# Primal-Dual Proximal Point Algorithm for Multicommodity Network Flow Problems

Satoru IBARAKI* and Masao FUKUSHIMA**

* Department of Applied Mathematics and Physics,
Faculty of Engineering, Kyoto University,
Kyoto 606, Japan

** Graduate School of Information Science,
Advanced Institute of Science and Technology, Nara,
Takayama, Ikoma, Nara 630-01, Japan

# 1. Introduction

The multicommodity network flow problem is an important class of network optimization problems, in which arcs are shared by several commodities and the flow of each commodity must be conserved at every node. The applications can be found in such diverse areas as data communication systems, transportation systems of crops, goods or vehicles and production lines of resources and products.

Let $G = (\mathcal{N}, \mathcal{A})$ be a directed graph, where $\mathcal{N} = \{1, 2, \ldots, m\}$ is the node set and $\mathcal{A} = \{1, 2, \ldots, n\}$ is the arc set. We consider a multicommodity network flow problem on $G$ having $K$ distinct commodities. To formulate the problem, we introduce some notations:

$x_{kj} \in R$: the flow of commodity $k$ on arc $j$,

$x_k = (x_{k1}, x_{k2}, \ldots, x_{kn})^T \in R^n$: the vector of flows of commodity $k$,

$x = (x_1, x_2, \ldots, x_n) \in R^{nK}$: the vector of flows for all commodities,

$y_j \in R$: the total flow on arc $j$, i.e. $y_j = \sum_{k=1}^{K} x_{kj}$,

$y = (y_1, y_2, \ldots, y_n)^T \in R^n$: the vector of total flow,

$f_k : R^n \to (-\infty, +\infty]$: the cost function associated with flow of commodity $k$,

$g : R^n \to (-\infty, +\infty]$: the cost function associated with total flow,

$E \in R^{m \times n}$: the node-arc incidence matrix of graph $G$,

$b_{ki} \in R$: the requirement for commodity $k$ at node $i$,

$b_k = (b_{k1}, b_{k2}, \ldots, b_{km})^T \in R^m$: the vector of requirements for commodity $k$.

Node $i$ is called a supply point for commodity $k$ if $b_{ki} > 0$, a demand point for commodity $k$ if $b_{ki} < 0$, and a transshipment point for commodity $k$ if $b_{ki} = 0$. We assume that, for each commodity $k$, the total demand equals the total supply, i.e. $\sum_{i=1}^{m} b_{ki} = 0$. We also assume that the cost functions $f_k$ and $g$ are closed proper convex.

Now we formulate the multicommodity network flow problem as follows:

$$\begin{aligned} \text{P: minimize} \quad & \sum_{k=1}^{K} f_k(x_k) + g(y) \\ \text{subject to} \quad & Ex_k = b_k, \quad k = 1, 2, \ldots, K, \qquad (1.1) \\ & y = \sum_{k=1}^{K} x_k. \qquad (1.2) \end{aligned}$$

Constraints (1.1) are the flow conservation equations for individual commodities, whereas (1.2) is a coupling constraint that links together the flows of all commodities. In particular, the latter constraint hampers straightforward decomposition of problem P into single commodity subproblems.

It is important to note that problem P explicitly contains equality constraints only. Inequality constraints such as arc capacity constraints may be regarded as a part of the cost functions $f_k$ and $g$, as shown by the following two important examples of multicommodity network flow problems. In these examples, we assume that the functions $f_k$ and $g$ are separable, i.e.

$$f_k(x_k) = \sum_{j=1}^{n} f_{kj}(x_{kj}), \quad \text{for all } k = 1, 2, \ldots, K \tag{1.3}$$

and

$$g(y) = \sum_{j=1}^{n} g_j(y_j). \tag{1.4}$$

This situation is often seen in practical applications and belongs to the important class of problems. In particular, when problem P has this assumption, the proposed algorithmcan be applied effectively. We consider the case that the cost function of P is separable in Section 3.

*Example* 1 (Linear multicommodity network flow problem [1, 2, 15]):

Let

$$f_{kj}(x_{kj}) = \begin{cases} a_{kj}x_{kj} & \text{if } 0 \le x_{kj} \le c_{kj}, \\ +\infty & \text{otherwise,} \end{cases}$$

for all $k = 1, 2, \ldots, K, j = 1, 2, \ldots, n$, and

$$g_j(y_j) = \begin{cases} 0 & \text{if } 0 \le y_j \le d_j, \\ +\infty & \text{otherwise,} \end{cases}$$

for all $j = 1, 2, \ldots, n$. Then problem P is rewritten as

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K}\sum_{j=1}^{n} a_{kj}x_{kj} \\ \text{subject to} \quad & Ex_k = b_k, \quad k = 1, 2, \ldots, K, \\ & \sum_{k=1}^{K} x_{kj} \le d_j, \quad j = 1, 2, \ldots, n, \\ & 0 \le x_{kj} \le c_{kj}, \quad k = 1, 2, \ldots, K, \quad j = 1, 2, \ldots, n. \end{aligned}$$

In this problem, $c_{kj}$ is the capacity for the flow of commodity $k$ on arc $j$, while $d_j$ is the capacity for the total flow on arc $j$.

*Example* 2 (Traffic assignment problem [6, 18]):

Let

$$f_{kj}(x_{kj}) = \begin{cases} 0 & \text{if } 0 \le x_{kj} \le c_{kj}, \\ +\infty & \text{otherwise,} \end{cases}$$

for all $k = 1, 2, \ldots, K, j = 1, 2, \ldots, n$. Let the functions $g_j$ be given by

$$g_j(y_j) = \int_0^{y_j} \tilde{g}_j(t)dt,$$

where $\tilde{g}_j : R \to [0, +\infty)$ is an arc travel cost function which is nonnegative, increasing and convex. Then the traffic assignment problem may be formulated as

$$\text{minimize} \quad \sum_{j=1}^{n} g_j(y_j)$$

$$
\begin{aligned}
\text{subject to} \quad & Ex_k = b_k, \quad k = 1, 2, \ldots, K, \\
& y_j = \sum_{k=1}^{K} x_{kj}, \quad j = 1, 2, \ldots, n, \\
& 0 \leq x_{kj} \leq c_{kj}, \quad k = 1, 2, \ldots, K, \quad j = 1, 2, \ldots, n.
\end{aligned}
$$

In this problem, it is often assumed that $c_{kj} = +\infty$ for all $k$ and $j$, in which case the Kuhn-Tucker conditions for the problem represent the well-known user optimal principle in a congested traffic network [6].

Various methods have been proposed to solve nonlinear multicommodity network flow problems, including the traffic assignment problem. For example, linear approximation methods [4, 5], Frank-Wolfe method [7, 18] and gradient projection method [3] belong to the class of algorithms that directly exploit the advantage of the network structure. On the other hand, algorithms based on the dual approach have also been studied extensively in conjunction with various optimization techniques, e.g. a subgradient method [8], descent methods [9, 11] and relaxation methods [19, 24]. Note that all of the above mentioned methods except [11] are concerned with the Lagrangian dual problem obtained by relaxing both coupling constraints and flow conservation equations, while the method of [11] utilizes another Lagrangian dual which is defined by relaxing coupling constraints only.

The purpose of this paper is to present a primal-dual proximal point algorithm for the convex multicommodity network flow problem P. The proximal point algorithm and its variants have been extensively studied in the literature. In particular, the primal-dual proximal point algorithm is considered in [13, 21, 23, 25]. The algorithm proposed in this paper is closely related the one presented by the authors in [13], which is tailored to solve linearly constrained convex programming problems. The method of [13], however, is concerned with the ordinary Lagrangian dual problem obtained by relaxing all linear constraints, while the method to be proposed in this paper deals with the Lagrangian function formed by relaxing the coupling constraints (1.2) only. Note that the dual optimality is attained when the primal feasibility is satisfied, i.e. the relaxed constraints of the primal problem are satisfied. However, since we cannot proceed the iteration of the algorithm infinitely, a solution obtained in practice usually satisfies the feasibility conditions only approximately. Nevertheless, an approximate solution produced by the proposed method always satisfies the flow conservation equations (1.1) for all commodities. This property often turns out to be very useful in practical applications.

In Section 2, we present the primal-dual proximal point algorithm based on [13] and discuss its convergence property. The convergence results do not depend on the separability of the objective function of problem P. In Section 3, we apply the algorithm to the separable problem and specify the details of the algorithm under this assumption. In Section 4, we provide some results of numerical experiments. Finally we give some concluding remarks in Section 5.

## 2. Primal-Dual Proximal Point Algorithm

Let $\hat{f}_k : R^n \to (-\infty, +\infty]$ be the function defined by

$$\hat{f}_k(x_k) = \begin{cases} f_k(x_k) & \text{if } Ex_k = b_k, \\ +\infty & \text{otherwise.} \end{cases} \tag{2.1}$$

Then problem P is reformulated as

$$\begin{aligned} &\text{minimize} && \sum_{k=1}^{K} \hat{f}_k(x_k) + g(y) \\ &\text{subject to} && y = \sum_{k=1}^{K} x_k. \end{aligned}$$

For this problem, let $p$ denote the vector of Lagrange multipliers and define the Lagrangian $L : R^{nK} \times R^n \times R^n \to (-\infty, +\infty]$ by

$$L(x, y, p) = \sum_{k=1}^{K} \hat{f}_k(x_k) + g(y) - \langle p, \sum_{k=1}^{K} x_k - y \rangle, \tag{2.2}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product.

Now we may define the dual of problem P as follows:

D: maximize $\psi(p)$,

where the dual objective function $\psi$ is given by

$$\psi(p) = \inf_{x \in R^{nK}, y \in R^n} L(x, y, p).$$

For $(\bar{x}, \bar{y})$ to solve P and $\bar{p}$ to solve D, it is necessary and sufficient that the following Kuhn-Tucker conditions hold:

$$\bar{p} \in \partial \hat{f}_k(\bar{x}_k), \quad k = 1, 2, \ldots, K,$$

$$-\bar{p} \in \partial g(\bar{y}),$$

$$\bar{y} = \sum_{k=1}^{K} \bar{x}_k,$$

where $\partial \hat{f}_k$ and $\partial g$ denote the subdifferential operators of $\hat{f}_k$ and $g$, respectively. These conditions imply that $(\bar{x}, \bar{y}, \bar{p})$ is a saddle point of the Lagrangian $L$. When problem P has an optimal solution $(\bar{x}, \bar{y})$, the existence of a multiplier vector $\bar{p}$ satisfying the above Kuhn-Tucker conditions is guaranteed, provided that P is strongly consistent, i.e., there is at least one feasible solution in the relative interior of the effective domain of the objective function [20, p. 300].

The primal-dual proximal point algorithm [13, 21, 23, 25] is an iterative method that generates a sequence of points converging to a Kuhn-Tucker point of the problem. Each iteration consists of finding an (approximate) saddle point of a convex-concave function, which is obtained by augmenting the Lagrangian by quadratic terms of both primal and dual variables. For problem P, the augmented Lagrangian $L^{(\mu)} : R^{nK} \times R^n \times R^n \to (-\infty, +\infty]$ at the $\mu$-th iteration is given by

$$L^{(\mu)}(x,y,p) = L(x,y,p) + \frac{1}{2\gamma^{(\mu)}}|x - x^{(\mu)}|^2 + \frac{1}{2\gamma^{(\mu)}}|y - y^{(\mu)}|^2 - \frac{1}{2\gamma^{(\mu)}}|p - p^{(\mu)}|^2, \tag{2.3}$$

where $\gamma^{(\mu)}$ is a positive constant and $|\cdot|$ denotes the Euclidean norm. Note that $L^{(\mu)}$ is strongly convex and strongly concave with modulus $\frac{1}{\gamma^{(\mu)}}$ in $(x,y)$ and in $p$, respectively.

There are two strategies to find an approximate saddle point of $L^{(\mu)}$: One is that we first maximize $L^{(\mu)}$ in $p$ and then approximately minimize the resulting function of $(x,y)$ [21, 23], i.e.

$$(x^{(\mu+1)}, y^{(\mu+1)}, p^{(\mu+1)}) \approx \arg \min_{x \in R^{nK}, y \in R^n} \left\{ \max_{p \in R^n} L^{(\mu)}(x,y,p) \right\}, \tag{2.4}$$

and the other we first minimize $L^{(\mu)}$ in $(x,y)$ and then approximately maximize the resulting function of $p$ [13], i.e.

$$(x^{(\mu+1)}, y^{(\mu+1)}, p^{(\mu+1)}) \approx \arg \max_{p \in R^n} \left\{ \min_{x \in R^{nK}, y \in R^n} L^{(\mu)}(x,y,p) \right\}. \tag{2.5}$$

In the proposed algorithm, we adopt the latter strategy to find an approximate saddle point of $L^{(\mu)}$. The difference between the two strategies will be clarified in the following.

In order to explain (2.5) in more detail, we define

$$(x(p), y(p)) = \arg \min_{x \in R^{nK}, y \in R^n} L^{(\mu)}(x,y,p) \tag{2.6}$$

and

$$\begin{aligned} \psi^{(\mu)}(p) &= L^{(\mu)}(x(p), y(p), p) \\ &= \min_{x \in R^{nK}, y \in R^n} L^{(\mu)}(x,y,p). \end{aligned} \tag{2.7}$$

Note that $(x(p), y(p))$ is the exact minimizer of $L^{(\mu)}$ with $p$ fixed, which uniquely exists because of the strong convexity of $L^{(\mu)}$ in $(x,y)$. In this paper, we assume that such exact minimizer can actually be computed. Then, by the definition (2.1) of $\hat{f}_k$, the flow conservation equation for each commodity is always satisfied by $(x(p), y(p))$. Note also that $\psi^{(\mu)}$ is a closed concave function [20, Th. 12.1]. Since $(x(p), y(p))$ is uniquely obtained, so the function $\psi^{(\mu)}$ is continuously differentiable [17, §8.5 Cor. 1] and its gradient is given by

$$\nabla \psi^{(\mu)}(p) = y(p) - \sum_{k=1}^{K} x_k(p) - \frac{1}{\gamma^{(\mu)}}(p - p^{(\mu)}). \tag{2.8}$$

Using (2.6) and (2.7), the formula (2.5) can be written as

$$p^{(\mu+1)} \approx \arg \max_{p \in R^n} \psi^{(\mu)}(p) \tag{2.9}$$

and

$$\left(x^{(\mu+1)}, y^{(\mu+1)}\right) = \left(x(p^{(\mu+1)}), y(p^{(\mu+1)})\right). \tag{2.10}$$

The above maximization of $\psi^{(\mu)}$ in (2.9) is a nonlinear unconstrained smooth optimization problem, which may usually be solved iteratively. Since the function $\psi^{(\mu)}$ is continuously differentiable, we can use a gradient-based algorithm like a quasi-Newton method. The gradient $\nabla\psi^{(\mu)}(p)$ is obtained as a by-product when we compute the value of $\psi^{(\mu)}(p)$. This implies that $(x(p), y(p))$ of (2.6) is calculated for every $p$ in the course of iterations to maximize $\psi^{(\mu)}$. For the iterations to maximize $\psi^{(\mu)}$, we may adopt one of the following two termination criteria:

$$|\nabla\psi^{(\mu)}(p^{(\mu+1)})| \leq \frac{\epsilon^{(\mu)}}{\gamma^{(\mu)}}, \tag{2.11}$$

where $\epsilon^{(\mu)}$ are positive constants such that

$$\sum_{\mu=0}^{\infty} \epsilon^{(\mu)} < \infty, \tag{2.12}$$

and

$$|\nabla\psi^{(\mu)}(p^{(\mu+1)})| \leq \frac{\delta^{(\mu)}}{\gamma^{(\mu)}}|(x(p^{(\mu+1)}), y(p^{(\mu+1)}), p^{(\mu+1)}) - (x^{(\mu)}, y^{(\mu)}, p^{(\mu)})|, \tag{2.13}$$

where $\delta^{(\mu)}$ are positive constants such that

$$\sum_{\mu=0}^{\infty} \delta^{(\mu)} < \infty. \tag{2.14}$$

Note that checking these criteria requires evaluating $(x^{(\mu+1)}, y^{(\mu+1)})$, because we need to know the values of $x^{(\mu+1)}$ and $y^{(\mu+1)}$ when we evaluate $\nabla\psi^{(\mu)}(p^{(\mu+1)})$ (see (2.8)).

By transferring the results in [13, 21] to the context of the algorithm based on the formula (2.5), or equivalently (2.9)-(2.10), we may obtain the following convergence theorem. Recall that the mapping $T^{-1}$ is said to be Lipschitz continuous at the origin with modulus $\alpha \geq 0$, if $T^{-1}(0)$ is single-valued and there exists $\tau > 0$ such that $|z - T^{-1}(0)| \leq a|w|$ for all $z \in T^{-1}(w)$ and $|w| \leq \tau$ [21].

**Theorem.** *Suppose that problem P is strongly consistent and has at least one optimal solution.*
*(a) If the algorithm is executed under criterion* (2.11) *with a sequence* $\{\gamma^{(\mu)}\}$ *bounded away from zero, then the sequence* $\{(x^{(\mu)}, y^{(\mu)}, p^{(\mu)})\}$ *generated by (2.5) is bounded and converges to* $(\bar{x}, \bar{y}, \bar{p})$, *where* $\bar{x}$ *and* $\bar{y}$ *are optimal for problem P and* $\bar{p}$ *is optimal for problem D.*
*(b) Let* $T_L : R^{nK} \times R^n \times R^n \to R^{nK} \times R^n \times R^n$ *be multifunction (point-to-set mapping) given by*

$$\begin{aligned} T_L(x, y, p) &= \{(u, v, w) \,|u_k \in \partial_{x_k} L(x, y, p), \forall k, v \in \partial_y L(x, y, p), w \in -\partial_p L(x, y, p)\} \\ &= \left\{(u, v, w) \,\middle|\, u_k \in \partial f_k(x_k) - p, \forall k, v \in \partial g(y) - p, w = \sum_{k=1}^{K} x_k - y\right\}. \end{aligned}$$

*Assume that* $T_L^{-1}$ *is Lipschitz continuous at the origin with modulus* $\alpha \geq 0$. *If the algorithm is executed under criterion* (2.13) *with a sequence* $\{\gamma^{(\mu)}\}$ *nondecreasing, then the sequence* $\{(x^{(\mu)}, y^{(\mu)}, p^{(\mu)})\}$ *is bounded and converges to* $(\bar{x}, \bar{y}, \bar{p})$, *where* $(\bar{x}, \bar{y})$ *and* $\bar{p}$ *are the unique optimal solutions for P and D, respectively. In addition, there exists an integer* $\bar{\mu}$ *such that*

$$|(x^{(\mu+1)}, y^{(\mu+1)}, p^{(\mu+1)}) - (\bar{x}, \bar{y}, \bar{p})| \leq \theta^{(\mu)} |(x^{(\mu)}, y^{(\mu)}, p^{(\mu)}) - (\bar{x}, \bar{y}, \bar{p})|, \quad \forall \mu \geq \bar{\mu},$$

*where*

$$\theta^{(\mu)} = \{\alpha(\alpha^2 + \gamma^{(\mu)})^{-1/2} + \delta^{(\mu)}\}(1 - \delta^{(\mu)})^{-1}.$$

## 3. Algorithm for Separable Problems

In this section, we assume that the cost function of problem P is separable, that is, the functions $f_k$ and $g$ in problem P are given by

$$f_k(x_k) = \sum_{j=1}^{n} f_{kj}(x_{kj}), \quad \text{for all } k = 1, 2, \ldots, K \tag{3.1}$$

and

$$g(y) = \sum_{j=1}^{n} g_j(y_j), \tag{3.2}$$

respectively. (See (1.3) and (1.4).) We assume that the functions $f_{kj}$ and $g_j$ are closed proper convex.

We specify the detail of the proposed algorithm, which exploits the special structure of the problem. From the separability of $f_k$ and $g$, it follows that the function $L^{(\mu)}$ in (2.3) is separable in $x_k$ and $y_j$ when $p$ is fixed. Therefore, the minimization of $L^{(\mu)}$ appearing in (2.6) is carried out separately with respect to $x_k$ and $y_j$. Specifically, $(x(p), y(p))$ in (2.6) is evaluated as follows:

$$x_k(p) = \arg\min_{x_k \in R^n} \left\{ \sum_{j=1}^{n} \left\{ f_{kj}(x_{kj}) + \frac{1}{2\gamma^{(\mu)}}(x_{kj} - x_{kj}^{(\mu)})^2 - p_j x_{kj} \right\} \middle| \, Ex_k = b_k \right\}, \tag{3.3}$$

for all $k = 1, 2, \ldots, K$, and

$$y_j(p_j) = \arg\min_{y_j \in R} \left\{ g_j(y_j) + \frac{1}{2\gamma^{(\mu)}}(y_j - y_j^{(\mu)})^2 + p_j y_j \right\}, \tag{3.4}$$

for all $j = 1, 2, \ldots, n$.

In (3.3), the computation of $x_k(p)$ amounts to solving a single commodity network flow problem whose objective function is separable and strongly convex. In practice, a variety of methods are available to solve such problems [12, 16, 22]. In (3.4), the computation of $y_j(p_j)$ becomes a univariate minimization problem with a strongly convex objective function. The minimizer $y_j(p_j)$ may often be expressed in a closed form, or it can at least be computed accurately enough by using an appropriate one-dimensional optimization technique such as binary section method, golden section method and quasi-Newton method.

Recall that the maximization in (2.9) is a differentiable unconstrained optimization problem whose objective function $\psi^{(\mu)}(p)$ and its gradient $\nabla\psi^{(\mu)}(p)$ may be computed from $x(p)$ and $y(p)$,

which are obtained by the minimization on the right-hand side of (3.3) and (3.4). Since the latter minimization problems are in general solved iteratively, the proposed algorithm may have a triple-loop structure. Namely, the outmost loop is the iteration of the primal-dual proximal point algorithm, the middle one is the iteration of maximizing $\psi^{(\mu)}$ to compute $p^{(\mu+1)}$ by (2.9), and the inmost one is the iteration to compute $x(p)$ and $y(p)$.

# 4. Numerical experiments

In this section, we present some numerical results for the proximal point algorithm with criterion (2.13) for the approximate solution of subproblem (2.9). In particular, we examine how performance of the algorithm is affected by varying parameter values. All codes were written in FORTRAN 77 and run in double precision on a FACOM M1800.

## 4.1. Test problems

All test problems are multicommodity network flow problems with separable costs:

$$\begin{aligned} \text{minimize} \quad & \sum_{k=1}^{K}\sum_{j=1}^{n} f_{kj}(x_{kj}) + \sum_{j=1}^{n} g_j(y_j) \\ \text{subject to} \quad & Ex_k = b_k, \quad k = 1, 2, \ldots, K, \\ & y = \sum_{k=1}^{K} x_k, \end{aligned}$$

where the components of $b_k$ are integers randomly generated from the interval $[1, 10]$ and the functions $f_{kj}$ and $g_j$ are respectively defined by

$$f_{kj}(x_{kj}) = \begin{cases} a_{kj}x_{kj} + \frac{1}{2}q_{kj}x_{kj}^2 & \text{if } 0 \le x_{kj} \le c_{kj}, \\ +\infty & \text{otherwise} \end{cases}$$

for $k = 1, 2, \ldots, K, \quad j = 1, 2, \ldots, n$, and

$$g_j(y_j) = a_{0j}y_j + \frac{1}{2}q_{0j}y_j^2$$

for $j = 1, 2, \ldots, n$, where $a_{kj}$, $q_{kj}$ and $c_{kj}$ are integers randomly generated from the intervals $[1, \alpha]$, $[1, \alpha]$ and $[1, 10]$, respectively, and $\alpha$ is a positive integer. We shall denote this problem by $P(m, n, K, \alpha)$. Note that $\alpha$ represents the condition of the problem, that is, the larger the value of parameter $\alpha$ is, the worse the condition of the problem.

## 4.2. Solution of subproblems

Computing $p^{(\mu+1)}$ by (2.9) amounts to maximizing the continuously differentiable concave function $\psi^{(\mu)}$ approximately. To solve this unconstrained optimization problem, we used a quasi-Newton (BFGS) method coded by Fukushima [14, pp. 141-166]. To evaluate the functional value $\psi^{(\mu)}(p)$

and the gradient value $\nabla\psi^{(\mu)}(p)$ at each trial point $p$, we have to compute $(x(p), y(p))$, which is the exact minimizer of the right-hand side of (2.6). (See (2.7) and (2.8).) Since the cost functions are separable in our test problems, $(x_k(p), y_j(p))$ can be computed separately for $k = 1, 2, \ldots, K$ and $j = 1, 2, \ldots, n$ by (3.3) and (3.4). In particular, (3.3) is a single commodity network flow problem with separable costs. To solve this problem, we used the method developed by the authors in [12].

### 4.3. Determination of parameters

The algorithm contains two parameters $\gamma^{(\mu)}$ and $\delta^{(\mu)}$, which shall respectively be called proximal and tolerance parameters.

For the proximal parameters $\gamma^{(\mu)}$, we tested two strategies: one is to set $\gamma^{(\mu)}$ to be a constant $\bar{\gamma} > 0$ and the other is to update $\gamma^{(\mu)}$ adaptively by

$$\gamma^{(\mu)} = \begin{cases} \beta\gamma^{(\mu-1)} & \text{if } \gamma^{(\mu-1)} < \bar{\gamma}, \\ \bar{\gamma} & \text{otherwise,} \end{cases} \quad \mu = 1, 2, \ldots, \tag{4.1}$$

with an initial value $\gamma^{(0)} > 0$.

Next, we consider the tolerance parameters $\delta^{(\mu)}$. Theoretically, $\delta^{(\mu)}$ has to tend to 0 as $\mu \to \infty$ by (2.14). But it is usually impractical to use too small a value of $\delta^{(\mu)}$, because then the criterion (2.13) becomes very severe and hence a lot of CPU time is required to satisfy (2.13). Here we simply set $\delta^{(\mu)} = 10^{-1}$ for all $\mu$.

### 4.4. Results

We summarize the results of our numerical experiments in three tables. The data in each table are the average of four different test problems randomly generated.

First, we examined the performance of the algorithm for various values of the proximal and tolerance parameters. Tables 1 and 2 respectively contain the results for test problems $P(100, 500, K, 10^2)$ and $P(100, 1000, K, 10^2)$ with $K = 4, 6, 8$, where

#ite: the number of major iterations,

#qn: the cumulative number of quasi-Newton iterations in computing (2.9), and

CPU: the CPU time (in second) spent to attain the following termination condition of the algorithm:

$$\left| \sum_{k=1}^{K} x_k^{(\mu+1)} - y^{(\mu+1)} \right| < 10^{-4} \left| \sum_{k=1}^{K} x_k^{(1)} - y^{(1)} \right|.$$

We examined the following four rules of controlling the proximal parameters $\gamma^{(\mu)}$:

(a) $\gamma^{(\mu)} = 2$ for all $\mu$,

Table 1: Results for quadratic multicommodity network flow problems $P(100, 500, K, 10^2)$.

| | $P(100,500,4,10^2)$ | | | $P(100,500,6,10^2)$ | | | $P(100,500,8,10^2)$ | | |
|---|---|---|---|---|---|---|---|---|---|
| rule | #it | #qn | CPU (sec) | #it | #qn | CPU (sec) | #it | #qn | CPU (sec) |
| a | 154 | 155 | 168.4 | 163 | 164 | 265.5 | 162 | 163 | 322.1 |
| b | 35 | 76 | 60.9 | 37 | 87 | 89.3 | 36 | 95 | 102.9 |
| c | 7 | 104 | 113.9 | 7 | 88 | 76.8 | 7 | 97 | 124.9 |
| d | 14 | 71 | 51.4 | 14 | 65 | 52.3 | 14 | 67 | 66.2 |

Table 2: Results for quadratic multicommodity network flow problems $P(100, 1000, K, 10^2)$.

| | $P(100,1000,4,10^2)$ | | | $P(100,1000,6,10^2)$ | | | $P(100,1000,8,10^2)$ | | |
|---|---|---|---|---|---|---|---|---|---|
| rule | #it | #qn | CPU (sec) | #it | #qn | CPU (sec) | #it | #qn | CPU (sec) |
| a | 141 | 141 | 320.8 | 138 | 139 | 381.9 | 138 | 138 | 448.2 |
| b | 32 | 79 | 132.9 | 31 | 84 | 161.0 | 31 | 91 | 197.8 |
| c | 6 | 80 | 108.3 | 6 | 82 | 125.8 | 6 | 83 | 143.6 |
| d | 14 | 63 | 91.6 | 14 | 68 | 113.1 | 14 | 68 | 129.4 |

(b) $\gamma^{(\mu)} = 10$ for all $\mu$,

(c) $\gamma^{(\mu)} = 100$ for all $\mu$,

(d) $\gamma^{(\mu)}$ are updated by (4.1) with $(\gamma^{(0)}, \beta, \bar{\gamma}) = (1, 1.5, 100)$.

Tables 1 and 2 indicate that it is effective to control the value of $\gamma^{(\mu)}$ as in the rule (d). This phenomenon is common to other proximal-type algorithms [10, 13]. Furthermore, it may be worth mentioning that #it and #qn are almost unrelated to the number of commodities.

We then compared the behavior of the algorithm as the condition of the problem is varied. We solved two sets of problems $P(100, 500, 4, \alpha)$ and $P(100, 1000, 4, \alpha)$, where $\alpha = 10^1, 10^2, 10^3, 10^4$. Table 3 shows the results, where the parameters of the algorithm are determined by $(\gamma^{(0)}, \beta, \bar{\gamma}, \delta^{(\mu)}) = (1, 1.5, 100, 10^{-1})$. We find that problems with larger $\alpha$ require more iterations and cpu time. This indicates that the performance of the algorithm heavily depends on the conditioning of the problems.

## 5. Concluding Remarks

In the paper, we have considered to apply the primal-dual proximal point algorithm of [13] to the multicommodity network flow problem P. The essence of this algorithm consists in how to find an approximate saddle point of $L^{(\mu)}$ defined by (2.3). As mentioned in Section 2, however, there is

Table 3: Results for problems $P(100, 500, 4, \alpha)$ and $P(100, 1000, 4, \alpha)$ with $\alpha = 10^1, 10^2, 10^3, 10^4$.

| | $P(100, 500, 4, \alpha)$ | | | $P(100, 1000, 4, \alpha)$ | | |
|---|---|---|---|---|---|---|
| $\alpha$ | #it | #qn | CPU (sec) | #it | #qn | CPU (sec) |
| $10^1$ | 14 | 53 | 20.3 | 14 | 58 | 83.3 |
| $10^2$ | 14 | 78 | 47.1 | 13 | 64 | 99.6 |
| $10^3$ | 28 | 219 | 98.5 | 25 | 196 | 273.2 |
| $10^4$ | 170 | 819 | 365.7 | 63 | 411 | 623.1 |

another implementation (2.4) for this computation, in which the order of max and min operations is reversed. In the following, we consider the possibility of applying the latter method, which is studied in [21] under the general setting, to problem P.

Let us define

$$p(x, y) = \arg \max_{p \in R^n} L^{(\mu)}(x, y, p)$$

and

$$\phi^{(\mu)}(x, y) = L^{(\mu)}(x, y, p(x, y)) = \max_{p \in R^n} L^{(\mu)}(x, y, p). \tag{5.1}$$

Then we may rewrite the formula (2.4) as follows:

$$(x^{(\mu+1)}, y^{(\mu+1)}) \approx \arg \min_{x \in R^{nK}, y \in R^n} \phi^{(\mu)}(x, y) \tag{5.2}$$

and

$$p^{(\mu+1)} = p(x^{(\mu+1)}, y^{(\mu+1)}).$$

Note that since $p$ appears quadratically in $L^{(\mu)}$, $p(x, y)$ can be easily computed as

$$p(x, y) = p^{(\mu)} - \gamma^{(\mu)}(\sum_{k=1}^{K} x_k - y). \tag{5.3}$$

Convergence of the algorithm is guaranteed if $(x^{(\mu+1)}, y^{(\mu+1)})$ in (5.2) is chosen to satisfy one of the following two termination criteria [21]:

$$\mathrm{dist}(0, \partial\phi^{(\mu)}(x^{(\mu+1)}, y^{(\mu+1)})) \leq \frac{\epsilon^{(\mu)}}{\gamma^{(\mu)}},$$

where $\epsilon^{(\mu)}$ are positive constants such that

$$\sum_{\mu=0}^{\infty} \epsilon^{(\mu)} < \infty,$$

and

$$\begin{aligned}&\mathrm{dist}(0, \partial\phi^{(\mu)}(x^{(\mu+1)}, y^{(\mu+1)}))\\ &\quad \le \frac{\delta^{(\mu)}}{\gamma^{(\mu)}}|(x^{(\mu+1)}, y^{(\mu+1)}, p(x^{(\mu+1)}, y^{(\mu+1)})) - (x^{(\mu)}, y^{(\mu)}, p^{(\mu)})|,\end{aligned}$$

where $\delta^{(\mu)}$ are positive constants such that

$$\sum_{\mu=0}^{\infty} \delta^{(\mu)} < \infty.$$

Here, $\partial\phi^{(\mu)}(x, y)$ and $\mathrm{dist}(0, S)$ denote the set of the subgradients of $\phi^{(\mu)}$ at $(x, y)$ and the distance from the origin to a set $S$, respectively. (Note that the function $\phi^{(\mu)}$ is convex but is not always differentiable because we do not assume the differentiability of the objective function of problem P.)

On the other hand, from (5.1) and (5.3), we have

$$\begin{aligned}\phi^{(\mu)}(x, y) &= L(x, y, p(x, y)) + \frac{1}{2\gamma^{(\mu)}}|x - x^{(\mu)}|^2 + \frac{1}{2\gamma^{(\mu)}}|y - y^{(\mu)}|^2\\ &\quad - \frac{1}{2\gamma^{(\mu)}}|p(x, y) - p^{(\mu)}|^2\\ &= L(x, y, p^{(\mu)}) + \frac{1}{2\gamma^{(\mu)}}|x - x^{(\mu)}|^2 + \frac{1}{2\gamma^{(\mu)}}|y - y^{(\mu)}|^2 + \frac{\gamma^{(\mu)}}{2}\left|\sum_{k=1}^{K} x_k - y\right|^2.\end{aligned}$$

Unfortunately, this function is not separable with respect to $x$ and $y$, and hence the minimization (5.2) of $\phi^{(\mu)}$ can not be decomposed in commodities. This might be a drawback when applied to problems with many commodities.

## Acknowledgment

The authors deeply appreciate Professor Toshihide Ibaraki for helpful comments and support.

## References

[1] Ali, I., Barnett D., Farhangian K., Kennington, J., Patty B., Shetty B., McCarl B. and Wong P.: "Multicommodity network problems: Applications and computations," *IIE Transactions*, Vol. 16 (1984), 127-134.

[2] Assad, A.A.: "Multicommodity network flows – A survey," *Networks*, Vol. 8 (1978), 37-91.

[3] Bertsekas, D.P.: "Algorithms for nonlinear multicommodity network flow problems," *International Symposium on Systems Optimization and Analysis* (eds. A. Bensoussan and J.L. Lions). Springer, New York, NY, 1979, 210-224.

[4] Chen, R.J. and Meyer R.R.: "Parallel optimization for traffic assignment," *Mathematical Programming* Vol. 42 (1988), 327-345.

[5] Florian, M.: "An improved linear approximation algorithm for the network equilibrium (packet switching) problem," *IEEE Proceedings, Decision and Control* (1977), 812-818.

[6] Florian, M.: "Mathematical programming applications in national, regional and urban planning," *Mathematical Programming, Recent Development and Applications* (eds. M. Iri and K. Tanabe), Kluwer Academic Publishers, Dordrecht, Holland, 1989, 57-81.

[7] Fukushima, M.: "A modified Frank-Wolfe algorithm for solving the traffic assignment problem," *Transportation Research* Vol. 18B (1984), 169-177.

[8] Fukushima, M.: "On the dual approach to the traffic assignment problem," *Transportation Research* Vol. 18B (1984), 235-245.

[9] Fukushima, M.: "A nonsmooth optimization approach to nonlinear multicommodity network flow problems," *Journal of Operarions Research Society of Japan* Vol. 27 (1984), 151-177.

[10] Fukushima, M.: "A descent algorithm for nonsmooth convex optimization," *Mathematical Programming* Vol. 30 (1984), 163-175.

[11] Hearn, D.W. and Lawphongpanich, S.: "A dual ascent algorithm for traffic assignment problems," *Transportation Research* Vol. 24B (1990), 423-430.

[12] Ibaraki, S., Fukushima, M. and Ibaraki, T.: "Dual-based Newton method for nonlinear minimum cost network flow problems," *Journal of the Operations Research Society of Japan* Vol. 34 (1991), 263-286.

[13] Ibaraki, S., Fukushima, M. and Ibaraki, T.: "Primal-dual proximal point algorithm for linearly constrained convex programming problems," *Computational Optimization and Applications* Vol. 1 (1992) 207-226.

[14] Ibaraki, T. and Fukushima, M.: *FORTRAN 77 Optimization Programming* (in Japanese), Iwanami-Shoten, Tokyo, 1991.

[15] Kennington, J.L.: "A survey of linear cost multicommodity network flows," *Operations Research* Vol. 26 (1978), 209-236.

[16] Klincewicz, J.G.: "Implementing an "exact" Newton method for separable convex transportation problems," *Networks*, Vol. 19 (1989), 95-105.

[17] Lasdon, L.S.: *Optimization Theory for Large Systems.* Macmillan, New York, NY, 1970.

[18] LeBlanc L.J., Morlok E.K. and Pierskalla W.P.: "An efficient approach to solving the road network equilibrium traffic assignment problem," *Transportation Research.* Vol. 9 (1975), 309-318.

[19] Nagamochi, H., Fukushima, M. and Ibaraki, T.: "Relaxation method for the strictly convex multicommodity flow problem with capacity constraints on individual commodities," *Networks* Vol. 20 (1990), 409-426.

[20] Rockafellar, R.T.: *Convex Analysis*. Princeton University Press. Princeton, NJ, 1970.

[21] Rockafellar, R.T.: "Augmented Lagrangians and applications of the proximal point algorithm in convex programming," *Mathematics of Operations Research* Vol. 1 (1976), 97-116.

[22] Tseng, P. and Bertsekas, D.P.: "Relaxation methods for problems with strictly convex separable costs and linear constraints," *Mathematical Programming*, Vol. 38 (1987), 303-321.

[23] Wright, S.J.: "Implementing proximal point methods for linear programming," *Journal of Optimization Theory and Applications* Vol. 65 (1990), 531-554.

[24] Zenios S.A.: "On the fine-grain decomposition of multicommodity transportation problems," *SIAM Journal on Optimization* Vol. 1 (1991), 643-669.

[25] Zhu, C.: "Modified proximal point algorithm for extended linear-quadratic programming," *Computatioal Optimization and Applications* Vol. 1 (1992), 185-206.