NOTE TO USERS

This reproduction is the best copy available.

UMI[®]

QoS Routing Using Traffic Forecast

– A Case Study of Time-Dependent Routing

By

Yuekang Yang
B.Eng. Shanghai Jiao Tong University

A thesis submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Applied Science

Ottawa-Carleton Institute for Electrical Engineering
Department of Systems and Computer Engineering

Carleton University
Ottawa, Ontario, Canada
August 2004

© Copyright 2004, Yuekang Yang

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-99588-7

Our file Notre référence

ISBN: 0-612-99588-7

NOTICE:

The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protègent cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

Chapter 1 Introduction

This chapter briefly introduces the Internet Quality-of-Service (QoS) routing problem. After that, this chapter will contain a discussion on the motivation of this thesis, which tries to address whether traffic forecasting improves QoS routing performance in IP networks compared with using current state information. The chapter will then describe the scope, contributions, and organization of the thesis.

1.1 IP QoS Routing

Over the past few decades, the topic of IP routing has been discussed extensively. Networks have evolved from circuit switching to packet switching. Many factors – such as economic considerations, the recent large-scale adoption of optic/electronic technologies, and changing software behaviors – have contributed to the evolution. As a key part of the necessary solution bundled in the packet-switching architecture, the IP routing solution has naturally attracted a great deal of attention due to its general running performance.

The tremendous growth of the Internet stands as a testimony to the fundamental success of the packet-switching architecture. As the number of computers, applications, and people using computers increases, the packet-switching architecture can dynamically facilitate the communication between computers on an as-needed basis. Interestingly, however, as the packet-switching architecture has become dominant in the telecommunication industry due to its high interconnectivity and flexibility, its performance has tended to become harder to control than the circuit-switching based systems. As a result, and as the Internet continues to grow, the QoS routing problem has

become one of the most challenging issues in the area of IP routing research, but the problem has not yet been solved satisfactorily.

1.1.1 The General Problem of IP Routing

In the context of the Internet, techniques that quickly and efficiently determine the best path from one node to another are referred to as IP routing. Similar problems have been widely studied in other disciplines: graph theory, operation research, and network routing.

Traditionally, IP routing belongs to the category of connectionless routing. The nodes typically run a distributed routing protocol that consistently decides the next hop routing tables, resulting in optimized and loop-free paths. A connectionless routing service does not guarantee the quality of the packet delivery in addition to carrying out the packet delivery, since connectionless routing operates on a best-effort basis, without any consideration of bandwidth allocation. Recently, however, some connection-oriented technologies have been introduced into IP routing. Connection-oriented routing establishes a virtual path (because it is still a packet-switching architecture) between the source and the destination before any data is transferred. Once the path is established, all data from the source to the destination travels along the same pre-established path through the network.

Algorithms that calculate the best path from any one node to every other node may be centralized or distributed. The performance of distributed algorithms approaches the optimality of centralized counterparts only when network conditions are steady. Otherwise, distributed algorithms may experience race conditions.

There is a widely used algorithm for selecting the “least cost” path in a network. The “least cost” path is sometimes referred to as the “shortest” path. In most simple cases, least-cost routing is equivalent to shortest-path routing. Nevertheless, IP routing can become complicated when routing is subject to constraints. Constraint-Based Routing (CBR) denotes a class of routing algorithms that base path selection on a set of constraints or requirements, in addition to the destination information [5].

1.1.2 Link Capacity Constraint

The most common and realistic constraint is link capacity. To satisfy this constraint, the routing algorithm should consider not only the “least cost” when making a routing decision, but also the link capacities. The algorithm should not offer traffic to any link in excess of its capacity. As we shall see later, in Chapter 4, the link capacity constraint is necessary for optimizing the network-wide performance.

In the “shortest path” algorithm, i.e., the Dijkstra or Bellman-Ford shortest path algorithm, the link capacity constraint is basically ignored.

1.1.3 QoS Constraints

Other constraints include QoS-related constraints, such as bandwidth, delay and packet loss requirements. In terms of routing algorithms, if QoS requirements are imposed in addition to the link capacity constraint, the associated IP routing is referred to as QoS routing.

The QoS constraints can be expressed by the maximum or minimum bound [26] or the notion of differentiated priorities using long-term traffic engineering statistics [2].

1.1.4 Other Constraints

Constraints other than link capacity and QoS requirements could include administrative policies. These policies can usually influence the way a QoS routing algorithm calculates and create routes. They might be regarded as an additional management measure to guide the behaviour of a network through high-level declarative instructions and to dynamically extend the functionality of existing static network management systems.

1.1.5 The Complexity of QoS Routing Algorithms

Given a set of QoS routing requests and a network, determining whether there is a routing solution that meets all QoS requests simultaneously is an NP complete problem [12]. Therefore, no known routing algorithm provides an optimal solution with the link capacity constraint and the QoS constraints using less computation than a brute comparison of all possible alternatives.

1.2 Thesis Motivation

The primary motivation for this thesis comes from Figure 1-1, adopted from reference [21], which shows three typical traces of link traffic in the Sprint Internet core in May 2002.

Figure 1-1 clearly shows that aggregated Internet traffic demonstrates strong predictability over a long period of time (i.e. hours, days and weeks). The two spikes between dotted lines represent Saturday and Sunday traffic respectively, while the remaining periodic movement represents the weekday load. The daily traffic pattern between peak hours and normal hours is easily observed.

Figure 1-1. Three Traces of Link Traffic

In this thesis, we question whether information regarding such strong traffic patterns can be utilized in helping to solve the QoS routing problem, or whether traffic forecasting is helpful in QoS routing. More precisely, there are a few sub-questions:

- ❑ Can traffic forecasting improve QoS routing performance?
- ❑ What if the forecast is not perfect?
- ❑ What kind of traffic forecast is required for QoS routing?

1.3 Thesis Scope

Realizing that the QoS routing problem is intractable in many aspects, we limit the scope of this thesis in order to provide us with some insights into the problem. The scope is four-fold:

First, the thesis assumes that aggregated Internet traffic has a periodic predictable pattern. Based on this assumption, we further assume that the forecast of traffic demand is roughly correct. Obviously, forecasting Internet traffic could be another orthogonal

work in itself, but this will not prevent us from making the assumption first and then looking back to evaluate what kind of forecast is valuable in terms of QoS routing.

Second, only the link capacity constraint and QoS constraints are considered in the discussion. Other possible constraints, such as administrative policies, are not addressed at this time.

Third, link or node failure and traffic backup are not considered in this thesis.

Finally, it is assumed that the networks discussed in this thesis fall under one single administration authority, or in one routing domain. Inter-domain QoS routing is important, but beyond the scope of this thesis.

1.4 Thesis Contributions

The contributions of this thesis include:

1. Proposing a time-dependent IP QoS Routing algorithm. Based on the Widest-Shortest Path (WSP) algorithm, the thesis proposes a new time-dependent QoS routing algorithm, Time-Dependent WSP (TDWSP), which can greatly improve the routing performance in a certain sphere by combining a fixed traffic forecast and an existing time-dependent QoS routing algorithm. The thesis further suggests that the combination of a priori knowledge of traffic demand and any state-dependent routing algorithms may yield positive results, if the Internet traffic is assumed to be predictable.
2. Identifying the feasibility of QoS routing based on a fixed traffic forecast. This thesis does not deal with an adaptive forecasting schema that runs dynamically at real-time and is based on traffic conditions. Instead, the thesis identifies that even a static piece of information is valuable in QoS routing and can remain effective

within a relatively wide range of traffic conditions.

3. Advocating the extreme traffic value forecast. From the results of the simulation, this thesis discloses the fact that in terms of IP routing performance, the sensitivity of the forecast is asymmetrical. In the context of IP routing, this fact has numerous implications, which could be utilized not only for simplifying QoS routing algorithms, but also for collecting, analyzing and predicting traffic statistics.

Part of this thesis appeared in the following publication:

- Y. Yang & C. H. Lung, "Traffic Forecast in QoS Routing," *Proceedings of Biennial Symposium on Communications*, pp. 277-279, Kingston, ON, June 2004.

1.5 Thesis Organization

The thesis is structured as follows. Chapter 2 provides an overview of related research, including proposed QoS routing algorithms and difficulties. Chapter 3 establishes a platform for discussion by representing networks and traffic and simplifying QoS constraints. Chapter 4 compares linear and non-linear objective functions and describes how to evaluate routing performance. Chapter 5 contains the results of the simulation and an analysis of state-dependent routing, followed by Chapter 6, which proposes a time-dependent algorithm. Corresponding simulations, implications of the results and advantages of the time-dependent algorithm are also provided in Chapter 6. Chapter 7 consolidates both state- and time-dependent routing concepts into a unified view, and makes some comments on QoS routing. The thesis is concluded in Chapter 8 with a summary of the results of the thesis, highlights of the conclusions, and suggested topics for future research.

Chapter 2 Related Work

There is a large body of literature on QoS routing. This chapter will begin with a discussion of the classifications of QoS routing, QoS standards from IETF, and traffic engineering, an adjacent research area of QoS routing. A brief review of Internet traffic forecasting is included towards the middle of the chapter. Since ATM was the first networking technology designed with QoS in mind, this chapter then describes an implemented QoS routing algorithm, PNNI, as well as some other proposed QoS routing algorithms. Finally, the chapter will conclude with a discussion of some of the major challenges faced by QoS routing algorithms.

2.1 Classifications of QoS Routing

Two classifications of QoS routing are introduced below. The first one is popular in the literature, the next one less so, but it is related to the goal of this thesis.

2.1.1 Connection-Oriented vs. Connectionless Paradigm

Within the context of IP packet-switching, although the connection-oriented (e.g. MPLS switching) and connectionless (e.g. traditional IP routing) paradigms forward information in a network in fundamentally different ways, the measures utilized to manage traffic and deliver QoS are surprisingly similar. This is no accident, because network designers, switch manufacturers and network engineers only have a limited solution set.

As the name implies, a pure connectionless paradigm never establishes connections of any kind, so guaranteeing packet delivery is not straightforward. For

example, the concept of QoS per connection does not make sense in a connectionless protocol like IP. Therefore, in the connectionless paradigm network designers and engineers adopt the notion of QoS to a flow of packets, such as a micro-flow. In another example, if any resource reservation protocols (such as RSVP) want to pass reservation requests between routers, the router that receives the request has to either accept or reject each request. Therefore, some aspects of a connection-oriented service are exhibited. In conclusion, a connectionless paradigm always has to bring in some connection-oriented behavior in order to provide QoS services.

No matter which paradigm researchers focus on, there are pros and cons to consider (in Chapter 7 we will extend this discussion, based on our observations). On one hand, the connection-oriented paradigm can straightforwardly reserve capacity to ensure QoS. Accurately measuring the usage of the connection and collecting statistics is simple and cheap. In addition, coming up with a performance estimation of the actual QoS delivered and a diagnosis of performance issues is easier than for its connectionless counterpart.

On the other hand, the complexity of the connection-oriented paradigm is much higher than that of the connectionless paradigm. The cost of maintaining a connection-oriented network via signaling protocols is expensive. Solving the scalability issue of delivering a connection-oriented QoS service is obviously a significant challenge.

2.1.2 State-Dependent vs. Time-Dependent Mechanism

Another classification divides QoS routing algorithms into two categories: state-dependent and time-dependent. Since the underlying concepts of IP routing are very

similar to those in the old telephony industry, researchers have simply transferred these classification terms to the Internet context.

A QoS routing algorithm works as follows: Upon the arrival of a routing request, the algorithm either selects a route that satisfies the QoS requirements, without significantly degrading the overall network performance, or rejects it. If the QoS routing algorithm decides to accept the request, corresponding network resources are reserved at each link of the route during the lifetime of the request. Note that more than one route can usually meet the request, and the best one, or the one that is good enough, is desired in terms of network performance.

A routing algorithm that is able to find the optimal route for every QoS request, without knowing either the history or the future traffic demand, is called a *state-dependent* mechanism. Conversely, a routing algorithm that has knowledge of the history or future traffic demand is called a *time-dependent* mechanism [11].

Interestingly, most proposed QoS routing algorithms [51, 52, 53] are state-dependent mechanisms. A minimal amount of research [12] has been done on the time-dependent category.

2.2 Internet QoS Standards

Internet RFCs often reflect the current status of QoS routing solutions, as many leading companies are deeply involved in the related task forces.

2.2.1 Integrated Services (IntServ)

The Integrated Services architecture [1] proposes a circuit switching like approach to ensure QoS on a per-flow basis. In this architecture, the resources are reserved for the end-to-end path for every micro-flow prior to the data transmission through RSVP

signaling [27]. The RSVP stores reservation information in a per-flow QoS table, which is checked when the router processes packets in the flow. Once a path has been reserved, all packets in the flow should be forwarded on a fixed path throughout the lifetime of the flow. The amount of resources reserved is determined by the service requirements of the individual micro-flow.

The Integrated Services architecture ensures that every micro-flow gets the correct amount of resources and that the quality of service is guaranteed. Network elements (typically a router) should support the per-flow QoS table and determine the buffer allocation and forward scheduling according to the table.

It is worth mentioning that the RSVP protocol, as a resource reservation protocol in IntServ, is not restricted in the Integrated Services architecture, though it was originally designed for the architecture. For example, its extension RSVP-TE [28] was proposed to help set up explicit MPLS Label Switched Paths (LSP) that can be purposely routed away from overloaded parts of the network.

2.2.2 Differentiated Services (DiffServ)

The Differentiated Services architecture [2] was proposed to address long-term resource allocation in a connectionless packet-switching manner. The differentiated services routers forward the packets based on the Differentiated Services Code Point (DSCP) determined and assigned to each packet by the edge routers. The DSCP has six DiffServ bits in the IP header, which reflect the service requirements of the micro-flow to which the packet belongs. The routers – other than the edge router – are provisioned with Per Hop Behavior (PHB) that corresponds to a specific DSCP, which defines how routers should forward packets (i.e. packet dropping and scheduling decisions).

Although PHB provides relative priorities and adaptive congestion avoidance using class-based queuing schema, the Differentiated Services architecture does not provide bounded values for bandwidth and delay.

2.3 Traffic Engineering

The objective of traffic engineering is to map traffic onto the network infrastructure to achieve specific performance objectives, e.g. the improvement of QoS provisioning and the efficiency of utilizing network assets, and also to achieve reliable network operation [4]. Its research is tightly related to the study of QoS routing.

2.3.1 Connection-Oriented Paradigm

MPLS [3] is a popular technology that was introduced primarily to enable efficient routing in IP networks. Since IP routers do not have to process the IP header in the MPLS packets, the processing burden on the routers is supposed to be significantly reduced. In recent years, MPLS has been attracting major attention as a protocol for enabling traffic engineering in the core IP networks.

However, the MPLS technique was not initially designed or targeted for supporting IP QoS. Though traffic engineering in the core network may solve some aspects of the service requirements, it is not really considered a solution for service differentiation. Particularly in networks where resources are limited, compared to the actual density of the QoS requests, MPLS plus traffic engineering may not be able to meet the IP QoS requirements.

2.3.2 Connectionless Paradigm

With the existing deployed base of interior gateway protocols (IGP) – such as Open Shortest Path First (OSPF) and Intermediate System-Intermediate System (ISIS) –

in mind, researchers have shown how the investment can be protected across the Internet, as well as how the configuration of link weights can be adapted to achieve the goals of traffic engineering [29]. They argue that the traditional shortest path algorithm is surprisingly effective for engineering traffic flows in a large IP core.

This approach treats traffic engineering as a network operation task, rather than a function of the routing protocols. The reconfiguration of link weights may happen once a week, or less often. Offline simulation tools can help administrators choose a suitable set of weights under different network operating conditions.

2.4 Internet Traffic Forecasting

Research [20, 21] shows that Internet traffic exhibits some daily usage patterns, and long-term traffic is predictable in a high aggregation level. A simple model, such as the Auto Regressive Integrated Moving Average (ARIMA) Model or the Kalman Filter Model, might be sufficient for long-term network forecasting.

2.5 Examples of QoS Routing Algorithms

No matter whether it is to be used for a connection-oriented or connectionless paradigm, the QoS routing algorithm is always a key part of the QoS solution. For readers interested in details on more QoS routing algorithms, please refer to [5] and the references therein.

2.5.1 PNNI

The Private Network-Network Interface (PNNI) specification [30] from the ATM Forum describes a well-known heuristic algorithm for implementing constraint-based routing. First, PNNI deploys a reliable flooding process so that every node has an up-to-date view of the network topology and capacity available on each link (the topology

database also contains other metrics, e.g. the ATM service categories that are supported, administrative link weights, and QoS parameters).

Then PNNI starts to prune the overall network topology to eliminate links that either fail to meet a requirement for the connection request or do not have sufficient capacity to accommodate the request. Based on the pruned network, the PNNI algorithm determines the shortest path, and sends a connection setup message that contains a Designated Transit List (DTL), which is a stack listing the sequence of node addresses on the specified path from the source to the destination. Note that proposed constraint-based routing algorithms for MPLS and DiffServ also employ a similar idea of explicit routes [31, 32].

If a race condition occurs, the connection setup request may find that there is insufficient link capacity available on a link in the middle. In this case, a “crank back” mechanism will be triggered, and the failed request will be sent back to the source with information explaining the reason for the failure. The source node then modifies its topology database accordingly and repeats the network pruning procedure.

2.5.2 WSP and SWP

Widest-Shortest Path (WSP) and Shortest-Widest Path (SWP) [15] are variations of the Shortest Path (SP) algorithm adopted in PNNI. Realizing that SP selects a path arbitrarily from multiple paths with the same shortest distance (sum of link costs) and that this procedure is not fair enough, Guérin et al. modified the SP slightly to improve QoS routing performance. In the case of WSP, it picks the link with the largest residual link capacity among multiple shortest paths, if available, while for SWP, it picks the link with

the shortest distance among multiple widest paths, if available. WSP is usually preferred, because the smallest link residual capacity is maximized.

2.5.3 Other Proposed Algorithms

The Minimum Interference Routing Algorithm (MIRA) proposed by Kodialam and Lakshman [17] addresses issues in more sophisticated scenarios when the selection of a path between a given source-destination pair can potentially influence many other pairs. Kodialam and Lakshman term as “interference” the phenomenon that routing a flow along a path can decrease the *maximum permissible flow* between some other source-destination pairs. They designed an algorithm to calculate the interference in order to avoid bottlenecks, and to select the best route for QoS requests.

Another, more recent work based on MIRA is Profile-Based Routing (PBR) [12]. Some points in this work actually inspired us to gather more insights into the issue of QoS routing throughout this research. PBR suggests a two-phase procedure for carrying out QoS routing:

- *The preprocessing phase.* This phase maximizes network utilization by solving a multi-commodity flow problem (a comprehensive treatment of the problem and related linear programming method can be found in [33]). The input of this phase is the traffic profile, an aggregated demand profile that records the expected flows between routers, and the output of the phase is a pre-allocated plan of the link capacity for various flows.
- *The online routing phase* is based on the pre-allocation plan obtained from the first phase. A minimum-hop path is selected between each source and destination pair, if one exists.

The traffic profile changes at run time. Once in a while, PBR creates a residual network that excludes occupied link bandwidth, and a residual profile that excludes currently routed flows. PBR then repeats the above two phases.

2.6 Challenges of IP QoS Routing

In general, QoS routing faces some primary challenges [5]:

- ❑ *Stability and Scalability.* When multiple resources are allocated and deallocated, a high frequency of state updates is required to avoid instability and route flapping [6, 7], but it does not scale well due to its high communication overhead for large networks. Unlike the cost in OSPF, the database of which is quasi-static, the cost of maintaining a synchronized QoS-database in a dynamic environment is high.
- ❑ *Robustness.* Routers always get state updates with delays, and there is no guarantee that resource information is accurate and up-to-date. Route computation and routing decisions should be robust enough to be based on imprecise states [8].
- ❑ *Routing Cost.* Processing state updates, implementing techniques related to the robustness issue, and conducting QoS routing all introduce considerable computational costs [9, 10]. In contrast, every QoS request expects highly responsive service from the QoS routing algorithms.

Chapter 3 Network and Traffic Model

Before the thesis moves on to an in-depth discussion of QoS routing, Chapter 3 will show several representations of the network and traffic demand involved in this work, and explain how simulated QoS requests are generated in Chapters 5 and 6. Moreover, this chapter will explain why QoS requirements are simplified in this thesis.

3.1 Network Model

The concepts in graph theory are fundamental to communication networks; hence, here they are considered in the context of Internet routing.

3.1.1 Terms in Graph Theory

Some terms in graph theory are frequently used for routing research to mathematically represent networks that consist of a set of abstract nodes, n , connected via a set of links, e . Note that the term in graph theory for a node is *vertex*, and the term for a link is *edge*. In this thesis, these terms are used interchangeably. Each link is indicated by an ordered pair of node numbers as a subscript. For example, e_{23} is a link from node 2 to node 3. Any bi-directional link is seen as two directional links in the model, though in normal networks they have the same link capacity.

Usually a router or a switch is represented as a node. For most ISPs, however, a node could be a POP (Point of Presence), which may contain several high capacity routers connected to each other in a metropolitan area.

A weight is associated with each unidirectional link. In the same way that the links are represented, weight w_{ij} indicates the weight of the link connecting node n_i to

node n_j . Weights are usually symmetric, which means that w_{ij} equals w_{ji} . However, this is not necessarily the case.

The pattern of links that connect nodes is called the network topology. Two network graphs are topologically equivalent if the nodes and their associated links in one graph can be moved around to yield to the other graph. Topologically equivalent graphs yield the same graphical representation. Topology only defines the nodes and interconnections between them; it has nothing to do with the physical location of routers or switches on a map.

3.1.2 Matrix Representations

Two types of matrices are used extensively in IP routing calculation. The first is the adjacency matrix A , which represents the direct connectivity for each node. A consists of a set of elements a_{ij} 's, where $a_{ij} = 1$ indicates that there is a link from node i to node j , and $a_{ij} = 0$ otherwise. In addition, A^2 represents the connectivity for all 2-hop paths, and A^n represents the connectivity for all n -hop paths. The second type is weight matrix W . W consists of weights w_{ij} for each pair of node i and j . w_{ij} equals ∞ when there is no link between node i and node j .

A network is termed *connected* if every node within it is reachable from any other node. IP routing generally deals with connected graphs only. Nevertheless, due to the QoS constraints imposed, a network could become *unconnected* in terms of one specific QoS request, or the result of QoS routing may yield an unconnected network if the bandwidth resource of one or a few links are exhausted.

3.1.3 Average Degree of a Network

The *degree* of a vertex in a graph is the number of edges that exit the vertex. The higher the degree of the vertex, the greater the level of connectivity. Thus the average degree of a graph ϕ indicates the connectivity of the graph, and it plays an important role in network design [34].

3.2 Traffic Model

Network traffic is a highly dynamic aspect of the QoS routing problem, and at this point understanding of it is limited. Fortunately, in terms of QoS requests, the traffic model can be concise.

3.2.1 Traffic Representation

Let the traffic from node i with destination node j be t_{ij} . Collecting the t_{ij} s into a matrix form yields a traffic matrix T . For instance, a four-node network has the following traffic matrix:

$$T = \begin{bmatrix} 0 & t_{12} & t_{13} & t_{14} \\ t_{21} & 0 & t_{23} & t_{24} \\ t_{31} & t_{32} & 0 & t_{34} \\ t_{41} & t_{42} & t_{43} & 0 \end{bmatrix} \quad (3-1)$$

In general, the traffic matrix is asymmetrical. The diagonal elements in T are zeros, as they do not require link capacity. Although intra-nodal traffic is important in nodal capacity planning, it is ignored here in the routing context.

The traffic matrix can also be represented in vector form when the topology information is not relevant to the discussion. Let N be the number of nodes, then the traffic demand in a four-node network can be expressed as a $N(N-1)$ -element vector. An example of such a vector for a four-node network is:

$$d = [t_{12}, t_{13}, t_{14}, t_{21}, t_{23}, t_{24}, t_{31}, t_{32}, t_{34}, t_{41}, t_{42}, t_{43}] \quad (3-2)$$

3.2.2 Connection-Oriented Assumption

When QoS requirements are considered in the Internet context, the behavior of the IP network becomes more like connection-oriented, since a certain amount of resources have to be assigned to specific QoS requests. To simplify our description of the scenarios, connection requests will be used as examples for the rest of the thesis. Within our context, the situation is similar for connectionless requests.

3.2.3 Generation of QoS Requests

For each source-destination pair (i,j) , QoS requests arrive according to a uniform distribution (in time) of intensity λ_{ij} . The duration of every QoS request is assumed to be infinite.

In QoS routing, the sequence of requests is actually more important than the exact time of each request. For example, there is no difference between the following two sets of requests in our consideration, though their exact arrival times are different:

Set 1		Set 2	
Arrival Time	Request	Arrival Time	Request
5:18am	r_1	5:18am	r_1
5:21am	r_2	5:19am	r_2
5:25am	r_3	5:22am	r_3
5:26am	r_4	5:23am	r_4
5:30am	r_5	5:30am	r_5

Table 3-1. Two Equivalent Sets of QoS Requests

The bandwidth requirement of each request is randomly selected from the discrete set $\{1, 2, 3\}$ of bandwidth unit (1 Mbps), in the same way that Kodialam et al. and Suri et al. did in their simulations [17, 12]. In our simulations, we chose discrete numbers and set the average bandwidth requirement to a small number (2 units, on average) in order to avoid the packing issue. Packing is still an open topic for research [35, 36]. When the results of the simulation are interpreted, it should be noted that all QoS routing algorithms will degrade slightly when the average bandwidth requirement for each request increases. The packing problem can become severe as the average bandwidth requirement approaches the order of the link capacities. Fortunately, this kind of situation does not usually occur in the real world.

The traffic model adopted in this research work is simple, admittedly, but it maintains the focus on two factors: the sequence of requests and the aggregated bandwidth requirement. Below, a sample sequence of requests is denoted as a *traffic trace*, e.g. set 1 in Figure 3-1. Multiple experiments were conducted with exactly the same traffic trace, in order to compare different QoS routing schemes. In addition, at least 20 randomly generated traffic traces were tested in every single experiment setting.

3.3 Simplifications to QoS Constraints

Typical QoS constraints include delay, packet loss, and bandwidth. A coupling between the routing decision and the performance of assigned QoS requests could be removed. The rationale is described below.

3.3.1 Simplify Delay and Packet-Loss Constraints

The packet transmission latency is affected by many factors. It is common to model the packet-switching network on queuing theory. Based on queuing theory, a

latency formula can be derived from the hypothesis that each queue behaves like an M/M/1 queue of packets [19]:

$$l_e(x_e) = \frac{x_e}{c_e - x_e} + D_e x_e \quad (3-3)$$

D_e is the processing and propagation delay on edge e , and c_e is the link capacity of edge e . Both D_e and c_e could be considered as constant. x_e is the total bandwidth required for all flows on edge e . Hence, the latency l_e on edge e is a function of x_e . It shows that the placement of a new QoS request will affect the delay of existing assigned flows on the same link. There is a disinclined coupling between the routing decision and the delay constraint, which contributes to the complexity of the QoS routing problem in general.

There is a similar coupling for packet-loss constraints. If the desired packet-loss ratio is given, the throughput a link can deliver increases nonlinearly with the link capacity [34, pp. 388]. In other words, if the link capacity is fixed, the packet-loss rate is a function of carried flows on the link. Therefore, the placement of a new QoS request will affect the packet-loss rate of the existing assigned flows on the same link.

The problem of finding routes that satisfy these constraints is again NP-complete. Therefore, the upper bounds of delay and loss for each link are adopted in this thesis to simplify the delay and packet-loss constraints. The simplification is justified by the fact that the measured loss probabilities and delay for the same priority flow on different routers are of a similar order [18]. This simplification is relatively conservative. It assumes that the network is not overloaded, and the delay and loss constraints of flows should be met. Hence, the translated number of hops along a route is considered to be the delay and packet-loss constraint.

3.3.2 Simplify Bandwidth Constraint

The bandwidth required can be calculated according to burst size, buffer size, flow peak/average ratio, and so on. Equivalent bandwidth is a feasible analytical technique for estimating the bandwidth requirement of variable-rate flows by combining two separate models, namely the fluid flow model and the statistical multiplexing gain model [37, 38].

Note that a traffic parameter control is necessary in order to deliver QoS over a network. There must be a mechanism, for example traffic policing and shaping, to check whether incoming traffic conforms to the specific contract or agreement [26].

3.3.3 Effect of Self-Similar Traffic

Self-similar traffic has some negative consequences for routing. Due to its extreme variability, for a certain level of QoS objectives (e.g., the probability of buffer overflow, connection blocking or a bound on latency), the self-similar traffic model requires more bandwidth for transportation, and hence lowers the ratio of the carried mean load divided by the required link/network capacity.

The simplification of the bandwidth requirement should already include the effect of self-similar traffic. However, it is always a good idea to be aware of this issue when analyzing QoS routing algorithms.

3.4 Remarks

It is critical that QoS constraints be simplified before the routing of QoS requests begins, although the method of simplification may vary. If the coupling between the routing decision and the performance of the assigned requests is removed, the QoS routing problem can be simplified to a manageable level.

It usually makes sense when the complexity of a traffic model is compatible with the complexity of a routing solution. Otherwise, it is difficult to capture the principles of the QoS routing problem and obtain a complete picture of it.

Chapter 4 Performance Evaluation of QoS Routing

This chapter illustrates the performance evaluation method adopted throughout this thesis. The key point is that instead of a linear objective function, a nonlinear one is necessary for solving the QoS routing problem.

4.1 Linear Objective Function

In general, the cost to a given link is an increasing function of the amount of traffic carried by the link. This function may be referred to as link utilization, average delay, or loss probability on the link when the link is not overloaded.

If the QoS routing problem is seen as an optimization problem, the objective function could be the sum of the link costs for all the links in a network. Let $x_{e,k}$ denote the k th flow on edge e . E is the set of all edges. K is the set of all traffic flows. The link cost induced by the flow is either a linear or nonlinear function, $f_{e,k}(x_{e,k})$, increasing in $x_{e,k}$. The general form of the objective is:

$$\text{minimize } \sum_{e \in E} \sum_{k \in K} f_{e,k}(x_{e,k}) \quad (4-1)$$

Both the most popular intra-domain routing protocols, OSPF and IS-IS, implement Equation 4-1 in a linear fashion, i.e. $f_{e,k}(x_{e,k}) = w_e x_{e,k}$, where w_e is a constant. No matter how the link weights are configured (except for dynamic configuring at real-time), it is well known that the linear objective function will inevitably introduce an overloaded section or bottleneck when the given traffic matrix does not match the

distributed network resources in a balanced manner. In fact, this is one of the major concerns with traffic engineering.

4.1.1 The Inequality of QoS Requests

Apart from the disadvantage of the linear objective function mentioned above, there is another drawback in terms of treating QoS requests equally among all the source-destination pairs.

A simple example is shown below: Figure 4-1 shows a 15-node artificial topology often used in QoS routing research [12, 17].

Figure 4-1. Topology of Network #0

To illustrate the issue concisely, it is not necessary to elaborate a realistic traffic matrix that may blur the point we are trying to make. Instead, traffic T_0 (in Kbps, see Figure 4-2) is used. Link weights are configured in a manner that is inversely proportional to link capacity, so the cost of each 1 Kbps traffic on a 4800 Kbps link is 1, while the cost on a 1200 Kbps link is 4.

Suppose the flows within the traffic matrix T_0 are splittable. In that case, the routing problem in this example can be formulated as a linear multi-commodity flow problem [33]. In the situation where it is impossible to satisfy all link capacity constraints, even the best solution may reject some bandwidth requirements. Additional edges (excess edges) can therefore be added in the network, so that the multi-commodity flow problem always has a feasible solution. A high weight value is associated with the excess edges to distinguish between the network edges and the excess edges. In addition, the link capacities of the excess edges between each ingress-egress pair are set to be infinite.

$$T_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4800 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4800 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4800 & 0 & 0 & 0 & 0 & 0 & 4800 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 4800 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 4800 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 4-2. Traffic Matrix of Network #0

An open source implementation of the minimum cost multi-commodity flow algorithm is publicly available as a PPRN package [22]. It is used in this thesis to calculate the optimized routing solution. The package was originally designed for solving multi-commodity network flow problem with nonlinear objective function and with linear

side constraints. But it also can be used as a general package for solving a high variety of network flow problems, including multi-commodity flow problem with linear objective function and with or without side constraints.

Table 4-1 shows the result of the best solution with an overall objective function value of 1,580,400. This shows that the network cannot meet all traffic demands in light of the link capacity constraint, and some requests are rejected.

It has now been established that the network cannot handle traffic greater than 14,400 Kbps (see the total accepted bandwidth in Table 4-1) for these six source-destination pairs. Otherwise, there should be a better solution with less overall objective function value than the one in Table 4-1, which conflicts with the definition of the optimization.

Source Node	Destination Node	Explicit Route	Accepted Bandwidth	Cost/Kbps	Cost
1	13	1_3_6_10_13	1200K	10	12000
5	9	5_2_3_7_9	1200K	7	8400
5	9	5_2_3_6_10_9	1200K	8	9600
4	2	4_1_2	1200K	8	9600
4	2	4_3_2	1200K	5	6000
4	2	4_9_7_3_2	1200K	10	12000
4	2	4_8_9_10_6_3_2	1200K	15	18000
5	15	5_2_3_6_10_14_15	1200K	9	10800
5	15	5_12_13_14_15	1200K	13	15600
12	4	12_5_2_1_4	1200K	13	15600
12	4	12_11_6_3_4	1200K	10	12000
14	8	14_15_9_8	1200K	9	10800
Total Accepted Bandwidth			14400K		140400
1	13	Rejected	3600K	100	360000
5	9	Rejected	2400K	100	240000
5	15	Rejected	2400K	100	240000
12	4	Rejected	2400K	100	240000
14	8	Rejected	3600K	100	360000
Total Rejected Bandwidth			14400K		1440000
Overall Objective Function Value					1580400

Table 4-1. 1st Solution of the Traffic Matrix T_0

After decreasing the traffic matrix T_0 to a small extent and solving the problem a few more times, we found another possible solution with an overall cost of 1,584,000, which is shown in Table 4-2. The motivation for doing this was to verify whether the rejected requests would be more evenly distributed among the source-destination pairs than the optimal solution. The total accepted bandwidth in Table 4-2 is 14,400 Kbps.

Note that both solutions have the same total accepted traffic for different source-destination pairs, but the one with the lowest cost of all the pairs is unfair because two pairs of 3,600 Kbps are rejected and one pair is totally accepted in Table 4-1, while the one that accepts traffic requests evenly has a slightly higher cost. Which is better, in practice?

Source Node	Destination Node	Explicit Route	Accepted Bandwidth	Cost/Kbps	Cost
1	13	1_3_7_10_13	1200K	10	12000
1	13	1_2_3_6_11_13	1200K	11	13200
5	9	5_2_3_7_9	1200K	7	8400
5	9	5_2_3_7_10_9	1200K	8	9600
4	2	4_3_2	1200K	5	6000
4	2	4_9_7_3_2	1200K	10	12000
4	2	4_8_9_10_6_3_2	1200K	15	18000
5	15	5_2_3_6_10_14_15	1200K	9	10800
5	15	5_12_13_14_15	1200K	13	15600
12	4	12_5_2_1_4	1200K	13	15600
12	4	12_11_6_3_4	1200K	10	12000
14	8	14_15_9_8	1200K	9	10800
Total Accepted Bandwidth			14400K		144000
1	13	Rejected	2400K	100	240000
5	9	Rejected	2400K	100	240000
4	2	Rejected	1200K	100	120000
5	15	Rejected	2400K	100	240000
12	4	Rejected	2400K	100	240000
14	8	Rejected	3600K	100	360000
Total Rejected Bandwidth			14400K		1440000
Overall Objective Function Value					1584000

Table 4-2. 2nd Solution of the Traffic Matrix T_0

One thing we learned from this example is that a routing algorithm based on a linear objective function can achieve good network resource utilization or low network-

wide cost, but may unfairly punish some customers by rejecting a disproportionate share of their flows.

Overall, if linear costs are associated with the excess edges, the optimization pursues the most efficient solution. However, if nonlinear cost functions are applied to the excess edges, the optimization may adjust its objective between efficiency and fairness.

4.2 Nonlinear Objective Function

This section demonstrates the effectiveness of designing a monotonically increasing nonlinear objective function to combine two major objectives in IP routing, namely minimizing average hop count and minimizing the maximum link utilization.

4.2.1 Minimize Average Hop Count

The most obvious objective is to provide routing service with overall low network resource consumption. If a QoS request can be served by multiple feasible routes (for example, a 3-hop route or a 4-hop route), the short route is usually preferred because it consumes fewer router resources (i.e., queues/buffers and processor power) and fewer link capacities. In other words, this objective can be rephrased as minimizing the average number of hops that those routes traverse, as long as their QoS requirements are satisfied.

A linear version of Equation 4-1 effectively addresses our interest in consuming fewer overall network resources:

$$\text{minimize } \sum_{e \in E} \sum_{k \in K} w_e x_{e,k} \quad (4-2)$$

where w_e is a positive constant.

In Equation 4-2, w_e is the weight of edge e , and $x_{e,k}$ is the bandwidth of the k th flow on edge e . K represents the set of all flows in a network.

If the value of w_e in Equation 4-2 equals 1 for all edges in the network, then $\sum_{e \in E} \sum_{k \in K} w_e x_{e,k}$ divided by the total volume of injected traffic $\sum_{i \neq j} t_{ij}$ yields the flow-weighted average hop count of the network. This flow-weighted average hop count, instead of the simple average hop count, will be referred to as the *average hop count* for the remainder of this thesis.

4.2.2 Minimize Maximum Link Utilization

The other primary objective is to avoid the situation where parts of the network are saturated, while other parts are under-loaded. The main reasons for supporting this objective include:

- Spare bandwidth available at various parts of the network is useful for accommodating unpredictable traffic requests, if the network resources are used evenly;
- In the case of a single link or node failure, smaller amounts of traffic are interrupted and detoured;
- The latency measurement on the Internet suggests that unbalanced traffic distribution over a network may dramatically induce extra forwarding delay if the major network routing concern is to route traffic over the path with minimum packet latency.

Figure 4-3, which was adopted from the *Internet Core Router Test* report [39], demonstrates that the latency occurring within a core router increases as the traffic load

increases on its OC-48 port. Three load levels – minimum, average and maximum – on the OC-48 ports are tested for each router, namely, Charlotte’s Aranea-1 (AraneOS 2.0), Cisco 12416 (IOS 12.0(14)SX), Foundry NetIron (7.1.00) and Juniper M160 (4.2 R2.4 / 4.2E). This kind of behavior is not surprising, according to Equation 3-3.

Figure 4-3. Core Router Latency Measurement with Internet Mix Traffic

Note that the objective to avoid overloading parts of the network is not valid for all incoming QoS requests. There may not be alternative routes for some requests. With the possible exception of heavily loaded conditions, there will generally be multiple feasible solutions or routes for most QoS requests. Hence, this objective should be considered when a QoS routing algorithm arranges these multiple-solution requests.

The general form of the objective function, in terms of link utilization, could be Equation 4-3. If w_e equals one over the link capacity of the edge e , then the equation is concreted as minimizing the maximum link utilization.

$$\text{minimize} \max_{e \in E} \left(w_e \cdot \sum_{k \in K} x_{e,k} \right) \quad (4-3)$$

where w_e is a positive constant.

4.2.3 Other Objectives

There are other objectives in routing, such as maximizing overall flows carried by the network; minimizing mean link utilization; or minimizing the rejected QoS requests. Those objectives are consistent with the two objectives discussed above. If the network resources are utilized efficiently, based on the two previous objectives, generally speaking these additional objectives should also be served well.

4.2.4 Combination of Two Objectives

However, the two objectives – minimizing average hop count and minimizing maximum link utilization – do not lead to the same optimization solution. In many cases, they conflict with each other.

Moreover, the function of minimizing the maximum link utilization can be overly sensitive to a few bottleneck links, which are hard to avoid. For example, a critical link with large capacity, which connects together two parts of the network, has to carry a large amount of traffic under any routing solution. An ideal routing algorithm is not supposed to penalize this structurally necessary traffic. In another example, when the overall network load is light, it is not worth forcing traffic to take a very long alternative route in order to reduce the link utilization of a critical link from 40% to 35%.

In order to deal with these issues, Bonaventure et al. [18] have proposed a nonlinear objective function that forms a compromise between the two previous

objectives. However, it remains up to the ISP to decide which objective is more important, according to network operation status and specific requirements.

More specifically, the objective function is shown in Equation 4-4:

$$\text{minimize } \sum_{e \in E} \left(w_e \cdot \sum_{k \in K} x_{e,k} \right)^n, n \in [1, \infty) \quad (4-4)$$

where w_e is a positive constant.

When $n=1$ the objective defined in Equation 4-4 reduces to the one in 4-2. As n increases, the objective defined by Equation 4-4 increasingly desires solutions that adhere to the objective defined by Equation 4-3, since the factor with the maximum $w_e \cdot \sum_{k \in K} x_{e,k}$ is the dominant factor in this equation.

The link cost $\left(w_e \cdot \sum_{k \in K} x_{e,k} \right)^n$ is illustrated in Figure 4-4 where w_e equals $1/c_e$. c_e is the link capacity of edge e . As the figure shows, the cost of using a link increases quickly with utilization. The curvature of the curve is solely controlled by n , defined in Equation 4-4. When link utilization is high, the higher the n value, the more sensitive the link cost to flows x_e carried on the link. On the other hand, when link utilization is low, the higher the n value, the less sensitive the link cost to flows x_e carried on the link.

The cost of a network-wide routing solution is now a combination of nonlinear link costs, and therefore the resulting optimization problem is nonlinear and cannot be solved easily. The resulting optimization problem is regarded as a network flow problem with the non-linearity of the objective function 4-4.

Figure 4-4. Nonlinear Link Cost

Fortunately, the objective function 4-4 is a differentiable convex of $x_{e,k}$, as shown in Figure 4-4. Consequently, the optimal solution can be calculated by the general gradient projection method [40, 41]. More in-depth discussion about the differentiable convex cost function and its optimization can be found in [19].

In essence, the general gradient projection method is an iterative method. It starts from an initial feasible solution, and at each iteration it finds the minimum first derivative of the objective function. The method then shifts part of the flow from the expensive paths to the inexpensive paths, so that it gradually improves our objective function.

In this thesis, the value of objective function 4-4 is regarded as being the level of network-wide performance. Moreover, for a given traffic demand matrix T , the value of the objective function for any routing algorithms is bounded by an optimal value – i.e., the lowest value, which can be computed if the traffic flow defined in T is splittable.

4.3 Remarks

IP QoS routing is nonlinear by its nature. If networks are never saturated, either partially or wholly, the shortest path algorithm is a good enough linear approximation. Otherwise, the behavior of the saturated network elements is nonlinear. Accordingly, a nonlinear convex function is a good choice for the objective function in optimization and for the metric to evaluate network-wide performance.

Chapter 5 Simulations of the State-Dependent Routing Mechanism

The purpose of this chapter is to demonstrate the underlying disadvantages of state-dependent routing algorithms based on simulations. This chapter will first describe the simulation environment and some typical routing algorithms. The simulation results will then be analyzed, followed by a discussion of the disadvantages of state-dependent routing mechanisms.

5.1 Simulation

To the best of our knowledge, the proposed state-dependent routing algorithms have not been compared with regard to the performance metric defined in Equation 4-4, and that will be done in this chapter. The routing algorithms implemented in the simulation include SP, SWP and WSP.

5.1.1 Simulation Setup

The simulations were performed on a PC with Intel Celeron 500MHz CPU and 128Mbyte memory, running on the Red Hat Linux version 8.3 operation system. The simulation code was first implemented in TCL script [23] based on NS-2 version 2.1b8a [24] and the MNS patch version 2.0 [42] in order to utilize existing MPLS features, namely the CR-LDP protocol. Later, some simulations were conducted purely on TCL version 8.3.4 to accelerate the simulation process.

The execution time of the simulations was quite long. For example, it took more than an hour to plot one line of WSP performance according to one traffic trace. For each

scenario, 20 traffic traces were randomly generated and were then fed into different routing algorithms. The mean value, the standard deviation and the confidence interval were calculated in the results. Normal distribution is assumed, with the chosen level of confidence at 95%. In simulation results, the confidence intervals of objective function value are relatively small. For example, in network #1, when the traffic demand is 99.4% of $T_{a,1}$, the average of the objective function value of SWP is 211,977, with a confidence interval of 211,142 ~ 212,811, and the interval $212,811 - 211,142 = 1,669$ is below 0.8 % of 211.977.

Due to the complexity of these routing algorithms, self-developed code was carefully tested under some extreme test cases. In addition, the code was fully verified to reproduce all the results presented in the literature [12] to ensure it performs exactly the same way as other simulation tools did in experiments designed by other researchers.

The PPRN package [22] has an extension to solve the network flow problem with a nonlinear objective function. Part of the C language code regarding the nonlinear objective function and its differential coefficient was modified to accommodate Formula 4-4 in simulations. An example of this program is attached in Appendix 1.

5.1.1.1 Value Setting

Two network topologies, network #1 and #2, were selected for the simulations. They were real topologies obtained from an ISP in North America. Network #1 is a 10-node network with 16 bi-directional links that are either OC-12 or OC-48 (see Figure 5-1), while network #2 is a 20-node network with 37 bi-directional links, including OC-3, OC-12 and OC-48 (see Figure 5-2). The traffic demands used in the simulations were collected from the ISP as well. Using these two real-world networks gives us realistic

insights into the routing issue. In addition, the study was conducted on networks of different sizes to help us better understand the simulation results.

Figure 5-1. Topology of Network #1

Figure 5-2. Topology of Network #2

The QoS requests used in the simulations were generated as described in section 3.2.3, until a certain level of traffic demand was reached. The traffic pattern – the ratio of traffic among the different source-destination pairs – was fixed in the simulations according to the actual measurement from the ISP, while the traffic volume might have

been scaled up or down. To make our diagrams more readable, the traffic demand is normalized to the maximum possible value – i.e., $T_{a,1}$ for network #1 (in Mbps, Figure 5-3) and $T_{a,2}$ for network #2 (in Mbps, Figure 5-4) – that the network can handle. The first subscript of T denotes the specific traffic matrix, and the second subscript denotes the network number.

$$T_{a,1} = \begin{bmatrix} 0 & 307 & 171 & 339 & 295 & 20 & 58 & 237 & 206 & 151 \\ 307 & 0 & 521 & 747 & 540 & 126 & 128 & 443 & 352 & 365 \\ 171 & 521 & 0 & 490 & 418 & 10 & 174 & 402 & 387 & 169 \\ 339 & 747 & 490 & 0 & 847 & 61 & 144 & 538 & 445 & 397 \\ 295 & 540 & 418 & 847 & 0 & 61 & 110 & 330 & 239 & 327 \\ 20 & 126 & 10 & 61 & 61 & 0 & 10 & 61 & 61 & 20 \\ 58 & 128 & 174 & 144 & 110 & 10 & 0 & 93 & 78 & 68 \\ 237 & 443 & 402 & 538 & 330 & 61 & 93 & 0 & 143 & 295 \\ 206 & 352 & 387 & 445 & 239 & 61 & 78 & 143 & 0 & 644 \\ 151 & 365 & 169 & 397 & 327 & 20 & 68 & 295 & 644 & 0 \end{bmatrix}$$

Figure 5-3. Traffic Matrix of Network #1

$$T_{a,2} = \begin{bmatrix} 0 & 85 & 218 & 22 & 122 & 41 & 28 & 14 & 107 & 17 & 42 & 241 & 27 & 19 & 6 & 169 & 146 & 52 & 103 & 210 \\ 85 & 0 & 194 & 22 & 110 & 37 & 28 & 14 & 99 & 17 & 7 & 231 & 27 & 19 & 6 & 144 & 122 & 74 & 59 & 124 \\ 218 & 194 & 0 & 114 & 371 & 91 & 779 & 90 & 260 & 45 & 21 & 531 & 80 & 58 & 19 & 315 & 250 & 124 & 168 & 384 \\ 22 & 22 & 114 & 0 & 11 & 11 & 0 & 0 & 22 & 0 & 0 & 66 & 0 & 0 & 0 & 66 & 66 & 11 & 6 & 66 \\ 122 & 110 & 371 & 11 & 0 & 124 & 14 & 7 & 120 & 8 & 4 & 348 & 13 & 9 & 4 & 286 & 275 & 59 & 37 & 297 \\ 41 & 37 & 91 & 11 & 124 & 0 & 14 & 7 & 48 & 8 & 4 & 103 & 13 & 9 & 39 & 66 & 55 & 22 & 32 & 78 \\ 28 & 28 & 779 & 0 & 14 & 14 & 0 & 0 & 28 & 0 & 0 & 84 & 0 & 0 & 0 & 84 & 84 & 14 & 6 & 84 \\ 14 & 14 & 90 & 0 & 7 & 7 & 0 & 0 & 14 & 0 & 0 & 44 & 0 & 0 & 0 & 44 & 44 & 7 & 7 & 44 \\ 107 & 99 & 260 & 22 & 120 & 48 & 28 & 14 & 0 & 17 & 7 & 282 & 27 & 117 & 6 & 210 & 458 & 59 & 68 & 232 \\ 17 & 17 & 45 & 0 & 8 & 8 & 0 & 0 & 17 & 0 & 0 & 48 & 0 & 0 & 0 & 48 & 48 & 8 & 8 & 48 \\ 42 & 7 & 21 & 0 & 4 & 4 & 0 & 0 & 7 & 0 & 0 & 21 & 0 & 0 & 0 & 21 & 21 & 4 & 4 & 21 \\ 241 & 231 & 531 & 66 & 348 & 103 & 84 & 44 & 282 & 48 & 21 & 0 & 80 & 58 & 19 & 382 & 316 & 135 & 176 & 602 \\ 27 & 27 & 80 & 0 & 13 & 13 & 0 & 0 & 27 & 0 & 0 & 80 & 0 & 0 & 0 & 80 & 80 & 13 & 13 & 1169 \\ 19 & 19 & 58 & 0 & 9 & 9 & 0 & 0 & 117 & 0 & 0 & 58 & 0 & 0 & 0 & 58 & 58 & 9 & 9 & 58 \\ 6 & 6 & 19 & 0 & 4 & 39 & 0 & 0 & 6 & 0 & 0 & 19 & 0 & 0 & 0 & 19 & 19 & 4 & 4 & 19 \\ 169 & 144 & 315 & 66 & 286 & 66 & 84 & 44 & 210 & 48 & 21 & 382 & 80 & 58 & 19 & 0 & 101 & 153 & 149 & 235 \\ 146 & 122 & 250 & 66 & 275 & 55 & 84 & 44 & 458 & 48 & 21 & 316 & 80 & 58 & 19 & 101 & 0 & 89 & 142 & 170 \\ 52 & 74 & 124 & 11 & 59 & 22 & 14 & 7 & 59 & 8 & 4 & 135 & 13 & 9 & 4 & 153 & 89 & 0 & 90 & 111 \\ 103 & 59 & 168 & 6 & 37 & 32 & 6 & 7 & 68 & 8 & 4 & 176 & 13 & 9 & 4 & 149 & 142 & 90 & 0 & 151 \\ 210 & 124 & 384 & 66 & 297 & 78 & 84 & 44 & 232 & 48 & 21 & 602 & 1169 & 58 & 19 & 235 & 170 & 111 & 151 & 0 \end{bmatrix}$$

Figure 5-4. Traffic Matrix of Network #2

These two traffic matrices are an approximation of a real measurement taken over several days. As will be discussed later, in section 6.5, it is difficult for the ISPs to measure connectionless IP traffic in a matrix form. This partly explains why these two matrices are symmetrical.

With regard to the network-wide performance evaluation, the value of the order n in Equation 4-4 is set to 2, and w_e equals to $1/c_e$.

5.1.2 Simulation Results

The performance of SP, WSP and SWP is compared with respect to the objective function value as traffic demand increases (Figure 5-5 and 5-6).

Figure 5-5. Network Performance as Traffic Changes from Zero to $T_{a,1}$
Network #1, $n=2$

In Figures 5-5 and 5-6, the optimal line represents the lower bound of the objective function value, which was computed as described in section 4.2.4. As the

simulation results will show in Chapter 6, this boundary is actually very tight. For any routing algorithms, however, in order to keep to the optimal line, some rearrangement of the traffic is unavoidable (more details on rearrangement appear in section 5.2.2). From our point of view, this elaborate rearrangement is not worth the effort, simply because the real-time traffic demand fluctuates highly over a small time scale [25]. In other words, the lower bound in the figures is ideal but very expensive and unnecessary.

For network #1, the performance of SP and WSP is surprisingly good, as diagrammed in Figure 5-5, especially WSP, which is very close to the optimal solution. This result prompts us to reexamine the widespread criticism of SP and WSP in the literature. At least in this particular topology, network #1, and under traffic demand $T_{a,1}$, these algorithms are able to make near optimal routing decisions with very simple routing mechanisms.

Figure 5-6. Network Performance as Traffic Changes from Zero to $T_{a,2}$
Network #2, $n=2$

One possible reason why SP and WSP succeed in Figure 5-5 but not in Figure 5-6 could be related to the network size. Network #1 has fewer alternative paths than network #2. It is more difficult to create interference in network #1 than in network #2, since the size of network #1 is relatively small. In other words, a source-destination pair in network #1 usually makes fewer hops than in network #2. The chance that routing a flow along a path will decrease the maximum permissible flow between some other source-destination pairs is low in network #1.

Based on this understanding, the performance of network #2 in Figure 5-6 is not surprising. The size of network #2 is relatively large, and the chance that interference will occur in it is relatively high. In this case, the SP and WSP lines move away from the optimal line, and the gap between SP and WSP is increased, partly because they have more choices regarding alternative routes in network #2, and the difference between the decision-making logic of SP and WSP is now more significant.

In general, as the traffic level increases, all state-dependent algorithms gradually leave the optimal line, which means their network-wide performance degrades gradually. In both networks #1 and #2, WSP has the best overall performance of the three QoS routing algorithms.

Figures 5-7 and 5-8 show the confidence interval of the objective function value over 20 trials. When the traffic load is low, these errors are not noticeable. However, when the traffic level approaches the maximum capacity of the network, errors become visible in the diagram. The reason for this is that routing algorithms become more sensitive to the sequence of QoS requests when free resources (e.g. free bandwidth on the links) decrease. More detail on this sensitivity phenomenon is included in section 5.2.2.

Figure 5-7. Performance with 95% Confidence Interval as Traffic Changes from Zero to $T_{a,1}$, Network #1, $n=2$

Figure 5-8. Performance with 95% Confidence Interval as Traffic Changes from Zero to $T_{a,2}$, Network #2, $n=2$

In relative terms, WSP has the smallest level of errors in both Figures 5-7 and 5-8, because WSP has the best performance metric, which implies that it has a shorter average hop count and lower maximum link utilizations. As a result, WSP usually provides a greater level of free bandwidth than the other two algorithms. WSP also gives rise to a more even distribution of free bandwidth over the network than the other two. Hence, in 20 trials, WSP was less sensitive to different sequences of QoS requests.

To further explain the meaning of the objective function value, let us look separately at two other diagrams: the average hop count (Figure 5-9) and the maximum link utilization (Figure 5-10). These two figures can be used to make a concrete interpretation of Figure 5-6 in terms of the network performance of network #2. Similar results obtained from network #1 are omitted.

Figure 5-9. Average Hop Count as Traffic Changes from Zero to $T_{a,2}$
Network #2

**Figure 5-10. Maximum Link Utilization as Traffic Changes from Zero to $T_{a,2}$
Network #2**

Recall that the average hop count in routing can be minimized by solving a linear multi-commodity flow problem; therefore a lower bound in the average hop count, the optimal line, can be plotted confidently in Figure 5-9.

Figure 5-9 illustrates the average hop count for three state-dependent routing algorithms. Obviously, the two curves of SP and WSP start from the optimal point, with an average of 1.83 hops per route, and move up as the traffic level increases when the traffic demand is about 76%. Both these algorithms always pick the shortest route first. Again because WSP assigns bandwidth more evenly than SP, WSP can retain the average hop count at a lower level when traffic load grows, but SP cannot do so.

SWP always picks the widest path first, so its average hop count is higher than the other two algorithms. It seems that SWP does not provide any benefits in terms of

minimizing average hop count, but the next figure demonstrates its value.

It is known that as n approaches ∞ , the objective function 4-4 tends to find a solution that satisfies the objective defined in Equation 4-3; i.e., minimizing the maximum link utilization in a network. In fact, when $n \geq 5$, the optimization solutions we obtained from various simulations are almost the same. The optimal value of maximum link utilization converges fast as n increases. Consequently, the values of the maximum link utilization computed when n equals 5 are treated as the optimal value in Figure 5-10.

Compared with other algorithms, the advantage of SWP can be seen in Figure 5-10: SWP is the most successful in terms of minimizing the maximum link utilization, followed by WSP and then SP. When the whole network is saturated, SWP finally reaches 100 percent at the end.

In summary, there is a trade-off between the two routing objectives detailed in sections 4.2.1 and 4.2.2. Some routing algorithms are good at minimizing hop count; others are good at minimizing link utilization. Developing a routing algorithm to satisfy both objectives based solely on the current network state (e.g. topology, link weights, and free bandwidth on each links) is challenging.

5.2 Disadvantages of the State-Dependent Mechanism

The state-dependent mechanism is ideal in the sense that no traffic demand has to be known in advance; hence, fewer restrictions are applied to routing algorithms. For instance, forecasting Internet traffic is risky and could make a simple routing algorithm very complicated and, in all likelihood, hard to maintain. Nevertheless, state-dependent QoS routing algorithms have some underlying disadvantages.

5.2.1 QoS Routing Challenges in the State-Dependent Routing Context

Traditionally, Internet routing holds one principal philosophy – simplicity. The evolution of the Internet and its continuous success suggest that simple mechanisms could survive better in complex environments rather than a well-designed sophisticated mechanism. Shortest Path routing satisfies the basic requirements and has been widely deployed. As a result, it has been a fundamental infrastructure for the Internet, even if it has some obvious flaws.

However, this philosophy encounters barriers when dealing with the QoS routing problem. In essence, a state-dependent mechanism faces all the challenges mentioned in section 2.6, no matter whether is designed as Pre-Computation Routing [13, 14] or On-Demand Routing [15, 16, 17]. In particular, pre-computing the paths for all possible QoS requirements [13] is extremely processor- and memory-consuming, as a state-dependent mechanism has no idea what the future demands will be.

5.2.2 Rearrangement Dilemma

Traffic rearrangement is a tricky aspect of QoS routing. Figure 5-11 provides an example that illustrates the issue. Because of the assumption in section 3.2.2, MPLS Label Switched Paths (LSPs) are adopted in the example to express the idea of rearrangement, yet the idea itself is broadly valid for both connection-oriented and connectionless situations.

Suppose there are nine routers and ten connections between them, and their link capacities (in one bandwidth unit) are marked on the links (see Figure 5-11). A series of LSP setup requests arrive sequentially, with specified source and destination router names and required bandwidth (also in one bandwidth unit).

Figure 5-11. Three LSP Requests and Their Ideal Routing Solution

In this example, the shortest path is first adopted for each request. In the meantime, we prefer to reserve their bandwidth requirement as evenly as possible. The arrowed curves in Figure 5-11 show the best routing solution. The reason that request #2 goes through the right path instead of the middle path is that the right path is shorter and saturates fewer links than the middle path does.

Figure 5-12. Two More LSP Requests and Their Ideal Routing Solution

Take Figure 5-12 as another example. Assume that two more LSP requests, LSP requests #4 and #5, arrive after a short period of time. The best routing solution, as described in Figure 5-12, is desired. Note that only in this routing solution, two additional LSP requests can be accepted by making request #2 traverse a longer route from A to E. Otherwise, they will be rejected, due to the fact that the link capacity on link FE is exhausted in Figure 5-11.

The problem is that the routing algorithm has to rearrange LSPs in a relatively non-disruptive manner. To take this example one step further, it is ascertained that periodical rearrangement is necessary to maintain the optimal level of the routing solution. A rearrangement takes place between Figure 5-11 to 5-12. In Figure 5-12, if requests are changed – for example, LSP requests #3 and #5 are released in 20 minutes – the network becomes sub-optimal again when the two LSP release requests take effect, and another rearrangement is required.

Rearrangement is not a desirable practice in routing. Customers experience disruption when there is rearrangement, and there are also sudden changes in packet latency and its jitter. Alternatively, additional computation resources and signaling overhead are introduced in order to minimize the disruption brought about by rearrangement.

There is another way to look at this example. The rearrangement in Figure 5-12 can be avoided if five LSP requests arrive in the following sequence: #3, #4, #5, #1 and #2. In other words, a routing algorithm can accept all five requests if they are in a certain order, but will block some of them if they are in another order. Generally speaking, most QoS routing algorithms are *sensitive* to the order of incoming QoS requests.

Consequently, the performance of most QoS routing algorithms is sensitive to the order of connection attempts.

There is a *rearrangement dilemma* in the case of state-dependent mechanisms. Suppose a state-dependent routing algorithm can approximate an optimal point, according to the current state. After a few minutes, when new LSP requests arrive, the routing algorithm has a high chance of rearranging existing user flows in order to keep the same level of routing performance. On the other hand, if fewer rearrangements are made, it has to stay away from the optimal points. The dilemma partially stems from not knowing the future traffic demand, which is a focus of this thesis.

5.3 Remarks

To better utilize the network resources, one idea is to approach the optimal points dynamically based on the changing traffic demand at real-time. The underlying barrier to this notion is traffic rearrangement, which means some flows have to be rearranged in terms of explicit routes and their assigned bandwidth. Rearrangement causes service disruption and significant signaling overhead. The cost of rearrangement increases dramatically as the frequency of the rearrangements increases.

The simulation result displayed in Figures 5-5 and 5-6 shows that the only intersection point between the line of a state-dependent algorithm and the optimal line is the point of origin. When traffic demand reaches a moderate level, a state-dependent algorithm is not able to approximate the optimal line, because a state-dependent algorithm avoids rearrangement.

The observations in this chapter imply that the development of the state-dependent routing mechanism meets with some substantial difficulties from the

perspective of resource utilization. The main cause of this is lack of knowledge of future traffic demand.

Chapter 6 Simulations of the Time-Dependent Routing Mechanism

In this chapter, first a time-dependent QoS routing algorithm will be designed. Later, its performance will be compared with the selected benchmark, WSP, from various different angles. To get a better idea of how far a single traffic forecast can survive under dynamic traffic demand, the partial conclusions in section 6.2 will be generalized in the solvable traffic demand space in section 6.3. Finally, the advantages and limitations of time-dependent QoS routing algorithms will be summarized.

6.1 TDWSP Algorithm

The observations in the preceding chapter prompted us to try a time-dependent routing algorithm in a dynamic network environment. A few design decisions have been made to reflect the Internet philosophy – *simplicity* – as much as possible.

Suri et al. proposed in [12] a way of using a priori knowledge of traffic (i.e. a traffic profile) to improve QoS routing. However, they used a linear optimization, rather than a nonlinear optimization, as discussed in Chapter 4. Instead of adopting a real-time adjustment of a traffic profile in PBR, in this thesis we propose a fixed traffic profile-based routing algorithm, namely, the Time-Dependent Widest Shortest Path (TDWSP).

TDWSP is a variation of WSP, which was chosen as our benchmark for evaluating the performance of TDWSP. In general, the idea of constructing TSWSP from WSP can also be applied to any other state-dependent routing algorithms. For example,

TDSWP can also be constructed based on a SWP algorithm. The only difference between TDWSP and WSP is that TDWSP knows the possible future traffic, but WSP does not.

The reason why WSP was selected as our benchmark is that WSP performed the best in the simulations in Chapter 5. It will be interesting to see how far TDWSP can go beyond WSP. The simulations in this chapter are designed to answer the following questions:

- Is information regarding traffic forecast valuable for improving the performance of WSP?
- How much performance improvement can information regarding traffic forecast achieve in TDWSP?
- Is the performance of TDWSP sensitive to the forecast? How much walk-off of forecast can TDWSP tolerate?

TDWSP consists of two phases: offline planning and online routing, which are similar to the two phases of PBR proposed by Suri et al. in [12]. These two phases are executed in sequence, and the first phase is preparation for the second phase. However, as a part of a network operation, the first phase in PBR is executed repeatedly when traffic demand changes, whereas the first phase of TDWSP runs one time only, as a part of network configuration. Those two phases are discussed below, with illustrations.

6.1.1 Offline Traffic Planning

Suppose a traffic demand matrix T is given, then an optimum solution of minimizing the formula 4-4 is calculated by solving a nonlinear network flow problem. As long as the flow for any source-destination pair is splittable in any percentage, the network flow problem can be solved efficiently. This is certainly not a realistic

assumption for QoS routing. However, the optimization solution based on this ideal assumption does provide us with a useful blueprint to arrange explicit routes of QoS requests.

The input of *offline traffic planning* is the matrix T , which represents a traffic demand forecast in a network; an order n in formula 4-4, which adjusts the preference of the optimization objectives; and the network topology information.

The output of *offline traffic planning* is then an optimum solution, indicating how to split one traffic flow for every source-destination pair into multiple sub-flows, their explicit route, as well as their assigned bandwidth. This optimal solution is considered as a guide or a reference for TDWSP when the online routing phase is conducted.

**Figure 6-1. Forecast in Fig. 6-3 and Calculated Preplanned Tree
Network #2, $n=2$**

In Figure 6-1, the partial routing solution of network #2 is illustrated, where traffic ingresses at node n_1 , and the traffic forecast is at 50.4% of $T_{a,2}$, demonstrated in

Figure 5-4. Please refer to Appendix 2 for a complete example of an optimum routing solution for the 10-node network #1.

After the offline traffic planning has taken place, the link capacities are broken into two parts: assigned bandwidth and unassigned bandwidth. Assigned bandwidth represents bandwidth reserved for expected traffic. Unassigned bandwidth refers to free bandwidth for unexpected traffic. Moreover, the assigned bandwidth is split into N trees, where N is the number of nodes in a network. Each ingress router owns a tree with its root at the ingress router itself. In TDWSP, we define every tree (such as the one in Figure 6-1) that consists of assigned bandwidth of flows from one source node as a *preplanned tree*, and a network that consists of unassigned bandwidth is defined as a *residue network*. The preplanned tree and the residue network have the following characteristics:

- *Preplanned tree*. For each ingress router, the preplanned tree is a directional graph without loops. The example of a preplanned tree shown in Figure 6-1 is marked with assigned bandwidth. For instance, there are two routes, 1-5-3 and 1-20-3, going to node 3, with a total bandwidth of 151 Mbps ($69.43+81.57=151$). Subtract 40 Mbps (the bandwidth leaving node 3 is $11+14+7+8=40$) from 151 Mbps. This leaves 111Mbps, which means 111 Mbps of traffic from node 1 sink into node 3.
- *Residue network*. In most cases, especially when the traffic level forecasted is not high, all the links in the network have unassigned bandwidth, since the maximum link utilization is minimized. Therefore, the residue network is generally a *connected network*.

6.1.2 Online Routing

Once the offline traffic planning has been completed, TDWSP is applied to process incoming QoS requests dynamically. Every ingress router has three pieces of information:

- *Forecasted traffic.* A router does not have to have a complete picture of the traffic forecast. Rather, only the forecasted traffic that it itself has injected is concerned. In other words, an ingress router knows how much forecasted traffic has already been accepted from itself to all other nodes and how much more traffic is expected. Note that this piece of information is local during the operation.
- *Preplanned tree.* There is one tree per router. For each router, this piece of information is determined after the offline computation has been completed. When online routing starts, the router also has to keep track of the resource consumption within the preplanned tree as QoS requests are accepted. In particular, the root router keeps a record of free bandwidth available within the preplanned tree, and this record is also local. Therefore, no information exchange with other routers is required to maintain this record.
- *Residue network.* Every router in a network shares the same residue network information. Like any other QoS routing algorithms, TDWSP keeps track of the available bandwidth within the residue network, which requires database synchronization among routers.

6.1.2.1 Pseudo Code of TDWSP Online Routing

TDWSP divides incoming QoS requests into two categories: expected and unexpected. Expected requests are those that do not exceed the forecasted traffic;

otherwise, the requests are deemed unexpected. If a request is expected, TDWSP applies the WSP algorithm within the preplanned tree. If the request is unexpected, TDWSP applies the WSP algorithm within the residue network.

Note that it is possible for an expected request to be rejected within the preplanned tree due to the packing issue. In that case, the expected request is given another chance within the residue network. For instance, there are 111 Mbps of traffic from node 1 to node 3 in the forecast, and TDWSP has accepted 109 Mbps of traffic from node 1 to node 3. TDWSP therefore expects 2 Mbps more of traffic from node 1 to node 3. However, within the preplanned tree only 1.43 Mbps are left on route 1-5-3 and 0.57 Mbps are left on route 1-20-3. As a result, the 2 Mbps request will be rejected within the preplanned tree, because neither route can accommodate the request. TDWSP then tries to accept this request within the residue network if there is enough free bandwidth available there.

For a request of bandwidth bw from source node s to destination node d , with hop count constraint c , the pseudo code for the TDWSP is as follows:

```

initialize expectedBw, plannedFreeBw, residueFreeBw
if bw < expectedBw(s,d) {
  do WSP(s,d,c) on plannedTree
  if success {
    expectedBw(s,d) = expectedBw(s,d) - bw
    plannedFreeBw(s,d) = plannedFreeBw(s,d) - bw
  } else {
    do WSP(s,d,c) on residueNet
    if success {
      residueFreeBw(s,d) = residueFreeBw(s,d) - bw
    }
  }
} else {
  do WSP(s,d,c) on residueNet
  if success {
    residueFreeBw(s,d) = residueFreeBw(s,d) - bw
  }
}

```

As can be seen in the pseudo code, the QoS constraint hop count c is fully controlled by WSP. TDWSP does not intend to change the existing mechanism in WSP that controls constraints, except for the bandwidth constraint bw .

6.2 Simulations

The performance of TDWSP is analyzed in the simulations below.

6.2.1 Simulation Setup

TDWSP is studied in both the topologies of network #1 and #2. To make the results compatible with those obtained in Chapter 5, most configurations have not been changed. TDWSP was repeated using exactly the same traffic traces used in Chapter 5.

6.2.1.1 Forecast in TDWSP

TDWSP knows the forecast denoted as f , i.e. $T_{b,1} = 53.9\% T_{a,1}$ and $T_{b,2} = 50.4\% T_{a,2}$ as the predicted traffic load for networks #1 and #2 respectively. The matrices of the predicted traffic demands are shown in Figures 6-2 and 6-3.

$$T_{b,1} = \begin{bmatrix} 0 & 167 & 93 & 184 & 160 & 11 & 32 & 129 & 112 & 82 \\ 167 & 0 & 283 & 405 & 293 & 68 & 69 & 240 & 191 & 198 \\ 93 & 283 & 0 & 266 & 227 & 5 & 95 & 218 & 210 & 92 \\ 184 & 405 & 266 & 0 & 459 & 33 & 78 & 292 & 241 & 215 \\ 160 & 293 & 227 & 459 & 0 & 33 & 59 & 179 & 130 & 177 \\ 11 & 68 & 5 & 33 & 33 & 0 & 5 & 33 & 33 & 11 \\ 32 & 69 & 95 & 78 & 59 & 5 & 0 & 50 & 42 & 37 \\ 129 & 240 & 218 & 292 & 179 & 33 & 50 & 0 & 77 & 160 \\ 112 & 191 & 210 & 241 & 130 & 33 & 42 & 77 & 0 & 349 \\ 82 & 198 & 92 & 215 & 177 & 11 & 37 & 160 & 349 & 0 \end{bmatrix}$$

Figure 6-2. Traffic Forecast $T_{b,1}$ of Network #1

$$T_{b,2} = \begin{bmatrix} 0 & 43 & 111 & 11 & 62 & 21 & 14 & 7 & 55 & 8 & 22 & 122 & 14 & 10 & 3 & 86 & 74 & 26 & 52 & 107 \\ 43 & 0 & 98 & 11 & 56 & 19 & 14 & 7 & 50 & 8 & 4 & 118 & 14 & 10 & 3 & 73 & 62 & 38 & 30 & 63 \\ 111 & 98 & 0 & 58 & 188 & 46 & 396 & 46 & 132 & 23 & 11 & 270 & 41 & 29 & 10 & 160 & 127 & 63 & 85 & 195 \\ 11 & 11 & 58 & 0 & 5 & 5 & 0 & 0 & 11 & 0 & 0 & 34 & 0 & 0 & 0 & 34 & 34 & 5 & 3 & 34 \\ 62 & 56 & 188 & 5 & 0 & 63 & 7 & 4 & 61 & 4 & 2 & 177 & 7 & 5 & 2 & 145 & 140 & 30 & 19 & 151 \\ 21 & 19 & 46 & 5 & 63 & 0 & 7 & 4 & 25 & 4 & 2 & 52 & 7 & 5 & 20 & 34 & 28 & 11 & 16 & 40 \\ 14 & 14 & 396 & 0 & 7 & 7 & 0 & 0 & 14 & 0 & 0 & 43 & 0 & 0 & 0 & 43 & 43 & 7 & 3 & 43 \\ 7 & 7 & 46 & 0 & 4 & 4 & 0 & 0 & 7 & 0 & 0 & 22 & 0 & 0 & 0 & 22 & 22 & 4 & 4 & 22 \\ 55 & 50 & 132 & 11 & 61 & 25 & 14 & 7 & 0 & 8 & 4 & 143 & 14 & 59 & 3 & 107 & 233 & 30 & 35 & 118 \\ 8 & 8 & 23 & 0 & 4 & 4 & 0 & 0 & 8 & 0 & 0 & 25 & 0 & 0 & 0 & 25 & 25 & 4 & 4 & 25 \\ 22 & 4 & 11 & 0 & 2 & 2 & 0 & 0 & 4 & 0 & 0 & 11 & 0 & 0 & 0 & 11 & 11 & 2 & 2 & 11 \\ 122 & 118 & 270 & 34 & 177 & 52 & 43 & 22 & 143 & 25 & 11 & 0 & 41 & 29 & 10 & 194 & 161 & 68 & 89 & 306 \\ 14 & 14 & 41 & 0 & 7 & 7 & 0 & 0 & 14 & 0 & 0 & 41 & 0 & 0 & 0 & 41 & 41 & 7 & 7 & 595 \\ 10 & 10 & 29 & 0 & 5 & 5 & 0 & 0 & 59 & 0 & 0 & 29 & 0 & 0 & 0 & 29 & 29 & 5 & 5 & 29 \\ 3 & 3 & 10 & 0 & 2 & 20 & 0 & 0 & 3 & 0 & 0 & 10 & 0 & 0 & 0 & 10 & 10 & 2 & 2 & 10 \\ 86 & 73 & 160 & 34 & 145 & 34 & 43 & 22 & 107 & 25 & 11 & 194 & 41 & 29 & 10 & 0 & 52 & 78 & 76 & 119 \\ 74 & 62 & 127 & 34 & 140 & 28 & 43 & 22 & 233 & 25 & 11 & 161 & 41 & 29 & 10 & 52 & 0 & 45 & 72 & 86 \\ 26 & 38 & 63 & 5 & 30 & 11 & 7 & 4 & 30 & 4 & 2 & 68 & 7 & 5 & 2 & 78 & 45 & 0 & 46 & 56 \\ 52 & 30 & 85 & 3 & 19 & 16 & 3 & 4 & 35 & 4 & 2 & 89 & 7 & 5 & 2 & 76 & 78 & 46 & 0 & 77 \\ 107 & 65 & 195 & 34 & 151 & 40 & 43 & 22 & 118 & 25 & 11 & 306 & 595 & 29 & 10 & 119 & 86 & 56 & 77 & 0 \end{bmatrix}$$

Figure 6-3. Traffic Forecast $T_{b,2}$ of Network #2

The setting of the forecast level is not arbitrary. Recall that $T_{a,1}$ is the maximum traffic demand matrix that network topology #1 can handle when the traffic demand increases or decreases proportionally to $T_{a,1}$ (i.e. 101% of $T_{a,1}$ is unsolvable due to link capacity constraints). Similarly, $T_{a,2}$ is for network #2. This means that the forecast is assumed to be at the middle level of network capacity, and can be justified by the fact that current operational networks are normally not running at a high utilization on average [20].

Equation 4-4 is regarded as the objective function for the offline traffic planning phase of TDWSP, where n equals 2. Various values of n were tested in different simulations, and we ascertained that a setting of 2 or 3 is sufficiently good in routing. If n is too large, the routing solution might contain unnecessarily long routes.

6.2.2 Simulation Results

Figure 6-4. Network Performance with WSP and TDWSP
Network #1, $n=2$

In Figures 6-4 and 6-5, TDWSP, an example of a time-dependent QoS routing algorithm, outperforms WSP, an example of a state-dependent QoS routing algorithm, at almost all load levels, especially when the traffic load is below the forecast (WSP already performs so well in network #1 that any improvement using TDWSP is not clear in Figure 6-4. Therefore, the result in Figure 6-4 is rephrased in Table 6-1.).

Traffic Demand (of $T_{a,1}$)	Mean Objective Function Value		
	WSP	TDWSP	Optimal
8.98%	1241	1229	1220
17.96%	4908	4857	4832
26.95%	11020	10894	10857
35.93%	19517	19246	19216
44.91%	30595	30144	30133
53.89%	43891	43259	43259
62.87%	59845	59047	58940

71.86%	77976	77078	76788
80.84%	98805	97836	97368
89.82%	122437	121330	120643
99.40%	176465	176814	167337

Table 6-1. Network Performance with WSP and TDWSP
Network #1, $n=2$

In the 10-node network #1, TDWSP cannot improve WSP much because it is bounded by the optimal line. However, in the 20-node network #2, which is a more sophisticated network topology for routing, it is clearer that TDWSP has a strong ability to stay close to the optimal line.

Figure 6-5. Network Performance with WSP and TDWSP
Network #2, $n=2$

An analysis of Figures 6-4 and 6-5 confirms another fact: If network resources can be optimized to meet a high traffic demand, it is not difficult to optimize the network to meet a relatively low traffic demand, according to the solution of high traffic demand.

From this perspective, the forecast of peak traffic demand or *extreme value* [49, 50] is important when configuring TDWSP. More generally, a time-dependent routing algorithm should pay more attention to peak traffic instead of other moderate traffic values.

Figure 6-6. Performance with 95% Confidence Interval
Network #1, $n=2$

Among 20 randomly generated QoS request traces, there is a small fluctuation in objective function values (Figures 6-6 and 6-7). When traffic demand is not high, this fluctuation is invisible for both WSP and TDWSP. Both algorithms are good at controlling their performance values within a small range, and TDWSP performs better. More detailed observations on deviation are provided in section 6.4.2.

Figure 6-7. Performance with 95% Confidence Interval
Network #2, $n=2$

Figure 6-8. Average Hop Count with WSP and TDWSP
Network #2, $n=2$

Again, the average hop count and maximum link utilization for TDWSP are examined for network #2. In Figure 6-8 TDWSP sacrifices only a little bit with regard to mean hop count compared to WSP. However, in Figure 6-9 TDWSP performs much better than WSP in terms of maximum link utilization.

**Figure 6-9. Maximum Link Utilization with WSP and TDWSP
Network #2, $n=2$**

A critically important benefit of TDWSP is that it can come up with a nice compromise in performance between the average hop count and the maximum link utilization. Note that the optimal line in Figure 6-8 is for average hop count only, and the optimal line in Figure 6-9 is for maximum link utilization only. Two optimal lines cannot be achieved at the same time because their interests do not coincide with each other. Therefore, as illustrated in Figures 6-8 and 6-9, the close proximity of the TDWSP line and the optimal line, due to knowledge of the traffic forecast, is noticeable.

The results exhibited so far shed light on the QoS routing problem and its possible solution by making use of traffic forecast information. In TDWSP, the traffic forecast is fixed and does not change in real-time. TDWSP regards the forecast information as part of its routing configuration, like link weights. Therefore, the implementation of TDWSP is simple, because it does not need to consider dynamic run-time situations and act accordingly. On the other hand, it seems that the forecast does not have to be very accurate or recent. Maybe a rough forecast is good enough for carrying out network traffic planning and significantly improving routing performance.

6.2.3 Effect of Predicted Traffic Demand

Figure 6-10. Network Performance with Different Forecast
Network #2, $n=2$

From Figures 6-4 and 6-5, it is suspected that the performance of TDWSP is insensitive to the position of forecast f . To further investigate this point, another forecast f

= 58.8% $T_{a,2}$ is verified in a simulation of network #2. Two lines of TDWSP are plotted in Figure 6-10: $f = 50.4\%$ and $f = 58.8\%$. The figure shows that the two lines are very close to each other, and hence f 's setting does not make much difference. More similar simulation results in network #1 and #2 reveal that the performance of TDWSP is not sensitive to the setting of forecasted traffic demand.

6.2.4 Effect of n

As discussed in section 6.2.1, $n = 2$ is assumed in the performance evaluation and in the offline network planning phase of TDWSP, and it is a reasonable assumption for real operational networks. A larger n value implies that Equation 4-4 tends to put more emphasis on minimizing the maximum link utilization rather than minimizing the average hop count traversed by the routes.

Figure 6-11. Network Performance When $n=3$
Network #2

The simulation in network #2 is repeated for $n = 3$ to see how this factor changes the results. More specifically, a network-wide performance evaluation uses formula 4-4 with $n = 3$, and the bandwidth pre-allocation plan of TDWSP was calculated at $n = 3$. As a comparison to Figure 6-5, the value of the optimization objective function for $n = 3$ is presented in Figure 6-11.

The effect of the value of n is illustrated in Figures 6-12 and 6-13. TDWSP with $n = 3$ has a larger average hop count in Figure 6-12, but, as shown in Figure 6-13, it exhibits better performance in terms of minimizing the maximum link utilization.

Figure 6-12. Average Hop Count with $n=2$ and $n=3$
Network #2

Figure 6-13. Maximum Link Utilization with $n=2$ and $n=3$
Network #2

It is very clear that there is a trade-off between minimizing the mean hop count and minimizing the maximum link utilization. The order n can be used as an effective tool by a network administration team for adjusting the network to achieve different performance objectives.

6.3 Generalization of the Simulation Results

The simulations presented in section 6.2 were all carried out under a similar traffic pattern but at different scales. In this section, structurally different traffic demands are considered.

6.3.1 The Solvable Traffic Demand Space

So far we have not used a vector representation of the traffic matrix (see section 3.2.1) to analyze the simulation results we obtained. Now the vector form will be used to

better understand the QoS routing environment. For instance, traffic matrix $T_{a,l}$ can be represented as vector $d_{a,l}$, and $T_{b,l}$ can be represented as $d_{b,l}$.

The traffic demand vector d is then a $N(N-1)$ -element vector. It is certain that each element in d is nonnegative, since the traffic demand for any source-destination pair is either zero or a positive value.

Imagine a space that consists of the end-points of all legitimate vector d , and call it the *traffic demand space*. The traffic demand space has boundaries, which are imposed by the nonnegative elements in vector d . However, the traffic demand space is infinite, as traffic demands can be very large positive numbers.

From the routing point of view, in this thesis we define a *solvable traffic demand space* as a subset of the traffic demand space with at least one feasible routing solution, no matter whether the feasible routing solution is optimized or not. Because the solvable traffic demand space requires points inside the space to be solvable, the solvable traffic demand space is a finite space when the network topology and link capacities are given. In other words, as long as a network is designed, it is already determined that some traffic demands d are routable and other traffic demands d are unroutable, due to link capacity constraints.

Note that the origin is part of the solvable traffic demand space.

Lemma 1: The solvable traffic demand space is convex finite.

The proof of this lemma is based on the definition of convexity. The solvable traffic demand space is said to be convex if, for all the points x and y in the space and any k in the interval $[0,1]$, the point $kx + (1-k)y$ is in the space as well. In other words, every

point on the straight line segment connecting two solvable points x and y is also in the solvable traffic demand space.

Let M be the number of links in a network, and let vector $C = [c_1 \ c_2 \ c_3 \ \dots \ c_M]$ represent the link capacity of every link in the network. Since x is routable, kx must be routable within link capacity kC . Likewise, $(1-k)y$ must be routable within link capacity $(1-k)C$. Therefore, the point $kx + (1-k)y$ must be routable within link capacity $kC + (1-k)C = C$, and hence is in the solvable traffic demand space.

Figure 6-14. Positions of Three Lines of Traffic Demand in the Solvable Space

In the rest of section 6.3, we will extend our simulation by testing TDWSP under different traffic demands within the solvable space. Three lines of traffic demand are constructed in the solvable space: The first line runs from the origin to demand d_a ; the second line is from the forecast demand d_b to d_c ; and the third line is from the origin to demand d_d . Their relative positions are illustrated in Figure 6-14. Because d_a , d_c and d_d are all set on the boundary of the solvable space, by changing the traffic demand along

these three lines, we intend to investigate the performance of TDWSP in the solvable space more completely.

The goal is to understand how TDWSP behaves in the solvable traffic demand space. Although it is impossible to investigate every point in the solvable space, more simulations are required to obtain a more complete picture. The simulation results when the traffic demand changes along the three lines in Figure 6-14 are presented in the following sections.

6.3.2 First Line, from the Origin to d_a

Let d_a represent the traffic demand for point a and $d_{a,1}$ be the demand d_a for network #1. Based on Lemma 1, all points on the straight line segment defined by two end-points, the origin and the point $d_{a,1}$, are in the solvable traffic demand space. Furthermore, these two points are two intersection points when the line penetrates the space. From this perspective, Figure 6-4 can be interpreted as the routing performance of multiple QoS routing algorithms when real traffic changes along the line segment from the origin to $d_{a,1}$. Similarly, Figure 6-5 can be interpreted as the routing performance of multiple QoS routing algorithms when real traffic changes along the line segment from the origin to $d_{a,2}$.

6.3.3 Second Line, from d_b to d_c

Two structurally different demands, $d_{c,1}$ and $d_{c,2}$, are invented for networks #1 and #2, respectively. Note that the focus of the simulation remains on the forecast point d_b , which corresponds to the traffic matrix in Figures 6-2 and 6-3. Therefore, the second line is constructed from d_b to d_c . In this way, the simulation result will show how TDWSP performs when the traffic demand leaves d_b for d_c .

6.3.3.1 Line Construction

There are a number of steps in constructing a straight line segment going through the solvable traffic demand space. Let's take network #2 as an example.

The first step is to obtain another traffic vector $d_{temp,2}$, which is different from point $d_{b,2}$, though at this time it is not necessary that $d_{temp,2}$ be routable. In terms of their corresponding traffic matrices $T_{temp,2}$ and $T_{b,2}$, some instructions have been applied to them in order to better serve our purpose in the simulation.

- Keep $T_{temp,2}$ and $T_{b,2}$ at the same network throughput level. The point of this simulation is to see how TDWSP reacts under different traffic patterns, instead of how TDWSP reacts with a lower or higher traffic demand. From this point of view, the injected traffic volume of each node is unchanged, but its distribution to the other nodes is changed. For example, in $T_{b,2}$, the total traffic injected into node 1 is 848 Mbps, so we keep this number unchanged in $T_{temp,2}$.
- Rearrange the demands randomly in $T_{b,2}$. It is especially important to replace low traffic demands with high values, and vice versa. In $T_{temp,2}$, the distribution of the 848Mbps among pairs 1-2, 1-3, 1-4, ..., and 1-20 is intentionally different from that in $T_{b,2}$. For instance, in $T_{b,2}$ the pair 1-15 has a 3 Mbps demand, but in $T_{temp,2}$ it has a demand of more than 30 Mbps. The motivation is to test TDWSP under very different situations, as its forecast is structurally almost contrary to the actual traffic demand.

$$T_{c,2} = \begin{bmatrix} 0 & 38 & 96 & 26 & 54 & 29 & 25 & 15 & 50 & 23 & 23 & 106 & 20 & 17 & 21 & 75 & 64 & 25 & 45 & 97 \\ 37 & 0 & 84 & 14 & 49 & 25 & 19 & 21 & 45 & 16 & 21 & 103 & 18 & 18 & 14 & 68 & 54 & 34 & 27 & 54 \\ 104 & 85 & 0 & 79 & 173 & 58 & 361 & 87 & 116 & 34 & 26 & 236 & 55 & 29 & 15 & 195 & 148 & 55 & 79 & 174 \\ 9 & 10 & 49 & 0 & 9 & 4 & 9 & 2 & 9 & 5 & 2 & 29 & 5 & 2 & 5 & 30 & 30 & 4 & 3 & 29 \\ 74 & 48 & 168 & 5 & 0 & 54 & 6 & 13 & 74 & 13 & 30 & 155 & 7 & 31 & 5 & 124 & 128 & 48 & 16 & 129 \\ 23 & 24 & 40 & 8 & 54 & 0 & 13 & 7 & 31 & 4 & 5 & 47 & 7 & 6 & 18 & 33 & 26 & 10 & 20 & 34 \\ 18 & 18 & 339 & 1 & 6 & 6 & 0 & 1 & 12 & 1 & 2 & 96 & 0 & 6 & 2 & 37 & 37 & 6 & 9 & 37 \\ 6 & 9 & 42 & 1 & 7 & 4 & 7 & 0 & 6 & 1 & 1 & 19 & 0 & 3 & 0 & 19 & 20 & 4 & 3 & 19 \\ 82 & 47 & 117 & 11 & 61 & 25 & 14 & 24 & 0 & 8 & 5 & 122 & 33 & 52 & 11 & 91 & 214 & 34 & 37 & 120 \\ 7 & 11 & 20 & 0 & 3 & 7 & 4 & 1 & 7 & 0 & 1 & 22 & 3 & 0 & 1 & 22 & 21 & 3 & 5 & 25 \\ 19 & 4 & 11 & 0 & 2 & 2 & 2 & 2 & 4 & 2 & 0 & 9 & 3 & 1 & 2 & 9 & 10 & 2 & 2 & 10 \\ 110 & 105 & 231 & 58 & 175 & 54 & 50 & 65 & 125 & 43 & 13 & 0 & 53 & 42 & 49 & 170 & 163 & 66 & 82 & 262 \\ 12 & 12 & 41 & 6 & 7 & 8 & 2 & 6 & 12 & 1 & 1 & 35 & 0 & 1 & 89 & 35 & 37 & 6 & 6 & 512 \\ 9 & 9 & 26 & 2 & 9 & 4 & 0 & 4 & 55 & 0 & 4 & 25 & 4 & 0 & 1 & 25 & 25 & 4 & 13 & 25 \\ 4 & 3 & 9 & 3 & 2 & 19 & 2 & 0 & 3 & 2 & 0 & 9 & 0 & 0 & 0 & 9 & 9 & 3 & 2 & 9 \\ 86 & 91 & 142 & 33 & 125 & 40 & 61 & 24 & 113 & 26 & 16 & 173 & 47 & 36 & 26 & 0 & 48 & 82 & 68 & 103 \\ 67 & 54 & 116 & 42 & 128 & 43 & 42 & 40 & 202 & 23 & 16 & 140 & 70 & 28 & 15 & 55 & 0 & 49 & 85 & 79 \\ 22 & 44 & 57 & 10 & 35 & 10 & 10 & 5 & 27 & 4 & 6 & 58 & 6 & 11 & 10 & 77 & 39 & 0 & 40 & 54 \\ 45 & 28 & 75 & 3 & 17 & 19 & 3 & 4 & 41 & 14 & 10 & 80 & 19 & 18 & 3 & 65 & 62 & 46 & 0 & 77 \\ 99 & 65 & 182 & 38 & 158 & 39 & 38 & 65 & 190 & 44 & 15 & 267 & 509 & 42 & 12 & 106 & 75 & 65 & 78 & 0 \end{bmatrix}$$

Figure 6-15. Traffic Matrix $T_{c,2}$ of Network #2

The second step is to decide on a straight line by points $d_{b,2}$ and $d_{temp,2}$. The solvable traffic demand space will intersect the line and we can ascertain where the two intersection points are. Applying the formula $kx + (1-k)y$ where $k \in [-\infty, +\infty]$, we obtain two points on the boundary of the solvable space: $d_{b',2}$ and $d_{c,2}$. Figure 6-15 represents $d_{c,2}$ in the matrix form $T_{c,2}$.

Surprisingly, $d_{b',2} = d_{b,2}$ for network #2. A close look at the matrix $T_{b,2}$ (Figure 6-3) indicates that this is not a coincidence. Using source destination pair 14-4 as an example, the number in $T_{b,2}$ is 0Mbps and the number in $T_{c,2}$ is 2 Mbps. Recall that the traffic demand defined in the matrix must be nonnegative, therefore the line from $d_{c,2}$ to $d_{b,2}$ cannot go further than point $d_{b,2}$, which is a zero value. It is concluded that any traffic matrix that has a non-diagonal element or elements of zero value has a corresponding

demand sitting right on the boundary of the solvable traffic demand space, hence it is likely that $d_{b'} = d_b$.

Likewise, for network #1, Figure 6-16 represents $d_{c,1}$ in the matrix form $T_{c,1}$. Another point $d_{b',1}$ equals $(-0.05) d_{temp,1} + (1 - (-0.05)) d_{b,1}$, so $d_{b',1}$ is very close to $d_{b,1}$.

$$T_{c,1} = \begin{bmatrix} 0 & 138 & 122 & 175 & 102 & 57 & 63 & 111 & 140 & 63 \\ 212 & 0 & 238 & 341 & 253 & 119 & 200 & 261 & 143 & 148 \\ 160 & 261 & 0 & 199 & 249 & 40 & 143 & 215 & 130 & 92 \\ 291 & 260 & 193 & 0 & 352 & 178 & 151 & 272 & 231 & 245 \\ 110 & 202 & 208 & 331 & 0 & 83 & 150 & 198 & 258 & 177 \\ 9 & 54 & 16 & 24 & 47 & 0 & 7 & 22 & 33 & 20 \\ 39 & 58 & 72 & 60 & 63 & 40 & 0 & 61 & 49 & 25 \\ 172 & 176 & 195 & 263 & 223 & 90 & 81 & 0 & 60 & 117 \\ 143 & 198 & 222 & 198 & 123 & 156 & 38 & 63 & 0 & 243 \\ 86 & 205 & 125 & 135 & 122 & 69 & 159 & 130 & 290 & 0 \end{bmatrix}$$

Figure 6-16. Traffic Matrix $T_{c,1}$ of Network #1

An observation related to both $T_{c,1}$ and $T_{c,2}$ is that the distance between the origin and $d_{a,1}$ is much longer than the distance between $d_{b,1}$ and $d_{c,1}$ for network #1, and the distance between the origin and $d_{a,2}$ is also much longer than the distance between $d_{b,2}$ and $d_{c,2}$ for network #2. Although more experimentation is required to confirm the observation, it does provide some insights into the shape of the solvable traffic demand space. More detailed discussion on this topic can be found in section 6.3.5.

6.3.3.2 Simulation Results of the Line

Figures 6-17 and 6-18 illustrate the performance results for networks #1 and #2. The forecast point at d_b is configured for TDWSP. The performance of TDWSP is very close to the optimal line. The 95% confidence interval is negligible, as demonstrated in Figures 6-19 and 6-20. There are two reasons why TDWSP performs close to the optimal:

Figure 6-17. Network Performance as Traffic Changes from $T_{b,1}$ to $T_{c,1}$
Network #1, $n=2$

Figure 6-18. Network Performance as Traffic Changes from $T_{b,2}$ to $T_{c,2}$
Network #2, $n=2$

Figure 6-19. Performance with 95% Confidence Interval as Traffic Changes from $T_{b,1}$ to $T_{c,1}$, Network #1, $n=2$

Figure 6-20. Performance with 95% Confidence Interval as Traffic Changes from $T_{b,2}$ to $T_{c,2}$, Network #2, $n=2$

- The network-wide traffic load is at about the middle of the network throughput, which is indicated by the mean link utilization of around 35% and 30% in networks #1 and #2 respectively, based on calculations. When the traffic load is not high, the performance of TDWSP is not sensitive to the configured forecast.
- The distance from d_b to d_c is short. For example, in network #2, the link utilization of all 74 unidirectional links in the optimal solution when the traffic demand is $T_{c,2}$ is very unevenly distributed. Most links are under-utilized, while a few other links are highly over-utilized. Because of this unbalanced traffic distribution, the distance between $d_{b,2}$ and $d_{c,2}$ is short. Any points beyond $d_{c,2}$ are unsolvable. Because of this short distance, TDWSP does not degrade much before the line goes beyond the solvable space.

Figure 6-21. Maximum Link Utilization as Traffic Changes from $T_{b,2}$ to $T_{c,2}$
Network #2, $n=2$

When the offered traffic is close to the point $d_{c,2}$, the standard derivation for TDWSP also increases, as shown in Figure 6-21. This can be explained by increasing the number of unexpected QoS requests. The forecast at $d_{b,2}$ becomes more and more inaccurate as actual traffic demand moves to $d_{c,2}$. The nature of TDWSP makes it more sensitive to the sequence of QoS requests when the number of unexpected requests increases.

6.3.4 Third Line, from d_d to the Origin

A third line from d_d to the origin can then be constructed, while the forecast of TDWSP at d_b is unchanged.

6.3.4.1 Line Construction

The third line is constructed from two points: point d_c and the origin. Because of Lemma 1, there must be two intersection points or less when one line penetrates the space. For network #1, one intersection point is the origin itself, and the other intersection point is at $d_{d,1} = 101\% d_{c,1}$. Coincidentally, for network #2, one intersection point is the origin itself, and the other intersection point is at $d_{d,2} = 101\% d_{c,2}$.

An important observation here is that in network #1, the distance from the origin to $d_{a,1}$ is much longer than the distance from the origin to $d_{c,1}$, and in network #2 the distance from the origin to $d_{a,2}$ is much longer than the distance from the origin to $d_{c,2}$ (refer to Figure 6-14). The key is that both networks #1 and #2 are real parts of an ISP's core networks on the Internet, and the demands $d_{a,1}$ and $d_{a,2}$ are based on real measurements. It is reasonable to believe that these networks are designed and modified according to real traffic patterns, and therefore they are more suitable for the real traffic pattern than for other artificial traffic patterns, such as $d_{c,1}$ and $d_{c,2}$. Usually the mapping

of artificial traffic patterns to a real network topology is unbalanced. Therefore, artificial traffic patterns cannot scale well to a large traffic load within the solvable traffic demand space.

6.3.4.2 Simulation Results of the Line

Figure 6-22. Network Performance as Traffic Changes from Zero to $T_{d,1}$
Network #1, $n=2$

The simulation results of the third line in networks #1 and #2 are satisfactory. As the demand changes from the origin to d_d , the curve of TDWSP is extremely close to the optimal curve, although the forecast configuration is based on demands $d_{b,1}$ and $d_{b,2}$ respectively.

Figure 6-23. Network Performance as Traffic Changes from Zero to $T_{d,2}$
Network #2, $n=2$

6.3.5 Some Observations about the Solvable Space

In network #1, three curves show the performance of TDWSP: the first curve as traffic demand changes from the origin to $d_{a,1}$ (Figure 6-4), the second curve as traffic demand changes from $d_{b,1}$ to $d_{c,1}$ (Figure 6-17), and the third curve as traffic demand changes from $d_{d,1}$ to the origin (Figure 6-22). Meanwhile, in network #2, there are also three curves that display the performance of TDWSP: the first curve as traffic demand changes from the origin to $d_{a,2}$ (Figure 6-5), the second curve as traffic demand changes from $d_{b,2}$ to $d_{c,2}$ (Figure 6-18), and the third curve as traffic demand changes from $d_{d,2}$ to the origin (Figure 6-23). The relationship of these performance curves is portrayed informally in Figure 6-24.

Recall that for all three curves in network #1, the forecast point configured in TDWSP is at $d_{b,1}$, and for all three curves in network #2, the forecast point configured in TDWSP is at $d_{b,2}$.

Figure 6-24. Three Lines of Traffic Demand in the Solvable Space

The author of this thesis does not intend to draw any strong conclusions regarding the effects of utilizing traffic forecast in QoS routing, since real networks are much more complicated. Nevertheless, the results obtained so far have some meaningful implications, despite the limitation of the simulations.

6.3.5.1 Performance of TDWSP

Whichever QoS routing algorithm is under consideration, a point representing the current traffic demand is moving within the traffic demand space, and accordingly a performance value is decided when the current traffic demand is given¹.

¹ Strictly speaking, it is not a single performance value. Most routing algorithms are sensitive to the order of LSP requests, so the value of the objective function may vary in a range for every given traffic demand.

It is impossible to walk through all the possible points in the traffic demand space to evaluate the performance of TDWSP and other state-dependent routing algorithms. In fact, the traffic demand space is not the main concern. Instead, we are primarily concerned with the solvable traffic demand space. For any routing algorithm, if the given traffic is within the traffic demand space but outside the solvable traffic demand space, routing algorithms *have to* reject or drop part of the traffic in order to bring the demand back into the solvable traffic demand space.

All the lines of traffic demand in Figure 6-24 are within the solvable traffic demand space, and all the marked points in the figure (except $d_{b,1}$), such as the origin, $d_{a,2}$, $d_{b,2}$, etc., are on the boundary of the solvable traffic demand space. Given the fact that the solvable traffic demand space is finite convex, it is relatively safe to formulate some basic conclusions regarding TDWSP's performance after an analysis of those three lines in Figure 6-24. In general, a time-dependent variation of a state-dependent routing algorithm could improve routing performance greatly in a large area within the solvable traffic demand space. Specifically,

- When the offered traffic load is not heavy in terms of the maximum traffic a network can handle, a time-dependent variation of the state-dependent routing algorithm could perform very close to the optimal boundary.
- When the offered traffic load is heavy, the performance of a time-dependent routing algorithm is degraded. However, this shortcoming is not substantial.
- If the demand forecast is at the moderate traffic level, a time-dependent routing algorithm could be insensitive to the position of the forecast. One fixed forecast may be effective enough to optimize the routing performance in a large area.

6.3.5.2 *Shape of the Solvable Traffic Demand Space*

Imagining the shape of the solvable traffic demand space in a high dimensional space is a challenging task (for example, in network #1, it is a 90-dimensional space). One thing that is certain that there should not be any irregular protuberance or hollow in it, for the solvable traffic demand space is convex.

As mentioned in section 6.3.4.1, for a well-designed network topology, the longest line from the origin to the maximum solvable traffic load should be a balancing traffic pattern, like the line from the origin to point $d_{a,1}$ in network #1, or the line from the origin to point $d_{a,2}$ in network #2 (see Figure 6-24). Lines from the origin to unbalancing traffic patterns cannot be longer than the first line before traffic demand becomes unsolvable.

The shape of the space is somewhat similar to that of an ice cream cone with smooth ice cream on top. The vertex of the cone is the point of origin. When the whole traffic load is light, the width of the cone is narrow. When the traffic load is moderate, the width of the cone is at its maximum. When the traffic load rises above moderate, the unbalanced traffic patterns become unsolvable. The greater the degree of unbalancing, the earlier the traffic pattern becomes unsolvable and the freedom of the solvable patterns decreases.

The purpose of this reasoning is to clarify the role of routing algorithms. We argue in this thesis that the accomplishment of any routing algorithm is confined within the solvable traffic demand space, which is not arbitrary and is limited in shape, like an ice cream cone with ice cream on top.

Furthermore, when the solvable traffic demand space is narrow in shape, it makes

a time-dependent routing algorithm with a fixed forecast, such as TDWSP, more feasible in the real world. TDWSP does not need a complex forecasting component to predict traffic in real time, if the solvable space is not large and if TDWSP is not sensitive to the forecast.

6.4 Advantages of the Time-Dependent Mechanism

The challenges mentioned in section 2.6 – namely, stability and scalability, robustness, and routing cost – will now be reexamined in the context of a time-dependent mechanism. As discussed below, when there is knowledge of the future traffic demand, a time-dependent mechanism can satisfy these challenges.

6.4.1 Fast Execution Path in TDWSP

TDWSP can process a QoS request faster than WSP. For most QoS requests expected by the TDWSP (refer to the pseudo code in section 6.1.2), the ingress router can make routing decisions based simply on preplanned *local* information. Only when unexpected requests arrive does TDWSP use the WSP algorithm on the residue network.

Supposing the demand forecast is roughly correct, then in Figure 6-25 there is a high possibility associated with the *yes* path and a low possibility associated with the *no* path. The cost of traversing the *no* path is higher, because it requires knowledge of the free bandwidth in every link in the network. Obtaining such knowledge or maintaining a synchronized QoS database in a large network is expensive. However, the cost of the *yes* path is much lower than the cost of the *no* path. No information from other routers is required on the *yes* path, so the ingress router can make a fast routing decision without synchronizing with other routers.

Figure 6-25. The Fast Execution Path

The existence of this *fast execution path* can mitigate all those challenges we discussed in section 2.6. For TDWSP, fewer state updates are required when network resources are allocated and deallocated, as state updates are triggered only by the *no* branch. Therefore, the stability and scalability problems are relieved. A lower frequency of state updates also decreases the chance that TDWSP will be in an inaccurate state. Thus, TDWSP is more robust than WSP. The routing cost of TDWSP associated with processing state updates and implementing techniques related to the robustness issue is decreased, so it can respond to a large amount of QoS requests promptly.

6.4.2 Insensitivity to the Sequence of QoS Requests

TDWSP has a lower standard deviation for the objective function value than WSP when the traffic load is below the level forecasted. In other words, TDWSP is less sensitive to the order of QoS requests than WSP.

Table 6-2 shows the average and standard deviation of the objective function value over 20 randomly generated traffic traces for each traffic demand. The data in this table forms part of the third line results obtained in section 6.3.4, and is therefore consistent with Figure 6-23.

For both WSP and TDWSP, standard deviation increases faster than the average value. The reason for this is that routing algorithms normally become more sensitive to the order of QoS requests as the amount of free network resources (in this case, the available bandwidth) shrinks.

Objective Function Value		WSP		TDWSP (forecast at $T_{b,2}$)	
		Average	Standard Deviation	Average	Standard Deviation
Traffic Demand (of $T_{d,2}$)	9.9%	1310	7	1097	0
	19.8%	4968	24	4219	6
	29.7%	11734	50	9916	35
	39.6%	20075	75	17016	46
	49.5%	32956	124	27976	51
	59.4%	45204	156	38387	82
	69.3%	61895	221	52563	96
	79.2%	80651	283	68604	120
	89.1%	102927	382	88012	154
	99.0%	125476	346	106719	227

**Table 6-2. Standard Deviation of Objective Function Value
Network #2, the Third Line in Fig. 6-23**

Objective Function Value		WSP		TDWSP (forecast at $T_{b,2}$)	
		Average	Standard Deviation	Average	Standard Deviation
Traffic Demand (of $T_{a,2}$)	8.4%	2460	10	2076	1
	16.8%	9511	34	7996	10
	25.2%	21312	46	17823	29
	33.6%	37242	80	31045	19
	42.0%	60158	123	50053	18
	50.4%	85217	127	70953	69
	58.8%	115411	244	97288	144
	67.2%	145282	315	127887	110
	75.6%	179596	411	164160	241
	84.0%	217847	426	205008	342
	92.4%	265907	2210	263822	2644
	99.2%	323728	4269	325099	4215

**Table 6-3. Standard Deviation of Objective Function Value
Network #2, the First Line in Fig. 6-5**

In Table 6-2, the standard deviation of TDWSP is relatively smaller than WSP under all traffic demand conditions. Table 6-3 shows another partial set of data results obtained from Figure 6-5. It shows that the majority of the standard deviations of TDWSP are smaller than those of WSP. Only when the traffic demand is at 99.2% of $T_{a,2}$ is the standard deviation of TDWSP greater than that of WSP. Note that the traffic demand at 99.2% of $T_{a,2}$ is a long way from the configured traffic forecast, which is at $T_{b,2} = 50.4\% T_{a,2}$ in both Tables 6-2 and 6-3.

Figure 6-1 provides the reason why TDWSP is less sensitive to the sequence of QoS requests than WSP. A characteristic of the preplanned tree in Figure 6-1 is that changing the sequence of QoS requests does not really create a bottleneck that prevents the preplanned tree from accepting other expected requests. In other words, the preplanned tree outlined in Figure 6-1 is highly insensitive to the sequence in which QoS requests arrive. For expected requests, TDWSP's routing decision relies on the preplanned tree; therefore, the TDWSP routing algorithm is less sensitive to the order of requests than the WSP.

In extreme conditions, when the actual traffic load is too high, many requests will be unexpected, and then TDWSP will perform WSP on the residue network, which has less available capacity. In this situation, TDWSP will be more sensitive than WSP. In normal conditions, however, if the traffic forecasts are not too far apart, it is reasonable to assume that most incoming QoS requests are expected requests, and thus TDWSP is less sensitive to the order of requests than WSP.

This feature is a bonus for the time-dependent mechanism because real network operators want to avoid rearrangement as much as possible. Supposing the traffic forecast

is acceptably correct, a time-dependent variation of any state-dependent routing algorithm will generally have a lower standard deviation with regard to routing performance when the order of requests is unknown.

6.4.3 Suitability of Distributed Implementation

For TDWSP, the offline traffic planning phase is totally offline centralized. The advantages and disadvantages of centralized or distributed online routing do not apply to the offline phase.

Once the pre-allocation plan in TDWSP has been created, this static information can be distributed to each router. For most expected requests, the router can make the routing decision locally, without needing to exchange information between routers. Therefore, the online routing phase of TDWSP is highly suitable for distributed implementation. More specifically, the modification from WSP to TDWSP could facilitate a distributed implementation for normal traffic.

6.5 Limitations of the Time-Dependent Mechanism

The time-dependent mechanism has limitations with regard to the measurement of IP traffic. A good traffic forecast needs accurate traffic measurement. However, unlike connection-oriented schema such as ATM, finding the traffic measurement for a connectionless IP is difficult. Direct measurement is extremely expensive and infeasible, in practical terms. Even flow-level measurement tools (e.g., Netflow from Cisco) would result in significant overhead on high-speed links. In addition, the forwarding tables in real Internet routers are highly dynamic and cannot be controlled by one ISP, since advertisements from neighboring domains can affect routing decisions inside one ISP through BGP protocol.

There has been a great deal of research [43] in this area. Some solutions include:

- *Tomography*. This method [44, 45] derives the data rate for every ingress-egress pair from the average data rate of every link in the network. Tomography assumes that the routing table in each router is known and remains static, although this assumption is often invalid in the context of the Internet. The most notable problem with tomography is that the traffic matrix cannot be fully decided by the data rate measured on every link. An expensive likelihood function is hence required for guessing the real traffic matrix.
- *Sampling*. This method [46] samples packets that traverse each link within a network, and regards packets sampled over a certain period of time as representative of overall traffic. A hash function is carefully designed to determine whether a packet should be sampled or not.

Although traffic measurement is not easy on the Internet, an aggregated or extreme-value measurement is possible. Moreover, by trading off between efficiency and accuracy, ISPs can generally choose an efficient method, as accuracy is not a major concern for time-dependent routing mechanisms.

6.6 Remarks

It is obvious that predicting demand can often help to allocate resources in a more efficient manner. Nevertheless, it is not so obvious whether a fixed forecast f in TDWSP can serve various traffic demands. The simulations presented in this chapter showed related performance results and were used to try to address this issue by exploring the solvable traffic demand space.

In practice, the entire solvable space does not have to be explored, because real demand shows strong daily and weekly patterns in the solvable traffic demand space and they are likely to be predictable. A few typical demands in the patterns would be sufficient to verify the effectiveness of TDWSP if such traffic pattern data is available.

Chapter 7 A Unified View of IP QoS Routing

A unified view of the QoS routing algorithm is provided in Figure 7-1. State-dependent and time-dependent routing mechanisms are divided with respect to the knowledge held by the routing algorithms.

Figure 7-1. Unified View of QoS Routing

7.1 State-Dependent Routing

The success of Internet routing protocols, such as OSPF, IS-IS and BGP, proves the effectiveness of state-dependent mechanisms on the Internet. However, state-dependent mechanisms are not fast enough to adapt to dynamic traffic when dealing with the QoS routing problem. They have problems related to scalability, robustness and routing cost. In addition, their routing performance does not approach the optimal boundary. This covers the lower part of Figure 7-1.

7.2 Time-Dependent Routing

For a long time, researchers have explored the possibility of utilizing a priori knowledge of traffic patterns to provide efficient routing services. As we move up along the knowledge axis, the complexity of the routing algorithms increases. Two choices exist in time-dependent routing: (i) *static routing*: routing analyses that assume that the traffic patterns are constant; (ii) *dynamic routing*: techniques that consider that traffic patterns vary with time [47].

7.2.1 Dynamic Routing

Usually a predictor is required in dynamic routing to learn the traffic patterns from measurement or experience. This corresponds to the upper-right corner of Figure 7-1. In this approach, a forecast is conducted periodically over time. Proposals such as MPLS-OMP [54], PBR [12], etc., have been discussed for a while, but the complexity of implementing them, compared to the simplicity of OSPF, prevents their being deployed on the Internet.

Dynamic routing is handicapped by the scalability issue addressed in section 2.6. Whether the predictor is centralized or distributed, more information exchanges are required in a network, and the volume of information, i.e., collected statistics, predicted traffic and optimized routing solutions, increases proportionally with N^2 , where N represents the number of nodes in a network.

Dynamic routing is also affected by the routing cost issue discussed in section 2.6. The computation power required for the predictor and solving the optimization problem is considerable.

7.2.2 Static Routing

In this thesis, we fill the upper-left corner of the figure with static routing, which simplifies the description of the traffic patterns using a fixed traffic forecast. An example of this, TDWSP, suggests that even a fixed forecast can fit satisfactorily into the solvable traffic demand space, and can deliver good routing performance over a wide area. TDWSP suggests that the time-dependent routing mechanism has the potential to balance the routing performance with the complexity of the routing algorithm.

In terms of the challenges to QoS routing, as mentioned in section 2.6, most can be met using a static forecast. A static forecast and optimized routing solution does not require additional information exchange in a network. Meanwhile, a static forecast does not incur additional computation cost.

7.3 Connection-Oriented vs. Connectionless Paradigm

By its nature, QoS routing always has some connection-oriented characteristics, so connection-oriented and connectionless paradigms are not part of the categorization in Figure 7-1 to confine the scope of this thesis. It is well known that there is a trade-off between higher operational costs in the connection-oriented approach and lower controllability of traffic in the connectionless approach.

However, this trade-off may change with the size of the network.

Our experience with the offline traffic planning phase indicates that the need to split one source-destination pair traffic into multiple different routes gradually vanishes as the network size increases. For example, the optimized routing solutions in the 10-node network #1 often contain two split pairs out of every nine pairs. In contrast, in the 20-node network #2 (see the routing solution in Figure 6-1), only one pair, 1-3, out of

nineteen pairs requires two alternative routes: 1-5-3 and 1-20-3. In other words, in the offline traffic planning phase, about 22% of the pairs in the 10-node network #1 have to be split, while only about 5% of the source-destination pairs in the 20-node network #2 need to be split.

To explain this phenomenon, we have derived a formula from a symmetrical network model with uniform traffic. For a network with node number N and average degree φ , McDysan presents the initial results of average routing distance α and the total number of links M in the network [34, pp. 386]:

$$\alpha(N, \varphi) \cong \frac{N}{2\varphi} \quad (7-1)$$

when $N \gg 1$ and $N \gg \varphi$, and

$$M(N, \varphi) = \frac{N \times \varphi}{2} \quad (7-2)$$

Let τ be the unidirectional traffic between every pair of nodes, namely $t_{ij} = \tau$ for all i and j in the traffic matrix. Consequently, the traffic offered to each link ρ is proportional to the average routing distance α , illustrated as follows:

$$\rho(N, \varphi) = \frac{N \times (N-1) \tau}{2} \times \frac{\alpha(N, \varphi)}{M(N, \varphi)} \quad (7-3)$$

Define γ as the ratio of the unidirectional traffic for a pair of nodes τ to the average traffic traversed on a link ρ as:

$$\gamma(N, \varphi) = \frac{\tau}{\rho(N, \varphi)} \quad (7-4)$$

Substituting Equation 7-3 into Equation 7-4, we obtain an approximate formula of ratio γ for a symmetric network model with uniform traffic:

$$\gamma(N, \varphi) \cong \frac{2\varphi^2}{N^2} \quad (7-5)$$

when $N \gg 1$ and $N \gg \varphi$.

Equation 7-5 demonstrates several facts. The ratio γ decreases with the square of the node number, if the degree of a graph is fixed. In Figure 5-1, φ of network #1 is 3.2 and N is 10, and hence its ratio γ equals 20.5%. By contrast, for the topology of network #2 in Figure 5-2, the average degree of the network is 3.7 and N is 20, so γ equals 6.8%. These numbers provide a hint associated with the question why so much of the source-destination pair traffic in network #1 has to be split in the optimal routing solution, but the pair traffic in network #2 does not have to be split, although further investigations are required.

This observation, combined with the conclusion from [48] – stating that for any given set of optimal routes (assuming optimal routes cannot be split) with respect to a set of traffic demands, a set of link weights can achieve exactly the same result by reproducing them as shortest path – reveals that in the context of QoS routing, the advantages of the connection-oriented approach are not as significant when the size of the network expands. An inference from [48] could be rephrased as follows: In the connectionless paradigm, the shortest path algorithm has high controllability over all source-destination pairs, as long as they are unsplittable. Our observation can push this inference further, using Equation 7-5: If γ is small enough, the connectionless paradigm can hold a high degree of control over the routes of every source-destination pair to approximate the optimal routing solution.

From this point of view, it is believed that a future study on connectionless time-dependent mechanisms might be promising.

Chapter 8 Conclusions

This chapter provides a summary, and recommends some possible future work.

8.1 Conclusions

TDWSP, as a case study of a time-dependent routing mechanism, provides us with a platform to demonstrate the ability and limitation of the time-dependent routing mechanism. The study also helps researchers better understand the time-dependent routing mechanism and the state-dependent mechanism.

Some conclusions of this thesis include:

1. From both the simulation and the analysis, we illustrated the inherent disadvantages of state-dependent mechanisms, due to the lack of prediction of traffic. In order to avoid rearrangement to support QoS routing, the performance of SP, WSP and SWP is not optimized when the traffic load grows. The higher the offered traffic in a network, the worse the routing performance.
2. The benefits of traffic forecasting in the time-dependent mechanism are further explained in light of its positive impact on routing performance. TDWSP outperforms WSP under almost all traffic conditions in both networks, #1 and #2. When it has some knowledge about the future traffic, TDWSP greatly relaxes the burden of QoS database synchronization among routers, and hence eases the challenges of scalability, robustness and routing cost.

3. A static forecast in TDWSP can accomplish the routing job satisfactorily. The routing performance of TDWSP is not sensitive to the position of the forecast, especially when the actual traffic load is lower than the level forecasted.

Based on the results in this thesis, we believe that a carefully designed time-dependent QoS routing algorithm could achieve better routing performance. Meanwhile, an elegant time-dependent routing algorithm could realize one of the most important legacies of the Internet – simplicity.

With regard to the time-dependent mechanism, if the actual traffic is structurally different from the forecast, the performance of the time-dependent mechanism could be worse than its counterpart, the state-dependent mechanism.

Traffic monitoring is necessary for ISPs to make sure that the network is running in a regular traffic pattern. If a significant change of traffic pattern is found by traffic monitoring, the configuration of TDWSP should be adjusted by conducting the offline traffic planning phase based on the new traffic forecast.

8.2 Implications of the Time-Dependent Mechanism

The new TDWSP routing algorithm itself is not the ultimate target of this thesis. The implications of this thesis are:

1. *Extreme Value Engineering*. Because of the asymmetrical insensitivity of the traffic forecast, it could be a nice idea to set the forecast at a high level. Nevertheless, if the setting is too high, the time-dependent routing algorithm could lose some of its adaptability to other traffic patterns because there is less bandwidth left in the residue network.

The asymmetrical insensitivity does suggest that the extreme value of the daily traffic orbit might be the focus of the traffic forecast. If the routing performance under peak traffic is good, then the routing performance during the rest of the day should be fine. A moderate value in the daily traffic orbit does not seem to be important with regard to QoS routing.

This perspective has further implications for traffic measurement. If network administrators care only about extreme amounts of traffic, the difficulties in traffic measurement will be relieved.

2. *Size of the Solvable Traffic Demand Space.* The size and shape of the solvable traffic demand space is determined if the topology of a network has been finalized. One consequence of network planning is concretizing the solvable traffic demand space from the traffic demand space. However, the solvable traffic demand space is not very large, given a topology. If the given traffic demand is outside the solvable space, the rejection of part QoS requests or packet loss is theoretically inevitable.

The mapping from the solvable space to specific routing solutions is the job of the QoS routing algorithms. If our focus is turned toward the solvable space that is not large in size, then it seems reasonable to introduce the time-dependent routing mechanism.

8.3 Future Work

The following topics are considered to be future work:

8.3.1 More Exploration of the Solvable Space

The solvable traffic demand space has not been thoroughly explored in this thesis. For example, although the three lines of traffic demand in Figure 6-14 were tested, a further line, from d_b to the upper-right corner, which implies a higher traffic level with a different traffic pattern, needs to be tested. We believe that the length of such a line (within the solvable space) cannot be long, but further validation of this is needed.

Another potential area for research is the relationship between the size of the solvable space and the size of a network. More specifically:

- How will the number of links in a network affect the size of the solvable space?
- How will the number of nodes in a network affect the size of the solvable space?
- How will different network topologies, including some extreme or special topologies, affect the size of the solvable space?

There is no general measurement to gain information about the characteristics of the solvable space. The information may be beneficial not only to QoS routing, but also to network planning.

8.3.2 Connectionless QoS Routing with Static Forecast

In this work, the focus is on time-dependent mechanisms with connection-oriented routing. Time-dependent routing with a static forecast in the context of connectionless QoS routing has yet not been investigated to a great extent. In the case of TDWSP, an optimal routing solution is used to reserve bandwidth in the networks, but this approach could be difficult to integrate with DiffServ, which is being supported by more and more networks. There might also be other approaches for utilizing traffic forecasting information in connectionless QoS routing. For instance, maintaining the

split-ratio of every source-destination pair in the optimized routing solution could be another method for carrying out time-dependent routing in the connectionless routing paradigm.

Abstract

In order to deal with some of the major challenges, QoS routing solutions can be classified into two categories, state-dependent and time-dependent, according to their awareness of the future traffic demand in the network. Compared with representative state-dependent routing algorithms, a time-dependent variation of WSP – TDWSP – is proposed in this thesis to meet those challenges, by customizing itself for a range of traffic demands. Our simulation results confirm the feasibility of traffic forecasting in the context of QoS routing, which empowers TDWSP to achieve better routing performance and to overcome QoS routing difficulties, even though completely accurate traffic prediction is not required.

The case study involving TDWSP further reveals that even a static forecast can remain effective over a large area in the solvable traffic demand space, if the network topology and the peak traffic value are given. Thus, the role of traffic forecasting in QoS routing becomes more prominent.

Acknowledgements

My thanks first go to my supervisor, Professor Chung-Horng Lung, for his guidance and help. Secondly, I would like to thank Ron Armolavicius from Nortel Networks for his feedback on an early outline of this work, and Marzia Zaman from Cistel Technology for her comments. Finally, thanks go to my family and everyone who supported and encouraged me when completing this thesis.

Table of Contents

Abstract	i
Acknowledgements	ii
Table of Contents	iii
List of Tables	vi
List of Figures	vii
List of Appendices	viii
List of Acronyms	ix
Chapter 1 Introduction	1
1.1 IP QoS Routing	1
1.1.1 The General Problem of IP Routing	2
1.1.2 Link Capacity Constraint	3
1.1.3 QoS Constraints	3
1.1.4 Other Constraints	4
1.1.5 The Complexity of QoS Routing Algorithms	4
1.2 Thesis Motivation	4
1.3 Thesis Scope	5
1.4 Thesis Contributions	6
1.5 Thesis Organization	7
Chapter 2 Related Work	8
2.1 Classifications of QoS Routing	8
2.1.1 Connection-Oriented vs. Connectionless Paradigm	8
2.1.2 State-Dependent vs. Time-Dependent Mechanism	9
2.2 Internet QoS Standards	10
2.2.1 Integrated Services (IntServ)	10
2.2.2 Differentiated Services (DiffServ)	11
2.3 Traffic Engineering	12
2.3.1 Connection-Oriented Paradigm	12
2.3.2 Connectionless Paradigm	12
2.4 Internet Traffic Forecasting	13
2.5 Examples of QoS Routing Algorithms	13
2.5.1 PNNI	13
2.5.2 WSP and SWP	14
2.5.3 Other Proposed Algorithms	15
2.6 Challenges of IP QoS Routing	16
Chapter 3 Network and Traffic Model	17
3.1 Network Model	17
3.1.1 Terms in Graph Theory	17
3.1.2 Matrix Representations	18
3.1.3 Average Degree of a Network	19
3.2 Traffic Model	19
3.2.1 Traffic Representation	19
3.2.2 Connection-Oriented Assumption	20
3.2.3 Generation of QoS Requests	20

3.3	Simplifications to QoS Constraints	21
3.3.1	Simplify Delay and Packet-Loss Constraints	21
3.3.2	Simplify Bandwidth Constraint	23
3.3.3	Effect of Self-Similar Traffic	23
3.4	Remarks	23
Chapter 4	Performance Evaluation of QoS Routing	25
4.1	Linear Objective Function	25
4.1.1	The Inequality of QoS Requests	26
4.2	Nonlinear Objective Function	30
4.2.1	Minimize Average Hop Count	30
4.2.2	Minimize Maximum Link Utilization	31
4.2.3	Other Objectives	33
4.2.4	Combination of Two Objectives	33
4.3	Remarks	36
Chapter 5	Simulations of the State-Dependent Routing Mechanism	37
5.1	Simulation	37
5.1.1	Simulation Setup	37
5.1.2	Simulation Results	41
5.2	Disadvantages of the State-Dependent Mechanism	47
5.2.1	QoS Routing Challenges in the State-Dependent Routing Context	48
5.2.2	Rearrangement Dilemma	48
5.3	Remarks	51
Chapter 6	Simulations of the Time-Dependent Routing Mechanism	53
6.1	TDWSP Algorithm	53
6.1.1	Offline Traffic Planning	54
6.1.2	Online Routing	57
6.2	Simulations	59
6.2.1	Simulation Setup	59
6.2.2	Simulation Results	61
6.2.3	Effect of Predicted Traffic Demand	66
6.2.4	Effect of n	67
6.3	Generalization of the Simulation Results	69
6.3.1	The Solvable Traffic Demand Space	69
6.3.2	First Line, from the Origin to d_a	72
6.3.3	Second Line, from d_b to d_c	72
6.3.4	Third Line, from d_d to the Origin	79
6.3.5	Some Observations about the Solvable Space	81
6.4	Advantages of the Time-Dependent Mechanism	85
6.4.1	Fast Execution Path in TDWSP	85
6.4.2	Insensitivity to the Sequence of QoS Requests	86
6.4.3	Suitability of Distributed Implementation	89
6.5	Limitations of the Time-Dependent Mechanism	89
6.6	Remarks	90
Chapter 7	A Unified View of IP QoS Routing	92
7.1	State-Dependent Routing	92
7.2	Time-Dependent Routing	93

7.2.1	Dynamic Routing	93
7.2.2	Static Routing.....	94
7.3	Connection-Oriented vs. Connectionless Paradigm.....	94
Chapter 8	Conclusions	98
8.1	Conclusions	98
8.2	Implications of the Time-Dependent Mechanism.....	99
8.3	Future Work	100
8.3.1	More Exploration of the Solvable Space	101
8.3.2	Connectionless QoS Routing with Static Forecast.....	101
References		103
Appendix 1	An Example of Nonlinear Functions in PPRN	110
Appendix 2	An Example of Optimized Routing Solutions from PPRN.....	111

List of Tables

Table 3-1.	Two Equivalent Sets of QoS Requests.....	20
Table 4-1.	1 st Solution of the Traffic Matrix T_0	28
Table 4-2.	2 nd Solution of the Traffic Matrix T_0	29
Table 6-1.	Network Performance with WSP and TDWSP.....	62
Table 6-2.	Standard Deviation of Objective Function Value	87
Table 6-3.	Standard Deviation of Objective Function Value	87

List of Figures

Figure 1-1.	Three Traces of Link Traffic.....	5
Figure 4-1.	Topology of Network #0.....	26
Figure 4-2.	Traffic Matrix of Network #0	27
Figure 4-3.	Core Router Latency Measurement with Internet Mix Traffic	32
Figure 4-4.	Nonlinear Link Cost.....	35
Figure 5-1.	Topology of Network #1	39
Figure 5-2.	Topology of Network #2	39
Figure 5-3.	Traffic Matrix of Network #1	40
Figure 5-4.	Traffic Matrix of Network #2	40
Figure 5-5.	Network Performance as Traffic Changes from Zero to $T_{a,1}$	41
Figure 5-6.	Network Performance as Traffic Changes from Zero to $T_{a,2}$	42
Figure 5-7.	Performance with 95% Confidence Interval	44
Figure 5-8.	Performance with 95% Confidence Interval	44
Figure 5-9.	Average Hop Count as Traffic Changes from Zero to $T_{a,2}$	45
Figure 5-10.	Maximum Link Utilization as Traffic Changes from Zero to $T_{a,2}$	46
Figure 5-11.	Three LSP Requests and Their Ideal Routing Solution	49
Figure 5-12.	Two More LSP Requests and Their Ideal Routing Solution.....	49
Figure 6-1.	Forecast in Fig. 6-3 and Calculated Preplanned Tree	55
Figure 6-2.	Traffic Forecast $T_{b,1}$ of Network #1	59
Figure 6-3.	Traffic Forecast $T_{b,2}$ of Network #2	60
Figure 6-4.	Network Performance with WSP and TDWSP.....	61
Figure 6-5.	Network Performance with WSP and TDWSP.....	62
Figure 6-6.	Performance with 95% Confidence Interval	63
Figure 6-7.	Performance with 95% Confidence Interval	64
Figure 6-8.	Average Hop Count with WSP and TDWSP	64
Figure 6-9.	Maximum Link Utilization with WSP and TDWSP	65
Figure 6-10.	Network Performance with Different Forecast	66
Figure 6-11.	Network Performance When $n=3$	67
Figure 6-12.	Average Hop Count with $n=2$ and $n=3$	68
Figure 6-13.	Maximum Link Utilization with $n=2$ and $n=3$	69
Figure 6-14.	Positions of Three Lines of Traffic Demand in the Solvable Space	71
Figure 6-15.	Traffic Matrix $T_{c,2}$ of Network #2	74
Figure 6-16.	Traffic Matrix $T_{c,1}$ of Network #1	75
Figure 6-17.	Network Performance as Traffic Changes from $T_{b,1}$ to $T_{c,1}$	76
Figure 6-18.	Network Performance as Traffic Changes from $T_{b,2}$ to $T_{c,2}$	76
Figure 6-19.	Performance with 95% Confidence Interval	77
Figure 6-20.	Performance with 95% Confidence Interval	77
Figure 6-21.	Maximum Link Utilization as Traffic Changes from $T_{b,2}$ to $T_{c,2}$	78
Figure 6-22.	Network Performance as Traffic Changes from Zero to $T_{d,1}$	80
Figure 6-23.	Network Performance as Traffic Changes from Zero to $T_{d,2}$	81
Figure 6-24.	Three Lines of Traffic Demand in the Solvable Space	82
Figure 6-25.	The Fast Execution Path.....	86
Figure 7-1.	Unified View of QoS Routing.....	92

List of Appendices

Appendix 1	An Example of Nonlinear Functions in PPRN	110
Appendix 2	An Example of Optimized Routing Solutions from PPRN.....	111

List of Acronyms

ARIMA	Auto Regressive Integrated Moving Average
CBR	Constraint-Based Routing
DiffServ	Differentiated Services
DSCP	Differentiated Services Code Point
DTL	Designated Transit List
IGP	Interior Gateway Protocol
IntServ	Integrated Services
ISIS	Intermediate System - Intermediate System
LSP	Label Switched Path
MIRA	Minimum Interference Routing Algorithm
MPLS	MultiProtocol Label Switching
OSPF	Open Shortest Path First
PBR	Profiled-Based Routing
PHB	Per Hop Behavior
PNNI	Private Network-Network Interface
POP	Point of Presence
QoS	Quality of Service
SP	Shortest Path
SWP	Shortest-Widest Path
TDWSP	Time-Dependent Widest-Shortest Path
WSP	Widest-Shortest Path

References

- [1] R. Braden, D. Clark & S. Shenker, "Integrated Services in Internet Architecture: An Overview," *RFC 1633*, IETF, June 1994.
- [2] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang & W. Weiss, "An Architecture for Differentiated Service," *RFC 2475*, IETF, December 1998.
- [3] E. Rosen, A. Viswanathan & R. Callon, "Multiprotocol Label Switching Architecture," *RFC 3031*, IETF, January 2001.
- [4] D. O. Awduche, "MPLS and Traffic Engineering in IP Networks," *IEEE Comm. Magazine*, December 1999, pp. 42-47.
- [5] O. Younis & S. Fahmy, "Constraint-Based Routing in the Internet: Basic Principles and Recent Research," *IEEE Comm. Surveys & Tutorials*, 3rd Quarter 2003.
- [6] E. Alitman, T. Basar, T. Jimenez & N. Shimkin, "Competitive Routing in Networks with Polynomial Cost," *Proc. of INFOCOM 2000. 19th Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 3, March 2000, pp. 1586-1593.
- [7] J. Wang & K. Nahrstedt, "Hop-by-Hop Routing Algorithms for Premium-Class Traffic in DiffServ Networks," *Proc. of INFOCOM 2002. 21st Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 2, June 2002, pp. 705-714.

- [8] R. Guérin & A. Orda, "QoS-Based Routing in Networks with Inaccurate Information: Theory and Algorithms," *IEEE/ACM Trans. Net.*, June 1999, pp. 350-364.
- [9] G. Apostolopoulos, R. Guérin, S. Kamat, A. Orda & S. K. Tripathi, "Intra-Domain QoS Routing in IP Networks: A Feasibility and Cost/Benefit Analysis," Special Issue of *IEEE Network*, vol. 13, no. 5, September 1999.
- [10] G. Apostolopoulos, R. Guérin, S. Kamat & S. Tripathi, "Quality of Service Based Routing: A Performance Perspective," *ACM Comp. Commun. Review*, 28(4), September 1998.
- [11] A. Elwalid, C. Jin, S. Low & I. Widjaja, "MATE: MPLS Adaptive Traffic Engineering," *Proc. of INFOCOM 2001. 20th Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 3, April 2001, pp. 1300-1309.
- [12] S. Suri, M. Waldvogel, D. Bauer & P. R. Warkhede, "Profile-based routing and Traffic Engineering," *Computer Communications*, vol. 26, 2003, pp. 351-365.
- [13] A. Orda & A. Sprintson, "QoS Routing: The Precomputation Perspective," *Proc. of INFOCOM 2000. 19th Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 1, March 2000, pp. 128-136.
- [14] D. Medhi, "QoS Routing with Path Caching - A Framework and Network Performance," *IEEE Communications Magazine*, December 2002
- [15] R. Guérin, A. Orda & D. Williams, "QoS Routing Mechanisms and OSPF Extensions," *Proc. of Second Global Internet Miniconference*, 1997.

- [16] M. Ericsson, M. G. C. Resende & P. M. Pardalos, "A Genetic Algorithm for the Weight Setting Problem in OSPF Routing," *Journal of Combina. Optim.*, vol. 6, 2002, pp. 299-333.
- [17] M. Kodialam & T. V. Lakshman, "Minimum Interference Routing with Applications to MPLS Traffic Engineering," *Proc. of INFOCOM 2000. 19th Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 2, March 2000, pp. 884-893.
- [18] O. Bonaventure, P. Trimintzios, G. Pavlou, et al., "Internet Traffic Engineering," in *Quality of Future Internet Services*, M. Smirnov et al. (eds), Springer-Verlag, 2003, pp. 118-134.
- [19] D. Bertsekas & R. Gallager, *Data Networks*, Prentice-Hall, 1986, pp. 374-403.
- [20] A. Odlyzko, "Data Networks are Lightly Utilized, and Will Stay That Way," *Technical Report*, AT&T Labs Research, Florham Park, New Jersey, July 1998.
- [21] K. Papagiannaki, N. Taft, Z. Zhang & C. Diot, "Long-Term Forecasting of Internet Backbone Traffic - Observations and Initial Models," *Proc. of INFOCOM 2003. 22nd Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 2, April 2003, pp. 1178-1188.
- [22] J. Castro & N. Nabona, "PPRN," Retrieved February 14, 2004, from Statistics and Operations Research Dept. at Universitat Politecnica de Catalunya Web site: <http://www-eio.upc.es/~jcastro/pprn.html>.
- [23] TCL, Retrieved February 14, 2004, from Tcl Developer Exchange Web site: <http://www.tcl.tk/>.

- [24] NS-2, Retrieved February 14, 2004, from Information Sciences Institute at University of Southern California Web site: <http://www.isi.edu/nsnam/ns/>.
- [25] K. Park & W. Willinger, "Self-Similar Network Traffic: An Overview," in *Self-Similar Network Traffic And Performance Evaluation*, K. Park & W. Willinger (eds), John Wiley & Sons, 2000, pp. 1-38.
- [26] S. Shenker, C. Partridge & R. Guérin, "Specification of Guaranteed Quality of Service," *RFC 2212*, IETF, September 1997.
- [27] R. Braden et al., "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification," *RFC 2205*, IETF, September 1997.
- [28] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan & G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels," *RFC 3209*, IETF, December 2001.
- [29] B. Fortz, J. Rexford & M. Thorup, "Traffic Engineering with Traditional IP Routing Protocols," *IEEE Comm. Magazine*, October. 2002, pp. 118-124.
- [30] ATM Forum, "Private Network-Network Interface Specification Version 1.0," *af-pnni-0055.00*, March 1996.
- [31] B. Jamoussi, L. Andersson, R. Callon, R. Dantu, L. Wu, P. Doolan, T. Worster, N. Feldman, A. Fredette, M. Girish, E. Gray, J. Heinanen, T. Kilty & A. Malis, "Constraint-Based LSP Setup using LDP," *RFC 3212*, IETF, January 2002.
- [32] C. Semeria, "Traffic Engineering for the New Public Network," Retrieved February 14, 2004, from Juniper Networks Web site: http://www.juniper.net/solutions/literature/white_papers/200004.pdf, 2000.
- [33] R. K. Ahuja, T. L. Magnanti & J. B. Orlin, *Network Flows: Theory, Algorithms and Applications*, Prentice-Hall, Upper Saddle River, NJ, USA, 1993.

- [34] D. McDysan, *QoS & Traffic Management in IP & ATM Networks*, McGraw-Hill, USA, 2000.
- [35] I. Matta, A. Bestavros & M. Krunz, "Load Profiling Based Routing for Guaranteed Bandwidth Flows," *European Transactions on Telecommunications (ETT)*, vol. 10, no. 2, March/April 1999.
- [36] E. Osborne & A. Simha, "Offline Strategic TE Design," in *Traffic Engineering with MPLS*, Cisco Press, July 2002, pp. 427-431.
- [37] R. Guérin, H. Ahmadi & M. Naghshineh, "Equivalent Capacity and Bandwidth Allocation," *IEEE Journal on Selected Areas in Communications*, September 1991.
- [38] M. Schwartz, *Broadband Integrated Networks*, Prentice-Hall, 1996.
- [39] Light Reading Inc., "Internet Core Router Test", Retrieved February 14, 2004, from Light Reading web site: http://www.lightreading.com/document.asp?doc_id=4009, March 6, 2001.
- [40] J. Castro & N. Nabona, "An implementation of linear and nonlinear multicommodity network flows," *European Journal of Operational Research*, vol 92, 1996, pp. 37-53.
- [41] D. Bertsekas, *Nonlinear Programming*, 2nd edition, Athena Scientific, 1999.
- [42] G. Ahn & W. Chun, "MPLS Network Simulator," Retrieved February 14, 2004, from Chungnam National University Web site <http://flower.ce.cnu.ac.kr/~fog1/mns/>, February 16, 2001.
- [43] M. Grossglauser & J. Rexford, "Passive Traffic Measurement for IP Operations," in *The Internet as a Large-Scale Complex System*, Oxford University Press, 2004.

- [44] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya & C. Diot, "Traffic Matrix Estimation: Existing Techniques and New Directions," *Proceedings of the 2002 SIGCOMM Conference*, 2002, pp. 161-174.
- [45] A. Nucci, R. Cruz, N. Taft & C. Diot, "Design of IGP Link Weight Changes for Estimation of Traffic Matrices," *Proc. of INFOCOM 2004. 23rd Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, Hong Kong, March 2004.
- [46] N. G. Duffield & M. Grossglauser, "Trajectory Sampling for Direct Traffic Observation," *IEEE/ACM Transactions on Networking*, 9(3), June 2001, pp. 280-292.
- [47] G. Ash, *Dynamic Routing in Telecommunications Networks*, McGraw-Hill, 1998.
- [48] Y. Wang, Z. Wang & L. Zhang, "Internet Traffic Engineering without Full Mesh Overlaying," *Proc. of INFOCOM 2001. 20th Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 1, April 2001, pp. 565-571.
- [49] S. B. Gershwin, R. V. Laue & E. Wolman, "Peak-Load Traffic Administration for a Rural Line Concentrator," 7th ITC, 1973.
- [50] D. H. Barnes, "Extreme Value Engineering of Small Switching Offices," 8th ITC, 1979.
- [51] S. Chen & K. Nahrstedt, "An Overview of Quality of Service Routing for Next-Generation High-Speed Networks: Problems and Solutions," *IEEE Network*, vol. 12, no. 6, Nov./Dec. 1998.
- [52] A. Juttner, B. Szviatovszki, I. Mecs & Z. Rajko, "Lagrange Relaxation-Based Method for the QoS Routing Problem," *Proc. of INFOCOM 2001. 20th Annual*

Joint Conf. of the IEEE Computer and Communications Societies. IEEE, vol. 2, April 2001, pp. 859-868.

- [53] T. Korkmaz and M. Krunz, "Multi-Constrained Optimal Path Selection," *Proc. of INFOCOM 2001. 20th Annual Joint Conf. of the IEEE Computer and Communications Societies. IEEE*, vol. 2, April 2001, pp. 834-843.
- [54] C. Villamizer, "MPLS – OMP," First International Workshop on Multi-protocol Label Switching, November 13, 1998.

Appendix 1 An Example of Nonlinear Functions in PPRN

This is an example of the nonlinear objective functions defined in the PPRN package. In this case, the file *net1_n2.c* calculates the Equation 4-4 in network topology #1 when the *n* parameter is 2.

```
main(argc,argv)
int argc;
char *argv[];
{
    char strdat[80];
    int nolin=1, iestat=0;
    void funobj();

    if (argc!=2) {
        printf("Usage: nolintest <*.dat>\n");
        exit(0);
    }
    strcpy(strdat,argv[1]);

    pprnl_(&nolin,strdat,&iestat,funobj);
    printf("exit= %d\n",iestat);
    exit(-1);
}

/* dummy routine */
void funobj(n,k,x,f,g,nstate)
int *n,*k,*nstate;
double *x,*f,*g;
{
    int dim,commodity,arc;
    double raux,c_sum[*n+1];
    double
    capacity[33]={1.0,2400.0,2400.0,2400.0,2400.0,2400.0,600.0,2400.0,2400.0,2400.0,24
    00.0,600.0,2400.0,2400.0,2400.0,2400.0,2400.0,2400.0,2400.0,600.0,600.0,600.0,600.0,600.
    0,600.0,2400.0,2400.0,2400.0,2400.0,2400.0,2400.0,600.0,2400.0,2400.0};

    dim=*n+1;

    *f=0.0;
    for (arc=1;arc<=*n;arc++) {
        c_sum[arc]=0.0;
        for (commodity=0;commodity<*k;commodity++) {
            raux=(x+(arc+commodity*dim));
            c_sum[arc]=c_sum[arc]+raux;
        }
        c_sum[arc]=c_sum[arc]*100.0/capacity[arc];
        *f=*f+c_sum[arc]*c_sum[arc];
        for (commodity=0;commodity<*k;commodity++) {
            *(g+(arc+commodity*dim))=2.0*c_sum[arc]/capacity[arc];
        }
    }
}
```

Appendix 2 An Example of Optimized Routing Solutions from PPRN

This is an example of the optimized routing solution calculated by the PPRN package. In this case, the network topology is network #1, and the traffic demand $T_{b,l}$ (see Figure 6-2) is solved by the objective function programmed in Appendix 1.

```
***** PARAMETERS *****
OBJECTIVE FUNCTION TYPE      : non linear
MAXIMUM NUMBER OF ITERATIONS : 2147483647
REINVERSION BASIS FREQUENCY  : 50
NUMBER NON BASIC FOR PRICING : 23
PRICING TYPE AT PHASE 1      : 1
PRICING TYPE AT PHASE 2      : 1
INFINITY VALUE CONSIDERED    : 1.E+32
OUTPUT FREQUENCY              : 10
SOLUTION OBTAINED AT PHASE 0 : optimal
BLACK LIST AT PHASE 0        : inactive
CONNECTIVITY CHECK           : inactive
L-U PIVOT TOLERANCE          : 1.E-11
PERCENTAGE ACTIVES           : 0.5
SPARSITY ETA-MATRICES        : 0.2
SPARSITY Q (WORKING MATRIX)  : 0.09
SPARSITY PATHS               : 0.8
DESCENT DIRECTION            : quasi Newton
MAXIMUM DIMENSION OF R'R     : 105
QUASI ACTIVES DIRECTION      : inactive
REDUCTION SUBSPACE GRADIENT  : 0.5
OPTIMALITY TOLERANCE         : 1.E-06
LINESEARCH TOLERANCE         : 0.1
T5 CONTROL                   : inactive
*****
```

STARTING PHASE 0.

```
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 1  ITERATIONS: 5
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 2  ITERATIONS: 3
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 3  ITERATIONS: 4
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 4  ITERATIONS: 7
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 5  ITERATIONS: 8
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 6  ITERATIONS: 6
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 7  ITERATIONS: 7
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 8  ITERATIONS: 7
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 9  ITERATIONS: 8
INITIAL OPTIMAL TREE FOUND FOR COMMODITY 10 ITERATIONS: 11
```

STARTING PHASE 1. INFEASIBILITIES= 0.

STARTING PHASE 2. OBJECTIVE FUNCTION= 49853.1563

```
Ph.....It.....Nb->Sup.....Rg.....Bs->Nb....Sup-
>Nb/Bs.....Fobj...Ncmp...Nsup.....Alph.....Nrg...Cv.
 2      10      11      2      30      2  0.46254500E+05      0
 1  0.124E+03 0.314E-01 FTFF
 2      20      25      6      32      6  0.44201349E+05      0
 2  0.833E+02 0.892E-02 TTFF
```

```

xxx the current search has evaluated f(x) 20 times
2 30 29 8 -0.332E-01 0.44084710E+05 0
  4 0.521E+01 0.333E-01 TTTT
2 40 6 3 -0.487E-01 0.43588044E+05 0
  5 0.796E+01 0.507E-01 TTTF
2 50 0.43420347E+05 0
  6 0.129E+02 0.333E-03 FTTF
2 60 0.43367442E+05 0
  7 0.386E+02 0.776E-03 TTFF
2 70 30 1 -0.843E-02 0.43361167E+05 0
  10 0.152E+01 0.837E-02 TTTT
2 80 32 5 25 5 0.43312444E+05 0
  10 0.246E+02 0.609E-02 TTFF

```

```

oooooooooooooooooooooooooooo
o                               o
o      E X I T                o
o                               o
oooooooooooooooooooooooooooo
STATUS: OPTIMAL SOLUTION FOUND

```

```

ITERATION: 87
PHASE: 2
PHASE 0 ITERATIONS: 66
PHASE 1 ITERATIONS: 0
PHASE 2 ITERATIONS: 87
DIMENSION OF WORKING MATRIX: 0
NUMBER OF SUPERBASICS: 12
OBJECTIVE FUNCTION EVALUATIONS: 769
RG NORM: 5.5206588E-07
PI NORM: 1.
RG NORM/PI NORM: 5.5206588E-07
OBJECTIVE FUNCTION VALUE: 43259.0375

```

1. VARIABLES.

....arc..commodity...	status	arc.....	flow.....	capacity
1	1	TREE	516.367709	+INFINITY
1	2	LOWER	0.000000	+INFINITY
1	3	LOWER	0.000000	+INFINITY
1	4	LOWER	0.000000	+INFINITY
1	5	LOWER	0.000000	+INFINITY
1	6	LOWER	0.000000	+INFINITY
1	7	LOWER	0.000000	+INFINITY
1	8	LOWER	0.000000	+INFINITY
1	9	LOWER	0.000000	+INFINITY
1	10	LOWER	0.000000	+INFINITY
2	1	TREE	453.632291	+INFINITY
2	2	LOWER	0.000000	+INFINITY
2	3	LOWER	0.000000	+INFINITY
2	4	LOWER	0.000000	+INFINITY
2	5	LOWER	0.000000	+INFINITY
2	6	LOWER	0.000000	+INFINITY
2	7	LOWER	0.000000	+INFINITY
2	8	LOWER	0.000000	+INFINITY
2	9	LOWER	0.000000	+INFINITY
2	10	LOWER	0.000000	+INFINITY
3	1	LOWER	0.000000	+INFINITY
3	2	TREE	528.969925	+INFINITY
3	3	LOWER	0.000000	+INFINITY
3	4	TREE	147.922841	+INFINITY
3	5	LOWER	0.000000	+INFINITY
3	6	TREE	5.000227	+INFINITY
3	7	LOWER	0.000000	+INFINITY
3	8	LOWER	0.000000	+INFINITY
3	9	LOWER	0.000000	+INFINITY
3	10	LOWER	0.000000	+INFINITY
4	1	LOWER	0.000000	+INFINITY
4	2	TREE	405.000000	+INFINITY
4	3	LOWER	0.000000	+INFINITY
4	4	LOWER	0.000000	+INFINITY

4	5	LOWER	0.000000	+INFINITY
4	6	TREE	33.000000	+INFINITY
4	7	LOWER	0.000000	+INFINITY
4	8	TREE	292.000000	+INFINITY
4	9	TREE	241.000000	+INFINITY
4	10	TREE	190.367530	+INFINITY
5	1	LOWER	0.000000	+INFINITY
5	2	TREE	391.636525	+INFINITY
5	3	LOWER	0.000000	+INFINITY
5	4	LOWER	0.000000	+INFINITY
5	5	LOWER	0.000000	+INFINITY
5	6	TREE	43.999773	+INFINITY
5	7	LOWER	0.000000	+INFINITY
5	8	TREE	179.000000	+INFINITY
5	9	TREE	130.000000	+INFINITY
5	10	LOWER	0.000000	+INFINITY
6	1	TREE	11.000000	+INFINITY
6	2	TREE	68.000000	+INFINITY
6	3	TREE	5.000000	+INFINITY
6	4	TREE	75.445967	+INFINITY
6	5	TREE	33.000000	+INFINITY
6	6	LOWER	0.000000	+INFINITY
6	7	LOWER	0.000000	+INFINITY
6	8	TREE	33.000000	+INFINITY
6	9	LOWER	0.000000	+INFINITY
6	10	LOWER	0.000000	+INFINITY
7	1	LOWER	0.000000	+INFINITY
7	2	TREE	520.393550	+INFINITY
7	3	LOWER	0.000000	+INFINITY
7	4	TREE	533.000000	+INFINITY
7	5	TREE	309.000000	+INFINITY
7	6	TREE	33.000000	+INFINITY
7	7	LOWER	0.000000	+INFINITY
7	8	LOWER	0.000000	+INFINITY
7	9	LOWER	0.000000	+INFINITY
7	10	LOWER	0.000000	+INFINITY
8	1	LOWER	0.000000	+INFINITY
8	2	TREE	68.363475	+INFINITY
8	3	TREE	93.000000	+INFINITY
8	4	LOWER	0.000000	+INFINITY
8	5	LOWER	0.000000	+INFINITY
8	6	SUPER	0.000227	+INFINITY
8	7	TREE	32.000000	+INFINITY
8	8	TREE	129.000000	+INFINITY
8	9	TREE	112.000000	+INFINITY
8	10	TREE	82.000000	+INFINITY
9	1	SUPER	68.367709	+INFINITY
9	2	LOWER	0.000000	+INFINITY
9	3	TREE	288.000000	+INFINITY
9	4	LOWER	0.000000	+INFINITY
9	5	LOWER	0.000000	+INFINITY
9	6	LOWER	0.000000	+INFINITY
9	7	SUPER	10.273445	+INFINITY
9	8	LOWER	0.000000	+INFINITY
9	9	LOWER	0.000000	+INFINITY
9	10	TREE	315.255558	+INFINITY
10	1	LOWER	0.000000	+INFINITY
10	2	LOWER	0.000000	+INFINITY
10	3	TREE	493.000000	+INFINITY
10	4	LOWER	0.000000	+INFINITY
10	5	LOWER	0.000000	+INFINITY
10	6	LOWER	0.000000	+INFINITY
10	7	TREE	137.000000	+INFINITY
10	8	LOWER	0.000000	+INFINITY
10	9	LOWER	0.000000	+INFINITY
10	10	TREE	201.632470	+INFINITY

....arc..commodity...	status_arc.....	flow.....	capacity
11	1	TREE	15.719498
11	2	TREE	69.000000
11	3	TREE	95.000000

11	4	SUPER	35.554031	+INFINITY
11	5	TREE	59.000000	+INFINITY
11	6	LOWER	0.000000	+INFINITY
11	7	LOWER	0.000000	+INFINITY
11	8	LOWER	0.000000	+INFINITY
11	9	LOWER	0.000000	+INFINITY
11	10	LOWER	0.000000	+INFINITY
12	1	TREE	339.280502	+INFINITY
12	2	SUPER	108.606450	+INFINITY
12	3	TREE	520.000000	+INFINITY
12	4	TREE	215.000001	+INFINITY
12	5	TREE	177.000000	+INFINITY
12	6	LOWER	0.000000	+INFINITY
12	7	LOWER	0.000000	+INFINITY
12	8	LOWER	0.000000	+INFINITY
12	9	LOWER	0.000000	+INFINITY
12	10	LOWER	0.000000	+INFINITY
13	1	LOWER	0.000000	+INFINITY
13	2	LOWER	0.000000	+INFINITY
13	3	LOWER	0.000000	+INFINITY
13	4	TREE	1161.368808	+INFINITY
13	5	LOWER	0.000000	+INFINITY
13	6	LOWER	0.000000	+INFINITY
13	7	LOWER	0.000000	+INFINITY
13	8	LOWER	0.000000	+INFINITY
13	9	LOWER	0.000000	+INFINITY
13	10	LOWER	0.000000	+INFINITY
14	1	LOWER	0.000000	+INFINITY
14	2	LOWER	0.000000	+INFINITY
14	3	LOWER	0.000000	+INFINITY
14	4	TREE	1011.631192	+INFINITY
14	5	LOWER	0.000000	+INFINITY
14	6	LOWER	0.000000	+INFINITY
14	7	LOWER	0.000000	+INFINITY
14	8	LOWER	0.000000	+INFINITY
14	9	LOWER	0.000000	+INFINITY
14	10	LOWER	0.000000	+INFINITY
15	1	LOWER	0.000000	+INFINITY
15	2	SUPER	98.636525	+INFINITY
15	3	LOWER	0.000000	+INFINITY
15	4	TREE	184.000000	+INFINITY
15	5	TREE	160.000000	+INFINITY
15	6	TREE	10.999773	+INFINITY
15	7	LOWER	0.000000	+INFINITY
15	8	LOWER	0.000000	+INFINITY
15	9	LOWER	0.000000	+INFINITY
15	10	LOWER	0.000000	+INFINITY
16	1	TREE	109.632291	+INFINITY
16	2	LOWER	0.000000	+INFINITY
16	3	LOWER	0.000000	+INFINITY
16	4	LOWER	0.000000	+INFINITY
16	5	TREE	635.000000	+INFINITY
16	6	LOWER	0.000000	+INFINITY
16	7	LOWER	0.000000	+INFINITY
16	8	LOWER	0.000000	+INFINITY
16	9	LOWER	0.000000	+INFINITY
16	10	LOWER	0.000000	+INFINITY
17	1	LOWER	0.000000	+INFINITY
17	2	LOWER	0.000000	+INFINITY
17	3	LOWER	0.000000	+INFINITY
17	4	SUPER	368.631192	+INFINITY
17	5	TREE	463.000000	+INFINITY
17	6	LOWER	0.000000	+INFINITY
17	7	LOWER	0.000000	+INFINITY
17	8	LOWER	0.000000	+INFINITY
17	9	LOWER	0.000000	+INFINITY
17	10	LOWER	0.000000	+INFINITY
18	1	TREE	184.000000	+INFINITY
18	2	LOWER	0.000000	+INFINITY
18	3	TREE	266.000000	+INFINITY
18	4	LOWER	0.000000	+INFINITY

18	5	TREE	459.000000	+INFINITY
18	6	LOWER	0.000000	+INFINITY
18	7	TREE	78.000000	+INFINITY
18	8	LOWER	0.000000	+INFINITY
18	9	LOWER	0.000000	+INFINITY
18	10	SUPER	24.632470	+INFINITY
19	1	LOWER	0.000000	+INFINITY
19	2	LOWER	0.000000	+INFINITY
19	3	LOWER	0.000000	+INFINITY
19	4	LOWER	0.000000	+INFINITY
19	5	LOWER	0.000000	+INFINITY
19	6	TREE	183.000000	+INFINITY
19	7	TREE	42.446072	+INFINITY
19	8	LOWER	0.000000	+INFINITY
19	9	LOWER	0.000000	+INFINITY
19	10	LOWER	0.000000	+INFINITY
20	1	LOWER	0.000000	+INFINITY
20	2	LOWER	0.000000	+INFINITY
20	3	LOWER	0.000000	+INFINITY
20	4	TREE	42.445967	+INFINITY
20	5	LOWER	0.000000	+INFINITY
20	6	TREE	49.000000	+INFINITY
20	7	LOWER	0.000000	+INFINITY
20	8	LOWER	0.000000	+INFINITY
20	9	LOWER	0.000000	+INFINITY
20	10	LOWER	0.000000	+INFINITY

....arc..commodity...	status_arc.....	flow.....	capacity	
21	1	LOWER	0.000000	+INFINITY
21	2	LOWER	0.000000	+INFINITY
21	3	LOWER	0.000000	+INFINITY
21	4	LOWER	0.000000	+INFINITY
21	5	LOWER	0.000000	+INFINITY
21	6	LOWER	0.000000	+INFINITY
21	7	TREE	274.273445	+INFINITY
21	8	LOWER	0.000000	+INFINITY
21	9	LOWER	0.000000	+INFINITY
21	10	LOWER	0.000000	+INFINITY
22	1	LOWER	0.000000	+INFINITY
22	2	LOWER	0.000000	+INFINITY
22	3	LOWER	0.000000	+INFINITY
22	4	LOWER	0.000000	+INFINITY
22	5	LOWER	0.000000	+INFINITY
22	6	LOWER	0.000000	+INFINITY
22	7	TREE	47.446072	+INFINITY
22	8	LOWER	0.000000	+INFINITY
22	9	TREE	33.000000	+INFINITY
22	10	TREE	11.000000	+INFINITY
23	1	LOWER	0.000000	+INFINITY
23	2	LOWER	0.000000	+INFINITY
23	3	LOWER	0.000000	+INFINITY
23	4	LOWER	0.000000	+INFINITY
23	5	LOWER	0.000000	+INFINITY
23	6	TREE	44.000000	+INFINITY
23	7	TREE	145.280483	+INFINITY
23	8	LOWER	0.000000	+INFINITY
23	9	LOWER	0.000000	+INFINITY
23	10	LOWER	0.000000	+INFINITY
24	1	LOWER	0.000000	+INFINITY
24	2	LOWER	0.000000	+INFINITY
24	3	LOWER	0.000000	+INFINITY
24	4	LOWER	0.000000	+INFINITY
24	5	LOWER	0.000000	+INFINITY
24	6	LOWER	0.000000	+INFINITY
24	7	SUPER	16.280483	+INFINITY
24	8	TREE	744.000000	+INFINITY
24	9	TREE	562.000000	+INFINITY
24	10	SUPER	73.111972	+INFINITY
25	1	LOWER	0.000000	+INFINITY
25	2	TREE	191.000000	+INFINITY
25	3	LOWER	0.000000	+INFINITY

25	4	TREE	241.000000	+INFINITY
25	5	TREE	130.000000	+INFINITY
25	6	LOWER	0.000000	+INFINITY
25	7	LOWER	0.000000	+INFINITY
25	8	TREE	77.000000	+INFINITY
25	9	LOWER	0.000000	+INFINITY
25	10	LOWER	0.000000	+INFINITY
26	1	LOWER	0.000000	+INFINITY
26	2	TREE	89.393550	+INFINITY
26	3	LOWER	0.000000	+INFINITY
26	4	LOWER	0.000000	+INFINITY
26	5	LOWER	0.000000	+INFINITY
26	6	LOWER	0.000000	+INFINITY
26	7	LOWER	0.000000	+INFINITY
26	8	TREE	557.000000	+INFINITY
26	9	LOWER	0.000000	+INFINITY
26	10	LOWER	0.000000	+INFINITY
27	1	LOWER	0.000000	+INFINITY
27	2	LOWER	0.000000	+INFINITY
27	3	LOWER	0.000000	+INFINITY
27	4	LOWER	0.000000	+INFINITY
27	5	LOWER	0.000000	+INFINITY
27	6	LOWER	0.000000	+INFINITY
27	7	LOWER	0.000000	+INFINITY
27	8	LOWER	0.000000	+INFINITY
27	9	TREE	639.000000	+INFINITY
27	10	LOWER	0.000000	+INFINITY
28	1	LOWER	0.000000	+INFINITY
28	2	LOWER	0.000000	+INFINITY
28	3	LOWER	0.000000	+INFINITY
28	4	LOWER	0.000000	+INFINITY
28	5	LOWER	0.000000	+INFINITY
28	6	LOWER	0.000000	+INFINITY
28	7	LOWER	0.000000	+INFINITY
28	8	LOWER	0.000000	+INFINITY
28	9	TREE	746.000000	+INFINITY
28	10	LOWER	0.000000	+INFINITY
29	1	LOWER	0.000000	+INFINITY
29	2	LOWER	0.000000	+INFINITY
29	3	LOWER	0.000000	+INFINITY
29	4	LOWER	0.000000	+INFINITY
29	5	LOWER	0.000000	+INFINITY
29	6	LOWER	0.000000	+INFINITY
29	7	LOWER	0.000000	+INFINITY
29	8	TREE	347.000000	+INFINITY
29	9	TREE	322.000000	+INFINITY
29	10	TREE	690.888028	+INFINITY
30	1	SUPER	16.280502	+INFINITY
30	2	LOWER	0.000000	+INFINITY
30	3	LOWER	0.000000	+INFINITY
30	4	SUPER	0.000001	+INFINITY
30	5	LOWER	0.000000	+INFINITY
30	6	LOWER	0.000000	+INFINITY
30	7	LOWER	0.000000	+INFINITY
30	8	TREE	50.000000	+INFINITY
30	9	TREE	75.000000	+INFINITY
30	10	TREE	48.000000	+INFINITY

....arc..commodity...	status	arc.....	flow.....	capacity
31	1	TREE	129.000000	+INFINITY
31	2	LOWER	0.000000	+INFINITY
31	3	TREE	218.000000	+INFINITY
31	4	LOWER	0.000000	+INFINITY
31	5	LOWER	0.000000	+INFINITY
31	6	LOWER	0.000000	+INFINITY
31	7	TREE	66.280483	+INFINITY
31	8	LOWER	0.000000	+INFINITY
31	9	LOWER	0.000000	+INFINITY
31	10	TREE	233.111972	+INFINITY
32	1	TREE	112.000000	+INFINITY
32	2	LOWER	0.000000	+INFINITY

32	3	TREE	210.000000	+INFINITY
32	4	LOWER	0.000000	+INFINITY
32	5	LOWER	0.000000	+INFINITY
32	6	TREE	33.000000	+INFINITY
32	7	TREE	42.000000	+INFINITY
32	8	LOWER	0.000000	+INFINITY
32	9	LOWER	0.000000	+INFINITY
32	10	TREE	349.000000	+INFINITY

2. MUTUAL CAPACITY CONSTRAINTS.

....arc...	status_slack.....	flow.....	slack.....	mutual_capacity
1	BASIC	516.367709	1883.632291	2400.000000
2	BASIC	453.632291	1946.367709	2400.000000
3	BASIC	681.892993	1718.107007	2400.000000
4	BASIC	1161.367530	1238.632470	2400.000000
5	BASIC	744.636298	1655.363702	2400.000000
6	BASIC	225.445967	374.554033	600.000000
7	BASIC	1395.393550	1004.606450	2400.000000
8	BASIC	516.363702	1883.636298	2400.000000
9	BASIC	681.896712	1718.103288	2400.000000
10	BASIC	831.632470	1568.367530	2400.000000
11	BASIC	274.273529	325.726471	600.000000
12	BASIC	1359.886954	1040.113046	2400.000000
13	BASIC	1161.368808	1238.631192	2400.000000
14	BASIC	1011.631192	1388.368808	2400.000000
15	BASIC	453.636298	1946.363702	2400.000000
16	BASIC	744.632291	1655.367709	2400.000000
17	BASIC	831.631192	1568.368808	2400.000000
18	BASIC	1011.632470	1388.367530	2400.000000
19	BASIC	225.446072	374.553928	600.000000
20	BASIC	91.445967	508.554033	600.000000
21	BASIC	274.273445	325.726555	600.000000
22	BASIC	91.446072	508.553928	600.000000
23	BASIC	189.280483	410.719517	600.000000
24	BASIC	1395.392455	1004.607545	2400.000000
25	BASIC	639.000000	1761.000000	2400.000000
26	BASIC	646.393550	1753.606450	2400.000000
27	BASIC	639.000000	1761.000000	2400.000000
28	BASIC	746.000000	1654.000000	2400.000000
29	BASIC	1359.888028	1040.111972	2400.000000
30	BASIC	189.280504	410.719496	600.000000
31	BASIC	646.392455	1753.607545	2400.000000
32	BASIC	746.000000	1654.000000	2400.000000