An Integer Programming Approach to the Path Selection Problems

Sungsoo Park [1], Deokseong Kim [1], Kyungsik Lee[2]
[1] Department of Industrial Engineering, Korea Advanced Institute of Science and Technology
Address : 373-1 Guseong-dong, Yuseong-gu, Daejeon 305-701, Republic of Korea
Phone: 82-42-869-3121, fax: 82-42-869-3110
Email:{sspark, kristal}@kaist.ac.kr
[2] School of Industrial Information & Systems Engineering, Hankuk University of Foreign Studies
Address : 89 Wangsan-ri, Mohyeon-myon, Yongin-si, Kyongki-do 449-791, Republic of Korea
Phone: 82-31-330-4941, fax: 82-31-330-4093
Email:globaloptima@hufs.ac.kr

April 2003

Abstract

We consider two types of path selection problems for arc capacitated network. Given an arc capacitated network and a set of commodities, one problem is to find a subset of commodities to be routed and an assignment of them to the paths so that profit is maximized. The other problem is to route all given commodities in the network so that cost is minimized. Bifurcation of flow is not allowed in both cases.

We formulate the problems as integer programming models and solve them. Column generation technique to solve the linear programming relaxation is proposed with two types of columns. To obtain an optimum integer solution for the problems, we propose a branching strategy in the branch-and-price scheme. Computational experiments show that the algorithm gives optimal solutions within reasonably small computing time.

keywords Routing, Network Models, Integer Multicommodity Flow, Path Selection, Integer Programming

1 Introduction

We consider two classes of path selection problems arising from transportation, communication, and production. They are defined over an arc-capacitated directed or undirected network. When we are given a set of commodities with their revenue and demand quantities and a unit flow cost on each arc, we need to select a subset of commodities to be routed and assign them to node-simple paths with the objective of maximizing profit. When we need to select all commodities, the problem becomes to cost minimization problem. For convenience, we call the first case the Path selection problem for a Subset of Commodities (PSC problem) and the second case the Path selection problem for All Commodities (PAC problem). Applications of these problems arise in the bandwidth packing problems [4] and package flow problems [2].

There have been numerous studies for these problems. Laguna and Glover [7] and Anderson et al. [1] developed tabu search methods to solve the bandwidth packing problem. In addition, Laguna and Glover [7] proposed an integer programming (IP) formulation of the problem and solved some instances using a commercial optimization package. According to their report, IP approach using a commercial optimization package is not practical since it takes too much CPU time to solve the (restricted) IP formulation.

Parker and Ryan [11] proposed a column generation algorithm for the bandwidth packing problem. They used the technique to solve the linear programming (LP) relaxation of the problem that has exponentially many variables. But, they did not incorporate any cutting-planes. To obtain an integer optimal solution, they used branch-and-bound technique with a branching strategy different from us. They reported that their algorithm could find near optimal solutions within small time bound.

Park et al. [12] proposed another integer programming approach to solve the problem. They used the delayed column generation technique and strong cutting plane (row) generation technique simultaneously to strengthen the LP relaxation. To obtain an integer optimal solution, they used branch-and-bound and branch-and-cut technique.

Barnhart et al. [2] proposed an integer programming approach to the integer multi-commodity flow problem. Their algorithm is similar to Park et al. [12] but with the difference in branching rule and other details. In their study,

they called the branch-and-bound algorithm incorporated with combined row and column generation algorithm branch-and-price-and-cut algorithm. Other applications of combined row and column generation in solving integer programming problems can be found in Nemhauser and Park [9] and Mehrotra [8] and those presented in the survey of Desrosiers et al. [5].

Regardless of the problem types, the solution approaches to solve the problems are very similar to each other. In this study, we propose a procedure to solve these problems. We use the column generation technique as above, but with two different kinds of column types. This approach also strengthens the LP relaxation bound at least as tight as the strong cutting plane approach. When the approach fails to obtain an integer solution, we use the branch-and-price approach. To do this, we propose a branching strategy. We test the proposed algorithm on several problem instances. The computational results show that our approach performs very well to find the optimal solutions within small time limits.

The paper is structured as follows. Section 2 presents the formulations of the problem. Section 3 presents the column generation procedure to solve the LP relaxation. Section 4 describes the overview of our algorithm. Computational results are shown in section 5. Finally, concluding remarks are given in Section 6.

2 Formulation of the Problem

In this section, we present the formulations of the problems. Given an arc-capacitated directed or undirected network $N = (V, E)$, the set of commodities, revenue and demand quantity of each commodity, and the unit flow cost of each commodity on each arc, we consider the maximization of total profit while satisfying the arc capacity restrictions on each arc. First, we give some notation to be used in the formulation of the problems.

Notation

K **:** set of commodities

$P(k)$ **:** set of (s,t)-paths, where s is the source and t is the destination of commodity k, where $k \in K$

$P(k;e)$ **:** set of paths in $P(k)$ that pass through arc e, where $e \in E, k \in K$

$E(p)$ **:** set of arcs of which path p consists

c_e^k **:** unit flow cost of commodity k on arc e, where $k \in K, e \in E$

b_e **:** capacity of arc e, where $e \in E$

v_k **:** revenue of commodity k, where $k \in K$

r_k **:** demand quantity of commodity k, where $k \in K$

w_p^k **:** profit obtained if we select commodity k and assign it to path p, where $w_p^k = v_k - r_k \sum_{e \in E(p)} c_e^k$

Using the notation, PSC problem can be formulated as the following using path variables.

$$
\begin{aligned}
\text{(PSCP)} \quad \max \ & \sum_{k \in K} \sum_{p \in P(k)} w_p^k y_p^k \\
\text{s.t.} \ & \sum_{p \in P(k)} y_p^k \le 1, \ \text{for all } k \in K && (1) \\
& \sum_{k \in K} \sum_{p \in P(k;e)} r_k y_p^k \le b_e, \ \text{for all } e \in E, k \in K && (2) \\
& y_p^k \in \{0,1\} \ \text{for all } p \in P(k), k \in K.
\end{aligned}
$$

Without loss of generality, we assume all data are integral. The decision variable y_p^k is 1 if commodity k is chosen and assigned to path p, 0 otherwise. Constraints (1) ensure that at most one path can be selected for each commodity. Constraints (2) mean that total amount of demand quantities that flows on each arc is less than or equal to the capacity of the arc. Note that the problem mentioned above is NP-hard. This can be seen by reducing a 0-1 knapsack problem to this problem. For more details, refer Parker and Ryan [11] and Park et al. [12].

When we need to select all commodities, the sense of constraints (1) will be changed to equality. For convenience, we call the corresponding model PACP. In this case, the constraints ensure that exactly one path should be selected for each commodity. Note that the sum of revenues is constant. Therefore, the problem becomes minimizing the total flow cost.

Now, we present alternative formulations of the path selection problems. The basic idea is to decompose the problems into a master problem and two subproblems. The master problem is constructed by a set of restricted number of columns that indicate some grouping configurations. The subproblems are constructed to generate columns to be entered into the master problem.

First, to formulate the master problem, we define a concept of a commodity-set pattern. A commodity-set pattern g is a set $K(g)$ of the commodities such that $\sum_{k\in K(g)} r_k \le b_e$ for arc $e \in E$. Let $G(e)$ be the set of commodity-set patterns for arc $e \in E$ and $G(e;k)$ be the set of commodity-set patterns in $G(e)$ that contain commodity k. Then, we can reformulate PSCP as the following:

$$\text{(MPS)} \quad \max \sum_{k\in K}\sum_{p\in P(k)} w_p^k y_p^k$$

$$\text{s.t.} \sum_{p\in P(k)} y_p^k \le 1, \text{ for all } k \in K \tag{3}$$

$$\sum_{g\in G(e)} z_e^g \le 1, \text{ for all } e \in E \tag{4}$$

$$\sum_{p\in P(k;\ e)} y_p^k \le \sum_{g\in G(e;\ k)} z_e^g, \text{ for all } e \in E, k \in K \tag{5}$$

$$z_e^g \in \{0,1\}, y_p^k \in \{0,1\} \text{ for all } g \in G(e), e \in E, p \in P(k), k \in K.$$

The decision variable $z_e^g = 1$ if commodity-set pattern g is selected for the arc e, 0 otherwise. Constraints (4) ensure that at most one commodity-set pattern can be selected for each arc. Constraints (5) mean that commodity k can pass through arc e of path p if commodity-set pattern g, containing commodity k, is selected for arc e. When we need to select all commodities, the sense of constraints (3) should be changed to equality. The others are also the same as mentioned above. For convenience, we call the corresponding model MPA.

Note that LP relaxation of MPS provides a bound at least as strong as the bound provided by the LP relaxation of PSCP. Note that MPA also has the same results.

Generally, there are exponentially many paths and exponentially many commodity-set patterns for an instance of each problem, that is, MPS has exponentially many decision variables. It is thus impractical to enumerate all the possible paths and commodity-set patterns to solve LP relaxation of MPS with all the decision variables at hand. However, it can be solved efficiently by using the column generation technique developed by Gilmore and Gomory [6].

3 Column Generation Problems

In this section, we give an explanation of column generation problem. We only explain the case of PSC problem since the column generation problem for case of PAC problem is the same as that of PSC problem.

First, we denote LP relaxation of MPS by MPL. Let U be the union of all possible paths ($P = \bigcup_{k\in K} P(k)$) and all possible commodity-set patterns ($G = \bigcup_{e\in E} G(e)$). We assume that a subset U' of U is given. Replacing U by U' in MPL yields the restricted linear programming MPL$'$ whose solutions are suboptimal to MPL. Then, using the optimal dual solution returned by the simplex method in the current MPL$'$, we search for profitable column (path or commodity-set pattern) whose addition to MPL$'$ may result in an increase of the optimal objective value of MPL$'$. If there are no such columns, the solution at hand is an optimal solution to MPL. Otherwise, we add the column to MPL$'$, and then repeat the above process.

Note that we need a feasible basis for MPL to use the column generation method. If it is difficult to find an initial feasible solution, we can introduce artificial variables with big cost coefficient. In the following, we mention how to find the profitable columns.

Given a feasible basis to MPL, we need to generate columns to enter the basis. Let $(\bar\alpha, \bar\beta, \bar\gamma)$ be the dual optimal solution associated to the constraints in (3), (4) and (5), respectively. Then, we may write the optimality condition for MPL as follows:

$$\min\{\sum_{e\in E(p)} \bar\gamma_e^k - w_p^k | p \in P\} \ge -\bar\alpha_k \tag{6}$$

$$\max\{\sum_{k\in K(g)} \bar\gamma_e^k | g \in G\} \le \bar\beta_e \tag{7}$$

Using condition (6), we can derive the first column generation problem (SP1(k)) associated to commodity k. The problem finds the shortest path from node s to node t, where s is the source and t is the destination of commodity k, respectively. Since the arc weights are nonnegative, SP1(k) can be solved efficiently by Dijkstra's algorithm [3]. If the resulting length of the shortest path is less than $-\bar\alpha_k$, the path can be added to the current formulation. Otherwise, no column is generated with respect to commodity k.

Using condition (7), we can derive the second column generation problem (SP2(e)) associated to arc e. The problem is a 0-1 knapsack problem for the arc e. The problem can be solved using dynamic programming [10] with time complexity $O(nb)$ where n is the number of variables and b is the capacity of the knapsack. If the resulting weight of the knapsack is greater than $\bar{\beta}_e$, the commodity-set pattern can be added to the current formulation. Otherwise, no column is generated with respect to arc e.

4 Overview of the Algorithm

4.1 Overview

In this section, we give a brief and overall explanation of our algorithm. We only explain the case of PSC problem since the algorithm for the case of PAC problem is the same as that of PSC problem. First, we construct initial formulation of MPL without the constraints (5). We call the formulation INIT.

After solving the INIT, we decide if condition (6) is satisfied. If it is not, new columns for path variables are generated and added to INIT. If the present solution satisfies condition (6), i.e., no more columns for path variables need to be added to INIT, we proceed to find a subset of constraints (5), which are violated by the current fractional solution, to be added. In this way, the augmented formulation ALP has been obtained.

After getting ALP, we solve ALP and decide if the condition (7) is satisfied. If it is not, new columns for commodity-set patterns are generated and added to ALP. If the present solution satisfies condition (7), no more columns for commodity-set patterns need to be added to ALP.

Then, we go though the same procedure as we do after the initial formulation of MPL is obtained. If the solution of ALP is dual feasible, we generate needed columns until it is optimally solved. Also for ALP with its present solution, we generate needed inequalities to be added to ALP.

When no more columns can be generated, we check if the solution obtained by solving the last LP is integral. If we have obtained an integral solution, we are done with an optimal solution of PSC problem. Otherwise, we have to initiate the branch-and-price procedure to find an integer solution. The procedure incorporates the column generation approach within the branch-and-bound scheme. The procedure yields an optimal solution of PSC problem.

4.2 Branching Strategy

In this study, we consider some branching rules. First, we consider the case of PSC problem. As the branching rules of this problem, we may consider 3 branching rules. The first rule is that of Parker and Ryan [11]. The second is that of Park et al. [12]. Now we consider the third rule, which shows better performance than the other rules in our test. In the computational results, we only report the result of the third branching rule because of its better performance. The results of the others are not reported. Our branching rule consists of 2 stages. In the first stage, we decide whether to select a commodity or not for the commodity that take on fractional value. In the second stage, we select a set of paths of the commodity whose flow on any arcs takes on fractional values for the selected commodity. In this stage, we used modified version of the Bahnhart et al. [2]. In the directed network, we can use directly the branching rule proposed by Barnhart et al. [2]. But, in the undirected network, we need to modify their branching rule as an undirected version.

When we need to select all commodities to solve the case of PAC problem, we do not perform the stage 1. Therefore, we only need to perform the stage 2.

5 Computational Results

We tested the proposed algorithm on some randomly generated problems. The underlying networks are devised to reflect the characteristics of the realistic networks according to the remarks shown in Anderson et al. [1]. In the following, we mention the characteristics of the general problems and then show the computational results.

The nodes of a generated network are chosen randomly in two-dimensional Euclidean plane [50, 50]. These nodes are connected by undirected arcs which are chosen randomly. The arcs are chosen so that nearer nodes are more likely to be connected than distant nodes. All of the networks are generated as above such that it satisfies the factor of connectivity.

For each network, we generated 10 problem instances of path selection problems that have different arc capacities and commodity tables. In the case of PSC problem, the capacities of the arcs are randomly chosen in the range from 10 to 50. In the case of PAC problem, the capacities of the arcs are randomly chosen in the range from 20 to 70. Commodity tables are also randomly generated. Each commodity has an origin, a destination, a demand quantity, and revenue. Demand quantity of each commodity was generated between 1 and 20. We managed revenue of a

Table 1: Computational results for PSC problem ($|V| = 30, |E| = 50$)

No	$\|K\|$	#COL1	#COL2	#CUT	#LP	Gap(%)	#BB	Time
1	100	497	1723	3900	1225	0.45	162	88.23
2	102	339	721	3774	348	0.15	10	14.06
3	94	234	366	3008	186	0.05	14	4.91
4	84	161	199	2688	93	0.08	2	1.65
5	93	210	401	2790	330	0.21	34	6.98
6	99	207	254	3168	122	0.09	6	2.93
7	103	184	140	3193	42	0.00	0	1.01
8	84	118	157	1932	82	0.10	4	1.08
9	77	176	372	2387	288	0.72	42	5.18
10	79	185	221	3002	58	0.13	2	1.33
Avg	91.5	231.1	455.4	2984.2	277.4	0.20	27.6	12.74
Max	103	497	1723	3900	1225	0.72	162	88.23
Min	77	118	140	1932	42	0.00	0	1.01

Table 2: Computational results for PAC problem ($|V| = 30, |E| = 78$)

No	$\|K\|$	#COL1	#COL2	#CUT	#LP	Gap(%)	#BB	Time
1	41	279	542	1353	365	0.70	50	5.53
2	48	227	268	1536	88	0.00	2	1.21
3	37	121	136	703	71	0.00	0	0.45
4	49	128	140	980	124	0.25	8	1
5	43	240	286	1462	95	0.00	0	1.38
6	52	461	698	2756	253	0.10	8	11.24
7	34	75	44	340	31	0.00	0	0.13
8	41	212	310	1066	181	0.08	16	1.84
9	44	145	145	748	106	0.10	10	0.8
10	41	279	542	1353	365	0.70	50	5.32
Avg	43	216.7	311.1	1229.7	167.9	0.19	14.4	2.89
Max	52	461	698	2756	365	0.70	50	11.24
Min	34	75	44	340	31	0.00	0	0.13

commodity to be from 100 to 1000 in scale of tens. The commodities are randomly generated among all possible pairs of nodes. The number of commodities is also randomly determined, proportional to the sizes of the network considered. For computational test, we used CPLEX callable library version 7.0 as an LP solver.

The computational results on the 2 networks are summarized in table 1-2. Table 1 shows the results for the case of PSC problem and Table 2 shows the results for the case of PAC problem.

In those tables, the headings $|K|$, #COL1, #COL2, #CUT and #LPs refer to the number of commodities, the number of generated columns of path type, the number of generated columns of commodity-set pattern type, the number of constraints added among the set of constraints (5), and the number of calls to LP solver in the algorithm, respectively. All data are accumulated up to the end of the whole procedure. Let LP denote the objective value obtained by solving the final linear programming formulation before the branch-and-bound procedure and OPT denote the value of the optimal integer solution. Gap(%) is defined as (LP-OPT)/OPT $\times$ 100.

The heading #BB refers to the numbers of nodes generated in the branch-and-price procedure. Finally, the heading Time refers to the accumulated CPU time in seconds needed to solve the problem. All problems are solved on Pentium PC (866MHz).

Note that the number of columns of path type generated in the procedure does not exceed 5 times the number of commodities on average. The number of columns of commodity-set pattern type generated does not exceed twice the number of columns of path type on average. The Gap is relatively small and usually within 1 percent. The column #BB shows that it does not exceed 50 in the number of enumeration nodes in average. The CPU times needed to solve the problems are also relatively small and do not exceed a few minutes.

6 Concluding Remarks

In this paper, we proposed an algorithm to solve the IP formulations of the path selection problems. We used the delayed column generation technique with two kinds of column types to strengthen the LP relaxation bound of the the problems. The technique has been used to solve the LP relaxation of the problems efficiently. Moreover, we proposed a branching rule for these problems. The computational results showed that the proposed algorithm gives optimal solutions in small running time.

References

[1] C.A. Anderson, F. Fraughnaugh, M. Parker, and J. Ryan, Path Assignment for Call Routing: An Application of Tabu Search, in F. Glover et al. (Eds.), Annals of Operations Research, Vol. 41, 1993.

[2] C. Barnhart, C. Hane and P.H. Vance, Using Branch-and-Price-Cut To Solve Origin-Destination Integer Multicommodity Flow Problems, Operations Research, 48, 318-326, 2000.

[3] J.A. Bondy and U.S.R. Murty, Graph Theory with Applications, Elsevier, New York, 1976.

[4] L.A. Cox, I. Davis, and Y. Qie, Dynamic Anticipatory Routing in Circuit-Switched Telecommunications Networks, in L. Davis (Ed.), Handbook of Genetic Algorithms, Van Nostrand Reinhold, New York, 1991.

[5] J. Desrosiers, Y. Dumas, M.M. Solomon, and F. Soumis, Time constrained routing and scheduling In Handbook in Operations Research and Management Science. M.E. Ball, T.L. Magnanti, C. Monma, G.L. Nemhauser (eds.). Elsevier, Amsterdam, 1995.

[6] P.C. Gilmore and R.E. Gomory, "A Linear Programming Approach to the Cutting-stock Problem", Operations Research, 9, 849-859, 1961.

[7] M. Laguna and F. Glover, "Bandwidth Packing: A Tabu Search Approach", Management Science, 39, 492-500, 1993.

[8] A. Mehrotra, "Constrained Graph Partitioning: Decomposition, Polyhedral Structure and Algorithms". Ph.D. Thesis, Georgia Institute of Technology, Atlanta, GA, 1992.

[9] G.L. Nemhauser and S. Park, "A Polyhedral Approach to Edge Coloring". Operations Research Letters, 10, 315-322, 1991.

[10] G.L. Nemhauser and L.A. Wolsey, Integer and Combinatorial Optimizatin, John Wiley & Sons, 1988.

[11] M. Parker and J. Ryan, "A Column Generation Algorithm for Bandwidth Packing". Telecommunication Systems, 2, 185-196, 1994.

[12] K. Park, S. Kang, and S. Park, "An integer programing approach to the bandwidth packing problem". Management Science, 42, 1277-1291, 1996.