Journal of the Operations Research Society of Japan
Vol. 61, No. 4, October 2018, pp. 261–271

A MIXED INTEGER PROGRAMMING APPROACH FOR THE MINIMUM MAXIMAL FLOW

Kuan Lu Shinji Mizuno
Tokyo Institute of Technology

Jianming Shi
Tokyo University of Science

(Received January 1, 2018; Revised July 17, 2018)

Abstract This paper concerns a minimum maximal flow (MMF) problem, which finds a minimum maximal flow in a given network. The problem is known to be NP-hard. We show that the MMF problem can be formulated as a mixed integer programming (MIP) problem and we propose to find the minimum maximal flow by solving the MIP problem.

By performing computational experiments, we observe that the proposed approach is efficient to the MMF problem even for relatively large instances, where the number of edges is up to 15,000, and that the growth rate of running time of our approach is slower than the rates of previous works when the sizes of the instances grow.

Keywords: Network Flow, Minimum Maximal Flow, Linear Programming, Linear Complementarity Conditions, Mixed Integer Programming

1. Introduction

The minimum maximal flow (MMF) problem is introduced by Shi and Yamamoto [10] in 1997 and it is known to be NP-hard. Let $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E}, s, t, \boldsymbol{c})$ be a network, where $\boldsymbol{V}$ is a set of $m+2$ nodes v_j ($j \in \{1, 2, \ldots, m+2\}$), $\boldsymbol{E}$ is a set of n edges e_k ($k \in \{1, 2, \ldots, n\}$), s is a source, t is a sink, and $\boldsymbol{c} \in \mathbb{R}^n$ is a positive vector whose k-th element c_k denotes the capacity of k-th edge e_k. In this paper, $\boldsymbol{V}$ includes both s and t ($s = v_{m+1}$ and $t = v_{m+2}$). Let $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ be the node-edge incidence matrix of the network $\boldsymbol{N}$, that is, each entry a_{jk} ($j \in \{1, 2, \ldots, m\}$ and $k \in \{1, 2, \ldots, n\}$) is defined as

$$a_{jk} = \begin{cases} 1 & \text{if edge } e_k \text{ leaves node } v_j, \\ -1 & \text{if edge } e_k \text{ enters node } v_j, \\ 0 & \text{otherwise.} \end{cases}$$

Then the set $\boldsymbol{X}$ of feasible flows is given by

$$\boldsymbol{X} = \{\boldsymbol{x} \in \mathbb{R}^n | \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0},\ \boldsymbol{0} \leqq \boldsymbol{x} \leqq \boldsymbol{c}\}, \tag{1.1}$$

where $\boldsymbol{0}$ is a vector whose each element is 0. The value of a flow $\boldsymbol{x}$ is computed by $\boldsymbol{d}^\top \boldsymbol{x}$, where the k-th element of $\boldsymbol{d} \in \mathbb{R}^n$ is

$$d_k = \begin{cases} 1 & \text{if edge } e_k \text{ leaves source } s, \\ -1 & \text{if edge } e_k \text{ enters source } s, \\ 0 & \text{otherwise.} \end{cases}$$

A feasible flow $\boldsymbol{x} \in \boldsymbol{X}$ is maximal if there are no feasible flows $\boldsymbol{y} \in \boldsymbol{X}$ such that $\boldsymbol{y} \geqq \boldsymbol{x}$ and $\boldsymbol{y} \neq \boldsymbol{x}$. The minimum maximal flow (MMF) problem is denoted as

$$\begin{array}{ll} \text{minimize} & \boldsymbol{d}^\top \boldsymbol{x} \\ \text{subject to} & \boldsymbol{x} \in \boldsymbol{X} \text{ is a maximal flow.} \end{array} \tag{1.2}$$

To our knowledge, there is no concrete description of this problem as mixed integer programming.

In traditional network flow problems, there are many excellent works, see [1,4]. These works are usually established on an assumption that all flows can be controlled in the network. Then the maximum flow value is attainable. However, a flow may not always satisfy this assumption in the real world. The minimum of the maximal flow values plays an important role in finding what happens when we cannot control the flow. The MMF problem is closely related to an uncontrollable flow raised by Iri [7].

Shi and Yamamoto [10] propose a global optimization algorithm for the MMF problem. The algorithm increases the lower bound of a flow on each edge and uses a global search method. In 2003, Shigeno et al. [11] propose an algorithm which minimizes a linear function over an efficient set. Later an algorithm, which also optimizes a function over an efficient set, is proposed by Gotoh et al. [5]. The algorithm transforms the MMF problem into a global optimization problem. In 2007, Yamamoto and Zenke [12] express the set of the maximal flows as a DC set and propose an algorithm which solves the problem with an outer approximation method and a local search method based on DC optimization theory [2,6]. In 2009, Chen et al. [3,8] use non-homogeneous Farkas' Theorem to define the set of maximal flows and propose an algorithm. In 2014, Muu and Thuy [9] use smoothing technique on DC algorithm to solve the problem locally. In those algorithms, the computational time highly depends on the size of n. Some papers mentioned above report results of computational experiments for solving the MMF problems, but the size of n is at most 200.

In this paper, we show that the MMF problem can be formulated as a mixed integer programming (MIP) problem and we propose to find the minimum maximal flow by solving the MIP problem. We show in Section 2 that the maximal flow of the network $\boldsymbol{N}$ can be expressed as a solution of linear complementarity conditions. By using the result of Section 2, we formulate the MMF problem as a MIP problem in Section 3. In Section 4, we report some results of preliminary computational experiments, in which we solve instances of the MIP problem by using a commercial solver. From the computational results, we observe that the proposed approach is efficient to the MMF problem even for relatively large instances, where the number of edges is up to 15,000, and that the growth rate of running time of our approach is slower than the rates of previous works when the sizes of the instances grow. In this sense, our approach has good features in computational efficiency.

2. Linear Complementarity Conditions for the Maximal Flow

In this section, we will show that the maximal flow of the network $\boldsymbol{N}$ can be expressed as a solution of linear complementarity conditions. More precisely, we will prove the next theorem.

Theorem 2.1. *A vector $\boldsymbol{x} \in \mathbb{R}^n$ is the maximal flow of the network $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E}, s, t, \boldsymbol{c})$ if* *and only if there exist* $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{u} \in \mathbb{R}^n$, *and* $\boldsymbol{v} \in \mathbb{R}^n$ *such that*

$$\begin{aligned}&\boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1},\\&(\boldsymbol{c} - \boldsymbol{x})^\top \boldsymbol{u} = 0,\\&\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0},\\&\boldsymbol{0} \leqq \boldsymbol{x} \leqq \boldsymbol{c},\\&\boldsymbol{u}, \boldsymbol{v} \geqq \boldsymbol{0},\end{aligned}$$

where $\boldsymbol{1} \in \mathbb{R}^n$ *is a vector whose each element is* 1.

Proof. Obviously, a flow $\boldsymbol{x} \in \boldsymbol{X}$ is maximal if and only if $\boldsymbol{y} = \boldsymbol{x}$ is an optimal solution of the following linear programming problem:

$$\begin{array}{ll}\text{maximize} & \boldsymbol{1}^\top \boldsymbol{y}\\ \text{subject to} & \boldsymbol{A}\boldsymbol{y} = \boldsymbol{0},\\ & \boldsymbol{x} \leqq \boldsymbol{y} \leqq \boldsymbol{c},\end{array} \tag{2.1}$$

where $\boldsymbol{y} \in \mathbb{R}^n$ is a vector of variables. From the optimal conditions (or the KKT conditions) of linear programming, $\boldsymbol{y}$ is an optimal solution of (2.1) if and only if there exist $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{u} \in \mathbb{R}^n$, and $\boldsymbol{v} \in \mathbb{R}^n$ such that

$$\begin{aligned}&\boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1},\\&(\boldsymbol{c} - \boldsymbol{y})^\top \boldsymbol{u} = 0,\\&(\boldsymbol{y} - \boldsymbol{x})^\top \boldsymbol{v} = 0,\\&\boldsymbol{A}\boldsymbol{y} = \boldsymbol{0},\\&\boldsymbol{x} \leqq \boldsymbol{y} \leqq \boldsymbol{c},\\&\boldsymbol{u}, \boldsymbol{v} \geqq \boldsymbol{0}.\end{aligned}$$

Hence a flow $\boldsymbol{x} \in \boldsymbol{X}$ is maximal if and only if $\boldsymbol{y} = \boldsymbol{x}$ is a solution of the above conditions, that is, there exist $\boldsymbol{z}$, $\boldsymbol{u}$, and $\boldsymbol{v}$ such that

$$\begin{aligned}&\boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1},\\&(\boldsymbol{c} - \boldsymbol{x})^\top \boldsymbol{u} = 0,\\&\boldsymbol{A}\boldsymbol{x} = \boldsymbol{0},\\&\boldsymbol{u}, \boldsymbol{v} \geqq \boldsymbol{0}.\end{aligned}$$

Then we obtain the result of the theorem from (1.1). □

3. A Mixed Integer Programming Problem

In this section, we will show that the minimum maximal flow of the network $\boldsymbol{N}$ can be computed by solving a mixed integer programming problem. More precisely, we will prove the next theorem.

Theorem 3.1. *A vector* $\boldsymbol{x} \in \mathbb{R}^n$ *is the minimum maximal flow of the network* $\boldsymbol{N} = (\boldsymbol{V}, \boldsymbol{E}, s, t, \boldsymbol{c})$ *if and only if there exist* $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{w} \in \mathbb{R}^n$, *and* $\boldsymbol{\alpha} \in \{0,1\}^n$ *such that* $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\alpha})$ *is an optimal solution of the following mixed integer programming problem:*

$$\begin{array}{ll}\text{minimize} & \boldsymbol{d}^\top \boldsymbol{x}\\ \text{subject to} & \boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{w} = \boldsymbol{1},\\ & \boldsymbol{c} - \boldsymbol{x} \leqq c_{max}\boldsymbol{\alpha},\\ & \boldsymbol{w} \leqq n(\boldsymbol{1} - \boldsymbol{\alpha}),\\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0},\\ & \boldsymbol{0} \leqq \boldsymbol{x} \leqq \boldsymbol{c},\\ & \boldsymbol{\alpha} \in \{0,1\}^n,\end{array} \tag{3.1}$$

where

$$c_{max} = \max\{c_k \mid k \in \{1, 2, \dots, n\}\}.$$

Proof. From (1.2) and Theorem 2.1, $\boldsymbol{x} \in \mathbb{R}^n$ is the minimum maximal flow of the network $\boldsymbol{N}$ if and only if there exist $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{u} \in \mathbb{R}^n$, and $\boldsymbol{v} \in \mathbb{R}^n$ such that $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}, \boldsymbol{v})$ is an optimal solution of the following linear programming problem with linear complementarity constraints:

$$\begin{array}{ll} \text{minimize} & \boldsymbol{d}^\top \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1}, \\ & (\boldsymbol{c} - \boldsymbol{x})^\top \boldsymbol{u} = 0, \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}, \\ & \boldsymbol{0} \leqq \boldsymbol{x} \leqq \boldsymbol{c}, \\ & \boldsymbol{u}, \boldsymbol{v} \geqq \boldsymbol{0}. \end{array} \tag{3.2}$$

Suppose that $\boldsymbol{x}^*$ is the minimum maximal flow of the network $\boldsymbol{N}$. Then there exist $\boldsymbol{z}^* \in \mathbb{R}^m$, $\boldsymbol{u}^* \in \mathbb{R}^n$, and $\boldsymbol{v}^* \in \mathbb{R}^n$ such that $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is an optimal solution of (3.2). Define a subset of $\{1, 2, \dots, n\}$ by

$$I^* = \{k \mid x_k^* = c_k\}.$$

Then $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is an optimal solution of the following linear programming problem for $I = I^*$:

$$\begin{array}{ll} \text{minimize} & \boldsymbol{d}^\top \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1}, \\ & x_k = c_k, \quad k \in I, \\ & u_k = 0, \quad k \notin I, \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}, \\ & \boldsymbol{0} \leqq \boldsymbol{x} \leqq \boldsymbol{c}, \\ & \boldsymbol{u}, \boldsymbol{v} \geqq \boldsymbol{0}. \end{array} \tag{3.3}$$

Let $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}, \boldsymbol{v})$ be any feasible solution of (3.3) for any $I \subset \{1, 2, \dots, n\}$. Then $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}, \boldsymbol{v})$ is a feasible solution of the problem (3.2), so we have that

$$\boldsymbol{d}^\top \boldsymbol{x}^* \leqq \boldsymbol{d}^\top \boldsymbol{x}.$$

Hence $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is the minimum solution of all the problems (3.3) for $I \subset \{1, 2, \dots, n\}$.

From Lemma 3.1 below, we can assume that

$$\|(\boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)\|_\infty \leqq n.$$

Define $\boldsymbol{\alpha}^* \in \{0, 1\}^n$ by

$$\alpha_k^* = \begin{cases} 0, & k \in I^*, \\ 1, & k \notin I^*. \end{cases}$$

Then $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*, \boldsymbol{\alpha}^*)$ is an optimal solution of the following mixed integer programming problem:

$$\begin{array}{ll} \text{minimize} & \boldsymbol{d}^\top \boldsymbol{x} \\ \text{subject to} & \boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1}, \\ & \boldsymbol{c} - \boldsymbol{x} \leqq c_{max}\boldsymbol{\alpha}, \\ & \boldsymbol{u} \leqq n(\boldsymbol{1} - \boldsymbol{\alpha}), \\ & \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}, \\ & \boldsymbol{0} \leqq \boldsymbol{x} \leqq \boldsymbol{c}, \\ & \boldsymbol{u}, \boldsymbol{v} \geqq \boldsymbol{0}, \\ & \boldsymbol{\alpha} \in \{0,1\}^n. \end{array} \tag{3.4}$$

Now we will show that the problem (3.4) is equivalent to the problem (3.1) in the sense that, for any feasible solution of one problem, we can get a feasible solution of another problem, where their objective function values are the same. For any $\boldsymbol{u} \in \mathbb{R}^n$ and $\boldsymbol{v} \in \mathbb{R}^n$, define $\boldsymbol{w}(\boldsymbol{u}, \boldsymbol{v}) \in \mathbb{R}^n$ by

$$\boldsymbol{w}(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u} - \boldsymbol{v}.$$

Then for any feasible solution $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{\alpha})$ of (3.4), $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}(\boldsymbol{u}, \boldsymbol{v}), \boldsymbol{\alpha})$ is a feasible solution of (3.1) and their objective function values are the same. And for any $\boldsymbol{w} \in \mathbb{R}^n$, define $\boldsymbol{u}(\boldsymbol{w}) \in \mathbb{R}^n$ and $\boldsymbol{v}(\boldsymbol{w}) \in \mathbb{R}^n$ by

$$\begin{array}{rcl} u_k(\boldsymbol{w}) & = & \max\{w_k, 0\}, \quad k \in \{1, 2, \ldots, n\}, \\ v_k(\boldsymbol{w}) & = & \max\{-w_k, 0\}, \quad k \in \{1, 2, \ldots, n\}. \end{array}$$

Then for any feasible solution $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\alpha})$ of (3.1), $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}(\boldsymbol{w}), \boldsymbol{v}(\boldsymbol{w}), \boldsymbol{\alpha})$ is a feasible solution of the problem (3.4) and their objective function values are the same.

Since $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*, \boldsymbol{\alpha}^*)$ is an optimal solution of (3.4), $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{w}(\boldsymbol{u}^*, \boldsymbol{v}^*), \boldsymbol{\alpha}^*)$ is an optimal solution of the problem (3.1), that is, there exist $\boldsymbol{z} \in \mathbb{R}^m$, $\boldsymbol{w} \in \mathbb{R}^n$, and $\boldsymbol{\alpha} \in \{0,1\}^n$ such that $(\boldsymbol{x}^*, \boldsymbol{z}, \boldsymbol{w}, \boldsymbol{\alpha})$ is an optimal solution of (3.1).

Conversely, suppose that $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{w}^*, \boldsymbol{\alpha}^*)$ is an optimal solution of the problem (3.1). From the discussion above, $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}(\boldsymbol{w}^*), \boldsymbol{v}(\boldsymbol{w}^*), \boldsymbol{\alpha}^*)$ is an optimal solution of the problem (3.4). Define I^* by

$$I^* \;=\; \{k \mid \alpha^*_k = 0, k \in \{1, 2, \ldots, n\}\}.$$

Then $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}(\boldsymbol{w}^*), \boldsymbol{v}(\boldsymbol{w}^*))$ is an optimal solution of (3.3) for $I = I^*$ and

$$\boldsymbol{d}^\top \boldsymbol{x}^* \leqq \boldsymbol{d}^\top \boldsymbol{x}$$

for any feasible solution $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{u}, \boldsymbol{v})$ of (3.3) for any $I \subset \{1, 2, \ldots, n\}$. Hence $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}(\boldsymbol{w}^*),$ $\boldsymbol{v}(\boldsymbol{w}^*))$ is an optimal solution of (3.2) and $\boldsymbol{x}^*$ is a minimum maximal flow of the network $\boldsymbol{N}$. □

Lemma 3.1. *Fix any $I \subset \{1, 2, \ldots, n\}$. If the linear programming problem (3.3) has an optimal solution, then there exists an optimal solution $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ which satisfies*

$$\|(\boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)\|_\infty \leqq n.$$

Proof. Suppose that the linear programming problem (3.3) has an optimal solution. From the fundamental theorem of linear programming, it has a basic optimal solution $(\boldsymbol{x}^*, \boldsymbol{z}^*, \boldsymbol{u}^*,$ $\boldsymbol{v}^*)$. So $(\boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)$ is a basic solution of the following linear system of equations:

$$\begin{array}{l} \boldsymbol{A}^\top \boldsymbol{z} + \boldsymbol{u} - \boldsymbol{v} = \boldsymbol{1}, \\ u_k = 0, \quad k \notin I. \end{array}$$

Since the coefficient matrix of the system above is totally unimodular, we obtain from Cramer's rule that

$$\|(\boldsymbol{z}^*, \boldsymbol{u}^*, \boldsymbol{v}^*)\|_\infty \leqq \|\mathbf{1}\|_1 = n.$$

□

4. Preliminary Computational Experiments

In this section, we will report some results of preliminary computational experiments for finding the minimum maximal flow of the network $\boldsymbol{N}$ by solving the MIP problem (3.1). We use INTLINPROG of Gurobi 8.00 and Matlab 2017a to solve instances of the problem. The OS is Windows 10, CPU is 3.20GHz Intel Core i5-6500, and the memory is 32GB.

We use two kinds of the MMF problems. First, we construct networks whose minimum maximal flow values are known, so that we can check whether the exact optimal solutions are computed. Then we use the MMF problems presented in Shigeno et al. [11] and Muu and Thuy [9], so that we can compare our approach with theirs. We also make experiments on relatively large instances.

4.1. Specially structured networks

Since the MMF problem is NP-hard, it is usually difficult to find an exact optimal solution. To check whether we find an optimal solution, we construct specially structured networks whose optimal solutions can be computed easily. We use networks of the following structure. The network has n edges $\{e_1, e_2, \ldots, e_n\}$ and $m+2$ nodes $\{v_1, v_2, \ldots, v_m, s, t\}$. First three edges are generated as (v_1, s), (t, v_m), (s, t), and the other $n-3$ edges are randomly generated by choosing 2 different nodes from $\boldsymbol{V}/\{s, t\}$ for each edge. The capacity c_k of each edge is randomly chosen from $\{1, 2, ..., 10\}$. See Figure 1 for the illustration of this network. The

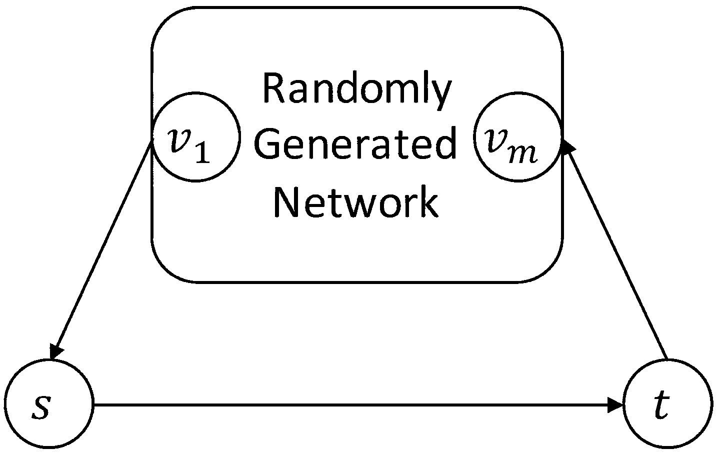

Figure 1: A Specially Structured Network

minimum maximal flow value of this network is equal to the maximum flow value from s to t minus the maximum flow value from t to s.

The results of computational experiments are shown in Table 1. The second column and the third column show the sizes of m and n, the fourth column shows the mean running time of 500 different instances in seconds, the fifth column shows the standard deviation of the running time, and the sixth column shows the rate of instances for which we find the exact optimal solution. We can see from Table 1 that the proposed approach find an optimal solution in a short running time.

Table 1: Experiments on specially structured networks

no.	m	n	Running time(s)	Standard Deviation	Accuracy(%)
1	10	20	0.0901	0.0154	100
2	20	40	0.0997	0.0172	100
3	40	80	0.1121	0.0147	100
4	80	160	0.1188	0.0136	100
5	100	200	0.1192	0.0127	100
6	200	400	0.1352	0.0136	100
7	400	800	0.1923	0.0207	100

4.2. Comparison with previous works

We use totally randomly generated networks presented in Shigeno et al. [11]. The network has n edges $\{e_1, e_2, \ldots, e_n\}$ and $m + 2$ nodes $\{v_1, v_2, \ldots, v_m, s, t\}$. Each edge is randomly generated by choosing 2 different nodes. The capacity c_k of each edge is randomly chosen from $\{1, 2, ..., 10\}$. See Figure 2 for the illustration of this network.

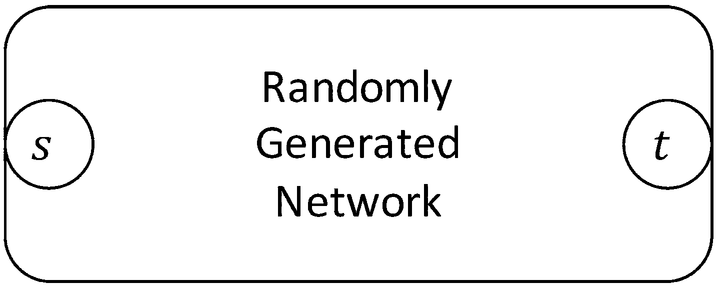

Figure 2: A Generally Structured Network

The results of computational experiments are shown in Table 2. The first 4 columns are the same as Table 1 and the fifth column shows the running time in [11]. In this table, the sizes of generated networks are based on previous work's experiments [11]. In Figure 3, we show the natural logarithm of the running time by Shigeno et al.'s algorithm and ours. From the figure, our running times are much less than Shigeno et al.'s in all tested instances. By regression analysis, we obtain that

$$\log(\text{MIP Running Time}) = 0.0083n - 2.2222, \quad R^2 = 0.8017,$$
$$\log(\text{Shigeno et al. [11] Running Time}) = 0.1032n - 0.7809, \quad R^2 = 0.91815,$$

where R^2 denotes the coefficient of determination. The slope of our approach is about 1/12 of previous work (0.0083 vs 0.1032). We observe that the running times and its growth rate are both much less than those in the previous work [11]. Please note that our computational environments are different from Shigeno et al.'s. Although the constant term of the linear regression highly depends on the environments, the slope is less affected by them.

We use the problems with the same size of Muu and Thuy [9] to compare with theirs. The results of the experiments are shown in Table 3, where the sizes of generated networks are based on their experiments [9]. In Figure 4, we show the natural logarithm of the running time by Muu and Thuy's algorithm and ours. It can be seen that our running times are less than theirs for all the instances. By regression analysis, we obtain that

$$\log(\text{MIP Running Time}) = 0.0022n - 2.129, \quad R^2 = 0.6685,$$
$$\log(\text{Muu and Thuy [9] Running Time}) = 0.0248n - 0.2729, \quad R^2 = 0.86433.$$

Table 2: Comparison of running time [sec.] with Shigeno et al. [11]

no.	m	n	MIP	Shigeno et al. [11]
1	14	20	0.1171	1.07
2	14	24	0.1191	4.07
3	14	28	0.1285	6.19
4	14	32	0.1412	12.89
5	14	36	0.1538	41.29
6	14	40	0.1633	42.64
7	14	44	0.1729	91.16
8	14	48	0.1760	113.36
9	14	52	0.1778	166.84
10	14	56	0.1835	172.63
11	14	60	0.1800	195.84
12	14	64	0.1817	344.29
13	14	68	0.1853	407.18
14	14	72	0.1836	504.10
15	14	76	0.1820	623.54

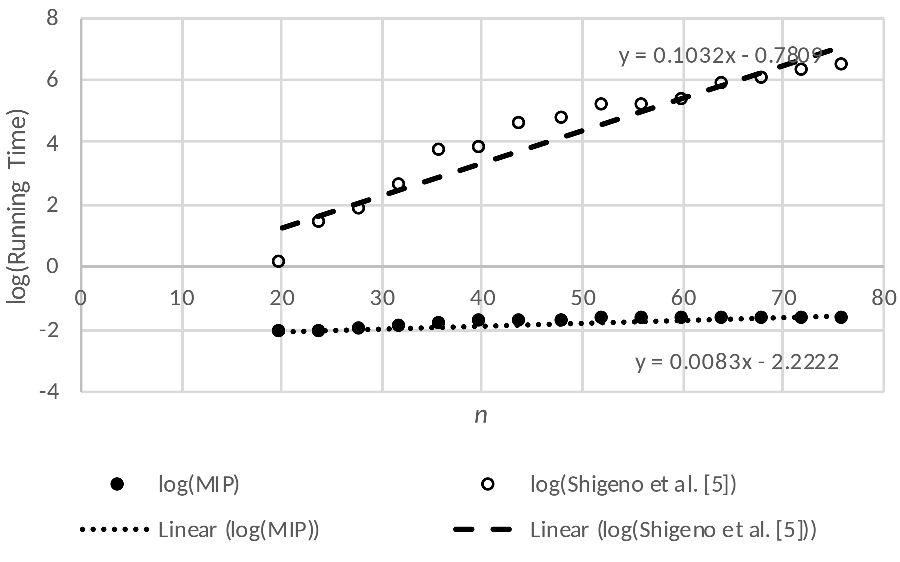

Figure 3: Comparison of log(Running Time) with Shigeno et al. [11]

The slope of our approach is estimated as about 1/11 of previous work (0.0022 vs 0.0248). We can find that the running times and its growth rate are both much less than those in the previous work.

Table 3: Comparison of running time [sec.] with Muu and Thuy [9]

no.	m	n	MIP	Muu and Thuy [9]
1	6	10	0.1141	0.516
2	16	20	0.1147	0.982
3	30	70	0.1706	15.362
4	100	200	0.1747	74.254

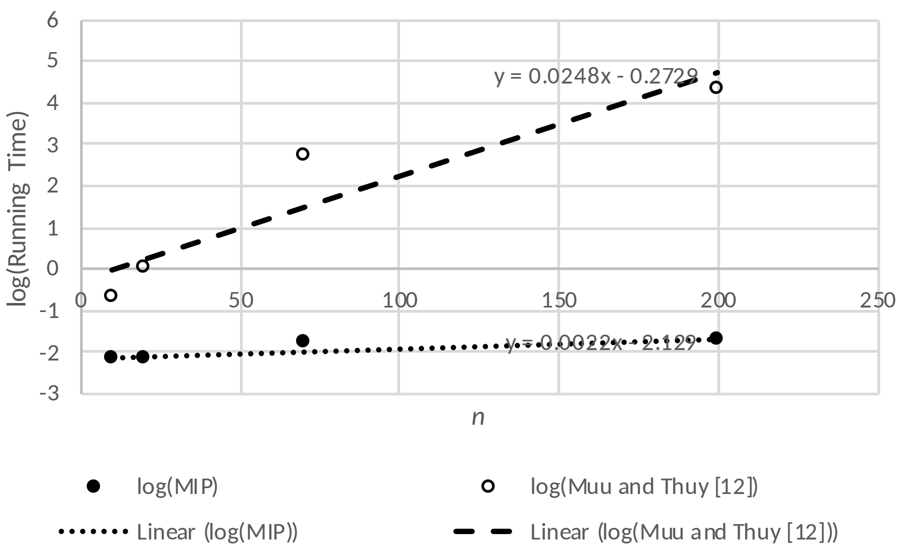

Figure 4: Comparison of log(Running Time) with Muu and Thuy [9]

4.3. Relatively large instances

As far as we know, the largest-scaled instance of the MMF problem is solved in Muu and Thuy [9] where $n = 200$. The results of computational experiments for large-scaled instances are shown in Table 4. We solve the MMF problems of specially structured networks (presented in Section 4.1) and generally structured networks (presented in Section 4.2). In the table, the column 'S RT' and the column 'G RT' show the mean running time in seconds of 50 different instances of specially structured networks and generally structured networks, and the fifth column and the eighth column show the standard deviation of 50 different instances. Accuracy shows the rate of instances for which we find the exact optimal solution.

From Table 4, we see that our approach is capable to compute the exact solution of the MMF problems up to the size of $m = 1,000$ and $n = 15,000$ in a short time and the accuracy is 100%.

5. Conclusion

We show that the minimum maximal flow problem of the network $\boldsymbol{N}$ can be formulated as the mixed integer programming problem (3.1), and we propose to find the minimum

Table 4: Experiments on large instances

no	m	n	S RT	S SD	Accuracy(%)	G RT	G SD
1	100	1,000	0.2862	0.0240	100	0.3259	0.0263
2	100	2,000	0.5566	0.0672	100	0.6071	0.0538
3	100	3,000	1.0794	0.1535	100	1.0642	0.0957
4	100	4,000	1.8428	0.3187	100	1.9177	0.2144
5	100	5,000	2.7916	0.4727	100	2.8733	0.4026
6	500	1,000	0.2452	0.0435	100	0.2787	0.0365
7	500	2,000	1.3790	0.8169	100	2.1319	1.0890
8	500	3,000	3.0705	1.1969	100	3.6991	1.7350
9	500	4,000	4.2061	1.6904	100	6.1973	3.2700
10	500	5,000	5.3831	1.8795	100	6.6702	2.3094
11	1,000	2,000	0.5811	0.1313	100	0.6088	0.0846
12	1,000	3,000	2.9635	1.9302	100	3.8681	2.6233
13	1,000	4,000	7.6708	5.6761	100	13.6337	8.6696
14	1,000	5,000	19.7831	14.1366	100	23.6851	18.5367
15	1,000	10,000	64.6637	6.7976	100	64.5310	6.3798
16	1,000	15,000	205.0781	18.2760	100	203.9493	16.6314

maximal flow by solving the problem. We observe from our preliminary computational experiments that the running times of our approach are much less than those in previous works and that relatively large instances can be solved. Also, the growth rate of running time is slowed down much more than those in previous works.

Acknowledgment

The authors would like to thank the anonymous referees for valuable comments and suggestions. This research is supported in part by Grant-in-Aid for Science Research (A) 26242027 of Japan Society for the Promotion of Science.

References

[1] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin: *Network Flows: Theory, Algorithms, and Applications.* (Prentice hall, 1993).

[2] L.T.H. An, P.D. Tao, and L. D. Muu: Numerical solution for optimization over the efficient set by dc optimization algorithms. *Operations Research Letters*, **19-3** (1996) 117–128.

[3] W. Chen, S. Masahiro, and J. Shi: The effective algorithm for minimum maximal network flow. *IPSJ SIG Technical Reports*, **2009-2** (2009), 1–6 (in Japanese).

[4] R. Diestel: *Graph Theory, 4th Edition.* (Springer, 2012).

[5] J. Gotoh, N.V. Thoai, and Y. Yamamoto: Global optimization method for solving the minimum maximal flow problem. *Optimization Methods and Software*, **18-4** (2003), 395–415.

[6] R. Horst and N.V. Thoai: Dc programming: overview. *Journal of Optimization Theory and Applications*, **103-1** (1999), 1–43.

[7] M. Iri: An essay in the theory of uncontrollable flows and congestion. *Technical Report TRISE*, **94-03** (Department of Information and System Engineering, Chuo University,

1994).

[8] Y. Ji and J. Shi: An algorithm for minimum maximal network flow. *IPSJ SIG technical reports*, **2011-1** (2011), 1–6 (in Japanese).

[9] L.D. Muu and L.Q. Thuy: On dc optimization algorithms for solving minmax flow problems. *Mathematical Methods of Operations Research*, **80-1** (2014), 83–97.

[10] J. Shi and Y. Yamamoto: A global optimization method for minimum maximal flow problem. *Acta Mathematica Vietnamica*, **22-1** (1997), 271–287.

[11] M. Shigeno, I. Takahashi, and Y. Yamamoto: Minimum maximal flow problem: an optimization over the efficient set. *Journal of Global Optimization*, **25-4** (2003), 425–443.

[12] Y. Yamamoto and D. Zenke: Outer approximation method for the minimum maximal flow problem. *Journal of the Operations Research Society of Japan*, **50-1** (2007), 14–30.

Kuan Lu
Department of Industrial Engineering and Management
Tokyo Institute of Technology
2-12-1 Ohokayama, Meguro-ku
Tokyo 152-8552, Japan
E-mail: `lu.k.aa@m.titech.ac.jp`