Journal of the Operations Research
Society of Japan
Vol. 33, No. 1, March 1990

A POLYNOMIAL–TIME BINARY SEARCH ALGORITHM FOR THE MAXIMUM BALANCED FLOW PROBLEM

Akira Nakayama
Otaru University of Commerce

(Received Janualy 20, 1988; Revised March 20, 1989)

Abstract We consider the maximum balanced flow problem of a two–terminal network N, *i.e.*, a maximum flow problem with an additional constraint described in terms of a balancing rate function $\alpha : A \rightarrow \mathbf{R}_+ - \{0\}$, where A is the arc set of N and $\mathbf{R}_+$ is the set of nonnegative reals. In this paper, we propose a polynomial time algorithm for the maximum balanced flow problem, on condition that all given functions in N are rational. The proposed algorithm, which is composed of a binary search algorithm and Dinic's maximum flow algorithm with a parameter, requires $O(\max\{\log(c^*), m\log(\eta^*), nm\}T(n,m))$ time, where $c^* = \max\{c^o(a) : a \in A\}$ for positive integral arc–capacities $(c^o(a) : a \in A)$ and $\eta^* = \max\{\eta(a) : a \in A\}$ for $\alpha(a) \equiv \zeta(a)/\eta(a) \leq 1$ such that $\zeta(a)$ and $\eta(a)$ are positive integers, and $T(n,m)$ is the time for the maximum flow computation for a network with n vertices and $m = |A|$ arcs.

1. Introduction

Minoux [10] considered *the maximum balanced flow problem*, i.e., the problem of finding a maximum flow in a *two-terminal* network such that each arc flow value of the underlying graph is bounded by a fixed proportion of the total flow value from *source* s to *sink* t. The maximum balanced flow problem is motivated by Minoux's research of reliability analysis of communication networks. If a flow from s to t is balanced, then it is guaranteed that the value of the blocked arc flow is at most the fixed proportion of the total flow value from s to t.

Several algorithms [2,3,10,11,13] are proposed for the maximum balanced flow problem. Cui [2,3] showed a simplex and a dual simplex methods without cycling on the underlying graph G of two-terminal network. When *balancing rate functions* are constant, Minoux's algorithm [10] and that of Nakayama [11] are proposed. The former needs $O({p_{\max}}^2 S(n,m))$ time, where $p_{\max}$ is the maximum number of arc disjoint directed paths from source to sink of G and $S(n,m)$ is the complexity of the shortest path problem for a network with n vertices and m arcs and with a nonnegative arc length function. The latter takes $O(\min\{m, \lfloor 1/r \rfloor\}T(n,m))$ time, where $\alpha(a) = r$ $(a \in A)$ for given balancing rate function $\alpha : A \rightarrow \mathbf{R}_+ - \{0\}$ ($\mathbf{R}_+$ is the set of nonnegative reals.), some real r and the arc set A of G, and $T(n,m)$ is the time for the maximum flow computation for a two-terminal network with n vertices and m arcs, and $\lfloor 1/r \rfloor$ is the maximum integer less than or equal to $1/r$. For general balancing rate functions, Zimmermann [13] proposed an algorithm with $O(T(n,m)^2)$ computation time.

On the other hand, Ichimori et al. [7,8] considered *the weighted minimax flow problem*, and Fujishige et al. [5] pointed out the equivalence of the maximum balanced flow problem and the weighted minimax flow problem. When *capacity function* $c : A \rightarrow \mathbf{Z}_+$ and *weight function* $w : A \rightarrow \mathbf{Z}_+$ are given for the set $\mathbf{Z}_+$ of nonnegative integers, the algorithm [8] takes $O(T(n,m)P)$ computation time, where

$P = \log(\max\{c(a)w(a) : a \in A\})$. The algorithm [7] runs in $O(T(n,m)^2)$ time for general weight functions, having the same speed as Zimmermann's.

We can see *the minimax transportation problem*, studied by Ahuja [1],of finding a feasible flow $(x(a) : a = (i,j) \in I \times J)$ from I to J such that $\max\{c(a)x(a) : a = (i,j) \in I \times J\}$ is minimum, where I is a set of *origins*, J is a set of *destinations* and $c(a)$ is the cost of unit shipment on each arc $a = (i,j) \in I \times J$. The minimax transportation problem may be regarded as a special version of the weighted minimax flow problem.

The objective of the present paper is to propose a polynomial time algorithm for the maximum balanced flow problem of a two-terminal network N, on condition that all given functions including $\alpha : A \to \mathbf{R}_+$ in N are rational. We put $\alpha(a) = \varsigma(a)/\eta(a)$ $(a \in A)$ for some two positive integers $\varsigma(a)$ and $\eta(a)$. The total complexity is $O(\max\{\log c^*, m \log \eta^*, nm\}T(n,m))$, where $c^* = \max\{c^o(a) : a \in A\}$ for arc-capacities $c^o(a) \in \mathbf{Z}_+ - \{0\}$ $(a \in A)$, $\eta^* = \max\{\eta(a) : a \in A\}$. The proposed algorithm, which is composed of a binary search algorithm and Dinic's maximum flow algorithm with a parameter, will be expected to be faster than known algorithms in case that all input data are rational.

2. The Maximum Balanced Flow Problem

Let $G = (V,A)$ be a directed graph where V is the vertex set and A is the arc set of G. For two capacity functions $c^o : A \to \mathbf{R}_+$ and $c_o : A \to \mathbf{R}_+$, a *balancing rate function* $\alpha : A \to \mathbf{R}_+ - \{0\}$ and a function $\beta : A \to \mathbf{R}$, consider a two-terminal network $N = (G = (V,A), c^o, c_o, \alpha, \beta, s, t)$ where $\mathbf{R}_+$ is the set of nonnegative reals, $\mathbf{R}$ is the set of reals, s is the *source* and t is the *sink* of G. *The maximum balanced flow problem* (P) for network N is formulated as follows.

$$
\begin{array}{lll}
(P) : & \text{Maximize } f(a^*) & \\
& \text{subject to} & \\
(1) & D \cdot f = 0, & \\
(2) & c_o(a) \le f(a) \le c^o(a) & (a \in A), \\
(3) & f(a) \le \alpha(a) f(a^*) + \beta(a) & (a \in A),
\end{array}
$$

where arc $a^* = (t,s) \notin A$ is added to G and D is the *vertex-arc incidence matrix* of G. We assume that c^o, c_o and β are integral, and that $c^o(a) > \beta(a)$ $(a \in A)$ and $\alpha(a) \equiv \varsigma(a)/\eta(a) \le 1$ $(a \in A)$ for some positive integers $\varsigma(a)$ and $\eta(a)$. Define θ by

$$(4) \qquad \theta = \prod \{\eta(a) : a \in A\}.$$

If the function $f : A^* \to \mathbf{R}_+$ $(A^* = A \cup \{a^*\})$ satisfies (1) $\sim$ (3), then f is called a *balanced flow* in network N. Let f^* be the value maximizing $f(a^*)$ in N, and define the *boundary* $\partial f : V \to \mathbf{R}$ of a function $f : A^* \to \mathbf{R}_+$ in N by

$$(5) \qquad \partial f(v) = \sum\{f((v,i)) : (v,i) \in A^*\} - \sum\{f((i,v)) : (i,v) \in A^*\},$$

where $v \in V$. Associated with problem (P), consider the following two problems (P^*) for network $N^* = (G = (V,A), c^o, c_o, s, t)$:

(P^*) : Maximize $g(a^*)$
subject to (1) and (2), where f should be replaced by g,

and $(P(y))$ for network $N(y) = (G = (V, A), (c^o(a, y) : a \in A), c_o, \beta, s, t)$, where y is a parameter and $c^o(a, y) = \min\{c^o(a), \alpha(a)y + \beta(a)\}$:

$(P(y))$: Maximize $f(a^*)$
subject to constraint (1) and

(6) $\quad c_o(a) \leq f(a) \leq c^o(a, y) \qquad (a \in A).$

Note that $(P(y))$ can be regarded as a maximum flow problem with parameter y in capacities $(c^o(a, y) : a \in A)$.

PROPOSITION 1. *Let $f^{**}(y)$ be the value maximizing $f(a^*)$ in network $N(y)$. If problem (P) is feasible, then we have $f^* = \max\{y : f^{**}(y) = y\}$.* □

Define the *capacity* $c(A(S))$ of a *cut* $A(S) := A^+(S) \cup A^-(S)$ by

$$c(A(S)) = \sum\{c^o(a) : a \in A^+(S)\} - \sum\{c_o(a) : a \in A^-(S)\},$$

where for $S \subset V(\ s \in S,\ t \notin S)$, $A^+(S) = \{(i,j) \in A : i \in S, j \notin S\}$ and $A^-(S) = \{(i,j) \in A : j \in S, i \notin S\}$. A *minimum cut* is defined to be a cut having the minimum capacity. Then we have:

THEOREM 2 [4]. *For any network the maximum flow value from the source to the sink is equal to the capacity of a minimum cut.* □

Let $A(S, y)$ be a minimum cut in network $N(y)$ at y, and $K'(S, y) = \{a \in A^+(S, y) : c^o(a) > \alpha(a)y + \beta(a)\}$ and $K''(S, y) = A^+(S, y) - K'(S, y)$. From theorem 2 we have $f^{**}(y) = U(S, y)y + W(S, y)$, where $U(S, y) = \sum\{\alpha(a) : a \in K'(S, y)\}$ and $W(S, y) = \sum\{\beta(a) : a \in K'(S, y)\} + \sum\{c^o(a) : a \in K''(S, y)\} - \sum\{c_o(a) : a \in A^-(S)\}$. $U(S, y)$ is called *slope* in $N(y)$ at y. Define b^o and b_o by

(7) $\quad b^o = \max\{(c^o(a) - \beta(a))/\alpha(a) : a \in A\},$

(8) $\quad b_o = \max\{\max\{(c_o(a) - \beta(a))/\alpha(a) : a \in A\}, 0\}.$

3. Algorithm for the Maximum Balanced Flow Problem

Consider two functions $z = f^{**}(y)$ and $z = y$ in a (y, z)-plane. From proposition 1, if problem (P) is feasible then the optimal value of (P) is the maximum y^* such that (y^*, y^*) is an intersection point of $z = f^{**}(y)$ and $z = y$. The outline of our algorithm is composed of the following two parts 1 and 2, though the detailed description will be shown in subsequent sections:

Part 1: By a binary search algorithm, we find y_o and y^o such that $y_o \leq f^* \leq y^o$ and $y^o - y_o < \gamma$ for some fixed value $\gamma \in \mathbf{R}_+$.

Part 2: We find f^* by Dinic's maximum flow algorithm with parameter y satisfying $y_o \leq y \leq y^o$.

3.1 Algorithm of Part 1

In later discussion, we assume that problem (P^*) is feasible. Let

(9) $$\gamma = 1/(\theta m^2(m+n+1)^{2n-6}2^w),$$

where $m = |\ A\ |$, $n = |\ V\ |$ and $w = 2mn + n^2 - 2m + n - 2$. *Algorithm I* of Part 1 is as follows.

Algorithm I:

Step 1: Put $FLAG0 = FLAG1 = 1$. Find the maximum flow value g^* in network N^*. If $g^* \geq b^o$, then we have the optimal value $f^* = g^*$ and stop. Otherwise, put $y^o = g^*$ and $y_o = b_o$.

Step 2: (2.1) If $y^o - y_o < \gamma$, then stop. Otherwise, put $y'' = (y^o + y_o)/2$. Do *WAIT-A-MINUTE* $(y'', y^o, y_o, FLAG0, N(y))$. If $FLAG0 = 0$ (y_o is renewed.), then go back to (2.1).

(2.2) Do *JUDGE* $(y'', y^o, y_o, FLAG1, N(y))$. If $FLAG1 = 0$, then stop. Otherwise, go back to (2.1).

In algorithm I, *WAIT-A-MINUTE* $(y'', y^o, y_o, FLAG0, N(y))$ and *JUDGE* $(y'', y^o, y_o, FLAG1, N(y))$ are the following procedures, where two variables $FLAG0$ and $FLAG1$ are in $\{0,1\}$ and $N(y) = (G = (V, A), (c^o(a, y) : a \in A), c_o, s, t)$.

Procedure *WAIT-A-MINUTE* $(y'', y^o, y_o, FLAG0, N(y))$:
Calculate the maximum flow value $f^{**}(y)$ of $N(y)$ at $y = y''$. If we have $y'' \leq f^{**}(y'')$ or no flows for $N(y)$, then put $y_o = y''$ and $FLAG0 = 0$. Otherwise, we put $FLAG0 = 1$.

Procedure *JUDGE* $(y'', y^o, y_o, FLAG1, N(y))$:
Find line $z = L(y)$ with slope $U(S, y'')$ for some $S \subset V$ containing point $(y'', f^{**}(y''))$. Then obtain the intersection point (y', y') of $z = L(y)$ and $z = y$. If $y' > y^o$ or $y' < y_o$, then put $FLAG1 = 0$. Otherwise, renew y^o or y_o as follows:
$y^o = y' \quad (y' \leq y'')$,
$y_o = y' \quad (y' > y'')$.

$FLAG0$ shows whether *JUDGE* $(y'', y^o, y_o, FLAG1, N(y))$ is carried out or not, while $FLAG1$ means that if $FLAG1 = 0$, then problem (P) is infeasible.

3.2 Algorithm of Part 2

Assume that $y^o - y_o < \gamma$ after algorithm *I*. Before describing *algorithm II*, change network $N(y)$ into network $N'(y) = (G' = (V', A'), (c'(a,y) : a \in A'), s', t')$ as follows.

(10) $V' = V \cup \{s', t'\}, \quad A' = A^* \cup A^+ \cup A^-,$

(11) $A^+ = \{(s', v) : v \in V,\ \partial c_o(v) < 0\}, \quad A^- = \{(v, t') : v \in V,\ \partial c_o(v) > 0\},$

(12) $c'(a,y) = c^o(a,y) - c_o(a) \quad (a \in A^*),$

(13) $c'((s',v),y) = -\partial c_o(v) \quad ((s',v) \in A^+),$

(14) $c'((v,t'),y) = \ \ \partial c_o(v) \quad ((v,t') \in A^-),$

where $c_o(a^*) = c^o(a^*, y) = y$, s' is the source and t' is the sink of $N'(y)$. Then we have the following proposition.

PROPOSITION 3 [9]. *We have a feasible flow in $N(y)$ satisfying $c_o(a^*) = c^o(a^*, y) = y$ if and only if we have a maximum flow $(f'(a,y) : a \in A')$ from s' to t' in $N'(y)$ such that $f'(a,y) = c'(a,y) \quad (\forall a \in A^+)$.* □

Let $q(y)$ and $q'(y)$ be linear functions of y, and $\Gamma = [r, r'] \subset \mathbf{R}$ be a closed interval. If either $q(y) \leq q'(y) \quad (\forall y \in \Gamma)$ or $q(y) \geq q'(y) \quad (\forall y \in \Gamma)$ then $q(y)$ and $q'(y)$ are *comparable* in Γ. Define *ROUTINE* $(q(y), q'(y), \Gamma, Y)$ as follows, where Y is a variable.

Procedure *ROUTINE*$(q(y), q'(y), \Gamma, Y)$:
If $q(y)$ and $q'(y)$ are comparable in Γ, then put $Y = -1$. Otherwise, obtain the solution $Y \in \mathbf{R}$ of equation $q(y) = q'(y) \quad (y \in \Gamma)$.

Now we show algorithm *II* of Part 2.

Algorithm II:

Step 1: Put $FLAG0 = FLAG1 = 1$. Calculate a maximum flow for network $N'(y)$ by Dinic's maximum flow algorithm: Construct *layered network* L of $N'(y)$ and find a maximal flow of L.

(1.1) Renew L and denote new layered network by L again. If we attain a maximum flow $(f'(a,y) : a \in A')$, then go to Step 2. Otherwise, find a maximal flow of L:

(1.1.1) Find a *flow-augmenting path* $Q(y)$ of L and choose two arc-capacities $q(y)$ and $q'(y)$ of $Q(y)$. (Note that $q(y)$ and $q'(y)$ are linear functions of y.)

(1.1.2) Do *ROUTINE*$(q(y), q'(y), [y_o, y^o], Y)$. If $Y = -1$, then go to (1.1.3). Otherwise, do *WAIT-A-MINUTE*$(Y, y^o, y_o, FLAG0, N(y))$. If $FLAG0 = 0$, then go to (1.1.3). Otherwise, do *JUDGE*$(Y, y^o, y_o, FLAG1, N(y))$. If $FLAG1 = 0$, then stop.

(1.1.3) If we calculated the minimum arc capacity of $Q(y)$, do the flow augmentation of $Q(y)$. Otherwise, find other two arc-capacities $q(y)$ and $q'(y)$ of $Q(y)$ and go to (1.1.2). If we have a maximal flow of L, then go to (1.1) of Step 1. Otherwise, go to (1.1.1).

Step 2: If we attain a maximum flow $(f'(a,y) : a \in A')$ such that $f'(a,y) = c'(a,y)$ for all $a \in A^+$, then we have the optimal value $f^* = \max\{y : y \in [y_o, y^o]\}$ and stop. Otherwise, (P) is infeasible.

4. The Validity and Complexity

The following proposition is easy to see:

PROPOSITION 4. *If problem (P) is feasible and we have not found the optimal value f^* after algorithm I, then we have $y_o \leq f^* \leq y^o$.* □

The *residual network* $N''(y) = (G'' = (V'', A''), (c''(a,y) : a \in A''), s', t')$ with respect to a flow $(f(a,y) : a \in A')$ in network $N'(y)$ is defined as

(15) $\quad V'' = V', \quad A'' = A'_1 \cup A'_2,$

(16) $\quad c''(a,\ y) = c'(a,y) - f(a,y) \qquad (a \in A'_1),$

(17) $\quad c''(a^-,y) = f(a,y) \qquad (a^- \in A'_2),$

where $A'_1 = \{a \in A' : f(a,y) < c'(a,y)\}$ and $A'_2 = \{a^- : a^-$ is the reversed arc of $a \in A'$ with $f(a,y) > 0\}$. Let

$$N''_i(y) = (G''_i = (V''_i, A''_i), (c''_i(a,y) : a \in A''_i), s', t')$$

be i–th residual network as to a maximal flow $(f_{i-1}(a,y) : a \in A''_{i-1})$ of $N''_{i-1}(y)$, where $N''_1(y) = N'(y)$. Let $L''_i(y)$ be the layered network of $N''_i(y)$, and $Q(y)$ be a flow augmenting path of $L''_i(y)$. The flow augmentation of $Q(y)$ is called *path-flow augmentation* of $L''_i(y)$.

PROPOSITION 5. *Let $n(i)$ be the number of path-flow augmentations of $L''_i(y)$. Then we have $n(i) \leq m' - i + 1$ for $m' = |\ A'\ |$.*

(Proof) Let Ξ_i be a set of the paths joining s' and t' of $L''_i(y)$. We see that each path in Ξ_i has the same length, say, $p(i)$. Then we have

$$p(i) + n(i) - 1 \leq |\ A(L''_i(y))\ | \leq m',$$

where $A(L''_i(y))$ is the arc set of $L''_i(y)$. From $i \leq p(i)$, we have $n(i) \leq m' - i + 1$. □

PROPOSITION 6. *Let $(f_{i\ j}(a,y) : a \in A(L''_i(y)))$ be a flow of $L''_i(y)$ obtained after j path-flow augmentations of $L''_i(y)$. Then we have:*

(18) $\quad f_{i\ j}(a,y) = \sum\{\kappa_i^a(e) c''_i(e,y) : e \in A(L''_i(y))\} \quad (\kappa_i^a(e) \in \mathbf{Z},\ a \in A(L''_i(y))),$

$$\max\{|\ \kappa_i^a(e)\ | : e \in A(L''_i(y))\} \leq 2^{j-1}.$$

(19) $\quad$ *If $f_{i\ j}(a,y) < c''_i(a,y)$, then we have $\kappa_i^e(a) = 0 \quad (e \in A(L''_i(y)))$,*

where $\mathbf{Z}$ is the set of integers, $\mathbf{Z}_+$ is the set of nonnegative integers and

$c''_i(e,y) \in \mathbf{Z}_+ - \{0\}$ is the capacity of arc e in $N''_i(y)$.

(Proof) We can prove (18) and (19) by induction on j. We note here that if $f_{i\ k}(a,y) = c''_i(a,y)$ for some $a \in A(L''_i(y))$ and some $k \le j$, then we have: $f_{i\ d}(a,y) = c''_i(a,y) \quad (k \le d \le j)$. □

PROPOSITION 7. *Let $(c''_i(a,y) : a \in A''_i)$ be capacity of the i-th residual network $N''_i(y)$, where $i \ge 2$. Then we have:*

(20) $\quad c''_i(a,y) = \sum\{\psi_i^a(e)c'(e,y) : e \in A'\} \quad (\psi_i^a(e) \in \mathbf{Z},\ a \in A''_i),$

(21) $\quad \max\{|\ \psi_i^a(e)\ | : e \in A'\} \le (m'+1)^{i-2}2^{u(i)},$

where $u(i) = (i-1)(2m'-i)/2$ and $m' = |\ A'\ |$.

(Proof) We use induction on i. From proposition 6, we have (20) and (21) for $i = 2$. Suppose that we carried out J path-flow augmentations to find a maximal flow $(f_{i\ J}(e,y) : e \in A(L''_i(y)))$ of $L''_i(y)$. From proposition 6 we have

(22) For each $a \in F_1 \equiv \{e \in A(L''_i(y)) : c''_i(e,y) = f_{i\ J}(e,y)\}$,

$c''_{i+1}(a^-,y) = c'(a,y) \quad (a \in A'),$

$c''_{i+1}(a^-,y) = c'(a^-,y) \quad (a \notin A'),$

(23) For each $a \in F_2 \equiv \{e \in A(L''_i(y)) \colon c''_i(e,y) > f_{i\ J}(e,y) > 0\}$,

$c''_{i+1}(a,y) = c''_i(a,y) - f_{i\ J}(a,y),$

$c''_{i+1}(a^-,y) = c'(a,y) - c''_i(a,y) + f_{i\ J}(a,y) \quad (a \in A'),$

$c''_{i+1}(a^-,y) = c'(a^-,y) - c''_i(a,y) + f_{i\ J}(a,y) \quad (a \notin A'),$

(24) For each $a \in (A''_i - A(L''_i(y)) \cup F_3) \cup F_4$, $c''_{i+1}(a,y) = c''_i(a,y)$,

where $F_3 = \{e^- : e \in F_1 \cup F_2\}$ and $F_4 = \{e \in A(L''_i(y)) : f_{i\ J}(e,y) = 0\}$. Let

$$f_{i\ J}(a,y) = \sum\{\kappa_i^a(e)c''_i(e,y) : e \in A(L''_i(y))\} \quad (\kappa_i^a(e) \in \mathbf{Z}).$$

Then we have $\max\{|\ \kappa_i^a(e)\ | : e \in A(L''_i(y)) - F_2\} \le 2^{J-1} \quad (a \in A(L''_i(y)))$.

From (22) $\sim$ (24), inductive assumption, $|\ A(L''_i(y))\ | \le m'$ and $J \le m' - i + 1$, we have (20) and (21) replacing i by $i+1$. Note that

$$1 + m'(m'+1)^{i-2}2^{u(i+1)} \le (m'+1)^{i-1}2^{u(i+1)}.$$ □

PROPOSITION 8. *Let $\rho(i) = (m'+1)^{i-2}2^{u(i)}$ in (21). Then we have:*
(25) $\quad \rho(i) \le \rho(n-1) = (m+n+1)^{n-3}2^{\nu} \quad (2 \le i \le n-1),$
where $\nu = (n-2)(2m+n+1)/2$.

(Proof) Let p be the length of the shortest directed path from s' to t' of network $N'(y)$. From $p \ge 3$, $i \le |\ V'\ | - 1$, $|\ V'\ | = n+2$, $m' \le m+n$ and proposition 7, we have (25). □

PROPOSITION 9. *If $Y \neq -1$ in WAIT-A-MINUTE$(Y, y^o, y_o, FLAG0, N(y))$, then we have $Y = \tau\theta/\chi$ for some $\chi \in \{z \in \mathbf{Z}_+ : 0 < z \leq \theta m 2^{m+n}\rho(n-1)\}$ and some $\tau \in \mathbf{Z}_+$.*

(Proof) Consider i-th layered network $L''_i(y)$. Assume that we are going to do J-th path-flow augmentation. From (10) $\sim$ (14) and proposition 7 we see that the solution Y is obtained from linear equation of y such that

(26) $\sum\{\kappa_i^1(e)\alpha(e) : e \in A\}y + \tau_1 = \sum\{\kappa_i^2(e)\alpha(e) : e \in A\}y + \tau_2,$

where $\kappa_i^d(e) \in \mathbf{Z}$, $|\ \kappa_i^d(e)\ | \leq \rho(i)2^{J-1}$ and $\tau_d \in \mathbf{Z}$ for $d = 1, 2$. From (4) we have $\varsigma'(a) \in \mathbf{Z}_+ - \{0\}$ $(a \in A)$ such that $\alpha(a) = \varsigma'(a)/\theta \leq 1$. Let

(27) $\chi = \sum\{\kappa_i^1(e)\varsigma'(e) : e \in A\} - \sum\{\kappa_i^2(e)\varsigma'(e) : e \in A\}.$

Assuming $\tau_{21} = \tau_2 - \tau_1 \geq 0$ we have $Y = \tau_{21}\theta/\chi$. From (27), propositions 5 and 8 and $\varsigma'(e) \leq \theta$ $(e \in A)$, we have $\chi \leq \theta m 2^{m+n}\rho(n-1)$. □

PROPOSITION 10. *WAIT-A-MINUTE $(Y, y^o, y_o, FLAG0, N(y))$ is carried out at most once for $Y \neq -1$.*

(Proof) Assume that *WAIT-A-MINUTE* $(Y, y^o, y_o, FLAG0, N(y))$ is carried out twice for $Y = y_1$ and y_2, where $y_1 \neq y_2$, $y_1 \neq -1$ and $y_2 \neq -1$. From proposition 9, we have

(28) $y_i = \tau_i\theta/\chi_i \qquad (\tau_i \in \mathbf{Z}_+,\ \ \chi_i \in \{z \in \mathbf{Z}_+ : 0 < z \leq \theta m 2^{m+n}\rho(n-1)\}),$

where $i = 1, 2$. From (9) and proposition 8, we have

(29) $$|\ y_1 - y_2\ | \geq \frac{\theta}{(\theta m 2^{m+n}\rho(n-1))^2} = \gamma.$$

From $|\ y_1 - y_2\ | \leq y^o - y_o < \gamma$ and (29), we have a contradiction. □

Concerning the total complexity of algorithms I and II, we have:

PROPOSITION 11. *The total computational complexity of algorithms I and II is*

$$O(\max\{\log c^*, m \log \eta^*, nm\}T(n, m)),$$

where $c^* = \max\{c^o(a) : a \in A\}$, $\eta^* = \max\{\eta(a) : a \in A\}$ and $T(n, m)$ is the time for the maximum flow computation for a two-terminal network with n vertices and m arcs.

(Proof) Consider algorithm I. We have $O(T(n, m))$ time for each step 2. Let k be the number of repetitions of Step 2. From $g^*/2^k < \gamma$ algorithm I takes $O(\max\{\log g^*, \log\theta, mn\}T(n, m))$ time, where g^* is the maximum flow value of network N^*. From proposition 10 and [6] algorithm II requires $O(n^2m + T(n, m))$ time. From $g^* \leq mc^*$ and $\theta \leq (\eta^*)^m$, we have this proposition. □

Now we show an example of our algorithm:

EXAMPLE: Consider network $N = (G = (V, A), c^o, c_o, \alpha, \beta, s, t)$ with $a^* = (t, s)$ in Fig.1, where $a^* \notin A$, $V = \{s, 1, 2, t\}$ and $A = \{a_i : 1 \le i \le 5\}$. The ordered triple attached to each $a \in A$ is $(c_o(a), c^o(a), \alpha(a)y + \beta(a))$. We have $b_o = 0$, $b^o = 20$, $g^* = 12$, $\theta = 24$ and $\gamma = 1/(24 \times 25 \times 100 \times 2^{48})$. In Fig.2 we have $z = y$ and $z = f^{**}(y)$. After Step 1 of algorithm I we have $y^o = 12$ and $y_o = 0$. Going to Step 2 we calculate value $f^{**}(y)$ of network $N(y)$ for $y = (12 + 0)/2 = 6$. From $f^{**}(6) = 17/2 > 6$, we put $y_o = 6$ and go to (2.1). Repeating Step 2, we finally have $y^o = 9 + 1/3$ and $y_o = 9 + \xi \quad (\xi = (1 - 1/2^{63})/3\ )$.

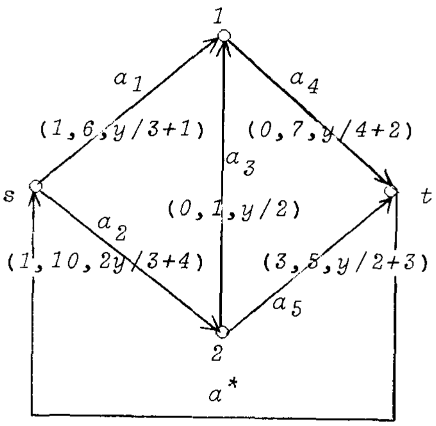

Fig.1

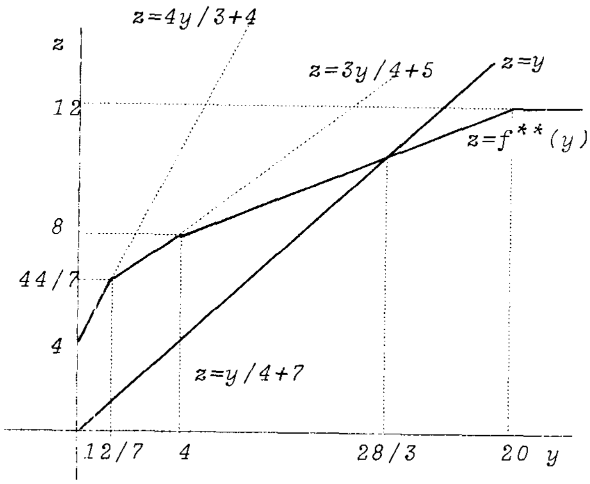

Fig.2

We have network $N'(y)$ in Fig.3 and the layered networks $L''_i(y)$ in Figs. 4-6, where the linear function of y beside each arc in each figure is the arc-capacity. From $1 \le y - 3 \quad (y \in [9 + \xi, 28/3])$, we have $L''_2(y)$ in Fig.5. Solving $1 - y/12 = 2y/3 - 6$ in Fig.6, we have the optimal value $f^* = 28/3$.

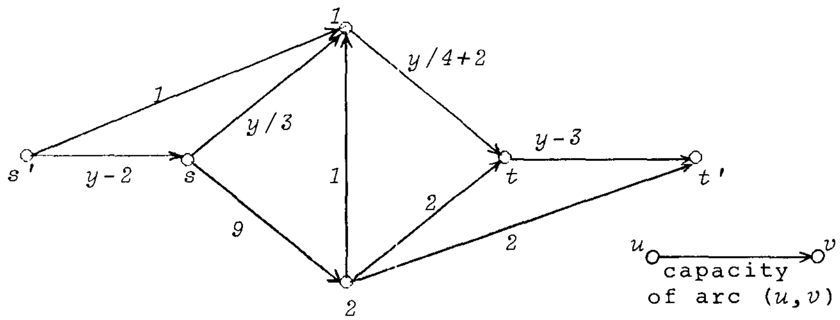

Fig.3

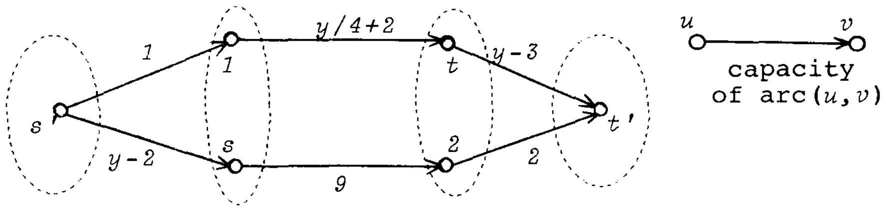

Fig.4

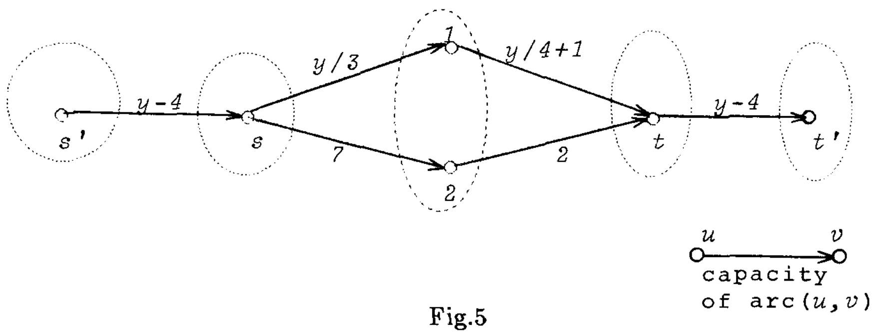

Fig.5

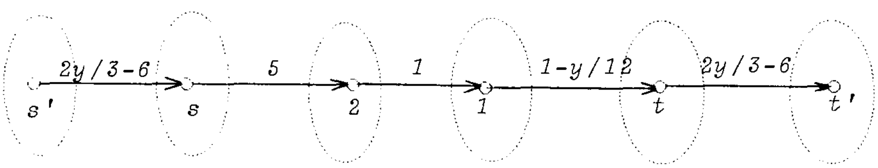

Fig.6

Acknowledgements

The auther wishes to thank referees for pointing out a few errors in the earlier draft of this paper. He also thanks Professor Satoru Fujishige of University of Tsukuba for giving valuable suggestions on this paper.

References

[1] Ahuja, R. K. : Algorithms for the Minimax Transportation Problem. *Naval Research Logistics Quarterly* **33** (1986) 725-739.

[2] Cui, W.-T. : An Algorithm for the Maximum Balanced Flow Problem. Second Year Essay, Doctoral Program in Socio-Economic Planning, University of Tsukuba, 1986.

[3] Cui, W.-T. : A Network Simplex Method for the Maximum Balanced Flow Problem. *Journal of the Operations Research Society of Japan* **31** No.4 (1988) 551-564.

[4] Ford, L. R., Jr. and Fulkerson, D. R. : Flows in Networks. Princeton University Press, Princeton, N.J., 1962.

[5] Fujishige, S., Nakayama, A. and Cui, W.-T. : On the Equivalence of the Maximum Balanced Flow Problem and the Weighted Minimax Flow Problem. *Operations Research Letters* **5** No.4 (1986) 207-209.

[6] Hu, T. C. : Combinatorial Algorithms. Addison-Wesley Publishing Company, 1982.

[7] Ichimori, T., Ishii, H. and Nishida, T. : Weighted Minimax Real-Valued Flows. *Journal of the Operations Research Society of Japan* **24** No.1 (1981) 52-60.

[8] Ichimori, T. and Nishida, T. : Finding the Weighted Minimax Flow in a Polynomial Time. *Journal of the Operations Research Society of Japan* **23** No.3 (1980) 268-271.

[9] Iri, M., Fujishige, S. and Oyama, T. : Graphs, Networks and Matroids, Lecture Series on Mathematical Programming, No.7, Sangyo-Tosho, 1986 (in Japanese).

[10] Minoux, M. : Flots Équilibrés et Flots avec Sécurité. *E.D.F-Bulletin de la Direction des Études et Recherches, Série C—Mathématiques, Informatique* **1** (1976) 5-16.

[11] Nakayama, A. : A Polynomial Algorithm for the Maximum Balanced Flow Problem with a Constant Balancing Rate Function. *Journal of the Operations Research Society of Japan* **29** No.4 (1986) 400-410.

[12] Nakayama, A. : Revised Polynomial-Time Binary Search Algorithm for the Maximum Balanced Flow Problem. Discussion Paper on Data, Theories and Computation in Economic and Management Sciences, Otaru University of Commerce, January, 1989.

[13] Zimmermann, U. : Duality for Balanced Submodular Flows. Preprint No.89, Fachbereich Mathematik, Universität Kaiserslautern, 1985.

Akira NAKAYAMA : Department of Management Sciences, Faculty of Commerce, Otaru University of Commerce, Otaru, Hokkaido, 047, Japan