Computationally Related Problems to the Maximal Flow Problem*

Ki-seong Kim**

【Abstract】

The maximal flow problem is the problem of finding a flow with the maximum value in a given network. It can be solved by any algorithm for the linear program or the minimal cost flow problem. Moreover, several specialized algorithms for the maximal flow problem have been developed and usually favored over general algorithms. In this paper, we show that some problems defined in a network are computationally related to the maximal flow problem. These include the minimal cut problem, the maximal closure problem, and the selection problem. It is notable that all these problems can be solved by any maximal flow algorithm. Alternatively, these related problems can be solved by proper algorithms that have uniquely been devised for each of them.

Keywords : maximal flow problem, minimal cut problem, maximal closure problem, selection problem

논문접수일 : 2010. 07. 12　　논문수정일 : 2010. 09. 08　　게재확정일 : 2010. 10. 06

* This study was supported by the Fund for Humanities & Social Studies at Pusan National University 2009.

** Professor, College of Business, Pusan National University(mgtkkim@pusan.ac.kr)

Ⅰ. Introduction

The maximal flow problem is the problem of finding a flow with the maximum value in a given network *W = (N, A, s, t, c),* where *N* and *A* are node set and arc set, *s* and *t* are source node and sink node, and *c* is a capacity vector for each arc. A flow (or feasible flow) *f = f (i, j)* in a network *W* is a mapping from arc set *A* to non-negative real numbers that satisfies

$$-\sum_{k\in A(j)} f(j,k)+\sum_{i\in B(j)} f(i,j)=\begin{cases}-v, if\ j=s\\ v, if\ j=t\\ 0, if\ j\neq s,t\end{cases} \qquad Eq.(1)$$

$$0\le f(i,j)\le c(i,j)\ for\ all\ (i,j)\in A \qquad Eq.(2)$$

where *A(j)={k (j, k)∈A}, B(j)={i (i, j)∈A}*, and the number *v* is value of the flow. *Eq.(1)* above is called the "flow conservation law", which requires that the sum of incoming flows be equal to the sum of outgoing flows at each node except source and sink node.

We have assumed that a network has a single source and a single sink. If a network has multiple sources and sinks, then it can be reduced to a single source and sink network as follows: (a) add two nodes "super source" and "super sink", (b) link the super source to each of the original source nodes and link each of the original sink nodes to the super sink, and (c) assign a capacity of infinity to each of the arcs added in (b).

The maximal flow problem was originally posed by Harris in 1955(Ford and Fulkerson, 1956: 399-404). Their problem was, in a rail network, to find a maximal flow between two cities that were connected by way of intermediate cities while satisfying the capacity of each link in the network. Naturally, similar straightforward applications of the maximal flow problem often arise in many networks such as transportation networks, communication networks, and networks for the distribution of energy and goods.

Shapiro(1979) gives an interesting application of the maximal flow problem to airline scheduling, where an airline wishes to minimize the number of aircraft required to schedule m flights. This problem is described as a graph in which each node represents a flight and an arc *(i, j)* means that the destination city of flight I is the origin city of flight *j*, and flight *j* departs after the arrival of flight *i*. He showed that the optimal solution to this problem can be obtained by solving a related maximal flow problem.

Other applications of the maximal flow problem include matrix rounding problem(Bacharach, 1966: 732–742), distributed computing on a two-processor computer(Stone, 1977: 85–93), and scheduling on uniform parallel machines (Federgruen and Groenevelt, 1986: 341–349).

In some applications of the maximal flow problem, such as water distribution, each node (as well as each arc) has a flow capacity. This node capacity can be converted to an arc capacity in the following manner. Split each node *n* into two nodes n' and n''. Link n' to n'' and let the capacity of the arc (n', n'') be the capacity of node *n*. All arcs entering n now lead to n' and all arcs leaving *n* now come out of n''. The equivalent standard network is thus constructed.

The maximal flow problem can be formulated as the following linear program (P):

(P)

maximize v

subject to

$$-\sum_{k\in A(s)} f(s,k) + \sum_{i\in B(s)} f(i,s) + v = 0$$
$$-\sum_{k\in A(t)} f(t,k) + \sum_{i\in B(t)} f(i,t) - v = 0$$
$$-\sum_{k\in A(j)} f(j,k) + \sum_{i\in B(j)} f(i,j) = 0 \text{ for all } j \neq s,t$$
$$0 \le f(i,j) \le c(i,j) \text{ for all } (i,j) \in A$$

Thus the maximal flow problem can be solved by any linear programming algorithm such as simplex method. Also, since the maximal flow problem can be viewed as a special case of minimal cost flow (or transshipment) problem, it can be solved by any minimal cost flow algorithm. Moreover, several specialized algorithms for the maximal flow problem have been developed and usually favored over general algorithms. These algorithms are very well surveyed in Ahuja et al.(1993: chapters 6 and 7). Recently, Hochbaum(2008: 992–1009) and Christiano et al.(2010) introduced new algorithms for the maximal flow problem. The fastest known algorithm is that of Goldberg and Rao(1998) with run time of $O(\min(n^{2/3}, m^{1/2})m$ $\log(n^2/m) \log U)$ for U the largest arc capacity.

In this paper, we show that some problems defined in a network are computationally related to the maximal flow problem. These include the minimal cut problem, the maximal closure problem, and the selection problem. It is notable that all these related problems can be solved by any maximal flow algorithm. These problems can be solved by proper algorithms that have uniquely been devised for each of them. However, it is sometimes more efficient to solve these problems by converting to equivalent maximal flow problems.

Ⅱ. Computationally Related Problems

1. Minimal Cut Problem

A cut in a network $W = (N, A, s, t, c)$ is a set of arcs (X, X'), where X and X' are complementary subsets of nodes N such that $s \in X$ and $t \in X'$. The capacity of a cut (X, X') is defined as

$$C(X, X') = \sum_{i \in X} \sum_{j \in X'} c(i, j) \qquad Eq.(3)$$

that is, the sum of the capacities of all arcs which go from nodes in X to nodes in X'. The minimal cut is a cut whose capacity is the minimum among all cuts. Then the minimal cut problem is the problem of finding a minimal cut in a given network.

We note that if all arcs directed from X to X' were deleted from the network, the value of the maximal flow in the resulting network would be zero. Since a cut blocks all chains from s to t, the value of any flow cannot exceed the capacity of any cut.

The following theorem asserts that the equality of flow value *v* and cut capacity *C(X, X')* holds by a suitable choice of flow *f* and cut*(X, X')*. The proof of this theorem can be found in Ford and Fulkerson(1962: 11–12).

[Theorem](Max–Flow Min–Cut Theorem) For any network, the value of the maximal flow is equal to the capacity of the minimal cut.

It is well known that the minimal cut problem is the dual of the maximal flow problem in linear programs. To be specific, the minimal cut problem can be formulated as the following linear program (D), which is the dual of linear program (P).

(D)

minimize $\sum_{(i,j)\in A} c(i,j)\theta(i,j)$

subject to

$$-\pi(i)+\pi(j)+\theta(i,j)\geq 0 \; for\, all\, (i,j)\in A$$
$$\pi(s)-\pi(t)\geq 1$$
$$\pi(j)\; is\; free\; for\, all\; j\in N$$
$$\theta(i,j)\geq 0 \; for\, all\, (i,j)\in A$$

We can see the dual relationship by associating each node *j* and arc *(i, j)* in (P) with dual variables *π(j)* and *Θ(i, j)* in (D) respectively. Every cut *(X, X')* in a network *W* determines a feasible solution to (D) by defining

$$\pi(j)=\begin{cases}1, & if\; j\in X\\ 0, & if\; j\in X'\end{cases} \qquad Eq(4)$$

$$\theta(i,j)=\begin{cases}1, & if\; j\in X\; and\; j\in X'\\ 0, & otherwise\end{cases} \qquad Eq(5)$$

Clearly, the minimal cut corresponds to the optimal solution to (D).

The minimal cut problem is often solved through its dual maximal flow problem since most algorithms for the maximal flow problem provide minimal cuts as a by–product. Alternatively, Phillips and Dessouky(1979: 396–404) suggested a "cut search" algorithm which directly locates the minimal cut in a network with both upper and lower bounds on arc capacities. Karger and Stein(1996: 601-640) gave a new randomized algorithm to the minimal cut problem.

When a maximal flow in a network is already known, a minimal cut in the same network may be located by a single pass of Ford-Fulkerson labeling procedure in $O(|A|)$. On the other hand, knowing a minimal cut does not help much for finding a maximal flow. Based on this observation, Picard and Queyranne(1982: 394-422) conjectured that the minimal cut problem is at most as hard as, and could possibly be easier than, the maximal flow problem.

Some direct applications of the minimal cut problem can be found in certain networks such as communication network and transportation network by interpreting the minimal cut as the cheapest disconnection of a network. These applications might arise in the context of defense and attack of such networks during a war.

An easy example is a network reliability problem by Ramanathan and Colbourn(1987: 253-261). If arcs of a network fail with some probability, it makes intuitive sense that the greatest danger of network disconnection is at a minimum cut.

Picard and Ratliff(1978: 422-433) showed that the rectilinear distance facility location problem can be solved by solving a sequence of minimal cut problems. The problem is to determine the locations of n new facilities in the plane when there are m old facilities already located such that the weighted sum of rectilinear distances is minimized. When there are only two old facilities, the problem can be solved as a minimal cut problem. The general problem can be solved by solving a sequence of at most $m-1$ minimal cut problems.

2. Maximal Closure Problem

Consider a directed graph $G = (N, A)$ where N is a set of nodes and A is a set of arcs. Let (j, k) denote an arc directed from node j to node k. Then, node j is called the predecessor of node k and node k is called the successor of node j. A closure of G

is defined as a subset L of nodes such that if a node belongs to L, then all its successors also belong to L. If a weight w_j is assigned to each node j, then the maximal closure L^* of graph G is a closure such that

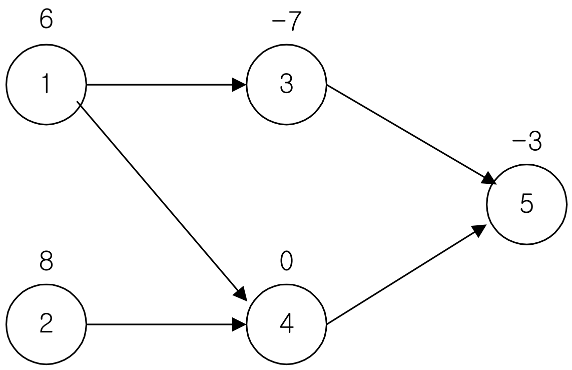
$$\sum_{j \in L^*} w_j \geq \sum_{j \in L} w_j \quad \textit{for all } L \qquad \textit{Eq.}(6)$$

Now the maximal closure problem is the problem of finding a maximal closure of G and can be formulated as a 0–1 integer program (MCP):

$$\begin{aligned}
&\text{(MCP)} \\
&\text{maximize} \quad z = \sum_{j \in N} w_j x_j \\
&\text{subject to} \\
&\qquad x_j - x_k \leq 0 \qquad \textit{for all } (j,k) \in A \\
&\qquad x_j = 0, 1 \qquad \textit{for all } j \in N
\end{aligned}$$

In this formulation, $x_j = 1$ if node j is to be included in a subset L. It can be easily seen that a feasible solution to (MCP) corresponds to a closure of graph G and an optimal solution corresponds to the maximal closure.

For an example of a maximal closure problem, consider the simple graph of 〈Figure 1〉. By definition, possible closures in this graph include {5}, {3, 5}, {2, 4, 5}, and {1, 2, 3, 4, 5}. The weight sums of these closures are −3, −10, 5, and 4 respectively. Notice that a node subset such as {1, 3, 5}, {1, 2}, and {2, 4} is not a closure. The maximal closure of this graph is {2, 4, 5}, which can be determined by enumeration or by solving (MCP).

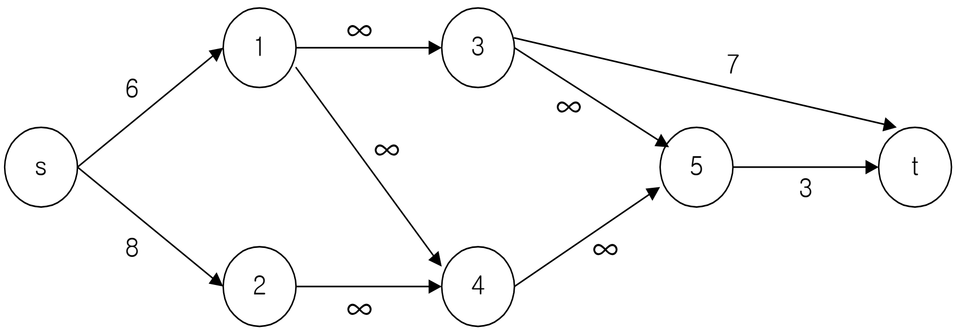


〈Figure 1〉 Example graph of a maximal closure problem

Picard(1976: 1268-1272) showed that any maximal closure problem can be converted to an equivalent maximal flow problem by the following process:

1) add source node s and sink node t
2) link s to each node with positive weight, and assign the weight as arc capacity
3) link each node with negative weight to t, and assign the absolute weight as arc capacity
4) assign an infinity capacity to each of the original arcs.

For example, the maximal closure problem in 〈Figure 1〉 is converted to an equivalent maximal flow problem in 〈Figure 2〉. Then the maximal closure corresponds to the labeled node set at optimal solution, that is, those nodes in the source side of the minimal cut. Indeed, the maximal closure problem is a special class of the minimal cut problem.



〈Figure 2〉 Equivalent graph of the maximal flow problem to 〈Figure 1〉

Thus a maximal closure problem can be solved by any maximal flow algorithm. Alternatively, it can be solved more efficiently by the unique algorithm of Faaland et al. (1990: 315-331).

An important application of the maximal closure problem is found in the mining industry(Lerchs and Grossmann, 1965: 47-54). The underground ore deposit is divided into small units or "blocks" (typically 40'x40'x40' or 100'x40'x40') by which the actual mining is accomplished. We associate with each block the net profit (or loss), that is, mine value minus extraction cost of the block. We note that the wall slopes of the pit must not exceed a certain angle to prevent cave-in. Therefore, in order to mine a certain block "smoothly", we have to get rid of some other blocks in the above levels.

Other applications of the maximal closure problem include optimal destruction of military targets(Orlin, 1987:605-617), investment portfolio decision making under contingency constraints(Weingartner, 1966: 485-516), and scheduling fabrication and assembly tasks in a job shop(Faaland and Schmitt, 1987: 378-388).

There exist some extensions of the maximal closure problem. For example, Tachefine and Soumis(1997: 981-990) formulated the problem of maximal closure with resource constraints. They proposed a solution approach to the problem.

The maximum-ratio closure problem is the problem of finding a closure whose ratio between sums of two different node weights is the maximum among all closures. An example of the problem is the problem of sequencing jobs on a single machine to minimize the total weighted completion time subject to precedence constraints(Sidney, 1975: 283-298).

3. Selection Problem

Suppose a finite set of "activities" together with their benefits and a finite set of "facilities" together with their costs are given. A facility may be shared by multiple activities and any activity depends on the existence of a particular subset of facilities necessary to that activity. The benefit of an activity is gained only when all facilities required by the activity are available. Then the selection problem is the problem of selecting activities and required facilities such that the sum of benefits minus the sum of costs is maximized. This problem is also called a provisioning problem by Lawler(1976).

To describe this problem formally, let b_j denote the benefit of activity a_j and c_k denote the cost of facility f_k. Also let S_j be the subset of facilities necessary to activity a_j. Then the selection problem can be formulated as a 0–1 integer program (SP):

(SP)

$$\text{maximize} \quad z = \sum_j b_j x_j - \sum_k c_k y_k$$

subject to

$$\begin{aligned} x_j - y_k &\le 0 && \textit{for all } f_k \in S_j \\ x_j &= 0, 1 && \textit{for all } a_j \\ y_k &= 0, 1 && \textit{for all } f_k \end{aligned}$$

The selection problem may be viewed as a special class of the maximal closure problem on a bipartite graph in which the node set is partitioned into activities nodes and facilities nodes, and an arc is assigned for each activity–facility relationship. Incidentally, it has been shown that any maximal closure problem can be transformed into an equivalent selection problem Johnson(Johnson, 1968).

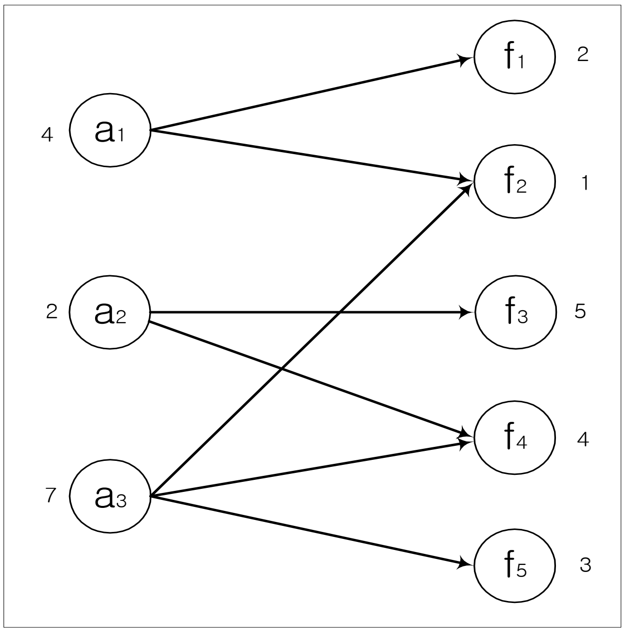
Rhys(1970: 200–207) and Balinski(1970: 230–231) established the equivalence of the selection problem to the minimal cut problem and thus made it possible to solve this problem by any maximal flow algorithm. The associated network is constructed by the following process:

1) add source node s and sink node t
2) link s to each activity node, and assign its benefit as arc capacity
3) link each facility node to t, and assign its cost as arc capacity
4) assign an infinity capacity to each of the original arcs.

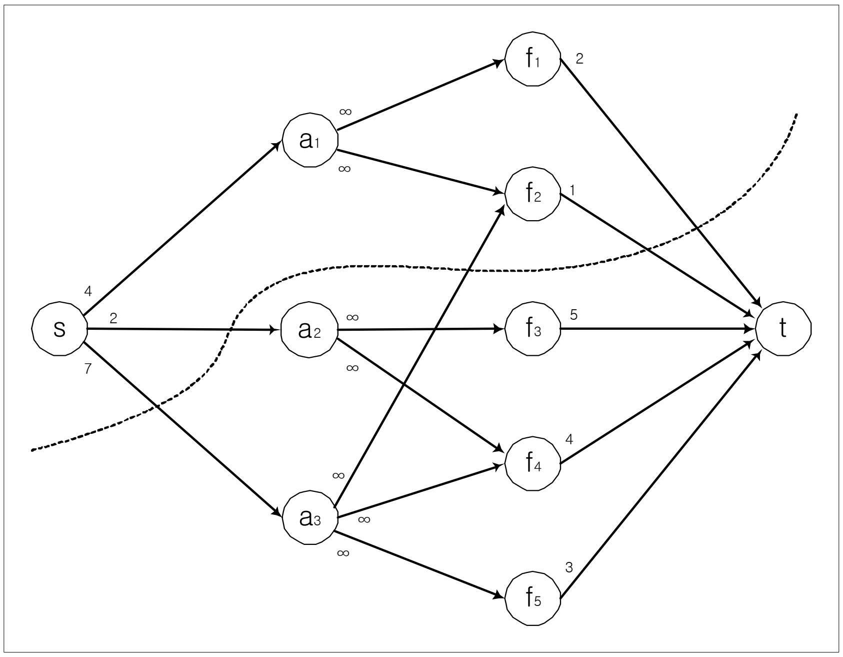
〈Example〉 Suppose we are given 3 activities (a_1, a_2, a_3) and 5 facilities (f_1, f_2, f_3, f_4, f_5) together with associated benefits (4, 2, 7) and costs (2, 1, 5, 4, 3). Activity a_1

can be selected only when facilities f_1 and f_2 are also chosen. The benefit 2 of a_2 may be obtained only if f_3 and f_4 are available. And prerequisites of a_3 are$\{f_2, f_4, f_5\}$. Our objective is to maximize the excess of benefit over cost while satisfying the relationships between activities and facilities.

This selection problem can be represented by a bipartite graph in 〈Figure 3〉. The example graph of a selection problem in 〈Figure 3〉 is converted to an equivalent graph of the maximal flow problem in 〈Figure 4〉. Then the optimal selection $\{a_1, f_1, f_2\}$ corresponds to those nodes in the source side of the minimal cut.



〈Figure 3〉 Example graph of a selection problem



〈Figure 4〉 Equivalent graph of the maximal flow problem to 〈Figure 3〉

Gusfield et al.(1987: 237-251) developed a specialized algorithm for the selection problem. Later, this algorithm has been improved by Ahuja et al.(1995: 906-933).

Rhys(1970: 200-207) illustrated some typical applications of the selection problem including the optimal selection of freight handling terminals. A particular pair of terminals permits service between those two terminals and a terminal may be concurrently used for multiple services. Each service is associated with net revenue and each terminal is associated with installation cost. In this case, activities are freight handling services and facilities are the terminals. We may have another example in international financing by replacing terminals with foreign offices and freight service with finance trade.

Eisner and Severance(1976: 619-635) presented an application of this problem to the record segmentation problem in large shared databases. They were motivated by the fact that often an entire data file is read and written while only a small subset of data items is requested by each user. To reduce the average cost of information retrieval, they proposed to partition data items into primary and secondary segments and store them on separate devices. Then only a small subset of highly useful data items in the primary segment will be retrieved by all database users. Their problem is to find the optimal assignment of data items to the primary segment and thus identify the users whose subsets are entirely in the primary segment by the given assignment. In this problem, users are activities and data items are facilities. Each data item is associated with its storage cost in the primary segment and each user is associated with the subset of data items retrieved by the user and the value of this retrieval.

Mamer and Smith(1982: 1328–1333) discussed another application of the selection problem. A standard kit of tools and repair parts must be specified to deal with a wide range of on–site repair jobs. The kit can be restocked between jobs, but its specific contents are fixed. If any one of required tools and parts is not available in a certain job, a "broken job" or incomplete service call arises and a penalty cost is charged. This penalty cost, combined with its occurrence rate, may be considered as the benefit or value of a successful job. Also we must consider an inventory cost for each tool or part in the kit. Their problem is to determine the optimal kit that maximizes the total job value minus total inventory cost. Here repair jobs correspond to activities and tools and parts correspond to facilities.

Ⅲ. Conclusion

The maximal flow problem can be formulated as a linear program. Thus it can be solved by any linear programming algorithm such as simplex method. Also, since the maximal flow problem can be viewed as a special case of minimal cost flow (or transshipment) problem, it can be solved by any minimal cost flow algorithm. Moreover, several specialized algorithms for the maximal flow problem have been developed and usually favored over general algorithms.

In this paper, we have shown that some problems defined in a network are computationally related to the maximal flow problem. These include the minimal cut problem, the maximal closure problem, and the selection problem. It is notable that all these problems can be solved by any maximal flow algorithm.

We noted that the minimal cut problem is the dual of the maximal flow problem in linear programs. Thus the minimal cut problem is often solved through its dual maximal flow problem since most algorithms for the maximal flow problem provide minimal cuts as a by-product. Alternatively, the minimal cut problem can be solved by the cut search algorithm or by a randomized algorithm.

We showed that any maximal closure problem can be converted to an equivalent maximal flow problem. Thus a maximal closure problem can be solved by any maximal flow algorithm. We also noted that it can be solved more efficiently by a proper algorithm.

We showed that the selection problem can be converted to an equivalent minimal cut problem. And thus can be solved by any maximal flow algorithm. We also mentioned that a specialized algorithm for the selection problem has been developed and improved.

Since those network problems discussed in this paper have so many applications, it is very important to obtain an efficient algorithm for each of them. Although various algorithms for these problems have been developed so far, it is still possible to find a new approach or to improve an existing algorithm. It also seems to be desirable to design a hybrid algorithm by employing different approaches to the computationally related problems. In this respect, it is very important to understand the equivalence between these related problems. This remains as a future research topic.

【References】

Ahuja, R., T. Magnanti, and J. Orlin, 1993, *Network Flows: Theory, Algorithms, and Applications*, Englewood Cliffs, NJ: Prentice-Hall.

Ahuja, R., J. Orlin, C. Stein, and R. Tarjan, 1995, "Improved Algorithms for Bipartite Network Flow," *Siam J. on Computing* 23, pp.906-933.

Bacharach, M., 1966, "Matrix Rounding Problems," *Management Science* 9, pp.732-742.

Balinski, M., 1970, "On a Selection Problem," *Management Science* 17, pp.230-231.

Christiano, P., J. Kelner, A. Madry, D. Spielmanz, and S. Teng, 2010, "Electrical Flows, Laplacian Systems, and Faster Approximation of Maximum Flow in Undirected Graphs," Working Paper.

Eisner, M. and D. Severance, 1976, "Mathematical Techniques for Efficient Record Segmentation in Large Shared Databases," *JACM* 23, pp.619-635.

Faaland, B., K. Kim, and T. Schmitt, 1990, "A New Algorithm for Computing the Maximal Closure of a Graph." *Management Science* 36, pp.315-331.

Faaland, B. and T. Schmitt, 1987, "Scheduling Tasks under Due Dates in a Fabrication/Assembly Process," *Operations Research* 35, pp.378-388.

Federgruen, A. and H. Groenevelt, 1986, "Preemptive Scheduling of Uniform Machines by Ordinary Network Flow Techniques," *Management Science* 32, pp.341-349.

Ford, L. and D. Fulkerson, 1956, "Maximal Flow Through a Network," *Canadian Journal of Mathematics* 8, pp.399-404.

Ford, L. and D. Fulkerson, 1962, *Flowsin Networks*, Princeton: Princeton University Press.

Goldberg, A. and S. Rao, 1998. "Beyond the flow decomposition barrier," *J.ACM* 45, pp.783 - 797.

Gusfield, D., C. Martel, and D. Fernandez-Baca, 1987, "Fast Algorithms for Bipartite Network Flow," *Siam J. on Computing* 16, pp.237-251.

Hochbaum, D., 2008, "The Pseudoflow Algorithm: A New Algorithm for the Maximum-Flow Problem," *Operations Research* 56, pp.992-1009.

Hochbaum, D., 2004, "Selection, Provisioning, Shared Fixed Costs, Maximum Closure, and Implication on Algorithmic Methods Today," *Management Science* 50, pp.709–723.

Johnson, T, 1968, "Optimum Open Pit Mine Production Scheduling." Research Report ORC 68–11, Operations Research Center, University of California at Berkeley.

Karger, D. and C. Stein, 1996, "A New Approach to the Minimal Cut Problem," *J.A.C.M.*,43, pp.601–640.

Lawler, E., 1976, *Combinatorial Optimization: Network sand Matroids*, NewYork, Holt, Rinehart and Winston.

Lerchs, H. and I. Grossmann, 1965, "Optimum Design of Open–PitMines, "*Canadian Mining and Metallurgical Bulletin* 58, pp.47–54.

Mamer, J. and S. Smith, 1982, "Optimizing Field Repair Kits Based on Job Completion Rate," *Management Science* 28, pp.1328–1333.

Orlin, J., 1987, "Optimal Weapons Allocation Against Layered Defenses," *Naval Research Logistics Quarterly* 34, pp.605–617.

Phillips, S. and M. Dessouky, 1979, "The Cut Search Algorithm with Arc Capacities and Lower Bounds," *Management Science* 25, pp.396–404.

Picard, J., 1976, "Maximal Closure of a Graph and Applications to Combinatorial Problems," *Management Science* 22, pp.1268–1272.

Picard, J. and M. Queyranne, 1982, "Selected Applications of Minimum Cuts in Networks," *INFOR* 20, pp.394–422.

Picard, J. and H. Ratliff, 1978, "A Cut Approach to the Rectilinear Distance Facility Location Problem," *Operations Research* 26, pp.422–433.

Ramanathan, A. and C. Colbourn, 1987, "Counting Almost Minimum Cutsets with Reliability Applications," *Math.* Prog.39, pp.253–261.

Rhys, J., 1970, "A Selection Problem of Shared Fixed Costs and Network Flows," *Management Science* 17, pp.200–207.

Shapiro, J. 1979, *Mathematical Programming: Structures and Algorithms*, NewYork, JohnWiley & Sons.

Sidney, J., 1975, "Decomposition Algorithms for Single–Machine Sequencing with Precedence Relations and Deferral Costs," *Operations Research* 22, pp.283–298.

Stone, H., 1977, "Multiprocessor Scheduling with the Aid of Network Flow Algorithms," *IEEETransactionson Software Engineering* 3, pp.85–93.

Tachefine, B. and F. Soumis, 1997, "Maximal Closure on a Graph with Resource Constraints," *Computers Ops. Res.* 24, pp.981–990.

Weingartner, H., 1966, "Capital Budgeting of Interrelated Projects: Survey and Synthesis," *Management Science* 12, pp.485–516.

〈국문초록〉

최대흐름문제와 계산적인 측면에서 관련된 문제들*

김기석**

최대흐름문제는 주어진 네트워크에서 최대값을 갖는 흐름을 발견하는 문제이다. 이 문제는 선형계획모형 또는 최소비용흐름모형으로 표현될 수 있으므로, 이들을 위한 어떠한 해법으로도 풀 수 있다. 또한 최대흐름문제 만을 위한 고유한 해법들이 다수 개발되어 있어 다른 해법들보다 선호되고 있다. 본 논문에서는 최적해의 계산적인 측면에서 최대흐름문제와 관련된 네트워크 문제들을 발굴하여 분석하였다. 이처럼 관련된 문제들로는 최소절단문제, 최대폐쇄문제, 활동설비선정문제가 확인되었다. 이들 세 문제는 모두 최대흐름문제로 변형하여 최적해를 구할 수 있음을 보여주었다. 또한 이들 각각을 위한 고유 해법들도 개발되어 있음을 소개하였다.

주제어 : 최대흐름문제, 최소절단문제, 최대폐쇄문제, 활동설비선정문제

* 이 논문은 2009년도 부산대학교 인문사회연구기금의 지원을 받아 연구되었음

** 부산대학교 경영대학 교수(mgtkkim@pusan.ac.kr)