Bulletin of the *Transilvania* University of Braşov • Vol 8(57), No. 1 - 2015
Series III: Mathematics, Informatics, Physics, 123-128

MAXIMUM CUTS FOR A MINIMUM FLOW

Eleonor CIUREA[1]

Abstract

In this paper we resolve the following three problems. Given a minimum flow f in a network $G = (N, A, l, u)$ determine:

(1) the maximum cut $[S, \bar{S}]$ with the property that for every other maximum cut $[X, \bar{X}], S \subseteq X$;

(2) the maximum cut $[T, \bar{T}]$ with the property that for every other maximum cut $[X, \bar{X}], X \subseteq T$;

(3) whether the network G has a unique maximum cut.

2000 *Mathematics Subject Classification:* 90B10, 90C35, 05C35, 68R10 .
Key words: network flow, minimum flow, maximum cut.

1 Introduction

The network flow models arise in a number of combinatorial applications that on the surface might not appear to be optimal flow problems at all. The problem also arises directly in applications. The maximum flow problem and its dual, the minimum cut problem, are classical combinatorial optimization problems with many applications in science and engineering; see, for example, Ahuja et al. [1]

Before formally defining the minimum flow problem, we give an application of minimum flow. We present the scheduling jobs on identical machines. Let $J = \{2, 3, \ldots, k\}$ be a set of jobs which are to be processed by a set of identical machines M. Each job $i \in J$ is processed by one machine $j \in M$. There is a fix schedule for the jobs, specifying that the job $i \in J$ must start at time $t_1(i)$ and finish at time $t_2(i)$. Furthermore, there is a transition time $t_3(i, j)$ required to set up a machine which has just performed the job i and will perform the job j. The goal is to find a feasible schedule for the jobs which requires as few machines as possible. We can formulate this problem as a minimum flow problem in network $G = (N, A, l, u)$, where: $N = \{1\} \cup N_1 \cup N_2 \cup \{n\}$, 1 is the source node, $N_1 = \{i | i = 2, 3, \ldots k\}$, $N_2 = \{j = k + i - 1 | i = 2, 3, \ldots k\}$, $n = 2k$ is the sink node, $A = A_1 \cup A_2 \cup A_3 \cup A_4$, $A_1 = \{(1, i) | i \in N_1\}$, $A_2 = \{(i, j) | i \in N_1, j \in$

[1]Faculty of Mathematics and Informatics, *Transilvania* University of Braşov, Romania, e-mail: e.ciura@unitbv.ro

$N_2\}$, $A_3 = \{(j,i)|i \in N_1, j \in N_2, t_2(j) + t_3(j,i) \leq t_1(i)\}$ $A_4 = \{(j,n)|j \in N_2\}$, $l(1,i) = 0$, $u(1,i) = 1$, $(1,i) \in A_1$, $l(i,j) = u(i,j) = 1$, $(i,j) \in A_2$ $l(j,i) = 0$ $u(j,i) = 1$ $(j,i) \in A_3$, $l(j,n) = 0$, $u(j,n) = 1$, $(j,n) \in A_4$. This problem has many practical applications, where machines might be workers, tankers, airplanes, truks, processors etc.

2 Minimum flow problem

Let $\mathbb{N}$ be the natural number set and $G = (N, A, l, u)$ a network with the nodes set $N = \{1, \ldots, n\}$, the arcs set $A = \{a_1, \ldots, a_k, \ldots, a_m\}$, $a_k = (i,j)$, the lower bound function $l : A \to \mathbb{N}$, the upper bound (capacity) function $u : A \to \mathbb{N}$, 1 the source node and n the sink node.

For a given pair of subset X, Y of the nodes set N we use the notation $(X,Y) = \{(i,j)|(i,j) \in A, i \in X, j \in Y\}$ and for a given function $g : A \to \mathbb{N}$ we use the notation $g(X,Y) = \sum_{(X,Y)} g(i,j)$.

A flow is a function $f : A \to \mathbb{N}$ satisfying the next conditions:

$$f(i,N) - f(N,i) = \begin{cases} v, & \text{if } i = 1 \\ 0, & \text{if } i \neq 1, n \\ -v, & \text{if } i = n \end{cases} \tag{1.1}$$

for some $v \geq 0$. We refer to v as the value of flow f. A flow is called feasible if f satisfies the following conditions:

$$l(i,j) \leq f(i,j) \leq u(i,j), \quad (i,j) \in A \tag{1.2}$$

The minimum flow problem is to determine a feasible flow f for which v is minimized.

A preflow f is a function $f : A \to \mathbb{N}$ satisfying the next conditions:

$$f(i,N) - f(N,i) \leq 0, i \in N - \{1,n\} \tag{2.1}$$

$$l(i,j) \leq f(i,j) \leq u(i,j), \quad (i,j) \in A \tag{2.2}$$

For any preflow f, we define the deficit of node i as

$$e(i) = f(i,N) - f(N,i), i \in N \tag{3}$$

We refer to a node i with $e(i) = 0$ as balanced. A preflow f satisfying the condition $e(i) = 0, i \in N - \{1,n\}$ is a flow. Thus, a flow is a particular case of preflow.

We further assume, without loss of generality, that if $(i,j) \in A$ the $(j,i) \in A$ (if $(j,i) \notin A$ we consider that $(j,i) \in A$ with $l(j,i) = u(j,i) = 0$).

A cut is set of arcs $[X, \bar{X}] = (X, \bar{X}) \cup (\bar{X}, X)$, $(X, \bar{X}) = \{(i,j) | (i,j) \in A, i \in X, j \in \bar{X}\}$, $X \subset N$, $\bar{X} = N - X$, $(\bar{X}, X) = \{(i,j) | (i,j) \in A, i \in \bar{X}, j \in X\}$. The set $(X, \bar{X})$ denote the set of forward arcs of the cut, and the set $(\bar{X}, X)$ denote the set of backward arcs of the cut. We refer to a cut $[X, \bar{X}]$ as a $1 - n$ cut if $1 \in X$ and $n \in \bar{X}$. For the minimum flow problem, the capacity $c[X, \bar{X}]$ of a $1-n$ cut is $c[X, \bar{X}] = l(X, \bar{X}) - u(\bar{X}, X)$. We refer to a $1 - n$ cut whose capacity is the minimum among all $1 - n$ cuts as a maximum cut.

Theorem 1. *The value of the minimum flow from the source node* 1 *to the sink node* n *in a network* $G = (N, A, l, u)$ *equals to the capacity of the maximum* $1 - n$ *cut.*

For the minimum flow problem, the residual capacity $\hat{r}(i,j)$ of any arc $(i,j) \in A$ with respect to a given flow (preflow) f is given by $\hat{r}(i,j) = c(j,i) - f(j,i) + f(i,j) - l(i,j)$. The residual network is $\hat{G} = (n, \hat{A}, \hat{r})$ with $\hat{A} = \{(i,j) | (i,j) \in A, \hat{r}(i,j) > 0\}$.

The minimum flow problem in a network $G = (N, A, l, u)$ can be solved in two phases:
(1) estabilish a feasible flow if it exists;
(2) if exists a feasible flow, estabilish a minimum flow;

The solution of first fase is presented in [1]. There are three approaches for solving the minimum flow problem:
(1)using decreasing path algorithms;
(2) using preflow algorithms;
(3) minmax algorithms.

Any directed path from the source nodes 1 to the sink node n in the residual network $\hat{G}$ corresponds to a decreasing path in the original network G. Thus, the decreasing path algorithms for minimum flow problem correspond to the augmenting path algorithms for maximum flow problem. The decreasing path algorithms for minimum flow are presented in Table 1, where $\bar{u} = max\{u(i,j) | (i,j) \in A\}$.

Decreasing path algorithms	**Running time**
Generic decreasing path	$O(nm\bar{u})$
Ford-Fulkerson labeling algorithm	$O(nm\bar{u})$
Gabow bit scaling algorithm	$O(nm \log \bar{u})$
Ahuja-Orlin maximum scaling algorithm	$O(nm \log \bar{u})$
Edmonds-Karp shortest path algorithm	$O(nm^2)$
Ahuja-Orlin shortest path algorithm	$O(n^2m)$
Dinic layered network algorithm	$O(n^2m)$
Ahuja-Orlin layered networks algorithm	$O(nm^2)$

Table 1

Also, the preflow algorithms for minimum flow problem correspond to the preflow algorithms for maximum flow problem. The preflow algorithms for minimum flow problem are presented in Table 2.

Preflow algorithms	**Running time**
Generic prefluw algorithm	$O(n^2m)$
FIFO preflow algorithm	$O(n^3)$
Highest label preflow algorithm	$O(nm\log\bar{u})$
Deficit scaling algorithm	$O(nm + n^2\log\bar{u})$

Table 2

The minmax algorithm computes a minimum flow in the following manner: knowing a feasible flow, we determine a minimum flow from the source node 1 to the sink node n by establishing a maximum flow in the residual network from the sink node n to the source node 1, we can use any maximum flow algorithm, including preflow algorithms.

For details with respect to minimum flow algorithm see [2], [3], [4], [5].

3 Maximum cuts for a minimum flow

Let f be a minimum flow of value v in network $G = (N, A, l, u)$ with 1 source node, n sink node and residual network $\hat{G} = (N, \hat{A}, \hat{r})$ with respect to minimum flow f. In $\hat{G}$ we consider and direct path $\hat{D} = (i, i)$.

The first problem is to determine the maximum $1-n$ cut $[S, \bar{S}]$ with property that for every other maximum $1-n$ cut $[X, \bar{X}]$, $S \subseteq X$. We determine this cut by

$$S = \{i | i \in N, \text{exists a directed path } \hat{D} \text{ in } \hat{G} \text{ from source node 1 to node } i\} \quad (4)$$

We have $1 \in S$ and determine $\bar{S} = N - S$. It's clear that $S \subseteq X$ for every other maximum $1-n$ cut $[X, \bar{X}]$. We start from source node 1 and use direct BF search.

The second problem is to determine the maximum $1-n$ cut $[T, \bar{T}]$ with property that for every other maximum $1-n$ cut $[X, \bar{X}]$, $X \subseteq T$. We determine this cut by

$$\bar{T} = \{i | i \in N, \text{exists a directed path } \hat{D} \text{ in } \hat{G} \text{ from node } i \text{ to sink node } n\} \quad (5)$$

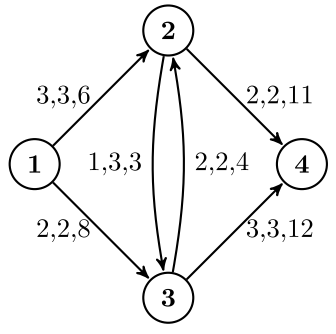
We determine a directed path $\hat{D} = ((i, j), \ldots, (k, n))$ starting from sink node n and use inverse BF search. We have $n \in \bar{T}$ an determine $T = N - \bar{T}$. It's clear that $X \subseteq T$ for every other maximum $1-n$ cut $[X, \bar{X}]$.

The third problem is if the network G has a unique maximum $1-n$ cut $[X, \bar{X}]$ and which is this cut. We determine the maximum $1-n$ cuts $[S, \bar{S}]$ and $[T, \bar{T}]$. If $S = T$ then the network G has a unique maximum $1-n$ cut and $X = S = T$.

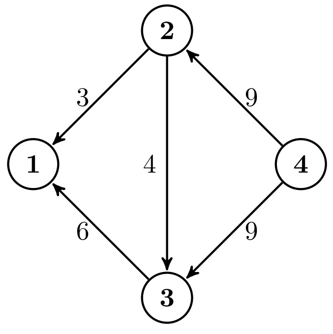
We remark that at the end of any algorithm for minimum flow problem we have the optimum residual network $\hat{G}$. We determine the minimum flow f with $f(i,j) = l(i,j) + max\{0, \hat{r}(i,j) - u(i,j) + l(j,i)\}$, $(i,j) \in A$.

4 Examples

In this section we present two examples. Figure 1a shows the network $G = (N, A, l, u)$ with 1 the source node, 4 the sink node and a minimum flow f. On each arc (i,j) we have $l(i,j), f(i,j), u(i,j)$ in this order. Figure 1b shows the residual network $\hat{G} = (N, \hat{A}, \hat{r})$ with respect to minimum flow f. The value of minimum flow f is $v = 5$.



(a)

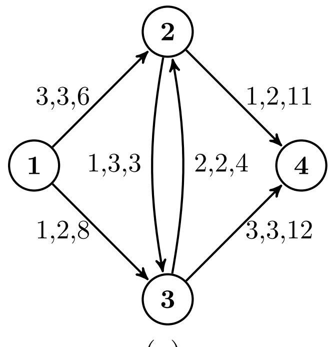


(b)

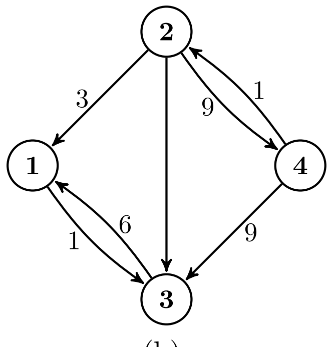
Figure 1

With formula (4) we obtain $S = \{1\}$ and we determine $\bar{S} = N - S = \{2,3,4\}$. We have $[S,\bar{S}] = (S,\bar{S}) \cup (\bar{S},S) = ((1,2),(1,3)) \cup \emptyset$ and $c[S,\bar{S}] = l(S,\bar{S}) - u(\bar{S},S) = l(1,2) + l(1,3) = 3 + 2 = 5 = v$. With formula (5) we obtain $\bar{T} = \{4\}$ and we determine $T = N - \bar{T} = \{1,2,3\}$. We have $[T,\bar{T}] = (T,\bar{T}) \cup (\bar{T},T) = ((2,3),(2,4)) \cup \emptyset$ and $c[T,\bar{T}] = l(T,\bar{T}) - u(\bar{T},T) = l(2,4) + l(3,4) = 2+3 = 5 = v$. Other maximum cut is $[X,\bar{X}] = (X,\bar{X}) \cup (\bar{X},X)$ with $X = \{1,3\}$, $\bar{X} = \{2,4\}$ and we have $c[X,\bar{X}] = l(X,\bar{X}) - u(\bar{X},X) = l(1,2) + l(3,2) + l(3,4) - u(2,3) = 5 = v$. It's clear that $S \subset X$ and $X \subset T$. Fourth $1-4$ cut, and last, in network G is $[Y,\bar{Y}] = (Y,\bar{Y}) \cup (\bar{Y},Y))$ with $Y = \{1,2\}$ and $\bar{Y} = \{3,4\}$. The capacity of this cut is $c[Y,\bar{Y}] = l(Y,\bar{Y}) - u(\bar{Y},Y) = l(1,3) + l(2,3) + l(2,4) - u(3,2) = 1$. Therefore this cut is not maximum cut.

Figure 2a shows the network G with the same significances as in network from Figure 1a. In figure 2b we present the residual network shown in Figure 2a. We obtain $S = \{1,3\}$, $\bar{S} = \{2,4\}$, $\bar{T} = \{2,4\}$, $T = \{1,3\}$. Because $S = T$ result that network G has a unique maximum $1-4$ cut. We have $c[S,\bar{S}] = 5 = v$.



(a)



(b)

Figure 2

References

[1] Ahuja, R., Magnanti, T., Orlin, J., *Network Flows. Theory, algorithms and applications*, Prentice Hall, Inc., Englewood Cliffs, NY, 1993.

[2] Ciupala, L., Ciurea, E., *About preflow algorithms for the minimum flow problems*, WSEAS Transaction on Computer Research, Issue 1, **3** (2008), 35-42.

[3] Ciurea, E., Ciupala, L., *Sequential and parallel algorithms for minimum flows*, Journal of Applied Mathematics and Computing **15** (2004), no. 1-2, 53-75.

[4] Ciurea, E., Ciupala, L., *Algorithms for minimum flows*, Computer Science Journal of Moldova **9(3)** (2001), 275-290.

[5] Georgescu, O., *Algorithms for minimum flows. Dynamic tree implementation*, Bulletin of Transilvania University of Brasov. Seria III: Mathematics, Informatics, Psysics **1(50)** (2008), 513-524.