Network Efficient QoS Routing for Multimedia Services in MPLS Networks

HahnEarl Jeon

The Graduate School
Yonsei University
Department of Electrical and Electronic Engineering

Network Efficient QoS Routing for Multimedia Services in MPLS Networks

A Dissertation
Submitted to the Department of
Electrical and Electronic Engineering
and the Graduate School of Yonsei University
in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

HahnEarl Jeon

December 2002

This certifies that the dissertation
of HahnEarl Jeon is approved.

Thesis Supervisor: Jaiyong Lee

Daesik Hong

Young-Keun Park

Chulhee Lee

Young Yong Kim

The Graduate School
Yonsei University
December 2002

Acknowledgements

7 년간의 연구실 생활을 마감하면서 이 논문은 그 작은 결실입니다. 저에게 많은 격려와 도움을 주신 모든 분들께 감사의 뜻을 전합니다.

지난 7 년간 지도와 격려를 아끼시지 않으셨던 이재용 교수님께 감사드립니다. 보다 나은 결실을 맺도록 지도해 주신 여러 교수님께 감사드리며, 특히 제 논문을 심사해 주시며 많은 조언을 아끼시지 않으신 홍대식 교수님, 박영근 교수님, 이철희 교수님, 김영용 교수님께 감사드립니다. 지금은 은퇴하셨지만 연구실원의 귀감이 되어주신 이상배 교수님의 건강을 기원합니다.

연구실 생활이 즐거울 수 있게 만들어준 91 학번 동기들 중신, 재성, 용훈, 익순, 영진, 성이, 재민이와 7 년간의 연구실 생활에서 인연이 되었던 모든 선후배님들께 감사를 드립니다.

힘들때면 같이해주고 지금은 가족처럼 되어버린 선영누나와 승한. 신사동에서 몰려다니던 고등학교 친구들 성무, 한승, 준명, 호원, 창오, 근만, 상남, ... 대학때 열정적으로 생활했던 사발 친구들 남훈, 강혁, 승우, 영재, 한수, 호기. 다시 만나게 되어서 반가운 국민학교 친구들 지윤, 용환, 대성, 승현, 정희, 재범, 정현, 현아, 덕수, 진호,.... 대학원 생활의 마지막을 즐겁게 마무리하게 해준 원동로얄친구들 경혁, 용우, 지원, 현주, 정화, 영윤, 수현형. 대학원 생활을 시작할 때 알게 되고 지금까지도 좋은 만남이 계속 되고 삶을 윤택하게 해준 재즈방 식구들. 좋은 동생들 승온, 수정, 정심, 동묵, 세욱. 늘 부러운 드러머 동진. 모두 저에게는 소중한 재산입니다.

오랫동안 공부하는 아들을 믿어주시며 사랑으로 보살펴주신, 제가 존경하는 아버님과 어머님께 이 논문을 바칩니다. 사랑하는 동생 수연이의 행복을 기원하며, 오랜 대학원 기간을 기다려 주고 평생을 같이할 사랑하는 나의 수영이와 저에게 꾸준한 관심을 보여주신 친지분들과 저를 아껴주신 모든 분들과 이 작은 기쁨을 함께하고자 합니다.

2002 년 12 월

드림.

Table of Contents

List of Figures

List of Tables

Abstract

Network Efficient QoS Routing for Multimedia Services in MPLS Networks

HahnEarl Jeon
Electrical and Electronic Eng. Dept
The Graduate School
Yonsei University

This dissertation investigates the single path calculation in order to satisfy the load balancing of the traffic. Today's Internet is a best effort service based on a single metric path selecting algorithm that can only support path for one requirement. Traffic is sent according to the selected path whatever the type of service is served. Various network services such as multimedia traffic require different quality-of-service (QoS). The Internet Engineering Task Force (IETF) has proposed many service models and mechanisms to meet the demand for QoS. Multiprotocol label switching(MPLS) received attention in many studies for its usefulness in supporting multiple protocol and fast transmission and the ability of reserving and controlling paths. MPLS is also a useful tool in traffic engineering.

In order to support QoS call, path is subject to multiple constraints on routing metric. In many cases, the path finding problem is NP-complete.

This dissertation proposes a routing algorithm that finds a feasible path in polynomial time. The algorithm uses cost metric that is based on delay metric and influenced by resource bandwidth metric and network state for traffic distribution. The use of dynamic network status enables the algorithm to cope with the network situation. The algorithm uses shortest path when network load is light and performs traffic engineering by moving to alternate path when network link is crowded.

The routing algorithm is implemented in routing server architecture and proposed routing manager structure. The routing process follows the constraint based routing for MPLS network to serve QoS and best-effort traffic. The performance is compared with other routing algorithms through simulation.

Key words : QoS, routing, multimetric, traffic engineering

Chapter 1

Introduction

1.1. Motivation

Traditional communication network architectures were designed to support users with simple quality of service(QoS). The Internet protocol architecture is based on a connectionless end-to-end packet service using the IP protocol. Routing in today's Internet is focused on connectivity that has the capability to serve only the "best-effort" service, which means it will try its best to forward user traffic. It is an unreliable packet delivery to the destination independently of any previous or following packets. Data packets of a session may follow different paths to the destination. Network resources are shared by packets from different sessions. Therefore, "best-effort" service can provide no guarantees regarding loss rate, bandwidth, delay, delay jitter, etc.

Exponentially increasing users and supporting various multimedia applications such as digital video and audio has increased the demand for wide spectrum of network service and broadband integrated service networks are expected to support multiple and diverse QoS requirements. These new applications require better transmission services than "best-effort" service. They need to find appropriate and available network resources and reserve it in order to support and guarantee the specific service.

The Internet Engineering Task Force (IETF) has proposed many service models and mechanisms to meet the demand for QoS. Among them are the integrated services/Resource Reservation Protocol(RSVP) model, the differentiated services (DS) model, multiprotocol label switching(MPLS), traffic engineering, and constraint based routing.

Routing is a call-level network control mechanism through which a path is derived for establishing communication between a source and a destination in a network. Current Internet routing protocols, e.g. OSPF, RIP use “shortest path routing”, i.e. routing that is optimized for a single arbitrary metric[Huitema]. Efficient algorithms for computing shortest paths have been used in communication networks. Path selection within routing is typically formulated as a shortest path optimization problem. Routing algorithms capable of “best-effort” routing can not support the QoS required by multimedia services with multiple metric constraints. Different from “best-effort” services, QoS service requires resource reservation. Moreover, Finding Paths with multiple constraints within polynomial time is needed. Best-effort routing algorithms use one metric to calculate the path. In order to satisfy the diverse QoS requirements, multiple metric calculation is required. However, it is known that this can not be done in real time basis. The computation becomes more difficult as constraints are introduced into the path calculation problem.

QoS routing refers to algorithms that compute paths that satisfy a set of end-to-end QoS requirements. QoS routing is one of the tools available to accommodate flows which expect certain QoS guarantees from the network. It enables the network to identify the sufficient resource path through the network. QoS routing is seen as a missing piece in Internet and an important factor in order serve various qualities required from network. QoS routing has been receiving attention in order to overcome IP limitation to serve various qualities of connection rather then best-effort service. In the past several years, QoS routing has been subject of several studies and proposals.

In order to extend current routing, routing algorithms calculate path under multiple constraints. According to current works, calculating most suitable path using more than one metric is known to be unfeasible[Wang 1996]. Multi-criteria shortest path problem is not well defined. Therefore, it is important to select most reliable path among many path and to optimize it. If the given metric is a dynamic metric that changes according to network status, accurate value is essential to select the path. The subjectivity of today's QoS requirements and the complex trade-off among them make it difficult to define an appropriate routing metric. Moreover, with distinct characteristics of different traffic types, the same metric is not universally applicable. The introduction of QoS negotiation further renders the meaning of an optimal path indeterminable. There is little or no additional utility for having a path whose associated QoS is more desirable than the user-specified QoS. On the other hand, there is considerable disutility for failing to find a path that meets the user-specified QoS. Hence, a new routing paradigm that emphasizes searching for an acceptable path satisfying various QoS requirements is needed for integrated communication networks.

Apart from issues of user-specified QoS requirements, there is also the problem of routing in a dynamic environment due to transient fluctuations in offered load, time-of-day changes in service demand, and incidental disruptions in the network. In addition, the routing topology may also change because of the possibility of dynamically adding and removing transmission facilities. Network size and topology aggregation and tradeoff with network traffic and routing update cause inaccurate routing information. Thus, modern routing strategies must be designed to adapt to changes in the network. [Guerin 1997] deals with the impact of inaccuracy on the ability of path-selection. Even though, given the QoS request of a flow or an aggregation of flows, a path that will provide best service to the connection must be selected.

When network load is light, best-effort service and integrated service differ little. Performance degrades when traffic load is heavy. Network congestion can be caused by lack of network resources or uneven distribution of traffic. Traffic engineering is a process of arranging how traffic flows through the network so that congestion caused by uneven network utilization can be avoided. Changing path to lower cost ([Salama], [Vutukury 1999]) or sending traffic along multiple path ([Rao]), setting cost according to bandwidth availability ([Fortz]) is a way to realize traffic engineering.

1.2 Research Goal

The process of QoS routing is very complicated and in order to satisfy the request of various quality of service paths routing function must work closely with other network functions. There are various problems regarding QoS routing. My research focuses on calculating path to support QoS routing. My interest is in the problems of multiple constraint and inaccurate network information. The research goal of this dissertation is as follows.

1. providing simple solutions to the NP-complete multi-metric QoS routing problem
2. providing network efficient routing solutions for dynamic networks with QoS traffic
3. providing routing solutions to support QoS traffic and best-effort traffic efficiently in MPLS networks

Simple path calculation and selection for multi metric constraint routing problem is studied.

1.3 Outline of the Dissertation

The remainder of the dissertation is organized as follows. Chapter 2 presents the terminology and basic concept of routing problem and traffic engineering. An overview of existing QoS routing algorithms are presented in Chapter 3. Basic algorithms are reviewed and compared. In Chapter 4, the problem of QoS routing in a dynamic network is studied. Widest-least cost algorithm is proposed to perform dynamic routing using network state dependent cost function. Chapter 5 discusses the implement of the proposed widest-least cost algorithm to support QoS and best-effort traffic. The performance is evaluated and compared with other existing routing algorithm through simulation in Chapter 6. Chapter 7 extends the proposed unicast algorithm to support multicast by selecting best path in setting up multicast routing tree. Chapter 8 concludes the dissertation.

Chapter 2

Preliminary

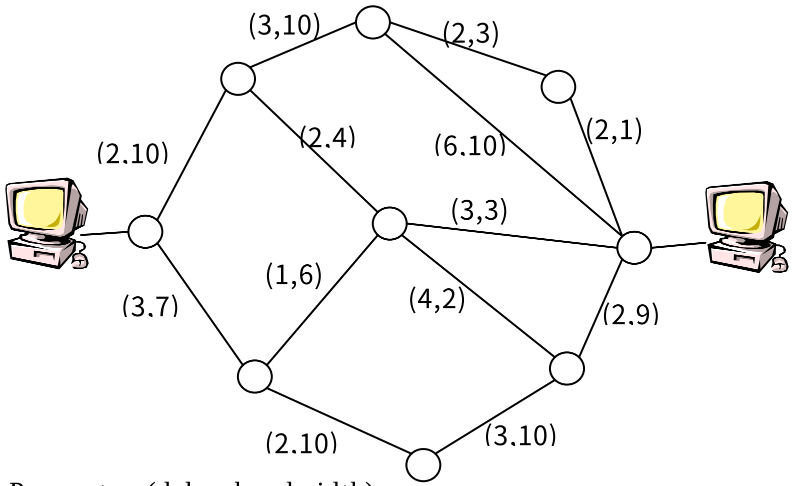
In this chapter, the basic concept and terminology used in the rest of the dissertation is introduced.

2.1 Graph Model

A network can be modeled as a graph G=(N,A). It is a finite nonempty set N of nodes and a collection A of pairs of distinct nodes from N. Each pair of nodes in A is called an arc. The nodes represent the router nodes and the arcs represent the communication links in the network.

In many applications involving graphs, it is useful to introduce a variable that measures the quantity flowing through each arc, like for example, electric current in an electric circuit, or water flow in a hydraulic network. Such variable is referred as the flow of an arc. The research interest is about the traffic flowing over the arcs of a network. Mathematically, the flow of an arc is simply a scalar value which each link is characterized and state measured.

Figure 2.1 shows an example of a network. Among possible link characteristic values (e.g. delay, bandwidth, jitter, cost, etc), links connecting the nodes are characterized by propagation delay and link bandwidth. These values are assigned to each link as metric.



Parameter=(delay, bandwidth)

Figure 2.1 Example network topology

2.2 Quality of Service (QoS) Metrics and Services

As defined in [Orda 1998], Quality-of-Service is “a set of service requirements to be met by the network while transporting a flow.” Standards define QoS on an end-to-end basis – the perspective most relevant to an end user. End-to-end QoS contains one or more intervening networks, where each of these networks may cause impact in end-to-end QoS due to multiplexing, switching, or transmission in the network.

A metric is a characterization parameter for a path or link in a network. Link and node metrics are used to represent the basic network properties of interest. A QoS metric is a metric that characterizes the quality of the path. Well-known metrics are delay, jitter, bandwidth, loss probability, etc. Since the flow QoS requirements have to be mapped onto path metrics, the metrics define the types of QoS guarantees the network can support. Four types of metrics are defined as follows:

A metric d is additive, if $d(p) = d(i,j) + d(j,k) + \cdots + d(l,m)$..

A metric d is multiplicative, if $d(p) = d(i,j) \times d(j,k) \times \cdots \times d(l,m)$..

A metric d is concave, if $d(p) = \min[d(i,j), d(j,k), \cdots, d(l,m)]$.

A metric d is convex, if $d(p) = \max[d(i,j), d(j,k), \cdots, d(l,m)]$.

Bandwidth is an example a concave metric, because on a path, the link with the smallest bandwidth determines the total available bandwidth on the path. Propagation delay is additive because it is the result of the sum of all the propagation delays on each link.

QoS service is the service that meets the users with the service quality that they have requested. Different kind of service requires different service quality. The different QoS services can be implemented by giving different packets different priorities in the router queues and/or by choosing appropriate routes.

The designers of the Internet Protocol already recognized the problem that packets of applications requiring different QoS delivery required a way to be differentiated and did reserve the second octet in the IP header to specify the Type of Service [RFC791]. The TOS field, which is 8 bits wide, contains 3 bits to indicate precedence as an indication of priority and 5 bits to indicate type of service for routing. Routing protocol normally attempt to compute the best route to a destination, but there are several algorithms, notably OSPF and BGP, support routing by “type of service.” Unfortunately, this field in the IP header is still ignored by most TCP/IP implementations.

In order to make it possible to provide different QoS services, it must be possible to identify the type of service that a packet belongs to. In other words, it must be possible to say for example "this packet belongs to a data stream (or "flow"), which requires the lowest possible delay jitter. Therefore, different from "best-effort" services, QoS service usually requires resource reservation. A path is pre-determined and associated resources(link bandwidth, buffer space, etc.) along the path are reserved before the actual transmission. In other words, the path or connection between source and destination is setup first. When the transmission finishes, the path and associated resources are released. To reserve the resources of a flow, the routers along the path need to keep track of the state of the flow. Some information is maintained in the routers regarding the state of the flow. In short, to provide QoS, both connection and state information are needed.

To provide QoS in the Internet, many techniques have been proposed and studied, including Integrated Services(IntServ) [RFC 1633], Differential Services(DiffServ) [RFC 2475], MPLS (MultiProtocol Label Switching) [RFC 3031], Traffic Engineering and QoS-based Routing. Specific working groups are also organized under IETF(Internet Engineering Task Force).

IntServ requires applications to signal their service requirements to the network through a reservation request. It serves QoS by differentiation of packets on a flow level, such that packets in the same flow are treated equally. Currently it uses RSVP as its end-to-end signaling protocol. Using RSVP, packets passing through a gateway host can be expedited based on policy and reservation criteria arranged in advance. In an IntServ system, the network is re-configured to serve a particular flow, with it's QoS guarantees, each time a new flow is started. It is a connection-oriented system where the QoS provisioning is made on connection setup and is very similar in concept to circuit-switched systems. Unfortunately, the IntServ/RSVP architecture does not scale to the global Internet.

The overhead associated with IntServ networks is huge for large networks. The overhead posed on backbone routers is enormous and impracticable. DiffServ is intended to address the scalability requirements for the global Internet. DiffServ works in the core of the network through a scalable aggregated service mechanism. In Differentiated Services system, the traffic entering a network is classified and possibly conditioned at the boundaries of the network, and assigned to different QoS behaviors. The type of QoS behavior is encoded in the DS codepoint, packets are classified as belonging to a flow if they have the same marking in their ToS byte (IPv4) or traffic-class byte (IPv6). This QoS behavior of a packet traversing a node is called per-hop behavior (PHB).

Diffserv takes the IP TOS field, renames it the DS byte, and uses it to carry information about IP packet service requirements. It operates at Layer 3 only and does not deal with lower layers. On the other hand, MPLS specifies ways that Layer 3 traffic can be mapped to connection-oriented Layer 2 transports like ATM and Frame Relay. MPLS adds a label containing specific routing information to each IP packet and allows routers to assign explicit paths to various classes of traffic. It also offers traffic engineering and techniques that can boost IP routing efficiency.

2.2 Routing

Routing algorithm is a network layer protocol that guides packets through the communication subnet to their correct destination. The times at which routing decisions are made depend on whether the network uses datagrams or virtual circuits. In a datagram network, two successive packets of the same user pair may travel along different routes, and a routing decision is necessary for each individual packet. In a virtual circuit network, a routing decision is made when each virtual circuit is set up.

Routing in a network typically involves a rather complex collection of algorithms that work more or less independently and yet support each other by exchanging services or information. The routing of a packet consists of two steps. First, collect the information from the network needed to establish a correct connection across the network. If the information of the network cannot be acquired and cannot set up a path across the network, every packet generated must be broadcasted to all nodes, which is obviously impractical. Network information can be exchanged between neighbor nodes or flooded in the network. Next, calculate path to destination using the network information acquired and forward the packet according to the path selected.

In a directed network G = (N, A) with an arc length a_{ij} associated with each arc $(i,j) \in A$. Problem of finding a shortest path from node s to node t is to

Minimize $\sum_{(i,j)\in A} a_{ij} x_{ij}$

Subject to $$\sum_{\{j|(i,j)\in A\}} x_{ij} - \sum_{\{j|(j,i)\in A\}} x_{ij} = \begin{cases} 1 & \text{if i = s} \\ -1 & \text{if i = t} \\ 0 & \text{otherwise} \end{cases}$$

$$0 \le x_{ij},\ \forall (i,j) \in A.$$

For forward path P from s to t the flow vector x with component given by

$$x_{ij} = \begin{cases} 1 & \text{if } (i,j) \text{belong to } P, \\ 0 & \text{otherwise.} \end{cases}$$

Then x is feasible for the problem and the cost of x is equal to the length of P. In a data communication network, a_{ij} may denote the average delay of a packet to cross the communication link (i,j). Figure 2.2 shows an example of shortest path selection in the network.

Routing algorithm calculates the route on basis of the collected information on topology can be classed as central and distributed algorithm. Central algorithm does not need the partial results from other routers to find a path to destination. In order to find a path to destination, every router has the topology information and is able to calculate independent decision of the route to destination. It is also a link state algorithm because all nodes know the state of the link to calculate the path and path is calculated independently on each node. Dijkstra's algorithm is an example

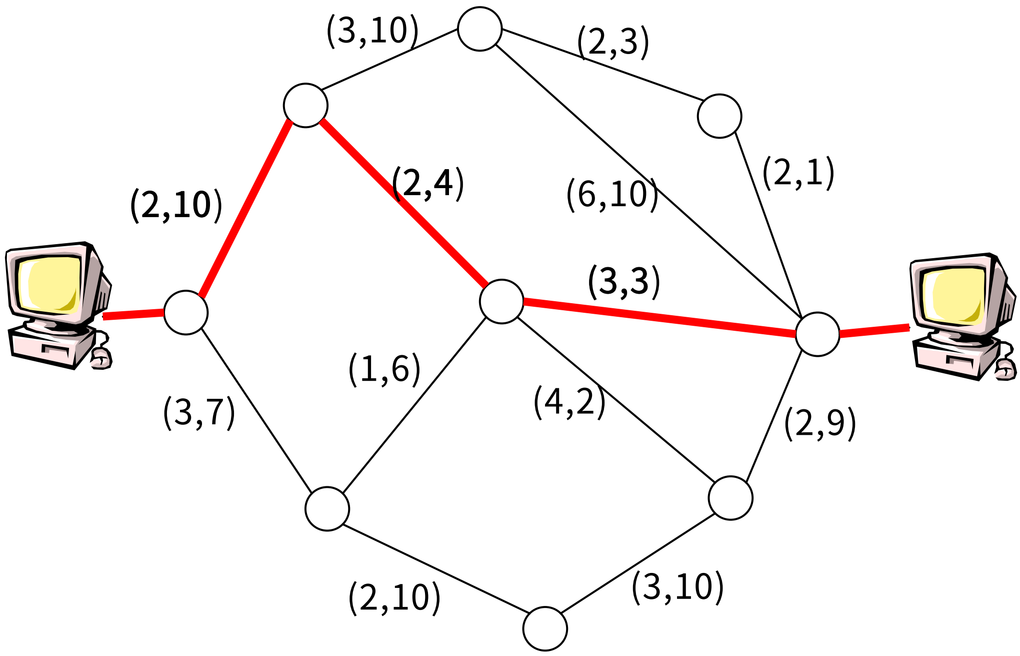


Figure 2.2 Path selecting example using shortest path algorithm

Distributed algorithm needs proper functioning of the same algorithm on the other nodes in order to have consistency of routing information. Partial results from other nodes are used to calculate the path to destination. Bellman-Ford algorithm is an example.

The packet forwarding at each node classifies the routing as source routing and hop-by-hop routing. In source routing, the complete route is calculated one time by the first router, and every router on the path from source to destination will follow the indications of the first route.

Hop-by-hop routing makes the decision of the link used changes for each packet arrival. The hop-by-hop routing is certainly less efficient because much more processing resources are needed, but is also much easier to implement than the source routing. Another advantage as opposed to source routing is the fact that it is considered more secure and it is possible for providers to establish routing policies

[Huitema] discusses all the routing algorithm used in current Internet completely.

2.3 QoS based Routing & Traffic Engineering

The Quality of Service problem has been traditionally solved by using various techniques along a fixed path chosen by normal shortest-path routing protocols. The techniques involved for example queueing algorithms on the routers which do take into account the various QoS classes of the incoming packets.

QoS routing is a class of solutions that are aimed at identifying cost-efficient network paths that have sufficient resources to meet certain performance and reliability constraints. The QoS routing problem can be resumed as "finding a route from source to destination, which respects the specifications of the needed QoS". It is mostly solved by using routing algorithms with multiple parameters. These algorithms choose, on the basis of multiple parameters (metrics) of the network, such as residual bandwidth, propagation delay, etc., and on the requested service, the best possible path. It is called multiple parameters routing because the algorithms are much more complicated than single parameter routing algorithms such as shortest-path routing.

Single parmeter routing look at the network in partial view. Partial knowledge of the network can result in inefficient use of network resources. Network congestion can be cased by lack of network resources or uneven distribution of traffic. Traffic engineering is the process of arranging how traffic flows through the network so that congestion caused by uneven network utilization can be avoided. Constraint-based routing is an important tool for making the traffic engineering process automatic. Constraint-based routing is used to compute routes that are subject to multiple constraints and evolves from QoS routing.

QoS routing is one of the tools available to accommodate flows that expect certain QoS guarantees from the network. It enables the network to identify the sufficient resource path through the network. QoS routing is seen as a missing piece in Internet and an important factor in order serve various qualities required from network. QoS routing has been receiving attention in order to overcome IP limitation to serve various qualities of connection rather then best-effort service. In the past several years, QoS routing has been subject of several studies and proposals.

Under QoS-based routing, paths for flows would be determined based on some knowledge of resource availability in the network, as well as the QoS requirement of flows. The main objectives of QoS-based routing are: First, dynamic determination of feasible paths: QoS-based routing can determine a path, from among possibly many choices, that has a good chance of accommodating the QoS of the given flow. Feasible path selection may be subject to policy constraints, such as path cost, provider selection, etc. Second, optimization of resource usage: A network state-dependent QoS-based routing scheme can aid in the efficient utilization of network resources by improving the total network throughput. Such a routing scheme can be the basis for efficient network engineering. Third, graceful performance degradation: State-dependent routing can compensate for transient inadequacies in network engineering (e.g., during focused overload conditions), giving better throughput and a more graceful performance degradation as compared to a state-insensitive routing scheme.

2.4 Multiprotocol Label Switching (MPLS)

As its name implies, MPLS is essentially a labeling system designed to accommodate multiple protocols. The use of MPLS labels enables routers to avoid the processing overhead checking each packet and performing complex route lookup operations based upon destination IP address. The motivation for MPLS is to use fixed-length label to decide packet handling. A label attached to a packet is used for making forwarding decision, whereas the IP destination address is used in IP networks.

Each MPLS packet has a header. The header contains a 20-bit label, a 3-bit Class of Service(COS) field, a 1-bit label stack indicator, and an 8-bit Time to Live(TTL) field. The MPLS header is encapsulated between the link layer header and the network layer header. An MPLS-capable router, call the label-switched router(LSR), examines only the label in forwarding the packet. The network protocol can be IP or others. This is why it is called *multiprotocol* label switching.

In MPLS, incoming packets are classified and routed at the ingress LSRs of an MPLS-capable domain as shown in Figure 2.3. MPLS headers are then inserted. MPLS set up label-switched paths(LSPs) which is similar to an ATM virtual circuit. The LSP between two routers can be the same as the L3 hop-by-hop route, or the sender LSR can specify an explicit route (ER) for the LSP. The ability to set up ERs is one of the most useful features of MPLS. A forwarding table indexed by labels is constructed as the result of label distribution. Each forwarding table entry specifies how to process packets carrying the indexing label. When an LSR receives a labeled packet, it will use the label as the index to look up the forwarding table. Each Label Switch Router (LSR) along the LSP can make forwarding decisions based solely upon the contents of the labels.

MPLS is also a useful tool for traffic engineering, and also noted for the ability to satisfy the user requirements.

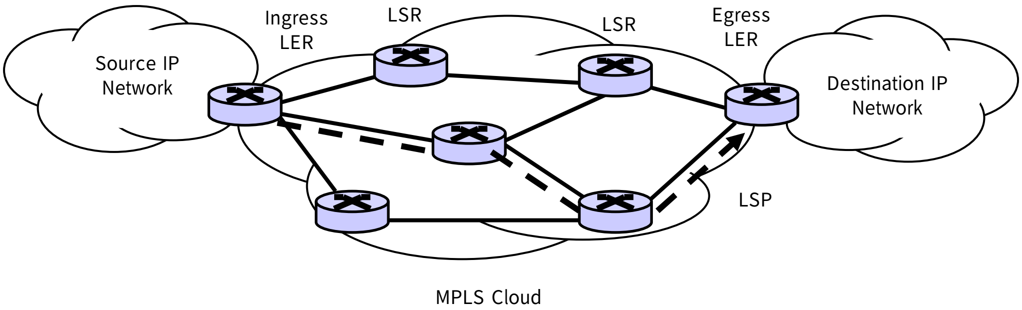


Figure 2.3 MPLS Concepts

2.5 Summary

The basic concept and terminology used in this dissertation is introduced in this chapter. Many studies have been made in order to provide QoS services. QoS routing is one of the issues in serving QoS. QoS routing finds a path to destination satisfying the QoS constraints. In order to provide QoS service along the selected path, resource reservation is required. MPLS ability to reserve and control paths is known to be a useful tool in providing QoS service, perform traffic engineering and boost routing efficiency.

Chapter 3

Related Work

In this chapter, the existing various QoS routing related algorithms and studies are discussed.

3.1 Routing with Multiple Parameters

QoS paths require multiple metric constraints in path selection. QoS routing must follow these required constraints in finding paths. Finding optimal path for the constrained metric is known to be NP-complete[Wang 1996]. The goal is to find feasible path with multiple metric in polynomial time.

Shortest-Widest Path Routing Algorithm [Wang 1996] : [Wang 1996] proposed an hop-by-hop routing suitable for distributed routing. The precedence is on bottleneck bandwidth and then propagation delay. The path search strategy is to find a path with maximum bottleneck bandwidth (a widest path), and when there are more than one widest path, it chooses the one with shortest propagation delay. This path is referred as the shortest-widest path. The delay metric eliminates the loops that causes problem in distributed routing. Two version of distance-vector and link state algorithm is proposed.

Bandwidth-Delay-Constrained Path Routing Algorithm [Wang 1996] : [Wang 1996] also proposed a centralized algorithm suitable for source routing.

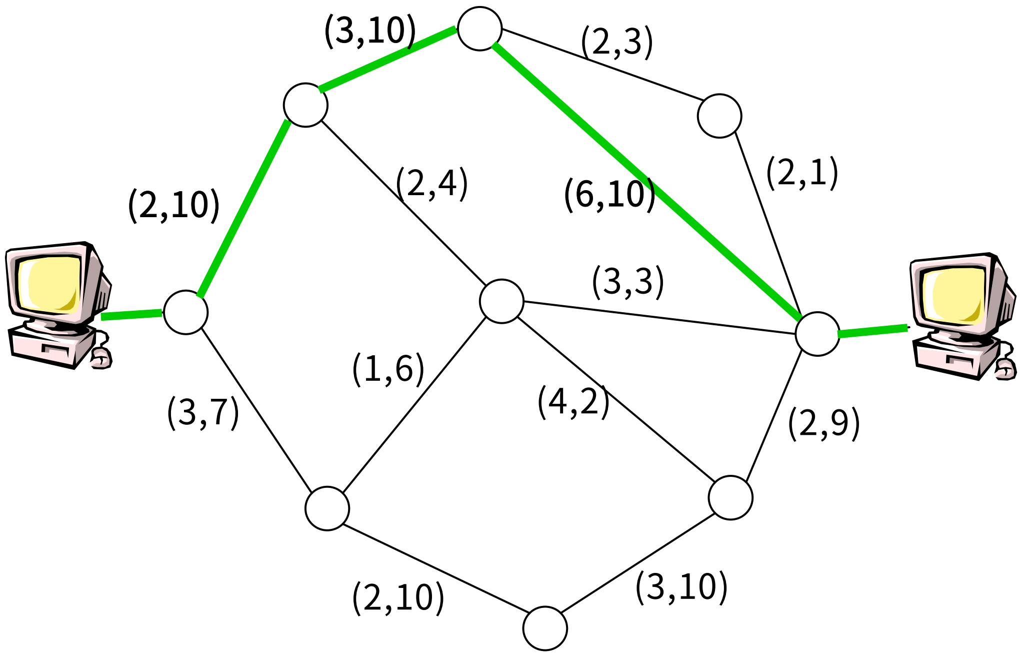


Figure 3.1 Path selecting example using shortest-widest path algorithm

Given bandwidth and delay constraints, the algorithm finds a path that satisfies both constraints, if such path exists. First, the algorithm eliminates all links that do not meet the bandwidth requirement by setting their delay to ∞. Then, the minimum delay path to destination is found using Dijkstra's algorithm. The path is feasible when delay constraint is satisfied. The worst computation complexity is same as Dijkstra's algorithm.

Widest-Shortest Path Routing Algorithm : This algorithm is also an hop-by-hop routing suitable for distributed routing. The difference with shortest-widest path algorithm is that precedence is on propagation delay and then bottleneck bandwidth. The search order is reversed. The path search strategy is to find a path with shortest propagation delay, and when there are more than one widest path, is chooses the one with maximum bottleneck bandwidth.

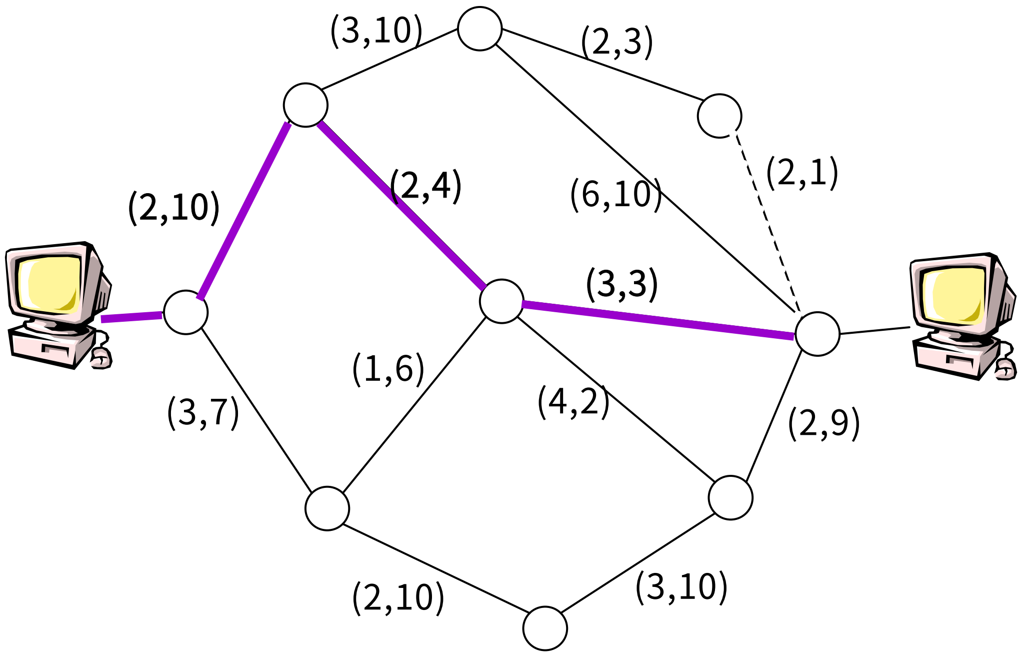


Figure 3.2 Path selecting example using bandwidth-delay-constained path algorithm

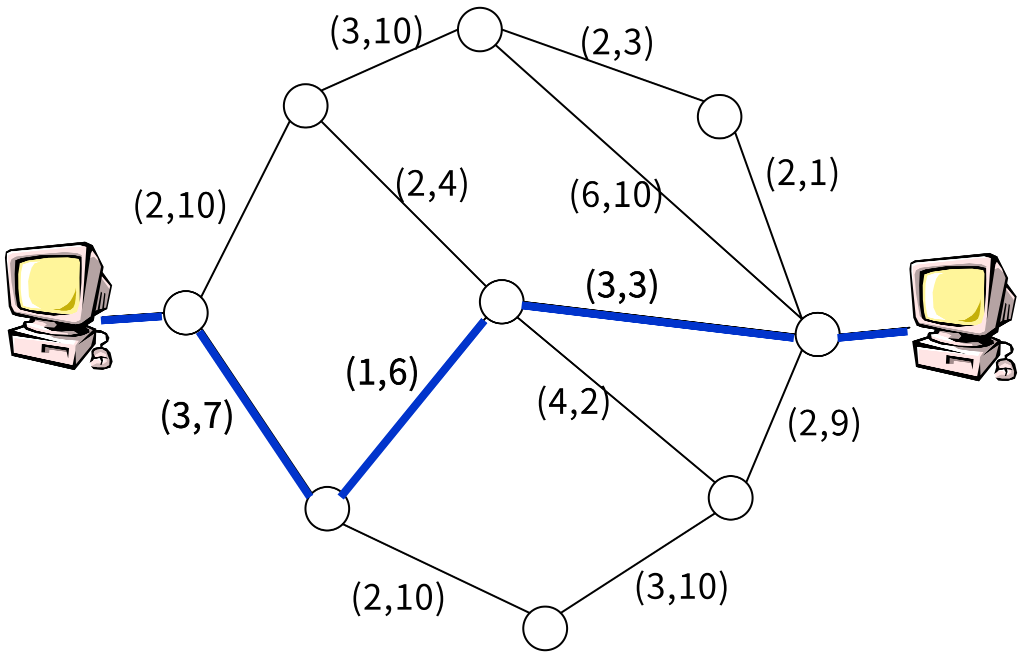


Figure 3.3 Path selecting example using widest-shortest path algorithm

Delay-Constrained Unicast Routing Algorithm [Salama] : [Salama] proposed a source-initiated algorithm that constructs a delay-constrained path connecting source node to destination node. The path is constructed one node at a time from the source to destination. A cost vector and a delay vector are maintained at every node by a distance-vector protocol. The cost (delay) vector contains for every destination the next node on the least-cost (least-delay) path. Any node *v* at the end of the partially-constructed path can select one of only two alternative outgoing links. One link is on the least cost path from *v* to the destination, while the other link is on the least delay path from *v* to the destination. This active node *v* checks if the link on least cost path does not violate the delay constraint to destination. If the link satisfies the constraint, there exist delay-constrained path from v to destination and the link is selected for the path. If not, the link on least delay path is selected.

[Apostolopoulos 1999b] discussed an implementation of QoS routing extension to the OSPF routing protocol. The extensions are enhancing the link state advertisements and the topology database to include network resource information (such as available bandwidth), and using an alternate route computation algorithm to compute routes that take this resource information into account.

[Korkmaz 1999] proposed a randomized heuristic algorithm with polynomial-time complexity to solve NP-complete problem of finding a path subject to multiple additive constraints. The algorithm prunes all the links that cannot be on any feasible paths and then uses a randomized heuristic search which is modified from Beadth-First-Search to find a feasible path, if one exists. [Korkmaz 2002] improved the computation complexity of path selection algorithm through binary search strategy of adequate weight for additive metrics to reduce the search space.

[Juttner] uses the concept of aggregated costs and provides an efficient method to find the optimal multiplier based on Lagrange relaxation. The benefit of this method is that it also gives a lower bound on the theoretical optimal solution along with the result. Instead of using heuristics to manipulate the weights of the different metrics in the aggregated cost function, the Lagrange relaxation method is used to find the optimal multiplier.

3.2 Load Distributing Algorithms

QoS routing has ability to find alternate path when a link is crowded. It uses dynamic network information to avoid network congestion and finds adequate multiple paths between source and destination pairs to balance the load among the network.

[Chen] proposed the Scout algorithm which computes paths within a network using dynamic link metrics. Scout's basic mechanism is message flooding. It is a destination-initiated algorithm which destination initiates a global path computation to itself using dynamic link metrics.

[Vutukury 1999] proposed a fast responsive routing algorithm that computes multiple path between each source-destination pair. The algorithm first computes short distances to destinations and then generalizes the shortest-path tree to shortest-multipaths that satisfies the constraint to obtain a loop-free routing graph for each destination.

[Ergun] proposed routing algorithm that uses cost function that reflects the prices paid in return for delay guarantees. It is to choose a source-destination path and determine a set of per link delay guarantees along this path so as to satisfy the constraint D while minimizing the total cost incurred.

[Fortz] also proposed a cost function that could be used in change of single weight of OSPF to optimally distribute the flow over all paths between source and destination. The cost function is the normalized weight of the arc according to the link load. The weight is to avoid congestion, i.e. overloading of links.

3.3 Routing with Inaccurate Information

QoS Routing requires periodic exchange of network information to acquire accurate QoS path. However, frequent update of information causes network overload and increase of network size causes delay in acquiring accurate information and aggregated network information. Dynamic network state causes network information to be dynamic. These factors cause inaccuracies. Routing with inaccurate information is inevitable.

[Guerin 1997] investigated the impact of inaccuracies in the available network state and metric information on the path selection process. It was focused on connections with QoS requirements, and considered the two cases of bandwidth requirements and end-to-end delay guarantees. The case of connection with only bandwidth requirement receives minimal impact of inaccuracies. However, in the case of connection with end-to-end delay guarantees, inaccuracies had major impact on the complexity of path selection process. The delay guarantee case is studied further in rate-based model and delay-based model. The rate-based model problem is intractable, but delay-based model can find solution through metric quantization in reasonable complexity cost. It is further investigated a number of heuristics based on splitting the end-to-end constraints into local constraints that can be applied to hierarchically interconnected network.

[Nelakuditi] focused on localized QoS routing scheme where the edge routers make routing decisions using only local information. QoS routing require periodic exchange of QoS state information among routers which impose overhead to network and routers. Source node is assumed to have only route-level statistics, such as number of flows blocked as QoS state information. Based on these statistics, adaptive proportional routing attempts to proportionally distribute the load from a source to destination among multiple paths according to their perceived quality (e.g., observed flow blocking probability).

[Hao] investigated the performance of techniques that can reduce QoS routing protocol overhead by topology aggregation. Topology aggregation is done in full mesh which is in port-to-port distances and symmetric star which represent the distance by a single parameter. Observation shows that topology aggregation does not always have a negative impact on routing performance. Star aggregation can reduce the routing information fluctuation and increase stability.

[Orda 2000] turn to precomputation perspective to select a path suitable for QoS requirements and global utilization of network resources by focusing on two major settings of QoS routing. First, the case where the QoS constraint in of the "bottleneck" type, e.g. a bandwidth requirement, and network optimization is sought through hop minimization. The standard Bellman-Ford algorithm offers a straightforward precomputation scheme. Then the general case of setting of "additive" QoS constraints and general link costs. It is focused on ε –optimal approximations that offers improvement over a "standard" approach.

3.4 Hierarchical Routing Algorithms

PNNI (Private Network-Network Interface) [PNNI] is a hierarchical link state source routing protocol used in ATM network. PNNI routing hierarchy starts from lowest hierarchical level. Lowest hierarchical level is grouped in peer groups as shown in Figure 3.4. A lowest level node becomes a logical node and peer group is the set of logical nodes. Logical node in peer group exchange information with other members and each member has same group view. One peer group is abstracted to a logical group node in higher hierarchical level. Each logical node calculates the path to destination within the peer group of each level. As an example, if logical node (A.1.3) wishes to communicate with node (C.2), node (A.1.3) first selects path within peer group (A.1) to path selected to destination in peer group (A.1) level. Peer group A selects path from (A.1) to (A.3) and each logical node selects specific path within the logical node in lower hierarchical logical nodes. If peer group A takes path (A.1)-(A.4)-(A.3), logical node (A.1) knows only the path from (A.1) to (A.3) and path within the node which is the path (A.1.3) to (A.1.1). When path is reached to (A.1.1), then routing is executed in higher level to reach from (A.1) to (A.3). (A.4) knows the path (A.4.5) to (A.4.6) and (A.3) knows (A.3.4) to (A.3.2). Routing process is performed in each level. Therefore, routing in peer group B is executed in peer group B.

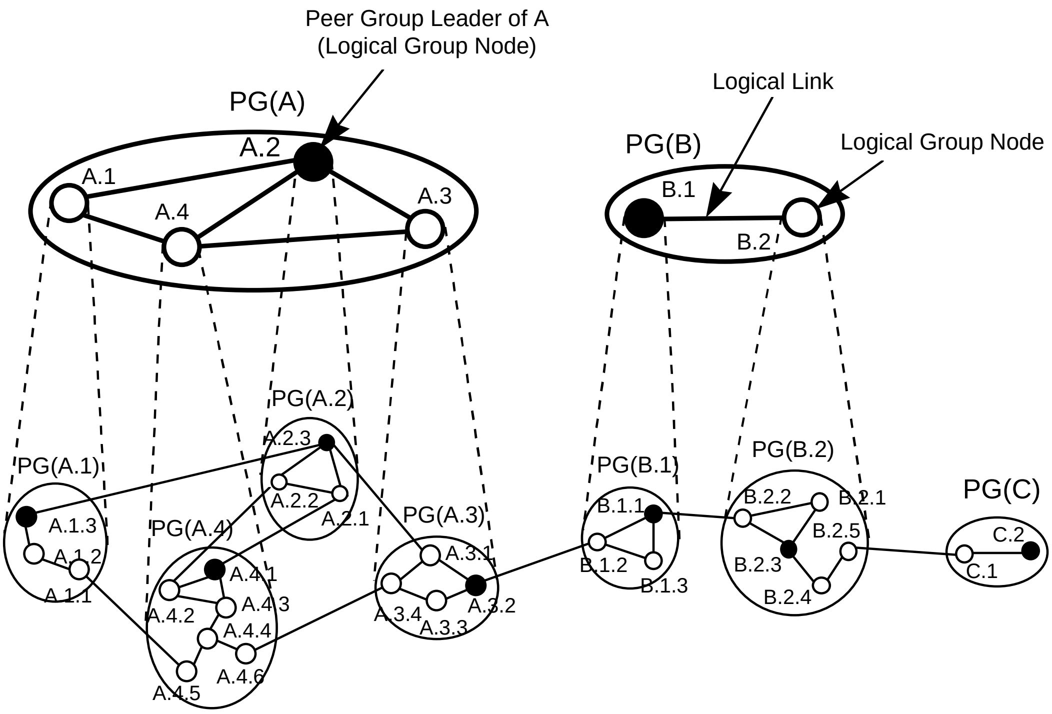


Figure 3.4 PNNI routing example

[Murthy] proposed a hierarchical routing algorithm named Hierarchical Information Path-based Routing(HIPR) algorithm, accommodates an arbitrary number of aggregation levels. HIPR provides a distributed implementation of Dijkstra's shortest path algorithm running over a hierarchical graph organized in areas according to McQuillan's scheme for hierarchical routing. Each router communicates to its hierarchical routing tree in an incremental fashion. Hence, for a hierarchical routing tree, an entire foreign area is simply another node in the tree.

3.5 Summary

The existing QoS routing related studies are reviewed. Routing with multiple metric is known to be NP-complete. Heuristics and weight cost are introduced to find feasible path in polynomial time. Routing with multiple metric also gives the ability to find an alternate multiple paths that satisfy the QoS constraints required for the path. The use of alternate and multiple path balances the load and prevents network congestion. Satisfying the constraints of a path requires accurate network information update. Dynamic network state result in dynamic network parameters and frequent update is required. However, frequent update result in additional network load. Routing with inaccurate information is inevitable. Topology aggregation and hierarchical routing is a solution for scalable large network.

Chapter 4

QoS Routing with Network Dependent Cost

4.1 Problem Definition

When routing in networks, routing metrics are the representation of network. Therefore, while computing paths between nodes, use of a metric will decide the resulting path. Path computation algorithms for a single metric, such as delay and hop-count, are well known and have been widely used in current network. Using a metric more than one can certainly model network more accurately, and will provide a path that supports various QoS requirement in connections.

The MCC problem is formalized in the following by identifying feasible paths considering multiple path metrics.

Definition 1 *Multi-cost-constrained routing problem (MCC)* : Given a directed graph $G(N,A)$, a source node s, a destination node d, cost functions $f_1, f_2, \ldots, f_n$, path constraints $c_1, c_2, \ldots, c_n$; the problem denoted as MCC($G, s, d, f_1, f_2, \ldots, f_n, c_1, c_2, \ldots, c_n$), is to find a path P from s to d such that the cost functions $f_1, f_2, \ldots, f_n$ satisfies path constraints $c_1, c_2, \ldots, c_n$ if such a path exists. When the path constraint c_1 is for an additive metric and c_2 is for a concave metric, the problem indicated as MCC($G, s, d, f_1, f_2, c_1, c_2$) is to find a path that satisfies $f_1(P) \leq c_1$ and $f_2(P) \geq c_2$ if such a path exists.

A path P which satisfies $f_1(P) \leq c_1$ and $f_2(P) \geq c_2$ is called a solution (feasible path) to MCC($G, s, d, f_1, f_2, c_1, c_2$). It is assumed that cost function f_1 is additive and f_2 is concave. [Wang 1999] shows that multiple combination of additive metric and multiplicative metric is NP-complete. However, there is a feasible combination solution for multiple combination of metric which is one additive or multiplicative and a concave metric.

Therefore, problem MCC($G, s, d, f_1, f_2, c_1, c_2$) is NP-hard. Several path selection algorithms related to this combination have been proposed. They include the widest-shortest path algorithm[Guerin 1996], the shortest-widest path algorithm[Wang 1996], and bandwidth-delay-constrained path algorithm[Wang 1996]. Even though the widest-shortest path and the shortest-widest path routing algorithms computes path with more than one metric, resulting path may depend mostly on first metric. The reason is that, during path computation, second metric is used only when first metric could not determine the best path. What is used for first metric will decide most of the performance of the routing algorithm and network efficiency. Therefore, shortest-widest algorithm first focuses on load balancing while widest-shortest algorithm focuses on shortest path.

In previous works, widest-shortest path algorithm has shown better performance in cases when network load is heavy[Ma 1997a], using in routing server[Apostolopoulos 1999a] and topology aggregation[Hao]. Its tendency to use first metric to select path causes the problem for widest-shortest path. If there is no competition between paths to destination and it selects only one path according to the first metric, no other path can be selected. Then, this selected path is the same path as the one calculated with one routing metric which is dijkstra algorithm. The path will not be able to redirect the path to destination when the network cannot accept another path according to first metric, and the path will be fixed whatever the network state may be.

Shortest-widest path algorithm's tendency to focus on wide bandwidth link has the ability to distribute the traffic over the network. The algorithm looks for wider paths and indifferent to links with not much bandwidth left. Therefore, the algorithm spreads the network traffic to wider paths and accepts more traffic into the network links with residual bandwidth. However, the ability to find wider links may cause the algorithm to select longer path and sometimes worst path regarding the distance to destination even though there is a short path with smaller but sufficient bandwidth.

Bandwidth-delay-constrained path algorithm also performs well on call acceptance because the algorithm eliminates links that does not meet the call requirement. However, the algorithm selects shortest path until there is no residual bandwidth left. It uses the link within the limit of the possible to the utmost. The algorithm does not distribute the traffic. It uses only the possible links and move over to other link if the preferred shortest link cannot be used. Therefore, the algorithm has the ability to find alternate path to use, but does not function well at load balancing.

4.2 Widest-Least Cost Routing Algorithm

The widest-least cost routing algorithm is provided to solve MCC($G, s, d, f_1, f_2, c_1, c_2$) problem. Key idea for the proposed algorithm is to combine delay and dynamic bandwidth metric acquired from the network to create cost metric which is calculated from both of these metrics. The algorithm contains two steps:

1. Create a new cost function f_3.
2. Solve MCC($G, s, d, f_1, f_2, c_1, c_2$) using f_3 in polynomial time.

A new cost metric is presented to be used in cost function f_3. It is an approach to define a function and generate a single metric from multiple parameters. Various pieces of information is mixed into a single measure and it is used as the basis for routing decisions. However, it is said that single mixed metric can only be used as an indicator at best as it does not contain sufficient information to assess whether user QoS requirements can be met or not. Mixed metric approach is a tempting heuristic but it can at best be used as an indicator in path selection[Wang 1996]. The cost function f_3 operates as an indicator to select a path P and the selected path will be checked whether if it satisfies the MCC(G, s, d, f_1, f_2, c_1, c_2) problem.

The cost metric used in function f_3 is based on delay metric and influenced by bandwidth metric. The cost metric is basically a delay metric. Weighted value will be added to the delay metric when the link usage increases over specific amount. Therefore, the metric operates as if it is not always used. Until some point during the operation, it is just like delay metric. This metric is applied when network load increases over predefined rate. This rate is the threshold rate when traffic should start to be moved to other links. Consequently, the amount of link usage over threshold rate is calculated as a weight and added to the delay which may reduce the usage of the link.

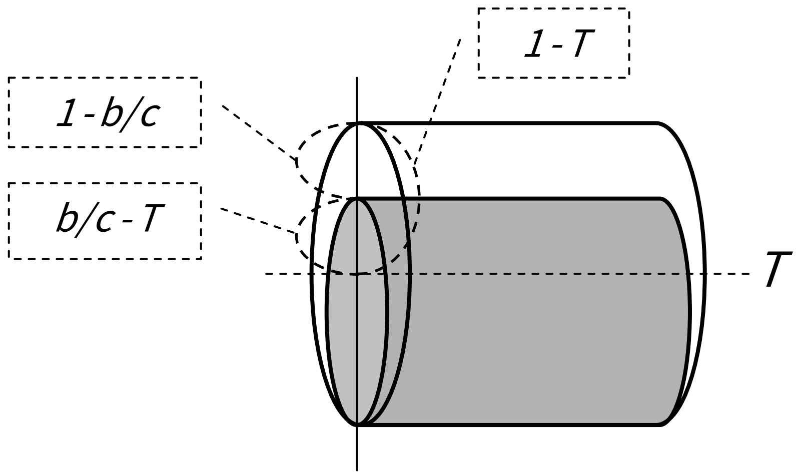


Figure 4.1 Example of a link whose utilization is more than threshold T

Figure 4.1 illustrates an example of a link with the usage of the link exceeding the threshold rate *T*. The amount of link usage above threshold rate is of concern in the study. As shown in the figure, if the link usage is over threshold rate, the usage rate of the link above threshold rate $((1-b/c)/(b/c-T))$ from the residual bandwidth of threshold $(1-T)$ is considered to be weighted to the delay. The amount of link usage exceeding threshold rate increases the weight value. Therefore, the weight function and cost of a link is defined as follows.

Definition 2 Link weight function w(*l*) of a link *l*. For an arc link $l \in A$, weight of a link is defined as,

$$w(l) = \begin{cases} 0, & \text{if } \dfrac{b_l}{c_l} \le T \\ d_l \times \left(\sigma \times \dfrac{\left(\dfrac{b_l}{c_l} - T\right)(1-T)}{1-\dfrac{b_l}{c_l}} \right), & \text{if } \dfrac{b_l}{c_l} > T \end{cases}$$

where d_l is delay of the link *l*, *T* is the threshold rate, σ is the bias value, b_l is current bandwidth of link *l*, c_l is capacity bandwidth of link *l*.

Therefore, cost *c* of a link *l* is,

$$c(l) = d_l + w(l)$$

$$= \begin{cases} d_l, & \text{if } \dfrac{b_l}{c_l} \le T \\ d_l \times \left(1 + \sigma \times \dfrac{\left(\dfrac{b_l}{c_l} - T\right)(1\text{-}\mathrm{T})}{1-\dfrac{b_l}{c_l}} \right), & \text{if } \dfrac{b_l}{c_l} > T \end{cases}$$

Bias value σ is used to control the total cost value. The equation has the form of threshold rate over bandwidth available rate. If σ is large, it will increase the cost. It is used as a control factor of the amount of changes in cost.

Therefore, the value of σ will change the performance of the algorithm. Moreover, σ must be a positive value weighted to the delay metric. If negative value is used, it would cause the weight to be negative and decrease the cost of a link to be smaller than delay. If this case occurs, then link with longer delay could be calculated as shortest delay path and selected to be the shortest path even if the actual shortest path selection is recommended. Therefore, this would bring confusion in shortest path calculation. The proposed algorithm requires that all arc lengths are nonnegative to find the solution. The value use as the bias will be discussed in Section 4.2.2.

The cost is same as delay when network load is under threshold rate, but when network load exceed the threshold rate, w(l) is used to calculate the value of bandwidth dependent cost metric. The delay characteristic of this metric enables the algorithm to find shortest path in general. Moreover, the weighted cost c is an additive metric, so the cost of the path P using function f_3 is calculated as follows.

Definition 3 Given a path P, its cost $f_3(P)$ is,

$$\begin{aligned} f_3(P) &= \sum_{l \in P} c(l) \\ &= \sum_{l \in P} \left(d_l + w(l)\right) \end{aligned}$$

Theorem 1 guarantees that if the path found by weighted function f_3 satisfies the constraint, it is a solution to MCC($G, s, d, f_1, f_2, c_1, c_2$).

Theorem 1 A solution to MCC($G, s, d, f_3, f_2, c_1, c_2$) must also be a solution to MCC($G, s, d, f_1, f_2, c_1, c_2$).

Proof : Let P be a solution to MCC($G, s, d, f_3, f_2, c_1, c_2$). Hence, $f_3(P) \leq c_1$ and $f_2(P) \geq c_2$. In order to prove P is also a solution to MCC($G, s, d, f_1, f_2, c_1, c_2$), $f_1(P) \leq c_1$ must be proved. Since f_3 is weighted function of f_1, weight w along path P is added to f_1.

$$f_3(P) = f_1(P) + w(P) \leq f_1(P)$$

Therefore,

$$f_1(P) \leq f_3(P) \leq c_1$$

$f_1(P) \leq c_1$ and hence theorem holds. □

Theorem 1 shows that solution to MCC($G, s, d, f_3, f_2, c_1, c_2$) using the proposed algorithm is guaranteed to find a solution to MCC($G, s, d, f_1, f_2, c_1, c_2$). However, even when MCC($G, s, d, f_3, f_2, c_1, c_2$) does not have a solution, the algorithm still has a chance to find a good solution.

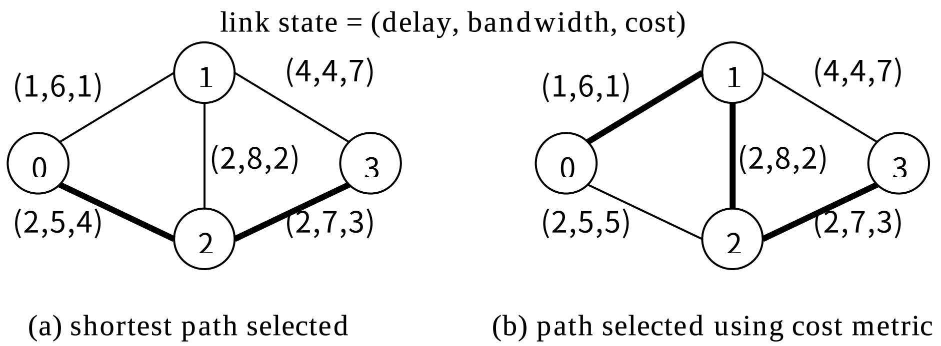


Figure 4.2 Example of path selection when delay constraint is 5 and bandwidth constraint is 5.

Figure 4.2 is an example of a network during operation. As shown in the Figure 4.2 (a) shortest path from node 0 to 3 is drawn is thick line. The selected path satisfies the required solution. However, there can be another path found by selecting least cost path shown in Figure 4.2(b). The selected path P does not solve MCC($G, s, d, f_3, f_2, c_1, c_2$) because it does not satisfy the constraint, but the selected path P satisfies the constraint and gives a solution to MCC($G, s, d, f_1, f_2, c_1, c_2$). The reason is that the cost is the weighted value and the actual delay is smaller than the cost.

4.2.1 Discussion of Threshold Selection

Threshold used in cost function is the point when weight is added to the delay. Therefore, the threshold decides when the weight is added and triggers the traffic to find alternate path. Threshold can be static or dynamic value. If static value is used, adequate value is hard to find. Different topology and various traffic characteristics expect different optimal threshold. Observation of network situation and operation or through great number of simulation may find the relevant threshold. However, there could be a merit for each link can have different value according to the operator's estimation. Each link can have static value and accessed according to the link consumption. Static value of 0.8 may trigger the traffic to consider alternate path when the link is 80% full.

Dynamic threshold can reflect the real-time status. Threshold changes according to the changes of the network can be reflected. However, it may require fast acquirement and computation of information. Therefore, dynamic value is not preferred in large networks like backbone network.

The study finds interest in dynamic load balancing. Therefore, we will use dynamic value in order to balance the network load effectively. Also, the interest in load balancing encourages the use of network utilization value for threshold value. Network utilization is the average of link utilization. By using network utilization value, the weight can be optimized according to network load.

For an arc link $l \in A$, link utilization is,

$$UT_l = \frac{b_l}{c_l}$$

where b_l is current bandwidth of link l, c_l is capacity bandwidth of link l.

Definition 4 Network utilization value is as follows,

$$NUT = \frac{\sum_{l \in A} UT_l}{|A|}$$

where $|A|$ is the number of links in network.

Using network utilization as threshold will change the threshold according to the network load. It will trigger link weight calculation for only the links with higher utilization than average and in turn increase network efficiency. The advantage of using network utilization is that the weight changes according to the traffic amount and distribution. If the traffic of the network is unbalanced, the difference of the network utilization and maximum value of link utilization is large which enlarges the weight value to make the link undesirable. On the contrary, balanced network traffic results in small difference of network utilization and link utilization values. The weight is small and path calculation will be done with almost the actual metric value.

However, procedure to monitor and model network condition of the topology is needed to have perfect value.

4.2.2 Discussion of Bias

In the proposed method, cost calculation will be executed if a link has higher utilization than threshold. There is also another factor in the function. Bias also changes the amount of weight imposed. In this section, the value of the bias to be applied will be discussed.

The new cost of link d_j' would be calculated as,

$$d_j' = d_j \left(1 + \sigma \left(\frac{b}{c} - T \right) \times \frac{1 - T}{1 - \frac{b}{c}} \right)$$

where d_j is the delay to the shortest path to neighbor node *j*.

Next shortest path except the directly connected shortest link to neighbor node *j* is found when the shortest path to the neighbor node is computed after eliminating the directly connected shortest link.

Lemma There is no alternate path to neighbor node *j* if there is no path to destination when the link to the neighbor node *j* is eliminated.

Proof : If there is a path *P* to destination *d*, it is a connected graph to destination. Neighbor node *j* is next hop in path *P* to destination *d*, eliminating the link to *j* also eliminates the path to destination. If the graph is still a connected graph to destination, there is an alternate path to destination. It is a connected graph from the neighbor node and destination, so there is also a path to neighbor node. If the graph is disconnected, the eliminated link is the cut separating the network to the neighbor node and the only link to destination. Therefore, there will be no alternate path to the neighbor node or to the destination node. □

Lemma If there is no alternate path to neighbor node, bias is 0.

Proof : Bias is needed to add weight in order to find alternate path. If there is no alternate path to neighbor node, the bias value is worthless. Therefore, it is set to 0. □

If d_{ij}'is the delay from node *i* to the next shortest path to neighbor node *j* except for the path of d_{ij}, weighted cost w_{ij} for the link $(i,j) \in A$ has to be larger value than d_{ij}' in order to use the alternate path.

$$w_{ij} = d_{ij}\left(1+\sigma\left(\frac{b}{c}-T\right)\times\frac{1-T}{1-\frac{b}{c}}\right) > d_{ij}'$$

If *k* is substituted for weight cost,

$$w_{ij} = (1+k)\times d_{ij} > d_{ij}'.$$

It can be seen in this equation that *k* is the increase percentage required between shortest delay and next shortest delay to neighbor node.

$$k > \frac{d_{ij}'}{d_{ij}} - 1$$

The k is utilized for the bias in cost function. The minimum value of k will be the bias σ of the cost equation.

$$\sigma \times \left(\frac{b}{c} - T\right)\frac{1-T}{1-\frac{b}{c}} = k > \frac{d_{ij}'}{d_{ij}} - 1$$

Minimum value of k is required for the cost to find alternate path, but k is not same for all neighboring nodes. Different type of network topology would result in various differences between shortest path and next shortest path to the neighbor nodes. Therefore, to move the traffic to the alternate path, different cost value will be needed for different network topology. Choosing σ to be too large would result in the calculated cost to be much higher than required, and small σ would cause little distribution of traffic. The averaged value of k is used, which is the difference average of shortest and next shortest delay to the neighbor node as topology characteristic. The average of k is defined as topology characteristic value.

Definition 5 Let a node set N_b as a set of nodes that have alternate path to neighbor nodes. Topology characteristic K is

$$K = \frac{\sum_{i \in N_b} \frac{\sum (d_{ij}' - d_{ij})}{|N_i|}}{|N_b|}$$

where $|N_b|$ is the number of nodes that have neighbor nodes and $|N_i|$ is the number of neighbor nodes for node i.

Only the nodes that have alternate path to neighbor nodes participate in topology characteristic calculation and nodes that do not have neighbor nodes are excluded.

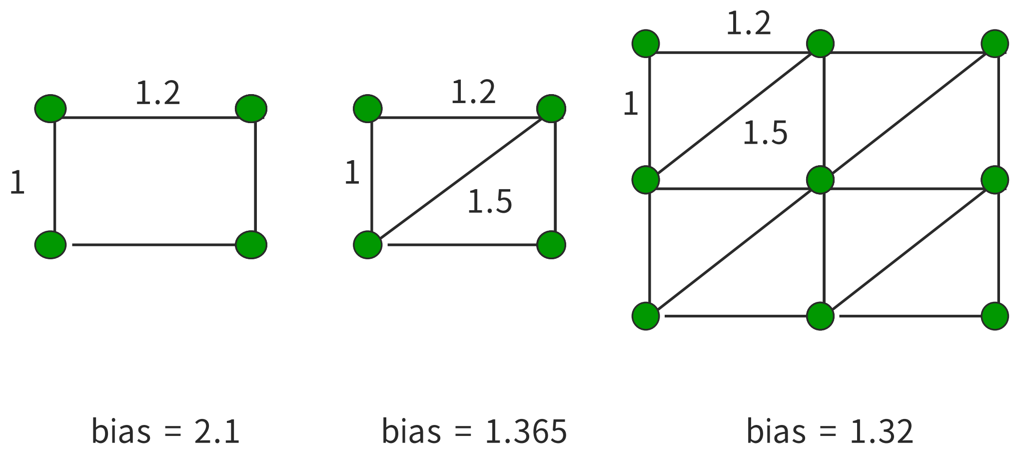


Figure 4.3 Examples of topology characteristic value for different type of topologies

Figure 4.3 shows some example network topology and calculated topology characteristic for the network. The resulting value is different according to the topology.

4.2.3 Algorithm for Path Selection

The proposed algorithm will be presented. Consider a directed graph $G = (N,A)$ with nodes numbered 1, …, N. Each arc $(i,j) \in A$ has a cost or "length" a_{ij} , available bandwidth b_{ij}, and bandwidth capacity c_{ij} associated with it. Given any directed forward path $p=(i, j, k, \ldots, l, m)$, the width of the path width(p) is defined as the bottleneck bandwidth of the path, i.e., width(p) = min$\{b_{ij}, b_{jk}, \ldots, b_{lm}\}$, and the length of the path length(p) is defined as the sum of arc length, i.e., length(p) = $a_{ij} + a_{jk} + \ldots + a_{lm}$. The path is said to be shortest if it has minimum length over all forward paths with the same origin and destination nodes. For some path p_i from *1* to i, d_i can be interpreted for all i as the length of path p_i, i.e., d_i = length(p_i). Let V as the candidate list and B_i be the width along the path *p*. The proposed algorithm can be produced by adding bandwidth checking when there are multiple equal cost paths.

The proposed widest-least cost algorithm is presented in Figure 4.4 to solve MCC($G, s, d, f_1, f_2, c_1, c_2$). The algorithm finds the shortest path in order of increasing path length. Path is selected starting from the initial node 1 as shown in the algorithm. The shortest of the shortest paths to node 1 must be the single-arc path from the closest neighbor of node 1. The node checks the outgoing links to candidate list *V* and selects the shortest one and removes from candidate list. Cost of the link from the selected node to the neighbor node is calculated. If link utilization is higher than network utilization, cost calculating equation is used during the link cost calculation. The neighbor node with the shortest distance is added to the candidate list. This is the process of selecting shortest path link to next shortest path link. It repeats until the cost to all the nodes in the network is calculated.

1) Initially, $V = \{1\}$ $d_1 = 0$, $d_i = \infty$, $B_i = \infty$, $\forall_i \neq 1$.

2) The method proceeds in iterations and terminated when V is empty.

Iteration of Cost Calculated Path Selection

3) Remove from the candidate list V a node i such that

$$d_i = \min_{j \in V} d_j$$

4) For each outgoing arc $(i, j) \in A$,

a) if $\frac{b_{ij}}{c_{ij}} > T$,

$$a_{ij} := d_{ij} \left(1 + \sigma \left(\frac{b_{ij}}{c_{ij}} - T \right) \times \frac{1 - T}{1 - \frac{b_{ij}}{c_{ij}}} \right)$$

b) if $d_j > d_i + a_{ij}$, set

$$d_j := d_i + a_{ij}$$

$$B_j = \min\{B_i, b_{ij}\}$$

c) else if $d_j = d_i + a_{ij}$,

if $B_j < \min\{B_i, b_{ij}\}$

$$d_j := d_i + a_{ij}$$

$$B_j = \min\{B_i, b_{ij}\}$$

5) add j to V if it does not already belong to V.

goto 2)

Figure 4.4 The Widest-Least Cost Algorithm

The proposed widest-least cost algorithm works like widest-shortest path algorithm when network load is under threshold rate. When network load surpasses threshold rate, the widest-least cost algorithm starts to distribute traffic to other links. Therefore, widest-shortest path algorithm can be viewed as a case of widest-least cost algorithm.

The computation complexity of the proposed algorithm is similar to that of dijkstra which is O(nlogn+l) [Bertsekas 1998]. As shown in above iteration, the proposed algorithm has additional computation after the link is selected. Therefore, two more computation must be done for each n links. One way to reduce this computation is to have all the nodes compute the cost value and distribute the cost value rather than the delay value. It is a distributed routing approach which path selection is executed with one weighted metric. It could be an effective method in current Internet. Weighted cost can balance the network load. However, how to define the bandwidth where there is no bandwidth reservation is the issue to be solved.

It is now illustrated how the proposed algorithm selects the alternate path using simple example. Consider the topology shown in Figure 4.5. Each link has a delay value of 1 and suppose node 1 is sending traffic to node 2. There are two possible paths to node 2, which is 1→2 and 1→3→2. However, most of the shortest path related algorithms will select the directly connected path 1→2. This selected path will be used until the link is full. Furthermore, they will select the path even if the link is congested.

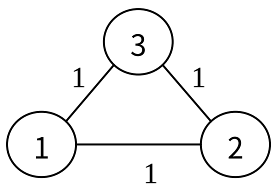


Figure 4.5 Illustration of path calculation

In the proposed algorithm, at start, the directly connected path 1→2 is selected when the link is not so much heavy in traffic. However, if the link is used up to the utilization 66%, the link cost is calculated into 2.01 with topology characteristic bias 1. This resulting cost is a larger value than the alternate path using node 3, which is 1→3→2. Therefore, next call would select the path through node 3. Thus, the proposed algorithm selects less utilized alternate path.

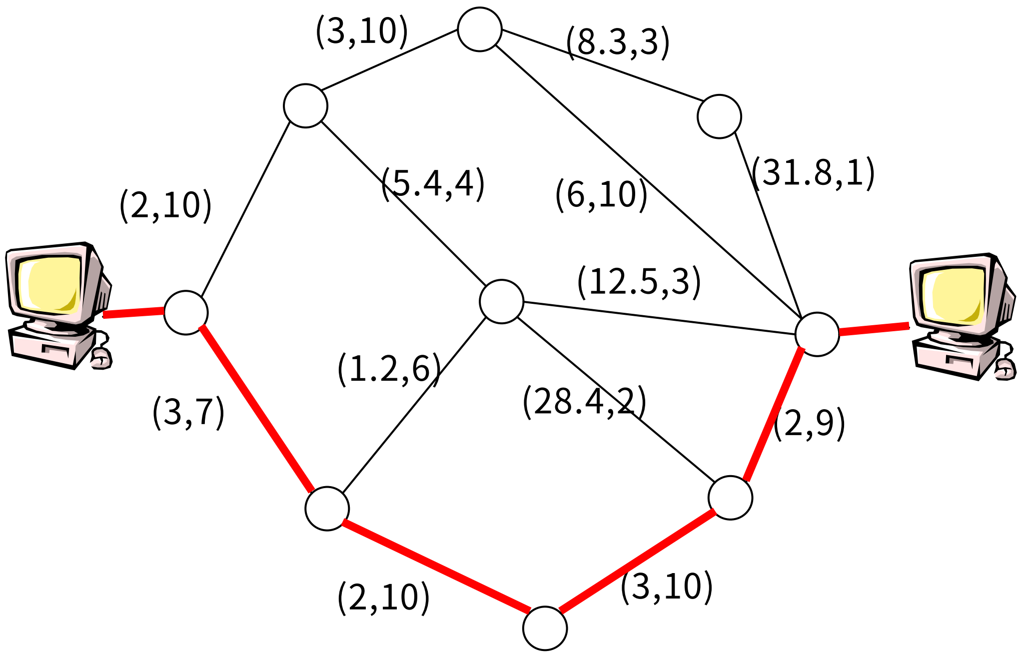


Figure 4.6 Result of a path selection using widest least cost algorithm applied to the network in Figure 3.1

Figure 4.6 shows an example of resulting changed cost of the links in example network shown in Figure 3.1. The cost result from network utilization value of 3.46 and bias 4.1. Selected path has end-to-end delay of 10ms.

4.3 Constrained widest-least cost routing algorithm

The study concern widest-least cost routing algorithm satisfying MCC. In order to satisfy the constraint strictly, the algorithm can be expanded to constrained widest-least cost routing algorithm. As shown in [Wang 1996] bandwidth-delay-constrained algorithm finds the path constrained to metric than shortest path first or widest-shortest path algorithm. Shortest path algorithm or widest-shortest algorithm may not return the resulting path that satisfies both metric. However, bandwidth-delay-constrained algorithm returns only the path that satisfies the constraint.

The method in bandwidth-delay-constrained algorithm is applied to the proposed algorithm to serve with strictly constrained path. The following algorithm finds a path between node 1 and m that has a bandwidth no less than B and a delay no more than D, if such a path exists. c_j is the delay cost from node 1 to node j. and d_j is the propagation delay.

Step 1) initializes values and removes the link that does not satisfy the required bandwidth from the topology and step 2)~5) applies widest-least cost algorithm to find the minimum cost path in the reduced topology. The reduced topology provides with the guarantee of bandwidth and widest-least cost algorithm performs load balancing. Bandwidth-delay-constrained algorithm terminates during the delay calculation if $d_j > D$, but the proposed algorithm uses cost to select path, therefore the algorithm continues until the destination is found.

However, the load balancing function of widest-least cost algorithm may reduce the necessity of the constrained network topology. Load balancing function of widest-least cost algorithm may spread the network traffic and when links starts to be eliminated from the topology, most of other links may also be crowded.

1) Initially, $V = \{1\}$ $d_1 = 0$, $d_i = \infty$, $c_1 = 0$, $c_i = \infty$ $B_i = \infty$, $\forall_i \neq 1$, $d_{ij} = \infty$ if b_{ij}<B.

2) The method proceeds in iterations and terminated when V is empty.

3) Remove from the candidate list V a node i such that

$$c_i = \min_{j \in V} c_j$$

4) For each outgoing arc $(i, j) \in A$,

a) if $\frac{b_{ij}}{c_{ij}} > T$,

$$a_{ij} := d_{ij}\left(1 + \sigma\left(\frac{b_{ij}}{c_{ij}} - T\right) \times \frac{1 - T}{1 - \frac{b_{ij}}{c_{ij}}}\right)$$

b) if $c_j > c_i + a_{ij}$, set

$$d_j := d_i + d_{ij}$$

$$c_j := c_i + a_{ij}$$

$$B_j = \min\{B_i, b_{ij}\}$$

c) else if $c_j = c_i + a_{ij}$,

if $B_j < \min\{B_i, b_{ij}\}$

$$c_j := c_i + a_{ij}$$

$$d_j := d_i + d_{ij}$$

$$B_j = \min\{B_i, b_{ij}\}$$

5) add j to V if it does not already belong to V.

if V contains node m, and $d_j < D$ a path is found and the algorithm teminates

goto 2)

Figure 4.7 The Constrained-Widest-Least Cost Algorithm

Constrained widest-least cost algorithm may not be worth using when network load is light. It might be useful when network load is high, and constrained path is needed. Moreover, it needs to have precise information of the network and

4.4 Summary

The widest-least cost algorithm if proposed for the solution of multi-cost-constrained routing problem. It is a simple method using a new cost function. The cost function increases weight when the link utilization is higher than threshold. The network utilization value is used as threshold in order to balance the network traffic. Also, bias value is determined. The widest-least cost algorithm uses this new cost function to find feasible alternate path when network load is crowded than other links. Constrained version is also shown.

Chapter 5

Routing QoS and Best-Effort services in MPLS

Although widest-least cost algorithm could have static value of threshold and bias given to each router by an operator, the necessity of up-to-date global information in order to cope with the dynamic network state requires the algorithm to be implemented in client/server environment. There are two potential method of client/server architecture in order to apply widest-least cost routing algorithm. The method of applying server based routing architecture is discussed first. Next, QoS routing architecture adequate to widest-least cost algorithm is proposed.

5.1 Server based QoS Routing

Server based routing is simple method to construct a client/server architecture. Introducing QoS routing requires important and potentially costly changes in the network infrastructure. Every network element need changes in order to cope with the new routing algorithm. Moreover, if some routers within the network do not follow the new algorithm may cause problem in finding path like path loops. Performing most of the tasks of QoS routing in a server, allows the other routers to remain simple and to a large extent unaware of the introduction of QoS capabilities. In particular, although some level of QoS awareness is still required from routers, its extent is much more limited and in some cases can be provided with minimal modifications, simply by extensions to the resource reservation protocol.

Any QoS routing architecture involves two major functional components: a) distribution of network state so that route computation is aware of the resource availability situation in the network, and b) path selection.

Distributed routing algorithms such as link state routing protocol requires every node in the network to maintain a complete and up-to-date network state topology database. The accuracy of the database decides the efficiency of the routing algorithm. However, maintaining the latest up-to-date information imposes network and routers with additional traffic load. When route computations are performed at a QoS route server, there is no need for other routers in the network to maintain network state. The fact that network state is located only at the server can simplify its maintenance.

Potential method for maintaining network state at the QoS route server are flooding and directed update[Apostolopoulos 1999a]. Flooding is the conventional method for maintaining link state topology databases. It is to maintain network state at each node and as a result is not well suited for a route server based architecture. Directed update is to simply forward the information directly to the server. This is clearly smaller number of messages and lower processing overhead in the network.

As discussed in [Kim 2001], path caching reduces path computation overhead and enhances the routing performance in server based QoS routing. QoS paths are cached and reused for routing subsequent requests. Each cache entry contains information about the path as well as its bottleneck capacity which is not necessarily up-to-date. A cached path is used to route a request only if its bottleneck capacity as recorded in the cache entry is larger than the requested amount of bandwidth.

The main issue when implementing a server based routing architecture is the communication between client routers and the server. In particular, message exchanges are needed for the route request/reply operation, sending link state updates to the server and also cache update operation. The COPS protocol[RFC 2748] has been developed for the client/server interactions that are necessary in a policy aware environment. This protocol can support reliable message exchange and it is easily extended by adding new message types.

As discussed in [Apostolopoulos 1999a], a route server approach to QoS routing is feasible in terms of the processing load that the server has to handle. In order to reduce the cost of client-server communication, the route server is located in (or close to) middle of the network.

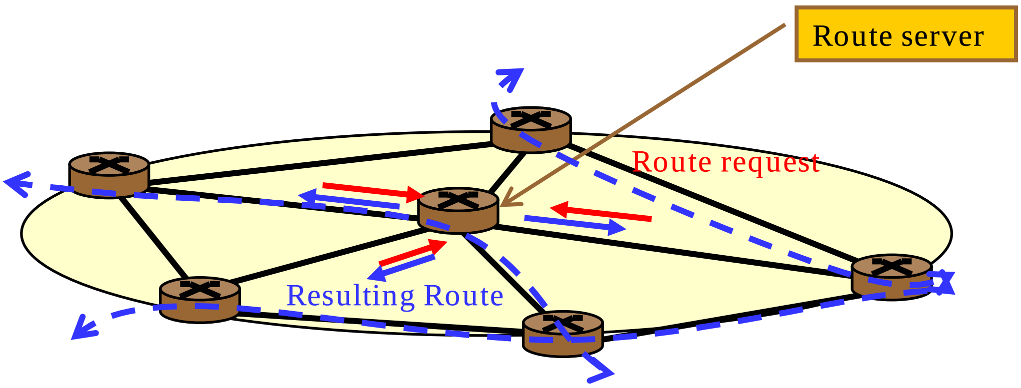


Figure 5.1 QoS routing with route server

Figure 5.1 shows the basic architecture of a server based routing network. As shown in Figure 5.1, the routing domain consists of a route server and clients. Route server maintains the network information database and routing table database and routing cache. Client router maintains shortest path routing table to other routers.

If a new request arrives at a client, the client request for a path query to route server if the request ask for QoS path. On receiving the query message, route server computes a new path using the network information database or selects an appropriate QoS from routing cache. Route server updates the network information database to reflect the resource assignments and replies to the client with the QoS path. If there is not sufficient path, route server replies with path reject message. QoS path can be set up from source route to destination or route server can send path setup message to all routers along the path. Client sends a path terminate message when the use of QoS is finished and route server updates the network information database with restoration of network resources.

5.2 Proposed Routing System

Now the routing architecture favorable for widest-least cost algorithm is introduced. In contrast to server based routing architecture, less load is laid to the server which is called manger.

In order to have up-to-date global network information, a client/server architecture is constructed. Figure 5.2 shows the basic components of the proposed routing architecture. Routing domain consists of a routing manager and client routers. Routing manager maintains network topology database. Network topology database stores the topology information of the network and link status information for each link in the network. The role of routing manager is to gather network information from routers, calculate network information such as network utilization, calculate neighbor node distance difference, and distribute network information to the routers. Client router needs additional topology information database besides current topology information database and routing table database in order to calculate and store link weight.

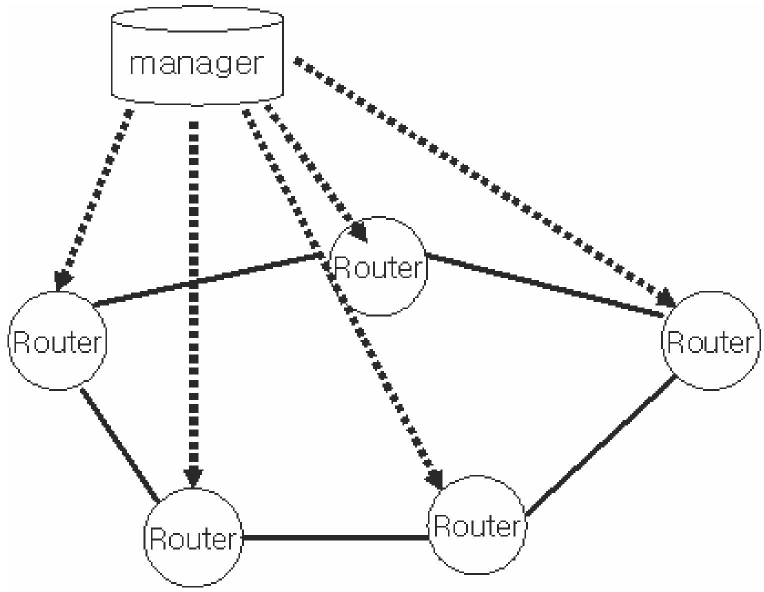


Figure 5.2 Routing structure.

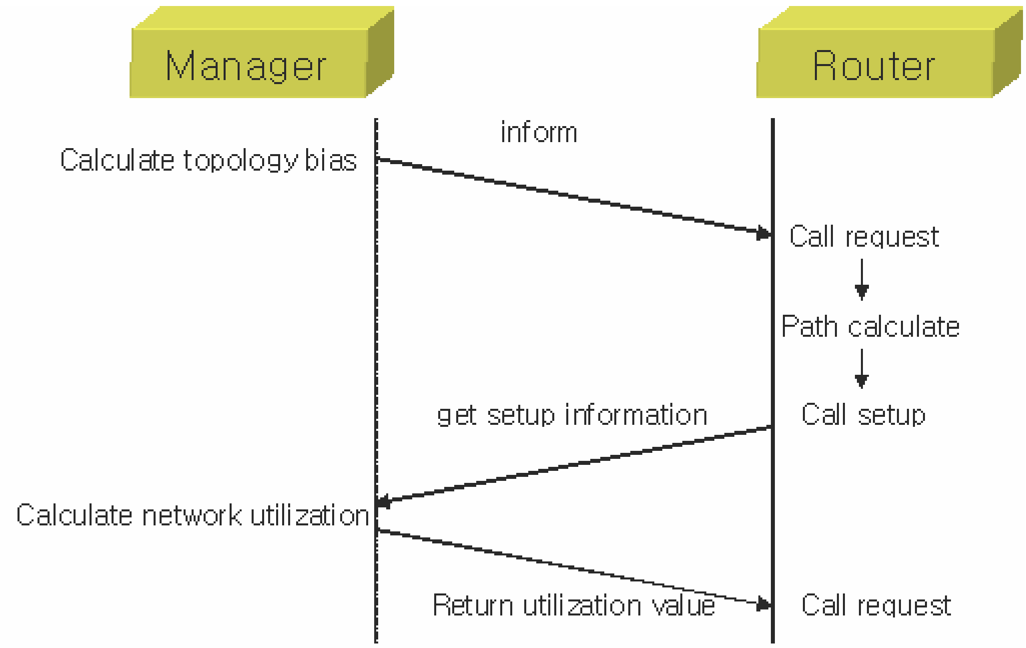


Figure 5.3 Call setup process scenario

The basic operation between routing manager and client router is shown in Figure 5.2. Before setting up any calls, routing manager first gathers topology information from routers. Router sends topology information to routing manager when it starts to operate and when topology change occurs. Routing manager computes the topology characteristic value used as bias in cost equation by calculating neighbor node distance differences for each node and distributes the information to all routers in the network. This value does not change until there is a change in topology.

If a new request arrives at a client router, it calculates the path to destination with cost equation if needed and tries to set up a path. If the path setup is successful, router informs the routing manager with the call request information. Routing manager updates the network topology database to reflect the resource assignment and calculate network utilization value. This value is distributed to all routers to use in next path calculation. Router also sends routing manager with terminated call information when call is finished in order to have routing manager update the network topology database. Routing manager again calculates network utilization value and distributes to all routers.

5.3 Routing Process in Router

In a sentence, constraint-based routing is used to compute routes that are subject to multiple constraints. Constraint-based routing evolves from QoS routing. Constraint-based routing enables a demand driven, resource reservation aware, routing paradigm to co-exist with current topology driven hop by hop Internet interior gateway protocols[RFC 2702]. Constraint-based routing assists in performance optimization of operational networks by automatically finding feasible paths that satisfy a set of constraints for traffic trunks.

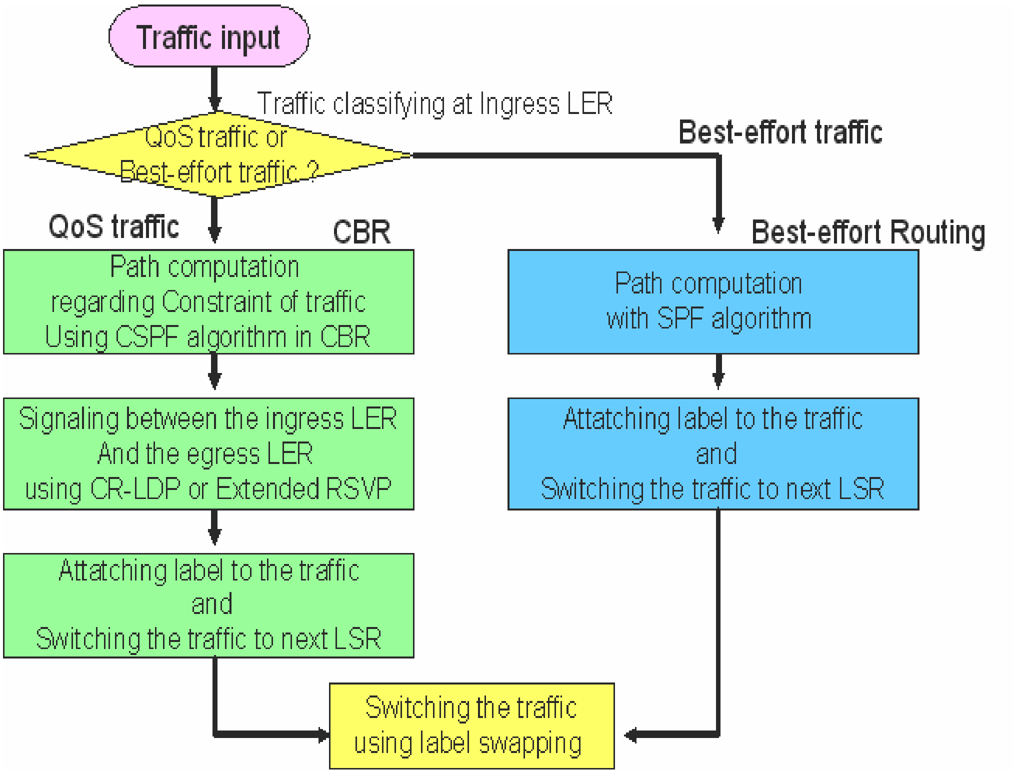


Figure 5.4 Constraint based Routing in MPLS

Figure 5.4 shows the procedure of routing when constraint based routing is used in MPLS network. It is a useful method for supporting QoS and best effort routing.

Constraint based routing uses bandwidth-delay-constrained algorithm to find a QoS path. First prune resources that do not satisfy the requirements of the traffic trunk attributes. Next, run a shortest path algorithm on the residual graph. The QoS path selecting algorithm is extended to use widest-least cost routing algorithm.

If a new call enters the ingress LER, the router classifies the traffic according to the request of constraint. If the traffic requires constraint, the router computes a path to the requested destination that satisfies the constraint. When a path is found, the router reserves network resources to destination by setting up a FEC by explicit routing provided by MPLS. Following traffic is switched to next LSR along the FEC according to the label. If the traffic does not require any constraint, it is a best-effort traffic that will be switched to next LSR according to shortest path to destination.

[Kim] discusses the method to route best-effort path to links that do not have much reserved bandwidth to QoS traffic. If the network can be balanced with QoS path effectively, there is much more chance for the best-effort traffic to reach the destination without being lost.

5.4 Routing in Uneven Bandwidth Network

Widest-least cost routing algorithm utilizes rate values in network state to impose weight cost. For a network with links having uneven size of bandwidth may cause unfairness in balancing network. If the bandwidth difference is small, the difference may be neglected, but if the difference is large such as 100Mbps and 10Mbps it could cause a problem.

For example, consider two links with bandwidth of 100Mbps and 10Mbps each and 100Mbps having shorter path. After some time, if the link with 100Mbps has used up 70% of its bandwidth, then 30Mbps bandwidth is left. This amount is larger than the link with 10Mbps. 100Mbps link having utilization value of 0.7 is larger than the network utilization 0.35. This causes the link to take weight even if its residual bandwidth is larger than 10Mbps link. 10Mbps link may be selected for next path. Of course the small bandwidth link utilization increases faster than large bandwidth link. Then large bandwidth link cost decreases. However, fairness is still required.

To achieve fairness in uneven bandwidth capacity link network, another threshold value *T_base* is suggested. *T_base* is the boundary value when widest-least cost algorithm is used. The minimum bandwidth link value in the network or adequate bandwidth to consider the link residual bandwidth equal is selected for *T_base*. Cost weight is not imposed until the residual bandwidth is smaller than *T_base*. Applying cost equation to links that has less residual bandwidth than *T_base* and calculating network utilization with the bandwidth amount that intrudes *T_base* bound enables to view the network to have same bandwidth when using cost equation. Now, it is same as routing in network with same amount of bandwidth.

5.5 Summary

The method and architecture to implement widest-least cost routing algorithm is discussed. The method of routing with route server is shown first. Then, the method using the proposed architecture of routing manager and router is observed. Using the routing architecture, routing of QoS traffic and best effort is studied.

Chapter 6

Performance Evaluation

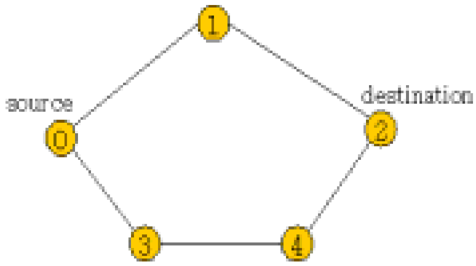
In this chapter, extensive simulation result is presented to evaluate the proposed routing algorithm and compare the performance with other existing routing algorithms. The purpose of the experiments is to determine that the proposed algorithm has better performance over other algorithms on network efficiency and load balancing which increases call acceptance in network.

6.1 Simulation Design

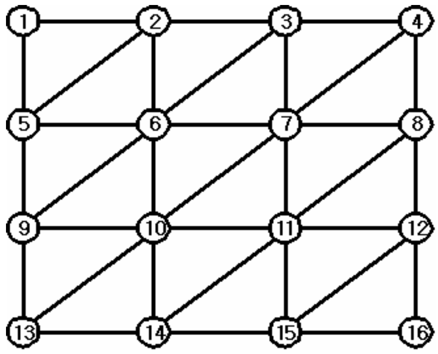
Performance of the algorithm is compared with existing algorithms in various network models through simulation coded in C. The simulation model is described first. Figure 6.1 shows the topology used in the study.

Figure 6.1(a) is a conceptual model topology used to show the basic operation and characteristics of the proposed algorithm. The topologies in Figure 6.1(b), Figure 6.1(c), Figure 6.1(d), and Figure 6.1(e) are used for performance evaluation. Figure 6.1(b) is a grid type topology with many alternate paths possible. Bandwidth is 10Mbps for all links. Link delay values are 10ms for horizontal and vertical links, and 15ms for diagonal links. Therefore, topology characteristic bias value σ is 12.4. Figure 6.1(c) is an octagonal type topology similar to meshed ring. Bandwidth is 10Mbps for all links. Link delay values are 10ms for neighbor nodes along the ring and 15ms for diagonal links. Topology characteristic bias value σ is 20. Topology used in Figure 6.1(d) is a tree type topology that does not have much alternate paths. Bandwidth is 10Mbps for all links. Link delay values are 10ms in branches, 12ms for horizontal links and 5ms for vertical links. Topology characteristic bias value σ is 17. Topology used in Figure 6.1(e) is a backbone network modified from ANSNET used in [Korkmaz 1999]. Bandwidth is 10Mbps for all links and link delay values are given proportional to its length. Topology characteristic bias value is 31.76

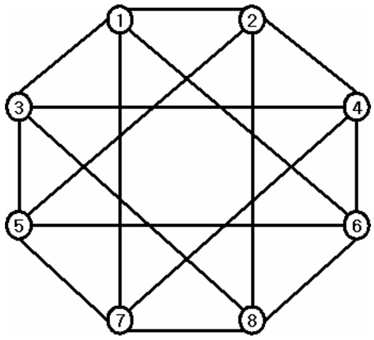
In order to generate a new call connection, simulation program selects source and destination nodes randomly from network. After call has been generated, feasible path for each call is searched using routing algorithms which performance is to be checked. If the algorithm finds a path for connection, each call tries to set up path along the returned path. It is considered a rejected call when a routing algorithm can not find a feasible path. While during a call setup, call will be rejected when one of the links along the path from source to destination does not fulfill the requested bandwidth or end-to-end delay exceeds requested value. Algorithms used for performance comparison are dijkstra algorithm, shortest-widest algorithm, widest-shortest algorithm, bandwidth-delay-constrained algorithm and proposed widest-least cost algorithm.



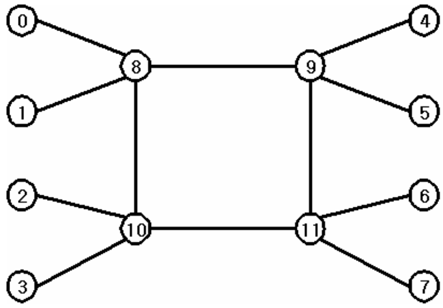
(a) Topology 1 : Basic topology



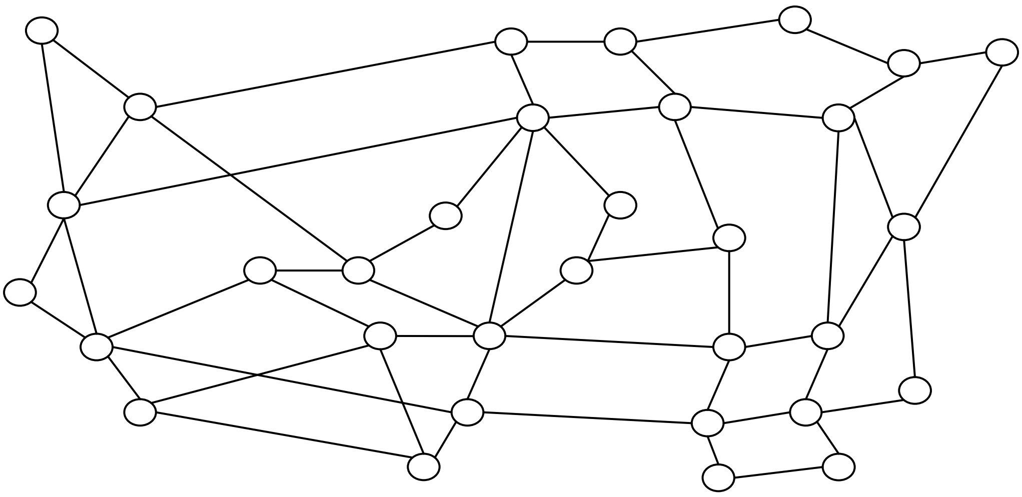
(b) Topology 2 : Grid type topology



(c) Topology 3 : Octagonal type topology



(d) Topology 4 : Tree type topology



(e) Topology 5 : Backbone type topology

Figure 6.1 Topologies used in the simulation

6.2 Traffic Load

It is assumed that connection calls arrive in the network following poisson distribution. The requested bandwidth is uniformly distributed from 64kbps to 512kbps. Most simulation studies of real-time traffic use an exponential call holding time distribution. The average holding time used in the simulation is 30 minutes. Traffic load is controlled by call generation rate. Traffic is generated according to poisson distribution which parameter is the average number of calls generated per minute.

The traffic loads are considered to be either evenly or unevenly distributed. For evenly distributed loads, a new session selects a pair of source and destination randomly. For unevenly distributed loads, the probability for a specific pair of source and destination node is increased to be selected more often than other pair.

6.3 Performance Metrics

In order to compare the performance of the proposed algorithm with other algorithms, four performance measures have been used: *call rejection rate* (CRR) and *bandwidth efficiency* (BWE), *average delay of paths* (AD) and *average hops per call*(AH).

Call rejection rate (CRR) is the main performance metric that is of interest. It is the percentage of the calls rejected among the all requested calls during the simulation :

$$\text{Call Rejection Rate} = \frac{n\_rej}{n\_req}$$

where *n_rej* is the number of calls rejected, *n_req* is the number of calls requested for connection.

Bandwidth efficiency (BWE) is the percent of current bandwidth usage of link capacity in certain time of operation :

$$\text{Bandwidth Efficiency} = \frac{b_l}{c_l}$$

Average bandwidth efficiency (ABWE) is the average value of *bandwidth efficiency* for a link during total time of operation. *Maximum bandwidth efficiency* is the maximum value among the *bandwidth efficiency* of the links in the network.

Average delay of paths (AD) is the average of path delay that has not been rejected :

$$\text{Average delay of Paths} = \frac{\sum d(P)}{n\_suc}$$

where *n_suc* is the number of calls that succeeded in path setup.

The goal of path selection is to reduce *call rejection rate* and choose feasible path that achieves efficient utilization of network resources.

6.4 Performance Results

First of all, lets consider two cases from Figure 6.1(a). This simple simulation will show the operation and the function of the proposed algorithm. First case is to set the link delay for the upper path and lower path to have same value between source and destination. Links of upper paths have delay of 1.5ms each and links of lower paths have delay of 1ms each. Results in Table 6.1 show that traffic is distributed equally on the network. ABWE for each link show that bandwidth usage of every link is similar. Maximum value and minimum value of ABWE have little difference between links. Each link in the topology is used equally. Also, no calls have been rejected.

Table 6.1 Simulation result of equal path length for topology 1.

	dijkstra	widest-shortest	widest-least cost
CRR	0.0	0.0	0.0
Max ABWE	53%	46%	47%
Min ABWE	52%	46%	47%

This situation changes in second case when the delay for upper path is shorter than lower path. Let the delay for each link have same value of 1. Then upper path will have end-to-end delay of 2, and lower path will have end-to-end delay of 3. Results in Table 6.2 show that for dijkstra and widest-shortest algorithm, traffic is not distributed well. All the traffic is located at one side. In this case it is upper path. No traffic is located at other links. Therefore only part of the network is being used. On the other hand, the proposed widest-least cost algorithm uses both upper and lower path for call connection. The results show that the proposed algorithm has lower value of call rejection rate.

Table 6.2 Simulation result of different path length for topology 1.

	Dijkstra	widest-shortest	widest-least cost
CRR	0.28	0.24	0.0
Max ABWE	84%	85%	71%
Min ABWE	0%	0%	22%

Comparing values of case1 and case2 reveals that while widest-shortest routing algorithm uses two metric for path calculation in some cases the resulting path depend only on first metric. Therefore, traffic is crowded on one side. The proposed routing algorithm performs perfectly on both cases. Besides path selection, it has ability to distribute traffic.

6.4.1 Call rejection rate and Path delay

For the topology shown in Figure 6.1(b), Figure 6.1(c), Figure 6.1(d) and Figure 6.1(e), simulation was executed with evenly and unevenly distributed traffic. Each call requires from 64Kbps to 512Kbps of bandwidth randomly selected from uniform distribution and delay of 100ms except in topology 5 which delay requirement is 500ms. The performance is checked by increasing the traffic load. The traffic load is increased with more frequent call requests. Adequate traffic load amount is different for variant network topology. In the simulation, the result is shown when call generation rate is from 1/3 call to 3 calls per minute. Result data shown in figures are the average value of 10 simulations with 600 minutes of simulation time each. Preferred result is to have lower call rejection rate and shorter path delay.

6.4.1.1 Result when traffic is evenly distributed

Figures 6.2, 6.3, 6.4 and 6.5 are the results when the traffic is evenly distributed in the network topology. In the figures, (a) show the result of *call rejection rate* in the simulation topologies, and (b) show the end-to-end path delay of the successful calls.

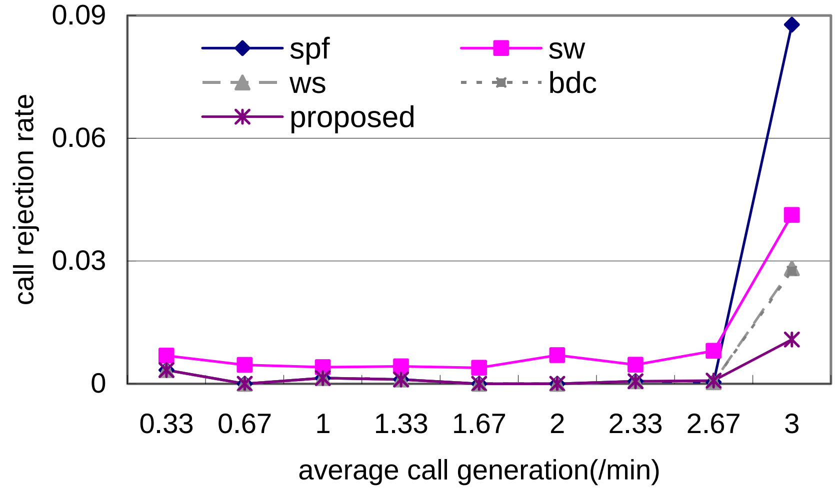
The *call rejection rate* of the proposed algorithm is better than other algorithms except for shortest-widest algorithm when there are many alternate paths. In Figure 6.2 (a) all algorithms are successful in path selection for most of the cases and show similar performance. Call rejection is less than 1% of the requested calls. There are almost no differences between algorithms.

Dijkstra algorithm starts to show performance degradation when network load increases but insignificant. Also, Figure 6.3 (a) shows that the proposed algorithm performance is good in light network loads and better than dijkstra, widest-shortest, and bandwidth-delay-constrained algorithm when network load increases.

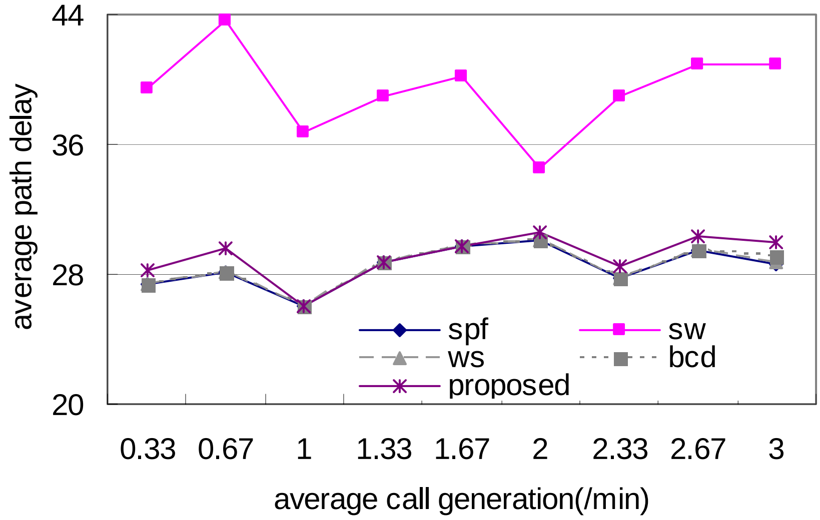
However, if there are few alternate paths the performance of the proposed algorithm is not preferred. As in case of topology 4, there are not much alternate paths and the results in Figure 6.4 (a) reveal that the proposed algorithm performs badly in this type of topology. Rejection rate is higher than other algorithm for most of the cases.

Even though shortest-widest algorithm show lowest call rejection rate in most cases, the tendency of selecting longer path than other algorithm reduces the preference of the algorithm. (b) figures show that shortest-widest algorithm finds longer delay paths than other algorithms. Other algorithms shows similar rejection rate as shortest-widest but takes shorter path to save network resources. Evenly distributed traffic lessens the advantage of shortest-widest algorithm.

This situation does not change in the case of topology 5. On the whole, performance difference from the simulation result is insignificant in evenly distributed traffic case. Therefore, well balanced network traffic does not require additional load balancing function.

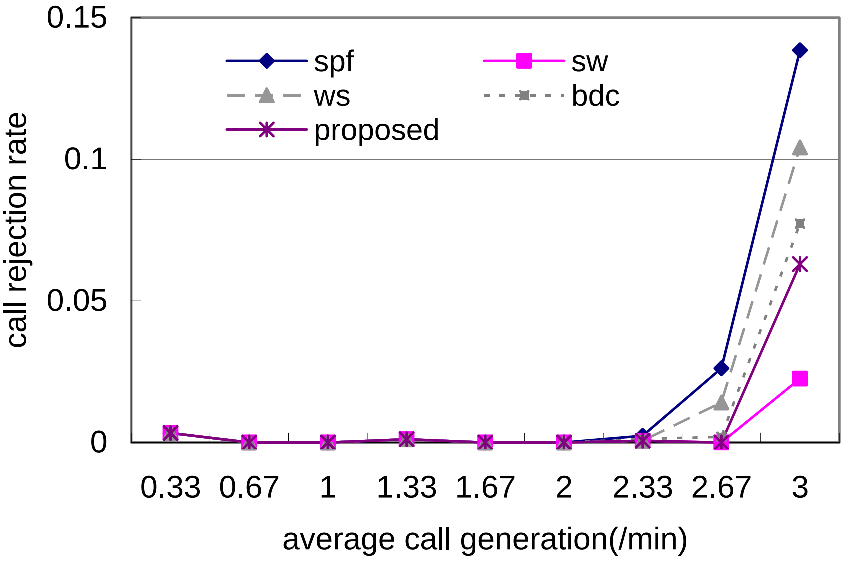


(a) call rejection rate

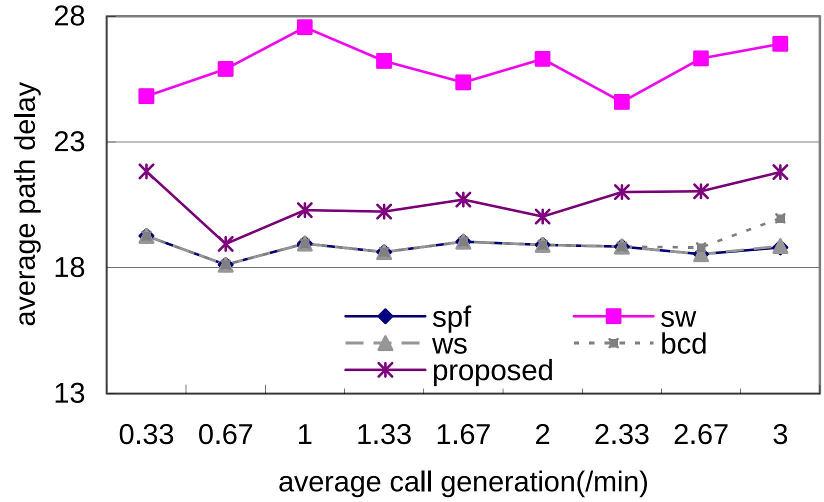


(b) average path delay

Figure 6.2 Result of evenly distributed network load in topology 2

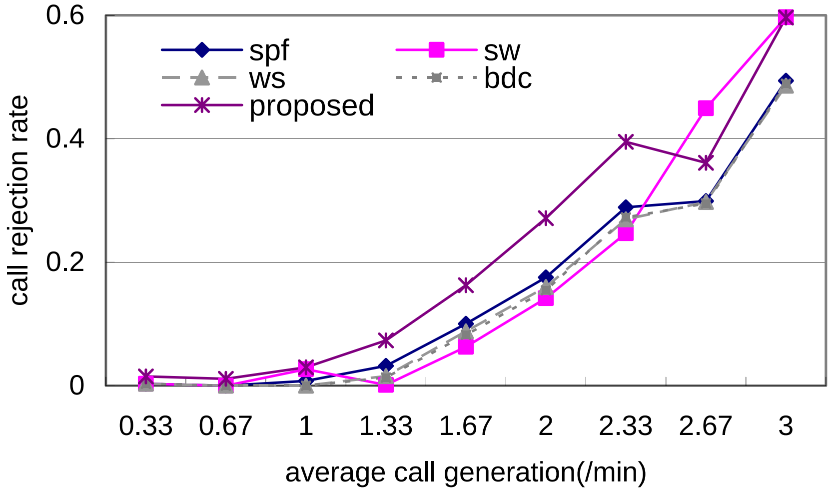


(a) call rejection rate

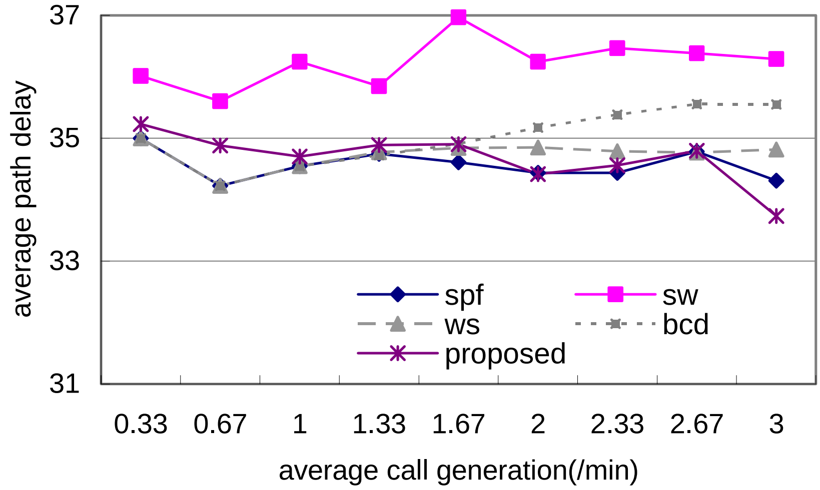


(b) average path delay

Figure 6.3 Result of evenly distributed network load in topology 3

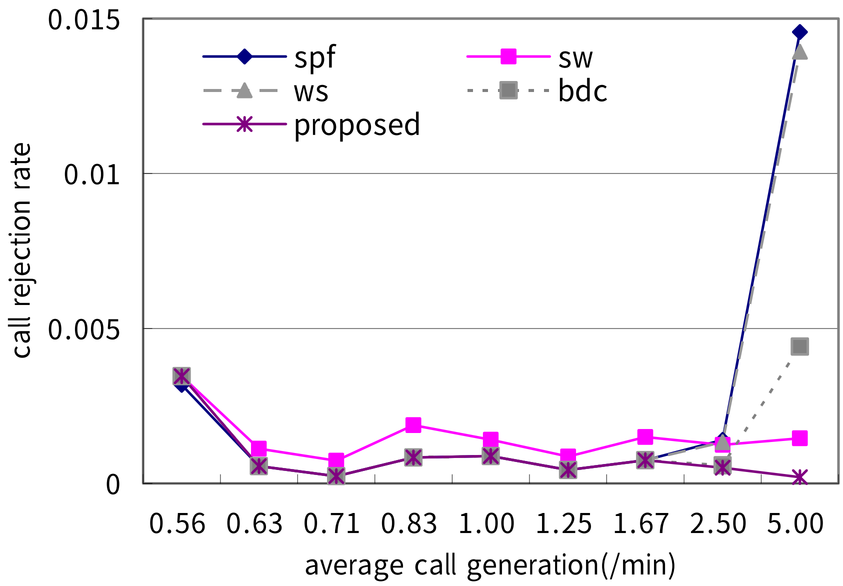


(a) call rejection rate

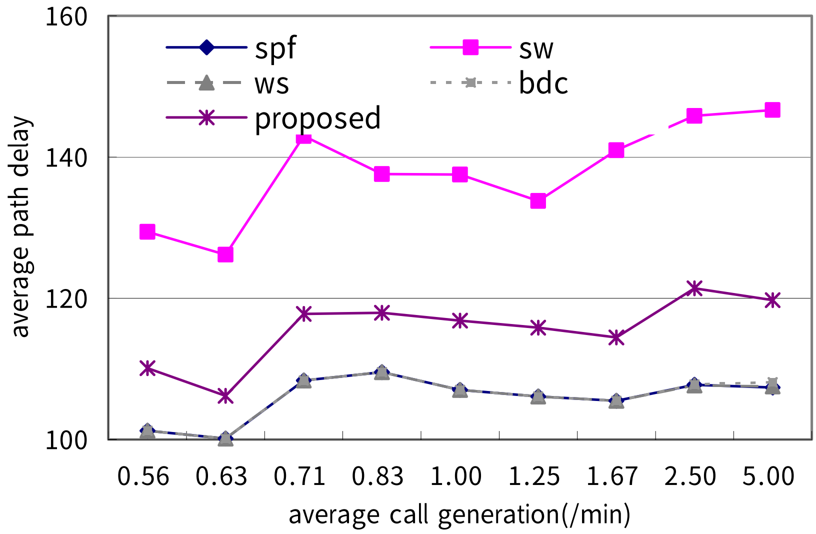


(b) average path delay

Figure 6.4 Result of evenly distributed network load in topology 4



(a) call rejection rate



(b) average path delay

Figure 6.5 Result of evenly distributed network load in topology 5

6.4.1.2 Result when traffic is unevenly distributed

Unlike evenly distributed traffic case, algorithms show performance difference when network load is not balanced. Figure 6.6, 6.7, 6.8 and 6.9 are the results when traffic load is not balanced. It is a result when 50% of generated traffic is concentrated in specific pair of nodes. The proposed algorithm exhibits its ability when network load is not distributed evenly and the results show that widest-least cost algorithm performs better in such conditions.

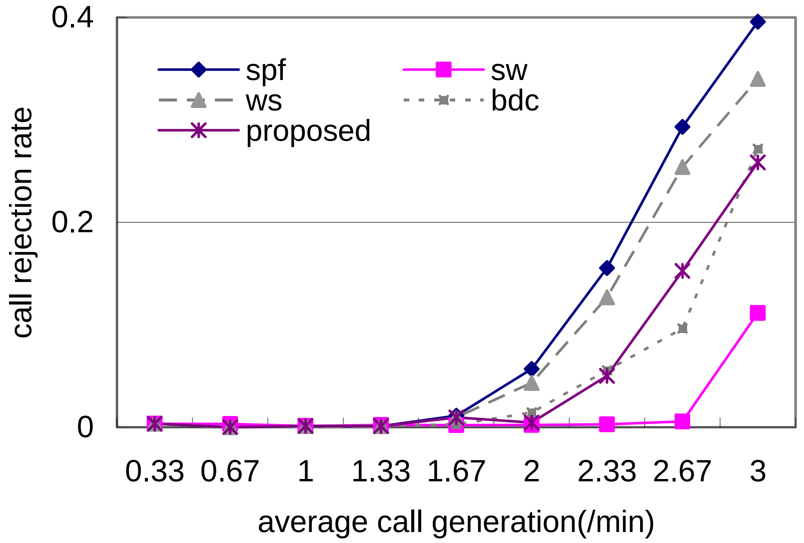
The *call rejection rate* of the proposed algorithm is lower than other algorithms except shortest-widest algorithm when the network topology provides many alternate paths. Figure 6.6 (a) shows that proposed algorithm has lower rejection rate than dijkstra and widest-shortest algorithm, similar rate as bandwidth-delay-constrained algorithm, and higher than shortest-widest algorithm. When network load is light all algorithms performs well. Performance difference is obvious when network load increases and traffic of some link is heavier than other links. Figure 6.7 (a) also shows that the proposed algorithm has lower rejection and performance difference is greater than the result in topology 2. The proposed algorithm shows good performance for long period even if network load increases.

The performance is worse than evenly distributed traffic when the network does not have many alternate paths. The proposed algorithm fails to find feasible path in many requests. Figure 6.8(a) show that balancing the load in this kind of topology is not preferred. Performance of load balancing shortest-widest algorithm show similar rejection rate as other short path algorithms.

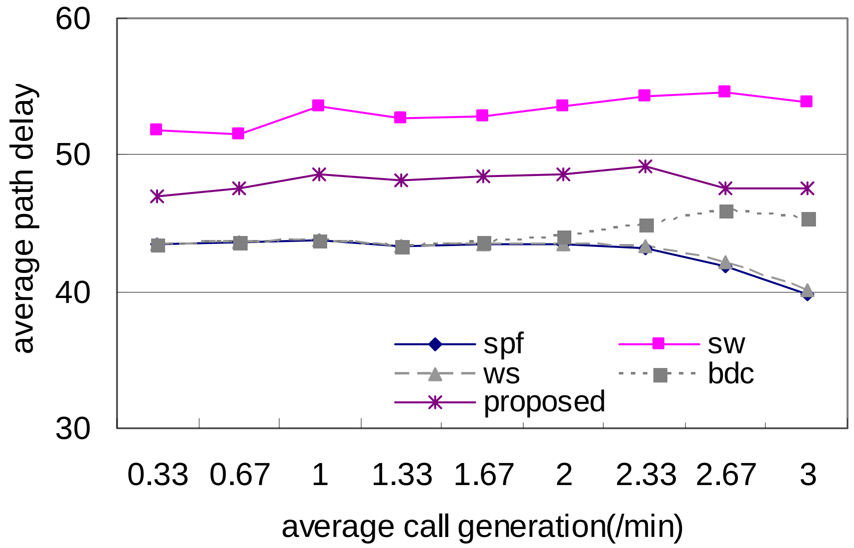
It may be preferred to have static threshold value for the link in case when there are not many alternate paths in network topology. It would cause the link not to be selected when link utilization is high as bandwidth-delay-constrained algorithm does. However, determining adequate threshold value is difficult.

Figure 6.6 (b) and Figure 6.7 (b) show that average delay of selected path for proposed algorithm have slighter longer paths than dijkstra shortest path and shorter than shortest-widest path. Path delay of dijkstra and widest-shortest algorithm decreases when network load is high. The reason is that heavy load cause network to be congested and only some of the source and destination pair located near can reserve resource for traffic. The longer paths selected by shortest-widest algorithm may result in almost 100% rejection for far apart source and destination pair if delay constraint is strict, for example 50ms in topology 2.

The result of simulation shows that if all the nodes generate similar amount of traffic to randomly distributed network, traffic is balanced and performance improvement is not distinct. However, if some nodes generate large amount of traffic, network traffic will be loaded in some links in the network, and performance of the proposed network is obvious. The proposed algorithm shows better performance when network has many alternate paths and traffic is unevenly distributed and shows similar performance when network load is balanced and network load is light. The result is Figure 6.9 shows that it can be applied to practical network with similar performance.

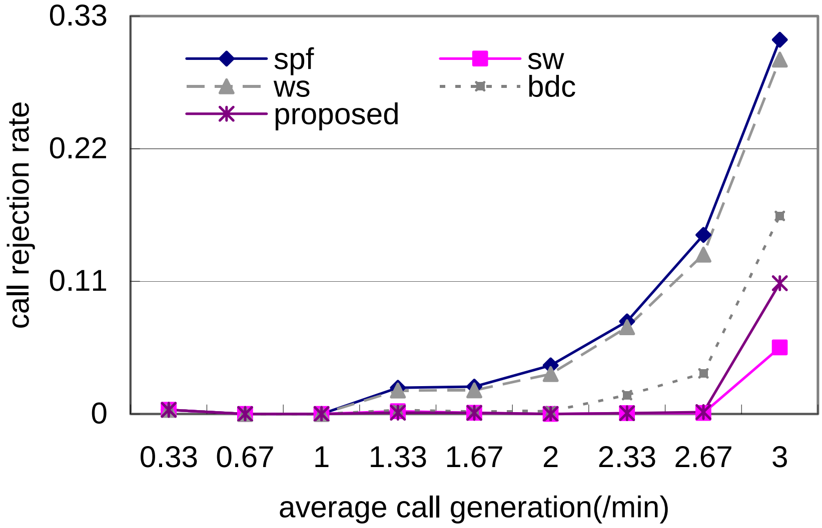


(a) call rejection rate

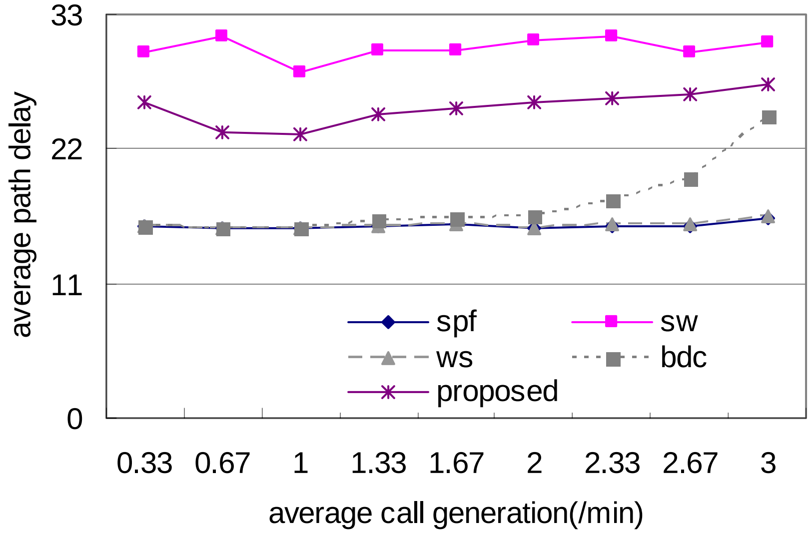


(b) average path delay

Figure 6.6 Result of unevenly distributed network load in topology 2

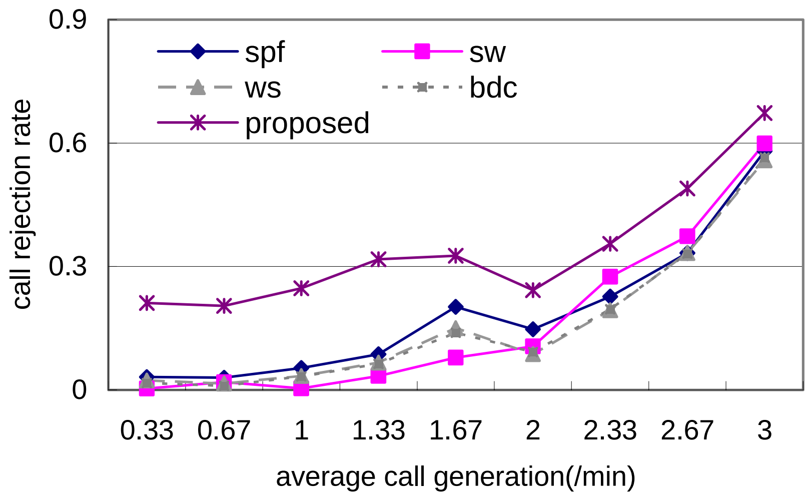


(a) call rejection rate

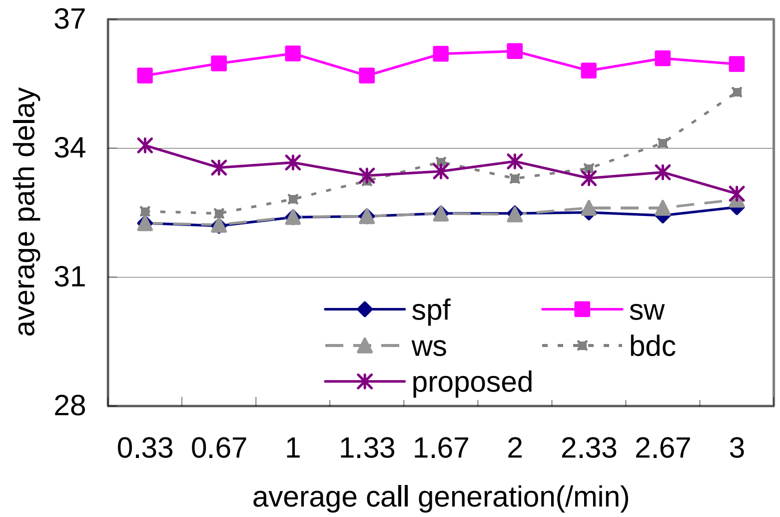


(b) average path delay

Figure 6.7 Result of unevenly distributed network load in topology 3

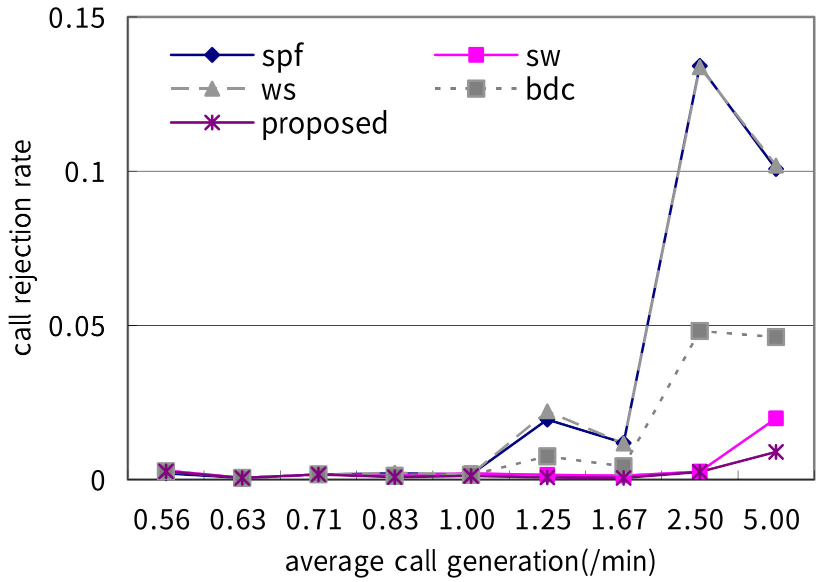


(a) call rejection rate

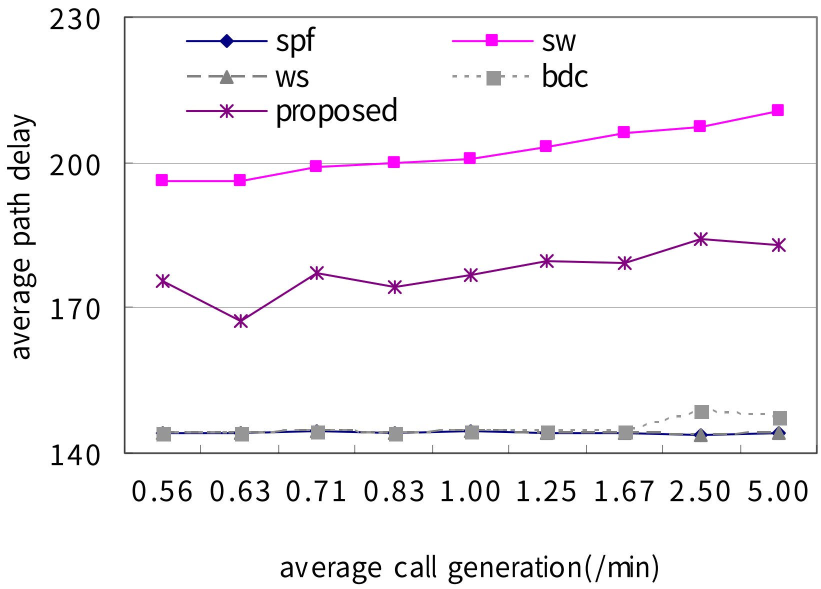


(b) average path delay

Figure 6.8 Result of unevenly distributed network load in topology 4



(a) call rejection rate



(b) average path delay

Figure 6.9 Result of unevenly distributed network load in topology 5

6.4.2 Network utilization

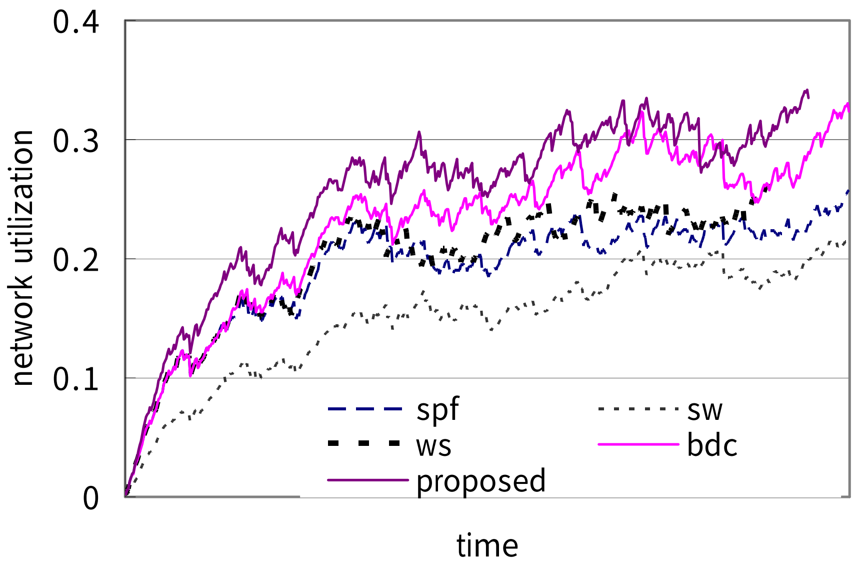
During the simulation, the network utilization value and maximum utilization value among the link utilization are logged. The results shown in Figure 6.10, Figure 6.11, Figure 6.12 and Figure 6.13 are the result for topology 2, 3, 4 and 5 each. It is the case when simulating with uneven traffic distribution and call generation rate is 2.33 calls / min.

The result from topology 2 and 3 show that that the proposed algorithm utilizes the network most. Network utilization value is the average value of all links. The proposed algorithm always shows higher value than other algorithms. This means that the network using the proposed algorithm utilizes the network most. It packs more traffic into the network. Also, the result from Figure 6.10(b) and 6.11(b) show that the maximum link utilization value is less than other algorithms except shortest-widest algorithm. This means that traffic is not loaded in links than the network using other algorithms.

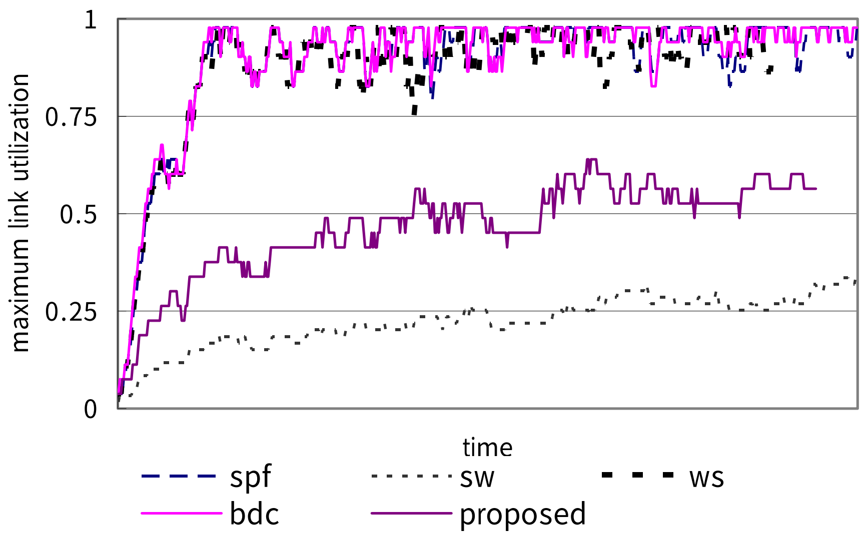
However, Figure 6.12 is the result from topology 4 where the proposed algorithm does not perform well. It shows difficulty in utilizing the network.

Figure 6.13 show that the proposed algorithm utilizes the network most and there is no loaded link when applied to practical network topology.

The result reveals that the proposed algorithm balances the load using alternate paths effectively.

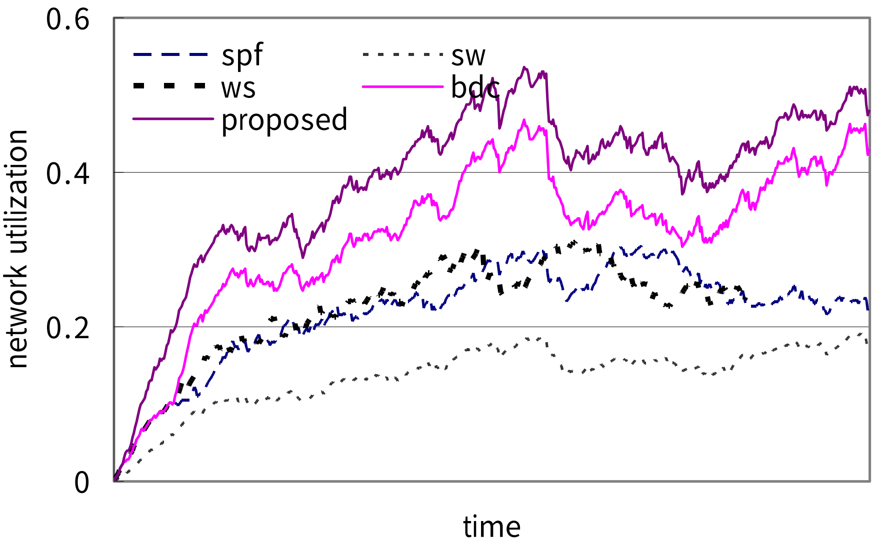


(a) Network Utilization

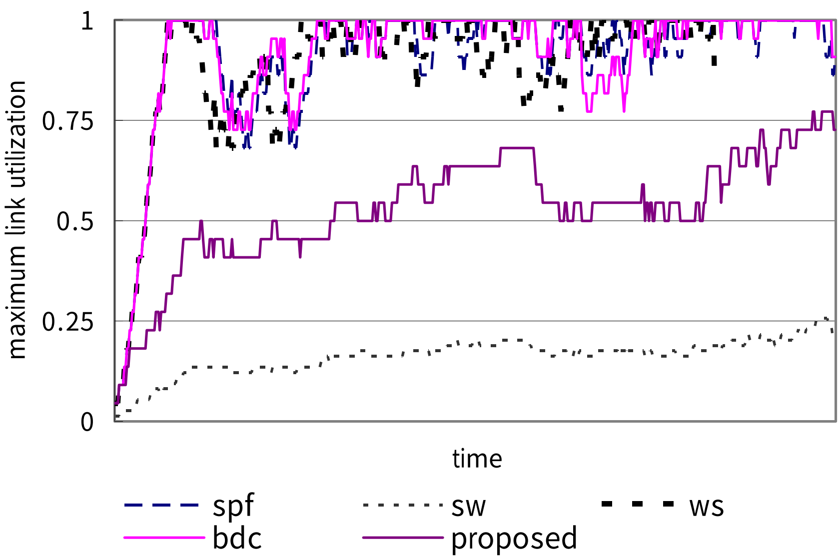


(b) Maximum link utilization

Figure 6.10 Utilization result in topology 2

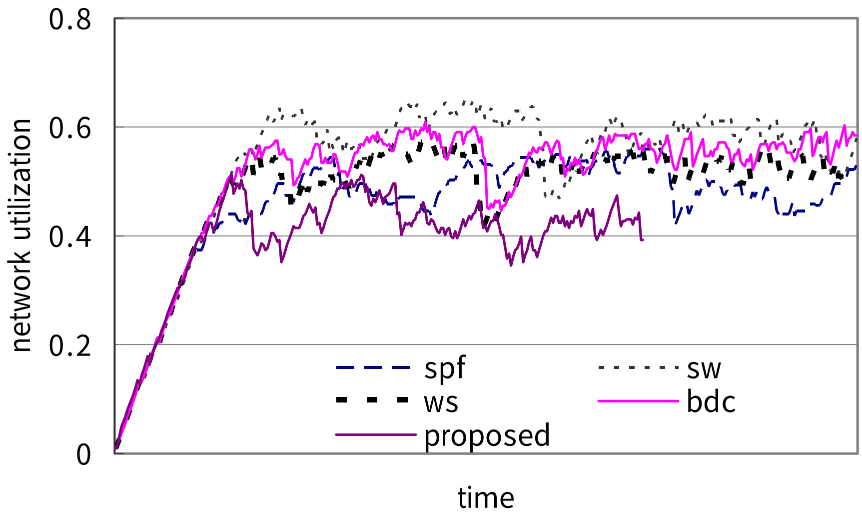


(a) Network Utilization

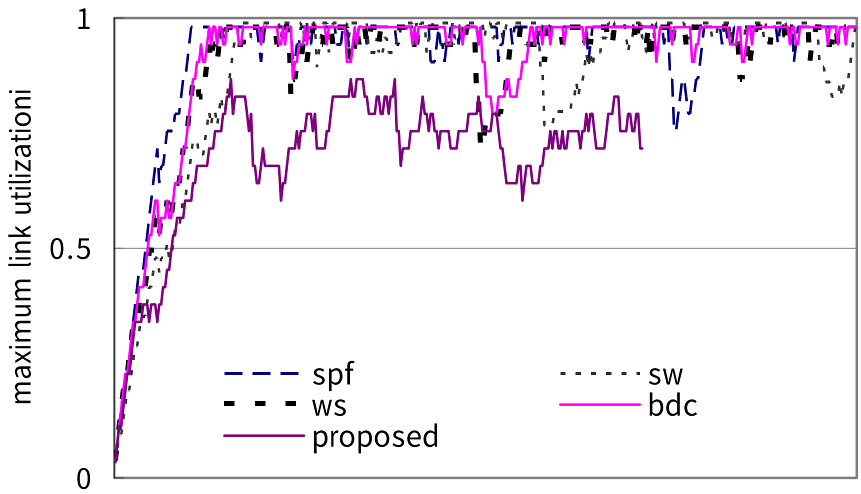


(b) Maximum link utilization

Figure 6.11 Utilization result in topology 3

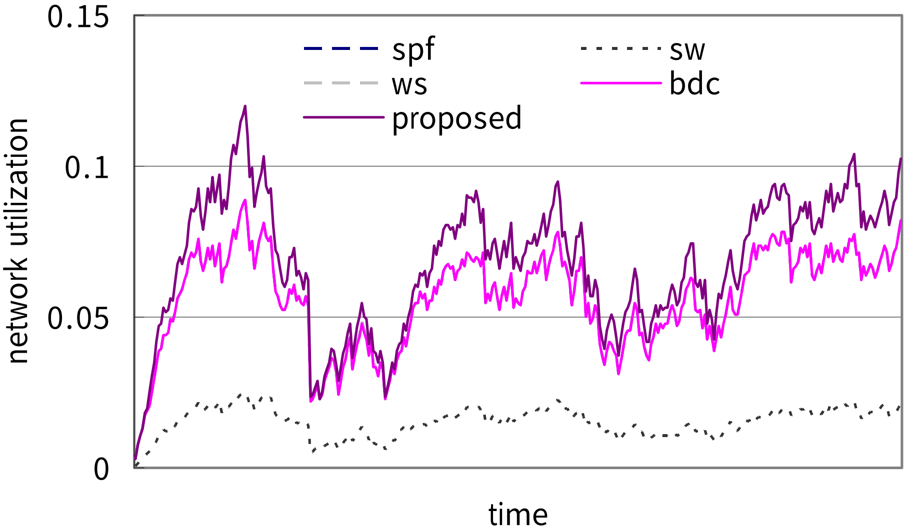


(a) Network Utilization

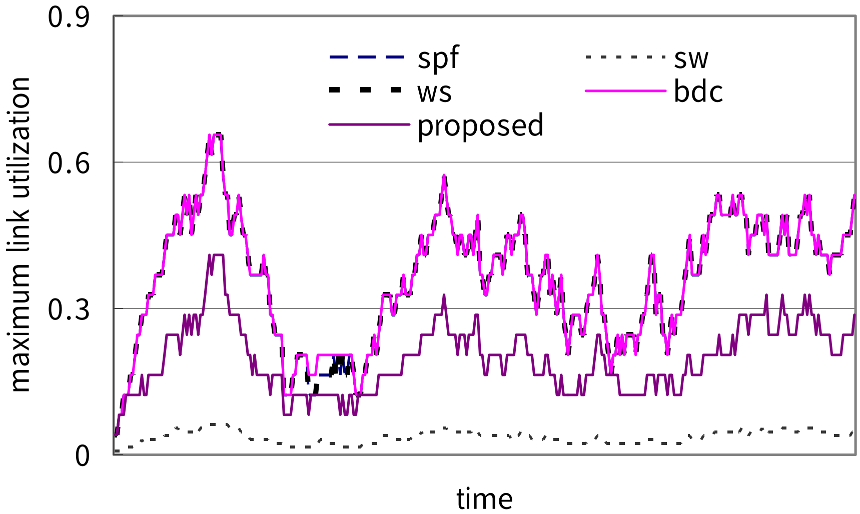


(b) Maximum link utilization

Figure 6.12 Utilization result in topology 4



(a) Network Utilization



(b) Maximum link utilization

Figure 6.13 Utilization result in topology 5

6.3 Complexity Comparison

Table 6.3 shows the comparison of routing algorithms. Proposed widest-least cost algorithm performs load balancing even though it has same time complexity as Dijkstra's algorithm and bandwidth-delay-constrained algorithm.

Table 6.3 Time complexity comparison of routing algorithms

algorithm	Routing strategy	Time complexity
Bellman-Ford	Distributed	O (n^2)
Dijkstra	Distributed	O ($n\log n + l$)
Bandwidth-delay-constrained [Wang 1996]	Source	O ($n\log n + l$)
Bandwidth-optimization [Wang 1996]	Distributed	O (nl)
Delay-constrained least cost [Salama]	Distributed	O (n^3)
Widest-Least Cost	Source	O ($n\log n + l$)

6.4 Summary

The statistical performance of the algorithm was studied through simulations. Performance is compared with Dijkstra's algorithm, shortest-widest algorith, widest-shortest algorithm, and bandwidth-delay-constrained algorithm. Result from various network topology shows that the proposed widest-least cost algorithm performs better when network topology has many alternate paths. With many alternate paths widest-least cost algorithm can select other paths to distribute the traffic. When network load is unevenly distributed, widest-least cost algorithm distributes traffic more than other algorithm. Therefore, call rejection is reduced. Moreover, network utilization is higher than other routing algorithms. Widest-least cost algorithm maximizes network resource usage.

Chapter 7

Multicast QoS Routing

with network dependent cost

Given a multicast group, the basic task of multicast routing is to construct a multicast tree. While, traditional multicast routing is concerned with connectivity, the proposed unicast routing algorithm is extended to construct multicast tree that support multicast routing. In this chapter, the method to extend the widest-least cost algorithm is discussed to be used in setting up the multicast tree.

7.1 Multicast routing

As shown in Figure 7.1, when a new member joins the multicast group, the tree must be extended to cover the new member. Simple method is to find a shortest path from the new member to the multicast tree. Since guarantee of service for all members is impossible, multicast service is served at best effort to the members. In order to support the members with best service possible, preferable path must be found in the first place. From the cost equation, unicast routing is extended to multicast tree set up by selecting path from new member to multicast tree using cost equation.

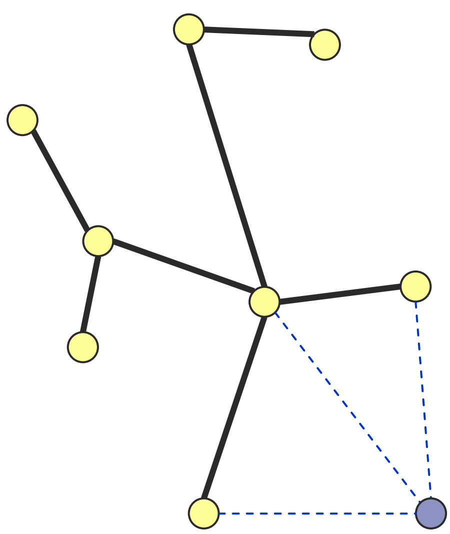


Figure 7.1 Adding a new member to multicast tree

In order to apply to multicast path setup, threshold must be redefined. Bias is obtained from the topology. Therefore, change of value is unnecessary.

$$d_j\left(1+\sigma\left(\frac{b}{c}-T\right)\times\frac{1-T}{1-\frac{b}{c}}\right)$$

Threshold can be defined two ways. First method is to change the definition of the variables used in the equation. Bandwidth metric is changed to residual bandwidth. If T is the residual bandwidth of the link and b is the requested bandwidth for the call, weight is imposed when requested bandwidth is larger than residual bandwidth T if a link. The exceeding amount of bandwidth requested to the residual bandwidth of the link is weighted. However, requested bandwidth may be considerable small compared to link capacity. Link weight may not be large enough to be effective. Increase of bias value may be needed.

Another method only changes the definition of threshold. If the requested bandwidth is *B*, threshold *T* is defined as $(1-B)/c_l$ for each link. Threshold is the minimum bandwidth rate required to satisfy the request. Therefore, weight is imposed with the amount of bandwidth surpassing the minimum required bandwidth.

7.2 Summary

Widest-least cost algorithm has been extended to support multicast routing. The cost equation of widest-least cost algorithm is applied to be suitable for multicast routing. The new member has more chance to have better service provided by better selection of path while joining multicast tree.

Chapter 8

Conclusions

Various network services require different Quality of Service (QoS). Many works have been done in order to serve QoS in the Internet. QoS routing is one of the pieces.

Routing with single metric result in partial view of the network. Guaranteeing QoS was to satisfy at the best with the requirement within the path selected with this partial view. Network links are characterized in various metrics. In order to reflect the characteristics of the links, routing with multiple metric is required. However, routing with more then one metric increases the complexity of calculation. It is a NP-complete problem which results in no solution acquired in polynomial time.

The widest-least cost algorithm is proposed for the NP-complete multi-metric constrained routing problem. It is a simple path selecting algorithm. The idea is to first reduce the NP-complete problem to a simpler one which can be solved in polynomial time by selecting adequate metric. Then, a new cost function is created which operates as an indicator of finding feasible path. The cost function adds weight to links that have higher utilization than other links in the network.

The implementation of algorithm to routing structure is described. Traditional server based routing architecture is used first. Then the routing manager structure is proposed. Routing manager collects the status information of network links needed to characterize the network status and distributes the information. Router calculates the paths. Router classifies the QoS traffic and best-effort traffic to provide adequate path for each service.

The performance of the proposed algorithm is evaluated by simulation. Even though the complexity of the proposed algorithm is same as Dijkstra's algorithm, widest-least cost algorithm has the ability to distribute the traffic to increase the network efficiency. Load balanced QoS traffic increases the possibility of best-effort traffic reaching the destination.

Widest-least cost algorithm is unicast routing algorithm. The algorithm is extended to multicast routing by constructing multicast tree with unicast connection of the new member to multicast routing tree.

For the future work, scalability of the routing algorithm is needed. Widest-least cost algorithm requires network utilization value and bias value that is acquired from the network and dynamic network information loses accuracy when network size increases.

Bibliography

[Apostolopoulos 1999a] G.Apostolopoulos, R.Guerin, S.Kamat and S.K. Tripathi, "Server Based QoS Routing", Globecom'99, pp.762-768, 1999.

[Apostolopoulos 1999b] G.Apostolopoulos, R.Guerin, and S.Kamat, "Implementation and Performance Measuremets of QoS Routing Extenstions to OSPF", INFOCOM'99.

[Bertsekas 1992] D.Bertsekas and R.Gallager, Data Networks, Prentice Hall, 1992.

[Bertsekas 1998] D.Bertsekas, Network Optimization: Continuous and Discrete Models, Athena Scientific, 1998.

[Chen] J. Chen, P.Druschel, and D.Subramanian, "A New Approach to Routing With Dynamic Metrics", INFOCOM'99.

[David] B. David and Y. Rekhter, MPLS Technology and Applications, Morgan Kaufmann Publishers, 2000.

[Elwalid] A. Elwalid, C. Jin, S. Low, and I. Widjaja, "MATE: MPLS Adaptive Traffic Engineering", INFOCOM2001, pp. 1300-1309, 2001.

[Ergun] F. Ergun, R.Sinha, and L.Zhang, "QoS Routing with Performance-Dependent Costs", INFOCOM2000, 2000.

[Fei] A. Fei, and M. Gerla, "Smart Forwarding Technique for Routing with Multiple QoS Constraints", GLOBECOM 00, pp. 599-604, 2000.

[Fortz] B. Fortz and M. Thorup, "Internet Traffic Engineering by Optimizing OSPF Weights", INFOCOM2000, 2000.

[Ghosh] D. Ghosh, V. Sarangan, and R. Acharya, "Quality-of-Service Routing in IP Networks", IEEE Transactions on Multimedia, Vol.3, No.2, June, 2001.

[Guerin 1996] R. Guerin, A. Orda, and D. Williams. "QoS Routing Mechanisms and OSPF Extensions", Internet draft, Nov.,1996.

[Guerin 1997] R. Guérin , and A. Orda, " QoS-based Routing in Networks with Inaccurate Information: Theory and Algorithms", INFOCOM'97, pp.75-83, 1997.

[Hao] F.Hao and E.Zegura, "On Scalable QoS Routing: Performance Evaluation of Topology", INFOCOM2000, 2000.

[Huitema] C.Huitema, Routing in the Internet, Prentice Hall, 1999.

[Jeon] H.Jeon, S.Kim, Y.Kim, H. Kim, and J.Lee, "Performance Analysis of QoS Routing with Network Dependent Delay Cost", ICOIN-15, 2001.

[Juttner] A. Juttner, B. Szviatovszki, I. Mecs, Z. Rajko, "Lagrange Relaxation Based Method for the QoS Routing Problem", INFOCOM2001, pp. 859-868, 2001.

[Kim] S. Kim, H. Jeon, J. Lee, Y. Kim, and H. Kim, "Load Balancing Using Modified Cost Function for Constraint-based Routing Over MPLS Networks", ICOIN-16, 2002.1.

[Kim 2001] S.H.Kim and M. Lee, "Server Based QoS Routing with Implicit Network State Updates", GLOBECOM2001, 2001.

[Korkmaz 1999] T.Korkmaz and M.Krunz, "A Randomized Algorithm for Finding a Path Subject to Multiple QoS Constraints", Globecom'99, pp.1694-1698, 1999.

[Korkmaz 2001] T. Korkmaz, and M. Krunz, "Multi-Constrained Optimal Path Selection", INFOCOM2001, pp. 834-843, 2001.

[Korkmaz 2002] T. Korkmaz, M. Krunz, and S. Tragoudas, "An Efficient Algorithm for Finding a Path Subject to Two Additive Constraints," Computer Communications Journal 25(3):225-238, 2002.

[Lee] W.C.Lee, M.G. Hluchyi, and P. Humblet, "Routing Subject to Quality of Service Constraints in Integrated Communication Networks", IEEE Network, pp.46-55, July / August, 1995.

[Liu] G. Liu, and K. Ramakrishnan, "A*Prune: An Algorithm for Finding K Shortest Paths Subject to Multiple Constraints", INFOCOM2001, pp. 743-749, 2001.

[Ma 1996] Q.Ma, P.Steenkiste, and H.Zhang, "Routing High-bandwidth Traffic in Max-min Fair Share Networks", ACM SIGCOMM'96.

[Ma 1997a] Q.Ma, and P.Steenkiste, "On path Selection for Traffic with Bandwidth Guarantees", IEEE International conference on Network Protocols, Oct., 1997.

[Ma 1997b] Q.Ma and P.Steenkiste, "Quality-of-Service Routing for Traffic with Performance Guarantees", IFTF 5th International Workshop on Quality of Service, May, 1997.

[Murthy] S.Murthy, and J.Garcia-Luna-Aceves, "Loop-Free Internet Routing Using Hierarchical Routing Trees", IEEE INFOCOM '97.

[Nelakuditi] S. Nelakuditi, Z.Zhang, and R. Tsang, "Adaptive Proportional Routing: A Localized QoS Routing Approach", INFOCOM2000, 2000.

[Orda 1998] A. Orda, "Routing with End to End QoS Guarantees in Broadband Networks", INFOCOM 98, pp.27-34, 1998.

[Orda 1999] A.Orda, "Routing with End-to-End QoS Guarantees in Broadband Networks", IEEE/ACM Transactions on Networking, Vol.7, No.3, June, 1999.

[Orda 2000] A. Orda and A.Sprintson, "QoS Routing: The Precomputation Perspective",INFOCOM 2000, 2000.

[PNNI] ATM Forum Private Network-Network Interface Specification Version 1.0, March 1996.

[Rao] N.Rao and S.Batsell, "QoS Routing Via Multiple Paths Using Bandwidth Reservation", IEEE INFOCOM'98, March 1998.

[RFC 791] "Internet Protocol DARPA Internet Program Protocol Specification", RFC 791, Sep., 1981.

[RFC 1633] R. Braden, D. Clark, and S. Shenker, “Integrated Services in the Internet Architecture: an Overview”, RFC 1633, June, 1994.

[RFC 2212] S. Shenker, C. Partridge, and R. Guerin, “Sprcification of Guaranteed Quality of Service”, RFC 2212, Sep., 1997.

[RFC 2215] S. Shenker, and J. Wroclawski, “General Characterization Parameters for Integrated Service Network Elements”, RFC 2215, Sep., 1997.

[RFC 2386] E.Crawley, R.Nair, B.Rajagopalan and H.Sandick, “A Framework for QoS-based Routing in the Internet”, RFC 2386, Aug. 1998.

[RFC 2474] K. Nichols, S. Blake, F. Baker, and D. Black, “Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers”, RFC 2474, Dec., 1998.

[RFC 2475] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, “An Architecture for Differentiated Services”, RFC 2475, Dec., 1998.

[RFC 2702] D. Awduche, “Requirements for Traffic Engineering over MPLS”, RFC 2702, Sep. 1999.

[RFC 2748] D. Durham, J. Boyle, R. Cohen, R. Rajan, and A. Sastry, “The COPS(Common Open Policy Service) Protocol”, RFC 2748, Jan. 2000.

[RFC 3031] E.Rosen, A. Viswanathan, and R. Callon, “Multiprotocol Label Switching Architecture”, RFC 3031, Jan., 2001.

[Salama] H.Salama, D. Reeves, and Y.Viniotis, “A Distributed Algorithm for Delay-Constrained Unicast Routing”, IEEE INFOCOM ‘97.

[Saragan] V. Sarangan, D. Ghosh, and R. Acharya, “Distributed QoS Routing for Multimedia Traffic”, GLOBECOM 00, pp. 455-459, 2000.

[Sobrinho] J. Sobrinho, “Algebra and Algorithms for QoS Path Computation and Hop-by-Hop Routing in the Internet”, INFOCOM2001, pp. 727-735, 2001.

[Vogel] R.Vogel, R.Herrtwich, W.Kalfa, H.Wittig, and L.Wolf, “QoS-Based Routing of Multimedia Streams in Computer Networks”, JSAC, vol 14, no.7, September 1996.

[Vutukury 1999] S.Vutukury and J.J.Garcia-Luna-Aceves, "A Distributed Algorithm for Multipath Computation", Globecom'99, pp.1689-1693, 1999.

[Vutukury 2001] S. Vutukury, and J.J. Garcia-Luna-Aceves, "MDVA: A Distance-Vector Multipath Routing Protocol", INFOCOM2001, pp. 557-564, 2001.

[Wang 1999] Z.Wang, "On The complexity of quality of service routing", *Information Processing Letters*, V.69, N.3, pp111~114, February 1999.

[Wang 1996] Z.Wang and J.Crowcroft, "Quality-of-Service Routing for Supporting Multimedia Applications", IEEE Journal on Selected Areas in Communications, Vol. 14, No. 7, Sep., 1996.

[Wang 2001] Y. Wang, Z. Wang, and L. Zhang, "Internet Traffic Engineering without Full Mesh Overlaying", INFOCOM2001, pp. 565-571, 2001.

[Yen] H. Yen, and F. Lin, "Near-optimal Delay Constrained Routing in Virtual Circuit Networks", INFOCOM2001, pp. 750-756, 2001.

[Yuan] X. Yuan, and X. Liu, "Heuristic Algorithms for Multi-Constrained Quality of Service Routing", INFOCOM2001, pp. 844-853, 2001.

[Zhang] Zhang, Sanchez, Salkewicz, and Crawley, "Quality of Service Extensions to OSPF", Internet draft, Sep., 1997.

국문 요약

MPLS 네트워크에서 멀티미디어 서비스를 위한 QoS 기반의 네트워크 효율적인 라우팅 방안

본 논문은 여러 개의 다중 메트릭을 이용하여 경로 선택을 수행하는 문제에 대해 연구했다. 현재 인터넷에서의 라우팅은 하나의 메트릭을 이용하여 경로선택을 수행하고 있으며, 하나의 메트릭에 의해 선택된 경로는 하나의 요구사항만을 충족시켜 줄 수 있는 best-effort 서비스이다. 서비스를 제공하기 위해 전달되는 트래픽은 종류에 상관없이 선택된 경로를 통해 전달이 된다. 그러나, 다양한 네트워크 서비스들은 서로 다른 서비스 품질(QoS)을 요구한다. The Internet Engineering Task Force (IETF)는 이러한 서비스 품질을 만족시키기 위한 방법으로 여러 서비스 모델과 방안을 제시하였다. 그중 Multiprotocol label switching(MPLS)가 많은 주목을 받았는데, MPLS 는 다중의 프로토콜들에 적용가능하며, 빠른 전송과 자원예약 및 경로 관리가 가능하다. 또한 MPLS 는 traffic engineering 을 수행하기 위한 유용한 도구이다.

여러 서비스 품질(QoS)을 요구하는 콜에 대해 만족시켜주기 위해서는 경로가 여러 제한에 걸리게 된다. 많은 경우에 경로 선택문제라는 것은 NP-complete 이다.

본 논문에서는 계산 가능한 시간내에 경로를 찾는 라우팅 알고리즘을 제안한다. 지연 메트릭에 기반을 두고 네트워크 자원에 대한 정보에 영향을 받는 비용 메트릭을 이용한 widest-least cost 라우팅 알고리즘을 제안한다. 이 알고리즘은 다중의 메트릭을 이용하는 라우팅 알고리즘으로써 트래픽 분산 능력을 갖고 있어서, 네트워크 링크가 부하가

적을 때는 최소거리 경로를 선택하도록 하는 것이고, 네트워크 부하가 클 경우에는 traffic engineering 이 수행되어 트래픽을 다른 경로로 이동시키도록 하는 것이 목표이다.

라우팅 알고리즘은 기존의 라우팅 서버 구조와 제안한 라우팅 매니저 구조에서 적용하였다. 라우팅 절차는 MPLS 네트워크의 제한조건기반 라우팅 (constraint based routing)을 따르도록 하여 특정 서비스 품질을 요구하는 트래픽과 best-effort 트래픽에 따라 경로 선택을 수행한다. 이러한 결과는 다른 알고리즘과의 실험 결과를 통해 비교하였다.

핵심되는 말 : QoS, 라우팅, traffic engineering, 다중메트릭