대한산업공학회/한국경영과학회 2002 춘계공동학술대회
한국과학기술원(KAIST) 2002년 5월 3일-4일

Hop-constrained multicast route packing with bandwidth reservation

강장하, 박성수
한국과학기술원 산업공학과
대전 광역시 유성구 구성동 373-1 (우편번호:305-701)
Tel.: 042)869-3121, Fax.: 042)869-3110, E-mail: sspark@kaist.ac.kr

Abstract

Multicast technology allows the transmission of data from one source node to a selected group of destination nodes. Multicast routes typically use trees, called multicast routing trees, to minimize resource usage such as cost and bandwidth by sharing links. Moreover, the quality of service (QoS) is satisfied by distributing data along a path having no more than a given number of arcs between the root node of a session and a terminal node of it in the routing tree. Thus, a multicast routing tree for a session can be represented as a hop constrained Steiner tree.

In this paper, we consider the hop-constrained multicast route packing problem with bandwidth reservation. Given a set of multicast sessions, each of which has a hop limit constraint and a required bandwidth, the problem is to determine a set of multicast routing trees in an arc-capacitated network to minimize cost. We propose an integer programming formulation of the problem and an algorithm to solve it. An efficient column generation technique to solve the linear programming relaxation is proposed, and a modified cover inequality is used to strengthen the integer programming formulation.

1. Introduction

Multicast technology can be used to enable the use of multimedia applications such as voice and video transmission. It allows the transmission of data from one source node to a set of destination nodes. Multicast routes typically use trees [2], called multicast routing trees, to minimize resource usage such as cost and bandwidth by sharing links. Moreover, the quality of service (QoS) is satisfied by distributing data along a path having no more than a given number of arcs between the root node of a session and a terminal node of it in the routing tree [12], [5], and [7]. Thus, a multicast routing tree for a session can be represented as a hop constrained Steiner tree.

The problem considered in this paper is to generate a set of multicast routing trees and to assign them to an arc-capacitated network. Two objectives have been considered for the problem: 1) to minimize the maximum utilization or congestion [2], and 2) to minimize the total installation cost [8]. Utilization for each arc is the ratio of the traffic loaded on it to the capacity of it. As the utilization for an arc increases, congestion may occur. Therefore, by minimizing the maximum utilization, we can reduce the maximum congestion of a network. While, given a target utilization, we can generate a set of routes to minimize the total installation cost. What we consider in this paper is this. By the way, as the maximum utilization or the installation cost is reduced, QoS for each session generally gets poorer. This can be avoided by introducing the hop constraint, that is, by generating a hop constrained Steiner tree. Therefore, we consider the hop-constrained multicast route packing problem with bandwidth reservation (HMRP).

Given a set of multicast sessions, each of which has a hop limit constraint and a required bandwidth, the problem is to determine a set of multicast routing trees in an arc-capacitated network to minimize cost. We propose an integer programming formulation of the problem and an algorithm to solve it. An efficient column generation technique to solve the linear programming relaxation is proposed, and a modified cover inequality is used to strengthen the integer programming formulation. For details of integer programming and column generation technique, refer to [1], [9], and [10].

The paper is structured as follows. Section 2 presents the formulation of the problem. In section 3, the column generation procedure to solve the LP relaxation is given, and valid inequalities used to strengthen the initial formulation are given in section 4. Section 5 presents the column generation procedure to solve the augmented LP with cuts. The overview of our algorithm is described in section 6.

2. Problem description and Formulation

In this section, we present the formulation of HMRP. Given an arc-capacitated network $G=(V,A)$ the set of multicast sessions, the capacity requirement and the hop limit for each session, HMRP finds a set of routing trees, one per session and assigns them to the network to minimize the total installation cost while satisfying the bandwidth capacity restrictions on each arc.

First, we give some notation to be used in the formulation of the problem.

a index for arc, $a \in A$,
c_a cost for the unit flow on arc a, where $a \in A$,
b_a capacity of arc a, where $a \in A$,
i index for multicast session, where $i=1,\cdots,m$
v_i^0 root node of session i, where $i=1,\cdots,m$
$V_i \subset V$ the set of destination nodes of session i, where $i=1,\cdots,m$
h_i hop limit of session i, where $i=1,\cdots,m$
w_i bandwidth requirement of session i, where $i=1,\cdots,m$
t index for multicast routing tree,
T_i set of multicast routing trees for session i, where $i=1,\cdots,m$
$T_i(a) \subset T_i$ set of routing trees for session i containing arc a, where $i=1,\cdots,m$ and $a \in A$,
$A(t)$ set of arcs of which routing tree t consists.

Using the above notation, HMRP can be formulated as follows:

$$\text{(HMRP)} \min \sum_{i=1}^{m} \sum_{t \in T_i} \bar{c}_{it} x_{it} \quad (1)$$

$$\text{s.t} \quad \sum_{t \in T_i} x_{it} = 1 \quad \forall i=1,\cdots,m \quad (2)$$

$$\sum_{i=1}^{m} \sum_{t \in T_i(a)} w_i x_{it} \le b_a \quad \forall a \in A, \quad (3)$$

$$\text{and} \quad x_{it} \in \{0,1\}, \quad \forall t \in T_i, \quad i=1,\cdots,m \quad (4)$$

where $\bar{c}_{it} = w_i \sum_{a \in A(t)} c_a$ Note that $\bar{c}_{it}$ is the installation cost of routing tree t for session i. Here, x_{it} is 1 if and only if multicast session i uses routing tree t; otherwise, it is 0. Constraints (2) ensure that each multicast session uses exactly one routing tree. Constraints (3) mean that the total amount of traffic loaded on an arc should not be greater than the capacity of it. A linear programming (LP) relaxation of the formulation can be obtained by removing integrality restrictions in (4). Let HMRPL be the LP relaxation of HMRP. HMRPL has exponentially many variables. However, we can solve the LP relaxation efficiently by the (delayed) column generation technique, which was successfully used in solving the bandwidth packing problem, which has a structure similar to HMRP; see [10]. In the following section, we describe the procedure to solve HMRPL.

3. Solving the LP relaxation

In this section, we propose an efficient procedure for solving HMRPL based on the column generation technique. Considering a subset of multicast routing trees $\bar{T}_i \subset T_i$ for each $i=1,\cdots,m$ we construct a restricted LP relaxation, referred to as RHMRPL, in the procedure. Let α_i be the dual variable associated to the constraint in (2) for each session i. Let β_a be the dual variable associated to the constraints in (3) for each arc a. Also let α^*, β^* be the values of the dual variables returned by the simplex method in the current RHMRPL. Then the column generation problem associated to session i can be formulated as follows.

$$\text{SP}(i) \min \ w_i \sum_{a \in A(t)} (c_a - \beta_a^*) \quad (5)$$

$$\text{s.t.} \quad t \in T_i. \quad (6)$$

Here, $\beta_a \le 0$ for each $a \in A$. Therefore, the objective coefficients are all nonnegative. Note that SP(i) is the problem that finds a minimum cost hop constrained Steiner tree, which is known to be NP-hard [5]. We will introduce how to solve the hop constrained Steiner tree problem, in section 6.

If the resulting cost of the routing tree is less than α_i^*, the routing tree can be added to the current restricted formulation. Otherwise, no column is generated with respect to session i.

4. Cutting-plane

In this section, we derive a class of cutting-planes used to enhance LP bound. First, we will consider a point set S defined by the following constraints:

$$\sum_{t \in T_i(a)} x_{it} \le 1, \quad \forall i = 1, \cdots, m \tag{7}$$

$$\sum_{i=1}^{m} \sum_{t \in T_i(a)} w_i x_{it} \le b_a \tag{8}$$

$$\text{and } x_{it} \in \{0,1\}, \quad \forall t \in T_i(a), \quad i = 1, \cdots, m \tag{9}$$

The above constraints can be obtained from HMRPL by selecting the inequality (3) related to an arc a, the inequalities (2), and the set of variables corresponding to the routing trees using the arc with the integrality constraints (4) related to the variables.

Note that the variables in (8) can be partitioned into disjoint subsets, one per session and that the variables in the same subset have the same coefficient. Therefore, S can be viewed as the set of a feasible solution to the multi-choice knapsack problem (MCKP), which is the knapsack problem where the set of items are decomposed into disjoint subsets, and at most one item per subset can be chosen. For the polyhedral results on MCKP, see [6], and [11]. Park, et al. [10] showed that every facet-defining inequalities for the convex set of S, referred to as conv(S), can be generated from the facets of the corresponding 0-1 knapsack polytope, referred to as conv(KP), which is defined by the convex set of the solutions satisfying the following constraints:

(KP)

$$\sum_{i=1}^{m} w_i y_i \le b_a \text{ where } y_i \in \{0,1\}, \forall i = 1, \cdots, m \tag{10}$$

The following proposition says this.

Proposition 1. (*Park, et al. [10]*)
$\sum_{i=1}^{m} \pi_i \sum_{t \in T_i(a)} x_{it} \le \pi_0$ is a facet-defining inequalities for conv(S) if and only if $\sum_{i=1}^{m} \pi_i y_i \le \pi_0$ is a facet-defining one for conv(KP).

A valid inequality for the knapsack polytope usually generated from a minimal cover inequality by performing a sequential lifting procedure; for details refer to [9].

We say that a subset $C = \{1, \cdots, m\}$s a cover when $\sum_{i \in C} w_i > b_a$ and that a cover C is minimal when and $\sum_{i \in C \setminus \{k\}} w_i \le b_a$, $\forall k \in C$. Given such a minimal cover C, the minimal cover inequality is written as follows:

$$\sum_{i \in C} y_i \le |C| - 1 \tag{11}$$

We will introduce the sequential lifting procedure in the section 6.

Now, we introduce the separation problem for the minimal cover inequality, which is defined as follows: (SEP) Given a solution x^* of current HMRPL, to generate the minimal cover inequality for S associated with arc a that is most violated by x^*. Then, the separation problem can be formulated as follows:

$$\text{(SEP) min } \zeta = \sum_{i=1}^{m} (1 - y_i^*) z_i \tag{12}$$

$$\text{s.t. } \sum_{i=1}^{m} w_i z_i > b_a, \tag{13}$$

$$\text{and } z_i \in \{0,1\} \quad \forall i = 1, \cdots, m \tag{14}$$

where $y_i^* = \sum_{t \in T_i(a)} x_{it}^*$

Suppose ζ is greater than or equal to 1, then x^* satisfies all of the minimal cover inequality for S associated with arc a. Otherwise, the optimal solution to SEP provides a most violated minimal cover inequality. Note that all of the optimal solution to SEP may not be minimal covers because z_i such that $1 - y_i^* = 0$can always be 1 in the solution to SEP without increasing the objective value. We will introduce how we can always generate a minimal cover in the section 6.

5. Augmented LP (ALP)

In this section, we consider the column generation procedure to the problem obtained by augmenting the valid inequalities obtained from S to HMRPL. Let $C(a)$ be the set of indices for the valid inequalities generated from S related to an arc a. Then the augmented LP relaxation is as follows:

(AHMRPL)

$$\min \quad \sum_{i=1}^{m} \sum_{t\in T_i} \bar{c}_{it} x_{it} \qquad (15)$$

$$\text{s.t.} \quad \sum_{t\in T_i} x_{it} = 1 \quad \forall i=1,\cdots,m \qquad (16)$$

$$\sum_{i=1}^{m} \sum_{t\in T_i(a)} w_i x_{it} \le b_a \; \forall a\in A, \qquad (17)$$

$$\sum_{i=1}^{m} \pi_i^k \sum_{t\in T_i(a)} x_{it} \le \pi_0^k \quad \forall k\in C(a),\; a\in A, \qquad (18)$$

$$x_{it} \ge 0, \quad \forall t\in T_i, \; i=1,\cdots,m \qquad (19)$$

Note that, in (18), the variables corresponding to the same session have the same coefficient (see Proposition 1). Hence we can easily determine the coefficients of the newly generated variables.

Let α_i, β_a, and τ_k be the dual variables associated to the constraints in (16), (17), and (18), respectively. If we fix a session i, the column generation problem for AHMPRL can be formulated as the following minimum cost Steiner tree problem:

$$\min \quad \sum_{a\in A(t)} \left(w_i c_a - w_i \beta_a^* - \sum_{k\in C(a)} \pi_i^k \tau_k^* \right) \qquad (20)$$

$$\text{s.t.} \qquad t\in T_i. \qquad (21)$$

If the resulting cost of the Steiner tree is less than α_i^*, the routing tree can be added to the current formulation. Otherwise, no column is generated with respect to session i.

6. Overview of the algorithm

6.1 Overview

In this section, we give a brief and overall explanation of our algorithm. To solve the HMRPL, an initial RHMPRL has to be provided. This initial RHMRPL must be feasible to ensure that proper dual information is passed to the column generation problem.

We can generate such a RHMRPL with a set of dummy columns, one per session, which have sufficiently large coefficients in the objective function. The dummy column corresponding to session i has coefficient 1 in the ith row of (2) and coefficient 0 in the rest.

After solving the current RHMRPL by simplex algorithm, we perform the column generation procedure. If new columns are generated, we add them to RHMPRL and repeat the same procedure. If no more new columns generated, i.e., the present solution is dual feasible, we proceed to find cutting planes which are violated by the optimal solution to the current RHMPRL and they will be added as cuts to it. So the formulation of AHMRPL is obtained.

If we get AHMRPL, we go through the same procedure as we do after the initial RHMRPL is obtained. We solve AHMRPL by simplex algorithm, and generate needed columns until there is no one generated. Also for AHMRPL with its present solution, we find minimal cover inequalities to add as cuts in AHMRPL.

When no more minimal cover inequalities can be found and hence no more columns can be generated, we get a lower bound of HMRP. If this solution is integral, we are done with an optimal solution to HMRP. Otherwise, we perform the branch-and-bound procedure to find an optimal integer solution to the final AHMRPL. This integer solution gives an upper bound, referred to as an incumbent solution. If the gap between the two bounds is 0, then the integer solution is also optimal to HMRP.

6.2 Column generation

As we've mentioned, the column generation problem for a multicast session is the hop-constrained Steiner tree problem, which is known to be NP-hard [5].

To solve the HMRPL to the optimality, the column generation algorithm should give an optimal solution. To our knowledge, there are two optimal guaranteeing algorithms for the hop-constraint Steiner tree problem, which are both based on the linear programming approach. Gouveia [5] formulated the hop-constrained Steiner tree problem as a directed multi-commodity flow model with flow variables. We gave a formulation using path variables and proposed a branch-and-pricing algorithm for it in [7].

Simplex algorithm incorporated in the branch-and-bound routine can easily solve the former formulation. However, the formulation has too many variables and constraints. Therefore, we consider the latter formulation in this paper.

Before introducing the formulation, we first define a feasible route r as a directed path from the root node (v_i^0) to a terminal node $v \in V_i$ using at most h_i arcs. Let $R(v)$ be the set of feasible paths to a terminal node $v \in V_i$, and $R(v,a) \subseteq R(v)$ be the set of paths in $R(v)$ containing $a \in A$. In the formulation, there are two types of variables: x_a and y_{rv}. The binary variables x_a, $a \in A$ indicate whether arc a is contained in the Steiner tree and the binary variables y_{rv}, $r \in R(v)$ and $v \in V_i$ indicate whether path r from v_i^0 to v is realized in the Steiner tree. The formulation is written as follows:

SP(i)

$$\min \quad \sum_{a \in A} \mu_a x_a \qquad (22)$$

$$\text{s.t.} \quad x_a - \sum_{r \in R(v,a)} y_{rv} \geq 0 \quad \forall a \in A, \; v \in V_i, \qquad (23)$$

$$\sum_{r \in R(v)} y_{rv} = 1 \quad \forall v \in V_i, \qquad (24)$$

$$x_a \in \{0,1\}, \quad \forall a \in A, \qquad (25)$$

$$y_{rv} \in \{0,1\}, \quad \forall r \in R(v), \; v \in V_i. \qquad (26)$$

Note that the coefficients $\{\mu_a\}$ in the objective function are all non-negative. Moreover, without loss of generality, we can assume all of them to be positive by letting the coefficient be $\mu_a + \varepsilon$ for each arc $a \in A$, where ε is a quite small positive number. Refer to [7] for the details of the algorithm.

6.3 Finding minimal Cover Inequalities

Here, we explain how a minimal cover is found. As noted earlier, the separation problem can be formulated as a knapsack problem, which does not always give a minimal cover. However, the following result shows an easily way to fix this problem.

Proposition 2. When the coefficients in the objective function are all positive, a cover corresponding to an optimal solution to SEP is always minimal.

proof)

We prove this by contradiction. Assume the resulting cover C is not minimal. Then there is $k \in C$ such that $\sum_{i \in C \setminus \{k\}} w_i > b_a$, and therefore $C \setminus \{k\}$ is also a cover. Moreover, $\sum_{i \in C \setminus \{k\}} c_i < \sum_{i \in C} c_i$, where $c_i = 1 - y_i^*$ because $c_i > 0$. This contradicts that C is corresponding to an optimal solution to SEP.

Therefore, we can always generate a minimal cover by letting the coefficients be $1 - y_i^* + \varepsilon$ where ε is a quite small positive number.

■

6.4 Lifting Procedure

To strengthen the generated minimal cover inequalities, we use the lifting procedure introduced in [9].

Every cover inequality generated from a minimal cover C can give rise to a lifted cover inequality of the form

$$\sum_{i \in N \setminus C} \delta_i y_i + \sum_{i \in C_2} \gamma_i y_i + \sum_{i \in C_1} y_i \leq |C_1| - 1 + \sum_{i \in C_2} \gamma_i \qquad (27)$$

where $C_1 \cap C_2 = \emptyset$, $C_1 \cup C_2 = C$ and $N = \{1, \cdots, m\}$

Moreover, $\{\delta_i\}$ and $\{\gamma_i\}$ can be chosen so that (27) defines a facet of the knapsack convex hull. The coefficients in (27) are obtained by sequential lifting. Unfortunately we know of no efficient way to consider all possible ordering of the elements of $N \setminus C$ that can be used in sequential lifting. From a practical point of view, we avoid this difficulty by choosing an ordering of the elements of $N \setminus C$ in a greedy fashion.

A Lifting Heuristic to obtain a lifted cover inequality of the form (27) with $C_2 = \emptyset$

Initialization: Given y^*, solve the knapsack problem to obtain a minimal cover C. If the optimal objective value is greater than $1 + \varepsilon_1$ where $\varepsilon_1 > 0$, no cutting planes are generated. Stop. Otherwise, let $L^1 = N \setminus C$ and let $k = 1$ Set $\delta_i = 1 \;\; \forall i \in C$. (Note even though the cover inequality corresponding to C is not violated, it may be able to be lifted to be violated.)

Iteration k: For all $i \in L^k$ find λ_i, which is the maximum value of π_i such that $\pi_i y_i + \sum_{j \in N \setminus L^k} \delta_j y_j \leq |C| - 1$ is valid. Let $i^* = \arg\max_{i \in L^k} \lambda_i y_i^*$ Set $L^{k+1} = L^k \setminus \{i^*\}$ and $\delta_{i^*} = \lambda_{i^*}$ If $L^{k+1} = \emptyset$ test whether $\sum_{i \in N} \delta_i x_i^* > |C| - 1$. If so, add the cut $\sum_{i \in N} \delta_i y_i \leq |C| - 1$ If $L_{k+1} \neq \emptyset$, $k \leftarrow k+1$ Return.

We have $\lambda_i = |C| - 1 - \zeta$ where

$$\zeta_i = \max\{ \sum_{j \in N \setminus L^k} \delta_j y_j ; \sum_{j \in N \setminus L^k} w_j y_j \le b_a - w_j,\ y \in B^{|N \setminus L^k|} \} \quad (28)$$

A simple extension of the lifting heuristic suggests how we can also search for extended cover inequalities of the form (27) with $C_2 \neq \emptyset$.

Separation Algorithm to obtain lifted cover inequality (27)

Step 1: Apply the lifting heuristic described above. If a violated inequality is found, stop.

Step 2: If not, choose $k = \arg\max_{i \in C} w_i y_i^*$ Set $C_2 = \{k\}$ and use the lifting heuristic to generate a facet-defining inequality for conv (S^k) from the cover $C \setminus \{k\}$, where $S^k = \left\{ y \in B^{n-1}: \sum_{i \in N \setminus \{k\}} w_i y_i \le b_a - w_k \right\}$

Step 3: The inequality in this step is of the following form

$$\sum_{i \in N \setminus C} \delta_i y_i + \sum_{i \in C_1} y_i \le |C_1| - 1$$

$$\text{where } C_1 = C \setminus \{k\} \quad (29)$$

Find the maximum value of γ_k such that the following inequality is valid:

$$\sum_{i \in N \setminus C} \delta_i y_i + \sum_{i \in C_1} y_i + \gamma_k y_k \le |C_1| - 1 + \gamma_k \quad (30)$$

We have $\gamma_k = \zeta_k - (|C_1| - 1)$ where

$\zeta_k = \max\{ \sum_{i \in N \setminus C} \delta_i y_i + \sum_{i \in C_1} y_i ; \sum_{i \in N \setminus \{k\}} w_i y_i \le b_a \}$

Step 4: Check the resulting inequality (30) for violation. Stop.

6.5 Column generation to AHMRPL

As noted earlier, column generation to HMRPL and AHMRPL can be done very efficiently. In the case of HMRPL, the coefficients of the newly generated column are decided according to the arcs that the corresponding routing tree uses. Suppose the newly generated column is for session i and its corresponding tree is t. The coefficient of the ith row of (2) is 1, and the coefficients of the rest of rows of (2) are 0. If routing tree t uses are a, then we set the coefficient of the ath row of (3) to be w_i; otherwise, 0.

In AHMPRL, the coefficients of the rows of (16) and (17) are decided in the same way as stated above. We will explain how to determine the coefficients of the rows of (18).

First, for each constraints in (18), identify from which constraints of (17) the row is derived. If kth row in (18) derived from the ath constraints in (17), and tree t corresponding to the new column uses the arc a, then the new column will be made to have the coefficient π_i^k in the row.

7 Future research topics

In this paper, we only proposed an algorithm based on the mathematical programming technique, which gives an tight lower-bound and an incumbent solution. To analize the performance of the algorithm, it is required to implement the algorithm. In addition, it is needed to develop a branch-and-cut procedure for it to generate an optimal solution.

Reference

[1] C. Barnhart, E.L. Johnson, G.L. Nemhauser, M.W.P. Savelsbergh, and P.H. Vance, ``Branch-and-price: Column generation for solving huge integer programs", *Operations Research* **46** (3) (1998) 316-329.

[2] Shiwen Chen, Oktay Gunluk, and Bulent Yener, ``The Multicast Packing Problem", *IEEE/ACM Transactions on networking* **86** (1999) 21-345.

[3] E.G. Coffman, M.R. Garey, and D.S. Johnson, ``Bin packing with divisible item sizes", *Journal of complexity* **3** (1987) 406-428.

[4] S. Even, A. Itai and A. Shamir, ``On the complexity of timetable and multicommodity flow problems", *SIAM Journal on Computing* **5** (4) (1976) 691-703.

[5] L. Gouveia, ``Using variable redefinition for computing lower bounds for minimal spanning and Steiner trees with hop constraints", *INFORMS Journal on Computing* **10** (2) (1998) 180-188.

[6] E.L. Johnson and M. W. Padberg, ``A note on the knapsack problem with special ordered sets", *Operations Research Letters*, **1** (1981) 18-22.

[7] J. Kang, D. Kang, and S. Park, "A new mathematical formulation for generating a multicast routing tree", *submitted to Electronic Letters* (2002).

[8] C. P. Low and N. Wang, ``An efficient

algorithm for group multicast routing with bandwidth reservation", *Computer Communications* 23 (2000) 1740-1746.

[9] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization* John Wilsey & Sons, New York, (1988).

[10] K. Park, S. Kang, and S. Park, ``An integer programming approach to the bandwidth packing problem", *Management Science* 42 (9) (1996) 1277-1291.

[11] H. D. Sherali and Y. Lee, ``Sequential and simultaneous liftings of minimal cover inequalities for generalized upper bound constrained knapsack polytopes", *SIAM Journal on Discrete Mathematics* 8 (1) (1995) 133-153.

[12] Stefan Voss, ``The Steiner tree problem with hop constraints", *Annals of Operations Research* 86 (1999) 321-345.