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

트래픽 엔지니어링을 위한 명시적 경로 설정 절차
An Explicit Routing Procedure for Traffic Engineering

Chang Sup Sung and Woo Suk Yang
Department of Industrial Engineering, KAIST
Tel. 042-869-3154 , Fax. 042-869-3110
cssung@sorak.kaist.ac.kr, parani@kaist.ac.kr

Abstract : This paper deals with an offline routing problem, which can be used as an explicit routing procedure in MPLS(Multiprotocol Label Switching) network, for traffic engineering. This problem is formulated as an MIP(Mixed Integer Programming) with the objective function which is to minimize the sum of the maximum link utilization for load balancing (link utilization) and the routing cost. Constraints are composed of link capacity restriction and demand requirement that has origin-destination pair, bandwidth requirement and hop restriction. The problem is proved to be NP-hard so that the Lagrangean relaxation method is applied to propose a Lagrangean heuristic. To test the effectiveness & efficiency of the proposed algorithm, computational experiments are performed with numerical instances. The experiment results show that the proposed algorithm solves the problem within a reasonable time.

1. Introduction

Path selection methods for Internet are based on shortest path algorithms using simple metrics such as hop-count and delay in traffic transmission. Using such shortest paths allows IP routing to be made in large networks. However, it may cause the following problems [2] : the shortest paths from different sources may overlap at some links so as to cause congestion on those links, and traffic from a source to a destination may exceed the capacity of the associated shortest path, while the longer path between these two routers may be underutilized.

By performing load balancing in networks, ISPs(Internet Service Provider) can greatly improve resource utilization and network performance. Thereby, revenue can be increased without large investments on upgrading network infrastructures. Therefore, load balancing is greatly useful to ISPs [1][2]. One method for load balancing is to determine explicitly all or part of paths required for each demand in advance, which is called the explicit routing and is different from the traditional destination-based routing in the sense that it adapts a path selection method before transmitting traffic.

The proposed problem of this thesis is mainly concerned with path selection methods in multiprotocol label switched (MPLS) networks. The terminology has been regarded in the same light as the load balancing in the literature which is concerned with path selection procedure to make its associated network load-balanced which is interesting to this thesis.

The most important advantage of the MPLS method is capable of performing traffic engineering via explicit routing in IP networks. The MPLS method uses short and fixed-length labels in each packet header. A MPLS-capable router, called a label-switching router (LSR), is incorporated to examine labels in forwarding each packet. Whole MPLS process is depicted in Fig.1. The paths between the ingress LSRs and egress LSRs are called label-switched paths (LSPs). It is represented by bold lines in the figure.

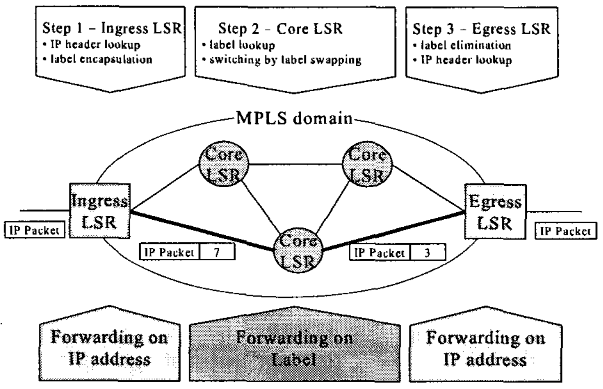

Fig.1. Whole MPLS Process

The MPLS path selection method for performing traffic engineering can be classified into the followings: online routing and offline routing. The online routing is activated by change in state or event and uses only local information in real-time. Therefore, the procedure may not be complex [3], so that as demand is very fluctuated over time, the online routing is expected to give a good performance. However, when demand appears to be of stationary characteristics, the online routing may not be attractive. That is because it uses only local information so that it does not accomplish the global optimum. In the situation, the offline routing rather is expected to give a better performance than the online routing. The offline routing functions periodically. An ISP's path selection method is applied to the network is based on a network characteristics.

As seen in the literature, many researches have been conducted to develop efficient routing algorithms for traffic engineering. Most online routing researches assume that only available bandwidth information on each link is given without knowledge of future demands. Their ultimate objective is to propose a simple metric which can be applied to the shortest path algorithm with given local information. Guerin et. al[4] have proposed an algorithm, based on the traditional min-hop algorithm, which attempts to perform traffic engineering. The algorithm, called the widest-shortest path algorithm (WSP), finds a feasible min-hop path between ingress and egress such that the chosen min-hop path has the maximum residual path bottleneck link capacity. Lar et. al[3] have developed a minimum interference routing algorithm (MIRA) that is based on the idea that a newly routed tunnel must follow a route that does not interfere too much with a route that may be critical to satisfy future demands between important ingress-egress pairs.

Most offline routing researches assume that all demands and available bandwidth information on each link are given. Heuristic approaches have been applied because the problem of traffic engineering is NP-hard [1][5]. Wang and Wang[1] have assumed that there is no hop restriction for each demand, and then have dealt with the routing problem for only load balancing without considering routing costs. They have applied LP relaxation and have developed a rerouting heuristic for the demands that violate integrality condition. Routing procedure for load balancing without considering routing costs may cause unnecessary operational costs. Although there are cheaper routes for some demands, their scheme may determine more expensive routes. Resende et. al[5] have assumed that there is no hop restriction for each demand, and then have dealt with the routing problem for optimizing the tradeoff between load balancing and routing costs. They have proposed an iterative heuristic which is composed of a construction phase to get an solution and a local search phase to improve the solution and make the solution feasible.

This thesis assumes that there is hop restriction for each demand, and then deals with an offline routing problem for optimizing the tradeoff between load balancing and routing costs. The proposed algorithm of this problem can be used as an explicit routing procedure in the MPLS path selection method, for traffic engineering. The problem can be proved to be an NP-hard problem so that the Lagrangean relaxation method is applied to propose a Lagrangean heuristic.

The remainder of this thesis is organized as follows. Section 2 is devoted to the problem description and formulation. A Lagrangean relaxation approach to the problem is presented in Section 3. In Section 4, an experimental design is performed to test the performance of the relaxation approach. Section 5 provides some concluding remarks.

2. Problem Description

The proposed backbone network is represented by a directed graph *G(N,E)*, where *N* is the set of nodes and *E* is the set of backbone links. The links in the network are directional, i.e., link *(i, j)* is different from link *(j, i)*, each with its own bandwidth capacity.

The nodes are composed of edge LSRs and core LSRs. For each demand, both of origin node and destination node must not belong to core LSRs, but belong to edge LSRs.

The followings are assumed. Routing costs include expenses for switching and maintaining route. Thus, this thesis assumes that the routing cost for each demand is proportional to any required bandwidth amount of each demand and unit operating cost of link. The proposed offline routing procedure for traffic engineering assumes that each demand informations is given. This is the common assumption in offline routing researches for traffic engineering[1][5]. Moreover, the route for each demand is assumed unsplittable. This means that the proposed procedure yields a single path that satisfies each demand. It is assumed that the network has the capability to cover all the given demands, which is reasonable, since the offline routing is concerned with reconfiguring any already-routed demands periodically to optimize network resources.

The following notation will be used throughout this thesis.

N	: Node set
E	: Link set
K	: Demand set
$E(k)$	: A set of the links which a demand $k \in K$ uses
$K((i,j))$	: A set of the demands which use a link $(i,j) \in E$
(i,j)	: A link from node i to node j
c_{ij}	: Bandwidth capacity of link $(i,j) \in E$
$cost_{ij}$	: Routing cost per unit demand using link $(i,j) \in E$
k	: A demand $(k \in K)$, $k = \langle p_k, q_k, d_k, h_k \rangle$
p_k	: Origin node for demand $k \in K$
q_k	: Destination node for demand $k \in K$
d_k	: Required bandwidth for demand $k \in K$
h_k	: Required hop count restriction for demand $k \in K$
w_1	: proportional constant for load balancing
w_2	: proportional constant for routing cost

The decision variables of the problem are denoted by

$$x_{ij}^k \begin{cases} 1, & \text{if demand } k \text{ is routed on link } (i,j) \\ 0, & \text{otherwise} \end{cases}$$

y : maximum link utilization $(0 \le y \le 1)$

Then, the proposed problem can be expressed as the following mixed integer problem :

Problem (HLBRP: Hop-constrained Load Balancing Routing Problem) :

$$Z_{opt} = \text{Min } w_1 y + w_2 \sum_{(i,j)\in E} \sum_{k \in K} cost_{ij}\, d_k\, x_{ij}^k$$

subject to

$$\sum_{j \in V} x_{ij}^k - \sum_{j \in V} x_{ji}^k = \begin{cases} 1, & \text{if } i = p_k \\ -1, & \text{if } i = q_k \\ 0, & \text{if } i \neq p_k, q_k \end{cases} \quad \forall k \in K (1)$$

$$\sum_{k \in K} x_{ij}^k \le y c_{ij} \qquad \forall (i,j) \in E (2)$$

$$\sum_{(i,j)\in E} x_{ij}^k \le h_k \qquad \forall k \in K (3)$$

$$x_{ij}^k \in \{0,1\} \qquad \forall k \in K,\ \forall (i,j) \in E (4)$$

$$0 \le y \le 1,\ y \in R^+ \qquad (5)$$

The objective function consists of the two parts. The former part represents the maximum link utilization and the latter represents the total routing costs. Constraints (1) contain the flow conservation equations which define a route (path) from p_k to q_k for each demand k. Constraints (2) represent the capacity utilization restriction on each link. Constraints (3) represents the hop restriction for demand k, which it means that demand k can not be routed over more than h_k links.

3. Solution Approach

Referring to [1], the Problem (HLBRP) can easily be proved to be NP-hard so that finding the optimal solution is a very difficult computational task. Moreover, any associated routing algorithm for Traffic Engineering is required to obtain routes quickly for each demand. Therefore, a heuristic algorithm to get a good, near-optimal solution is rather desirable.

In this thesis, the Lagrangean relaxation approach will be used to develop a heuristic solution procedure as a method for obtaining lower bounds for the minimization problem as well as upper bounds for finding good primal solutions for the proposed mixed integer programming problem. Moreover, in order to increase the efficiency of the heuristic algorithm, the Property 1 is derived.

Property 1.
Constraints (5) can be replaced with the following constraints :

$$\frac{mb}{mc} \le y \le 1,\ y \in R^+ \quad (5)'$$

where $mb = \operatorname{Max}_{k\in K} d_k$ is a constant value

and $mc = \operatorname{Max}_{(i,j)\in E} c_{ij}$ is a constant value.
(The proof is omitted.)

By Property 1, the lower bound obtained from the associated Lagrangean relaxed problem can be strengthen.

3.1. Lagrangean Relaxation

For the proposed problem, constraints (2) are dualized to obtain the following Lagrangean relaxation Problem $L(\lambda)$, which can be decomposed into two independent subproblems, $L_1(\lambda)$ and $L_2(\lambda)$.

Problem $L(\lambda)$:

$$Z_L(\lambda) = \text{Min } w_1 y + w_2 \sum_{(i,j)\in E} \sum_{k\in K} cost_{ij}\, d_k\, x_{ij}^k + \sum_{(i,j)\in E} \lambda_{ij}\left(\sum_{k\in K} d_k x_{ij}^k - y c_{ij} \right)$$

subject to (1), (3), (4) and $(5)'$,
where λ is the vector of $\{\lambda_{ij} \ge 0\}$.

Subproblem $L_1(\lambda)$:

$$Z_{L_1}(\lambda) = \text{Min } \sum_{k\in K} d_k \sum_{(i,j)\in E} (w_2 cost_{ij} + \lambda_{ij}) x_{ij}^k$$

subject to (1), (3) and (4).

Subproblem $L_2(\lambda)$:

$$Z_{L_2}(\lambda) = \text{Min } (w_1 - \sum_{(i,j)\in E} \lambda_{ij} c_{ij}) y$$

subject to $(5)'$.

For any $\lambda \ge 0$, the optimal objective function value of Problem $L(\lambda)$, $Z_L(\lambda)$, becomes a lower bound on Z_{opt} which is the objective function value of the original Problem (HLBRP).

One of the important factors for the relaxation method to be successful is concerned with how tight bound it can obtain. In the proposed problem, it is desired to determine the greatest lower bound by

$$Z_L(\lambda^*) = \operatorname{Max}_{\lambda\ge 0} Z_L(\lambda) = \operatorname{Max}_{\lambda \ge 0} \{Z_{L_1}(\lambda) + Z_{L_2}(\lambda)\}.$$

Property 2.
$Z_L(\lambda^*) \ge Z_{LP}$, where Z_{LP} is obtained in LP relaxation by ignoring constraints (4) of Problem (HLBRP).
(the proof is omitted.)

Property 2 shows that the proposed Lagrangean method of Problem (HLBRP) guarantees a tighter lower bound than the LP relaxation method.

3.1.1. Solution of Subproblem $L_1(\lambda)$

The Problem $L_1(\lambda)$ can be further decomposed into $|K|$ independent subproblems by each demand k, where the subproblem corresponding to demand k is denoted by $L_1^k(\lambda)$.

Subproblem $L_1^k(\lambda)$:

$$Z_{L_1}^k(\lambda) = \text{Min } d_k \sum_{(i,j)\in E} (w_2 cost_{ij} + \lambda_{ij}) x_{ij}^k$$

subject to
(1), (3) and (4), corresponding to demand $k \in K$.

For any given set of Lagrange multipliers $\lambda = \{\lambda_{ij}\}$, let's define an adjusted routing cost for each demand $k \in K$ and every link (i, j) as follows:

$$cost_{ij}^k = cost_{ij} + \frac{\lambda_{ij}}{d_k} \text{ for all } (i,j) \in A, \text{ all } k \in K.$$

If this transformation is used in $Z_{L_1}^k(\lambda)$, then the objective function of the Problem $L_1^k(\lambda)$ can be written as

$$Z_{L_1}^k(\lambda) = \text{Min } d_k w_2 \sum_{(i,j)\in E} cost_{ij}^k x_{ij}^k$$

To solve the problem $L_1^k(\lambda)$, the hop-constrained shortest directed path from p_k to q_k containing at most h_k links must be found, where the adjusted routing costs $cost_{ij}^k$ serve as arc lengths. Next, multiplying the length of the hop-constrained shortest p_k-to-q_k path by $d_k w_2$ gives the optimal value

$L_1^k(\lambda)$.

The hop-constrained shortest path problem can be solved by using a truncated version of the method of successive approximations[6] whose computational complexity is of $O(n^2 h_k)$.

The above algorithm is much more time-consuming than Dijkstra algorithm whose complexity is of $O(n^2)$, so that Problem $L_1^k(\lambda)$ is solved in this thesis as follows : the path for each demand is found by using the Dijkstra algorithm. If the path traverses the number of links more than h_k , then the above hop-constrained shortest path algorithm will be performed again for the corresponding demand. It is expected this method may be efficient when the hop restriction is loose.

3.1.2. Solution of Subproblem $L_2(\lambda)$

The problem $L_2(\lambda)$ is a simple LP problem, whose solution can be obtained as follows:

$$y = \begin{cases} \dfrac{mb}{mc}, & \text{if } (w_1 - \sum_{(i,j)\in E} \lambda_{ij} c_{ij}) > 0 \\ 1, & \textit{otherwise} \end{cases}$$

3.2. Subgradient Optimization

In this section, a good lower bound is to be obtained from a good set of multipliers, which is, in general, known to be a very difficult task except for a few special cases. One of the most popular methods to select values for the Lagrangean multipliers is known as the subgradient optimization algorithm, which will be used in this thesis.

In the subgradient optimization algorithm for the Problem $L(\lambda)$, Lagrange multipliers are generated using the following rule ;

$$\lambda_{ij}^{t+1} = \max\{0, \lambda_{ij}^t + T^t * G_{ij}\}, \quad \forall (i,j) \in E,$$

where λ_{ij}^t denotes the multiplier for an edge (i, j) at iteration t. G_{ij} is defined as subgradient for the relaxed constraints, evaluated at the current solution, by

$$G_{ij} = \sum_{k\in K} d_k x_{ij}^k - y c_{ij}, \quad \forall (i,j) \in E.$$

Let T^t denote step size, which is determined at iteration t as

$$T^t = \frac{\pi(\overline{Z_{opt}} - Z_L(\lambda^t))}{\sum_{(i,j)\in E} (G_{ij})^2},$$

where $\overline{Z_{opt}}$ represents the best upper bound among ones found until now by the Lagrangean heuristic in Section 3.3 and π, $0 \le \pi \le 2$ is a user defined parameter. The parameter is generally set to the value 2 at beginning of the procedure and then halved if the lower bound is not improved in a predetermined number of consecutive iterations (20 consecutive iterations in this thesis).

3.3. Heuristic Procedures

In this section, two heuristic solution procedures are introduced. One procedure is to generate an initial feasible solution which is used as the starting upper bound. The other one is to generate a feasible solution to the primal problem, Problem (HLBRP), at every iteration of the overall procedure which will be described in Section 3.4.

3.3.1. Procedure for an Initial Feasible Solution

This section presents the procedure called Procedure-IFS for an initial feasible solution which will be used as the starting upper bound for the overall procedure.

Procedure-IFS

Step 1: Order all the demands by sorting them in the non-increasing order of their requested bandwidth.

Step 2: Set f_{ij} to represent the current utilization for each link $(i,j) \in E$ and y to the value 0.

Step 3: For each demand k, find a hop-constrained shortest path

3.1: Perform the Dijkstra algorithm to find a path based on the following link cost metric; for each link $(i,j) \in E$,

$$a_{ij} = w_1\left(\frac{f_{ij}+d_k}{c_{ij}} + A \cdot \max\left\{0, \frac{f_{ij}+d_k}{c_{ij}} - y\right\}\right) + w_2 \cdot cost_{ij} \cdot d_k ,$$

where A is a tunable parameter.

3.2: If the path found by the Dijkstra algorithm does not violate the hop restriction, then route the d_k units of bandwidth along the shortest path from p_k to q_k containing at most h_k links and go to Step 4. Otherwise, proceed the next 3.3 step.

3.3: Perform the hop-constrained shortest path algorithm described in Section 3.1.1 to find a path based on the above link cost metric and route the d_k units of bandwidth along the hop-constrained shortest path from p_k to q_k containing at most h_k links.

Step 4: Update the link utilization f_{ij} and calculate the new y value.

Step 5: Take out the routed demand. If there is no demand to route, terminate the procedure. Otherwise, go to Step 3.

The link cost metric in Procedure-IFS is composed of the first part associated with load balancing and the second part associated with routing cost. In the first part, $\frac{f_{ij}+d_k}{c_{ij}}$ represents the increase of the link utilization and $\max\left\{0, \frac{f_{ij}+d_k}{c_{ij}} - y\right\}$ is the increase of the maximum link utilization caused by routing the demand. $w_2 \cdot cost_{ij} \cdot d_k$ is related to routing costs. Since one of the objectives is to minimize the maximum link utilization, the parameter A

is used as a penalty to keep the demand from increasing the maximum link utilization. It is set to the value 50 in this thesis.

3.3.2. Procedure for a Primal Feasible Solution

This section presents the procedure, called Procedure-PFS, for a feasible solution to the primal problem, at every iteration of the overall procedure which will be described in Section 3.4. When a solution to the Lagrangean relaxed problem $L(\lambda)$ is infeasible to the primal problem, this procedure is used. It is the method to reroute the demands that violate the capacity restriction on links. Before the procedure is described, it is helpful for the procedure to define some terminologies.

Let $\overline{x_{ij}^k}$ represent x_{ij}^k value in the optimal solution of the Lagrangean relaxed problem $L(\lambda)$. A link $(i,j)\in E$ is called a bottleneck link if utilization on the link exceeds its capacity (i.e., $\sum_{k\in K}\overline{x_{ij}^k} > c_{ij}$), where the set of bottleneck links is denoted by $\overline{E}$. Moreover, a demand k is called a bottleneck demand that traverses at least once on the bottleneck link. The maximum utilization among the links which belong to $E-\overline{E}$ is denoted by $\underline{y}$. It is almost similar to Procedure-IFS, but the objective demands for rerouting are different. This procedure reroutes some among bottleneck demands.

Procedure-PFS

```
Step 1: Calculate $f_{ij}$ , $y$ and $\underline{y}$ .
Step 2: Check $y$. If $y$ is not greater than 1, then terminate the procedure. Otherwise, continue next steps.
Step 3: Remove the demand corresponding to $arg \underset{k}{\text{Max}}\ L_1^k(\lambda)$ among the demands which use bottleneck links, and add the demand $k$ to the set of removed demands $R_K$.
Step 4: Update $f_{ij}$ and $y$ . And check $y$ such that if $y$ is not greater than $\underline{y}$, then continue the next step. Otherwise, go to Step 3.
Step 5: Reroute all demands in $R_K$, one at a time in the First-In-First-Out manner by the procedure of Step 6.
Step 6: For each demand $k\in R_K$, find a hop-constrained shortest path
    6.1: Perform the Dijkstra algorithm with link cost metric $a_{ij}$ .
    6.2: If the path found by the Dijkstra algorithm does not violate the hop restriction, route the $d_k$ units of bandwidth along the shortest path from $p_k$ to $q_k$ containing at most $h_k$ links and go to Step 7. Otherwise, proceed the next Step 6.3.
    6.3: Perform the hop-constrained shortest path algorithm with the above link cost metric and route the $d_k$ units of bandwidth.
Step 7: Update $f_{ij}$ and calculate the new $y$.
Step 8: Take the rerouted demand out of $R_K$. If there is no demand to reroute in $R_K$, then terminate the procedure. Otherwise, go to Step 5.
```

3.4. Overall Procedure

In this section, the overall procedure called Procedure-OP is described. It is terminated after a specific number (set to the value 1000 in this thesis) of iterations or when π defined in Section 3.2 is too small ($\pi\le 0.005$ in this thesis) so that it can hardly improve solution. Also, it is terminated when the solution gap $\left(\frac{(\overline{Z_{opt}})^* - Z_L(\lambda^*)}{Z_L(\lambda^*)}\times 100\right)$ is less than 0.1%.

Procedure-OP

```
Step 1: Generate the initial primal feasible solution $\{ (x_{ij}^k)^0\}$, and $y^0$ by using Procedure-IFS. Compute the corresponding value of $Z_{opt}$ and set the current upper bound on $Z_{opt}^*$, $\overline{Z_{opt}}$ at the value. Set the best upper bound on $Z_{opt}^*$, $(\overline{Z_{opt}})^*$ to $\overline{Z_{opt}}$ , and set the best feasible solution until now as $\{ (x_{ij}^k)^*\}=\{ (x_{ij}^k)^0\}$, $y^*=y^0$.
Step 2: Initialize Lagrange multipliers and the parameters.
    2.1: Set the improvement counter as $imp$=0, the iteration counter as $t$=0, the stepsize as $T^t$ =2 and $\pi$=2.
    2.2: Set Lagrange mutipliers $(\lambda_{ij})^t$ at the value 1 and let $(\lambda_{ij})^*=(\lambda_{ij})^t$ and set the current best value of $Z_L(\lambda^*)$ to negative infinity ( $10^{-10}$ being used in this thesis).
Step 3: Solve the Lagrangean Problem $L(\lambda^t)$ using $(\lambda_{ij})^t$ as the Lagrange multipliers, and obtain the values for $Z_L(\lambda^t)$, $(x_{ij}^k)^t$, $y^t$.
Step 4: Update the lower bound and generate the upper bound.
    4.1: If $Z_L(\lambda^t)> Z_L(\lambda^*)$, then let $Z_L(\lambda^*)= Z_L(\lambda^t)$, $(\lambda_{ij})^*=(\lambda_{ij})^t$, and $imp$=0. Otherwise, let $imp$=$imp$+1.
    4.2: If the value of $imp$ reaches a prespecified limit (20 in this thesis), then set as $\pi=\pi/2$ .
Step 5: Generate the upper bound and update it and the best feasible solution until now.
    5.1: Generate a primal feasible solution $\{ (x_{ij}^k)^t\}$ and $\{ y^t\}$ by using Procedure-PFS (see Section 3.3.2). Compute the corresponding
```

value of Z_{opt} and set the current upper bound on Z^{*}_{opt}, $\overline{Z_{opt}}$ at the value.

5.2: If $\overline{Z_{opt}} < (\overline{Z_{opt}})^{*}$, then let $(\overline{Z_{opt}})^{*} = \overline{Z_{opt}}$, $\{(x^{k}_{ij})^{*}\} = \{(x^{k}_{ij})^{t}\}$ and $y^{*} = y^{t}$.

Step 6: Terminate this procedure when the iteration counter t exceeds a prespecified limit(1000 iterations in this thesis) or π defined in Section 3.2 is too small ($\pi \le 0.005$ in this thesis) or the solution gap satisfies the relation $\frac{(\overline{Z_{opt}})^{*} - Z_L(\lambda^{*})}{Z_L(\lambda^{*})} \times 100 \le 0.1\%$.

Step 7: Compute a new subgradient and update the Lagrange multipliers.

7.1: Compute a new subgradient :

$$G_{ij} = \sum_{k \in K} d_k x^{k}_{ij} - y c_{ij}\ , \quad \forall (i,j) \in E$$

7.2: Update the Lagrange multipliers for the $(t+1)^{st}$ iteration :

$$\lambda^{t+1}_{ij} = \max\{0, \lambda^{t}_{ij} + T^{t} * G_{ij}\},\ \forall (i,j) \in E,$$

where $T^{t} = \frac{\pi(\overline{Z_{opt}} - Z_L(\lambda^{t}))}{\sum_{(i,j) \in E} (G_{ij})^2}$

7.3: Set $t = t + 1$ and go to Step 3.

4. Computational Results

The Lagrangean-based algorithm presented in Section 3.4 is programmed in C language and tests are performed on a Pentium 4 1.7GHz personal computer. All of the tests in this thesis are performed using a network topology of [3] which has 15 nodes and 56 directional links, as shown in Fig.2. In Fig.2, 7 gray-colored nodes represent edge nodes which can be starting nodes or destination nodes for a demand and each undirected link represents two directed links oriented in opposite directions. Thus, the maximum number of different O-D pairs is 7 ×6=42.

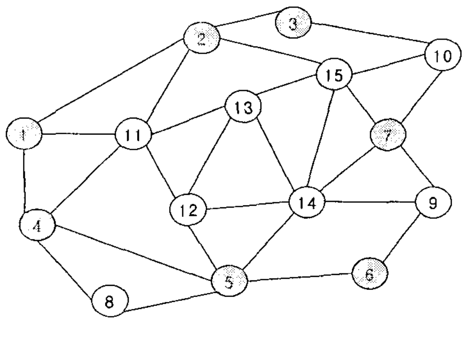

Fig.2. An Illustrative Network Topology

To test the effectiveness of the suggested algorithm, the gap between a heuristic solution value and the lower bound obtained by solving the Lagrangean dual problem is used. The total elapsed time in the simulation is used to test the efficiency of the algorithm.

To investigate how different configurations affect the effectiveness and efficiency of the proposed algorithm, the following four experiments are designed;

1. Experiment on (w_1,w_2)'s effect
2. Experiment on capacity size effect
3. Experiment on the effect of tightness of hop-restriction
4. Experiment on the effect of O-D pairs' distribution.

Except where it is specially noted, experiments have the following conditions. (w_1, w_2) is set at the pair of values (10000, 1), the size of link capacity is generated in U[150, 300] and link cost per unit demand is generated in U[1, 5]. For each demand, the hop restriction is generated in U[MH, 3MH] where MH represents the minimum number of hops by which the demand can be transported, the bandwidth requirement is generated in U[1, 30] and the O-D pair is selected randomly from all the 42 pairs between edge nodes. Each experiment is performed with 5 respective instances which is randomly generated according to the number of demands varying from 60, 70, 80, 90, and 100 so that total 25 tests according to each experiment are performed.

4.1. Experiment on (w1,w2)'s effect

This experiment is performed with the same conditions when (w_1, w_2) is considered at each of the pair values (1000, 1), (10000, 1) and (50000, 1). The result of these cases are shown in Table 1.

Table 1. Experiment Result for (w1,w2)'s effect

|K|	w_1=1000		w_1=10000		w_1=50000	
	Gap	Time	Gap	Time	Gap	Time
100	2.850	70.277	5.056	72.479	5.185	77.431
90	3.650	68.564	4.530	69.041	4.625	69.368
80	1.393	49.988	5.584	58.565	7.445	56.664
70	1.183	61.122	5.218	46.206	7.353	43.278
60	0.932	46.162	4.552	46.264	5.063	54.417
Ave.	2.002	59.223	4.988	58.511	5.934	60.232

All the instances were solved within 110 seconds. There was no consistent difference in total elapsed time according to the three cases. All the instances had gaps within 10%, while in the case where the contribution of routing costs to the objective value is larger than the contribution of load balancing, the proposed algorithm seems to give better performance than in the other cases. It is because the constraints which contain the maximum link utilization term related to load balancing are relaxed so that Subproblem $L_2(\lambda)$ become too simple. Therefore, it is thought that the method which tightens the feasible region of Subproblem $L_2(\lambda)$ having the maximum link utilization term related to load balancing is necessary as a further research. For example, the method of inserting any redundant constraints into the original problem may be interesting to consider.

In the subsequent experiments, w_1 is set at the value 10000. However, as seen above, the author think the proposed algorithm may give better performance than this case when w_1 is set at any less value than 10000.

4.2. Experiment on capacity size effect

This experiment is performed with the same conditions when the link capacity is generated in U[150, 300] and U[250, 500]. The result of these cases are shown in Table 2.

Table 2. Experiment Result for Capacity Size Effect

|K|	Capacity~U[150,300]		Capacity~U[250,500]	
	Gap	Time	Gap	Time
100	4.633	81.908	5.705	51.497
90	7.707	54.283	6.787	51.067
80	6.495	59.132	6.063	41.411
70	8.070	42.869	6.915	39.514
60	6.011	35.357	5.136	36.474
Ave.	6.583	54.710	6.121	43.993

Referring to Table 2, all the instances were solved within 124 seconds and there was no consistent difference in total elapsed time between the instances of large link capacity and the instances of small link capacity. All the instances had gaps within 10.6%. And there was no consistent difference in gap between the two cases.

4.3. Experiment on the effect of tightness of hop restriction

This experiment is performed with the same above conditions when the hop restriction for each demand is either tight or loose. In the tight case, the hop restriction for each demand is set to MH+1 and in the loose case, it is set to 3MH. The result of these cases are shown in Table 3.

Table 3. Experiment Result for the Effect of Tightness of Hop Restriction

|K|	Hop-restriction=MH+1		Hop-restriction=3MH	
	Gap	Time	Gap	Time
100	2.949	71.265	5.471	63.097
90	3.021	80.919	3.831	77.271
80	1.579	84.256	6.128	62.385
70	5.914	58.577	9.142	46.915
60	4.784	43.464	5.448	33.058
Ave.	3.649	67.696	6.004	56.545

Referring to Table 3, all the instances were solved within 122 seconds, and for the respective number of demands on average, the tight case consumed more time than the loose case. It is be because the hop-constrained algorithm which is time-consuming is used in the tight case more than in the loose case. In the proposed algorithm, when $L_1(\lambda)$ is -solved and, Procedure-IFS and Procedure-PFS are performed, the path for each demand is found by using the Dijkstra algorithm and the demands corresponding to any paths which violate the associated hop restriction, among all the paths, is rerouted by using the hop-constrained algorithm presented in the Section 3.1.1.

All the instances had gaps within 13.1%. The tight case outperformed the loose case. It is because the tight case has the number of feasible paths for each demand less than the loose case. In other words, the feasible region in the tight case is smaller than in the loose case so that enough path search for each demand using the proposed algorithm can be performed in the tight case. Therefore, it is thought that the research on any problem reduction scheme is further needed, especially in the loose case.

4.4. Experiment on the effect of O-D pairs' distribution

This experiment is performed with the same above conditions when the demand's spread is either sparse or dense. In the sparse case, the O-D pair for each demand is selected randomly from among all the 42 pairs between edge nodes. In the dense case, it is selected randomly from among 10 pairs out of 42 pairs (selected in advance). The results of these cases are shown in Table 4.

Table 4. Experiment Result for the Effect of O-D Pairs' Distribution

|K|	Dense		Sparse	
	Gap	Time	Gap	Time
100	6.451	57.502	6.452	60.999
90	7.739	21.782	8.302	18.784
80	4.818	54.949	5.942	76.882
70	6.558	39.373	9.648	39.908
60	7.739	45.375	7.073	37.402
Ave.	6.393	46.457	7.554	44.202

Referring to Table 4, all the instances were solved within 99 seconds and had gaps within 12.9%. The consistent difference in the average gap and in total elapsed time can not be found between the two cases.

5. Concluding Remarks

This thesis deals with an offline routing procedure, which can be used as an explicit routing procedure in MPLS, for load balancing and routing costs optimization.

The problem is formulated as an MIP (Mixed Integer Programming) with the following objective function and constraints. The objective function of the problem is to minimize the sum of the maximum link utilization and the routing cost. Constraints include link capacity restriction and demand requirement that have origin-destination pair, bandwidth requirement and hop-restriction. The problem is proved to be NP-hard so that the Lagrangean relaxation method is applied and a Lagrangean heuristic is proposed. In order to evaluate effectiveness and efficiency of the proposed algorithm, computational experiments were performed with the instances that were randomly generated to reflect various cases. The experiment results showed that the proposed algorithm solved the problem within a reasonable time and produced good solutions with the total average gap of 2.0% ~ 7.6% according to each case. It is also found that the proposed algorithm gives a better performance when the contribution of the routing cost to the objective value is larger than the contribution of load balancing and when the hop restriction is tighter than any opposite case.

For a further research, the followings may be interesting to make the proposed algorithm stable ; the method of tightening the feasible region of Subproblem $L_2(\lambda)$ especially when the contribution of load balancing to the objective value is large, and the research on problem reduction scheme especially when the hop restriction for each demand is loose. Moreover, it may be interesting to deal with the problem considering any discrete link capacity expansion which is, however, a non-linear problem because the decision variable y is multiplied by another decision variable of link capacity expansion.

REFERENCES

[1] Yufei Wang, Zheng Wang, "Explicit Routing Algorithms for Internet Traffic Engineering", Proceedings Eight International Conference on Computer Communications and Networks, pp.582-588, 1999

[2] Xipeng Xiao, Alan Hannan, Brook Bailey, "Traffic Engineering with MPLS in the Internet", IEEE Network, Vol.142, pp.28-33, 2000

[3] Koushik Lar, Murali Kodialam, T.V. Lakshman, "Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications", IEEE Journal on selected areas in communications, Vol.18, No.12, Dec 2000

[4] R. Guerin, D. Williams, A. Orda, "QoS Routing Mechanisms and OSPF Extensions", Proc. Globecom, 1997

[5] Mauricio G. C. Resende, Celso C. Ribeiro, "A Grasp with Path-Relinking for Private Virtual Circuit Routing", AT&T Labs Research Technical Report, June 2001

[6] E. L. Lawler, "Combinatorial Optimization: Networks and Matroids", Holt, Rinehart & Winston, New York