\pi_e \geq c_e \\ l_e(\Delta\pi_e - c_e) & \text{if } \Delta\pi_e \leq c_e. \end{cases} \tag{4}$$

Equation (4) shows that the classic dual circulation (co-circulation) objective is the piecewise linear function shown in Figure 2 (note that since we assume w.l.o.g. that $l \leq u$, this function is indeed convex). Another piecewise linear applications is in telephone network area switching (Monma and Segal [29]).

1.2 The Pure and Piecewise Quadratic Case

Applications with pure quadratic arc functions arise in papers such as Schneider and Zenios [37], where one is trying to minimize the ℓ_2 norm of the difference between a known previous circulation and an unknown circulation w.r.t. perturbed data, and Ahlfeld, et al [1], where one is minimizing the (quadratic) power loss in a resistive network. One can imagine piecewise quadratic applications as well.

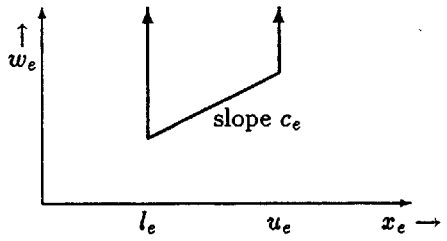


Figure 1: The arc objective function for the classic linear circulation problem with lower bound l_e, upper bound u_e, and cost c_e.

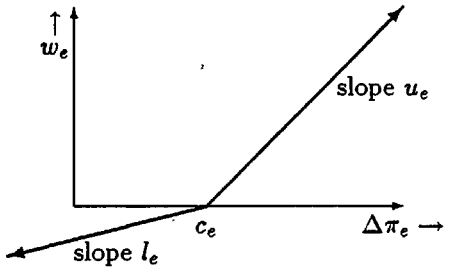


Figure 2: The arc objective function for the classic linear co-circulation problem with lower bound l_e, upper bound u_e, and cost c_e.

1.3 The Piecewise Mixed Linear and Quadratic Case

The piecewise mixed linear and quadratic case arises in applications like minimizing the Euclidean distance to certain polyhedra. Frank and Karzanov [6] showed how to modify the minimum mean cycle canceling method in [9] to get a polynomial algorithm for finding the Euclidean distance from the origin to the perfect bipartite matching polytope. Ibaraki, Karzanov and Nagamochi [14] extended this algorithm to determine the Euclidean distance to the circulation polyhedron of a network $G = (V, E)$ with lower and upper arc capacities l and u (possibly $l_e = -\infty$ and/or $u_e = \infty$). In our terms, this problem has $w_e(r)$ defined as r^2 for $l_e \leq r \leq u_e$, and defined by (2) otherwise.

Another interesting application is to linear costs with quadratic penalties for violating bounds, such as is considered in McCormick [22], who was interested in finding a "least infeasible" circulation in an infeasible network that minimized the ℓ_2-norm of the arc bound violations.

The problems in [6, 22] were our original motivation for developing the methods in this paper.

2 The Minimum Mean Canceling Method in Unimodular Linear Spaces

The circulation and co-circulation problems that we considered in Section 1 can be solved by algorithms that are specializations of a general method that can handle a wide class of problems that includes circulations and co-circulations. We now define this class and then describe the method. Later, Section 7 will show how the method can be implemented for circulations and co-circulations in networks.

In what follows we utilize some well-known facts in convex analysis, linear algebra, optimization, and matroid theory; for references, see, e.g., Rockafellar [34], Schrijver [38], or Welsh [40].

Let L be a subspace of the Euclidean space $\mathbf{R}^E$ with coordinates indexed by the elements of a set E. We study the unconstrained (within L) optimization problem:

(5) Find $x \in L$ such that $w(x) = \sum_{e \in E} w_e(x_e)$ is as small as possible.

where each w_e satisfies condition (C). We may assume that L contains a non-zero point, else the problem is trivial. We also assume that (5) has a *finite* optimal solution x. [This assumption can be checked by pre-processing the problem and computing an $x \in L$ that satisfies any bounds implicit in the w_e's, and a dual vector in the orthogonal complement of L that satisfies any dual bounds (see [35] for an explanation of what this means); this pre-processing involves at worst solving two linear programs, and ensures that a finite optimal solution to (5) exists. Note that we are solving the "continuous" or "real" verson of the problem, not the version where x is also required to be integer; however, we shall show how our methods extend to integer problems in Section 10.]

The convexity of w_e ensures the existence of the *right derivative*, denoted by $c_e^+(r)$, and the *left derivative*, $c_e^-(r)$, at every $r \in \mathbf{R}$. Note that $c_e^-(r) < c_e^+(r)$ is possible (the left and right derivatives of a convex function coincide except for a finite or countable set, but in any reasonable application, for each w_e there will be only a finite number of *breakpoints* r where $c_e^-(r) < c_e^+(r)$). We refer to $c_e^+(r)$ and $c_e^-(r)$ as the *left cost* and *right cost* of e at r, respectively. Because of the convexity of w_e,

(6) $$\text{if } r < r' \text{ then } c_e^-(r) \leq c_e^+(r) \leq c_e^-(r') \leq c_e^+(r');$$

in particular, both $c_e^-(r)$ and $c_e^+(r)$ are monotone non-decreasing in r. Recall that any function w_e satisfying (C) can be given via a monotone non-decreasing function c_e (of cost) on $\mathbf{R}$. [Such a c_e has a Riemann integral, and, up to adding a constant, c_e defines $w_e(r)$ to be $\int_0^r c_e(r)\mathrm{d}r$ for $r \geq 0$ and $-\int_r^0 c_e(r)\mathrm{d}r$ for $r < 0$. Furthermore, $c_e^+(r)$ is $\lim_{r' > r, r' \to r} c_e(r')$ and $c_e^-(r)$ is $\lim_{r' < r, r' \to r} c_e(r')$.] The convexity of w_e implies that

(7) for any $r \in \mathbf{R}$, c_e^+ is continuous at r in the set $[r, \infty)$ and c_e^- is continuous at r in $(-\infty, r]$.

It will be enough for us to assume that w_e is given by an *oracle* which:

(O) (i) when given a "point" $r \in \mathbf{R}$, will return the costs $c_e^+(r)$ and $c_e^-(r)$;

(ii) when given a "scalar" $q \in \mathbf{R}$, will return a point r with $c_e^-(r) \leq q \leq c_e^+(r)$.

[The only time when there is more than one possible r satisfying (O) (ii) is when q is the slope of a linear piece of w_e; in this case we assume that the oracle returns a (finite) endpoint of the linear piece. Note that the oracle assumption is justified for almost all functions one encounters in practice. In the case of piecewise linear or quadratic functions, the oracle reduces to a simple linear or binary search, followed by solving a one-variable linear equation. A less trivial example is the application to queueing networks [20] we referred to in the Introduction. It uses $w_e(x_e) = a_e x_e/(u_e - x_e)$ as its arc cost function. Certainly part (i) of (O) is easy to do, and for part (ii) an oracle just needs to compute $r = u_e - \sqrt{a_e u_e / q}$.]

We now impose the important restriction on L that it is the solution set of the linear system $Mx = 0$ with a totally unimodular $n \times m$ matrix M (whose columns correspond to the elements of E, so $|E| = m$). We assume w.l.o.g. that M has full row rank. Then the orthogonal complement $L^\perp$ of L can be also represented as the solution set of a system $Kx = 0$ where K is a full row rank $(m - n) \times m$ totally unimodular matrix. For convenience we will often call the vectors in L ($L^\perp$) *circulations* (*co-circulations*).

A vector $\xi \in L - \{0\}$ ($\xi \in L^\perp - \{0\}$) such that its support $\mathrm{supp}(\xi) = \{e \in E : \xi_e \neq 0\}$ is inclusion minimal is said to be a *cycle* (resp. *co-cycle*) of L (we borrow this terminology from regular

matroid theory). Clearly, if ξ is a cycle (co-cycle) then $-\xi$ is a cycle (resp. co-cycle) as well. A linear system represented by a totally unimodular matrix is distinguished by the fact that all its cycles are multiples of $(0, \pm 1)$ vectors. Thus we will speak of cycles and co-cycles as being $(0, \pm 1)$ vectors.

Example. Let $\mathcal{D} = (V, E)$ be a digraph with node-arc incidence matrix M. Then M is totally unimodular and $L = \{x \in \mathbf{R}^E : Mx = 0\}$ is the set of circulations for $\mathcal{D}$. The set of cycles for this L is exactly the set of simple circuits (with possible forward and backward arcs) in $\mathcal{D}$, and the set of co-cycles is exactly the set of simple cuts (with possible forward and backward arcs) in $\mathcal{D}$. The dual space $L^\perp$ is the space of co-circulations. Conversely, if M is instead a matrix whose rows are the incidence vectors of a circuit basis, then M is again totally unimodular and $L = \{x \in \mathbf{R}^E : Mx = 0\}$ is the set of co-circulations for $\mathcal{D}$. The set of cycles for this L is exactly the set of simple cuts in $\mathcal{D}$, and the set of co-cycles is exactly the set of simple cycles in $\mathcal{D}$. The dual space $L^\perp$ is the space of circulations.

We recall that any point (any integral point) $x \in L$ is a nonnegative (resp. nonnegative integer) combination $\alpha_1 \xi^1 + \ldots + \alpha_k \xi^k$ of $k \leq |E|$ cycles $\xi^1, \ldots, \xi^k \in L$; moreover, these ξ^i's can be chosen so that

$$\xi^i_e \xi^j_e \geq 0 \quad \text{for } i, j = 1, \ldots, k \text{ and } e \in E. \tag{8}$$

This is sometimes called a *conformal decomposition* of x, since (8) says that the signs of each ξ^i must conform to the signs of x.

In what follows we associate a $(0, \pm 1)$-vector ξ in $\mathbf{R}^E$ with the pair $F_\xi = (A_\xi, B_\xi)$, where $A_\xi = \{e : \xi_e = 1\}$ and $B_\xi = \{e : \xi_e = -1\}$. If ξ is a cycle (co-cycle) of L, we say that F_ξ is a *cycle* (resp. *co-cycle*) in E corresponding to ξ; the set of such F_ξ's is denoted by $\mathcal{F}$ (resp. $\mathcal{F}^\perp$). Clearly, $F = (A, B) \in \mathcal{F}$ implies $\overline{F} = (B, A) \in \mathcal{F}$; the cycle $\overline{F}$ is called *opposite* to F. Denote $|A| + |B|$ by $|F|$ (the *size* of F). Key parameters for us will be ϕ ($\phi^\perp$), the maximum size of a cycle (resp. co-cycle), and $\psi - 1$ ($\psi^\perp - 1$), the rank of M (resp. K); clearly $\phi \leq \psi = n + 1$ and $\phi^\perp \leq \psi^\perp = m - n + 1$.

Later on for a function $f : S \to \mathbf{R}$ and a subset $S' \subseteq S$, $f(S')$ denotes $\sum(f_e : e \in S')$. Also when it is not confusing, for a fixed point $x \in L$ and a cycle $F = (A, B)$ we denote $\sum_{e \in A} c^+_e(x_e)$ by $c^+(A)$ and denote $\sum_{e \in B} c^-_e(x_e)$ by $c^-(B)$.

Suppose that we have a cycle $F = (A, B)$ and a circulation $x \in L$, and we want to "augment x around F", i.e., increase x by an amount $\varepsilon > 0$ on A and decrease x by ε on B. Then the local change (the change per unit of ε) in objective function value is $c^+(A) - c^-(B)$, the *cost of F at x*, denoted $c(x, F)$. We say that F is a *negative cycle* at x if $c(x, F) < 0$. The *mean cost* of F at x is $\overline{c}(x, F) := c(x, F)/|F|$. We say that F is a *minimum mean cycle* if $\overline{c}(x, F)$ is as small as possible. Define

$$\lambda(x) = \min\{0, -\min_{F \in \mathcal{F}} \overline{c}(x, F)\}. \tag{9}$$

If $\lambda(x)$ is positive, it is called the *absolute minimum mean value* for x.

The following lemma gives an optimality criterion for (5).

Lemma 2.1 *Point $x \in L$ is an optimal solution to (5) if and only if there is no negative cycle at x.*

Since $\lambda(x) = 0$ if and only if there are no negative cycles, we can alternatively state that x is optimal if and only if $\lambda(x) = 0$. This lemma can be derived from general optimality theorems in convex programming. However, to make our description self-contained, we show how to prove it directly. Let $x \in L$. Suppose that $\overline{c}(x, F) = -\lambda < 0$ for some $F = (A, B) \in \mathcal{F}$. Then "push a small amount $\varepsilon > 0$ around F" to get $x' \in L$ with $w(x') < w(x)$, thus proving that x is not optimal. More precisely, we put

$$x'_e = \begin{cases} x_e + \varepsilon & \text{for } e \in A, \\ x_e - \varepsilon & \text{for } e \in B, \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

(i.e., $x' = x + \varepsilon\xi$ for the cycle $\xi \in L$ corresponding to F), where $\varepsilon > 0$ is chosen so that $\delta_e(\varepsilon) := c_e^+(x_e + \varepsilon) - c_e^+(x_e) < \lambda$ for each $e \in A$ and $\delta_e(\varepsilon) := c_e^-(x_e) - c_e^-(x_e - \varepsilon) < \lambda$ for each $e \in B$. Such an ε exists because of (7). Moreover, since c^+ and c^- are monotone, $w_e(x'_e) - w_e(x_e)$ is at most $\varepsilon c_e^+(x_e) + \varepsilon\delta_e(\varepsilon)$ for $e \in A$ and at most $-\varepsilon c_e^-(x_e) + \varepsilon\delta_e(\varepsilon)$ for $e \in B$, whence

$$w(x') - w(x) \le \varepsilon c(x, F) + \sum_{e \in A \cup B} \varepsilon\delta_e(\varepsilon) < \varepsilon c(x, F) + \varepsilon\lambda|F| = 0.$$

This proves the "only if" part of Lemma 2.1. The "if" part will follow from the next lemma. A key concept in this and many following results is *reduced costs*: given a circulation x and a co-circulation h, the positive reduced cost of element e is $c_e^+(x_e) - h_e$, and its negative reduced cost is $h_e - c_e^-(x_e)$. This lemma characterizes $-\lambda(x)$ as being the largest possible lower bound on reduced costs for x fixed as h varies over all vectors in $L^\perp$.

Lemma 2.2 *The number λ is an upper bound on $\lambda(x)$ if and only if there exists $h \in L^\perp$ such that the reduced costs satisfy*

$$c_e^+(x_e) - h_e \ge -\lambda \quad \text{and} \quad h_e - c_e^-(x_e) \ge -\lambda \quad \text{for all } e \in E. \tag{11}$$

Moreover, if $\lambda = \lambda(x) > 0$ and $F = (A, B)$ is a minimum mean cycle, then $c_e^+(x_e) - h_e = -\lambda$ for all $e \in A$ and $h_e - c_e^-(x_e) = -\lambda$ for all $e \in B$.

This can be proved via linear programming duality. A direct proof is as follows.

Proof: Define $\widehat{c}_e^+(x_e) = c_e^+(x_e) + \lambda$ and $\widehat{c}_e^-(x_e) = c_e^-(x_e) - \lambda$. Then for any $F = (A, B) \in \mathcal{F}$,

(12) the mean cost of F w.r.t. $(\widehat{c}^+, \widehat{c}^-)$ is λ larger than the mean cost of F w.r.t. (c^+, c^-).

Suppose that for some λ there is an h satisfying (11). Since $h \in L^\perp$, $\widehat{c}^+(A) - \widehat{c}^-(B) = (\widehat{c}^+ - h)(A) - (\widehat{c}^- - h)(B) \ge 0$ (by (11)). Thus the min mean cycle value for $(\widehat{c}^+, \widehat{c}^-)$ is nonnegative, and by (12) $\lambda(x) \le \lambda$.

Now suppose that $\lambda \ge \lambda(x)$. Then (12) implies that there are no negative cycles w.r.t. $(\widehat{c}^+, \widehat{c}^-)$. Furthermore, $\widehat{c}_e^+ - \widehat{c}_e^- = c_e^+ - c_e^- + 2\lambda \ge 2\lambda \ge 0$ for each $e \in E$ (as $c_e^+ - c_e^- \ge 0$ by (6)). Choose functions h and f on E such that: (i) $\widehat{c}_e^+ \ge h_e \ge f_e \ge \widehat{c}_e^-$; (ii) there are no negative cycles w.r.t. (h, f); and (iii) the sum of differences $\Delta_e := h_e - f_e$ over $e \in E$ is minimum subject to (i) and (ii).

Suppose that $\Delta_u > 0$ for some $u \in E$. Since we cannot decrease h_u, there is a cycle $F = (A, B)$ with $h(A) - f(B) = 0$ and $u \in A$. Similarly, there is a cycle $F' = (A', B')$ with $h(A') - f(B') = 0$ and $u \in B'$. Let ξ, ξ' be the vectors in L corresponding to F, F', respectively. Represent $\xi + \xi'$ as the sum of cycles $\xi^1, \ldots, \xi^k$ satisfying (8), and let $F_i = (A_i, B_i)$ stand for F_{ξ^i}. Then $\{A_1, \ldots, A_k\}$ is a partition of $(A \cup A') - (B \cup B')$ and $\{B_1, \ldots, B_k\}$ is a partition of $(B \cup B') - (A \cup A')$, whence

$$\sum_i (h(A_i) - f(B_i)) = (h(A) - f(B)) + (h(A') - f(B')) - \sum_{e \in (A \cap B') \cup (A' \cap B)} \Delta_e \le 0 + 0 - \Delta_u < 0.$$

So some F_i is negative for (h, f); a contradiction.

Thus, $\Delta_e = 0$ for all $e \in E$, i.e., $f = h$. Since $(h(A) - h(B)) + (h(B) - h(A)) = 0$ for any pair $F = (A, B)$ and $\overline{F} = (B, A)$ of opposite cycles and there are no negative cycles for (h, h), we have $h(A) - h(B) = 0$ for every $F = (A, B) \in \mathcal{F}$. Since L is generated by cycles, the latter means that $hx = 0$ for all $x \in L$, i.e., $h \in L^\perp$. Now (11) follows from $c_e^+ + \lambda = \widehat{c}_e^+ \ge h_e$ and $c_e^- - \lambda = \widehat{c}_e^- \le f_e = h_e$.

For the second part, if F is a minimum mean cycle with, say, $c_e^+ - h_e > -\lambda(x)$ for some element in A, then since (11) is true for all elements of F, we would have $\overline{c}(x, F) > -\lambda$, contradicting that $\lambda = \lambda(x)$. □

From Lemma 2.2 we now derive the "if" part of Lemma 2.1. Suppose that $x^* \in L$ admits no negative cycles, i.e., $\lambda(x^*) = 0$. Then (11) yields

$$c_e^+(x_e^*) \geq h_e \geq c_e^-(x_e^*) \quad \text{for each } e \in E. \tag{13}$$

An alternative statement of (13) is that all reduced costs are non-negative.

We show that (13) implies optimality of x^*, i.e., $w(x) \geq w(x^*)$ holds for any $x \in L$. Since w is convex and any point in L is a nonnegative combination of cycles, it suffices to prove this for $x = x^* + \xi$, where ξ is a cycle of L. Let $F_\xi = (A, B)$. If $e \in A$, we have $x_e = x_e^* + 1$ and $c_e^+(x_e) \geq c_e^+(x_e^*)$, whence $w_e(x_e) - w_e(x_e^*) \geq c_e^+(x_e^*)(x_e - x_e^*) = c_e^+(x_e^*)$. If $e \in B$, we have $x_e = x_e^* - 1$ and $c_e^-(x_e) \leq c_e^-(x_e^*)$, whence $w_e(x_e) - w_e(x_e^*) \geq -c_e^-(x_e^*)$. Therefore,

$$w(x) - w(x^*) \geq \sum_{e \in A} c_e^+(x_e^*) - \sum_{e \in B} c_e^-(x_e^*) \geq h(A) - h(B) = h\xi = 0,$$

as required.

Remark. Suppose $x^* \in L$ is optimal as proven by $h^* \in L^\perp$ satisfying (13). Then, arguing similarly to above, we can conclude that $x \in L$ is optimal if and only if $c_e^+(r) = c_e^-(r) = h_e^*$ for each $e \in E$ and any r in the open interval $(\min\{x_e, x_e^*\}, \max\{x_e, x_e^*\})$. That is, x must differ from x^* only by augmentations around cycles $F = (A, B)$ where w_e is linear between x_e^* and x_e for every element $e \in A \cup B$, and where the (linear) cost of F at x is zero.

Lemma 2.2 is crucial for our *Min Mean Canceling Method* (MMCM). We assume that there is a procedure available to solve the following auxiliary problem:

(AP) : Given $x \in L$, find $\lambda = \lambda(x)$, and if $\lambda > 0$, find a minimum mean cycle $F = (A, B)$ at x and $h \in L^\perp$ satisfying (11).

[There exist general strongly polynomial methods for computing "min mean values" under various assumptions; see Karzanov [18], Megiddo [25, 26] or Radzik [32]. The Megiddo and Radzik approaches both assume that we have a black box available for solving the associated linearized problem. Megiddo requires that the algorithm be "linear" in a technical sense, but it only needs to be able to recognize if its objective is negative. Radzik allows a wider range of algorithms, but they must be able to find a most negative linear solution. The approach in [18] assumes that the objects of interest can be represented as the extreme rays of a cone which is the intersection of a linear space and the non-negative orthant, but then requires only that we can solve linear programs over this cone. In particular, the approach to solving (AP) in [18] also works well with spaces L of bigger, though bounded, fractionality, as we shall see in Section 8. In the important cases of circulations or co-circulations in a network, there are fast methods to solve (AP), as explained in Section 7. Finally, Section 4 gives an alternate method to solve (5) that does not need to solve (AP).]

MMCM starts with any reasonable $x \in L$, and computes $\lambda = \lambda(x)$. If $\lambda = 0$, stop, as x is optimal (by Lemma 2.1). Otherwise ($\lambda > 0$), find F and h as in (AP). For each element $e \in F$, use answer (ii) of the oracle (O) for w_e to compute the local step length

$$\varepsilon_e = \text{an increment } \Delta \text{ s.t.} \begin{cases} c_e^-(x_e + \Delta) \leq c_e^+(x_e) + \lambda = h_e \leq c_e^+(x_e + \Delta) & \text{if } e \in A; \\ c_e^-(x_e - \Delta) \leq c_e^-(x_e) - \lambda = h_e \leq c_e^+(x_e - \Delta) & \text{if } e \in B \end{cases} \tag{14}$$

(the equalities come from the second part of Lemma 2.2); clearly $\varepsilon_e > 0$. The overall step length ε is $\min\{\varepsilon_e : e \in A \cup B\}$. That is, we choose ε as a step length that makes some reduced cost (w.r.t. h) in F increase to become non-negative, while other reduced costs stay non-positive. Now push ε around F forming x' as in (10), and make x' the new current point in L. Continue until λ is "small enough".

We claim that this method converges geometrically in λ. We show this using two lemmas. The first is similar to a lemma in Goldberg and Tarjan [9], and the second is similar to a lemma in Frank and Karzanov [6] with an improvement generalizing a result in Radzik and Goldberg [33]. Note that (14) and (6) imply

$$(15) \qquad \begin{aligned} c_e^+(x_e) \le c_e^-(x'_e) \le c_e^+(x_e) + \lambda \quad & \text{for } e \in A; \\ c_e^-(x_e) \ge c_e^+(x'_e) \ge c_e^-(x_e) - \lambda \quad & \text{for } e \in B. \end{aligned}$$

Lemma 2.3 *If x' is obtained from x at an iteration then $\lambda(x') \le \lambda(x)$.*

Proof: Let $h \in L^\perp$ be a co-circulation proving the optimality of $\lambda = \lambda(x)$, i.e., h is as in Lemma 2.2. It is enough to show that (11) holds for $c^+(x')$ and $c^-(x')$, because then for any $F' = (A', B') \in \mathcal{F}$ we have

$$c(x', F') \ge h(A') - \lambda|A'| - h(B') - \lambda|B'| = h(A') - h(B') - \lambda|F'| = -\lambda|F'|,$$

or $\overline{c}(x', F') \ge -\lambda$, implying that $\lambda(x') \le \lambda(x)$.

Since x' and x coincide outside F, it suffices to examine only elements in the minimum mean cycle $F = (A, B)$ at x that is chosen at the iteration. For $e \in A$, the inequality

$$c_e^+(x'_e) \ge c_e^+(x_e)$$

(as $x'_e > x_e$) together with (11) implies

$$c_e^+(x'_e) \ge h_e - \lambda;$$

and the inequality

$$c_e^-(x'_e) \le c_e^+(x_e) + \lambda$$

(by (15)) together with the equality $c_e^+(x_e) = h_e - \lambda$ (by Lemma 2.2) implies

$$-c_e^-(x'_e) \ge -c_e^+(x_e) - \lambda = -h_e + \lambda - \lambda > -h_e - \lambda,$$

as required. The proof for $e \in B$ is similar. □

For the next lemma we superscript everything by the iteration number.

Lemma 2.4 *After every $\psi^\perp$ iterations $\lambda = \lambda(x)$ decreases by a factor of at least $(1 - 1/2\phi)$, i.e, $\lambda^{i+\psi^\perp} \le \lambda^i(1 - 1/2\phi)$.*

Proof: Fix $h := h^i \in L^\perp$ proving the optimality of $\lambda = \lambda(x^i)$, and consider iterations $i, i+1, \ldots, i+\psi^\perp$. For each $j = i, \ldots, i+\psi^\perp$, call $e \in E$ *positively (negatively) close to h* if $c_e^+(x_e^j) - h_e \ge -\lambda/2$ (resp. $h_e - c_e^-(x_e^j) \ge -\lambda/2$), and otherwise call it positively (negatively) *far from h*. Note that $c_e^+(x_e) - c_e^-(x_e) \ge 0$ implies that e cannot be positively and negatively far at the same time. Equivalently, we could talk instead of the reduced costs as being close to or far from zero. The proof rests on the following two claims.

Claim 1: If for all $k = i, \ldots, j-1$ the chosen cycle $F^k = (A^k, B^k)$ is such that each $e \in A^k$ is positively far from h and each $e \in B^k$ is negatively far from h, then $c_e^+(x_e^j) - h_e \ge -\lambda$ and $h_e - c_e^-(x_e^j) \ge -\lambda$ for all $e \in E$.

Proof: We use induction on j. Then $c_e^+(x_e^j) \ge c_e^+(x_e^{j-1}) \ge h_e - \lambda$ for $e \in E - B^{j-1}$ and $-c_e^-(x_e^j) \ge -c_e^-(x_e^{j-1}) \ge -h_e - \lambda$ for $e \in E - A^{j-1}$. Consider the remaining two cases. For $e \in A^{j-1}$ we have

$$c_e^-(x_e^j) \le c_e^+(x_e^{j-1}) + \lambda^{j-1}$$

(by (15)) and

$$c_e^+(x_e^{j-1}) < h_e - \lambda/2$$

(as e is positively far from h). Putting these inequalities together yields $c_e^-(x_e^j) < h_e - \lambda/2 + \lambda^{j-1} < h_e + \lambda^{j-1}$, whence $-c_e^-(x_e^j) > -h_e - \lambda^{j-1} \geq -h_e - \lambda$ (as $\lambda^{j-1} \leq \lambda$ by Lemma 2.3). The case for $e \in B^{j-1}$ is similar. □

We say that an element e in a cycle $F = (A, B) \in \mathcal{F}$ (i.e., $e \in A \cup B$) is close (far) if e is positively close (far) whenever $e \in A$, and negatively close (far) whenever $e \in B$.

Claim 2: There is some iteration j ($i < j \leq i + \psi^\perp$) such that either (i) $\lambda^j < \lambda/2$, or (ii) F^j includes an element close to h.

Proof: Suppose that for $j = 1, \ldots, i+\psi^\perp - 1$ no iteration j satisfies (i), i.e., $\lambda^j \geq \lambda/2$, and suppose that all iterations up to and including j have all elements of their cycles far from h, so that Claim 1 applies. Denote by P (N) the current set of elements positively (resp. negatively) close to h, and let $C = P \cap N$. By the *rank* of C we shall mean the rank of the matrix formed by the columns of K corresponding to elements of C, where K is a matrix representing $L^\perp$. Obviously, the rank of K is $\psi^\perp - 1$. We shall show that each iteration j increases the rank of C by at least one. This implies that after iteration $i+\psi^\perp - 1$, C meets the support of every cycle $\xi \in L$. Therefore, the cycle $F^{i+\psi^\perp}$ must include some element of C, proving (ii).

First we show that if $e \in E$ belongs to P (N) before iteration j, then e remains in P (resp. N) after the iteration. Consider a positively close e (the proof is similar for a negatively close e). The only way for e to become positively far is for $c_e^+(x_e)$ to decrease, which implies that $e \in B^j$. But then e is negatively far from h at iteration j, i.e., $-c_e^-(x_e^j) < -h_e - \lambda/2$. This together with $c_e^+(x_e^{j+1}) \geq c_e^-(x_e^j) - \lambda^j$ (by (15)) yields $c_e^+(x_e^{j+1}) > h_e + \lambda/2 - \lambda^j \geq h_e - \lambda/2$, so that e is not positively far from h after iteration j. Thus each of the sets P, N, C is monotone non-decreasing.

Next we show that there is at least one element e in F^j such that iteration j makes e close (recall that all elements in F^j are far). Lemma 2.3 says that $\lambda^j \leq \lambda$. By the choice of ε^j, either (a) A^j contains e with $c_e^+(x_e^{j+1}) \geq c_e^+(x_e^j) + \lambda^j$, or (b) B^j contains e with $c_e^-(x_e^{j+1}) \leq c_e^-(x_e^j) - \lambda^j$. Suppose (a) takes place (case (b) is similar). Then (using Claim 1 for j) we get

$$c_e^+(x_e^{j+1}) \geq c_e^+(x_e^j) + \lambda^j \geq h_e - \lambda + \lambda^j \geq h_e - \lambda/2.$$

Thus iteration j adds this e as a new element to C.

Finally, we show that adding this e increases the rank of C. Suppose to the contrary that the rank of the set C at the beginning of iteration j equals the rank of $C \cup \{e\}$. Then a minimal vector $\xi' \in L^\perp$ proving the dependence of e on C corresponds to a co-cycle, and $e \in \mathrm{supp}(\xi')$. Let $\xi \in L$ be the vector corresponding to F^j. As $\xi\xi' = 0$ and $\xi_e\xi'_e \neq 0$, there exists another element $e' \neq e$ in both F^j and $\mathrm{supp}(\xi')$. But e is the only far element in F^j, so $e' \in C$. But each element of F^j was far at the beginning of iteration j; a contradiction. □

We can assume that $\lambda^j \geq \lambda/2$ for $j = i, \ldots, i + \phi^\perp$, else Lemma 2.3 shows that we are already done. Thus there must be an iteration j satisfying (ii) of Claim 2, i.e., with either (a) A^j contains an element u positively close to h, or (b) B^j contains an element u' negatively close to h. Consider the smallest j with this property and assume that (a) takes place (case (b) is similar). Using Claim 1, we have

$$\begin{aligned} c(x^j, F^j) &= c(x^j, F^j) - h(A^j) + h(B^j) \\ &= \sum_{e \in A^j - \{u\}} (c_e^+(x_e^j) - h_e) + (c_u^+(x_u^j) - h_u) + \sum_{e \in B^j} (h_e - c_e^-(x_e^j)) \\ &\geq -\lambda(|A^j| - 1) - \lambda/2 - \lambda|B^j| = -\lambda|F^j| + \lambda/2, \end{aligned}$$

whence $\lambda^j \leq \lambda(1 - 1/2\phi)$. □

Theorem 2.5 *Let β be a bound on $\lambda(x)$ for the initial $x \in L$, and let ε be the absolute accuracy desired in our final answer. Then MMCM uses $O(\psi^\perp \phi \log(\beta/\varepsilon))$ iterations.*

Proof: Lemma 2.4 shows that $2\psi^\perp\phi$ iterations reduce λ by a factor of at least e^{-1}. Thus $2\psi^\perp\phi \log(\beta/\varepsilon)$ iterations reduce λ from β to ε. □

Remark: We note that all arithmetic operations involved in this method (and all succeeding variants) are just addition (subtraction, and comparison) of rationals, and division of a rational by an integer not exceeding $|E|$ (while computing $\lambda(x)$). Thus as long as we start with a rational x whose size is not too large, we can arrange the computations so that the size of all intermediate numbers is bounded by a polynomial in the size of the input.

3 Cost Distance, and Dual and Primal Convergence

In the previous section we established fast convergence of the method in terms of minimum mean cycle value $\lambda(x)$. Since $\lambda(x)$ is the best possible lower bound on reduced costs over all $h \in L^\perp$, it is a measure of dual convergence. However, it remains unclear whether this fact forces fast convergence of the current point x to an optimal point. In this section we introduce another, closely related measure of the quality of the current point x which will enable us to show fast convergence of x to an optimal point in some special cases of w, and to compute exact optimal solutions in Sections 5 and 6.

Given $x \in L$ and $h \in L^\perp$, we introduce a kind of distance from x to h. First for $e \in E$ define

$$\langle x_e, h_e \rangle = \begin{cases} h_e - c_e^+(x_e) & \text{if } c_e^+(x_e) < h_e, \\ c_e^-(x_e) - h_e & \text{if } c_e^-(x_e) > h_e, \\ 0 & \text{if } c_e^-(x_e) \leq h_e \leq c_e^+(x_e); \end{cases} \tag{16}$$

and then define

$$\langle x, h \rangle = \max_{e \in E} \langle x_e, h_e \rangle. \tag{17}$$

That is, this distance is the absolute value of the most negative reduced cost for x and h. Clearly, $\langle x, h \rangle \geq 0$. Moreover, if $\langle x, h \rangle = 0$ then x is optimal as (16) turns into (13). We call $\langle x, h \rangle$ the *cost distance* from x to h. The next lemma reveals its relation to costs of cycles.

Lemma 3.1 *Let x^* be an optimal solution to (5) and $h^* \in L^\perp$ prove its optimality, i.e., x^* and h^* satisfy (13). Let $x \in L$ and suppose that $\delta = \langle x, h^* \rangle > 0$. Then there exists a cycle $F \in \mathcal{F}$ whose cost $c(x, F)$ is at most $-\delta$.*

Proof: Let $u \in E$ be an element with $\langle x_u, h_u^* \rangle = \delta$. Then $x_u \neq x_u^*$ (otherwise $\delta = 0$). Suppose for definiteness that $x_u < x_u^*$ (the reverse case is similar). Represent $z = x^* - x$ as $z = \alpha_1 \xi^1 + \ldots + \alpha_k \xi^k$, where $\alpha_1, \ldots, \alpha_k > 0$ and $\xi^1, \ldots, \xi^k$ are cycles of L satisfying (8). Since $z_u > 0$, there is ξ^i with $\xi_u^i = 1$; let $F = (A, B) \in \mathcal{F}$ correspond to ξ^i. Then $u \in A$, and from (8) it follows that

$$x_e^* > x_e \text{ for } e \in A \quad \text{and} \quad x_e^* < x_e \text{ for } e \in B.$$

By (6) and (13), $c_e^+(x_e) \leq c_e^-(x_e^*) \leq h_e^*$ for $e \in A$ and $c_e^-(x_e) \geq c_e^+(x_e^*) \geq h_e^*$ for $e \in B$. Also $c_u^+(x_u) = h_u^* - \delta$. Thus,

$$c(x, F) = \sum_{e \in A} c_e^+(x_e) - \sum_{e \in B} c_e^-(x_e) \leq -\delta + h^*(A) - h^*(B) = -\delta,$$

as required. □

Corollary 3.2 $\langle x, h^* \rangle \leq \phi\lambda(x)$.

Proof: If $\langle x, h^* \rangle > \phi\lambda(x)$, Lemma 3.1 shows that there is a cycle F with $c(x, F) < -\phi\lambda(x)$. But then $-\lambda(x) \leq \overline{c}(x, F) < -\phi\lambda(x)/|F| \leq -\lambda(x)$ (as $\phi \geq |F|$); a contradiction. □

In Section 5 we will also need to bound $\lambda(x)$ by $\langle x, h^* \rangle$ on the other side:

Lemma 3.3 $\lambda(x) \leq \langle x, h^* \rangle$.

Proof: Recall that $\lambda(x)$ is the minimum $\lambda \geq 0$ such that there is an $h \in L^\perp$ satisfying (11). Since h^* is one such h, we have

$$\lambda(x) \leq \max_e \max\{h^* - c_e^+(x_e), c_e^-(x_e) - h^*, 0\} = \langle x, h^* \rangle . \quad \square$$

At first glance, it might seem that the cost distance $\langle x, h^* \rangle$ is not much help in estimating the distance from the current point x to an optimal x^*. Indeed, if we replace each $w_e(r)$ by $w'_e(r) = kw_e(r/k)$ for an integer $k > 1$, then the right cost c'^+ for w' satisfies $c'^+_e(kr) = c_e^+(r)$, and similarly for the left cost cost c'^-. In other words, if we define $x' = kx$ and $x'^* = kx^*$, then x'^* is optimal for the problem with w', while h^* remains optimal for w'. Therefore, $\langle x', h^* \rangle = \langle x, h^* \rangle$, whereas the ℓ_∞-distance between x' and x'^* has increased by a factor of k.

However, the cost distance is indeed a measure of primal convergence for an important class of functions w. Let us say that w_e is *fast growing* (f.g.) *with speed* α ($\alpha > 0$) in an (open or closed) interval σ of $\mathbf{R}$ if for any $r, r' \in \sigma$, $r < r'$, one has $c_e^-(r') - c_e^+(r) \geq \alpha(r' - r)$. For example, the quadratic function $w_e(r) = ar^2 + br + d$ ($a > 0$) is fast growing with $\alpha = 2a$ in $(-\infty, \infty)$. Other examples of f.g. functions are piecewise quadratic, cubic, and exponential functions. If w_e is fast growing for all $e \in E$, we say that w is fast growing, and its speed is defined to be the smallest speed of any w_e.

Lemma 3.4 *If w_e is fast growing with speed α in an interval σ, and both the current value x_e and optimal value x_e^* are in σ, then $|x_e - x_e^*| \leq \alpha^{-1}\langle x_e, h_e^* \rangle$.*

Proof: Assume that $x_e < x_e^*$ (the reverse case is similar), so that $\langle x_e, h_e^* \rangle = h_e^* - c_e^+(x_e)$. Then w_e f.g. means that $x_e^* - x_e \leq \alpha^{-1}(c_e^-(x_e^*) - c_e^+(x_e))$. But optimality of x^* and h^* implies that $h^* \geq c_e^-(x_e^*)$ (by (13)). Plugging this into the bound on $x_e^* - x_e$ and using the value for $\langle x_e, h_e^* \rangle$ gives the result. □

Thus, if w is f.g. with speed α, and if for each $e \in E$ both x_e and x_e^* are in an interval where w_e is fast growing, then the ℓ_∞-distance $\|x - x^*\|_\infty$ is bounded by the cost distance $\langle x, h^* \rangle$ times α^{-1}.

4 A Faster Cancel and Tighten Method

Although MMCM achieves geometric convergence, each iteration requires a potentially expensive min mean cycle computation (though there exist fairly fast algorithms for this in the case of circulations and co-circulations in networks, see Section 7). Moreover, it can take up to $\psi^\perp$ iterations to reduce $\lambda(x)$ by the factor $(1 - 1/2\phi)$ (these bounds can be tight, see [33]). It is reasonable to ask if there is a faster way to achieve the same reduction, particularly since both Goldberg and Tarjan [9] (the "Cancel and Tighten" algorithm) and Ervolina and McCormick [3] (the "Dual Cancel and Tighten" algorithm) show how to do this in the cases of circulations and co-circulations in networks, respectively. This section shows how these same ideas work in the general case.

The outline of the improved method is that we alternate *Cancel* steps that improve a primal x, and *Tighten* steps that improve a dual h. The Cancel step "cancels" (augments x around, in the sense

of (10)) cycles whose elements are restricted to be sufficiently far away from the optimality condition (13). When no such cycles remain, we can compute a direction d in which to move our current dual estimate h that guarantees a $(1-1/2\psi)$ decrease (which is almost as good as the $(1-1/2\phi)$ decrease in MMCM). Thus our speedup comes from two sources: first, we do not have to compute minimum mean cycles, but rather we can cancel any sufficiently far cycles. Second, we get a decrease after only one iteration of Cancel and Tighten, rather than $\psi^\perp$ iterations of MMCM.

Instead of computing a min mean cycle to find the exact value of $\lambda(x)$, we keep a dual vector $h \in L^\perp$ that approximately satisfies the optimality condition (13) with x. We define $\overline{\lambda}$ as the absolute value of the most negative reduced cost w.r.t. h (i.e., $\overline{\lambda} = \langle x, h \rangle$), so that $\overline{\lambda}$ is an upper bound on $\lambda(x)$. Note that h proves that $\overline{\lambda}$ is a valid upper bound (compare with (11)) via satisfying

$$c_e^+(x_e) - h_e \geq -\overline{\lambda} \quad \text{and} \quad h_e - c_e^-(x_e) \geq -\overline{\lambda} \quad \text{for all } e \in E. \tag{18}$$

Note that if x and h satisfy (18) with $\overline{\lambda} = 0$, then they are optimal, by (13). Thus we try to improve x and h by decreasing the gap between the costs and h. Define the subset $P^{\text{far}} \subseteq E$ (resp. $N^{\text{far}} \subseteq E$) consisting of elements that are positively (negatively) far from h as defined in the proof of Lemma 2.4, and define $E^{\text{far}} = P^{\text{far}} \cup N^{\text{far}}$.

Call a cycle $F = (A, B)$ with $A \subseteq P^{\text{far}}$ and $B \subseteq N^{\text{far}}$ a *far cycle*. The Cancel step consists of finding and canceling far cycles until no more exist. Define $C = E - (P^{\text{far}} \cup N^{\text{far}})$ as the set of elements close to h. Then, at each inner iteration of the Cancel step, we find a minimal support non-zero solution to the system

$$\begin{aligned} Mz &= 0 \\ z_e &\geq 0 \quad \text{for } e \in P^{\text{far}} \\ &\leq 0 \quad \text{for } e \in N^{\text{far}}. \\ &= 0 \quad \text{for } e \in C. \end{aligned} \tag{19}$$

System (19) can be solved using linear programming techniques.

The computation of the step length ε for a far cycle $F = F_z = (A, B)$ found by solving (19) resembles (14), but is a bit different since the gaps between the costs and h_e's do not all have the same value $-\lambda$ here. Again using answer (ii) of the oracle (O) for w_e, we compute

$$\varepsilon_e = \text{an increment } \Delta \text{ s.t.} \begin{cases} c_e^-(x_e + \Delta) \leq h_e \leq c_e^+(x_e + \Delta) & \text{if } e \in A; \\ c_e^-(x_e - \Delta) \leq h_e \leq c_e^+(x_e - \Delta) & \text{if } e \in B \end{cases} \tag{20}$$

and the overall step length $\varepsilon = \min\{\varepsilon_e : e \in A \cup B\}$; then $\varepsilon > 0$.

Lemma 4.1 *A Cancel step preserves (18), and cancels at most $\psi^\perp$ far cycles.*

Proof: Since we are canceling cycles consisting only of elements far from h, Claim 1 in the proof of Lemma 2.4 shows that we cannot create any elements violating (18). Furthermore, the proof of Claim 2 shows that we cannot create any new elements far from h either: Comparing (14) with (20) in the case where $e \in P^{\text{far}}$ (the other case is similar), since $h_e = c_e^+(x_e) + (h_e - c_e^+(x_e))$ for element e in (20), we are effectively using $h_e - c_e^+(x_e)$ for the λ in (14); since $h_e - c_e^+(x_e) > \overline{\lambda}/2$, case (i) of Claim 2 cannot hold. Thus $|P^{\text{far}}|$ and $|N^{\text{far}}|$ are monotone non-increasing.

At least one element in each far cycle canceled (namely an element determining ε) must become close to h and so drop out of P^{far} or N^{far}. In the same way as in the proof of Claim 2, each such newly close element must increase the rank of the submatrix of K induced by the elements close to h, and hence we can cancel at most $\psi^\perp$ cycles before no more far cycles exist. □

It might seem that canceling $\psi^\perp$ far cycles might be as much work as canceling $\psi^\perp$ min mean cycles, but for the important specializations to networks in Section 7, it will turn out that the total

work to cancel the far cycles in one Cancel can be done much faster than even one min mean cycle computation.

The Tighten step now looks for a *sufficient improving direction*, which is a vector $d \in \mathbf{R}^E$ that satisfies

[A] d is an integral vector in $L^\perp$.

[B] For all e, $|d_e| \leq \psi - 1$.

[C] For all $e \in P^{\text{far}}$, $d_e \geq +1$.

[D] For all $e \in N^{\text{far}}$, $d_e \leq -1$.

Lemma 4.2 *If d is a sufficient improving direction and we replace h by $h' := h - (\overline{\lambda}/2\psi)d$, then $\overline{\lambda}$ is reduced to at most $\overline{\lambda}' := (1 - 1/2\psi)\overline{\lambda}$, i.e., (18) holds for h' and $\overline{\lambda}'$.*

Proof: By [A], $h' \in L^\perp$.

Suppose that e was positively close to h. Then by [B]

$$\begin{aligned} c_e^+(x_e) - h'_e &= c_e^+(x_e) - h_e + d_e\overline{\lambda}/2\psi \\ &\geq c_e^+(x_e) - h_e - (\psi - 1)\overline{\lambda}/2\psi \\ &= c_e^+(x_e) - h_e - \overline{\lambda}/2 + \overline{\lambda}/2\psi. \end{aligned}$$

Since $c_e^+(x_e) - h_e \geq -\overline{\lambda}/2$ for e positively close to h, we have that this e also satisfies (18) in the positive direction for h' and $\overline{\lambda}'$. In the negative direction, we can assume that $h_e - c_e^-(x_e) \geq 0$, else we would deal with this in the negatively close or far from h case. Then, by [B],

$$\begin{aligned} h'_e - c_e^-(x_e) &= (h_e - d_e\overline{\lambda}/2\psi) - c_e^-(x_e) \\ &\geq h_e - c_e^-(x_e) - (\psi - 1)\overline{\lambda}/2\psi \\ &\geq -(\psi - 1)\overline{\lambda}/2\psi > -\overline{\lambda}'. \end{aligned}$$

The case where e is negatively close to h is similar.

By [C], if $e \in P^{\text{far}}$, then

$$\begin{aligned} c_e^+(x_e) - h'_e &= c_e^+(x_e) - (h_e - d_e\overline{\lambda}/2\psi) \\ &\geq c_e^+(x_e) - h_e + \overline{\lambda}/2\psi. \end{aligned}$$

Since e satisfied (18) in the positive direction for h and $\overline{\lambda}$, it also satisfies (18) in the positive direction for h' and $\overline{\lambda}'$. The negative direction is similar to the case above, and the case where $e \in N^{\text{far}}$ is similar. □

Lemma 4.2 shows that moving a distance of $\overline{\lambda}/2\psi$ in a sufficient improving direction reduces $\overline{\lambda}$ by a factor of at least $(1 - 1/2\psi)$, as desired. (An alternate way of computing a tighter step length that should work well in practice is suggested in [9], and it can also be adopted here. However, it does not improve the worst-case bound.) If we can show how to (efficiently) compute a sufficient improving direction, then this will finish the Tighten step.

As a start on computing such a d, note that when Cancel is done the system (19) has $z = 0$ as its only feasible solution. In particular, for each $e \in E^{\text{far}}$ there is no far cycle containing e. The following shows that there is a co-cycle D^e with corresponding incidence vector ξ^e proving this fact.

Lemma 4.3 *An element $u \in E^{\text{far}}$ is contained in no far cycle if and only if there is a co-cycle $\xi^u \in L^\perp$ such that $\xi^u_u \neq 0$, $\xi^u_e \geq 0$ for all $e \in P^{\text{far}}$ and $\xi^u_e \leq 0$ for all $e \in N^{\text{far}}$.*

Proof: Applying Farkas' Lemma to (19), we observe that u is in no far cycle if and only if there is a vector $\xi = y^T M$ feasible to

$$\begin{array}{rcll} \xi_e & \geq & 0 & \text{for } e \in P^{\text{far}} \\ \xi_e & \leq & 0 & \text{for } e \in N^{\text{far}} \\ \xi_u & \neq & 0. & \end{array} \tag{21}$$

Now if (21) has a solution in $L^\perp$ then it has a solution that is a co-cycle ξ^u. Conversely, if ξ^u as in the lemma exists then it satisfies (21), thus proving (by Farkas' Lemma) that no far cycle contains u. □

Now Tighten can compute d as follows. Find a subset B^{far} of E^{far} such that $\{\xi^u : u \in B^{\text{far}}\}$ forms a basis for $\{\xi^u : u \in E^{\text{far}}\}$. Then define d to be $\sum_{u \in B^{\text{far}}} \xi^u$.

Lemma 4.4 *This d is a sufficient improving direction.*

Proof: Since d is a sum of incidence vectors of co-cycles, certainly d satisfies [A].

The dimension of the subspace of $L^\perp$ spanned by $\{\xi^e : u \in E^{\text{far}}\}$ (i.e., the cardinality of B^{far}) is at most the dimension of $L^\perp$, which is $\psi - 1$. This proves [B].

Consider $e \in E^{\text{far}}$. By Lemma 4.3, for all $u \in E^{\text{far}}$, $\xi^u_e \geq 0$ if $e \in P^{\text{far}}$ and $\xi^u_e \leq 0$ if $e \in N^{\text{far}}$. Furthermore, for the vector ξ^e we have $\xi^e_e \neq 0$. Now since $\{\xi^u : u \in B^{\text{far}}\}$ generates all of $\{\xi^{e'} : e' \in E^{\text{far}}\}$, there must at least one $u \in B^{\text{far}}$ such that $\xi^u_e \neq 0$. By Lemma 4.3, every such ξ^u_e must have the same sign as ξ^e_e. This proves [C] and [D]. □

Computationally, to construct a sufficient improving direction d efficiently it suffices to find a basic feasible solution of the linear program obtained by adding to (21) the constraints $-\psi+1 \leq \xi_e \leq \psi-1$ $(e \in E)$, $\xi_e \geq 1$ $(e \in P^{\text{far}})$ and $\xi_e \leq -1$ $(e \in N^{\text{far}})$. Also, we shall see in Section 7 that computing a sufficient improving direction can be done very efficiently when we specialize to circulations and co-circulations in networks.

Lemma 4.2 says that a single round of Cancel and Tighten reduces $\overline{\lambda}$ to at most $(1 - 1/2\psi)\overline{\lambda}$. Since this single round of Cancel and Tighten achieves almost the same reduction that takes $\psi^\perp$ iterations of MMCM, Theorem 2.5 now becomes:

Theorem 4.5 *Let β be a bound on $\lambda(x)$ for the initial $x \in L$, and let ε be the accuracy desired in our final answer. Then Cancel and Tighten uses $O(\psi \log(\beta/\varepsilon))$ iterations.* □

5 (Strongly) Polynomial Bounds on Computing Exact Solutions with Piecewise Linear Costs

We continue to consider the case of unimodular L. Now we will show that either of our generic algorithms (MMCM or Cancel and Tighten) can find an exact solution in the piecewise linear case in polynomial time when all the data is integral (which is a common and defensible assumption for linear costs). We will also show that MMCM has a strongly polynomial bound in this case, and that Cancel and Tighten can be modified slightly to also have a strongly polynomial bound.

Note that if we start with an integral x^0 (often $x^0 = 0$ will work), then with integral data the algorithm will always maintain an integral solution. Thus in this case we will be able to guarantee an integer optimal solution when the data is integral. Denote the total number of breakpoints over all

the piecewise linear element cost functions by B. Note that since each breakpoint has an associated value and an associated adjacent slope, we can justifiably use B in a strongly polynomial bound.

Note that piecewise linear functions are not fast growing in the sense of Section 3. Thus we will need to develop different techniques for proving that our methods can find exact optimal solutions.

5.1 A Polynomial Bound

We borrow an optimality characterization from Goldberg and Tarjan [9] that depends directly on $\lambda(x)$ and not on $\langle x, h^* \rangle$.

Lemma 5.1 *In the piecewise linear case, if $\lambda(x) < 1/\phi$ then x is exactly optimal.*

Proof: Let $h \in L^\perp$ be a set of dual variables proving that $\lambda(x) < 1/\phi$, and consider the reduced costs w.r.t. h. Note that if F is any cycle, then its x-cost w.r.t. c^+ and c^- is the same as its x-cost w.r.t. the reduced costs. But $\lambda(x) < 1/\phi$ implies that the reduced cost of every element in F is more than $-1/\phi$, which implies that the x-cost of F is strictly larger than -1. But the cost of F is an integer, so the cost of F is non-negative. Thus, by Lemma 2.1, x must be optimal. □

Let C denote the largest number appearing among the finite slopes in the cost functions, and let x^0 denote our initial circulation. Each finite $c_e^-(x_e^0)$ and $c_e^+(x_e^0)$ will be equal to one of the $O(B)$ possible slopes, so that $\lambda(x^0)$ and $\overline{\lambda}$ are at most C. Thus we need to reduce $\lambda(x)$ or $\overline{\lambda}$ only from at most C to below $1/\phi$ (by Lemma 5.1), a gap of ϕC, in order to achieve optimality. Thus by essentially the same proof as for Theorem 2.5 we get:

Theorem 5.2 *For piecewise linear objective functions, MMCM needs $O(\psi^\perp \phi \log(\phi C))$ iterations and Cancel and Tighten needs $O(\psi \log(\phi C))$ iterations to achieve exact optimality.* □

5.2 A Strongly Polynomial Bound

The Goldberg and Tarjan [9] proof of strong polynomiality for the classic linear case depends in part on knowing that the cost of a cycle does not change, even after many iterations. But for us, the cost of an element e can change substantially if its x_e moves into a different linear piece, which changes the cost of any cycle containing e. Thus we cannot directly adapt the proof in [9]. However, the alternate proof below (which has the same flavor as a proof that [33] uses for the case of networks) is just as strong as the ones in [9, 3] as we will be able to derive exactly the same bounds on running time in Section 7. Also, this proof is somewhat simpler than the proofs in [9, 3].

We use the very close relationship between $\lambda(x)$ and $\langle x, h^* \rangle$ given by Corollary 3.2 and Lemma 3.3. These say that neither number can get very far from the other. A key observation is that by (16) and (17), $\langle x, h^* \rangle$ always equals $h_e^* - c_e^+(x_e)$, $c_e^-(x_e) - h_e^*$, or 0, for some element e. Each $c_e^+(x_e)$ and $c_e^-(x_e)$ equals one of the only $O(B)$ slopes, and h^* stays fixed throughout the algorithm. Thus there are only $O(B)$ different values that $\langle x, h^* \rangle$ can take, a strongly polynomial number. Since $\lambda(x)$ declines geometrically, by Lemma 3.3 it will take only a strongly polynomial number of iterations to get $\lambda(x)$ far enough below from the previous value of $\langle x, h^* \rangle$ that, by Corollary 3.2, it forces $\langle x, h^* \rangle$ to decrease at least to its next smaller possible value. This gives an overall strongly polynomial bound.

Theorem 5.3 *In the piecewise linear case, MMCM requires $O(B\psi^\perp \phi \log \phi)$ iterations to compute an exact optimal circulation.*

Proof: Define a *big iteration* as a sequence of iterations that reduce $\lambda(x)$ by a factor of more than $1/\phi$; by the arguments in the proof of Theorem 2.5, a big iteration consists of $O(\psi^\perp \phi \log \phi)$ min

mean cycle cancellations. Let x be our point before a big iteration, and x' be the point after a big iteration. Then

$$\begin{aligned} \langle x', h^* \rangle &\leq \phi\lambda(x') && \text{(by Corollary 3.2)} \\ &< \lambda(x) && \text{(by the definition of big iteration)} \\ &\leq \langle x, h^* \rangle && \text{(by Lemma 3.3).} \end{aligned} \tag{22}$$

This shows that $\langle x, h^* \rangle$ strictly decreased during the big iteration. Thus after at most B big iterations, $\langle x, h^* \rangle$ has decreased B times, and its only possible remaining value is $\langle x, h^* \rangle = 0$, implying that x is optimal. This gives a bound of $O(B\psi^{\perp}\phi \log \phi)$, as required. □

Remark. We can remove the factor of $O(\log \phi)$ in Theorem 5.3 if we use an idea in [33, Lemma 11] worked out for the linear circulation case. Define the *optimal reduced cost* of x_e in the positive direction as $c_e^+(x_e) - h_e^*$, and in the negative direction as $h_e^* - c_e^-(x_e)$. As before there are only $O(B)$ possible values for optimal reduced costs, and so certainly only $O(B)$ possible negative optimal reduced costs. Note that since $h^* \in L^{\perp}$, the cost of a cycle at x w.r.t. optimal reduced costs is the same as its original cost at x.

Now, [33, Lemma 11] requires the three conditions on circulation x^j at iteration j: (i) $\lambda(x^j) \geq 0$, (ii) $\lambda(x^j) \leq \lambda(x^{j+1})$, and (iii) the absolute value of each optimal reduced cost at iteration j is at most $\phi\lambda(x)$. Condition (i) is clearly true, (ii) is Lemma 2.3, and (iii) can be derived from Corollary 3.2. Then [33, Lemma 11] bounds the maximum number of cancellations by $O(pqr)$, where p is the number of negative optimal reduced cost elements, q is maximum number of elements in a cycle, and r is the maximum number of cancellations necessary to reduce $\lambda(x)$ by $(1 - 1/q)$. For us, $p = O(B)$, $q = O(\phi)$, and $r = O(\psi^{\perp})$ (by Lemma 2.4).

The argument in Theorem 5.3 does not work for Cancel and Tighten because Lemma 3.3 might not be true for $\overline{\lambda}$ (but Corollary 3.2 *is* true with $\overline{\lambda}$ in place of $\lambda(x)$). That is, there is nothing in Cancel and Tighten to force $\overline{\lambda}$ to take large downwards steps, even when this is possible due to $\langle x, h^* \rangle$ becoming smaller than an optimal reduced cost. However, we can use an idea from [9] and replace a Tighten step by a min mean cycle computation once every big iteration, and so preserve the strongly polynomial bound. That is, at the beginning of a big iteration, we use a min mean cycle computation to replace $\overline{\lambda}$ by the exact value of $\lambda(x)$. In applications to networks, the (slow) min mean cycle iterations can be amortized over the (fast) Tighten iterations so as to not slow down the fast asymptotic running time of Cancel and Tighten; see Section 7. More formally,

Theorem 5.4 *In the piecewise linear case, Cancel and Tighten, with an exact min mean cycle computation in place of Tighten every $O(\psi \log \phi)$ iterations, computes an exact optimal circulation in $O(B\psi \log \phi)$ iterations.*

Proof: Redefine a big iteration to be enough Cancel and Tighten iterations to reduce $\overline{\lambda}$ by a factor of at least $1/\phi$, or $O(\psi \log \phi)$ iterations, with an exact min mean cycle iteration substituted for the first Tighten. Since we force $\overline{\lambda} = \lambda(x)$ at the beginning of each big iteration, (22) becomes true, and the rest of the proof of Theorem 5.3 goes through as before. □

6 Polynomial Bounds on Computing Exact Solutions with Quadratic Costs

For this case we assume that for each $e \in E$ the function w_e consists of a sequence of functions $f_e^i = a_e^i x_e^2 + b_e^i x_e + d_e^i$, $i = 1, \ldots, k_e$, where each f_e^i is defined on the interval between the breakpoints p_e^{i-1} and p_e^i ($p_e^{i-1} < p_e^i$); so $p_e^0 = -\infty$, $p_e^{k_e} = \infty$ and $f_e^i(p_e^i) = f_e^{i+1}(p_e^i)$. We assume that all a_e^i, b_e^i, d_e^i

are integers with $a_e^i \geq 0$, and that each finite p_e^i is an integer. If $a_e^i = 0$ then this piece is linear, else it is quadratic. Thus we have the case of *piecewise mixed linear and quadratic* costs. If all a_e^i are positive, then we have *piecewise quadratic* costs. Let A be the maximum number among all $4a_e^i$, $2|b_e^i|$ and (finite) $2|p_e^i|$ $(e \in E)$.

Note that if each w_e is purely quadratic then our problem (5) becomes quite easy as it turns into the linear algebra problem: find $x \in L$ and $h \in L^\perp$ satisfying the equalities $h_e = c_e(x_e)$ where $c_e(r)$ is the derivative $w_e'(r) = 2a_e r + b_e$ of $w_e(r)$. In the piecewise case optimal x and h can be defined by a linear system as follows. Each x_e will be bounded by p_e and p_e', and each h_e will be bounded by g_e and g_e'. Fix an optimal circulation x^*. If x_e^* is interior to piece i, set $p_e = p_e^{i-1}$ and $p_e' = p_e^i$ and force h_e to be the derivative of f_e^i by setting $g_e = g_e' = 2a_e^i x_e + b_e^i$. If instead $x_e^* = p_e^i$, then set $p_e = p_e' = p_e^i$ and set $g_e = c_e^-(p_e^i)$ and $g_e' = c_e^+(p_e^i)$. Then any x and h feasible to the following linear system must be optimal.

$$
\begin{array}{rcll}
Mx & = & 0 & \\
Kh & = & 0 & \\
p_e \leq & x_e & \leq p_e', & e \in E \\
g_e \leq & h_e & \leq g_e', & e \in E.
\end{array}
\tag{23}
$$

One can deduce that the absolute value of any subdeterminant of the matrix combining all constraints in (23) has upper bound A^m, where $m = |E|$. This implies the following.

Lemma 6.1 *In the piecewise mixed linear and quadratic case there are optimal solutions x^* and h^* such that every component x_e^* of x^* and every component h_e^* of h^* is a rational with numerator and denominator at most A^m.* □

We will also need a bound on the cost distance $\langle x^0, h^* \rangle$ between an initial x^0 and an optimal h^*. We can feasibly choose $x^0 = 0$. Each h_e^* is bounded above by $2a_e^i x_e^* + b_e^i$ for some i. By Lemma 6.1 each $x_e^* \leq A^m$, so $\langle x^0, h^* \rangle = O(A^{m+1})$, and by Lemma 3.3, $\beta = \lambda(x^0) \leq O(A^{m+1})$. Our goal will be to reduce $\langle x, h^* \rangle$ below $1/2A^{2m}$; by Corollary 3.2 to do this it suffices to reduce $\lambda(x)$ below $\varepsilon = 1/(2\phi A^{2m})$. The running time bounds involve $\log(\beta/\varepsilon)$, which is $O(m \log A)$ since $\phi = o(A^{2m})$.

6.1 The Piecewise Quadratic Case

Since w is strictly convex in this case, problem (5) has a unique optimal solution x^*. Thus, if we have reduced $|x_e - x_e^*|$ below $1/2A^{2m}$, then there is a unique rational closest to x_e with denominator at most A^m, and by Lemma 6.1 this rational must be x_e^*. Furthermore, x_e can be efficiently rounded to x_e^* using standard continued fractions techniques (see, e.g., [38]). We observe that if we define α to be the minimum of $2a_e^i$ among all e, i, then in this case each w_e is fast growing with speed α. Therefore, by Lemma 3.4 it suffices to decrease $\langle x_e, h_e^* \rangle$ (which is an upper bound on $\alpha|x_e - x_e^*|$) below $\alpha/2A^{2m}$. We summarize as follows (using Theorems 2.5 and 4.5).

Theorem 6.2 *In the pure and piecewise quadratic cases, MMCM needs $O(\psi^\perp \phi m \log A)$ iterations and Cancel and Tighten needs $O(\psi m \log A)$ iterations to find an exact optimal solution.* □

6.2 The Piecewise Mixed Linear and Quadratic Case

The difference between this case and the previous one is that some pieces f_e^i are linear, i.e., $a_e^i = 0$, so the w_e need not be fast growing.

Our strategy is to run the algorithm long enough to be able to round the elements whose x_e^*'s fall into quadratic pieces as above. After we fix x_e to be x_e^* on such pieces, we are left with a piecewise

linear problem where each element has at most two pieces with finite slope, which we can solve in strongly polynomial time as in Section 5.

More precisely, let $x \in L$ be any circulation. We say that e is a *quadratic element* of x if (i) x_e is interior to a quadratic piece, or (ii) x_e equals a breakpoint between two quadratic pieces, or (iii) x_e equals a breakpoint between a quadratic piece and a linear piece with arbitrarily large slope representing a lower or upper bound; otherwise e is a *linear element* of x. Define $\Delta = 1/A^m$ and re-define α to be the minimum of $2a_e^i$ over quadratic pieces.

Now run the algorithm in "phase 1" until $\langle x, h^* \rangle < \Delta^2/2$, so we can use continued fractions to round each x_e to a unique x'_e with denominator at most A^m. The next lemma will show that whenever e is a quadratic element of x', then we can set $x_e^* = x'_e$ without loss of optimality. Furthermore, it will show that when e is a linear element of x', we can constrain x_e^* to be in one of the at most two linear pieces adjacent to x'_e, and so can complete x^* by solving a piecewise linear "phase 2".

Lemma 6.3 *If element e is quadratic for x'_e, then $x'_e = x_e^*$. If x'_e is interior to a linear piece f_e^i, then every optimal x_e^* is in f_e^i's piece. If x'_e equals the breakpoint p_e^i between linear pieces f_e^{i-1} and f_e^i, then every optimal x_e^* is in f_e^{i-1}'s piece or f_e^i's piece. If x'_e equals the breakpoint p_e^i between a linear piece and a quadratic piece, then every optimal x_e^* is in the linear piece.*

Proof: Suppose that e is quadratic for x'_e, and let x_e^* and h_e^* come from optimal solutions with denominators at most A^m as in Lemma 6.1. If $x_e^* > x'_e$ (the reverse case is similar), then $x_e^* - x'_e \geq \Delta^2$. Since $|x'_e - x_e| < \Delta^2/2$, we get $x_e^* - x_e \geq \Delta^2/2$. If x'_e is interior to a quadratic piece, let f_e^i be that quadratic piece, else x'_e equals a breakpoint, say p_e^{i-1}, and we may assume that f_e^i is quadratic. Then $h_e^* \geq 2a_e^i x_e^* + b_e^i$. But now, since $c_e^+(x_e) \leq \max\{2a_e^i x_e + b_e^i, 2a_e^i x'_e + b_e^i\}$, we have $\Delta^2/2 > \langle x_e, h_e^* \rangle \geq 2a_e^i(\Delta^2/2)$, a contradiction. Thus we must have $x'_e = x_e^*$, so that e is quadratic for this x_e^*, and so for all optimal solutions.

If x'_e is interior to linear piece f_e^i with slope b_e^i, then x_e must also be interior to f_e^i. Thus $\langle x_e, h_e^* \rangle < \Delta^2/2$ implies that

$$|b_e^i - h_e^*| < \Delta^2/2. \tag{24}$$

If there is an optimal x^* with corresponding h^* where x_e^* does not belong to f_e^i, then by Lemma 6.1 we can assume that there is such a solution with denominators at most $1/\Delta$. If x_e^* is in the interior of a linear piece with slope $b_e^j = h_e^*$, then $|b_e^i - b_e^j| = |b_e^i - h_e^*| \geq 1$, contradicting (24). Otherwise, x_e^* is either interior to a quadratic piece or at a breakpoint non-adjacent to f_e^i. In either case, since x_e^* is outside of f_e^i's interval by at least $\Delta^2/2$, h_e^* must differ from b_e^i by more than $\Delta^2/2$, again contradicting (24). Thus x_e^* is in f_e^i's interval, forcing every optimal solution to use f_e^i's interval.

The remaining two cases have proofs much like the previous case. □

Now we have our algorithm: if e is quadratic for x', then we can fix $x_e^* = x'_e$. Otherwise we use one of the three linear cases in Lemma 6.3 to constrain x_e to lie either in a single linear piece or in the union of two adjacent linear pieces. Since the number of linear pieces B is only $O(m)$ in this problem, by Theorems 5.3 and 5.4 we can solve it in $O(m\psi^\perp \phi \log \phi)$ MMCM iterations, or $O(m\psi \log \phi)$ Cancel and Tighten iterations.

Theorem 6.4 *To find an exact optimal solution in the piecewise mixed linear and quadratic case, it suffices to perform $O(\psi^\perp \phi m \log(A\phi))$ MMCM iterations or $O(\psi m \log(A\phi))$ Cancel and Tighten iterations.* □

Note that many applications of piecewise mixed linear and quadratic costs do not need to solve the piecewise linear second phase. For example, if the only piecewise linear part is bounds (as in [6, 14]), then all elements are quadratic for every solution, so the optimal solution will be fully determined after the first phase. Alternatively, if there is only one linear piece per element (as in [22]), then the second phase can be solved more simply as just a linear feasibility problem with bounds over L.

7 Implementations for Networks

We now show how to specialize our generic methods MMCM and Cancel and Tighten to the particular cases of circulations and co-circulations in networks. Recall from the Example in Section 2 that both circulations and co-circulations are defined by totally unimodular matrices, so these truly are special cases. We assume that the digraph $G = (V, E)$ has n nodes and $m = |E|$ arcs, so that the maximum size of a circuit is $O(n)$ and the maximum size of a cut is $O(m)$. Note that an initial circulation satisfying any bounds implicit in the arc cost functions can be computed with one max flow, and similarly an initial co-circulation can be computed with one shortest path [5, 2]. Thus it is easy to initialize the algorithms and to verify that optimal solutions exist.

7.1 Network Circulation Algorithms

In this case our generic cycles become (simple) circuits, and co-cycles become (simple) cuts, so that ϕ and ψ are $O(n)$, and $\phi^\perp$ and $\psi^\perp$ are $O(m)$. Primal variables x are circulations, and dual variables h are co-circulations (differences of node potentials).

To implement MMCM for circulations, we need to solve (AP), which specializes in this case to finding a simple circuit C in the graph that minimizes

$$\left(\sum_{e \text{ forw. in } C} c_e^+(x_e) - \sum_{e \text{ back. in } C} c_e^-(x_e) \right) / |C|.$$

Existing min mean cycle algorithms find min mean *directed* cycles. To deal with this we define an auxiliary directed graph $\mathcal{D}(x)$ with arc costs w.r.t. a circulation x as follows: For each arc $e = (i, j)$ of G, add e and the *reverse arc* $\overline{e} = (j, i)$ to $\mathcal{D}(x)$. An arc $e \in E$ has cost $c_e = c_e^+(x_e)$ and $\overline{e}$ has cost $c_{\overline{e}} = -c_e^-(x_e)$. Then a min mean directed cycle in $\mathcal{D}(x)$ corresponds to a min mean simple circuit in L.

There exist many (strongly) polynomial algorithms for computing min mean directed cycles [16, 17, 18, 21, 25, 26, 30, 32]; the currently fastest algorithms run in $O(mn)$ time (Karp [16]), or $O(\sqrt{nm}\log(nC))$ time, where C is the size of the largest cost (Orlin and Ahuja [30], or McCormick [21]). All of these algorithms also compute a potential difference h proving the optimality of the min mean cycle.

Once we have a min mean cycle, we compute ε as in (14). The augmentation (10) of x around C is just an ordinary flow augmentation around a circuit. This is all we need to specialize MMCM to circulations, so we see that an iteration of MMCM is polynomial for circulations, and its running time (exclusive of calling the oracle) is dominated by computing a min mean cycle. This algorithm generalizes algorithms in [9, 6]. Theorem 2.5 specializes in this case to:

Theorem 7.1 *Let β be a bound on $\lambda(x)$ for the initial circulation, and let ε be the accuracy desired in our final answer. Then MMCM specialized to circulations uses $O(mn\log(\beta/\varepsilon))$ iterations.* □

We now consider how to implement Cancel and Tighten for circulations. Make an auxiliary graph containing the arcs in P^{far} and the reverses of the arcs in N^{far}, so that far cycles are directed cycles. For Cancel, a simple implementation is to use depth first search to find and cancel directed (far) cycles as suggested in Goldberg and Tarjan [9]. This takes $O(m)$ time for the search, and $O(n)$ time per cycle canceled, for a total of $O(mn)$ time.

To speed this up we adapt a trick from [9] for the classic linear circulation problem. They develop an algorithm based on dynamic trees for the problem of canceling all cycles consisting solely of negative reduced cost augmentable (admissible) arcs. The key of an arc in their method is the residual capacity of the arc in this direction. We need to choose instead the number ε_e defined in (20) for every arc e far from h. Once their method finds and cancels a cycle, it then cuts all arcs that are

no longer augmentable out of the data structure. We apparently need to cut out all arcs that have become close to h, but instead we relax our concept of Cancel a bit. Note that the proof of Claim 1 in Lemma 2.4 remains true even if each cycle contains elements satisfying only that $c_e^+(x_e) - h_e \leq 0$ and $h_e - c_e^-(x_e) \leq 0$ (non-positive reduced costs). Thus we can cut out only arcs whose reduced costs have become zero; this may cause some arcs that have become newly close to h to stay in the dynamic trees and so participate in some future cancellation, but this will not cause a violation of (18). Furthermore, the proof of Claim 2 in Lemma 2.4 shows that no new arcs far from h can be created. Since each cancellation cuts at least one arc far from h out of the dynamic trees, we still get the $O(m)$ bound on the number of cancellations. Thus, in the same way as in [9], this method takes only $O(m \log n)$ time.

When Cancel is done, the auxiliary graph is acyclic, so an acyclic node labeling l exists. The label differences $\Delta l_{ij} := l_j - l_i$ form a sufficient improving direction d for Tighten, and can be computed in linear time (see [9]). Thus one round of Cancel and Tighten costs $O(m \log n)$ time, which is a lot smaller than the time to compute even one min mean cycle. Theorem 4.5 specializes in this case to:

Theorem 7.2 *Let β be a bound on $\lambda(x)$ for the initial circulation, and let ε be the accuracy desired in our final answer. Then Cancel and Tighten specialized to circulations uses $O(n \log(\beta/\varepsilon))$ iterations, and $O(mn \log n \log(\beta/\varepsilon))$ time overall.* □

Now consider the piecewise linear case from Section 5. If we want to use Cancel and Tighten for circulations in networks in the piecewise linear case and we want to achieve the strongly polynomial bound of Theorem 5.4, we are required to substitute a $O(mn)$ [16] min mean cycle computation once every $O(n \log n)$ Tightens. Since $O(n \log)$ Tightens cost $O(mn \log^2 n) > O(mn)$ time, this does not affect our Cancel and Tighten time bound. In this case, Theorems 5.2, 5.3, and 5.4 become:

Theorem 7.3 *MMCM can find an exact optimal solution for circulations in networks with piecewise linear costs in $O(\min\{mn \log(nC), Bmn\})$ iterations. Cancel and Tighten (with an exact min mean cycle computation at the beginning of each big iteration) can find an exact optimal solution for circulations in networks with piecewise linear costs in $O(\min\{n \log(nC), Bn \log n\})$ iterations, and $O(m \log n \min\{n \log(nC), Bn \log n\})$ total time.* □

Note that in the classic linear case where $B = O(m)$ (there are two breakpoints per arc) these bounds match the bounds in Goldberg and Tarjan [9, Theorems 3.9 and 4.3] as improved by [33].

7.2 Network Co-circulation Algorithms

In this case our generic cycles become (simple) cuts, and co-cycles become (simple) circuits, so that ϕ and ψ are $O(m)$, and $\phi^\perp$ and $\psi^\perp$ are $O(n)$. Primal variables x are co-circulations, and dual variables h are circulations.

To implement MMCM for co-circulations, we need to solve (AP), which specializes in this case to finding a simple cut D in the graph that minimizes

$$\left(\sum_{e \text{ forw. in } D} c_e^+(x_e) - \sum_{e \text{ back. in } D} c_e^-(x_e) \right) / |\{\#\text{ arcs crossing } D\}|.$$

Most existing mean cut algorithms deal with networks having lower and upper bounds $l \leq u$, and find a cut D *maximizing*

$$\left(\sum_{e \text{ back. in } D} l_e(x_e) - \sum_{e \text{ forw. in } D} u_e(x_e) \right) / |\{\#\text{ arcs crossing } D\}|.$$

To deal with this we define an auxiliary directed graph $\mathcal{D}(x)$ with $l_e := c_e^-(x_e)$ and $u_e := c_e^+(x_e)$; we have $l \leq u$ by (6). Then a max mean cut in $\mathcal{D}(x)$ corresponds to (the reverse of) a min mean cut w.r.t. c^+ and c^-, which corresponds to a min mean cycle in L.

There exist many (strongly) polynomial algorithms for computing max mean cuts [15, 18, 21, 23, 31, 32, 36]; the currently fastest algorithms run in $O(\min\{n, \frac{\log(nU)}{1+\log\log(nU)-\log\log n}\}MC(m,n))$ time, where $MC(m,n)$ is the time to compute one min cut and U is the largest arc bound (McCormick and Ervolina [23], Radzik [31, 32], or Rote [36]), or $O(mn\log(nU))$ time (Iwano, et al [15], McCormick [21], or Radzik [31, 32]). All of these algorithms also compute a circulation h proving the optimality of the max mean cut.

Once we have a max mean cut D, we compute ε as in (14). The augmentation (10) of x around D is just an ordinary change of potentials across a cut. This is all we need to specialize MMCM to co-circulations, so we see that an iteration of MMCM is polynomial for co-circulations, and its running time (exclusive of calling the oracle) is dominated by computing a min mean cut. Theorem 2.5 specializes in this case to:

Theorem 7.4 *Let β be a bound on $\lambda(x)$ for the initial co-circulation, and let ε be the accuracy desired in our final answer. Then MMCM specialized to co-circulations uses $O(mn\log(\beta/\varepsilon))$ iterations.* □

We now consider how to implement Cancel and Tighten for co-circulations. Make an auxiliary graph containing the arcs in P^{far}, the reverses of the arcs in N^{far}, and both the forward and reverses of arcs close to h, so that far cycles are directed cuts. For Cancel, a simple implementation is to designate an arbitrary node r as a root. Do a directed search in the auxiliary graph for nodes having a directed path from r. This search is blocked by a directed cut pointing towards r, which we can cancel in $O(m)$ time. This gives us at least one new arc close to h crossing the cut with which we can continue the search. Continue like this until no directed cuts pointing towards r remain. Then repeat the search, but this time looking for nodes with a directed path to r, and being blocked by directed cuts pointing away from r. After the first search there is a directed path from r to every other node in the auxiliary graph, and after the second there is a directed path from every node to r. Thus the auxiliary graph is strongly connected, so no more directed (far) cuts remain. Each search costs $O(m)$ time total. Since each cut canceled allows at least one new node to be added to the search, we cancel only $O(n)$ cuts per search. Thus the two searches take $O(mn)$ total time.

To speed this up we adapt a trick from [3] for the classic linear co-circulation problem. They develop an algorithm based on solving two shortest path problems with non-negative distances to simultaneously cancel enough directed cuts in the auxiliary graph to be able to apply Tighten. We define arc lengths l_e to be 0 if e is close to h, and to be the ε_e from (20) if e is far from h. Now $l \geq 0$, so we can use a Dijkstra-type shortest path algorithm to find optimal shortest path distances l^* from r to every other node. Update x to $x + \Delta l^*$; this has the effect of canceling all cuts corresponding to level sets of l^*. As in the previous subsection, this can effectively result in canceling some cuts containing arcs that have become close to h, but as before it does not matter. Now re-form the auxiliary graph and find shortest path distances l^* from every node to r and update x to $x - \Delta l^*$. This can all be done in $O(m + n\log n)$ time.

When Cancel is done, the auxiliary graph contains a tree directed out from r (namely the first shortest paths tree) and a tree directed into r (the second shortest paths tree), so it is strongly connected. Thus we can cover the auxiliary graph with cycles such that each arc in E^{far} gets covered at least once in the proper orientation, and such a circuit cover can be computed in linear time by pushing excesses on the trees (see [3]). The sum of the incidence vectors of these circuits is a circulation that is a sufficient improving direction d for Tighten. Thus one round of Cancel and Tighten costs $O(m + n\log n)$ time, which is a lot smaller than the time to compute even one min mean cut. Theorem 4.5 specializes in this case to:

Theorem 7.5 *Let β be a bound on $\lambda(x)$ for the initial co-circulation, and let ε be the accuracy desired in our final answer. Then Cancel and Tighten specialized to co-circulations uses $O(m \log(\beta/\varepsilon))$ iterations, and $O(m(m + n \log n) \log(\beta/\varepsilon))$ time overall.* □

Now consider the piecewise linear case from Section 5. If we want to use Cancel and Tighten for co-circulations in networks in the piecewise linear case and we want to achieve the strongly polynomial bound of Theorem 5.4, we are required to substitute a $O(nMC(m,n))$ [23, 32] min mean cut computation once every $O(m \log n)$ Tightens. However, $O(m \log n)$ Tightens cost $O(m \log n(m + n \log n)) < O(nMC(m,n))$ time, which would affect our Cancel and Tighten time bound. However, an idea of Radzik quoted in [3] allows the same strongly polynomial bound to hold even with a weaker *semi-exact* min mean cut computation that costs only $O(MC(m,n))$ time. This can now be amortized without loss of efficiency; see [3] for details. In this case, Theorems 5.2, 5.3, and 5.4 become:

Theorem 7.6 *MMCM can find an exact optimal solution for co-circulations in networks with piecewise linear costs in $O(\min\{mn \log(nC), Bmn\})$ iterations. Cancel and Tighten (with a semi-exact min mean cut computation at the beginning of each big iteration) can find an exact optimal solution for circulations in networks with piecewise linear costs in $O(\min\{m \log(nC), Bm \log n\})$ iterations, and $O((m + n \log n) \min\{m \log(nC), Bm \log n\})$ total time.* □

Note that in the classic linear case where $B = O(m)$ (there is one breakpoint per arc) these bounds match the bounds in Ervolina and McCormick [3, Theorems 5.13 and 6.10] as improved by [33].

8 How to Solve the Auxiliary Problem

To make the description of our methods more complete we now explain how to solve the auxiliary problem (AP) in Section 2 in strongly polynomial time for an arbitrary unimodular linear space L. We assume that L is explicitly given via system $Mx = 0$ with a totally unimodular $n \times m$ matrix M whose columns are indexed by elements of a set E. An easy polynomial algorithm results if we use standard techniques to reduce (AP) to a fractional linear program over L. Then we use binary search to solve the fractional linear program by solving a (weakly) polynomial number of linear programs over L. Lastly, the linear programs over L can be solved in strongly polynomial time using, e.g., the modified version [39] of the ellipsoid method [19] (see also [10]).

However, this sort of binary search tends to be impractical since its worst case running time is also its average case. An algorithm that is more likely to be fast in practice derives from a general method in [18] that enables us to reduce (AP) to a number of linear programs that is polynomial in n and m. This method will also allow us to solve (AP) for systems which are not totally unimodular. We need some terminology and notation.

Suppose that L' is a (not necessarily unimodular) linear subspace of $\mathbf{R}^{E'}$ given by a system $M'y = 0$ with an integer $n' \times m'$ matrix M' ($m' = |E'|$). Consider the cone $\mathcal{K} = L' \cap \mathbf{R}^{E'}_+ = \{y \in L' : y \geq 0\}$. An integer vector $y \in \mathcal{K} - \{0\}$ is called *elementary* if $y = y' + y''$ is impossible for any two integral vectors $y', y'' \in \mathcal{K} - \{0\}$. Such an y belongs to an extreme ray of $\mathcal{K}$ and, conversely, each extreme ray of $\mathcal{K}$ contains an elementary vector. We say that the maximum $q = q(\mathcal{K})$ of y_e's among all elementary vectors y and $e \in E'$ is the *fractionality index*, or, briefly, the *fractionality* of $\mathcal{K}$ (clearly q does not exceed the maximum absolute value of a subdeterminant of M'). Note that if M' is totally unimodular then $q = 1$ and the set of elementary vectors is just the set of nonnegative cycles ξ of L' (i.e., $\xi_e \in \{0, 1\}$ for any $e \in E'$). Problem (AP) for L with $q > 1$ will be used in the next section.

Next, let $|y|$ denote the ℓ_1-norm $\sum_{e\in E'}|y_e|$. Then, given a vector $d \in \mathbf{R}^{E'} - \{0\}$, the *minimum mean problem* (MMP) for $\mathcal{K}, d$ is to find $y \in \mathcal{K} - \{0\}$ minimizing the objective

$$\nu_d(y) = dy/|y|.$$

Clearly, $\nu_d(ry) = \nu_d(y)$ for any positive real r, and $\nu_d(y+y') \geq \min\{\nu_d(y), \nu_d(y')\}$ for any $y, y' \in \mathbf{R}^{E'}_+ - \{0\}$. This implies that if y is an optimal solution to MMP and $y = \lambda_1 y^1 + \ldots + \lambda_k y^k$ for some $y^1, \ldots, y^k \in \mathcal{K}$ and $\lambda_1, \ldots, \lambda_k \in \mathbf{R}_+ - \{0\}$ then each y^i is an optimal solution as well. In particular, there is an optimal solution that is an elementary vector.

We now define the extension L' of the space L used in (AP) as follows. For each $e \in E$ add the "opposite" element $\overline{e}$, forming E' (so $|E'| = 2|E|$). For each column $M_{\bullet e}$ of M, add the column $-M_{\bullet e}$ to correspond to $\overline{e}$, forming M'. The resulting matrix M' defines the desired L' to be the solution set of system $M'y = 0$ ($y \in \mathbf{R}^{E'}$). Clearly M' is totally unimodular if and only if M is totally unimodular. Moreover, there is almost one-to-one correspondence between the cycles of L and the elementary vectors of $\mathcal{K} = L' \cap \mathbf{R}^{E'}_+$. More precisely, if ξ is a cycle of L then $\xi' \in \mathbf{R}^{E'}$, defined by $\xi'_e = \max\{0, \xi_e\}$ and $\xi'_{\overline{e}} = \max\{0, -\xi_e\}$ for $e \in E$, is an elementary vector of $\mathcal{K}$, and every elementary vector ξ' of $\mathcal{K}$ which is not obtained in this way satisfies $\xi'_e = \xi'_{\overline{e}} = 1$ and $\xi'_{e'} = 0$ ($e' \in E' - \{e, \overline{e}\}$) for some $e \in E$; call the latter vector ξ' *trivial*.

Next, for $x \in L$ and $e \in E$ put $d_e = c^+_e(x_e)$ and $d_{\overline{e}} = -c^-_e(x_e)$. Let $\nu^* = \nu^*_d$ denote the minimum $\nu_d(y)$ over $y \in \mathcal{K} - \{\emptyset\}$. The above arguments show that if $\lambda(x) = 0$ then $\nu^* \geq 0$ and if $\lambda(x) > 0$ then $\nu^* = -\lambda(x)$ (taking into account that $\nu_d(\xi') = d\xi'/|\xi'| = (c^+_e(x_e) - c^-_e(x_e))/2 \geq 0$ for any trivial elementary vector $\xi' \in \mathcal{K}$ corresponding to $\{e, \overline{e}\}$, $e \in E$). Thus, at least the part of (AP) concerning determining the number $\lambda(x)$ is reduced to MMP for $\mathcal{K}, d$. To make the present paper self-contained, we now describe the efficient algorithm to solve MMP for general systems given in [18].

For $y \in \mathbf{R}^{E'}$ let $S(y) = \mathrm{supp}(y)$ and let $S^+(y)$, $S^-(y)$ and $S^0(y)$ denote the sets of $e \in E'$ with y_e positive, negative and zero, respectively. W.l.o.g. one may assume that $\mathcal{K}$ spans E' (i.e., $\cup(S(y) : y \in \mathcal{K}) = E'$), else we could decrease E', not affecting our problem. Each iteration of the algorithm transforms the current objective vector d by adding to it a vector $h \in (L')^\perp$ and, possibly, by shifting d by a constant $\delta \in \mathbf{R}$ (i.e., by setting $d_e := d_e + \delta$ for all $e \in E'$). Clearly the former operation does not change $\nu_d(y)$ for any $y \in \mathcal{K}$, while the latter one increases it by δ (and therefore increases ν^* to $\nu^* + \delta$). At the beginning of the algorithm we shift the initial d by a sufficiently large negative r in order to ensure that the inequality $\nu^*_d \leq 0$ holds. Throughout the algorithm $\nu^*_d \leq 0$ is maintained.

For a current d let d^+ denote the positive part of d, i.e., $d^+_e = \max\{0, d_e\}$. At an iteration, an element $u \in E'$ with $d_u < 0$ is fixed, and we solve the following pair of dual linear programs:

$$\min\{d^+y : y \in \mathcal{K},\ y_u = 1\} \tag{25}$$

and

$$\max\{h_u : h \in (L')^\perp,\ h_e \geq -d^+_e \text{ for } e \in E' - \{u\}\}. \tag{26}$$

To see that (25) and (26) are dual LPs, re-write (25) as $\min\{d^+y : M'y = 0,\ y \geq 0,\ y_u = 1\}$, let the dual vector for the constraints $M'y = 0$ be γ, and the dual scalar for $y_u = 1$ be σ. Then if we define $h = -\gamma M'$, since the rows of M' are a basis of $(L')^\perp$ we must have $h \in (L')^\perp$, and it is clear (since $d^+_u = 0$) that $h_u = \sigma$ at optimality in the dual of (25).

Since $\mathcal{K}$ spans E', there is a cycle containing u whose incidence vector is feasible to (25). Since $h = 0$ is feasible to (26), (25) has an optimal solution. When feasible solutions y and h to (25) and (26) are optimal then the following (complementary slackness condition) holds:

$$\text{for any } e \in E' - \{u\},\quad d^+_e + h_e > 0 \text{ implies } y_e = 0. \tag{27}$$

Let y and h be optimal, and define $\widehat{d} = d + h$. If $\widehat{d}_u \geq 0$, we finish the iteration with the new objective vector $d' = \widehat{d}$. If instead $\widehat{d}_u < 0$, we in addition shift $\widehat{d}$ by $\delta := -\widehat{d}y/|y|$, and the resulting vector d' becomes the new objective vector. In the latter case we also keep y as a possible candidate for being an optimal solution of the original (and each intermediate) problem; put $y^* := y$. Then $d'y^* = 0$.

From (27) we observe that

$$\text{for } e \in S(y) - \{u\}, \quad \widehat{d}_e = \begin{cases} 0 & \text{if } d_e \geq 0 \\ d_e & \text{if } d_e < 0. \end{cases} \tag{28}$$

Since y is nonnegative, (28) implies

$$\widehat{d}y \leq \widehat{d}_u y_u = \widehat{d}_u, \tag{29}$$

hence $\delta > 0$ (in case $\widehat{d}_u < 0$). Note also that (28) together with $h_u = d^+y \geq 0$ (as $d^+ \geq 0$ and $y \geq 0$) and $\delta > 0$ preserves the following monotone property (in both cases):

$$S^-(d') \subseteq S^-(d), \quad \text{and } d'_e \geq d_e \text{ for any } e \in S^-(d'). \tag{30}$$

The algorithm finishes if the iteration results in d' with $d'_e \geq 0$ for all $e \in E'$, i.e., $S^-(d') = \emptyset$. Then for the y^* obtained at the last iteration when a shift of d occurs, we have $d'y^* \geq 0$. On the other hand, after that iteration the current d was transformed only by adding vectors from $L^\perp$, which does not change the inner product of d and y^*. Hence, $d'y^* = 0$ (so $d'_e = 0$ for all $e \in S(y^*)$). This and $\nu^*_{d'} \geq 0$ (as $d' \geq 0$) yield that y^* is an optimal solution to the original MMP.

Let us say that the iteration is *big* if $S^-(d')$ is strictly contained in $S^-(d)$, and *small* otherwise. The number of big iterations does not exceed $N = |S^-(d^0)|$ for the d^0 obtained by the initial shift of the original objective vector. So we have to estimate the number of consecutive small iterations. We impose the additional rule that the same element u is fixed throughout consecutive small iterations. Suppose that the current and the next iterations are small. Let y', h', d' be the corresponding objects found at the next iteration. Note that by the definition of small iterations, $S^-(d) = S^-(d')$, so that $Q := Q(d) := S^+(d) \cup S^0(d)$ is independent of d during this sequence of small iterations.

Lemma 8.1 $y(Q) > y'(Q)$.

Proof: Obviously, $d'_e \geq \delta$ for each $e \in Q$, and (28) implies $d'_e = \delta$ for $e \in S(y) \cap Q$. This and $d'y = 0$ give

$$d'_u = d'_u y_u \geq \sum(d'_e y_e \,:\, e \in S^-(d)) = -\sum(d'_e y_e \,:\, e \in Q) = -\delta y(Q). \tag{31}$$

On the other hand, we have $\widehat{d}'_u = d'_u + h'_u < 0$ (as the second iteration is small), so

$$-d'_u > h'_u = (d')^+ y' \geq \delta y'(Q) \tag{32}$$

(using LP strong duality for the second iteration and the fact that $d'_e \geq \delta$ for $e \in Q$). Comparing (31) and (32) yields $\delta y(Q) > \delta y'(Q)$, and the result follows because $\delta > 0$. □

Note that the preliminary shift of the original d can be chosen so that the above number N is at most ψ (the rank of M' and M). Indeed, let $p_1 < p_2 < \ldots < p_k$ be all different values of d_e, $e \in E'$, and let $X_i = \{e \in E \,:\, d_e \leq p_i\}$. Solving corresponding LP problems, find the minimum i such that X_i contains the support of some nonzero vector of $\mathcal{K}$. Shift d by $-p_i$. This makes the elements of X_{i-1} ($X_i - X_{i-1}$) be of negative (resp. zero) weight, and therefore there is $y \in \mathcal{K} - \{0\}$ with $\nu_d(y) \leq 0$. Clearly we have $N = |X_{i-1}| \leq \psi$.

Theorem 8.2 *The number of iterations of the algorithm is at most* $\psi \min\{q^2\phi, \eta\}$, *where* ϕ *is the maximum support cardinality of an elementary vector, and* η *is the number of elementary vectors in* K. *In particular, if* q *is bounded by a polynomial in* $|E'|$ *and a strongly polynomial algorithm is used to solve (25) and (26), then this algorithm is strongly polynomial for MMP.*

Proof: Consider two consecutive small iterations as in Lemma 8.1. As we have remarked, there must be extreme optimal solutions y and y' to (25) in these iterations. By Lemma 8.1, $y(Q) > y'(Q)$, so the number of consecutive small iterations does not exceed the number of distinct values of $z(Q)$ for extreme vectors z of K with $z_u = 1$. But if z is an integral extreme vector of K, then $z(Q)$ is an integer between 1 and $q\phi$, and there are at most q ways to normalize z so that $z_u = 1$. Thus there can be at most $\min\{q^2\phi, \eta\}$ values for $x(Q)$. Now the result follows from the fact that the number of big iterations is at most ψ. □

For the original function d and the final function d^*, we have $d^* = d + h^* + \delta^* \mathbb{1}_{E'}$, where δ^* is the total shift and h^* is the sum of solutions to (26) over the iterations. Since d^* is nonnegative and $d^*_e = 0$ for any $e \in S(y^*)$,

$$d_e + h^*_e \begin{cases} = -\delta^* & \text{for } e \in S(y^*), \\ \geq -\delta^* & \text{otherwise.} \end{cases} \tag{33}$$

In particular, $\nu^*_d = -\delta^*$. (Note also that (33) proves the equality $\nu^*_d = \max_{h \in (L')^\perp} \min_{e \in E'}(d_e + h_e)$.)

We now show how this algorithm yields the cycle F and dual vector h that (AP) asks for. We know that if $\nu^*_d \geq 0$ then $\lambda(x) = 0$, and if $\nu^*_d < 0$ then $\lambda(x) = -\nu^*_d = \delta^*$. Consider the case $\nu^*_d < 0$. The definition of M' shows that $h^*_e = -h^*_{\bar{e}}$ for each $e \in E$ (as h^* is a linear combination of row vectors of M'). Put $h_e = -h^*_e$, $e \in E$. Then h is a linear combination of row vectors of M, whence $h \in L^\perp$. Furthermore, it is easy to see that (33) implies (11).

To find the desired cycle, we first observe that $y^*_e y^*_{\bar{e}} = 0$ for any $e \in E$. For if both y^*_e and $y^*_{\bar{e}}$ are positive then decreasing these by $\varepsilon = \min\{y^*_e, y^*_{\bar{e}}\}$ makes a vector $y \in L'$ with $|y| < |y^*|$ and $dy^* - dy = (c^+_e(x_e) - c^-_e(x_e))\varepsilon \geq 0$. But then $\nu_d(y) < \nu_d(y^*)$ (in view of $\nu_d(y^*) = \nu^*_d < 0$); a contradiction.

For $e \in E$ define $z_e = y^*_e - y^*_{\bar{e}}$. Then $z \in L$ and the mean cost of z is $-\lambda(x)$, so that z is a "minimum mean circulation" at x. Now standard linear algebra techniques enable us to identify a cycle ξ whose support is a subset of $\mathrm{supp}(z)$ that conforms to z (in the sense of (8)). Then ξ must be a min mean cycle. (Note that if M is totally unimodular and y^* is a basic optimal solution to (25) then z is already a cycle, but for generality we do not require this of y^*.)

9 The Methods for Linear Spaces of Bigger Fractionality

The method developed in Section 2 is easily generalized to an arbitrary linear space $L \subseteq \mathbf{R}^E$ given by system $Mx = 0$ with an integer $n \times |E|$-matrix. We briefly describe how to do this, emphasizing only main differences in the statements and algorithm compared with those in Section 2, and leaving details to the reader.

Like the notion of elementary vector in a cone from the previous section, by a *cycle* of L we mean an integer non-zero vector $x \in L$ such that any integer vector $x' \in L$ with $\mathrm{supp}(x') \subseteq \mathrm{supp}(x)$ satisfies $x' = \alpha x$ for some integer α. The *fractionality* $q = q(L)$ of L is the maximum $|\xi_e|$ among all cycles ξ and $e \in E$. For $x \in L$ let $S^+(x) = \{e \in E : x_e > 0\}$ and $S^-(x) = \{x \in E : x_e < 0\}$. Then the cost at x of a cycle ξ is defined as

$$c(x, \xi) = \sum(c^+_e(x_e)\xi_e : e \in S^+(\xi)) + \sum(c^-_e(x_e)\xi_e : e \in S^-(\xi)),$$

and its mean cost $\overline{c}(x, \xi)$ is $c(x, \xi)/|\xi|$. Then ξ is a minimum mean cycle at x if $\overline{c}(x, \xi)$ is as small as possible, and $\lambda(x)$ is defined to be $\max\{0, -\overline{c}(x, \xi)\}$, where ξ is a minimum mean cycle.

One can check that Lemmas 2.1 and 2.2 remain true in this case. MMCM is refined in a natural way, as follows. For the current $x \in L$ we solve the analog of the auxiliary problem (AP) to compute $\lambda = \lambda(x)$, and if $\lambda > 0$, to find a minimum mean cycle ξ of L at x and $h \in L^\perp$ satisfying (11). If $\lambda > 0$, we "push some amount $\varepsilon > 0$ around ξ", i.e. transform x into the new current point $x' \in L$ defined as $x' = x + \varepsilon\xi$ (cf. (10)). To compute the step length ε, we first compute the local step lengths (cf. (14))

$$\varepsilon_e = \text{an increment } \Delta \text{ s.t.} \begin{cases} c_e^-(x_e + \Delta\xi_e) \le c_e^+(x_e) + \lambda = h_e \le c_e^+(x_e + \Delta\xi_e) & \text{if } e \in S^+(\xi); \\ c_e^-(x_e - \Delta\xi_e) \le c_e^-(x_e) - \lambda = h_e \le c_e^+(x_e - \Delta\xi_e) & \text{if } e \in S^-(\xi), \end{cases}$$

and then $\varepsilon = \min\{\varepsilon_e : e \in S^+(\xi) \cup S^-(\xi)\}$.

In this case Lemma 2.3 has essentially the same proof, while Lemma 2.4 and Theorem 2.5 should be modified as follows.

Lemma 9.1 *After every $\psi^\perp$ iterations $\lambda = \lambda(x)$ decreases by a factor of at least $(1 - 1/(2\phi q))$.*

Proof: Similar to that of Lemma 2.4. The only non-trivial point that needs some re-proof is when, at some iteration j $(i < j \le i + \psi^\perp)$, for the chosen cycle ξ^j there is an element $u \in \mathrm{supp}(\xi^j)$ close to h, and for all $e \in S^+(\xi^j) - \{u\}$ $(e \in S^-(\xi^j) - \{u\})$ one has $c_e^+ - h_e \ge -\lambda$ (resp. $h_e - c_e^- \ge -\lambda$); for simplicity, we replace $c_e^+(x_e^j)$ and $c_e^-(x_e^j)$ by c_e^+ and c_e^- here. For definiteness let u be positively close to h. Then

$$\begin{aligned} c(x^j, \xi^j) &= c(x^j, \xi^j) - h\xi^j \\ &= \sum_{e \in S^+(\xi^j) - \{u\}} (c_e^+ - h_e)\xi_e^j + (c_e^+ - h_u)\xi_u^j - \sum_{e \in S^-(\xi^j)} (h_e - c_e^-)\xi_e^j \\ &\ge -\lambda \sum_{e \in S^+(\xi^j) - \{u\}} \xi_e^j - \frac{1}{2}\lambda\xi_u^j + \lambda \sum_{e \in S^-(\xi^j)} \xi_e^j \\ &= -\lambda|\xi^j| + \frac{1}{2}\lambda\xi_u^j \ge -\lambda|\xi^j| + \lambda/2, \end{aligned}$$

whence $-\lambda^j = \overline{c}(x, \xi^j) \ge -\lambda + \lambda/2|\xi^j| \ge -\lambda(1 - 1/2\phi q)$ (as $|\xi^j| \le \phi q$), and the result follows. □

Theorem 9.2 *Let β be a bound on $\lambda(x)$ for the initial $x \in L$, and let ε be the absolute accuracy desired in our final answer. Then the algorithm uses $O(\phi\psi^\perp q \log(\beta/\varepsilon))$ iterations.* □

Thus, compared with the unimodular case, the convergence of $\lambda(x)$ can slow down by a factor of q. Also, as we have seen in the previous section (Theorem 8.2), solving (AP) at each iteration can take $O(q^2)$ more time. Nevertheless, if q is "small enough" (e.g. bounded by a polynomial in $|E|$), the whole algorithm is still effective. For example, for the fractional b-matching problem mentioned in the Introduction, it is easy to see that all elementary vectors have entries $(0, \pm 1, \pm 2)$. Thus $q = 2$ for this problem, so our methods are still polynomial. The same is true for the second example in the Introduction, which has $q \le 2^k$ for some fixed k.

Note also that if q is bounded by a polynomial in $|E|$, then the method remains strongly polynomial in the piecewise linear case. This follows from arguments in Section 5 and the observation that $\lambda(x) \le \langle x, h^* \rangle \le \phi q \lambda(x)$ (the second inequality comes from the corresponding modification of Corollary 3.2). Moreover, compared with Theorem 5.3, to compute an exact optimal solution now takes $O(B\psi^\perp \phi q)$ iterations.

We leave it to the reader to examine other special cases of w for spaces L with $q(L) > 1$ and modify the corresponding results in Sections 4–6.

10 Solving Integer Problems

For some applications it is more realistic to consider problems where the decision variables are restricted to be integers. This version of our problem has received a lot of attention; see, e.g., Minoux [28] or Ahuja, et al [2, Chapter 14]. We show here that our methods adapt to this new restriction without any loss of efficiency.

Formally, we want to solve

(34) Find an integral $x \in L$ such that $w(x) = \sum_{e \in E} w_e(x_e)$ is as small as possible.

With the integer restriction, there is no difference in (34) if we replace all w_e's with their piecewise linear approximations, where we put breakpoints at all integer values for x_e. Of course, we do not want to do this explicitly as there could be an unbounded number of linear pieces to even write down. Instead, we will use our oracle for w_e to compactly represent this approximation.

To reflect the discreteness, we replace the continuous right cost by $c_e^+(x_e) := w_e(x_e+1) - w_e(x_e)$, and the continuous left cost by $c_e^-(x_e) := w_e(x_e) - w_e(x_e - 1)$. The only other change we need to make is in using our oracle to choose a step length in (14). We use our continuous oracle to choose an ε_e as before, but now we need to round ε_e to either $\lfloor \varepsilon_e \rfloor$ or $\lceil \varepsilon_e \rceil$, depending on which is the correct answer for the discretized costs. This allows us to simulate an oracle for the piecewise linear approximation to w_e using only $O(1)$ computation with the continuous oracle. Note that since this makes all ε_e integral, ε will also be integral, and the algorithm will keep an integral circulation at all times. Otherwise we run our algorithm as before.

This modified algorithm simulates the performance of the original algorithm on the piecewise linear approximation. Thus Lemma 5.1 and so also Theorems 5.2, 7.3, and 7.6 apply to this situation. (We ignore the bounds involving B in Theorems 7.3 and 7.6 since the number of breakpoints could be infinite in the approximated functions.) Therefore, the modified algorithm has the following performance:

Theorem 10.1 *Modified MMCM exactly solves (34) in $O(\psi^\perp \phi \log(\phi C))$ iterations, and modified Cancel and Tighten solves (34) in $O(\psi \log(\phi C))$ iterations. When specialized to circulations in a network, Cancel and Tighten exactly solves (34) in $O(mn \log n \log(nC))$ total time. When specialized to co-circulations in a network, Cancel and Tighten exactly solves (34) in $O(m^2 \log(nC) + mn \log n \log(nC))$ total time.* □

Note that our bound of $O(mn \log n \log(nC))$ total time for Cancel and Tighten for circulations on networks compares favorably with the bound of $O(m \log U(m + n \log n))$ for the algorithm for the same problem in [2, Chapter 14]. Under the widely-used *Similarity Assumption* that C and U are $O(n^{O(1)})$ [8], our algorithm is $O(mn \log^2 n)$ while the algorithm in [2] is $O(m(m + n \log n) \log n)$, so in most practical cases, our bound is faster.

11 The Methods for Problems with Approximate Oracles

When the element cost functions w_e are (piecewise) linear and/or quadratic with rational coefficients, it is reasonable to assume that the oracle (O) gives exact answers. However, if some w_e's are more general, then it may not even be possible to exactly represent c_e^+ and c_e^- in a finite word length, leading to an *evaluation error* δ_e. Similarly, an exact solution to (O) (ii) may be transcendental as well, and/or it might not be easy to solve the equation $w_e'(r) = q$ exactly (this is roughly what (O) (ii) requires). Then (O) (ii) might be implemented as a numerical root-finding procedure with some known error, leading to a *root-finding error* δ_r. We now start using the convention that symbols for approximate quantities are the same symbols as the exact quantities with tildes added. More formally, we now assume that we have an oracle that

($\tilde{\text{O}}$) (i) when given a "point" $r \in \mathbf{R}$, will return costs $\widetilde{c}_e^+(r)$ and $\widetilde{c}_e^-(r)$ satisfying $|\widetilde{c}_e^+(r) - c_e^+(r)| \le \delta_e$ and $|\widetilde{c}_e^-(r) - c_e^-(r)| \le \delta_e$;

(ii) when given a "scalar" $q \in \mathbf{R}$, will return a point $\widetilde{r}$ with $c_e^-(\widetilde{r}) - \delta_r \le q \le c_e^+(\widetilde{r}) + \delta_r$ (so $\widetilde{c}_e^-(\widetilde{r}) - \delta_e - \delta_r \le q \le \widetilde{c}_e^+(\widetilde{r}) + \delta_e + \delta_r$).

The first consequence of these errors is that $\widetilde{\lambda}(x)$, the approximate min mean cycle value that we compute w.r.t. $\widetilde{c}_e^+$ and $\widetilde{c}_e^-$, will not exactly equal $\lambda(x)$. However, since $\widetilde{c}_e^+$ and $\widetilde{c}_e^-$ differ from c_e^+ and c_e^- by at most δ_e per element, it is easy to see that $\widetilde{\lambda}(x)$ satisfies

$$|\lambda(x) - \widetilde{\lambda}(x)| \le \delta_e. \tag{35}$$

A quick scan of Lemmas 2.3 and 2.4 (the key lemmas proving polynomial convergence) shows that our arguments there all have a slack of at least $\lambda/2$ in them, except for the argument in Claim 2 of Lemma 2.4 that no new element far from h can be created. We want to claim that whenever $\widetilde{\lambda}(x)$ is large enough w.r.t. δ_e and δ_r, the arguments in those lemmas will still hold true with the approximate data $\widetilde{c}_e^+$, $\widetilde{c}_e^-$, and $\widetilde{\lambda}(x)$ substituted for c_e^+, c_e^-, and $\lambda(x)$. The argument in Claim 2 depends on knowing that for some e (superscripting by iteration number)

$$\begin{aligned} \widetilde{c}_e^+(x_e^j) + \widetilde{\lambda}^j &\le \widetilde{c}_e^+(x_e^{j+1}) \quad \text{if } e \in A^j, \text{ and} \\ \widetilde{c}_e^-(x_e^j) - \widetilde{\lambda}^j &\ge \widetilde{c}_e^-(x_e^{j+1}) \quad \text{if } e \in B^j, \end{aligned} \tag{36}$$

but ($\tilde{\text{O}}$) (ii) might cause these inequalities to be violated by as much as $\delta := \delta_e + \delta_r$. We can fix this, and obtain an easy analysis of the approximate oracle case, by slightly increasing our local step length from (14) to now be

$$\varepsilon_e = \text{an increment } \Delta \text{ s.t.} \begin{cases} \widetilde{c}_e^-(x_e + \Delta) - \delta \le \widetilde{c}_e^+(x_e) + \widetilde{\lambda} + \delta \le \widetilde{c}_e^+(x_e + \Delta) + \delta & \text{if } e \in A; \\ \widetilde{c}_e^-(x_e - \Delta) - \delta \le \widetilde{c}_e^-(x_e) - \widetilde{\lambda} - \delta \le \widetilde{c}_e^+(x_e - \Delta) + \delta & \text{if } e \in B. \end{cases} \tag{37}$$

That is, we call ($\tilde{\text{O}}$) (ii) with $q = \widetilde{c}_e^+(x_e) + \widetilde{\lambda} + \delta_e + \delta_r$ when $e \in A$, and $q = \widetilde{c}_e^-(x_e) - \widetilde{\lambda} - \delta_e - \delta_r$ when $e \in B$. Since $x_e^{j+1} = x_e^j + \Delta$ for an e achieving the minimum ε_e, this ensures that (36) will hold for this e.

There are two factors that limit how much we can reduce $\widetilde{\lambda}(x)$ before the inaccuracies catch up with us: (i) the maximum error in a c_e^+ or c_e^-, namely δ_e, can be at most the slack in the arguments in Lemmas 2.3 and 2.4, namely $\widetilde{\lambda}(x)/2$; and (ii) the increment in reduced cost, namely $\widetilde{\lambda}(x) + \delta_e + \delta_r$, plus the maximum error in c_e^+ and c_e^-, namely δ_e, must be smaller than $\widetilde{\lambda} + \widetilde{\lambda}/2$ to ensure that we don't make a positively far element negatively far or vice versa. Now (i) gives the bound $\widetilde{\lambda}(x) > 2\delta_e$, and (ii) gives $\widetilde{\lambda}(x) \ge 2\delta_r + 2\delta_e$. That is, we must have

$$\widetilde{\lambda}(x) \ge 2\delta_e + 2\delta_r. \tag{38}$$

Now, as long as $\widetilde{\lambda}(x)$ satisfies (38), and since we contrived (37) to make (36) true, the whole analysis of Lemmas 2.3 and 2.4 becomes true with $\widetilde{c}_e^+$, $\widetilde{c}_e^-$, and $\widetilde{\lambda}(x)$ in place of c_e^+, c_e^-, and $\lambda(x)$. In particular, this shows that $\widetilde{\lambda}(x)$ will geometrically decrease. Once the algorithm has driven $\widetilde{\lambda}(x)$ down to $2\delta_e + 2\delta_r$, (35) says that $\lambda(x) \le 3\delta_e + 2\delta_r$. This estimate can be used with the cost distance defined in Section 3 and the bounds based on it developed in Section 6 to get a bound on how close x is to an optimal solution.

These arguments suggest that when ($\tilde{\text{O}}$) (ii) is implemented as a root-finding procedure with an error estimate available at each iteration, then it will be faster to run the root-finder only until the current error in its root, δ_r', is small enough so that $\widetilde{\lambda}(x) - 2\delta_r' > 2\delta_e$.

The same sort of analysis can be pushed through for the Cancel and Tighten method. All of this shows that even with an approximate oracle, our methods can obtain an answer whose accuracy is close to the accuracy of the underlying evaluation and root-finding machinery.

12 Duality and Practical Implementations

Throughout our methods, we have liberally used "dual" vectors $h \in L^{\perp}$, and at optimality we have an "optimal" dual vector h^*. It turns out that h^* is not only optimal in the sense of proving x^*'s optimality via (13), but that h^* is the optimal solution of a related optimization problem dual to (5). The form of this problem is

$$\min_{h \in L^{\perp}} \sum_{e \in E} v_e(h_e), \tag{39}$$

where each v_e is the *convex conjugate* of the corresponding w_e. See, e.g., Rockafellar [35] (who would say that we have a dual pair of monotropic programming problems) for a full description of how w_e and v_e are related; we cover a few details here.

For example, suppose that w_e is piecewise linear. Then v_e is also piecewise linear, and the breakpoints of v_e are the slopes of w_e, and the slopes of v_e are the breakpoints of w_e. For example, the convex conjugate of the cost function for linear circulations in networks in Figure 1 is the function for linear co-circulations in networks in Figure 2. Similarly, w_e is piecewise mixed linear and quadratic if and only if v_e is piecewise mixed linear and quadratic. It can be more difficult to derive convex conjugates (and their associated oracles) for more complicated convex functions, but as we have seen, the piecewise linear and quadratic cases cover many of the applications for this model.

It can be shown from the theory of conjugate duality that the h^* that our methods compute is an optimal solution of (39) (h^*'s optimality is proved by x^*). Since we get this h^* for free when we solve the primal problem (5), if we want to compute x^* we have the luxury of being free to solve either the primal problem (5) and getting x^* directly, or solving the dual problem (39) and getting x^* as the optimal set of dual variables for (39). This free choice has some practical implications, because although the primal and dual problems are in theory completely symmetric, in practice it may be easier to solve one than the other.

Let us focus on solving circulation and co-circulation problems in networks, which is certainly the largest class of applications for our methods. If we implement MMCM, the number of iterations is $O(mn \log(\beta/\varepsilon))$ for both cases (Theorems 7.1 and 7.4). However, the time per iteration is dominated by the min mean cycle computation, which is, up to log factors, $O(mn)$ for circulations and $O(mn^2)$ for co-circulations using strongly polynomial algorithms, and $O(\sqrt{n}m)$ for circulations and $O(mn)$ for co-circulations using polynomial algorithms under the Similarity Assumption. Thus is is clearly better to use MMCM on the circulation version of the problem when we have a choice.

For the more efficient, and thus more practical Cancel and Tighten method, the work per iteration is the same up to log factors: $O(m)$ for both circulations and co-circulations. However, the number of iterations is $O(n \log(\beta/\varepsilon))$ for circulations but $O(m \log(\beta/\varepsilon))$ for co-circulations (Theorems 7.2 and 7.5). In the important piecewise linear case, the iteration bound for circulations is $O(\min\{n \log(nC), Bn \log n\})$, and for co-circulations is $O(\min\{m \log(nC), Bm \log n\})$ (Theorems 7.3 and 7.6). Thus in both cases solving the circulation version of the problem instead of the co-circulation version will result in a speed-up by a factor of roughly $O(m/n)$.

Thus we can conclude that (whenever it is not too painful to compute the convex conjugates of the objective functions, possibly at a pre-processing step), if one is interesting in finding an optimal co-circulation in a network it should be faster to dualize the problem and solve the circulation version of the problem instead.

References

[1] Ahlfeld, D.P., R.S Dembo, J.M. Mulvey, and S.A. Zenios (1987). Nonlinear Programming on Generalized Networks. *ACM Transactions on Mathematical Software*, **13**, 350–367.

[2] Ahuja, R.K., T.L. Magnanti, and J.B. Orlin (1993). *Network Flows: Theory, Algorithms and Applications.* Prentice Hall, New York, NY.

[3] Ervolina, T.R. and S.T. McCormick (1991). Two Strongly Polynomial Cut Cancelling Algorithms for Minimum Cost Network Flow. To appear in *Discrete Applied Mathematics.*

[4] Florian, M. (1986). Nonlinear Cost Network Models in Transportation Analysis. *Math. Prog. Study*, **26**, 167–196.

[5] Ford, L.R., Jr., and D.R. Fulkerson (1962). *Flows in Networks.* Princeton University Press, Princeton, NJ.

[6] Frank, A. and A.V. Karzanov (1992). Determining the Distance to the Perfect Matching Polytope of a Bipartite Graph. Technical Report RR 895-M, Laboratoire ARTEMIS IMAG, Université Joseph Fourier, Grenoble, France.

[7] Fujishige, S. (1984). Submodular Systems and Related Topics. *Math. Prog. Study*, **22**, 113–131.

[8] Gabow, H.N. (1985). Scaling Algorithms for Network Problems. *J. of Comput. Sys. Sci.* **35** pp. 148–168.

[9] Goldberg, A.V., and R.E. Tarjan (1989). Finding Minimum-Cost Circulations by Canceling Negative Cycles. *JACM* **36**, no. 4, 873–886.

[10] Grötschel, M., L. Lovász, and A. Schrijver (1988). *Geometric Algorithms and Combinatorial Optimization.* Springer Verlag.

[11] Hassin, R. (1983). The Minimum Cost Flow Problem: A Unifying Approach to Dual Algorithms and a New Tree-Search Algorithm, *Math. Prog.* **25**, 228–239.

[12] Hassin, R. (1993). A Flow Algorithm for Network Synchronization. Working Paper, Dept. of Statistics and Operations Research, Tel-Aviv University.

[13] Hochbaum, D.S. and J.G. Shanthikumar (1990). Convex Separable Optimization is Not Much Harder than Linear Optimization. *JACM*, **37**, 843–862.

[14] Ibaraki, T., A.V. Karzanov and H. Nagamochi (1993). Determining the Distance to the Flow Polyhedron of a Network. Private communication.

[15] Iwano, K., S. Misono, S. Tezuka, and S. Fujishige (1990). A New Scaling Algorithm for the Maximum Mean Cut Problem. IBM Research Report RT 0049, Tokyo, Japan; to appear in *Algorithmica.*

[16] Karp, R.M. (1978). A Characterization of the Minimum Cycle Mean in a Digraph. *Discrete Mathematics* **23**, 309–311.

[17] Karp, R.M. and J.B. Orlin (1981). Parametric Shortest Path Algorithms with an Application to Cyclic Staffing. *Discrete Applied Mathematics* **3**, 37–45.

[18] Karzanov, A.V. (1985). Minimal Mean Weight Cuts and Cycles in Directed Graphs. In: *Methods for Solving Operator Equations*, Yaroslavl State Univ., Yaroslavl, 72–83, in Russian. (English translation to appear in *Soviet Math. Translations*, series 2, Amer. Math. Soc. 1993.)

[19] Khachiyan, L.G. (1979). A Polynomial Algorithm in Linear Programming. *Doklady Akademii Nauk SSSR*, **244**, 1093–1096, in Russian. (English translation in *Soviet Mathematics Doklady*, **20** (1979), 191–194).

[20] Magnanti, T.L. (1984). Models and Algorithms for Predicting Urban Traffic Equilibria. In *Transportation Planning Models*, edited by M. Florian, North-Holland, Amsterdam, 153–186.

[21] McCormick, S.T. (1993). Approximate Binary Search Algorithms for Mean Cuts and Cycles. *OR Letters*, **14**, 129–132.

[22] McCormick, S.T. (1993). How to Compute Least Infeasible Flows. UBC Faculty of Commerce Working Paper 93-MSC-007, Vancouver, BC; an extended abstract appears in *NETFLOW93*, 160–164.

[23] McCormick, S. T. and T. R. Ervolina (1990). Computing Maximum Mean Cuts. UBC Faculty of Commerce Working Paper 90-MSC-011, Vancouver, BC; to appear in *Discrete Applied Mathematics*.

[24] McCormick, S. T. and L. Liu (1991). An Experimental Implementation of the Dual Cancel and Tighten Algorithm for Minimum Cost Network Flow. Extended Abstract accepted by the Dimacs Netflow Challenge Workshop, full paper to appear in the proceedings of the workshop.

[25] Megiddo, N. (1979). Combinatorial Optimization With Rational Objective Functions. *Mathematics of Operations Research* **4**; 414–424.

[26] Megiddo, N. (1983). Applying Parallel Computation Algorithms in the Design of Serial Algorithms. *JACM* **30**, 852–865.

[27] Minoux, M. (1984). A Polynomial Algorithm for Minimum Quadratic Cost Flow Problems. *European J. of OR*, **18**, 377–387.

[28] Minoux, M. (1986). Solving Integer Minimum cost Flows with Separable Convex Cost Objective Polynomially. *Math. Prog. Study*, **26**, 237–239.

[29] Monma, C.L. and M. Segal (1982). A Primal Algorithm for Finding Minimum-cost Flows in Capacitated Networks with Applications. *Bell System Technical Journal*, **61**, 449–468.

[30] Orlin, J.B. and R.K. Ahuja (1988). New Scaling Algorithms for the Assignment and Minimum Cycle Mean Problems. *Mathematical Programming* **54**, 41–56.

[31] Radzik, T. (1992). Minimizing Capacity Violations in a Transshipment Network. *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms*, Orlando 1992, 185–194

[32] Radzik, T. (1992). Newton's Method for Fractional Combinatorial Optimization. Stanford Computer Science technical report, Stanford, CA; an extended abstract appears in *Proc. 33rd IEEE Annual Symp. of Foundations of Computer Science*, 659–669.

[33] Radzik, T., and A.V. Goldberg (1991). Tight Bounds on the Number of Minimum-Mean Cycle Cancellations and Related Results. *Proceedings of the Second ACM-SIAM Symposium on Discrete Algorithms*, pp. 110–119; to appear in *Algorithmica*.

[34] Rockafellar, R.T. (1970). *Convex Analysis.* Princeton University Press, Princeton, NJ.

[35] Rockafellar, R.T. (1984). *Network Flows and Monotropic Optimization.* Wiley Interscience, New York, NY.

[36] Rote, G. (1991). An Improved Time Bound for Computing Maximum Mean Cuts. Talk given at the *Fourteenth International Symposium on Mathematical Programming*, Amsterdam 1991.

[37] Schneider, M.H. and S.A. Zenios (1990). A Comparative Study of Algorithms for Matrix Balancing. *Operations Research*, **38**, 439–455.

[38] Schrijver, A. (1986). *Theory of Linear and Integer Programming.* John Wiley & Sons, New York, NY.

[39] Tardos, É. (1986). A Strongly Polynomial Algorithm to Solve Combinatorial Linear Programs. *Operations Research*, **34**, 250–256.

[40] Welsh, D.J.A. (1976). *Matroid Theory.* Academic Press, London.