ELSEVIER

Computer Networks and ISDN Systems 29 (1997) 797–810

COMPUTER NETWORKS and ISDN SYSTEMS

Internetting connectionless data networks with a wide area public ATM network [1]

I-Fei Tsai, Rong-Hong Jan *

Department of Computer and Information Science, National Chiao Tung University, Hsinchu 30043, Taiwan, ROC

Accepted 15 January 1997

Abstract

Connectionless data services on ATM-based B-ISDN can be realized *directly* by means of the *connectionless service function* which is provided in *connectionless servers*. In this paper, we consider how to locate a certain amount of connectionless servers among the switching nodes in a public ATM network for the internetworking of connectionless data networks. The problem is formulated as a network optimization problem which is similar to the *p-median* problem. Two algorithms, one based on the *greedy method* and the other using *branch-and-bound strategy*, are presented to determine the locations of connectionless servers. By finding the optimal locations of connectionless servers, an optimal virtual overlay network which has minimized *total transport costs* for the connectionless data traffics can be constructed. © Elsevier Science B.V.

Keywords: Connectionless service function; Connectionless server; Virtual overlay network; Internetworking (LAN/MAN); ATM

1. Introduction

One of the most important applications in the ATM-based B-ISDN is to provide the interconnection between LANs/MANs. Connectionless (CL) service is the predominant mode of communications in LANs/MANs. However, ATM networks are inherently connection-oriented (CO). In order to support the CL data service on B-ISDN, two alternatives, *direct* and *indirect* approaches for the ATM networks have been identified by CCITT [1,2].

For *direct* approach, CL service is provided directly within the B-ISDN. Connectionless data traffics between the B-ISDN customers are transferred by way of the *connectionless servers* (*CLSs*) in which *connectionless service functions* (*CLSFs*) are provided. Connectionless protocol functions such as routing, addressing and quality of service selection are handled by CLSs. To serve as a cell-based router, the CLS routes cells to their destination or intermediate CLS according to the routing information included in the user data. Although logically a separate entity, the CLSF should be physically located in the same building as the ATM switch or may even be a physical part of the switch itself [3,4].

For *indirect* approach, CL data traffics between customers are transparently transferred above the ATM layer via the connection-oriented service. However, the use of fully-meshed semipermanent connections may be not feasible due to the limit of

* Corresponding author. Email: rhjan@cis.nctu.edu.tw.

[1] This work was partially supported by National Science Council, Republic of China under grant NSC85-2213-E-009-112.

PII S0169-7552(97)00002-0

the number of virtual path identifiers (VPIs) which are supported by a switch. The use of switched connections will introduce call setup delay and overheads for call control functions within the network. In addition, the indirect approach cannot fully utilized the benefits from multiple path routing of CL data traffic that is free from out of sequence inherently. Therefore, from the point of scalability and efficiency, *direct* CL services in a large scale network realized by means of the CLSF seems to be a more reasonable approach. There is growing consensus that providing connectionless service for the interconnection of LANs and MANs in a public ATM environment will more likely entail the use of CLSs [5].

A number of studies concerned about supporting CL data service on B-ISDN have been proposed. However, most of the works only concentrate on the general framework of the protocol architectures [6–15], bandwidth management and traffic control [16–19]. As for the CLS locating and connecting problem, Stavrov et al. [20] propose a *simulating annealing* method to solve it. In this paper we give two algorithms, greedy algorithm and branch-and-bound algorithm for solving the problem.

Based on the *direct* approach, we focus on the topological optimization for the internetting LANs/MANs with a wide area public ATM network. For the sake of congestion avoidance and load balancing, a large scale B-ISDN necessitates more than one CLS. In general, the fewer the CLS, the higher the load on each VP, and the higher the total transport cost in the virtual overlay network. On the other hand, it is not necessary to locate CLS at every switching node. The number of switching nodes with CLSF depends upon the size of network and the volume of CL traffic to be handled. The determination of where to place CLSs and how to interconnect CLSs has important performance consequences. As we will show later, to determine the optimal number of CLSs in an ATM network is as hard as to determine the minimum size of *vertex cover* in the corresponding graph, where the vertex cover problem is NP-complete [21].

After a determinant number of CLSs are located among the ATM switching nodes, an efficient (semi-)permanent VP layout should be preconfigured around the CLSs and the interworking unit (IWUs)

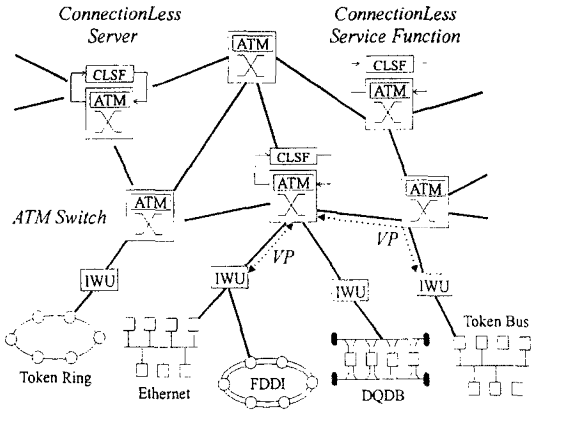


Fig. 1. Interconnection of LANs/MANs with a wide area public ATM network.

to which LANs/MANs are attached. In spite of virtually any embedded topology being possible, it is desirable that there must exist at least one CLS lying on the virtual path connection for each IWU pair. The IWUs between the LANs/MANs and the CLSs work as an interface of CL and CO environment. CL traffic between LANs/MANs will be transported by way of the VPs between the IWUs and intermediate CLSs (see Fig. 1).

In this paper, we focus on how to locate a certain amount CLSs, say p, on an ATM network. The problem of locating p CLSs can be formulated as an optimization problem. The objective is to minimize the *total transport costs* between all the IWU pairs. It is very similar to a well-known NP-complete problem, the *p-median* problem [22]. The *p-median* problem is to minimize the sum of distances from demand points to their respective nearest supply points. This kind of network location problem occurs when new facilities are to be located on a network. A formal description of *p-median* problem is given in Appendix A. In this paper, we will show that our CLS location problem is also NP-complete even when the network has a simple structure. Most of the problem solving techniques that are available for this kind of location problems rely on integer programming for a final solution. The emphasis of this work is first, to develop two efficient algorithmic approaches for the optimal location of CLSs in the ATM networks, and second, to determine the optimal virtual topology. The effect of statistical multiplexing on the optimization of virtual overlay network will be given in the last section.

The organization of this paper is as follows. In the next section, we will describe our problem formally. In Section 3, a heuristic algorithm based on the greedy method for finding the optimal locations of CLSs is presented. It can be shown that, by using this algorithm, we can get an optimal solution if p is not smaller than a threshold with respect to the number of ATM switches in the network model; else, the solution is near-optimal. Since the value of p (the number of CLSs) may be restricted in the hardware costs and routing complexity (e.g. network with the capabilities of broadcasting and multicasting may influence the design of virtual topology [23,24]), we still have to consider those cases (a smaller p) where the optimal solution can not be found by the greedy method. In Section 4, a branch-and-bound strategy is proposed to find the exact solution for this problem. In Section 5, according to the location of CLSs, we discuss the efficient VP layout for CL data traffic. Conclusions and discussions are given in the last section.

2. Problem formulation

Let $G=(V, E)=(V_g \cup V_s, E_{gs} \cup E_{ss})$ be an ATM network where V_g is the set of gateways, V_s is the set of ATM switches, E_{gs} is the set of links between gateway and switch, and E_{ss} is the set of links between two switches. As shown in Fig. 2, LANs/MANs $n_{i,j,k}$ are attached to gateways $g_{i,j}$, and gateways $g_{i,j}$ are connected to switches s_i. Here we assume that physically all the entities are *single-homed*. That is, for every LAN/MAN it only connected to one gateway, and, for every gateway it only connected to one switch. The CL traffics can be divided into three classes. Class 1 is the traffic from $n_{i,j,h}$ to $n_{i,j,k}$ ($\forall h \neq k$), class 2 is the traffic from $g_{i,j}$ to $g_{i,k}$ ($\forall j \neq k$), and class 3 is the traffic from $g_{i,j}$ to $g_{k,l}$ ($\forall i \neq k$). Note that the first class traffic (local traffic) is handled by gateways $g_{i,j}$. And, for the sake of locality, (semi-)permanent VPs should be preconfigured for the second class traffic between two gateways attached to the same switch. Here we concern only with the third class traffic between two gateways attached to different switches.

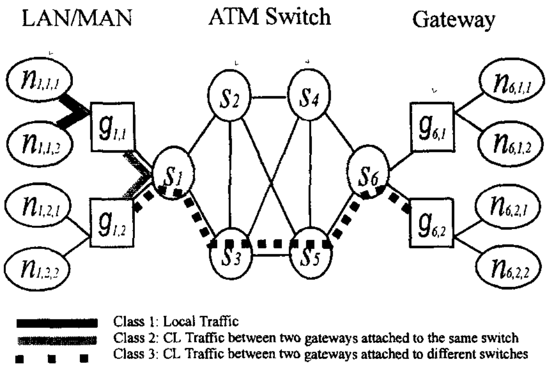


Fig. 2. A sample network model.

For the network G, let matrix $T=[w(g_{a,b}, g_{c,d})]_{|V_g| \times |V_g|}$ be the estimated long-term traffic pattern of G where $w(g_{a,b}, g_{c,d})$ is the traffic flow between two gateways $g_{a,b}$ and $g_{c,d}$, and let $A=[c(i, j)]_{|V| \times |V|}$ where $c(i, j)$ represents the cost per unit flow of the link (i, j). Link costs can be interpreted as metrics other than distances. They are often used to represent hop count, delay, lossage, penalties, or any other quantity that accumulates linearly along a path and that one wishes to minimize.

The problem considered in this paper is to optimize the virtual topology embedded in network G. The goal is to locate p CLSs among $|V_s|$ ATM switches such that the total CL data flow cost is minimized. Connectionless data flows between all the gateway pairs will be transported via the CLSs. That is, for each gateway pair, CL data flows from source gateway to destination gateway must pass through at least one intermediate CLS. However, it is generally favorable for them to be handled by exactly one CLS, due to the extra processing delay being introduced into each intermediate CLS. In such a case, the source sends the CL traffic to the intermediate CLS first, and then CLS forwards it to the destination.

Therefore, the objective function for the optimal locating of CLSs M is to minimize

$$\sum_{\substack{\forall g_{a,b}, g_{c,d} \in V_g \\ a \neq c}} \min_{\substack{m_i \in M \\ 1 \leqslant i \leqslant p}} w(g_{a,b}, g_{c,d}) \times \{d(g_{a,b}, m_i) + d(m_i, g_{c,d})\}, \quad (1)$$

where M is a subset of V_s, m_i is one of the p switching nodes in M. In Eq. (1), the value of $d(i, j)$ is the summation of all the links' cost along the shortest path from node i to node j. That is, distance $d(i, j)$ of path $\langle i = v_0, v_1, \ldots, v_k = j \rangle$ is the sum of the costs of its constituent links:

$$d(i, j) = \sum_{n=0}^{k-1} c(v_n, v_{n+1}).$$

The transport cost for gateway pair $(g_{a,b}, g_{c,d})$ is equal to the amount of traffic flow $w(g_{a,b}, g_{c,d})$ multiplied by the minimum distance of virtual path connection from $g_{a,b}$ to $g_{c,d}$ passing by one CLS. Since the minimum distance of virtual path connection from $g_{a,b}$ to $g_{c,d}$ via CLS m_i is

$$\min\{d(g_{a,b}, m_i) + d(m_i, g_{c,d}) \mid i = 1, 2, \ldots, p\},$$

the transport cost of gateway pair $(g_{a,b}, g_{c,d})$ with CLS set M is

$$\min_{\substack{m_i \in M \\ 1 \leq i \leq p}} w(g_{a,b}, g_{c,d})\{d(g_{a,b}, m_i) + d(m_i, g_{c,d})\}.$$

Our goal is to select a set M from V_s such that the total transport cost for all pairs of gateway in Eq. (1) is minimized, i.e.,

$$\min_{M \subseteq V_s} \sum_{\substack{\forall g_{a,b}, g_{c,d} \in V_g \\ a \neq c}} \min_{\substack{m_i \in M \\ 1 \leq i \leq p}} w(g_{a,b}, g_{c,d}) \times \{d(g_{a,b}, m_i) + d(m_i, g_{c,d})\}. \quad (2)$$

Notice that the CL traffic flow of a gateway pair may pass through more than one CLS. However, the only one CLS is selected to handle this flow.

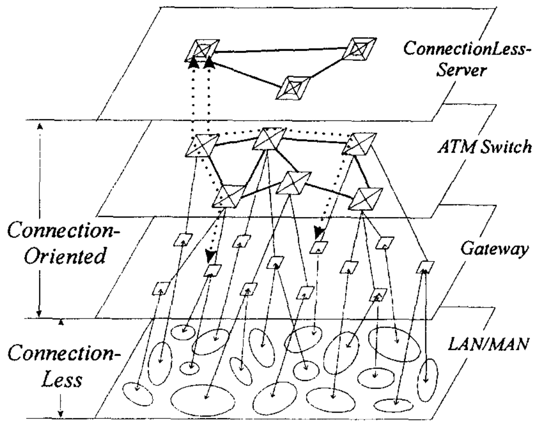


Fig. 3. A virtual overlayed CL network.

Once the optimal location of CLS sites have been determined, the optimal virtual path connections between all the gateway pairs can also be determined. Of course, the CL traffic routing is also decided. Fig. 3 shows a conceptual *virtual overlayed network* for the direct provision of CL data service in the ATM networks. In this figure VPs are set between the gateways and CLSs, and the messages from the source gateway are routed to the destination gateway by CLS.

Note that for each gateway $g_{i,j}$, the switching node s_i is the only incoming and outgoing port, thus the minimized total transport cost in Eq. (2), say OPT, is composed of the costs on E_{gs} and E_{ss}, i.e.,

$$\begin{aligned} OPT &= OPT_{gs} + OPT_{ss} \\ &= \sum_{a=1}^{|V_s|} \sum_{b=1}^{n(s_a)} \sum_{\substack{c=1 \\ c \neq a}}^{|V_s|} \sum_{d=1}^{n(s_c)} w(g_{a,b}, g_{c,d}) \\ &\quad \times \{c(g_{a,b}, s_a) + c(s_c, g_{c,d})\} \\ &\quad + \min_{M \subseteq V_s} \sum_{\substack{\forall s_a, s_c \in V_s \\ a \neq c}} \min_{\substack{m_i \in M \\ 1 \leq i \leq p}} w(s_a, s_c) \\ &\quad \times \{d(s_a, m_i) + d(m_i\ s_c)\}, \end{aligned} \quad (3)$$

where $n(s_i)$ is the number of gateways attached to switch s_i. Since OPT_{gs} is a constant, the location problem is equivalent to

$$OPT_{ss} = \min_{M \subseteq V_s} \sum_{\substack{\forall s_a, s_c \in V_s \\ a \neq c}} \min_{\substack{m_i \in M \\ 1 \leq i \leq p}} w(s_a, s_c) \times \{d(s_a, m_i) + d(m_i, s_c)\}, \quad (4)$$

where the aggregated traffic flow between s_a and s_c is

$$w(s_a, s_c) = \sum_{i=1}^{n(s_a)} \sum_{j=1}^{n(s_c)} w(g_{a,i}, g_{c,j}).$$

By inspection, the lower bound of Eq. (3), say LB, can be reached only if for each gateway pair there always exists at least one CLS on one of its shortest paths. Thus,

$$\begin{aligned} LB &= \sum_{\substack{\forall g_{a,b}, g_{c,d} \in V_g \\ a \neq c}} w(g_{a,b}, g_{c,d}) d(g_{a,b}, g_{c,d}) \\ &= LB_{gs} + LB_{ss} \\ &= OPT_{gs} + \sum_{\substack{\forall s_a, s_c \in V_s \\ a \neq c}} w(s_a, s_c) d(s_a, s_c), \qquad (5) \end{aligned}$$

where

$$LB_{ss} = \sum_{\substack{\forall s_a, s_c \in V_s \\ a \neq c}} w(s_a, s_c) d(s_a, s_c)$$

is the lower bound of Eq. (4).

In this section, we have formulated our problem as a topological optimization problem. Like the p-median problem, this optimization problem is also NP-complete. A proof by restriction is given in Appendix B. From Appendix B, the well-known NP-complete problem, *vertex covering*, is a special case of the p-CLS problem.

3. Problem solving using greedy method

To be direct, we may compute the total transport cost for each subset M of V_s, where $|M| = p$, and then compare these values to find an optimal solution. However, the brute-force method takes $O(\binom{|V_s|}{p} \cdot p|V_g|^2)$ time. In this section, a greedy algorithm with lower computational complexity for the location of p CLSs will be presented. The complete algorithm is described in Section 3.1; and the numerical results are given in Section 3.2.

3.1. The proposed algorithm

At first, the *Floyd–Warshall algorithm* [25] is applied to find the all pairs shortest paths for all gateway pairs. In the mean time, a *predecessor*(i, j) index for each node pair is maintained. By using the predecessor index, we can easily obtain the shortest path for each gateway pair by backtracking from the destination node to the source node.

Let L be the set of CLSs to be located (initially empty), and P be the set of all pairs of gateways initially, i.e., $P = \{(g_{a,b}, g_{c,d}) | \forall g_{a,b}, g_{c,d} \in V_g, a \neq c\}$, and $v(s_i)$ be the accumulated weight on switching node s_i for all gateway pairs' shortest paths passing through s_i. The value of $v(s_i)$ can be obtained by traversing all gateway pairs $(g_{a,b}, g_{c,d})$ shortest paths and incrementing it by $w(g_{a,b}, g_{c,d})$ whenever s_i is visited. After all the paths have been traversed once, we pick up one node with the maximum value of $v(s_i)$ as a CLS, say, s^* (if more than one node is feasible then select arbitrary one). Set L to be $L \cup \{s^*\}$. And then remove those gateway pairs passed through s^* from P. Repeat the above traversal procedure for all the gateway pairs in P until P becomes an empty set.

It is not hard to verify that the total transport cost for node set L is equal to the lower bound LB in Eq. (5). Thus, if p is equal to $|L|$ then L is exactly the optimal solution for the objective function. If $p > |L|$ then any node set containing L with the cardinality p will be an optimal solution since the lower bound can be reached if only we locate $|L|$ nodes. In fact, it is not necessary to locate so many CLSs in the network. In addition to the minimal total transport cost can be fulfilled if only $|L|$ CLSs are located, it may introduce more complexity while multicasting/broadcasting [23]. It is important to note that if $p \geqslant |L|$ then the optimal locations of CLS will not change even though the traffic load $(T = [w(g_{s,t}, g_{u,v})]_{|V_g| \times |V_g|})$ is varying with time.

Next, consider that if p is less than $|L|$. If M (the optimal node set of p-CLS) is contained in L, we can find these p CLS among L but not V_s. This will reduce the computational complexity since $|L|$ is less than $|V_s|$. Unfortunately, M is not always contained in L (the set of p-CLS may not contain the set of $(p-1)$-CLS, see Table 1). Although the above algorithm based on the greedy method can not

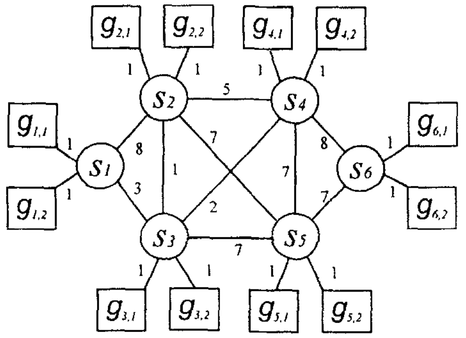


Fig. 4. A sample network.

Table 1
The experimental result of 6 nodes network model

p	greedy method		brute force		relative error rate
	CLS locations	TTC_L	CLS locations	OPT_{ss}	$(TTC_L - OPT_{ss})/OPT_{ss}$
1	s_3	1160	s_3	1160	0.0000
2	s_3, s_5	1056	s_3, s_6	1048	0.0076
3	s_3, s_5, s_4	1024	s_3, s_4, s_5	1024	0.0000
4	s_3, s_5, s_4, s_1	1024	s_1, s_3, s_4, s_5	1024	0.0000
5	s_3, s_5, s_4, s_1, s_2	1024	s_1, s_2, s_3, s_4, s_5	1024	0.0000
6	s_3, s_5, s_4, s_1, s_2, s_6	1024	s_1, s_2, s_3, s_4, s_5, s_6	1024	0.0000

always find the optimal solution in case of $p < |L|$, the first p nodes taken by it will form a near-optimal solution. This is because the first p nodes have the largest accumulated weight, and the near-optimality is a direct result of transporting the majority of traffic volume in a least cost manner (by way of shortest path).

This greedy method is a shortest path first (SPF) based algorithm. Since the shortest path from $g_{i,j}$ to $g_{k,l}$ starts at $g_{i,j}$ and goes to s_i, through the shortest path from s_i to s_k, and then to $g_{k,l}$, i.e., $d(g_{i,j}, g_{k,l}) = c(g_{i,j}, s_i) + d(s_i, s_k) + c(s_k, g_{k,l})$. When we look for the shortest paths for all gateway pairs $(g_{i,j}, g_{k,l})$ from the graph $G = (V_g \cup V_s, E_{gs} \cup E_{ss})$, we only have to find them for all switch pairs (s_i, s_k) from its subgraph $G' = (V_s, E_{ss})$. This will save considerable time in computation. A brief pseudocode is given in the following procedure. It is easy to verify that the computational complexity of this greedy method is $O(p \cdot |V_s|^3)$.

Procedure GREEDY(G', p)
  Floyd–Warshall(G')
  for all $s_i \in V_s$ **do** $v(s_i) \leftarrow 0$
  $L \leftarrow \emptyset$
  $P \leftarrow \{(s_a, s_c) | \forall s_a, s_c \in V_s, a \neq c\}$
  while ($P \neq \emptyset$ and $|L| < p$) **do**
    for each $(s_a, s_c) \in P$ **do**
      traverse its shortest path, increment $v(s_i)$ by $w(s_a, s_c)$ whenever s_i is visited
    $v(s^*) = \max\{v(s_i) | \forall s_i \in V_s - L\}$
    $L \leftarrow L \cup \{s^*\}$
    $P \leftarrow P - \{(s_a, s_c) | \text{shortest path from } s_a \text{ to } s_c \text{ passed by } s^*\}$
  return(L)

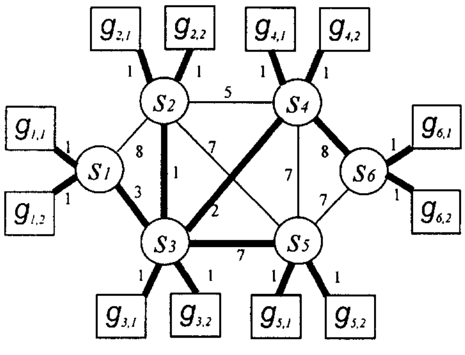


Fig. 5. A virtual shortest path tree rooted at $g_{1,x}$.

3.2. Numerical results

A numerical example of our network model is shown in Fig. 4. Suppose that $w(g_{a,b}, g_{c,d}) = 1$ for all gateway pairs. The results of locating p CL

Table 2
The optimal virual paths for all the gateway pairs

gateway pair	optimal virutal path
$(g_{1,x}, g_{2,y})$	$g_{1,x} \leftrightarrow s_1 \leftrightarrow s_3 \leftrightarrow s_2 \leftrightarrow g_{2,y}$
$(g_{1,x}, g_{3,y})$	$g_{1,x} \leftrightarrow s_1 \leftrightarrow s_3 \leftrightarrow g_{3,y}$
$(g_{1,x}, g_{4,y})$	$g_{1,x} \leftrightarrow s_1 \leftrightarrow s_3 \leftrightarrow s_4 \leftrightarrow g_{4,y}$
$(g_{1,x}, g_{5,y})$	$g_{1,x} \leftrightarrow s_1 \leftrightarrow s_3 \leftrightarrow s_5 \leftrightarrow g_{5,y}$
$(g_{1,x}, g_{6,y})$	$g_{1,x} \leftrightarrow s_1 \leftrightarrow s_3 \leftrightarrow s_4 \leftrightarrow s_6 \leftrightarrow g_{6,y}$
$(g_{2,x}, g_{3,y})$	$g_{2,x} \leftrightarrow s_2 \leftrightarrow s_3 \leftrightarrow g_{3,y}$
$(g_{2,x}, g_{4,y})$	$g_{2,x} \leftrightarrow s_2 \leftrightarrow s_3 \leftrightarrow s_4 \leftrightarrow g_{4,y}$
$(g_{2,x}, g_{5,y})$	$g_{2,x} \leftrightarrow s_2 \leftrightarrow s_5 \leftrightarrow g_{5,y}$
$(g_{2,x}, g_{6,y})$	$g_{2,x} \leftrightarrow s_2 \leftrightarrow s_3 \leftrightarrow s_4 \leftrightarrow s_6 \leftrightarrow g_{6,y}$
$(g_{3,x}, g_{4,y})$	$g_{3,x} \leftrightarrow s_3 \leftrightarrow s_4 \rightarrow g_{4,y}$
$(g_{3,x}, g_{5,y})$	$g_{3,x} \leftrightarrow s_3 \leftrightarrow s_5 \leftrightarrow g_{5,y}$
$(g_{3,x}, g_{6,y})$	$g_{3,x} \leftrightarrow s_3 \leftrightarrow s_4 \leftrightarrow s_6 \leftrightarrow g_{6,y}$
$(g_{4,x}, g_{5,y})$	$g_{4,x} \leftrightarrow s_4 \leftrightarrow s_5 \leftrightarrow g_{5,y}$
$(g_{4,x}, g_{6,y})$	$g_{4,x} \leftrightarrow s_4 \leftrightarrow s_6 \leftrightarrow g_{6,y}$
$(g_{5,x}, g_{6,y})$	$g_{5,x} \leftrightarrow s_5 \leftrightarrow s_6 \leftrightarrow g_{6,y}$

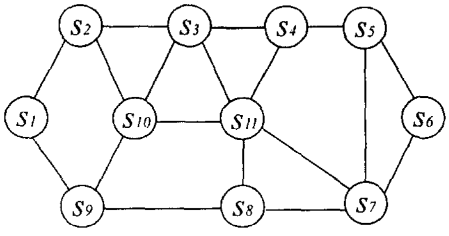


Fig. 6. Simulation model.

servers ($p = 1, 2, \ldots, 6$) by the greedy method and brute force are shown in Table 1.

In this example, $|L|$ is equal to 3. If $p \geqslant 3$, the CLSs obtained by greedy method will always be an optimal solution. If $p < 3$ then the solution found by the greedy method is near optimal; and sometimes it may be the optimal one by coincidence, e.g., $p = 1$.

A *virtual shortest path tree* rooted at $g_{1,x}$ for routing CL data traffic is shown in Fig. 5 if the located CLS set is $\{s_3, s_4, s_5\}$. The complete optimal virtual path connections for all the gateway pairs is given in Table 2. Notice that there will always exist at least one CLS on each virtual path connection.

It is noted that if we use the *indirect* approach then it needs $|V_g|(|V_g|-1)$, say $12(12-1) = 132$, VPs. If the *direct* approach is adopted then the minimum requirement on the number of VPs will fall in the range of $[2(|V_g|+p-1), 2p|V_g|]$, where $|V_g|+p-1$ is the edge number of a hierarchical virtual topology in which CLSs form a tree and every gateway connected to exactly one CLS, and $p|V_g|$ stands for every gateway has a virtual connection to each CLS (a bipartite graph in which there is no connection between CLSs). In this example the minimum number of VPs required is only 38 which is within the range from 28 to 72 ($V_g = 12$, $p = 3$).

In fact, in addition to the selection criterion described in the previous section, we have also compared it with two alternatives based on the same approach. For the first alternative, in each iteration of selection, we pick up the switching node s_i with a minimum value of the sum of $w(s_a, s_c)\{d(s_a, s_i) + d(s_i, s_c)\}$ for all the node pairs, i.e.,

$$\min_{s_i \in V_s} \sum_{\substack{\forall s_a, s_c \in V_s \\ a \neq c}} w(s_a, s_c)\{d(s_a, s_i) + d(s_i, s_c)\}.$$

The second alternative (random method) simply selects an arbitrary one switching node in each iteration to be a CLS. Notice that the first alternative will always get the optimal solution for the location of a single CLS.

Consider a simulation model with 11 switching nodes shown in Fig. 6. Applying the original greedy method will excel the others both at the size of covering set $|\bar{L}|$ (i.e., the minimum number of CLSs reaching the lower bound) and the relative error rate $((TTC_L - OPT)/OPT)$. The numerical result is given in Table 3 and Fig. 7. It is a great surprise that the results obtained by greedy method is very good. From Fig. 7, we note that the total transport cost got by this algorithm is very close to that of the optimal solution if $p \leqslant 6$ on average, else it can get an optimal solution.

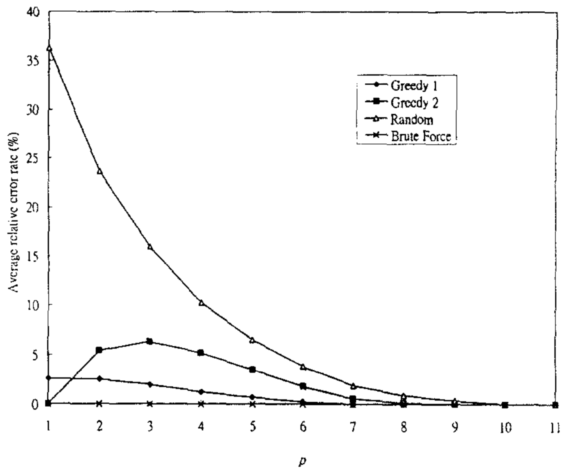


Fig. 7. Average relative error rate.

Table 3
Average size of covering set

Method	$\lvert\bar{L}\rvert$	$\lvert\bar{L}\rvert/\lvert V_s\rvert$
brute force	5.777	0.525
greedy 1	6.391	0.581
greedy 2	7.479	0.679
random selection	8.925	0.811

4. Problem solving using branch-and-bound strategy

In the previous section, the heuristic algorithm based on the greedy method fails to find the optimal solution for all cases. In this section, we propose another approach based on the branch-and-bound method to find the optimal solution.

In Section 4.1, we will describe the algorithm in detail. In Section 4.2, numerical results are given to show the effectiveness of the branch-and-bound method.

4.1. The proposed algorithm

A *current optimal solution* I^* is initialized to be the near-optimal(/optimal) solution L which is obtained by the greedy method (for those solutions of $p < |L|$, we can not tell whether they are near-optimal or exactly optimal). And a temporary *upper bound UB* is set to be the value of total transport cost for the current optimal solution, i.e., TTC_L. A possible tree organization for the *state space* of network model in Fig. 4 is shown in Fig. 8, in which a left branch represents the inclusion of a particular switching node (located as a CLS) while the right branch represents the exclusion of that node (not located as a CLS).

For each instance I in the state space tree, we maintain its lower bound $lb(I)$ and an array R_I for all the switching nodes to record their status, where

$$R_I[s_i] = \begin{cases} 1, & \text{if node } s_i \text{ is included,} \\ -1, & \text{if node } s_i \text{ is excluded,} \\ 0, & \text{otherwise (node } s_i \text{ is in the} \\ & \text{don't care condition).} \end{cases}$$

Let I_0 be the initial instance in which all the nodes are in a don't care condition (i.e., $R_I[s_i] = 0$, $\forall s_i$), and its lower bound $lb(I_0)$ is set to be LB_{ss}. In the first branch step we split I_0 into two instances I_1 and I_2. In the left child instance I_1, the status of a certain switching node, e.g., node 3 (the first node obtained by the greedy method) is set to be included (i.e., $R_I[s_3] = 1$). On the other hand, in the right child instance I_2, node 3 is set to be excluded (i.e., $R_I[s_3] = -1$). The other nodes still remain in the don't care condition. The following child instances will be further individually split into two instances in the similar way, e.g., include 5 and exclude 5 (node 5 is the second node obtained by the greedy method). The status of other nodes inherit its parent instance.

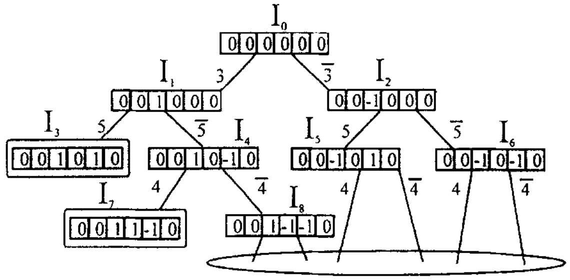


Fig. 8. State space tree for $p = 2$.

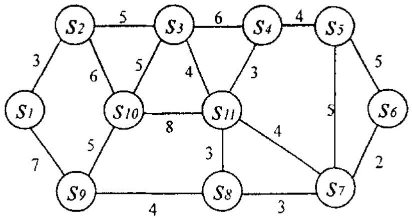


Fig. 9. A sample network.

The candidate instance which is selected from priority queue [2] Q to be splitted is the live node with a minimum lower bound. Initially, I_0 is the only one element in Q. Both of its children, I_1 and I_2, are non-leaf, and thus will be insert back into Q for further consideration. If a leaf instance I is visited, i.e., the number of included nodes is equal to p, compute its value of total transport cost TTC_I. If the value is less than the upper bound UB, the current optimal solution and upper bound are changed to the leaf instance and TTC_I. In addition, nodes in priority queue Q with larger or equal lower bound than TTC_I are removed.

The lower bound for each non-leaf instance can be computed as follows. The lower bound of left child instance will always inherit from its parent instance, but that of the right child does not. For all the node pairs, if there always exists at least one un-excluded intermediate switching node on its shortest paths, the lower bound for this instance is just the same as its parent instance. Otherwise, the

[2] A priority queue is a data structure for maintaining a set of elements, each with an associated priority value. At any time we can insert an element with arbitrary priority. Without following the first-in-first-out (FIFO) queueing discipline, a priority queue deletes the element with the highest (or the lowest) priority. In our case, we delete an instance I from Q with minimum lower bound.

lower bound for the instance should be updated. That is, if all the intermediate switching nodes along a gateway pair's shortest path are excluded, we have to find another shortest path (in distance) with an un-excluded node lying on it instead of the original one to calculate the lower bound. For example, in Fig. 9, if $R_I[s_i]_{i=1,2,\ldots,11} = (-1, -1, 1, -1, -1, -1, 0, -1, 0, -1, -1)$, then desirable virtual connection from s_1 to s_{10} can not be $\langle s_1, s_2, s_{10} \rangle$ but $\langle s_1, s_9, s_{10} \rangle$. The distance of virtual connection from s_i to s_j can be found by taking a minimum value of $d(s_i, s_m) + d(s_m, s_j)$ where s_m is one of the elements in the set of un-excluded nodes, i.e., $\min\{d(s_i, s_m) + d(s_m, s_j) \mid R_I[s_m] \neq -1, \forall m\}$. Notice that the shortest virtual connection from s_1 to s_2 is $\langle s_1, s_2, s_3, s_2 \rangle$, which is not a *simple path.* [3] The new lower bound for this kind of instances can be computed by LB_{ss} adding the difference of transport cost between the virtual path connections and the original shortest paths for all the gateway pairs.

In the branch-and-bound search scheme, we always keep track of all partial solutions (live nodes) contending for further consideration. The node with least cost selecting from Q is extended one level at each iteration, splitting into two child nodes. Next, these new nodes are considered, along with the remaining old ones: again, the least-cost is extended. When a complete solution (leaf node) is reached, compare its total transport cost with that of current optimal solution, and decide whether the current optimal solution will be replaced by it. This process repeats to extend all partial solutions until their lower bounds are not lower than the total transport cost of current optimal solution. That is, once we make sure that all the live nodes are not better than the current optimal solution, then we can claim that the current optimal solution is exactly an optimal solution. A brief pseudocode for the algorithm is shown in the following procedure.

Procedure BRANCH-AND-BOUND(G', L, p)
  $I^* \leftarrow L$
  $UB \leftarrow TTC_L$
  create initial instance I_0
  for all $s_i \in V_s$ **do** $R_{I_0}[s_i] \leftarrow 0$
  $lb(I_0) \leftarrow LB_{ss}$
  $Q \leftarrow \{I_0\}$
  while ($Q \neq \emptyset$) **do**
    $lb(I') = \min\{lb(I) \mid \forall I \in Q\}$
    $Q \leftarrow Q - \{I'\}$
    create right child instance I_r of I'
    R_{I_r}[sequence[level(I') + 1]] $\leftarrow -1$
    $lb(I_r) \leftarrow$ LOWERBOUND(I_r)
    if $lb(I_r) < UB$ then $Q \leftarrow Q \cup \{I_r\}$
    create left child instance I_l of I'
    R_{I_l}[sequence[level(I') + 1]] $\leftarrow 1$
    if cardinality(I_l) = p **then**
      if $TTC_{I_l} < UB$ **then**
        $I^* \leftarrow I_l$
        $UB \leftarrow TTC_{I_l}$
        $Q \leftarrow Q - \{I \mid lb(I) \geqslant UB, \forall I \in Q\}$
    else
      $lb(I_l) \leftarrow lb(I')$
      $Q \leftarrow Q \cup \{I_l\}$
  return(I^*)

The computational complexity for brute-force method is $O(\binom{|V_s|}{p} \cdot p \mid V_s \mid^2)$. In this branch-and-bound strategy, the time to calculate the value of TTC for a left child instance is $O(p \mid V_s \mid^2)$ if the instance is a leaf node. Otherwise, it only takes O(1) time to obtain the lower bound. A right child instance is always a non-leaf node. Its complexity is $O(\bar{h} \mid V_s \mid^2)$, where $\bar{h}$ is the mean internodal distance. For a connected semi-random network with 100 nodes and degree 4, $\bar{h} \approx 3$ [26]. Let n_{L_1}, n_{L_2}, and n_R denote the number of left leaf, left non-leaf, and right child generated by this algorithm, respectively. Thus, the complexity for the branch-and-bound method can be written as $O(n_{L_1} \cdot p \mid V_s \mid^2 + n_{L_2} + n_R \cdot \bar{h} \mid V_s \mid^2)$ where the values of n_{L_1}, n_{L_2}, n_R depend on network instance.

Procedure LOWERBOUND(I)
  $lb(I) \leftarrow LB_{ss}$
  for each switching node pair (s_a, s_c) **do**
    $flag \leftarrow 0$
    $s_i \leftarrow s_c$
    while ($flag = 0$ **and** $s_i \neq s_a$) **do**
      if $s_i \in V_s - V_{ex}$ **then**
        $flag \leftarrow 1$
        break
      $s_i \leftarrow$ predecessor(s_a, s_i)

[3] A path is *simple* if all vertices in the path are distinct.

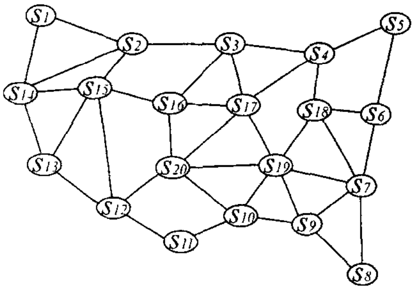


Fig. 10. Simulation model.

```
    if flag = 0 then
        d'(s_a, s_c) ← min{d(s_a, s_j) +
            d(s_j, s_c) | ∀s_j ∈ V_s − V_ex}
        lb(I) ← lb(I) + d'(s_a, s_c) − d(s_a, s_c)
return(lb(I))
```

4.2. Numerical results for branch-and-bound strategy

A simulation model is shown in Fig. 10. Comparing to the brute force, Table 4 shows the excellence of the branch-and-bound strategy by a simulation study. The number of instance traversals will not continuously increase proportional to p but tend to decrease for large p's. The proportion of branch-and-bound strategy to brute force method in the total number of instance traversals is given in Table 5. Notice that the order of feasible solutions in the branch step using the order of greedy sequence, i.e., obtained by the greedy method, will not have a smaller search tree. In other words, if we use the greedy sequence as the branching sequence, it could not get the optimal solution sooner by pruning the search tree properly.

5. Virtual topology construction

In Section 2, we have mentioned that once the locations of CLSs have been determined, the virtual path connections between all the gateway pairs are also determined. Since the CL traffic flow for a gateway pair may pass by more than one CLS, we have to select one CLS to handle this CL traffic. In the next step, we determine the locations of end-points for each VP along the virtual path connection. And then allocate proper bandwidth to them according to the traffic pattern matrix $T=[w(g_{a,b}, g_{c,d})]$.

If the CL traffic flow for gateway pair $(g_{a,b}, g_{c,d})$ passed by exactly one CLS, say s_m, then there is only one choice: set up the virtual paths from $g_{a,b}$ to s_m and from s_m to $g_{c,d}$. For example, if the 3 CLSs for Fig. 9 is $\{s_3, s_7, s_9\}$, we will set up two VPs for $(g_{2,1}, g_{4,1})$ from $g_{2,1}$ to CLS s_3 and from CLS s_3 to $g_{4,1}$ (i.e., the intermediate endpoint is on s_3). The allocated bandwidth for them can be preconfigured to $w(g_{2,1}, g_{4,1})$.

If the flow passed by more than one intermediate CLS then it has more than one intermediate endpoint for choice. For example, the CL traffic flow from $g_{2,1}$ to $g_{6,1}$ will go through the virtual path connection $\langle g_{2,1}, s_2, s_3, s_{11}, s_7, s_6, g_{6,1}\rangle$ with two CLSs lying on it, i.e., s_3 and s_7. In this case, for the sake of load balancing, we prefer to select the one with

Table 4
Number of instances traversed. Simulation 100 times, $|V_s|=20$

P	branch-and-bound						brute-force
	greedy sequence			random sequence			$100\binom{\lvert V_s\rvert}{p}$
	n_{L_1}	n_{L_2}	n_R	n_{L_1}	n_{L_2}	n_R	
2	14,382	1,650	16,032	13,545	1,632	15,177	19,000
3	62,520	12,840	75,360	60,627	12,603	73,230	114,000
4	169,386	53,766	223,152	168,105	52,365	220,470	484,500
5	323,460	149,742	473,202	322,710	141,789	464,499	1,550,400
6	491,121	313,104	804,25	500,754	292,473	793,227	3,876,000
7	556,770	471,375	1,028,145	585,102	399,234	984,336	7,752,000
8	444,576	512,520	957,096	502,590	497,361	999,951	12,597,000
9	289,725	417,567	707,292	312,117	406,440	718,557	16,796,000
10	138,162	257,433	395,595	178,707	291,822	470,529	18,475,600

Table 5
Proportion in total number of instance traversals

p	greedy sequence	random sequence
2	1.6876	1.5976
3	1.3221	1.2847
4	0.9212	0.9101
5	0.6104	0.5992
6	0.4150	0.4093
7	0.2653	0.2540
8	0.1520	0.1588
9	0.0842	0.0856
10	0.0428	0.0510

less CL data traffic handled by it. If we take s_3 as the intermediate endpoint, then s_7 is transparent to this traffic flow (CL data packets from $g_{2,1}$ to $g_{6,1}$ will pass by s_7 as an ordinary switch).

By backtracking the virtual path connection for each gateway pair $(g_{a,b}, g_{c,d})$, the CLSs $M[1], M[2], \dots, M[n]$ consecutively being passed by the VPC of $(g_{a,b}, g_{c,d})$ can be obtained. Let $\ll i, j \gg$ denote the VP for shortest path from i to j. The following pseudocode illustrates how to layout a virtual overlay network.

Procedure LAYOUT(G, p)
  for each gateway pair $(g_{a,b}, g_{c,d})$ **do**
    $\langle M[1], M[2], \dots, M[n] \rangle \leftarrow$ backtracking$(g_{a,b}, g_{c,d})$
  for $i = 1$ **to** p **do** $v(M[i]) \leftarrow 0$
  sort all pairs of gateway by the combination of n (i.e., the number of CLSs they pass by) in ascending order, and weight in descending order into an ordered list L
  for each gateway pair (x, y) in L with $n = i$ **do**
    $v(t) = \min\{v(M[j]) \mid j = 1, \dots, i\}$
    setup $\ll x, t \gg$ and $\ll t, y \gg$
    $v(t) \leftarrow v(t) + w(x, y)$

6. Conclusions and discussions

Direct CL service is particularly suitable for delay-tolerant or non-real-time applications, i.e., those which do not require tightly constrained delay and delay variation, such as traditional computer communications applications. From the view point of efficiency, *direct* provision of CL data service is a better choice in a large-scale ATM network.

In this paper, we have formulated the p-CLS problem as a mathematical optimization and shown that it is NP-complete. The objective function of the p-CLS problem is to minimize total transport cost. Minimized total transport cost implies minimized average transport cost, thus the minimum average end-to-end delay or bandwidth requirement can be achieved. A branch-and-bound algorithm for optimal location of CLSs has been developed and illustrated. Further reduction in computational complexity is also available by using the greedy algorithm. By the optimal locating CLSs, we can obtain the optimal virtual topology for the interconnection of LANs/MANs with a wide area public ATM network. It is easily seen that the proposed solution can be applied to ATM LANs by replacing gateways with ATM stations. It is also useful in designing the *broadband virtual private networks* [27].

We have considered taking one CLS as the intermediate endpoint for every gateway pair in this paper. Comparing with taking more than one intermediate endpoint, it has the advantage of less delay but is defective in *scalability*. That is, if we take only one intermediate endpoint for all pairs of gateway, the required minimum number of VPIs will be greater than that of taking more than one, since all the VPs with the same endpoints can be statistically multiplexed to one large VP to utilize the physical trunk. However, in spit of the fact that taking more than one intermediate endpoints will have the benefit in bandwidth and VPI efficiency (due to the effect of statistical multiplexing), it suffers from much more delay. This is because of the more intermediate endpoints, the more the delay for data processing, routing (and segmentation, reassembly for the reassembly mode server [12]) at each CLS.

In general, a smaller p will result in a higher multiplexing gain if taking more than one CLS is not prohibited. However, the smaller p implies a larger total transport cost. A good compromise can be found, in addition to introducing CLS cost under a certain budget consideration, by unifying the cost metric for both the transport cost and multiplexing gain. (The evaluation of statistical multiplexing gain for inter-LAN/MAN CL data traffic is beyond the scope of this paper.)

One may argue that, first, why the objective function does not subject to the constraint on link capacity; second, while the connectionless flow is additive, the bandwidth required to carry flow does not grow linearly with flow. However, connectionless data traffics tend to be bursty by nature, thus it is difficult to predict the accurate values of traffic parameters (e.g. peak rate and burstiness) on the vast majority of existing data networking applications. To optimize the link utilization, several control methods (e.g. rate-based and credit-based mechanisms) for two new *QoS* service classes in ATM network, say *ABR* (*Available Bit Rate*) and *UBR* (*Unspecified Bit Rate*), are proposed to carry bursty and unpredictable traffics in a best effort manner [28]. No resource reservation is performed before transmission. They are supposed to make an efficient use of bandwidth, by dynamically allocating the available bandwidth on an *as need* basis to all the active connections, with extremely low cell loss. It is remarkable that best effort services do not reserve guaranteed non-preemptable bandwidth. Thus, the effective gain of statistical multiplexing is almost intractable. It is our belief that it makes sense to neglect the effect of statistical multiplexing on VP connecting CLSs at the time we try to find the optimal location of CLSs in our formulation.

Appendix A. The *p*-median problem

Let $G=(V, E)$ be a connected undirected graph where V and E represent the sets of vertices and edges of graph G respectively. Each vertex v in V is associated with a nonnegative number $w(v)$ (the weight of v), and a positive number $l(e)$ (the length of e) is binding to each edge e in E. The *p-median* problem is to find a set M of p vertices in V so as to

$$\min_{M\subseteq V} \sum_{\forall v\in V} \min_{\substack{m_i\in M\\ 1\leqslant i\leqslant p}} w(v)d(v, m_i),$$

where d is a distance function on G.

It has been shown [20] that the problem of finding *p-median* of a network is an NP-hard problem even in the case when the network is a planar graph of maximum vertex degree 3 all of whose edges are of length 1 and all of whose vertices have weight 1.

Appendix B. NP-completeness of the *p*-CLS problem

In this appendix, the problem formulated in Section 2 is shown to be NP-complete by restricting its instance to the *vertex cover* problem. The optimization problem in Eq. (4) can be converted to the following decision problem, the p-CLS problem:

Instance: Graph $G'(V_s, E_{ss})$, for each node pair (s_i, s_j) of G' weight $w(s_i, s_j)\in Z^+$, for each edge $\{s_i, s_j\}\in E_{ss}$, length $c(s_i, s_j)\in Z^+$, positive integer $p\leqslant |V_s|$, positive integer B.

Question: Let $d(s_i, s_j)$ is the length of the shortest path from s_i to s_j. Is there a node set of size p or less for G', i.e., a subset $M\subseteq V_s$ with $|M|\leqslant p$ such that

$$TTC_M=\sum_{\substack{\forall s_a,s_c\in V_s\\ a\neq c}} \min_{\substack{m_i\in M\\ 1\leqslant i\leqslant p}} w(s_a, s_c) \times\{d(s_a, m_i)+d(m_i, s_c)\}\leqslant B? \qquad (A.1)$$

We can restrict the p-CLS problem to a *vertex cover* problem by allowing only instances having $c(s_i, s_j)=1$ for all $\{s_i, s_j\}\in E_{ss}$ and having B is equal to the lower bound of our objective function in Eq. (4), say LB_{ss}. The NP-completeness of p-CLS problem can be derived from the following lemmas.

Lemma 1. *A subset M^* of V_s, with cardinality p, is a feasible solution of p-CLS problem for $G'(V_s, E_{ss})$ if and only if for each node pair of G' there always exists at least one node of M^* lying on one of its shortest paths.*

Proof. As we have described at Section 2, the lower bound of Eq. (4) can be reached only if, in the ideal case, for each gateway pair there always exists at least one CLS on one of its shortest paths. In other words, for each node pair of G' there always exists at least one node of M^* lies on one of its shortest paths if and only if the total transport cost for M^*, say TTC_{M^*}, is equal to LB_{ss}. Since LB_{ss} is the lower bound of the value of optimal solution, i.e., $TTC_M\geqslant LB_{ss}$ for all subset M of V_s. Therefore, M^* of size p for which total transport cost TTC_{M^*} is equal to

LB_{ss} if and only if M^* is an optimal solution of p-CLS of G'. □

Lemma 2. *For each node pair of G' there always exists at least one node of M^* lying on one of its shortest paths if and only if for each edge $\{s_i, s_j\} \in E_{ss}$ at least one of s_i and s_j belongs to M^*.*

Proof. ($\Rightarrow$) For those adjacent node pairs (s_i, s_j), there always exists at least one node of M^* lying on its only shortest path $\langle s_i, s_j \rangle$ indicating that $s_i \in M^*$ or $s_j \in M^*$.

($\Rightarrow$) For each node pair, its shortest path always contains no less than one edge. Thus, if for each edge $\{s_i, s_j\} \in E_{ss}$ at least one of s_i and s_j belongs to M^* then for each node pair there always exists at least one node of M^* lying on one of its shortest paths. □

Theorem. *The p-CLS problem is* NP-*complete even if the network is a planar graph of maximum vertex degree* 3 *all whose links are of cost* 1 *and all of whose node pairs have weight* 1.

Proof. We shall argue first that p-CLS $\in$ NP. Then we shall show that p-CLS is NP-hard.

To show that p-CLS belongs to NP: Suppose we are given a graph $G' = (V_s, E_{ss})$, and integers p and B. The certificate M we choose is the subset of V_s. The verification algorithms affirms that $|M| \leqslant p$, and then it checks whether the inequation (A.1) is satisfied. This verification can be performed straightforwardly in polynomial time.

Garey and Johnson have shown that the *vertex cover* in planar graphs with maximum degree 3 is NP-complete [23]. Let $G'(V_s, E_{ss})$ be a planar graph of maximum degree 3, edge length $c(s_i, s_j) = 1$ for all $\{s_i, s_j\} \in E_{ss}$ and weight $w(s_a, s_c) = 1$ for all node pairs (s_a, s_c) of G'. According to Lemmas 1 and 2, we can conclude that M^* is a vertex cover of size p for G. Therefore, the vertex cover problem is a special case of the p-CLS problem, thus completing the proof. □

References

[1] CCITT Recommendation I.211, B-ISDN service aspects, CCITT SGXVIII, TD42, 1992.

[2] CCITT Recommendation I.364, Support of broadband connectionless data service on B-ISDN, CCITT SGXVIII, TD58, 1992.

[3] D.S. Omundsen, R. Kaye and S.A. Mahmoud, A pipelined, multiprocessor architecture for a connectionless server for Broadband ISDN, *IEEE/ACM Trans. Networking* 2 (20) (1994) 181–192.

[4] P. Newman, ATM Local Area Networks, *IEEE Commun. Mag.* (March 1994) 86–98.

[5] B.J. Vickers and T. Suda, Connectionless service for public ATM networks, *IEEE Commun. Mag.* (August 1994) 34–42.

[6] S. Iisaku and M. Ishikura, ATM network architecture for supporting the connectionless service, in: *IEEE INFOCOM* (1990) 796–802.

[7] Y. Ujihashi, T. Shikama, K. Watanabe and S. Aoyama, An architecture for connectionless data service in B-ISDN, in: *IEEE GLOBECOM* (1990) 1739–1743.

[8] T. Kawasaki, S. Kuroyanagi, R. Takechi and K. Hajikano, A study on high speed data communication system using ATM, in: *IEEE GLOBECOM* (1991) 2105–2109.

[9] G. Fioretto, T. Demaria, R. Vaglio, A. Forcina and T. Moro, Connectionless service handling within B-ISDN, in: *IEEE GLOBECOM* (1991) 217–222.

[10] T.V. Landegem and R. Peschi, Managing a connectionless virtual overlay network on top of an ATM network, in: *IEEE ICC* (1991) 988–992.

[11] I.S. Venieris, J.D. Angelopoulos and G.I. Stassinopolous, Efficient use of protocol stacks for LAN/MAN-ATM interworking, *IEEE J. Selected Areas Commun.* 11 (8) (1992) 1160–1171.

[12] D.F. Box, D.P. Hong and T. Suda, Architecture and design of connectionless data service for a public ATM network, in: *IEEE INFOCOM* (1993) 722–731.

[13] K. Yamazaki, Y. Wakahara and Y. Ikeda, Networks and switching for B-ISDN connectionless communications, *IEICE Trans. Commun.* E76-B (3) (1993) 229–235.

[14] P. Corcetti, L. Fratta, M. Gerla, M.A. Marsiglia and D. Romanò, Interconnection of LAN/MANs through SMDS on top of an ATM network, *Comput. Commun.* 16 (2) (1993) 83–92.

[15] J.L. Boudec, A. Meier, R. Oechsle and H.L. Truong, Connectionless data service in an ATM-based customer premises network, *Comput. Networks ISDN Systems* 26 (1994) 1409–1424.

[16] M. Gerla, T.Y. Tai and G. Gallassi, Internetting LAN's and MAN's to B-ISDN's for connectionless traffic support, *IEEE J. Selected Areas Commun.* 11 (8) (1992) 1145–1159.

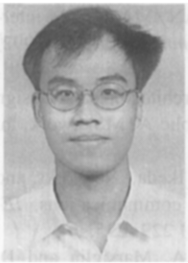
[17] M. Yamamoto, T. Hirata, C. Ohta, H. Tode, H. Okada and Y. Tezuka, Traffic control scheme for interconnection of FDDI networks through ATM networks, in: *IEEE INFOCOM* (1993) 411–420.

[18] W. Schödl, U. Briem, H. Kröner and T. Theimer, Bandwidth allocation mechanism for LAN/MAN internetworking with an ATM network, *Comput. Commun.* 16 (2) (1993) 93–99.

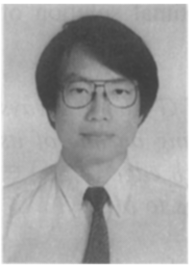
[19] J.A.S. Monteiro, S. Fotedar and M. Gerla, Bandwidth allocation, traffic control and topology design in ATM overlay networks, in: *IEEE GLOBECOM* (1994) 1404–1408.

[20] B. Stavrov, J. de Jogh, A. van Lomwel, R. in't Velt, *Bnet:*

Designing a connectionless broadband data service in an ATM network by simulated annealing, in: *Proc. 2nd Internat. Workshop on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS'95)* (1995) 253–256.
[21] M.R. Garey and D.S. Johnson, The rectilinear Steiner tree problem is NP-complete, *SIAM J. Appl. Math.* 32 (4) (1977).
[22] O. Kariv and S.L. Hakimi, An algorithmic approach to network location problems. Part 2: The *p*-medians, *SIAM J. Appl. Math.* 37 (3) (1979) 539–560.
[23] M. Katoh, H. Mukai, T. Kawasaki, T. Soumiya, K. Hajikano and K. Murakami, A network architecture for ATM-based connectionless data service, *IEICE Trans. Commun.* E76-B (3) (1993) 237–247.
[24] P. Corcetti, L. Fratta, M. Gerla and M.A. Marsiglia, SMDS multicast support in ATM networks, *Comput. Networks ISDN Systems* 27 (1994) 117–132.
[25] R.W. Floyd, Algorithm 97 (SHORTEST PATH), *Comm. ACM* 5 (6) (1962) 345.
[26] C. Rose, Mean internodal distance in regular and random multihop networks, *IEEE Trans. Commun.* 40 (8) (1992) 1310–1318.
[27] S. Fotedar, M. Gerla, P. Crocetti and L. Fratta, ATM virtual private networks, *Comm. ACM* 38 (2) (1995) 101–109.
[28] ATM Forum, Traffic management specification, ver. 4.0, 1995.



I-Fei Tsai received the B.S. degree in Computer Science and Information Engineering from Fu-Jen Catholic University, and the M.S. degree in Computer and Information Science from National Chiao Tung University, Taiwan, in 1992, and 1994, respectively. Since 1994, he has been working toward the Ph.D. degree in computer and Information Science at National Chiao Tung University, Taiwan. His research interests include LAN/MAN interconnection, resource management of integrated broadband networks, and media access control of wireless communication systems.



Rong-Hong Jan received the B.S. and M.S. degrees in Industrial Engineering, and the Ph.D. degree in Computer Science from National Tsing Hua University, Taiwan, in 1979, 1983, and 1987, respectively. He joined the Department of Computer and Information Science, National Chiao Tung University, in 1987, where he is currently a Professor. During 1991–1992, he was a Visiting Associate Professor in the Department of Computer Science, University of Maryland, College Park, MD. His research interests include computer networks, distributed systems, network reliability, and operations research.