
A LOCALIZED METHOD FOR MULTICOMMODITY FLOW PROBLEM

A PREPRINT

Pengfei Liu

Building No.1, Zhonguancun Road, Haidian District
Beijing, China
pengfeil89@foxmail.com

February 16, 2024

ABSTRACT

This paper presents a local-control approach to address the multicommodity flow problem. The method involves relaxing both capacity constraints and flow conservation constraints. If the flow exceeds the capacity on an edge, the edge would incur a congestion cost. If the flow into a vertex is not equal to that out of the vertex, the vertex would have a height. Subsequently, a new concept, stable pseudo-flow, is introduced. Potential difference reduction algorithms, which don't rely on any shortest path or augmenting path, are designed to obtain stable pseudo-flow. If the stable pseudo-flow is a nonzero-stable pseudo-flow, there exists no feasible solution for multicommodity flow problem. Conversely, if the stable pseudo-flow is a zero-stable pseudo-flow, the feasible solution exists and the zero-stable pseudo-flow is the feasible solution. Notably, the algorithms work in a localized manner and can be efficiently implemented in parallel, which would further enhance performance.

Keywords local-control · optimality condition · exact algorithm · potential difference · multicommodity flow

1 Introduction

The *multicommodity flow problem* involves designing the flow for several different commodities through a common network with varying edge capacities. Given a directed graph $G(V, E)$ which has n vertexes and m edges, a capacity function $u : E \rightarrow Q^+$, K origin-destination pairs of nodes defined by (s_k, t_k, d_k) where s_k and t_k are the origin and destination of commodity k , and d_k is the demand, the objective is to obtain an assignment of flow which satisfies the demand for each commodity without violating the capacity constraints. The constraints can be summarized as follows:

$$\begin{aligned} \sum_{k \in \mathcal{K}} f_{ij,k} &\leq u_{ij}, \forall (i, j) \in E \\ \sum_{j \in \delta^+(i)} f_{ij,k} - \sum_{j \in \delta^-(i)} f_{ji,k} &= \begin{cases} d_k, & \text{if } i = s_k \\ -d_k, & \text{if } i = t_k \\ 0, & \text{if } i \in V - \{s_k, t_k\} \end{cases}, \forall k \in \mathcal{K}, i \in V \\ f_{ij,k} &\geq 0, \forall k \in \mathcal{K}, (i, j) \in E, \end{aligned} \quad (1)$$

where $\delta^+(i) = \{j | (i, j) \in E\}$, $\delta^-(i) = \{j | (j, i) \in E\}$ and $\mathcal{K} = \{1, 2, \dots, K\}$. In this paper, we assume $s_k \neq t_k$. The first expressions are capacity constraints. The second are flow conservation constraints, and the last are non-negative constraints.

Multicommodity flow problems have attracted great attention since the publication of the works of [3] and [4]. Comprehensive surveys are given about this problem [5, 6, 7]. Many special solution methods based on linear programming have been suggested to exploit, in some way, the special block-angular structure of multicommodity flow problems [15, 16, 17, 18, 20, 32, 19, 14, 21]. Yet, the studies conducted by [46] and [47] demonstrate that general

linear programs can be transformed into Two-Commodity Flow through Nearly-Linear Time Reductions. Recently, Brand and Zhang[44] give a high accuracy algorithm for multi-commodity flow problem from single-commodity methods. Besides, network equilibrium [9, 10, 11] is proposed to solve multicommodity flow problem.

There are also many approximation methods for multicommodity flow problem [33, 24, 23, 22]. Awerbuch and Leighton[25, 26] propose an approximation algorithm which uses local-control techniques similar to the preflow-push algorithm of Goldberg and Tarjan [27, 8]. Unlike previous approximation algorithms that attempt to find shortest paths or augmenting paths to push flow towards sinks, their algorithm uses an "edge-balancing" technique that attempts to send a commodity across an edge (i, j) if there is more of the commodity queued at i than that queued at j .

Goldberg and Tarjan[27, 8] relax the flow conservation constraints and propose the push-relabel algorithm for the maximum-flow problem, which is considered one of the most efficient maximum flow algorithms. Recent improvements compute the max-flow by a sequence of electric flows[37, 38, 39, 40, 41, 42, 43, 48]. There are also some pseudo-flow approaches[34, 35, 36] for solving max-flow problem.

Liu[9] gives an algorithm for the multi-commodity flow problems by relaxing the capacity constraints. In this paper, not only the capacity constraints but also the flow conservation constraints are relaxed. We introduce the congestion function for each edge and height function for each vertex and commodity. Then a simple rule is given, that is, the commodity k should flow from vertex i to vertex j if the height difference between vertex i and vertex j for commodity k is greater than the congestion of edge (i, j) . If the feasible region of Expression (1) is not empty, the feasible solution would be obtained by this simple rule. This is consistent with our common sense: water finds its level.

The network notation introduced here is summarized in Table 1. Further notation is introduced as needed.

Table 1: Basic Network Notation

$G(V, E)$	a directed graph
V	node (index) set
E	edge (index) set
\mathcal{K}	set of commodities, i.e., $\mathcal{K} = \{1, \dots, k, \dots, K\}$
$f_{ij,k}$	flow of commodity k on edge (i, j) , $\mathbf{f} = (\dots, f_{ij,k}, \dots)$
f_{ij}	flow on edge (i, j) , i.e., $f_{ij} = \sum_{k \in \mathcal{K}} f_{ij,k}$
(s_k, t_k, d_k)	s_k and t_k are the origin and destination of commodity k , and d_k is the demand
u_{ij}	the capacity of edge (i, j)
$\delta^+(i)$	$\{j (i, j) \in E\}$
$\delta^-(i)$	$\{j (j, i) \in E\}$
ψ_{ij}	congestion function of edge (i, j)
h_{ik}	height function of vertex i for commodity k
Δ_{ik}	the amount by which the k th flow into the vertex i exceeds that out of the
r_{ij}	the unused capacity for edge (i, j)
\mathbf{x}	the concat of \mathbf{f} and \mathbf{r} , i.e., $\mathbf{x} = \begin{pmatrix} \mathbf{f} \\ \mathbf{r} \end{pmatrix}$
\mathbf{x}_j	the j th component of \mathbf{x}
m	the number of edges of graph $G(V, E)$
n	the number of vertexes of graph $G(V, E)$
λ_{max}	the largest eigenvalue of $Q^T Q$
Λ	the maximum degree of graph $G(V, E)$
$\phi_{ij,k}$	the potential difference of edge (i, j) and commodity k

1.1 Our Contribution

It is certainly worth noting that *the worst-case runtime of our algorithm is still an open question*. So why have we spent time writing the paper? There are several reasons summarized as follows.

- (i) By introducing height and congestion functions, we give a very simple and intuitive optimality condition, which may inspire the research on fast and practical algorithms for the multicommodity flow problem.
- (ii) Unlike the previously existing algorithms for the multicommodity flow problem, our new algorithm could be executed in a localized manner, which are especially suited to environments where global control is not possible. Besides, the algorithm can work in parallel, which would greatly improve the computational efficiency.
- (iii) We give a practical potential difference reduction algorithm by using an inexact line search method.

- (iv) We give an algorithm with complexity $O(K|E|\frac{\lambda_{max}}{\alpha} \ln \frac{\Lambda \sum_{k \in \mathcal{K}} d_k}{\epsilon \alpha})$ for the multicommodity flow problem, where Λ is the maximum degree of graph $G(V, E)$ and λ_{max} the largest eigenvalue of $Q^T Q$ (see Expression 13). For the definition of α , see the proof of Lemma 6. Note that α may be very small and the estimate of its lower bound remains an open question.

2 Optimality Condition

2.1 Stable pseudo-flow

Unlike other methods, our method *does not maintain the capacity constraints and flow conservation constraints*. The method, however, maintains a *pseudo-flow*, which is a function $\mathbf{f} : K \times E \rightarrow \mathbb{R}^+$. Let Δ_{ik} be the amount by which the flow of commodity k into the vertex i exceeds that out of the vertex, i.e.,

$$\Delta_{ik} = \sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} + \hat{\Delta}_{ik}, \forall i \in V, k \in \mathcal{K}, \quad (2)$$

where $\delta^+(i) = \{j | (i, j) \in E\}$, $\delta^-(i) = \{j | (j, i) \in E\}$ and $\hat{\Delta}_{ik}$ is defined as:

$$\hat{\Delta}_{ik} = \begin{cases} 0 & \text{if } i \in V - \{s_k, t_k\} \\ d_k & \text{if } i = s_k \\ -d_k & \text{if } i = t_k. \end{cases} \quad (3)$$

For each commodity k at each vertex i , a height function $h_{ik} = h_{ik}(\Delta_{ik})$ is introduced, where $h_{ik}(\cdot)$ represents the relationship between Δ_{ik} and the height of vertex i for commodity k . Specifically, h_{ik} is defined as

$$h_{ik}(\Delta_{ik}) = \Delta_{ik}. \quad (4)$$

That is, the height of vertex i for commodity k is the amount by which the flow of commodity k into the vertex exceeds that out of that vertex.

Additionally, a *congestion function* $\psi_{ij} = \psi_{ij}(f_{ij})$ for each edge is introduced, where $\psi_{ij}(\cdot)$ represents the relationship between the flow and the degree of congestion for edge (i, j) . Specifically, ψ_{ij} is defined as

$$\psi_{ij}(f_{ij}) = \begin{cases} 0 & \text{if } f_{ij} \leq u_{ij} \\ f_{ij} - u_{ij} & \text{if } f_{ij} > u_{ij}, \end{cases} \quad (5)$$

where u_{ij} is the capacity of edge (i, j) and $f_{ij} = \sum_{k \in \mathcal{K}} f_{ij,k}$. If the flow on edge (i, j) is less than the capacity, the congestion of edge (i, j) is zero. Otherwise, the congestion of edge (i, j) is the amount by which the flow exceeds the capacity.

The greater the h_{ik} is, the higher the vertex i for commodity k is. The greater the ψ_{ij} is, the more 'congested' the edge (i, j) is. As is often said, water finds its level. Intuitively, if the height difference between vertex i and vertex j for commodity k is greater than the congestion of edge (i, j) , the flow $f_{ij,k}$ should be increased; if the height difference is less than the congestion, the flow $f_{ij,k}$ should be decreased. In the remainder of this paper, we would give a strict proof to demonstrate that the feasible solution for the multicommodity flow problem may be obtained by this intuitive idea.

Firstly, we introduce the concepts of *potential difference* and *stable pseudo-flow* here.

Definition 1 For every edge (i, j) and commodity k , define the potential difference $\phi_{ij,k}$ as the amount that the height difference between vertex i and vertex j minus the congestion of edge (i, j) , i.e.,

$$\phi_{ij,k} = h_{ik} - h_{jk} - \psi_{ij}. \quad (6)$$

Definition 2 By using $\{\psi_{ij}, \forall (i, j) \in E\}$ as the congestion function and $\{h_{ik}, \forall i \in V, \forall k \in \mathcal{K}\}$ as the height function, a pseudo-flow \mathbf{f} is called a *stable pseudo-flow* if it satisfies the following conditions:

- (i) for any used edge of commodity k , the height difference between vertex i and vertex j for commodity k is equal to the congestion of edge (i, j) , i.e., the potential difference $\phi_{ij,k}$ is zero;
- (ii) for any unused edge of commodity k , the height difference between vertex i and vertex j is less than or equal to the congestion of edge (i, j) , i.e., the potential difference $\phi_{ij,k}$ is less than or equal to zero;

where an edge (i, j) is called *used* by commodity k if there exists $s_k - t_k$ flow on edge (i, j) , otherwise it is called *unused*.

Definition 3 A stable pseudo-flow \mathbf{f} is called zero-stable pseudo-flow if $\psi_{ij} = 0$, $h_{ik} = 0$, $\forall (i, j) \in E, i \in V, k \in \mathcal{K}$. Otherwise, it is called nonzero-stable pseudo-flow.

From the definitions above, a zero-stable pseudo-flow is a feasible flow that satisfies Expression (1). Therefore, we have the following theorem:

Theorem 1 Given $\{(s_k, t_k, d_k) : k \in \mathcal{K}\}$ and capacity reservation $\{u_{ij} : (i, j) \in E\}$, the feasible region of Expression (1) is not empty if and only if there exists zero-stable pseudo-flow.

In fact, if a nonzero-stable pseudo-flow exists, there is no feasible solution for Expression (1). Before proving this conclusion, we would give a non-linear programming formulation of multicommodity flow problem, whose solution is stable pseudo-flow.

2.2 Basic Formulation

Let f_{ij} be the sum of the flow of all pairs on edge (i, j) and Δ_{ik} defined as Expression (2). Define the following programming:

$$\begin{aligned} \min \quad z &= \sum_{(i,j) \in E} \int_0^{f_{ij}} \psi_{ij}(\omega) d\omega + \sum_{i,k} \int_0^{\Delta_{ik}} h_{ik}(\omega) d\omega \\ \text{s.t} \quad & f_{ij,k} \geq 0, \forall k \in \mathcal{K}, (i, j) \in E \end{aligned} \quad (7)$$

where ψ_{ij} is the congestion function and h_{ik} the height function.

In the above programming, the objective function is the sum of the integrals of the edge congestion functions and the integrals of the vertex height functions. It should be noted that there are only non-negative constraints present and no capacity constraint or flow conservation constraint. According to the definitions of the congestion function denoted by ψ and the height function denoted by h , the following lemma can be derived:

Lemma 1 The feasible region of Expression (1) is not empty if and only if the minimum value of the objective function of Expression (7) is zero. Conversely, the feasible region of Expression (1) is empty if and only if the minimum value of the objective function of Expression (7) is greater than zero.

2.3 Equivalence

To demonstrate the equivalence between the stable pseudo-flow and the optimal solution of Programming (7), it has to be shown that any flow pattern that solves Programming (7) satisfies the stable conditions. This equivalency is demonstrated by proving that the Karush-Kuhn-Tucker conditions for Programming (7) are identical to the stable conditions.

Lemma 2 Let \mathbf{f}^* be a solution of Programming (7). \mathbf{f}^* is the optimal solution of Programming (7) if and only if it satisfies the Karush-Kuhn-Tucker conditions of Programming (7).

Proof: Firstly, the objective function of Programming (7) is convex. Secondly, the inequality constraints of Programming (7) are continuously differentiable concave functions. Therefore, Karush-Kuhn-Tucker conditions are necessary and sufficient for the optimality of Programming (7) (see [12]). \square

Since Programming (7) is a minimization problem with non-negativity constraints, the Karush-Kuhn-Tucker conditions of such formulation are as follows:

Stationarity

$$-\frac{\partial z}{\partial f_{ij,k}} = -\mu_{ij,k}, \forall k \in \mathcal{K}, (i, j) \in E$$

Primal feasibility

$$-f_{ij,k} \leq 0, \forall k \in \mathcal{K}, (i, j) \in E \quad (8)$$

Dual feasibility

$$\mu_{ij,k} \geq 0, \forall k \in \mathcal{K}, (i, j) \in E$$

Complementary slackness

$$\mu_{ij,k} f_{ij,k} = 0, \forall k \in \mathcal{K}, (i, j) \in E.$$

Obviously,

$$\begin{aligned} \frac{\partial z}{\partial f_{ij,k}} &= \psi_{ij}(f_{ij}) \frac{\partial f_{ij}}{\partial f_{ij,k}} + h_{ik}(\Delta_{ik}) \frac{\partial \Delta_{ik}}{\partial f_{ij,k}} + h_{jk}(\Delta_{jk}) \frac{\partial \Delta_{jk}}{\partial f_{ij,k}} \\ &= \psi_{ij}(f_{ij}) + (-h_{ik}(\Delta_{ik})) + (h_{jk}(\Delta_{jk})) \\ &= \psi_{ij} + h_{jk} - h_{ik}. \end{aligned} \quad (9)$$

Substituting the expression above into the Stationarity expression in KKT conditions,

$$h_{ik} - h_{jk} - \psi_{ij} = -\mu_{ij,k}, \forall k \in \mathcal{K}, (i, j) \in E.$$

For any used edge of commodity k , i.e. $f_{ij,k} > 0$, by complementary slackness $\mu_{ij,k} f_{ij,k} = 0$ in KKT conditions, we have $\mu_{ij,k} = 0$. Therefore,

$$h_{ik} - h_{jk} - \psi_{ij} = 0, \forall k \in \mathcal{K}, (i, j) \in E.$$

That is, the potential difference between vertex i and vertex j for commodity k is equal to zero for any used edge of commodity k .

For any unused edge of commodity k , due to $\mu_{ij,k} \geq 0$, we have

$$h_{ik} - h_{jk} - \psi_{ij} = -\mu_{ij,k} \leq 0, \forall k \in \mathcal{K}, (i, j) \in E.$$

That is, the potential difference between vertex i and vertex j for commodity k is less than or equal to zero for any unused edge of commodity k .

With the interpretation above, it is now clear that:

- (i) for any used edge of commodity k , the potential difference between vertex i and vertex j for commodity k is zero, i.e. $\phi_{ij,k} = 0$;
- (ii) for any unused edge of commodity k , the potential difference between vertex i and vertex j for commodity k is less than or equal to zero, i.e. $\phi_{ij,k} \leq 0$.

So we have the following lemma:

Lemma 3 *The optimal solution of Programming (7) is a stable pseudo-flow.*

Obviously, a stable pseudo-flow also satisfies the KKT conditions. By Lemma 2, we have

Lemma 4 *The stable pseudo-flow is the optimal solution of Programming (7).*

By Lemma 1, Lemma 3 and Lemma 4, Theorem 2 holds.

Theorem 2 *Given $\{(s_k, t_k, d_k) : k \in \mathcal{K}\}$ and capacity reservation $\{u_{ij} : (i, j) \in E\}$, the feasible region of Expression (1) is empty if and only if there exists nonzero-stable pseudo-flow.*

Remark 1 *By Theorem 1 and Theorem 2, the optimality condition for multicommodity flow problem may be summarized as follows:*

- (i) *if there exists a nonzero-stable pseudo-flow, there exists no feasible solution for multicommodity flow problem;*
- (ii) *if there exists zero-stable pseudo-flow, there exists a feasible solution and the zero-stable pseudo-flow is the feasible solution.*

Remark 2 In fact, both Theorem 1 and Theorem 2 are true if the height function and the congestion function satisfy the following conditions:

(i) the height function $h_{ik}(\Delta_{ik})$ is a strictly monotone increasing function and $h_{ik}(0) = 0$;

(ii) the congestion function ψ_{ij} satisfies the following definition:

$$\psi(f_{ij}) = \begin{cases} 0 & \text{if } f_{ij} \leq u_{ij} \\ g(f_{ij} - u_{ij}) & \text{if } f_{ij} > u_{ij}, \end{cases}$$

where $g(0) = 0$ and $g(\cdot)$ is a strictly monotone increasing function.

Remark 3 Expression (9) shows that the potential difference is exactly the same as the negative gradient of the objective function of Programming (7), which is the key to prove the convergence of the following algorithms.

3 Potential Difference Reduction Algorithm

Theorem 1 and Theorem 2 above suggest a simple algorithmic approach for solving the multicommodity flow problem. That is, design an algorithm to obtain the stable pseudo-flow, which may be achieved by adjusting the commodity flow $f_{ij,k}$ until the potential difference $\phi_{ij,k} = 0$ or $f_{ij,k} = 0$. In other words, if the potential difference between vertex i and vertex j for commodity k is greater than zero, the flow $f_{ij,k}$ should be increased; if the potential difference is less than zero and $f_{ij,k} > 0$, the flow $f_{ij,k}$ should be decreased. In this paper we call this algorithm the potential difference reduction algorithm. Before describing the algorithm, we need to rewrite Programming (7) as a quadratic programming problem with nonnegative constraints.

3.1 Quadratic Programming

Firstly,

$$\int_0^{\Delta_{ik}} h_{ik}(\omega) d\omega = \frac{1}{2} \Delta_{ik}^2 = \frac{1}{2} \left(\sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} + \hat{\Delta}_{ik} \right)^2. \quad (10)$$

By the definition of congestion function ψ_{ij} , we have

$$\int_0^{f_{ij}} \psi_{ij}(\omega) d\omega = \begin{cases} 0 & \text{if } f_{ij} \leq u_{ij} \\ \frac{1}{2} (f_{ij} - u_{ij})^2 & \text{if } f_{ij} > u_{ij}. \end{cases} \quad (11)$$

By introducing an auxiliary variable $\{r_{ij} : r_{ij} \geq 0\}$ which is the unused capacity for edge (i, j) , Programming (7) could be rewritten as

$$\begin{aligned} \min \quad & z = \frac{1}{2} \sum_{(i,j) \in E} \left(\sum_{k \in \mathcal{K}} f_{ij,k} + r_{ij} - u_{ij} \right)^2 + \frac{1}{2} \sum_{i,k} \left(\sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} + \hat{\Delta}_{ik} \right)^2 \\ \text{s.t.} \quad & f_{ij,k} \geq 0, \forall k \in \mathcal{K}, (i, j) \in E \\ & r_{ij} \geq 0, \forall (i, j) \in E \end{aligned} \quad (12)$$

Programming (12) is a quadratic programming problem with non-negative constraints, which could be rewritten in the matrix form.

To write the matrix form, we define a $n \times m$ matrix A of $\text{Graph}(V, E)$ with element A_{ve} for the vertex v and the edge e (connecting vertexes v_i and v_j) defined by

$$A_{ve} = \begin{cases} -1 & \text{if } v = v_i \\ 1 & \text{if } v = v_j \\ 0 & \text{otherwise.} \end{cases}$$

Now define P as a $nK \times m(K+1)$ matrix composed of $K \times (K+1)$ blocks of size $n \times m$ each:

$$P = \begin{bmatrix} A & 0 & \cdots & 0 & 0 \\ 0 & A & & \vdots & \vdots \\ \vdots & & \ddots & 0 & \vdots \\ 0 & \cdots & 0 & A & 0 \end{bmatrix}.$$

Define C as a $m \times m(K + 1)$ matrix constructed from $K + 1$ blocks of identity matrices of dimension $m \times m$:

$$C = (I_{m \times m} I_{m \times m} \cdots I_{m \times m}).$$

For simplicity, define

$$Q = \begin{pmatrix} P \\ C \end{pmatrix}, \quad \mathbf{x} = \begin{pmatrix} \mathbf{f} \\ \mathbf{r} \end{pmatrix}. \quad (13)$$

Then the objective function of Programming (12) could be rewritten as:

$$\begin{aligned} z &= \frac{1}{2} \sum_{(i,j) \in E} \left(\sum_{k \in \mathcal{K}} f_{ij,k} + r_{ij} - u_{ij} \right)^2 + \frac{1}{2} \sum_{i,k} \left(\sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} + \hat{\Delta}_{ik} \right)^2 \\ &= \frac{1}{2} (\mathbf{Q}\mathbf{x} - \mathbf{b})^T (\mathbf{Q}\mathbf{x} - \mathbf{b}) \\ &= \frac{1}{2} \mathbf{x}^T \mathbf{Q}^T \mathbf{Q} \mathbf{x} - \mathbf{b}^T \mathbf{Q} \mathbf{x} + \frac{1}{2} \mathbf{b}^T \mathbf{b}, \end{aligned} \quad (14)$$

where $b_{ik} = -\hat{\Delta}_{ik}$ and $b_{ij} = u_{ij}$.

Therefore, Programming (12) may be rewritten as:

$$\begin{aligned} \min \quad & z = \frac{1}{2} \mathbf{x}^T \mathbf{Q}^T \mathbf{Q} \mathbf{x} - \mathbf{b}^T \mathbf{Q} \mathbf{x} + \frac{1}{2} \mathbf{b}^T \mathbf{b} \\ \text{s.t.} \quad & \mathbf{x} \geq \mathbf{0} \end{aligned} \quad (15)$$

The gradient projection methods [13, 1, 2] are efficient for the above problem. Bertsekas [13] shows a linear rate of convergence of the gradient projection method under the assumption that the smallest eigenvalue of the matrix $\mathbf{Q}^T \mathbf{Q}$ is larger than zero. In this paper, we would show that the gradient projection method for Programming (15) is linearly convergent, though $\mathbf{Q}^T \mathbf{Q}$ is positive semi-definite.

3.2 Algorithm of linear convergence

Let λ_{max} be the largest eigenvalue of $\mathbf{Q}^T \mathbf{Q}$ and $\mathbf{x} \geq \mathbf{0}$ means that every element of \mathbf{x} is greater than or equal to zero. The potential difference reduction algorithm for multicommodity flow problem is as follows:

Algorithm 1 Potential difference reduction algorithm

1: Given $\mathbf{x}^0 \geq 0$; $\sigma \in (0, 1)$ and set the step size $\beta \leq \frac{2\sigma}{\lambda_{max}}$

2: **for** $l = 0, 1, \dots$, **do**

3: Calculate the height $h_{ik}^l, \forall i \in V, k \in \mathcal{K}$:

$$h_{ik}^l = \sum_{j \in \delta^-(i)} f_{ji,k}^l - \sum_{j \in \delta^+(i)} f_{ij,k}^l + \hat{\Delta}_{ik}.$$

4: Calculate the congestion $\psi_{ij}^l, \forall (i, j) \in E$:

$$\psi_{ij}^l = \sum_{k \in \mathcal{K}} f_{ij,k}^l + r_{ij}^l - u_{ij}.$$

5: Calculate the potential difference $\phi_{ij,k}^l, \forall (i, j) \in E, k \in \mathcal{K}$:

$$\phi_{ij,k}^l = h_{ik}^l - h_{jk}^l - \psi_{ij}^l.$$

6: Update $f_{ij,k}^{l+1}, r_{ij}^{l+1}, \forall (i, j) \in E, k \in \mathcal{K}$:

$$\begin{aligned} f_{ij,k}^{l+1} &= \max\{f_{ij,k}^l + \beta\phi_{ij,k}^l, 0\} \\ r_{ij}^{l+1} &= \max\{r_{ij}^l - \beta\psi_{ij}^l, 0\}. \end{aligned}$$

7: **end for**

Note that the flow $f_{ij,k}$ is updated by adding the potential difference, but r_{ij} is updated by subtracting its gradient. As is shown by Equation (9), the negative gradient of the objective function of Programming (7) is exactly the same as the potential difference. Additionally, the gradient $\frac{\partial z}{\partial \mathbf{x}} = Q^T Q \mathbf{x} - Q^T b$. For simplicity, Algorithm 1 can be reduced to a much more condensed matrix form.

Algorithm 2 Potential difference reduction algorithm in matrix form

Initialization: Given $\mathbf{x}^0 \geq 0$, $\sigma \in (0, 1)$. Set the step size $\beta \leq \frac{2\sigma}{\lambda_{max}}$ and $\mathbf{y}^0 := \mathbf{x}^0$.

Iterative step: Given \mathbf{x}^l , calculate:

$$\begin{cases} \mathbf{y}^{l+1} = \mathbf{x}^l - \beta(Q^T Q \mathbf{x}^l - Q^T b) \\ \mathbf{x}^{l+1} = \max(\mathbf{y}^{l+1}, 0) \end{cases} \quad (16)$$

Remark 4 Obviously, Algorithm 1 and Algorithm 2 are exactly equivalent.

Lemma 5 $\{\mathbf{x}^l\}_{l \geq 0}$, generated by Algorithm 2, is convergent. Furthermore, $\{\mathbf{x}^l\}_{l \geq 0}$ converges to an optimal solution of Programming (15).

Remark 5 Lemma 5 can be directly deduced as a special case from Theorem 6 in [31]. We refer readers to [31] for more details.

Lemma 6 Let $\{\mathbf{x}^l\}_{l \geq 0}$ and $\{\mathbf{y}^l\}_{l \geq 0}$ be the sequence generated by Algorithm 2. Then there exists a real $q \in (0, 1)$ such that, for all $l \geq 0$,

$$\|\mathbf{y}^{l+2} - \mathbf{y}^{l+1}\|_2 \leq (1 - q)\|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2.$$

Proof: Set $B = \beta Q^T Q$, which is a symmetric positive semi-definite matrix. Let λ_{max}^B the largest eigenvalue of B . Obviously, $\lambda_{max}^B = \beta \lambda_{max} \leq 2\sigma < 2$. So we have

$$\|(I - B)x\|_2 \leq \|x\|_2. \quad (17)$$

We denote by $\mathcal{N}(Q)$ the null space of the matrix Q and define the real number η

$$\eta = 1 - \max \left\{ \frac{\|P_{\mathcal{N}(Q)}(\mathbf{x})\|_2}{\|\mathbf{x}\|_2} \mid \mathbf{x} \geq 0, \mathbf{x} \neq 0 \right\}, \quad (18)$$

where $P_{\mathcal{N}(Q)}(\mathbf{x})$ is the projection of \mathbf{x} onto $\mathcal{N}(Q)$.

Since

$$\begin{aligned} \mathbf{x}^T B \mathbf{x} &= \beta \left(\sum_{(i,j) \in E} \left(\sum_{k \in \mathcal{K}} f_{ij,k} + r_{ij} \right)^2 + \sum_{i,k} \left(\sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} \right)^2 \right) \\ &\geq \beta \left(\sum_{(i,j) \in E} \sum_{k \in \mathcal{K}} f_{ij,k}^2 + \sum_{(i,j) \in E} r_{ij}^2 \right) \\ &= \beta \mathbf{x}^T \mathbf{x}, \forall \mathbf{x} \geq 0, \end{aligned} \quad (19)$$

we have $\mathbf{x}^T B \mathbf{x} \geq \beta \mathbf{x}^T \mathbf{x} = \beta \|\mathbf{x}\|_2^2$.

Let $\mathbf{x} = \mathbf{x}_0 + \mathbf{x}_1$ where $\mathbf{x}_0 \in \mathcal{N}(Q)$ and \mathbf{x}_1 is orthogonal to $\mathcal{N}(Q)$. Since $\mathbf{x}^T B \mathbf{x} = \mathbf{x}_1^T B \mathbf{x}_1 \leq \lambda_{max}^B \|\mathbf{x}_1\|_2^2$, we have $\beta \|\mathbf{x}\|_2^2 \leq \lambda_{max}^B \|\mathbf{x}_1\|_2^2$. That is,

$$\begin{aligned} \|\mathbf{x}_0\|_2^2 &= \|\mathbf{x}\|_2^2 - \|\mathbf{x}_1\|_2^2 \\ &\leq \left(1 - \frac{\beta}{\lambda_{max}^B}\right) \|\mathbf{x}\|_2^2 \\ &\leq \left(1 - \frac{\beta}{2}\right) \|\mathbf{x}\|_2^2 \\ &\leq \left(1 - \frac{\beta}{4}\right)^2 \|\mathbf{x}\|_2^2. \end{aligned} \quad (20)$$

By Inequality (20) and Equation (18), we have

$$\frac{\beta}{4} \leq \eta \leq 1. \quad (21)$$

Let J be a nonempty subset of $\{1, 2, \dots, N\}$ where N is the dimension of \mathbf{x} . Denote the family of all such subsets by \mathcal{J} . For each $J \in \mathcal{J}$, introduce the subspace $X_J \in R^N$,

$$X_J = \{\mathbf{x} \in R^N \mid \mathbf{x}_j = 0, \forall j \notin J\}. \quad (22)$$

Define $H_J = X_J \cap \mathcal{N}(Q)$ and denote by Y_J the subspace of X_J which is orthogonal to H_J . Given any $J \in \mathcal{J}$, let α_J be the largest non-negative number for which

$$\|(I - B)\mathbf{y}\|_2 \leq (1 - \alpha_J) \|\mathbf{y}\|_2, \forall \mathbf{y} \in Y_J, \quad (23)$$

where I denotes the unit matrix. Clearly, α_J is actually positive.

The first line of Equation (16) could be rewritten as:

$$\mathbf{y}^{l+1} = \mathbf{x}^l - (B\mathbf{x}^l - \beta Q^T \mathbf{b}). \quad (24)$$

Therefore,

$$\mathbf{y}^{l+1} - \mathbf{y}^l = (I - B)(\mathbf{x}^l - \mathbf{x}^{l-1}). \quad (25)$$

Furthermore, if $\mathbf{y}_j^{l+1} \geq 0$ or $\mathbf{y}_j^l \geq 0$, using $\mathbf{x}^l = \max(\mathbf{y}^l, 0)$, we have

$$|\mathbf{x}_j^{l+1} - \mathbf{x}_j^l| + |\mathbf{y}_j^{l+1} - \mathbf{x}_j^{l+1}| + |\mathbf{y}_j^l - \mathbf{x}_j^l| = |\mathbf{y}_j^{l+1} - \mathbf{y}_j^l|. \quad (26)$$

Since

$$\begin{aligned} \mathbf{h}^T (B\mathbf{x} - \beta Q^T \mathbf{b}) &= \mathbf{h}^T B \mathbf{x} - \mathbf{h}^T \beta Q^T \mathbf{b} \\ &= \mathbf{h}^T (\beta Q^T Q) \mathbf{x} - \mathbf{h}^T (\beta Q^T \mathbf{b}) \\ &= \beta (Q\mathbf{h})^T (Q\mathbf{x} - \mathbf{b}) \\ &= 0, \forall \mathbf{h} \in \mathcal{N}(Q), \end{aligned}$$

it follows that

$$\mathbf{h}^T (\mathbf{y}^{l+1} - \mathbf{x}^l) = 0, \forall \mathbf{h} \in \mathcal{N}(Q). \quad (27)$$

Now let $\mathbf{w}^l = \mathbf{x}^l - \mathbf{x}^{l-1}$ and define

$$J_l = \{j \mid \mathbf{y}_j^l \geq 0 \text{ or } \mathbf{y}_j^{l-1} \geq 0\}. \quad (28)$$

Decompose \mathbf{w}^l , which clearly belongs to X_{J_l} , as $\mathbf{w}^l = \mathbf{w}^{l,0} + \mathbf{w}^{l,1}$ where $\mathbf{w}^{l,0} \in H_{J_l}$ and $\mathbf{w}^{l,1} \in Y_{J_l}$, and consider the following two cases.

Case I. $\|\mathbf{w}^{l,0}\|_2 \leq (1 - \varepsilon)\|\mathbf{w}^l\|_2$ for some $\varepsilon \in (0, 1)$.

$$\begin{aligned}
\|(I - B)\mathbf{w}^l\|_2^2 &= \|(I - B)\mathbf{w}^{l,0}\|_2^2 + \|(I - B)\mathbf{w}^{l,1}\|_2^2 \\
&\leq \|\mathbf{w}^{l,0}\|_2^2 + (1 - \alpha_J)^2 \|\mathbf{w}^{l,1}\|_2^2 \\
&= (2\alpha_J - \alpha_J^2) \|\mathbf{w}^{l,0}\|_2^2 + (1 - \alpha_J)^2 \|\mathbf{w}^l\|_2^2 \\
&\leq (1 - \varepsilon)^2 (2\alpha_J - \alpha_J^2) \|\mathbf{w}^l\|_2^2 + (1 - \alpha_J)^2 \|\mathbf{w}^l\|_2^2 \\
&= (1 - \varepsilon\alpha_J(2 - \varepsilon)(2 - \alpha_J)) \|\mathbf{w}^l\|_2^2.
\end{aligned} \tag{29}$$

Hence,

$$\|(I - B)\mathbf{w}^l\|_2^2 \leq (1 - \varepsilon\alpha_J) \|\mathbf{w}^l\|_2^2. \tag{30}$$

By Equation (25) and Equation (26),

$$\begin{aligned}
\|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2 &= \|(I - B)(\mathbf{x}^l - \mathbf{x}^{l-1})\|_2 \\
&\leq \sqrt{(1 - \varepsilon\alpha_J)} \|\mathbf{x}^l - \mathbf{x}^{l-1}\|_2 \\
&\leq \sqrt{(1 - \varepsilon\alpha_J)} \|\mathbf{y}^l - \mathbf{y}^{l-1}\|_2.
\end{aligned} \tag{31}$$

Case II. $\|\mathbf{w}^{l,0}\|_2 \geq (1 - \varepsilon)\|\mathbf{w}^l\|_2$ for some $\varepsilon \in (0, 1)$.

Rewrite $\mathbf{w}^l = \mathbf{y}^l - \mathbf{x}^{l-1} + \mathbf{x}^l - \mathbf{y}^l$. Define $\bar{\mathbf{x}}$ as following:

$$\bar{\mathbf{x}}_j = \begin{cases} \mathbf{x}_j^l - \mathbf{y}_j^l, \forall j \in J_l \\ 0, \forall j \notin J_l \end{cases}.$$

By Equation (27), the vector $\mathbf{y}^l - \mathbf{x}^{l-1}$ is orthogonal to H_{J_l} . Therefore, $\mathbf{w}^{l,0} = P_{H_{J_l}}(\mathbf{x}^l - \mathbf{y}^l) = P_{H_{J_l}}(\bar{\mathbf{x}})$ is the projection of $\bar{\mathbf{x}}$ onto H_{J_l} , from which, by Equation (18), we have

$$\begin{aligned}
(1 - \varepsilon)\|\mathbf{w}^l\|_2 &\leq \|\mathbf{w}^{l,0}\|_2 = \|P_{H_{J_l}}(\bar{\mathbf{x}})\|_2 \\
&\leq (1 - \eta)\|\bar{\mathbf{x}}\|_2.
\end{aligned}$$

Therefore,

$$\|\mathbf{w}^l\|_2 \leq \frac{(1 - \eta)}{(1 - \varepsilon)} \|\bar{\mathbf{x}}\|_2,$$

which, together with Equation (26), implies

$$\|\mathbf{w}^l\|_2 \leq \frac{(1 - \eta)}{(1 - \varepsilon)} \|\mathbf{y}^l - \mathbf{y}^{l-1}\|_2. \tag{32}$$

By Equation (25) and Inequality (17), we have the following the inequality

$$\begin{aligned}
\|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2 &= \|(I - B)\mathbf{w}^l\|_2 \\
&\leq \|\mathbf{w}^l\|_2 \leq \frac{(1 - \eta)}{(1 - \varepsilon)} \|\mathbf{y}^l - \mathbf{y}^{l-1}\|_2.
\end{aligned} \tag{33}$$

Taking $\varepsilon = \frac{\eta}{1 + \alpha_J}$, we can summarize Inequalities (31) and (33) as

$$\begin{aligned}
\|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2 &\leq \left(1 - \frac{\eta\alpha_J}{2(1 + \alpha_J)}\right) \|\mathbf{y}^l - \mathbf{y}^{l-1}\|_2 \\
&\leq (1 - q) \|\mathbf{y}^l - \mathbf{y}^{l-1}\|_2,
\end{aligned} \tag{34}$$

where

$$q = \min \left\{ \frac{\eta\alpha_J}{2(1 + \alpha_J)} \mid \forall J \in \mathcal{J} \right\}.$$

Set $\alpha = \min\{\alpha_J \mid \forall J \in \mathcal{J}\}$, then we have

$$q = \frac{\eta\alpha}{2(1 + \alpha)}. \tag{35}$$

□

Remark 6 The proof above is similar to that of Theorem 10 in [31]. The main difference, in comparison with the result in [31], is that here we do not assume every element in matrix Q to be greater than or equal to zero.

Remark 7 Although $Q^T Q$ is positive semi-definite, Algorithm 2 is of linear convergence rate, by Lemma 6.

Remark 8 According to the definition of α_J , it is evident that α is greater than zero. However, α may be very small and the estimate of its lower bound remains an open question.

Theorem 3 Let $\{\mathbf{x}^l\}_{l \geq 0}$ and $\{\mathbf{y}^l\}_{l \geq 0}$ be the sequence generated by Algorithm 2. \mathbf{x}^* and \mathbf{y}^* are the limits of the two sequences, respectively. Set the step size $\beta = \frac{2\sigma}{\lambda_{max}}$. Then the following statements hold:

- (i) we can find an index L in $O(\frac{\lambda_{max}}{\alpha} \ln \frac{\Lambda \sum_{k \in \mathcal{K}} d_k}{\epsilon \alpha})$ iterations such that $\|\mathbf{x}^l - \mathbf{x}^*\|_2 \leq \epsilon \forall l \geq L$;
- (ii) each iteration can be done in $O(K|E|)$ time.

Proof:

By Formulation (35) and Inequality (21),

$$q = \frac{\eta\alpha}{2(1+\alpha)} \geq \frac{\alpha\beta}{8(1+\alpha)} = \frac{\alpha\sigma}{4(1+\alpha)\lambda_{max}}. \quad (36)$$

By Lemma 6,

$$\|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2 \leq (1-q)^l \|\mathbf{y}^1 - \mathbf{y}^0\|_2. \quad (37)$$

Therefore,

$$\begin{aligned} \|\mathbf{y}^L - \mathbf{y}^*\|_2 &\leq \sum_{l=L}^{\infty} \|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2 \\ &\leq \|\mathbf{y}^1 - \mathbf{y}^0\|_2 \sum_{l=L}^{\infty} (1-q)^l \\ &= \|\mathbf{y}^1 - \mathbf{y}^0\|_2 \frac{(1-q)^L}{q}. \end{aligned} \quad (38)$$

Since $\mathbf{x}^l = \max\{\mathbf{y}^l, 0\}$, the following inequality holds,

$$\|\mathbf{x}^{l+1} - \mathbf{x}^l\|_2 \leq \|\mathbf{y}^{l+1} - \mathbf{y}^l\|_2, \quad (39)$$

together with Inequality (38), we have

$$\|\mathbf{x}^L - \mathbf{x}^*\|_2 \leq \|\mathbf{y}^1 - \mathbf{y}^0\|_2 \frac{(1-q)^L}{q}. \quad (40)$$

Define the starting point as $\{f_{ij,k}^0 = 0, r_{ij}^0 = u_{ij}, \forall (i, j) \in E, k \in \mathcal{K}\}$, i.e., $\mathbf{x}^0 = \mathbf{y}^0 = \begin{pmatrix} \mathbf{f}^0 \\ \mathbf{r}^0 \end{pmatrix}$. So

$$\begin{aligned}
\|\mathbf{y}^1 - \mathbf{y}^0\|_2 &= \|\beta(Q^T Q \mathbf{x}^0 - Q^T b)\|_2 \\
&= \beta \sqrt{\sum_{k \in \mathcal{K}} \sum_{(i,j) \in E} \left(\frac{\partial z}{\partial f_{ij,k}^0}\right)^2 + \sum_{(i,j) \in E} \left(\frac{\partial z}{\partial r_{ij}^0}\right)^2} \\
&= \beta \sqrt{\sum_{k \in \mathcal{K}} \sum_{(i,j) \in E} (\psi_{ij}^0 + h_{jk}^0 - h_{ik}^0)^2 + \sum_{(i,j) \in E} (\psi_{ij}^0)^2} \\
&= \beta \sqrt{\sum_{k \in \mathcal{K}} \sum_{(i,j) \in E} (0 + h_{jk}^0 - h_{ik}^0)^2 + \sum_{(i,j) \in E} 0^2} \\
&= \beta \sqrt{\sum_{k \in \mathcal{K}} \sum_{(i,j) \in E} (\hat{\Delta}_{jk} - \hat{\Delta}_{ik})^2} \tag{41} \\
&\leq \beta \sum_{k \in \mathcal{K}} \sum_{(i,j) \in E} |\hat{\Delta}_{jk} - \hat{\Delta}_{ik}| \\
&\leq \beta \sum_{k \in \mathcal{K}} (2\Lambda d_k) \\
&= 2\beta\Lambda \sum_{k \in \mathcal{K}} d_k \\
&= \frac{4\sigma\Lambda \sum_{k \in \mathcal{K}} d_k}{\lambda_{max}},
\end{aligned}$$

where Λ is the maximum degree of Graph $G(V, E)$.

$$\frac{(1-q)^l}{q} \|\mathbf{y}^1 - \mathbf{y}^0\|_2 \leq \epsilon \tag{42}$$

is satisfied, provide that

$$l * \ln(1-q) + \ln \|\mathbf{y}^1 - \mathbf{y}^0\|_2 \leq \ln(\epsilon q). \tag{43}$$

Since $\ln(1+q) \leq q, \forall q > -1$, Inequality (43) is true if

$$l \geq \frac{1}{q} \ln \frac{\|\mathbf{y}^1 - \mathbf{y}^0\|_2}{\epsilon q}. \tag{44}$$

By Inequality (36) and (41),

$$\frac{1}{q} \ln \frac{\|\mathbf{y}^1 - \mathbf{y}^0\|_2}{\epsilon q} \leq \frac{4(1+\alpha)\lambda_{max}}{\alpha\sigma} \ln \frac{16(1+\alpha)\Lambda \sum_{k \in \mathcal{K}} d_k}{\epsilon\alpha}. \tag{45}$$

Therefore, if

$$\begin{aligned}
l \geq L &= \frac{4(1+\alpha)\lambda_{max}}{\alpha\sigma} \ln \frac{16(1+\alpha)\Lambda \sum_{k \in \mathcal{K}} d_k}{\epsilon\alpha} \\
&= O\left(\frac{\lambda_{max}}{\alpha} \ln \frac{\Lambda \sum_{k \in \mathcal{K}} d_k}{\epsilon\alpha}\right), \tag{46}
\end{aligned}$$

Inequality (42) holds. By Inequality (40), the first statement of this theorem holds.

Obviously, the height $\{h_{ik}, \forall i \in V, k \in \mathcal{K}\}$ can be obtained in $2K|E|$ time; the congestion function $\{\psi_{ij}, \forall (i, j) \in E\}$ can be obtained in $O(K|E|)$ time. Therefore, each iteration of Algorithm 1 can be done in $O(K|E|)$ time. So the proof is complete. \square

Remark 9 Lemma 7 would give an estimate of the upper bound of λ_{max} . However, the estimate of the lower bound of α is still an open problem.

Remark 10 Since the largest eigenvalue λ_{max} is usually unavailable, it is not known a priori how small one should take the step size β . A possible way is to substitute the λ_{max} by its upper bound in Lemma 7.

Lemma 7 Let λ_{max} be the largest eigenvalue of $Q^T Q$, then

$$\lambda_{max} \leq K + 1 + 2\Lambda, \quad (47)$$

where Λ is the maximum degree of graph $G(V, E)$.

Proof:

$$\begin{aligned} \mathbf{x}^T Q^T Q \mathbf{x} &= \sum_{(i,j) \in E} \left(\sum_{k \in \mathcal{K}} f_{ij,k} + r_{ij} \right)^2 + \sum_{i,k} \left(\sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} \right)^2 \\ &\leq \sum_{(i,j) \in E} \left((K+1) \left(\sum_{k \in \mathcal{K}} f_{ij,k}^2 + r_{ij}^2 \right) \right) + \sum_{i,k} \left(\Lambda \left(\sum_{j \in \delta^+(i)} f_{ij,k}^2 + \sum_{j \in \delta^-(i)} f_{ji,k}^2 \right) \right) \\ &= (K+1) \sum_{(i,j) \in E} \left(\sum_{k \in \mathcal{K}} f_{ij,k}^2 + r_{ij}^2 \right) + 2\Lambda \sum_{k \in \mathcal{K}} \left(\sum_{(i,j) \in E} f_{ij,k}^2 \right) \\ &\leq (K+1+2\Lambda) \left(\sum_{(i,j) \in E} \sum_{k \in \mathcal{K}} f_{ij,k}^2 + \sum_{(i,j) \in E} r_{ij}^2 \right) \\ &= (K+1+2\Lambda) \mathbf{x}^T \mathbf{x}, \forall \mathbf{x}. \end{aligned}$$

Since $\mathbf{x}^T Q^T Q \mathbf{x} \leq (K+1+2\Lambda) \mathbf{x}^T \mathbf{x}$, $\lambda_{max} \leq K+1+2\Lambda$. \square

3.3 A Practical Algorithm

When λ_{max} is large, the step size β is small, which would lead to slow convergence. In this section we would give a practical potential difference reduction algorithm whose step size is determined by an inexact line search method.

Let $\phi(\mathbf{f})$ be the vector of potential difference when the flow is \mathbf{f} , i.e., $\phi(\mathbf{f}) = (\dots, \phi_{ij,k}, \dots)$. And $\max(\mathbf{u}, \mathbf{v})$ denotes the component-wise maximum of \mathbf{u}, \mathbf{v} . Algorithm 3 describes the adaptive potential difference reduction algorithm.

Algorithm 3 Adaptive potential difference reduction algorithm

- 1: Set $\beta_0 = 1, \mu = 0.5, v = 0.9$, given \mathbf{f}^0
 - 2: **for** $l = 0, 1, \dots$, if the stopping criterion is not satisfied, **do**
 - 3: calculate $\phi(\mathbf{f}^l)$ by Algorithm 4;
 - 4: $\hat{\mathbf{f}}^l = \max\{\mathbf{f}^l + \beta^l \phi(\mathbf{f}^l), \mathbf{0}\}$;
 - 5: calculate $\phi(\hat{\mathbf{f}}^l)$ by Algorithm 4;
 - 6: $\omega^l = \beta_l \|\phi(\mathbf{f}^l) - \phi(\hat{\mathbf{f}}^l)\|_2 / \|\mathbf{f}^l - \hat{\mathbf{f}}^l\|_2$.
 - 7: **while** $\omega^l > v$ **do**
 - 8: $\beta^l := \beta^l * 0.8 / \omega^l$;
 - 9: $\hat{\mathbf{f}}^l = \max\{\mathbf{f}^l + \beta^l \phi(\mathbf{f}^l), \mathbf{0}\}$;
 - 10: Calculate $\phi(\hat{\mathbf{f}}^l)$ by Algorithm 4;
 - 11: $\omega^l = \beta_l \|\phi(\mathbf{f}^l) - \phi(\hat{\mathbf{f}}^l)\|_2 / \|\mathbf{f}^l - \hat{\mathbf{f}}^l\|_2$.
 - 12: **end while**
 - 13: $\mathbf{f}^{l+1} = \hat{\mathbf{f}}^l$;
 - 14: $\phi(\mathbf{f}^{l+1}) = \phi(\hat{\mathbf{f}}^l)$.
 - 15: **if** $\omega^l \leq \mu$ **then**
 - 16: $\beta_l := \beta_l * 1.5$.
 - 17: **end if**
 - 18: **end for**
-

Algorithm 4 Given $f \geq 0$, compute potential difference

1: Calculate the height $h_{ik}, \forall i \in V, k \in \mathcal{K}$:

$$h_{ik} = \sum_{j \in \delta^-(i)} f_{ji,k} - \sum_{j \in \delta^+(i)} f_{ij,k} + \hat{\Delta}_{ik}. \quad (48)$$

2: Calculate the congestion function $\psi_{ij}, \forall (i, j) \in E$:

$$\psi_{ij} = \begin{cases} 0 & \text{if } \sum_{k \in \mathcal{K}} f_{ij,k} \leq u_{ij} \\ \sum_{k \in \mathcal{K}} f_{ij,k} - u_{ij} & \text{if } \sum_{k \in \mathcal{K}} f_{ij,k} > u_{ij}. \end{cases} \quad (49)$$

3: Calculate the potential difference $\phi_{ij,k}, \forall (i, j) \in E, k \in \mathcal{K}$:

$$\phi_{ij,k} = h_{ik} - h_{jk} - \psi_{ij}. \quad (50)$$

Remark 11 By Equation (9), the negative gradient of the objective function of Programming (7) is exactly the same as the potential difference. Therefore, Algorithm 3 is a gradient projection method for Programming (7). For the convergence of gradient projection method, see [13, 29].

Remark 12 The step-size rule in Algorithm 3 is to start with an initial guess for the step size and then, if some condition is not satisfied, reduce the step size successively by multiplication with a constant scalar until it meets the condition. The reducing rule for β_l is of Armijo type [30, 13].

Remark 13 Too small step size β_l will lead to slow convergence. If $\omega^l \leq \mu$, we would enlarge β_l by $\beta_l := \beta_l * 1.5$ (see [28]).

4 Further Discussion on Optimality Condition

As commonly understood, directed edges correspond to pipes of the physical network and vertices are pipe junctions. In practice, for any physical network, it is impossible for the flow in a pipe to exceed its capacity. In this section we would discuss the optimality condition for multicommodity flow problem when only relaxing the flow conservation constraints.

The basic formulation is as follows:

$$\begin{aligned} \min \quad & z = \sum_{i,k} \int_0^{\Delta_{ik}} h_{ik}(\omega) d\omega \\ \text{s.t.} \quad & \sum_{k \in \mathcal{K}} f_{ij,k} \leq u_{ij}, \forall (i, j) \in E \\ & f_{ij,k} \geq 0, \forall k \in \mathcal{K}, (i, j) \in E \end{aligned} \quad (51)$$

where h_{ik} is the height function and Δ_{ik} defined as Expression (2).

In the programming above, the objective function is the sum of the integrals of the vertex height function. The flow conservation constraints are relaxed but the capacity constraints are maintained. Obviously, the feasible region of Expression (1) is not empty if and only if the minimum value of the objective function of Programming (51) is zero; the feasible region of Expression (1) is empty if and only if the minimum value of the objective function of Programming (51) is greater than zero. Since Programming (51) is convex, the solution f^* of Programming (51) is optimal if and only if it satisfies the following Karush-Kuhn-Tucker conditions:

Stationarity

$$-\frac{\partial z}{\partial f_{ij,k}} = -\mu_{ij,k} + \gamma_{ij}, \forall k \in \mathcal{K}, (i, j) \in E$$

Primal feasibility

$$\begin{aligned} \sum_{k \in \mathcal{K}} f_{ij,k} &\leq u_{ij}, \forall (i, j) \in E \\ -f_{ij,k} &\leq 0, \forall k \in \mathcal{K}, (i, j) \in E \end{aligned}$$

(52)

Dual feasibility

$$\begin{aligned} \mu_{ij,k} &\geq 0, \forall k \in \mathcal{K}, (i, j) \in E \\ \gamma_{ij} &\geq 0, \forall (i, j) \in E \end{aligned}$$

Complementary slackness

$$\begin{aligned} \mu_{ij,k} f_{ij,k} &= 0, \forall k \in \mathcal{K}, (i, j) \in E \\ \gamma_{ij} \left(\sum_{k \in \mathcal{K}} f_{ij,k} - u_{ij} \right) &= 0, \forall (i, j) \in E. \end{aligned}$$

Obviously,

$$\begin{aligned} \frac{\partial z}{\partial f_{ij,k}} &= h_{ik}(\Delta_{ik}) \frac{\partial \Delta_{ik}}{\partial f_{ij,k}} + h_{jk}(\Delta_{jk}) \frac{\partial \Delta_{jk}}{\partial f_{ij,k}} \\ &= (-h_{ik}(\Delta_{ik})) + (h_{jk}(\Delta_{jk})) \\ &= h_{jk} - h_{ik}. \end{aligned}$$

(53)

Substituting the expression above into Stationarity expression in KKT conditions,

$$\gamma_{ij} = \mu_{ij,k} + h_{ik} - h_{jk}, \forall k \in \mathcal{K}, (i, j) \in E.$$

If edge (i, j) is not saturated, by complementary slackness, $\gamma_{ij} = 0$. If edge (i, j) is saturated, $\gamma_{ij} \geq 0$. Intuitively, γ_{ij} could be interpreted as the *pressure* on pipes generated by the fluid. If edge (i, j) is not saturated, the *pipe pressure* is zero; otherwise it is greater than or equal to zero.

For any used edge of commodity k , i.e. $f_{ij,k} > 0$, by complementary slackness $\mu_{ij,k} f_{ij,k} = 0$ in KKT conditions, we have $\mu_{ij,k} = 0$. Therefore,

$$\gamma_{ij} = h_{ik} - h_{jk}, \forall k \in \mathcal{K}, (i, j) \in E.$$

That is, the height difference between vertex i and vertex j for commodity k is equal to the pipe pressure γ_{ij} of edge (i, j) .

For any unused edge of commodity k , due to $\mu_{ij,k} \geq 0$, we have

$$\gamma_{ij} = \mu_{ij,k} + h_{ik} - h_{jk} \geq h_{ik} - h_{jk}, \forall k \in \mathcal{K}, (i, j) \in E$$

That is, the height difference between vertex i and vertex j for commodity k is less than or equal to the pipe pressure γ_{ij} .

Note that the pipe pressure γ_{ij} is zero when the edge is unsaturated. Therefore, the optimality conditions can be concluded as follows:

- (i) for any used edge of commodity k , if it is unsaturated, the height difference between vertex i and vertex j for commodity k is zero;
- (ii) for any used edge of commodity k , if it is saturated, the height difference between vertex i and vertex j for commodity k is equal to the pipe pressure γ_{ij} ;
- (iii) for any unused edge of commodity k , the height difference between vertex i and vertex j for commodity k is less than or equal to the pipe pressure γ_{ij} .

5 Conclusion

In this paper, we present a localized approach to solve the multicommodity flow problem. By relaxing both the capacity constraints and flow conservation constraints, we define a *congestion function* ψ for each edge and a *height*

function h for each vertex and commodity. Utilizing the congestion function ψ and the height function h , we propose a simple and intuitive optimality condition for multicommodity flow problem. Furthermore, by the optimal condition, we give an algorithm based on potential difference reduction approach, which works in a localized manner without depending on any shortest path or augmenting path. Overall, this work contributes a novel perspective on solving the multicommodity flow problem, which would help people better understand flow problems and may inspire further research in this field.

References

- [1] Goldstein, Alan A. "Convex programming in Hilbert space." *Bulletin of the American Mathematical Society* 70.5 (1964): 709-710.
- [2] Levitin, Evgeny S., and Boris T. Polyak. "Constrained minimization methods." *USSR Computational mathematics and mathematical physics* 6.5 (1966): 1-50.
- [3] Ford, Lester Randolph, and Delbert Ray Fulkerson. "Flows in networks." *Flows in Networks*. Princeton university press, 2015.
- [4] Hu, T. Chiang. "Multi-commodity network flows." *Operations research* 11.3 (1963): 344-360.
- [5] Wang, I-Lin. "Multicommodity network flows: A survey, Part I: Applications and Formulations." *International Journal of Operations Research* 15.4 (2018): 145-153.
- [6] Wang, I-Lin. "Multicommodity network flows: A survey, part II: Solution methods." *International Journal of Operations Research* 15.4 (2018): 155-173.
- [7] Salimifard, Khodakaram, and Sara Bigharaz. "The multicommodity network flow problem: state of the art classification, applications, and solution methods." *Operational Research* 22.1 (2022): 1-47.
- [8] Goldberg, Andrew V., and Robert E. Tarjan. "A new approach to the maximum-flow problem." *Journal of the ACM (JACM)* 35.4 (1988): 921-940.
- [9] Liu, Pengfei. "A Combinatorial Algorithm for the Multi-commodity Flow Problem." *arXiv preprint arXiv:1904.09397* (2019).
- [10] Bui, Minh N. "A decomposition method for solving multicommodity network equilibria." *Operations Research Letters* 50.1 (2022): 40-44.
- [11] Yu, Yue, et al. "Variable demand and multi-commodity flow in Markovian network equilibrium." *Automatica* 140 (2022): 110224.
- [12] Boyd, Stephen, Stephen P. Boyd, and Lieven Vandenbergh. *Convex optimization*. Cambridge university press, 2004.
- [13] Bertsekas, Dimitri P. "On the Goldstein-Levitin-Polyak gradient projection method." *IEEE Transactions on automatic control* 21.2 (1976): 174-184.
- [14] Kamath, Anil, and Omri Palmon. "Improved Interior Point Algorithms for Exact and Approximate Solution of Multicommodity Flow Problems." *SODA*. Vol. 95. 1995.
- [15] Kennington, Jeff L. "Solving multicommodity transportation problems using a primal partitioning simplex technique." *Naval Research Logistics Quarterly* 24.2 (1977): 309-325.
- [16] Geoffrion, Arthur M. "Primal resource-directive approaches for optimizing nonlinear decomposable systems." *Operations Research* 18.3 (1970): 375-403.
- [17] Shetty, Bala, and R. Muthukrishnan. "A parallel projection for the multicommodity network model." *Journal of the Operational Research Society* 41.9 (1990): 837-842.
- [18] Frangioni, Antonio, and Giorgio Gallo. "A bundle type dual-ascent approach to linear multicommodity min-cost flow problems." *INFORMS Journal on Computing* 11.4 (1999): 370-393.
- [19] Zenios, Stavros A., Mustafa C. Pinar, and Ron S. Dembo. "A smooth penalty function algorithm for network-structured problems." *European Journal of Operational Research* 83.1 (1995): 220-236.
- [20] Goffin, J-L., et al. "Solving nonlinear multicommodity flow problems by the analytic center cutting plane method." *Mathematical programming* 76.1 (1997): 131-154.
- [21] Kapoor, Sanjiv, and Pravin M. Vaidya. "Speeding up Karmarkar's algorithm for multicommodity flows." *Mathematical programming* 73.1 (1996): 111-127.

- [22] Shahrokhi, Farhad, and David W. Matula. "The maximum concurrent flow problem." *Journal of the ACM (JACM)* 37.2 (1990): 318-334.
- [23] Leighton, Tom, et al. "Fast approximation algorithms for multicommodity flow problems." *Journal of Computer and System Sciences* 50.2 (1995): 228-243.
- [24] Garg, Naveen, and Jochen Könemann. "Faster and simpler algorithms for multicommodity flow and other fractional packing problems." *SIAM Journal on Computing* 37.2 (2007): 630-652.
- [25] Awerbuch, Baruch, and Tom Leighton. "A simple local-control approximation algorithm for multicommodity flow." *Proceedings of 1993 IEEE 34th Annual Foundations of Computer Science*. IEEE, 1993.
- [26] Awerbuch, Baruch, and Tom Leighton. "Improved approximation algorithms for the multi-commodity flow problem and local competitive routing in dynamic networks." *Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*. 1994.
- [27] Goldberg, Andrew, and Robert Tarjan. "Solving minimum-cost flow problems by successive approximation." *Proceedings of the nineteenth annual ACM symposium on Theory of computing*. 1987.
- [28] He, B. S., and Li-Zhi Liao. "Improvements of some projection methods for monotone nonlinear variational inequalities." *Journal of Optimization Theory and applications* 112.1 (2002): 111-128.
- [29] Korpelevich, G. M. "An extragradient method for finding saddle points and for other problems, *Ekonomika i Matematicheskie Metody*, 12 (1976), 747-756." Search in.
- [30] Armijo, Larry. "Minimization of functions having Lipschitz continuous first partial derivatives." *Pacific Journal of mathematics* 16.1 (1966): 1-3.
- [31] Johansson, Björn, et al. "The application of an oblique-projected Landweber method to a model of supervised learning." *Mathematical and computer modelling* 43.7-8 (2006): 892-909.
- [32] Moura, Leonardo FS, Luciano P. Gasparly, and Luciana S. Buriol. "A branch-and-price algorithm for the single-path virtual network embedding problem." *Networks* 71.3 (2018): 188-208.
- [33] Brun, Olivier, Balakrishna Prabhu, and Josselin Vallet. "A penalized best-response algorithm for nonlinear single-path routing problems." *Networks* 69.1 (2017): 52-66.
- [34] Hochbaum, Dorit S. "The pseudoflow algorithm: A new algorithm for the maximum-flow problem." *Operations research* 56.4 (2008): 992-1000
- [35] Chandran, Bala G., and Dorit S. Hochbaum. "A computational study of the pseudoflow and push-relabel algorithms for the maximum flow problem." *Operations research* 57.2 (2009): 358-376.
- [36] Fishbain, Barak, Dorit S. Hochbaum, and Stefan Mueller. "Competitive analysis of minimum-cut maximum flow algorithms in vision problems." *arXiv preprint arXiv:1007.4531* (2010).
- [37] Spielman, Daniel A., and Shang-Hua Teng. "Nearly-linear time algorithms for graph partitioning, graph sparsification, and solving linear systems." *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. 2004.
- [38] Daitch, Samuel I., and Daniel A. Spielman. "Faster approximate lossy generalized flow via interior point algorithms." *Proceedings of the fortieth annual ACM symposium on Theory of computing*. 2008.
- [39] Christiano, Paul, et al. "Electrical flows, laplacian systems, and faster approximation of maximum flow in undirected graphs." *Proceedings of the forty-third annual ACM symposium on Theory of computing*. 2011.
- [40] Kelner, Jonathan A., et al. "An almost-linear-time algorithm for approximate max flow in undirected graphs, and its multicommodity generalizations." *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*. Society for Industrial and Applied Mathematics, 2014.
- [41] Axiotis, Kyriakos, Aleksander Mądry, and Adrian Vladu. "Faster Sparse Minimum Cost Flow by Electrical Flow Localization." *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2022.
- [42] Gao, Yu, Yang P. Liu, and Richard Peng. "Fully dynamic electrical flows: Sparse maxflow faster than goldberg-rao." *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2022.
- [43] Chen, Li, et al. "Maximum flow and minimum-cost flow in almost-linear time." *arXiv preprint arXiv:2203.00671* (2022).
- [44] van Den Brand, Jan, and Daniel J. Zhang. "Faster high accuracy multi-commodity flow from single-commodity techniques." *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2023.
- [45] Kyng, Rasmus, and Peng Zhang. "Hardness results for structured linear systems." *SIAM Journal on Computing* 49.4 (2020): FOCS17-280.

-
- [46] Itai, Alon. "Two-commodity flow." *Journal of the ACM (JACM)* 25.4 (1978): 596-611.
- [47] Ding, Ming, Rasmus Kyng, and Peng Zhang. "Two-commodity flow is equivalent to linear programming under nearly-linear time reductions." *arXiv preprint arXiv:2201.11587* (2022).
- [48] Van Den Brand, Jan, et al. "A deterministic almost-linear time algorithm for minimum-cost flow." *2023 IEEE 64th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, 2023.