NETWORK DESIGN UNDER DEMAND UNCERTAINTY

by

Koonlachat Meesublak

B.S. in Electrical Engineering, The Johns Hopkins University, 1997

M.S. in Electrical Engineering (Systems), The University of Michigan, Ann Arbor, 1999

Submitted to the Graduate Faculty of
the School of Information Sciences in partial fulfillment
of the requirements for the degree of

Doctor of Philosophy

University of Pittsburgh

2007

UNIVERSITY OF PITTSBURGH

DEPARTMENT OF INFORMATION SCIENCE AND TELECOMMUNICATIONS

This dissertation was presented

by

Koonlachat Meesublak

It was defended on

April 17, 2007

and approved by

Dr. David Tipper, University of Pittsburgh

Dr. Richard Thompson, University of Pittsburgh

Dr. Prashant Krishnamurthy, University of Pittsburgh

Dr. Bryan A. Norman, University of Pittsburgh

Dr. Deep Medhi, University of Missouri - Kansas City

Dissertation Director: Dr. David Tipper, University of Pittsburgh

NETWORK DESIGN UNDER DEMAND UNCERTAINTY

Koonlachat Meesublak, PhD

University of Pittsburgh, 2007

A methodology for network design under demand uncertainty is proposed in this dissertation. The uncertainty is caused by the dynamic nature of the IP-based traffic which is expected to be transported directly over the optical layer in the future. Thus, there is a need to incorporate the uncertainty into a design model explicitly. We assume that each demand can be represented as a random variable, and then develop an optimization model to minimize the cost of routing and bandwidth provisioning. The optimization problem is formulated as a nonlinear Multicommodity Flow problem using Chance-Constrained Programming to capture both the demand variability and levels of uncertainty guarantee. Numerical work is presented based on a heuristic solution approach using a linear approximation to transform the nonlinear problem to a simpler linear programming problem. In addition, the impact of the uncertainty on a two-layer network is investigated. This will determine how the Chance-Constrained Programming based scheme can be practically implemented. Finally, implementation guidelines for developing an updating process are provided.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1.0 INTRODUCTION

Due to rapid growth of the Internet, IP-based traffic is expected to be the majority of all the traffic carried over data networks in the future. Today's data networks can be typically described as a four-layer stack. The IP traffic is transported over an ATM layer, over a SONET/SDH layer, and over a WDM layer. The main drawback of this stack is that it does not scale well with increasing traffic [1]. Although the ATM layer provides great traffic engineering functionality (e.g., optimizing network performance and usage), ATM overhead significantly decreases transmission efficiency. Furthermore, it is unlikely that ATM switches will be made to support capacities beyond OC-48c (2.488 Gb/s) due to problems in processing at the packet level (i.e., segmentation and reassembly) [2]. The SONET/SDH layer provides transport functionality, but its equipment is very expensive and an upgrade is unlikely [3]. Thus, eliminating the intermediate layers on the stack would lead to significant savings in terms of equipment and operation costs. It is expected that the ATM and SONET/SDH layers will be bypassed, resulting in a two-layer stack, where IP traffic can be transported directly over the WDM layer. Such an architecture is known as *IP-over-WDM* [4], [5]. This network evolution, as illustrated in Figure 1.1, is made possible by technology breakthroughs in fast photonic switching together with the *Generalized Multiprotocol Label Switching* (GMPLS) framework.

Unlike traditional circuit-switched networks, an IP-over-WDM network allows dynamic routing and bandwidth provisioning in the WDM layer. The amount of capacity reserved for demands carried on each network link can change abruptly; this behavior is termed *demand uncertainty.* A short-term network design for handling such uncertainty is needed. However, this should not be confused with a long-term planning design (i.e., in months or years time

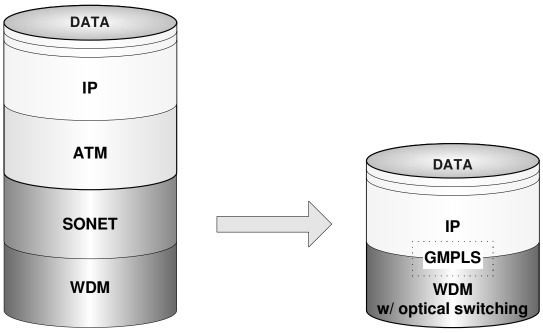


Figure 1.1: Evolution of data networks

frame) such as a capacity expansion problem that will determine the network topology or capacities on network links. In that problem, the focus is on demand forecasting and how to design a network to tolerate the future outcomes and to minimize penalties as a result of forecasting errors. The long-term model needs to consider demand growth rates, economic factors, the expected future technologies, as well as possible demand scenarios that might occur in the future. In contrast, the network design problem discussed in this dissertation has a different goal since we assume that the network architecture and capacities are given. This can be classified as a short-term virtual topology design, and the objective is to minimize the cost of routing and bandwidth allocation subject to network constraints on random demands.

Based on the assumption that an aggregated traffic demand on a network link can be modeled as a Gaussian random variable, the demand characteristic consisting of the mean and the variance can be used in a new approach, which is an alternative to the mean-based reservation and the peak-based reservation approaches. Obviously, the mean-based design is used for handling traffic with low variation around the mean, and therefore a one-time optimization process can be used. In addition, when the traffic has different patterns depending on time period, but the patterns are completely known or highly predictable, network resources can be optimized according to the demand pattern in each period, and thus there are different optimal solutions for each of time periods. This type of design is referred to as the Multi-Hour network design [6]. For example, the optimal routing for the 9-12 A.M period may be different than that of the 1-5 P.M. period.

In contrast, the peak-based approach intends to cover most of the traffic variations by requiring the bandwidth be at least the highest rate known by traffic measurement or estimation processes. This approach can be costly because the peak rate could be many times higher than the average. Thus, this approach is suitable for a network with plentiful resources, where cost is not the main issue. In this dissertation, we propose a technique whereby the variation above the mean level could be handled statistically based on the specified levels of guarantee. The Chance-Constrained Programming (CCP) approach [7] is adapted to this application for constructing bandwidth bounds and specifying levels of probabilistic guarantee for random demands. The resulting problem is a nonlinear Multicommodity Flow (MCF) optimization problem [8]. However, solving an exact nonlinear optimization problem is not practical for dynamic routing and provisioning operations. Therefore, there is a need to develop heuristic algorithms, which are simple, but produce good-quality solutions. The second part of the dissertation concentrates on how to integrate the model and algorithms into an uncertainty framework, including a two-layer model that could be applied to future networks. The research contributions are as follows:

1. A new use of the Chance-Constrained Programming optimization in a communication network context, considering both uncertainty and service guarantees.
2. New mathematical formulations for network design under traffic uncertainty are developed. This framework is expected to be applied to the design of virtual networks in IP-over-WDM network.
3. A set of fast efficient heuristic algorithms are developed to solve such design problems with reasonably good quality. The solution algorithms are intended for usage in a time-critical environment such as on-line optimization update. Furthermore, this could be applied in solving other similar nonlinear optimization problems.
4. Implementation guidelines for developing a corrective process for handling traffic changes are provided.

The dissertation is organized as follows. Chapter 2 provides background on current network architecture. Chapter 3 discusses the related current literature, and introduces a design approach using the Chance-constrained Programming technique. Chapter 4 presents the proposed network design technique, and mathematical programming models and formulations. Chapter 5 describes heuristic solution algorithms and their performance. Chapter 6 discusses the implementation issues and an adaptive approach in a two-layer network. Chapter 7 concludes the dissertation and provides future research suggestions.

2.0 BACKGROUND ON NETWORK ARCHITECTURES

2.1 OPTICAL LAYER

The *optical layer*, defined by the International Telecommunication Union-Telecommunication Standardization Sector (ITU-T), is composed of three sub-layers: the *optical channel* (OCh) layer, the *optical multiplex section* (OMS) layer, and the *optical transmission section* layer [9]. The OCh layer provides routing functionality for end-to-end connections known as *lightpaths*. The OMS layer supports optical signals at an aggregate level (i.e., the fiber level, which can contain multiple wavelengths). The optical transmission section is responsible for how to transmit wavelengths over a fiber. This can be done via two multiplexing techniques: optical time division multiplexing (OTDM) and wavelength division multiplexing (WDM). WDM is preferred over OTDM due to the higher bandwidth it can provide [10]. In WDM, data is transmitted over wavelengths simultaneously on a fiber by using multiple channels at different wavelengths. In the rest of this dissertation, the terms *optical layer* and *WDM layer* will be used interchangeably.

A logical connection between routers in the IP layer is provided at the OCh layer (or OXC layer) by a lightpath, which is established by optical switches called *optical cross-connects* (OXCs). Each lightpath traverses a number of point-to-point WDM systems known as *optical transport systems* (OTSs), which is a pair of WDM terminals. When an IP router requests a connection, OXCs provision (establish) a lightpath by cross-connecting an incoming OTS channel (wavelength) to an outgoing OTS-channel. An OXC may or may not have wavelength conversion capability (i.e., able to change the wavelength of an incoming optical signal). If wavelength converters are equipped along a route, a lightpath may traverse one or more wavelengths.

Initially WDM technology has been deployed to support point-to-point systems, where optical signals are added and dropped at every intermediate node. Each node requires optical-electronic-optical (O-E-O) conversion of signals. Such a network is called an *opaque network*. The advantages are that the impairment accumulation along the path can be prevented and performance monitoring is simple. However, this system cannot achieve full transparency (i.e., the independence of protocols, formats, and bit rates). Furthermore, transmission speed is limited due to the electronic bottleneck of O-E-O conversion and large buffers are required at intermediate nodes [11]. The future optical network is expected to be *all-optical*, where there is no O-E-O conversion along a lightpath. Optical signals can be bypassed at a transit node by optical add/drop multiplexer (OADM), which can drop some traffic at a particular point and let the rest bypass the intermediate node. The main advantages of all-optical networks are transparency support, lower transmission time, and cost savings from intermediate devices. OXCs, with dynamic routing, provisioning and survivability capabilities, will have a major role in future optical networking, using the GMPLS control plane.

2.2 IP-MPLS

Multiprotocol Label Switching (MPLS) technology has been developed by the Internet Engineering Task Force (IETF) to support applications such as Traffic Engineering, Virtual Private Networks, Quality of Service (QoS) for differentiated services, and restoration at the IP layer. MPLS is a connection-oriented scheme used to forward an IP packet based on a label appended to an IP header. The content of an IP packet is not examined by intermediate MPLS routers, and thus this fast-forwarding capability can reduce routing congestion at electronic routers. Routers in an MPLS network are called *Label Switching Routers* (LSRs). When IP packets enter a MPLS network at the ingress LSR, the LSR attaches labels to packets based on their *Forwarding Equivalent Class* (FEC). An FEC can be assigned based on a variety of factors (e.g., QoS class, security). A common approach is to base the FEC on the destination address of packets. The packets with the same FEC will be forwarded in the same way across the network along a connection called a *Label Switched Path* (LSP). An LSP can be viewed as analogous to a virtual path connection in ATM. An MPLS label is locally significant to an LSR, and labels can be stacked in a last-in, first-out manner. It can be pushed (appended to a packet), swapped to another label, or popped off. MPLS supports hierarchal routing by these stacking operations. Lower-order LSPs can be merged into higher-order LSPs, and the topmost label signifies the highest-order LSP. When the packets exit the MPLS network at the egress LSR, the labels are removed and the packets are then routed via traditional IP routing protocols.

To establish an LSP, LSRs perform path calculation via a *constraint-based routing protocol.* It aims to find the optimal path that does not violate a set of constraints such as performance constraints (e.g., maximum available bandwidth) or administrative constraints (e.g., a set of links to be excluded). Thereafter, a path is set up via a signaling protocol such as Constraint-Based Label Distribution Protocol (CR-LDP) or Resource ReSerVation Protocol with traffic engineering extension (RSVP-TE). A comprehensive tutorial on MPLS is given in [2].

2.3 GMPLS AND FUTURE NETWORK MODELS

2.3.1 GMPLS

In order to achieve the integration of the IP and optical domains, a common control plane for routing and signaling is needed. This began with the development of *Multiprotocol Lambda Switching* (MPλS) [12], which is an extension of MPLS to be used with OXCs in the optical domain. The initial goal of MPλS was to provide dynamic lightpath provisioning. This is because provisioning has been done manually via network management, which consists of planning and reconfiguration processes. Such processes are time-consuming and therefore not suitable for the dynamic nature of IP demands. With MPλS, an OXC can perform data forwarding in a way similar to LSR packet forwarding, except that the label is implicitly assigned, specified by a wavelength used for a lightpath.

A generalized version of MPλS is being developed called *Generalized Multiprotocol Label Switching* (GMPLS). GMPLS extends MPLS to support non-packet switching, including timeslot, wavelength, and fiber switching. The proposed GMPLS control plane provides the following functions [13]:

1. Neighbor discovery by the proposed *Link Management Protocol* (LMP) for detection of links between neighboring OXCs and keeping track of link status (e.g., up/down, bandwidth availability).

2. Link state dissemination by an Interior Gateway Protocol (IGP) such as Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS) with traffic engineering extensions for distribution of topology and resource availability of the network

3. Route computation by a constraint-based routing protocol (as in MPLS) for calculating working and backup lightpaths.

4. Path management (i.e., establishing, maintaining, and tearing down lightpaths) by a signaling protocol such as RSVP-TE or CR-LDP.

5. Fault management (protection and restoration).

GMPLS allows sharing of optical network topology and resource availability information between OXCs and LSRs. The resource information exchanged includes wavelength and bandwidth availability, physical layer constraints (e.g., maximum distance and bit error rate limited by a network), and link protection and diversity for failure protection. With the above GMPLS capabilities, OXCs can dynamically provision connections requested from the client IP routers. How the clients request connections from the optical layer is described by IP-over-WDM network models in the next section.

2.3.2 IP-over-WDM Models

There are two main proposed IP-over-WDM network models: an overlay model and the peer model [1], [12]. These models are classified based on how network information is interchanged between layers.

Overlay Model

In the overlay model or client-server model, the IP domain (client) and the optical domain (server) have independent control planes for routing and signaling. Routers interact with the optical core network via the *user-to-network interface* (UNI), while the subnets in the optical core network communicate with each other via the *network-to-network interface* (NNI). The internals of the optical core is hidden from the IP domain. In this model, an IP router requests a connection without any knowledge of the optical core network, and the optical layer only provides connections to the IP layer, i.e., OXCs setup lightpath connections for IP connections. In summary, the operations in both the optical and the IP layers are independent of each other. The overlay model is considered as the first generation of the IP-over-WDM network.

Peer Model

In this model, both the IP and optical domains use the same control plane, and there is no difference between UNI and NNI in terms of routing and signaling. The full functionality of GMPLS previously described is assumed in this model. Routers are *peers* of OXCs and they can exchange topology and resource information. Unlike the overlay model, the routers can take benefit of having the full knowledge of the optical core network. Thus, resource efficiency or performance goals could be satisfied by routing with such knowledge.

On a technical basis, the overlay model will be feasible before the peer model. The peer model needs more intelligence in optical networking and requires a lot of standardization. Moreover, since the core networks may belong to many service providers, some topology information may need to be kept private. However, the peer model can eliminate the duplication of control plane functionality of the two layers. During the transformation from overlay to peer, there would be an intermediate model known as the *Augmented Model.* This model would still have two separate control planes, one each for the IP and optical domains, but some information from the optical core network would be exchanged with the IP domain.

2.4 SUMMARY

This chapter provides relevant background information on network architectures and technologies. Nowadays there are many difficulties in implementing an IP-over-WDM network due to differences in the IP and optical layers. The main issues include routing, resource management, signaling, and survivability. The uncertainty framework in this dissertation focuses on the routing and resource management aspects.

3.0 NETWORK DESIGN UNDER DEMAND UNCERTAINTY

3.1 MOTIVATION

In Chapter 2, the promising future of IP-over-WDM networks with the MPLS/GMPLS protocol was described. However, one of the main issues in designing such a network is that the traditional optimization models are no longer appropriate due to the following changes.

1. Change in network architecture

In the current IP/ATM/SONET/WDM network stack, a hierarchical multiplexing effect reduces the *dynamic* variability of the demand seen at the bottom layer (i.e. the WDM layer). In other words, the demand at the WDM layer appears to be *static* in a sense that the capacity allocated between each two points is fixed over a long period of time. However, placing the IP traffic directly over the WDM layer will have an impact on the design since it is now necessary to incorporate the *dynamic* nature of the IP demand. Thus, future networks may need to apply the concept of dynamic allocation of bandwidth into the lower layers.

2. Change in bandwidth-allocation duration

The traditional design for a circuit-switched voice network assumes that a virtual trunk connection setup for a demand lasts for a long period of time (e.g. on the order of months or years). This assumption will no longer be true for a data network since in the future a user may request a connection in a short-term basis (e.g., on the order of weeks or days, even hours). Therefore, bandwidth in the lower layers will be considered as a volatile resource. This change will make the optimization of network bandwidth allocation much more complicated.

Thus, there is a need for a new methodology to incorporate dynamic factors into a design model. In this dissertation, we consider the demand dynamic in terms of the volume variation, and will use the term *demand uncertainty* throughout the the dissertation to characterize this variation.

3.2 OPTIMIZATION MODELS

Based on how demand uncertainty is handled in an optimization problem, network models can be roughly classified into two categories as follows.

3.2.1 Deterministic Models

Deterministic models assume that the demand volume is known or can be forecasted with small errors. Therefore, it need not consider characteristics of uncertain parameters. This simplicity is preferred in traditional network designs with a use of the *mean value* to summarize the data. The more conservative approach uses the *worst-case* value to guarantee performance under peak loads. However, in general some parameters can not be estimated with high accuracy. Optimization using the mean-value or worst-case scenario could yield large error bounds [14].

3.2.2 Stochastic Models

Stochastic models handle uncertainty explicitly. It considers changes in parameters such as randomness of demand, and then incorporates this factor into the design. In the Operations Research (OR) literature, Stochastic Programming (SP) and its variations (e.g, Robust Optimization and Chance-constrained Programming) have many applications for decision-making with uncertainty. Note that SP is quite different than sensitivity analysis.

In sensitivity analysis, the goal is to determine the impact of parameter perturbations on the solutions; it does not specify how to control such uncertainty. Therefore, sensitivity analysis is a reactive approach, while the SP is a proactive approach [15]. The following describes three major approaches that incorporate uncertainty within optimization models.

3.2.2.1 Stochastic Programming with recourse SP with recourse is a classic optimization problem that is used for planning with uncertainty. It assumes that the probability distributions of random variables (i.e., uncertain parameters) are known or can be estimated. The simplest form of this model is a two-stage recourse model. Each stage consists of a set of decision variables. For example, in a capacity expansion problem, the amount of capacity to be allocated is a decision variable in the first stage. Capacity constraints in this stage could be the mean value based on past experience or initial estimation. Since the outcome may not be as expected, the capacity assigned in the first stage may not meet the future demand described by a random variable. Therefore, the second stage involves taking a corrective or recourse action to compensate for the planning error, e.g., buying more capacity (which is the decision variable in the second stage). Hence, the goal is to minimize the cost of the first-stage decision plus the expected cost of the second-stage recourse decision. Birge and Louveaux [14] extensively discuss the SP with recourse problem, in both the modeling and solution approaches.

3.2.2.2 Robust Optimization Robust Optimization (RO) seeks to determine an optimal solution that is *robust* or immune to uncertainty [15], [16]. Uncertain parameters are described by a set of possible outcomes or scenarios, and the RO model considers a tradeoff between feasibility and optimality for all the given scenarios. Although this is similar to the basic SP with recourse technique, there are some differences. First, while the SP model takes into account the expected value of a random variable, the RO model considers not only the expected value, but also the variance or the higher moments so that it can minimize the impact of variability. Second, RO explicitly uses a penalty function for the purpose of risk aversion. The penalty function, which is also called the utility or regret function, is used to penalize infeasibility or errors resulting from inexact planning.

3.2.2.3 Chance-constrained Programming Developed by Charnes and Cooper [7], Chance-constrained Programming (CCP) handles uncertainty by permitting infeasibility of a constraint with uncertain parameters within some specified probability value. Unlike the RO and SP with recourse models, the CCP is a *Here-and-Now*-type decision making model. Therefore, it does not require the compensation or penalty cost for the unwanted future outcomes, which is unknown or difficult to estimate. This dissertation emphasizes a network design approach using the CCP model, which will be discussed in detail in subsequent sections.

3.3 RELATED WORK

Typical designs for telecommunications networks do not consider demand as a random variable. Therefore, applications using stochastic optimization are rare in the literature. Currently, there are just a few research papers in the literature focusing on problems such as routing or capacity expansion given probabilistic information of the future demand. Sen et al.[17] uses the number of requested connections as a random variable in a capacity planning problem for an incremental design of a telephone switching network. This is done by using the two-stage SP with recourse model. The goal is to minimize unserved demands and to seek to add capacities (or lines) subject to a capacity budget. Kennington et al.[18] apply the RO approach to minimize optical equipment cost in a WDM network. This work uses a regret function to penalize errors resulting from a demand forecast (i.e., in case of overestimate or underestimate). The uncertain future demand is described as a set of scenarios with given probabilities of occurrence. Similarly, Leung and Grover [19, 20] use scenario-based demand as an input to a two-stage SP with recourse problem for the design of working and spare capacity allocation in a survivable optical network.

All of these studies use a set of demand scenarios. The main disadvantage is that it is difficult to generate scenarios and associated probabilities that represent the future outcomes. Furthermore, If the number of scenarios is too small, the model could overlook some important cases. On the other hand, if the number of scenarios is large, it would be computationally impractical for solving a large-scale problem. In addition, the cost of performing the corrective action in the future (as in the SP) or the regret function (as in the RO) are always difficult to obtain. Although the SP with recourse and RO models are suitable for some planning horizon problems, this dissertation focuses on a model that can be applied to Traffic Engineering (TE) applications such that a decision on routing and bandwidth provisioning will be made on a short-term basis. Thus, the CCP model is better suited to this type of application. With probabilistic guarantees (as will be discussed next), the uncertainty will always be carried, but the amount of bandwidth provisioned will be based on statistics through the level of guarantee and the characteristic of the demand.

The work by Mitra and Wang [21] is an economic model with revenue risk consideration. This scheme optimizes bandwidth provisioning and path selection under demand uncertainty by maximizing the revenue from serving demands, using a probabilistic distribution (truncated Gaussian distribution) of demands as inputs. Based on the mean-risk analysis, the scheme can help a carrier make a decision to determine the tradeoff between revenue and the risk of having revenue falling below the threshold. For example, the carrier may take risk by provisioning bandwidth for the uncertainty in order to obtain more revenue, or giving up some revenue if it is not paid off. This is based on the fact that bandwidth can be sold in wholesale or retail, and the price of a volatile demand could be high; this is the revenue risk. Then, the objective is to maximize the mean revenue with the risk of revenue shortfall. Therefore, based on the tradeoff decision, the uncertainty is not always carried.

A CCP's application related to our work is described in a study of Medova [22] for dimensioning and traffic management of an ATM network. Uncertain demand (in a connection-level) is guaranteed with a fixed call rejection (or call blocking) rate. The author assumes that the call blocking rate is very small, and then use an approximation technique to eliminate the chance-constraint. The resulting approximated constraint (or deterministic equivalent) contains no probabilistic information of random demand. Unlike the current literature, this dissertation incorporates statistical information on a random demand (i.e., mean and variance) with a probabilistic level-guarantee of network resource into an optimization model. The next section describes the formulation of CCP problems in detail.

3.4 FORMULATION OF CHANCE-CONSTRAINED PROGRAMMING PROBLEMS

Consider N random variables, denoted by ξ_i's (e.g., random demands). Let x_i be a decision variable associated with cost c_i. In a network context, x_i could represent capacity or bandwidth allocated on link i within bound b_i. Suppose that the bandwidth can be used to carry the random demand with some specified probability value, a simple generic formulation of a CCP problem is as follows:

$$\min_{x_i} \sum_{i=1}^{N} c_i x_i \tag{3.1}$$

subject to:

$$\begin{aligned} P\,(\;x_i \geq \xi_i\;) \quad &\geq \quad \alpha_i, \quad \forall i = 1 \ldots N && (3.2)\\ x_i \quad &\leq \quad b_i,\; \forall i = 1 \ldots N && (3.3)\\ x_i \quad &\geq \quad 0,\; \forall i = 1 \ldots N && (3.4)\end{aligned}$$

where $0 \leq \alpha_i \leq 1$. Constraint (3.2) is a chance-constraint used to ensure that capacity or bandwidth allocated on link i is greater than or equal to an uncertain parameter ξ_i for at least probability α_i. This constraint implies that the model can permit constraint violations up to a probability limit of $1 - \alpha_i$. For example, the quantity $1 - \alpha$ can be viewed as a call blocking probability, and can be included in a model to ensure the random demand can be carried with an α-level of guarantee.

Constraints of the form of (3.2), namely

$$P\,(\;x \geq \xi\;) \quad \geq \quad \alpha \tag{3.5}$$

can be converted into a deterministic equivalent ([23], [24]) as follows. Suppose that a random variable ξ has a cumulative distribution function $\Phi(.)$, and its inverse transform $\Phi^{-1}(.)$. Let $\Phi^{-1}(\alpha) = K$. Thus, $P\,(\ K \geq \xi\ ) = \alpha$. It is true that $P\,(\ x \geq \xi\ ) \geq \alpha$ if and only if $x \geq K$. The value of K will be determined from the probabilistic information of a random variable. In this work, we assume that the traffic demand ξ is normally distributed with mean μ and variance σ^2. Standardizing inequality (3.5) using the unit normal random variable z, we obtain $P\left(\ \frac{x-\mu}{\sigma} \geq z\ \right) \geq \alpha$ if and only if $\frac{x-\mu}{\sigma} \geq \Phi^{-1}(\alpha)$, which can be written as

$$x \ \geq \ \mu \ + \ \Phi^{-1}(\alpha)\sigma \tag{3.6}$$

Hence, this is a deterministic equivalent of the chance-constraint (3.5). It can be interpreted as follows. *To guarantee that the link can support the random demand at least the 100(α)%, we need to allocate capacity at least the amount of* $\Phi^{-1}(\alpha)\sigma$ *beyond the mean (μ) of the demand.* In the minimization problem presented in the next chapter, the required link capacity for an α-level guarantee is based on (3.6), but the minimum value is used:

$$x \ = \ \mu \ + \ \Phi^{-1}(\alpha)\sigma \tag{3.7}$$

and the *uncertainty factor*, $\Phi^{-1}(\alpha)\sigma$, is the key difference when compared with a deterministic design. It captures both the level of uncertainty-guarantee (α) and variability of the demand volume (σ).

3.5 SUMMARY

This chapter discusses the need for an uncertainty framework due to the changes in network architecture and the dynamic of demands. Then, the related optimization models and literature on demand uncertainty was described. Finally, the basic Chance-Constrained Programming concept was introduced.

4.0 UNCERTAINTY MODELS

4.1 ISSUES AND ASSUMPTIONS

The network models used in this dissertation are one-layer and two-layer models. The one-layer model could represent a generic network, and will be used in the first stage of development of our uncertainty framework. We assume that the demand between each two points is a large aggregation of traffic. Given a set of traffic demands, the logical or virtual topology is to be determined, and then all the demands are routed over virtual paths in the topology. The term virtual path or virtual connection implies that the nodes that lie in such path are virtually connected although they may not be physically connected to each other.

The two-layer network as illustrated in Figure 4.1 could represent an *overlay model* (see Section 2.3.2), which is considered as the first generation of the IP-over-WDM network. The top layer consists of IP router nodes, and the nodes in bottom layer represent optical routers or OXCs. The overlay model implies that each layer performs routing calculations separately. How the nodes in the top layer are connected is described by a virtual topology. The relation of these two layers is defined by a mapping, where a virtual or logical link between routers in the IP layer is provided by a physical connection (i.e., fiber link) between OXCs in the lower layer. Selecting which wavelengths on a fiber can be performed via the Routing and Wavelength Assignment (RWA) algorithms extensively discussed in the literature [10] and will not be included in the optimization process in this dissertation.

In summary, this optimization problem can be classified as virtual topology design, and our goal is to minimize the cost of dynamic routing and bandwidth allocation subject to network

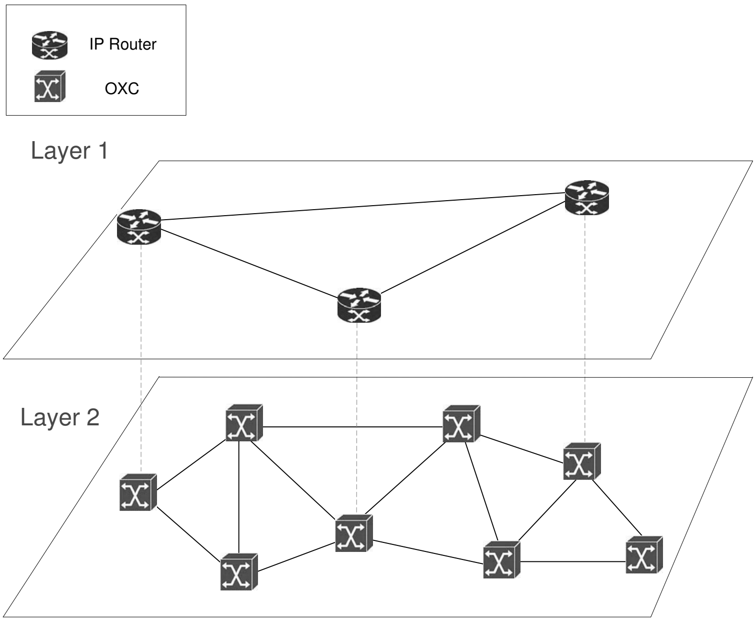


Figure 4.1: A two-layer network

constraints on random demands. To design a network supporting dynamic IP-based demand, there are three main implementation issues that need to be considered:

1. How much demand uncertainty can the network tolerate?
2. How can guarantees be provided to an uncertain demand?
3. How can a network designer/manager obtain statistical information on an uncertain demand?

The first issue arises due to limitations on network resources and switching/multiplexing technologies. The second issue is closely related to the first issue. However, this depends not only on resource limitations, but also on a network service provider's policy. In order to achieve high resource utilization, the service provider could treat the uncertain portion (i.e., the variation beyond an expected volume) of the traffic similar to IP best-effort traffic. From the chance-constraint, the mean value of demand volume, which is called the *certainty factor*, is always 100% guaranteed. In contrast, the demand variability represented by a function of variance and a parameter α (called the *uncertainty factor*) is probabilistically guaranteed based on the assigned value of α.

With the use of a chance-constraint as discussed in the previous chapter, both the first two issues can be addressed directly. First, the chance-constraint is used to construct an approximate bound on the bandwidth required for a random demand. Second, a level of uncertainty-guarantee is provided according to a probabilistic value assigned on each network link; thus, this will ensure a *per-link or link-level guarantee.* Alternatively, one can provide an *end-to-end* or *per-path* guarantee. In this case, each traffic flow receives its own level of guarantee regardless of the links used in a route. Traffic demands can be specified by class and have the same level of guarantee for the same class, which complies with the DiffServ or MPLS QoS routing standard. In this case, the design is simpler than that of the per-link case since it is a linear optimization problem as shown in Section 4.3.2, unlike the per-link case which is non-linear as will be shown in Section 4.3.1.

For the third issue, we assume that a network manager can obtain a probability distribution for demands by traffic monitoring, or making some approximations. We expect that the characteristics of the future Internet will support a dynamic bandwidth-on-demand. Therefore, an optimization solution that is optimal in some period of time may not be optimal in another period, and thus a re-optimization process is needed. This depends on the uncertainty bound approximated by a chance-constraint. If the bound is good enough to cover the traffic variation, the update need not be made often. As will be explained in Chapter 6, sometimes it is not necessary to re-optimize the entire network due to the costs associated with the process (e.g., computational cost, traffic disruption cost, etc.). However, at some point, when the bounds obtained by CCP approximation no longer satisfy demand requirements and service level agreements, one will have to request or add more bandwidth on some existing paths if it is allowed, reroute or reconfigure some paths, or re-optimize the network (this would be the last choice). For example, bandwidth based on the initial distributions of the traffic may need to be adjusted when there is a significant traffic change. Depending on the network policy, the network manager can set a specified threshold related to the level of traffic change so that an adaptive process can decide whether to re-optimize the network or not. Generally, a small traffic change can be resolved with rerouting or adding more bandwidth to the existing paths, but for a significant change, re-optimization would be needed.

The next question is what is an appropriate choice for modeling the traffic distribution. The appropriate probability distribution modeling traffic demands depends on the level of traffic aggregation. Sufficient aggregation means that large aggregations of a number of traffic streams occurs over long time scales [25]. As networks get large, the traffic will be sufficiently aggregated and the distribution can be approximated by the Normal or Gaussian model. If this assumption is not satisfied, a non-Guassian model such as the M-Pareto model could be used instead [26], [27]. In short, the M-Pareto model can be used to study traffic when there is a smaller level of aggregation (i.e., a few flows). It can be used to investigate traffic based on a per-flow basis, especially for bursty Internet traffic. This could be useful in predicting network queuing performance. This model is basically a Poisson process, but the rate λ represents the length of traffic burst, and has a Pareto distribution. This model can be used to design a network with self-similar or fractal traffic, which is characterized by burstiness on different timescales. It is known that the more self-similar the traffic, the longer the queue size. The consequence is that there will be greater packet losses and network delays; thus, buffer size, path length (or the number of hops), link bandwidth, and other factors related to QoS network design must be included. The formulation based on the M-Pareto model is more complicated than that of the Normal model, but its deterministic equivalent can be derived similarly. In fact, the Normal distribution can be used to approximate the M-Pareto model when λ is large [26], [27]. This dissertation will consider large aggregate of traffic case, and thus the Normal distribution will be used. Independence of traffic demands are also assumed for simplicity. However, if the demands are not independent, the deterministic equivalent can be adjusted by including a covariance factor.

The last important concern is the complexity of the problem. As can be seen in the following sections, the computational time to solve the problem grows exponentially with the network size. In general, the computational time can be improved by reducing the number of choices that a decision variable can take. As shown in an example in Chapter 5, the problem considered in this dissertation is a path-based routing design (i.e., a route is selected from a given set of candidate paths for each demand pair). In general, reducing the number of paths in a path set, rather than keeping all the possibilities, will improve the computational time with some reduction in the quality of solutions. This approach has been advocated in the literature [28], especially when the solution process should meet a time-critical requirement. In addition to the reduction in an input, what significantly affects the computational time are the problems themselves (e.g., highly-constrained, nonlinear, or large-scale problems), which will be described in Chapter 5.

4.2 NETWORK DESIGN PROCEDURE

The design model in this dissertation is based on the basic problem stated as follows.

Given network information and a demand matrix, determine the routes and the amount of bandwidth to be allocated on such routes so that the total network cost subject to network constraints is minimized.

The network design problem consists of two main tasks: routing and bandwidth allocation, as illustrated in Figure 4.2. The routing procedure determines routes or paths for traffic demand with uncertainty considerations. The bandwidth allocation procedure determines the amount of bandwidth reservation to support demand uncertainty with the required guarantee levels. These two tasks can be included in a single mathematical programming problem, which will be discussed in the next section. The required inputs to the design problem are as follows:

- Demand matrix and demand statistics (mean and variance)
- Uncertainty-guarantee levels
- Network topology
- Cost information (e.g., bandwidth cost, fixed charge costs, etc.)
- Resource constraints

The routing part requires an equivalent bandwidth calculation of the uncertainty based on

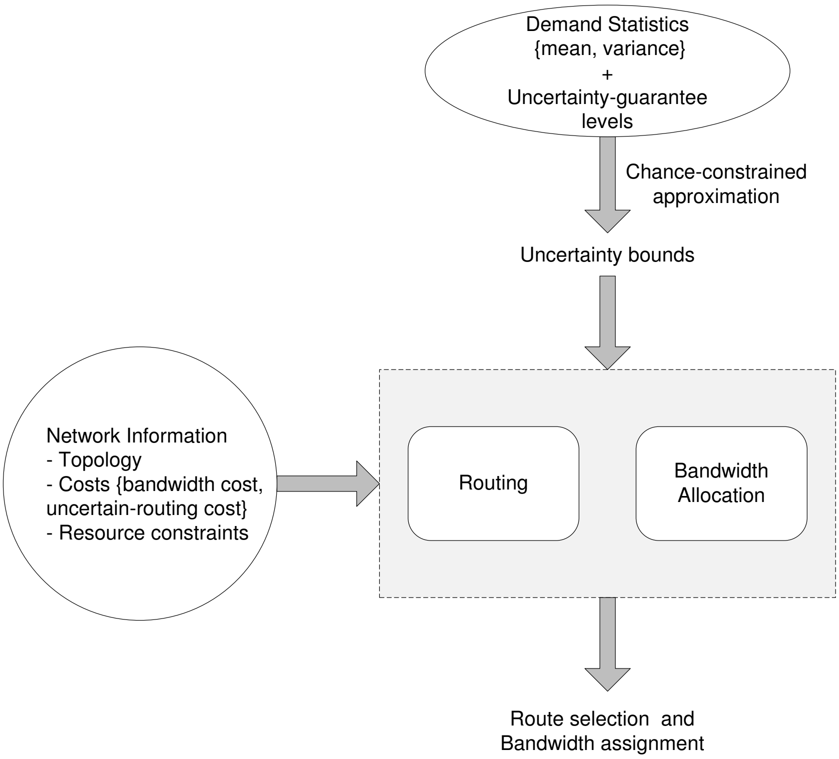

Demand Statistics
{mean, variance}
+
Uncertainty-guarantee levels
Chance-constrained approximation
Uncertainty bounds
Network Information
- Topology
- Costs {bandwidth cost, uncertain-routing cost}
- Resource constraints
Routing
Bandwidth Allocation
Route selection and Bandwidth assignment

Figure 4.2: General network design procedure

the Chance-Constrained approximation approach. Such bandwidth approximations are also used as a set of constraints in the design problem, resulting in bandwidth allocation on network links after the problem has been solved for route solutions. To summarize, first, in this chapter, optimization models are developed to represent the network design problem in an uncertainty environment. Then, in Chapter 5, efficient algorithms that can solve the proposed optimization problems in reasonable time with good quality are developed. Finally, issues related to any large traffic change that could happen in the future will be discussed in Chapter 6.

4.3 MATHEMATICAL FORMULATIONS

Next we show how mathematical programming based on the Chance-Constrained Programming (CCP), as introduced in Chapter 3, can be used to formulate optimization problems in the context of a network problem as follows. Suppose that we need to determine an amount of bandwidth to be allocated on link j, assuming this link carries traffic demands $\xi_1 \ldots \xi_K$, which are normally distributed with mean μ_k and variance σ_k^2, $k = 1 \ldots K$, as depicted in Figure 4.3.

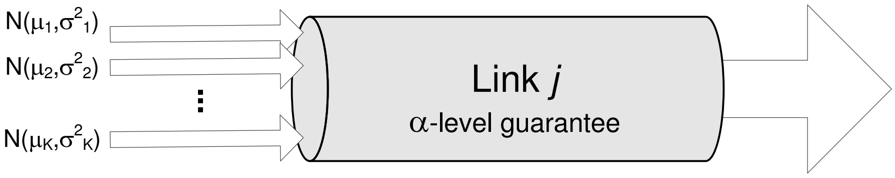


Figure 4.3: Demands carried on link j

For simplicity of the design, we assume that these random demands are independent of each other.[1] Since the sum of K independent normal random variables is also a normal random variable distributed with mean $\sum_{k=1}^{K} \mu_k$ and variance $\sum_{k=1}^{K} \sigma_k^2$, Equation (3.7), which considers only one random demand, can be extended for K demands. Specifically we define x_j as the amount of bandwidth required on a link to provide a α_j level guarantee to the aggregate traffic, which is given by

$$x_j \quad = \quad \sum_{k=1}^{K} \mu_k \; + \; \Phi^{-1}(\alpha_j) \sqrt{\sum_{k=1}^{K} \sigma_k^2} \quad . \tag{4.1}$$

This is the deterministic equivalent of the chance-constraint with α-level guarantee, which is the bound on the bandwidth required on link j to carry this set of random demands.

[1] If they are not independent, a covariance factor will be included.

4.3.1 Mathematical Formulation 1

Let a network be defined by a graph $(\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is a set of nodes, and $\mathcal{A}$ is a set of links. The notation is as follows:
Let

- $\mathcal{D}$ denote the set of point-to-point traffic demands. Each demand is specified with mean μ_k and variance σ_k^2.
- $\mathcal{P}^k$ denote a predefined set of candidate paths for demand $k \in \mathcal{D}$.
- c_j denote the cost per bandwidth unit on link $j \in \mathcal{A}$.
- F_j denote the fixed charge cost of using link $j \in \mathcal{A}$.
- α_j denote the level of guarantee of link $j \in \mathcal{A}$.
- W_j denote the bandwidth bound on the total volume of traffic carried on link $j \in \mathcal{A}$.
- $f^{k,p}$ denote a decision variable, which is equal to 1 if the flow of demand k selects path p from the predetermined path set $\mathcal{P}^k$, and 0 otherwise.
- y_j denote a decision variable. It is equal to 1 if link $j \in \mathcal{A}$ is used, and 0 otherwise.
- $\delta_j^{k,p}$ denote a binary parameter. It is equal to 1 if path $p \in \mathcal{P}^k$ for demand k uses link j, and 0 otherwise. Note that $\delta_j^{k,p}$ is determined when the set of possible paths $\mathcal{P}^k$ is calculated.

Basic Uncertainty Model

The goal of this model is to determine the optimal path assignment (i.e., the best set of $f^{k,p}$) such that the total network cost resulting from routing and allocating capacities to the random demands carried on such paths is minimized. There are two types of costs in the model. The first type is the cost per bandwidth unit (c_j) for both the *certain* and *uncertain* demand volumes carried on each link. The second type is the fixed charge (F_j) for carrying the uncertain demand on link j. This can be used as an uncertainty-avoidance penalty cost associated with resource limitations or network policy. For example, if the network manager tries to avoid routing random traffic through some links or a certain area, he can easily adjust this quantity. The fixed charge can also be used as a distance-based cost since the longer the link, the more physical-layer network devices required.

The inputs to the model are the following parameters: network topology, a set of random traffic demands with probabilistic information (i.e., mean and variance), precalculated sets of candidate paths, cost parameters, and the guarantee level on each link. The outputs of the model are the route selection with the uncertainty bounds required on links and network cost. Given the notation above, the minimum cost network design with demand uncertainty can be formulated as follows.

$$\min_{f^{k,p},\ y_j} \sum_{j \in \mathcal{A}} \sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} c_j \ \mu_k \ \delta_j^{k,p} \ f^{k,p} \ + \ \sum_{j \in \mathcal{A}} c_j \ \Phi^{-1}(\alpha_j) \sqrt{\sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} \sigma_k^2 \ \delta_j^{k,p} \ f^{k,p}} \ + \sum_{j \in \mathcal{A}} F_j \ y_j \tag{4.2}$$

subject to:

$$\sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} \mu_k \ \delta_j^{k,p} \ f^{k,p} \ + \ \Phi^{-1}(\alpha_j) \sqrt{\sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} \sigma_k^2 \ \delta_j^{k,p} \ f^{k,p}} \ \leq \ W_j \ , \ \forall j \in \mathcal{A} \tag{4.3}$$

$$\sum_{p \in \mathcal{P}^k} f^{k,p} \ = \ 1 \ , \ \forall k \in \mathcal{D} \tag{4.4}$$

$$\delta_j^{k,p} \ f^{k,p} \ \leq \ y_j \ , \ \forall j \in \mathcal{A}, \ \forall k \in \mathcal{D}, \ \forall p \in \mathcal{P}^k \tag{4.5}$$

$$f^{k,p}, \ \ y_j \ \in \ \{0,1\} \tag{4.6}$$

The objective function (4.2) and constraints (4.3) contain both the certain and the uncertain parts. These terms are straightforwardly derived from the mean (μ_k) and the variance (σ_k^2) with the multiplication of the flow decision variable $f^{k,p}$. Constraints (4.3) ensures that the amount of capacity allocated on link j does not exceed the bound W_j, which is the upper bound on link bandwidth determined by the network manager or physical link capacities. Constraints (4.4) forces demand k to be routed on a single path to preserve flow integrity.

If link j is selected, constraints (4.5) adds a fixed charge to the link cost.

This is classified as a multicommodity flow (MCF) optimization problem. Since the decision variable $f^{k,p}$ in (4.2) and (4.3) appears inside the square root, the uncertainty factor introduces a nonlinearity in both the objective function and constraints. Note that the linear MCF integral flow problem is known to be NP-complete. Hence, this nonlinear problem is even harder to solve. Especially, when dealing with a time-critical application (e.g., dynamic routing and bandwidth allocation in a IP-over-WDM network), obtaining an optimal solution from an exact nonlinear MCF problem may be impractical. Thus, there is a need for a heuristic approach that is simple, but yields good quality solutions as discussed in Chapter 5.

4.3.2 Mathematical Formulation 2

In addition to the formulation above, Chance-Constrained Programming can also be applied in another approach to incorporating uncertainty into the network design problem. Consider the following case illustrated in Figure 4.4. This is different from Figure 4.3 since each demand now has its own specified level of uncertainty guarantee (α_k). Each traffic will be guaranteed with its specified α level along the route from source to destination. In the other words, the uncertainty is handled through an end-to-end agreement between the network provider and the user, and the guarantee of the uncertainty is statistically provided on end-to-end individual traffic or end-to-end aggregated traffic flows. In contrast, the previous case provides link-by-link guarantees, which support statistical multiplexing of flows on a link thereby reducing the resources needed at a link, whereas the end-to-end approach aggregates traffic at end level.

Unlike the uncertainty bound described in Equation 4.1 of the previous model, here the total capacity satisfying all traffic demands on the link could be determined as follows:

$$x_j = \sum_{k=1}^{K} \mu_k + \Phi^{-1}(\alpha_k)\sigma_k \tag{4.7}$$

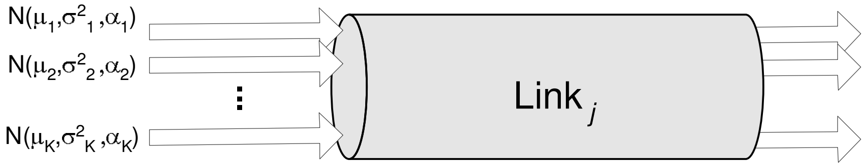


Figure 4.4: Demands carried on link j

Note that the lack of square root term make the problem much easier to solve compared with the previous problem, as will be discussed next. Similar to the previous model, the inputs to the model consists of parameters: network topology, a set of random traffic demands with probabilistic information (i.e., mean and variance), precalculated sets of candidate paths, cost parameters, and the guarantee levels per traffic flow. The outputs of the model are route selection with the uncertainty bounds, bandwidth required on links, and network cost. Given the notation above, the minimum cost network design with demand uncertainty can be formulated as follows.

$$\min_{f^{k,p},\ y_j} \sum_{j \in \mathcal{A}} \sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} c_j \ \left(\mu_k + \Phi^{-1}(\alpha_k)\sigma_k\right) \ \delta_j^{k,p} \ f^{k,p} \ + \ \sum_{j \in \mathcal{A}} F_j \ y_j \tag{4.8}$$

subject to:

$$\sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} \left(\mu_k + \Phi^{-1}(\alpha_k)\sigma_k\right) \delta_j^{k,p} \ f^{k,p} \ \leq \ W_j \ , \ \forall j \in \mathcal{A} \tag{4.9}$$

$$\sum_{p \in \mathcal{P}^k} f^{k,p} \ = \ 1 \ , \ \forall k \in \mathcal{D} \tag{4.10}$$

$$\delta_j^{k,p} \ f^{k,p} \ \leq \ y_j \ , \ \forall j \in \mathcal{A}, \ \forall k \in \mathcal{D}, \ \forall p \in \mathcal{P}^k \tag{4.11}$$

$$f^{k,p}, \ \ y_j \ \in \ \{0,1\} \tag{4.12}$$

The formulation of this model is similar to the previous one, except that now the aggregation of the uncertainty part (which involves the variance term) is much simpler. In this case, when combined with the flow variable in the both the objective function and constraints, the resulting problem is linear. The formulation above is a mixed integer programming problem and can be solved using the well known branch-and-bound technique for small problem cases. While Mathematical Formulation 2 is linear, it is still a NP-hard problem, and for large problems, a heuristic approximation solution technique must be adapted.

4.3.3 Bandwidth reservation comparison

A network manager may select a bandwidth reservation approach based on the guarantee-level needed. When considering an equivalent end-to-end guarantee of the demands using shared allocated bandwidth on a route (which consists of a series of links with specified link-level guarantees, i.e., Formulation 2), this value can be approximated by multiplying all the link-level guarantees of links used by that route. For example, consider a four-hop route with a 0.99 level of guarantee for all links as shown in Figure 4.5. The approximated end-to-end calculation will be equal to $0.99^4 = 0.961$. If the link-level guarantee is increased to 0.999, the equivalent end-to-end guarantee value will be 0.999^4 or 0.996. Thus, the equivalent end-to-end guarantee calculation of a route may be used as an additional metric if the network manager decides to achieve some level of end-to-end guarantee. Furthermore, the equivalent end-to-end guarantee value can be used as an additional constraint for some specific routes in the link-level guarantee optimization problem.

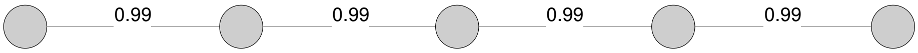


Figure 4.5: A four-hop route.

Numerical results considering flow aggregation on a single-hop link are shown in Table 4.1. This includes four types of bandwidth reservation approaches, which are mean-rate, peak-rate, link-level guarantee (common pooling), and end-to-end guarantee. For a simple illustration, the following calculations consider the required bandwidth of traffic demands carried on one link only. The demand characteristics such as mean rate, peak rate, and variance of the demands, are from actual traffic measurement in [29]. The measured mean rate and peak rate are 225, and 342 Mbps, respectively, and the traffic standard deviation is 25 Mbps.

Table 4.1: Bandwidth reservation calculations for a single-hop link

Case	Reservation Type	Guarantee Level	Bandwidth for 10 demands (Mbps)	Bandwidth for 200 demands (Mbps)
1	Mean rate	mean level	2250.00	45000.00
2	Link-level	0.95	2380.05	45581.60
3	Link-level	0.99	2433.89	45822.37
4	Link-level	0.999	2494.29	46092.48
5	End-to-end	0.95	2661.25	53225.00
6	End-to-end	0.99	2831.50	56630.00
7	End-to-end	0.999	3022.50	60450.00
8	Peak rate	peak level	3420.00	68400.00

As seen in Table 4.1, for a small number of demands (e.g., 10 demands), the differences of reserved link bandwidths among cases are not significant. However, as the number of demands increase from 10 to 200 sources, the differences are clearly observed. For example, when compared with the results of the end-to-end and the peak-rate approaches, the bandwidth saving gains in the link-level approach (case 2 - case 4) increase. Thus, such gain can be used as a factor to determine whether the common-pooling reservation (or link-level guarantee) is preferred.

4.4 SUMMARY

This chapter discusses important issues and assumptions in the uncertainty framework. Basic mathematical programming models are described. The resulting formulation of the proposed model is a nonlinear multicommodity flow problem, which requires heuristic solution algorithms. The bandwidth savings gain is a crucial factor for such statistical aggregation approach.

5.0 SOLUTION ALGORITHMS

To solve the nonlinear problem as discussed in the previous chapter, an approximate linear model is developed so that an optimization process could be applied efficiently. The following sections describe the model, solution algorithms, and their performance.

5.1 HEURISTIC METHODS

The heuristic solution approach proposed here consists of two phases. The first phase determines a good feasible solution or the *base-solution* obtained by an approximation algorithm of the proposed nonlinear problem. Then, the algorithms in the second phase use the base-solution as a starting point to further improve the quality.

5.1.1 Phase I: Linear Approximation Method

The main idea is to approximate the nonlinear objective function and constraints using a linear method so that the modified problem is then easily solved with typical linear programming techniques. We approximate the nonlinear part of the objective function and constraints with linear terms by the following steps. Consider the uncertainty part in Equation (4.1), the square root of the sum of variances is the source of difficulty since the similar terms are used in the objective function (4.2) and the constraints (4.3). By inspecting the sum of variances inside the square root, we approximate the real value by computing each variance separately with a scaling factor β, and then sum up all terms as follows.

$$\sqrt{\beta\ \sigma_1^2} + \sqrt{\beta\ \sigma_2^2} + \cdots + \sqrt{\beta\ \sigma_K^2} \ \approx\ \sqrt{\sigma_1^2 + \sigma_2^2 + \cdots + \sigma_K^2} \tag{5.1}$$

where $0 < \beta \leq 1$ $(\beta = 1$ when $K = 1)$.

The parameter β is used to scale down each variance on the left hand side of (5.1) so that the summation is close to the square root of the sum of variances on the right hand side. For example, let $\sigma_1^2 = \sigma_2^2 = 8$, so $\sqrt{\sigma_1^2 + \sigma_2^2} = 4$. If we choose $\beta = 0.5$, this is a perfect match. However, if choose $\beta = 0.45$, thus $\sqrt{\beta\ \sigma_1^2} + \sqrt{\beta\ \sigma_2^2} = 3.79$, introducing an approximation error. Thus, a parameter-tuning process is needed. If we can achieve an accurate approximation, the optimization problem of Section 4.3.1 will change to a linear version, which can be viewed as routing each traffic individually, without the need of computing the common variance term of all the demands carried on a link. Thus, the approximate amount of bandwidth required for flow k routed on link j equals:

$$\mu_k \quad + \quad \Phi^{-1}(\alpha_j)\sqrt{\beta\ \sigma_k^2} \tag{5.2}$$

Claim 4.1. Varying $\beta \in (0, 1]$ in (5.2) covers the exact value of the uncertainty factor.
Proof: Varying the parameter β in a range $(0, 1]$ yeilds the upper and lower bounds:

$$0 \quad < \quad \Phi^{-1}(\alpha_j)\sum_{k=1}^{N}\sqrt{\beta\ \sigma_k^2} \quad \leq \quad \Phi^{-1}(\alpha_j)\sum_{k=1}^{N}\sqrt{\sigma_k^2} \tag{5.3}$$

Since the amount of uncertainty of N demands carried on link j has an exact value of $\Phi^{-1}(\alpha_j)\sqrt{\sum_{k=1}^{N}\ \sigma_k^2}$, this falls in the same range. Thus, the range of the linear-approximation includes the exact value within it.

With this approximation, the nonlinear objective function (4.2) and the constraints (4.3) are transformed to (5.4) and (5.5), respectively, as shown in the following optimization problem.

<u>Problem Heuristic LP:</u>

$$\min_{f^{k,p},\ y_j} \sum_{j\in\mathcal{A}} \sum_{k\in\mathcal{D}} \sum_{p\in\mathcal{P}^k} c_j \left(\mu_k + \Phi^{-1}(\alpha_j)\sqrt{\beta\ \sigma_k^2}\right) \delta_j^{k,p}\ f^{k,p} \ + \ \sum_{j\in\mathcal{A}} F_j\ y_j \tag{5.4}$$

subject to:

$$\sum_{k\in\mathcal{D}}\sum_{p\in\mathcal{P}^k}\left(\mu_k+\Phi^{-1}(\alpha_j)\sqrt{\beta\ \sigma_k^2}\right)\delta_j^{k,p}\ f^{k,p}\ \leq\ W_j\ ,\ \forall j\in\mathcal{A} \tag{5.5}$$

$$\sum_{p\in\mathcal{P}^k} f^{k,p}\ =\ 1\ ,\ \forall k\in\mathcal{D} \tag{5.6}$$

$$\delta_j^{k,p}\ f^{k,p}\ \leq\ y_j\ ,\ \forall j\in\mathcal{A},\ \forall k\in\mathcal{D},\ \forall p\in\mathcal{P}^k \tag{5.7}$$

$$f^{k,p},\ \ y_j\ \in\ \{0,1\} \tag{5.8}$$

This transformed problem is a linear mixed integer programming problem, thus it can be solve with a linear mixed integer programming optimization solution technique such as the Branch and Bound method. Thus, the linear approximation method proposed here consists of two stages. First, for each β, we use a linear approximation to find feasible paths (solved by a linear optimization solver). Second, we calculate the real value of the uncertainty and then check the bandwidth constraints as shown in constraints (4.3). If all the constraints are satisfied, a feasible solution is found. Otherwise, the problem is infeasible, and parameter β is incremented in the next iteration. The algorithm is described as follows.

Phase I: *LinearApprox*(β) Algorithm

```
Begin
    $\beta := 0$
    $iteration := 0$
    $step := 0.1$
    do{
        Find feasible paths by solving Problem *Heuristic LP*
        $Feasible$ := TRUE
        for (each link $j$ in the feasible path set){
            Calculate $x_j := \sum_{k=1}^{K}\mu_k \ + \ \Phi^{-1}(\alpha_j)\sqrt{\sum_{k=1}^{K}\sigma_k^2}$
            if ($x_j > W_j$) {
                $Feasible$ := FALSE
                Exit this for-loop
            }
        }
        if ($Feasible$ == TRUE) {
            Calculate $TotalCost[iteration]$
        }
        $\beta := \beta + step$
        $iteration := iteration + 1$
    } while ($\beta \leq 1$)
    $LinearApprox(\beta)$ solution = minimum cost solution of all iterations
End
```

Note that the parameter *step* could be changed, ranging from fine values (e.g., $step = 0.05$) to coarse values (e.g., $step = 0.1$). When K=1, there are only one demand carried on a link, and then a real β must be unity in order to satisfy Equation (5.1). However, generally, β is less than unity. For example, if there are three traffic demands with the same variance, which are carried on the same link, the real value of β, obtained by solving Equation (4.1), is 0.333. If there are many more demands on this link, the value of β will be much smaller than unity. Now let's consider what would happen if the iterative algorithm starts at the largest value of β, which is unity, and then steps down to zero. The larger the β, the more bandwidth will be reserved for the uncertainty for all the demands carried on this link (see constraints (4.3)). Therefore, if the bandwidth of the considered link is scarce, starting the algorithm at large β's would force some demand to be routed on another link in order to make the link bandwidth constraint feasible. As a result, although the obtained solution is feasible, the cost would be higher when compared with the feasible solutions obtained from small β's. Thus, starting the algorithm from small β's, and then increasing to large values would be the better way to find feasible solutions.

Generally, the quality of solutions obtained with the fine step values is not different than those obtained with the coarse step values unless the network resources are scarce, e.g., having a very small link bandwidth bound (W_j). Therefore, we could save computational time in this phase by using coarse step values. Numerical results in this chapter are obtained with the step value of 0.1.

5.1.2 Phase II Heuristic Algorithms

An initial solution obtained from the *LinearApprox*(β) algorithm may be further improved by additional steps in a second phase, which should be fast and simple to implement following the first phase. We first consider the pattern of a solution, called the *solution string*, is of the form:

$$p_1 \;\; p_2 \;\; p_3 \;\; p_4 \;\; p_5 \;\; p_6 \;\; p_7 \;\; p_8 \;\; p_9 \;\; \cdots$$

where p_k is the selected path from a given path set of demand k.
The first step in Phase II uses a simple operation, by changing a path of each demand one at a time to see improvement, while keeping the selected paths of other demands obtained from Phase I unchanged. This is a routing based heuristic similar to the one proposed in [30].

Phase II: Step1 Algorithm

```
Begin
     $BestCost :=$ cost of the $LinearApprox(\beta)$ solution
     $BestPath :=$ path solution of the $LinearApprox(\beta)$
     for (each demand $k$ in $\mathcal{D}$){
          $r :=$ path selected in $LinearApprox(\beta)$ solution
          for (each $q$ in $\mathcal{P}^k - \{r\}$){
               Calculate $TotalCost$ with path $q$ of demand $k$ and paths selected
                                     of the other demands in $\mathcal{D} - \{k\}$
               if ($TotalCost < BestCost$) {
                    $BestCost := TotalCost$
                    new path of demand $k := q$, update $BestPath$
               }
          }
     }
```

```
End
```

The computational time for this step is negligible due to its simplicity. Then, the best solution obtained so far will be an input into the next algorithm, which is named the *Step2 Algorithm*. To understand the concept, first we need to consider the solution space of Problem (5.4). From the mathematical formulation, there are three main parameters that affect the solution space: network arc or link $j \in \mathcal{A}$, traffic demands $k \in \mathcal{D}$, and path $p \in \mathcal{P}^k$. The set of network links ($\mathcal{A}$) generally varies with the network size, which cannot be controlled. Another factor that cannot be controlled is the number of traffic demands (in set $\mathcal{D}$). So the combination of large network size and large number of demands could lead to a large solution space, requiring long computational time as expected. However, we can control one parameter, which is the size of the path set for each demand. Many studies have appeared in the literature showing that instead of using the whole precalculated path set, considering paths from a smaller subset is more efficient with little sacrifice in optimality [28]. This is because the paths from the top of the list (i.e., shorter paths) usually have a higher chance to be selected than those from the bottom of the list. Therefore, we set a limit on the path set in the algorithm. As a result, the solution space could be further reduced. The algorithm is described as follows:

Phase II: Step2 Algorithm

```
Begin
    $BestCost := BestCost$ from Step 1
    $BestPath :=$ path solution from Step 1
    $TempPath := BestPath$
    for (each solution $i : i \leq MaxNumber$){
        $Feasible$ := FALSE
        do {
            for (each demand $k$ in $\mathcal{D}$){
                if ($Uniform(0,1) \leq P_{mutation}$)
                    $TempPath[k] :=$ Integer random number from $RandomSet[k]$
            }
            if ($TempPath$ satisfies problem constraints) {
                $Feasible$ := TRUE
                Calculate $TotalCost$
                if ($TotalCost < BestCost$) {
                    $BestCost := TotalCost$
                    $BestPath := TempPath$
                }
            }
        } while ($Feasible$ := FALSE)
    }
End
```

The Phase II Algorithm allows each path in the solution to be changed probabilistically similar to the mutation process in a Genetic Algorithm. For each demand, if a random number generated is within some specified probability ($P_{mutation}$), a new path will be selected randomly from a given path set. Otherwise, that path is not changed from the initial solution. The reason why the minimum cost path is not always selected is that selecting only the minimum cost path for every demand does not always make the whole problem feasible, especially in a highly constrained problem.

Each iteration is complete when a feasible solution is found. Therefore, it could require longer time to obtain a feasible solution for some difficult problem (e.g., scarce network resources with high level of demand) when compared with easier problems. The algorithm continues searching for the best solution until the maximum number of trials ($MaxNumber$) is reached. Parameter $P_{mutation}$ is set based on trial-and-error. If $P_{mutation}$ is set too high, the solution string would be completely random and cannot take benefit of having an initial solution from the previous steps. In contrast, if it is set too low, a solution would rarely improved. In our studies, we set $P_{mutation} = 0.4$. Parameter $MaxNumber$ could be set flexibly based on the tradeoff between computation time and quality of solution. This value is set to

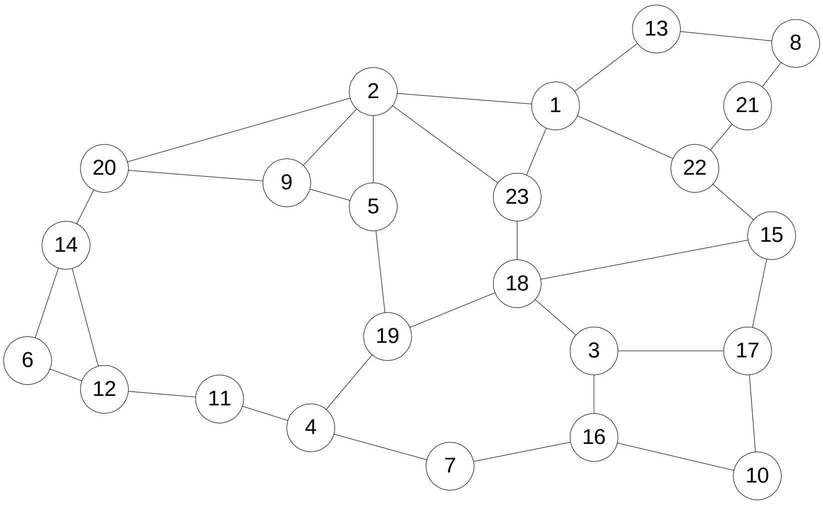


Figure 5.1: Net23 (23 nodes, 33 links)

20,000 in our experiments. As mentioned earlier, parameter *RandomSet* is designed to reduce the size of solution space by considering path(s) from a smaller subset, instead of using the whole precalculated path set. An example of choosing this parameter is explained below.

Example 1: Consider a 23-node network in Figure 5.1. The following candidate path set is used for routing traffic from node 3 to node 8. This path set consists of the first eight shortest paths.

```
Path 0 (5 hops):  3  17  15  22  21   8
Path 1 (5 hops):  3  18  15  22  21   8
Path 2 (5 hops):  3  18  23   1  13   8
Path 3 (6 hops):  3  17  15  22   1  13   8
Path 4 (6 hops):  3  18  15  22   1  13   8
Path 5 (6 hops):  3  18  23   1  22  21   8
Path 6 (6 hops):  3  18  23   2   1  13   8
Path 7 (7 hops):  3  16  10  17  15  22  21   8
```

In this example, parameter *RandomSet* could be set based on the number of hops of the selected path in the initial solution. Two rules can be applied:

1. **Strict Rule**: use all paths with the same or less number of hops. For example, if the initial path solution for demand 3-8 is `Path 0`, `Path 0 - 2` will be included in *RandomSet*. If `Path 3` is an initial path, *RandomSet* will consist of `Path 0 - 6`.

2. **One-more-level Rule**: also consider the path(s) with more hops. For example, if `Path 0` is the initial path, the next hop-level paths (i.e., `Path 3 - 6`) will also be considered; thus, `Path 0 - 6` will be included in *RandomSet*. If `Path 3` is an initial solution, *RandomSet* is the same as the given candidate path set, and `Path 0 - 7` will be included.

There are reasons to considered these rules. This is because the objective is to minimize the total cost, and in general selecting the paths on the top of the path set (shorter paths) is more likely to cost less than selecting the paths on the bottom of the set (longer paths). Therefore, the *Strict Rule* reduces the solution space by not including the paths with more number of hops, which have lower probability to improve the solution. However, depending on network constraints, some demand may need to be routed through a longer path, thus the *One-more-level Rule* will include this possibility to increase a chance of obtaining an optimal solution. If the link bandwidth is large, then each traffic demand will be routed with its minimum cost path since the resource is always sufficient. In this case, the *Strict Rule* would be very practical. However, in general, especially in a highly-constrained problem, some traffic demand could not be routed on its minimum cost path, the *One-more-level Rule* would be practical to reduce the size of path set in this case.

Example 2: Consider Net23 in Figure 5.1 with a demand matrix and network information, Net23-04, in Appendix A. The heuristic algorithms obtain the following results.

In Phase 1, the *LinearApprox*(β) algorithm starts with $\beta = 0$, and causes constraint violations on link 3-18 and 18-23. Then β is increased to 0.1 in the next iteration, and the problem is feasible. Next, increasing β does not improve the solution. Thus, the output or the initial solution string is

`0 0 0 1 0 2 1 0 0 0 0 1 0 0 0 0`, cost = 4535.28

In Phase 2, the Step 1 simply reroutes one demand at a time. The best solution is
`0 0 0 1 0 2 1 0 0 0 0 1 1 0 0 0` , cost = 4532.20
The Step 2 algorithm further improves the solution, resulting in the final output:
`0 0 0 1 0 1 1 0 0 1 1 1 1 0 0 0` , cost = 4516.64

Table 5.1 shows more examples on solution improvement. All the experiments use the same input demand matrix, but are performed on Net23 with different network characteristics. Note that sometimes the initial solution of the first phase could be improved as in Experiment 23-0, 23-02, and 23-03, but not in Experiment 23-01.

Table 5.1: Examples on solution improvement

Experiment	Phase 1	Phase 2, Step 1	Phase 2, Step 2
23-0	4395.61	no improvement	4395.14
23-01	4207.45	no improvement	no improvement
23-02	4394.14	4393.70	4380.91
23-03	4535.28	4532.20	4516.64

5.1.3 Heuristic solution algorithm with dynamic β

Next, we investigate another similar solution algorithm, which is obtained by modifying the linear approximation method. In the previous approach, the same value of β is assigned to all network links. If any bandwidth constraint violation is found, β will be incremented for all links, and then the problem will be resolved. In this modified algorithm, parameter β, which is denoted by β_j for link j, can take different values depending on violations of link bandwidth constraints. β_j is incremented whenever the calculation for an actual bandwidth constraint for link j is violated. The algorithm will terminate whenever the feasible solution is found or some β_j reaches the unity. In the algorithm shown below, there is an optional step called *Decreaseβ* to be used in case that decreasing a β_j of some link is needed. The criteria for choosing which β_j to be decreased can be based on randomly selection or the link utilization level. For the link utilization criterion, β_j is decremented when link j that β_j

has been incremented in the previous iteration has low link utilization level. In other words, it is possible that large β_j's could be no longer necessary when smaller β_j can also make the problem solution feasible, and reducing β_j values may improve the solution quality. The algorithm is described as follows.

Linear Approximation with dynamic β

```
Begin
    $iteration := 0$
    $\beta_j := 0$
    $step := 0.1$
    do{
        Find feasible paths by solving Problem *Heuristic LP* with $\beta_j$
        $Feasible$ := TRUE
        for (each link $j$ in the feasible path set){
            Calculate $x_j := \sum_{k=1}^{K} \mu_k \; + \; \Phi^{-1}(\alpha_j)\sqrt{\sum_{k=1}^{K} \sigma_k^2}$
            if ($x_j > W_j$) {
                $Feasible$ := FALSE
                $\beta_j := \beta_j + step$
                Exit this for-loop
            }
        }
        if ($Feasible$ == TRUE) {
            Calculate $TotalCost[iteration]$
            $Decrease\beta$ (optional step)
        }
        $iteration := iteration + 1$
    } while ($Feasible$ == FALSE *or* All $\beta_j$'s $\leq 1$)
     Solution = minimum cost solution of all iterations
End
```

Example 3: Consider a Net23 network with Net23-13 demand matrix and network information described in Appendix A. The algorithm searches for solutions as follows:

Iteration 0: Start with $\beta = 0$ with the step of 0.1, and then solve the Heuristic LP problem. The bandwidth constraints on five links, which are (1,23), (2,20), (3,18), (14,20), and (18,23), are infeasible. Then parameter β_j of these links are incremented to 0.1 for the next iteration, whereas the rest of β_j's are unchanged.

Iteration 1: The results contains two bandwidth constraint violations on link (1,23) and link (4,19). Link (1,23), whose constraint is infeasible in Iteration 0, need to increase β_{1-23} to 0.2. β_{4-19} is increased to 0.1.

Iteration 2: The number of constraint violations is now decreased to one. Link (15,18) is infeasible, so β_{15-18} is increased to 0.1.

Iteration 3: Now the infeasible link is link (3,18), which is the link whose constraint is violated before. Thus, β_{3-18} is increased to 0.2.

Iteration 4: Link (3,18) is still infeasible, and β_{3-18} is increased to 0.3.

Iteration 5: Link (3,18) is still infeasible, and β_{3-18} is increased to 0.4.

Iteration 6: The problem is now feasible. The algorithm terminates, resulting in the cost of 5136.36.

Compared this result with the one obtained by the previous heuristic. The previous algorithm requires two iterations of solving the problem (with a final β equal to 0.2), and then obtains the same solution. In general, the second heuristic may take more number of iterations to obtain a solution since it gradually increases only β_j of the links whose constraints are infeasible. Therefore, it is possible that trying to satisfy some constraints at a time results in violating another constraints in the next iteration due to fluctuation in traffic routing. Furthermore, the results with the β-decreasing option produce no noticeable improvement in our studies due to route change (routing fluctuation) similar to the above situation. For instance, decreasing β_j in one place can make the traffic demands change routes, thus forcing β_j to be increased in another places. Hence, this approach will be consider as optional to be investigated in future work.

The next step is to determine performance of the heuristic. Due to the pattern of a solution, it is appropriate to compare our heuristic with a Genetic Algorithm (GA). The GA is suited to this specific problem since the algorithm could be simply implemented to search for a solution with precalculated path sets. The solution string can be viewed as a *chromosome* in the GA literature. The GA used in the comparisons is described in the next section.

5.2 GENETIC ALGORITHM

A Genetic Algorithms (GA) is a stochastic search technique inspired by evolutionary principles or the Darwinian concept of "Survival of the Fittest". In nature, the fittest individuals are more likely to survive well, and then their good genetic features will be passed from parents to offspring, to the next generation and so on. Such parents are more likely to be selected from a pool (called *population*) for breeding. The selection of highly fit individuals deals with an evaluation of the fitness of each individual in the population with the assumption that such fitness values are associated with good performance. It is possible that some good genetic features from each of the parents could be combined to create children with better performance through a *crossover* or *recombination* process. In addition, some individuals' genes could be slightly changed through a *mutation* process. As a result, after a number of generations, better solutions can be obtained, and the best individual ever obtained during the run is the solution to the problem. The advantages of a GA are that it tends to achieve good solutions in many types of problems. Furthermore, it can be robust to getting stuck at local optimal solutions. The disadvantage is that a GA may be computational expensive (i.e., take a long time to obtain satisfactory solutions). In addition, due to the nature of its stochastic search, the optimum solution is not guaranteed.

To implement the GA, first, an individual or a *chromosome* is used to represent a solution to the problem in the form of a string of symbols or numbers. This fits our pattern of solution (i.e., a string of paths, where each element in the string is a path selected by each traffic demand). For example, the chromosome is coded in the form of a vector composted of N

elements: $\mathbf{x} = p_1\ p_2\ p_3\ \ldots\ p_N$. In our optimization problem, the goal is to select a path for each of the N demands to minimize the cost of the whole network subject to the given constraints. The basic GA processes are described as follows.

- Initialization process

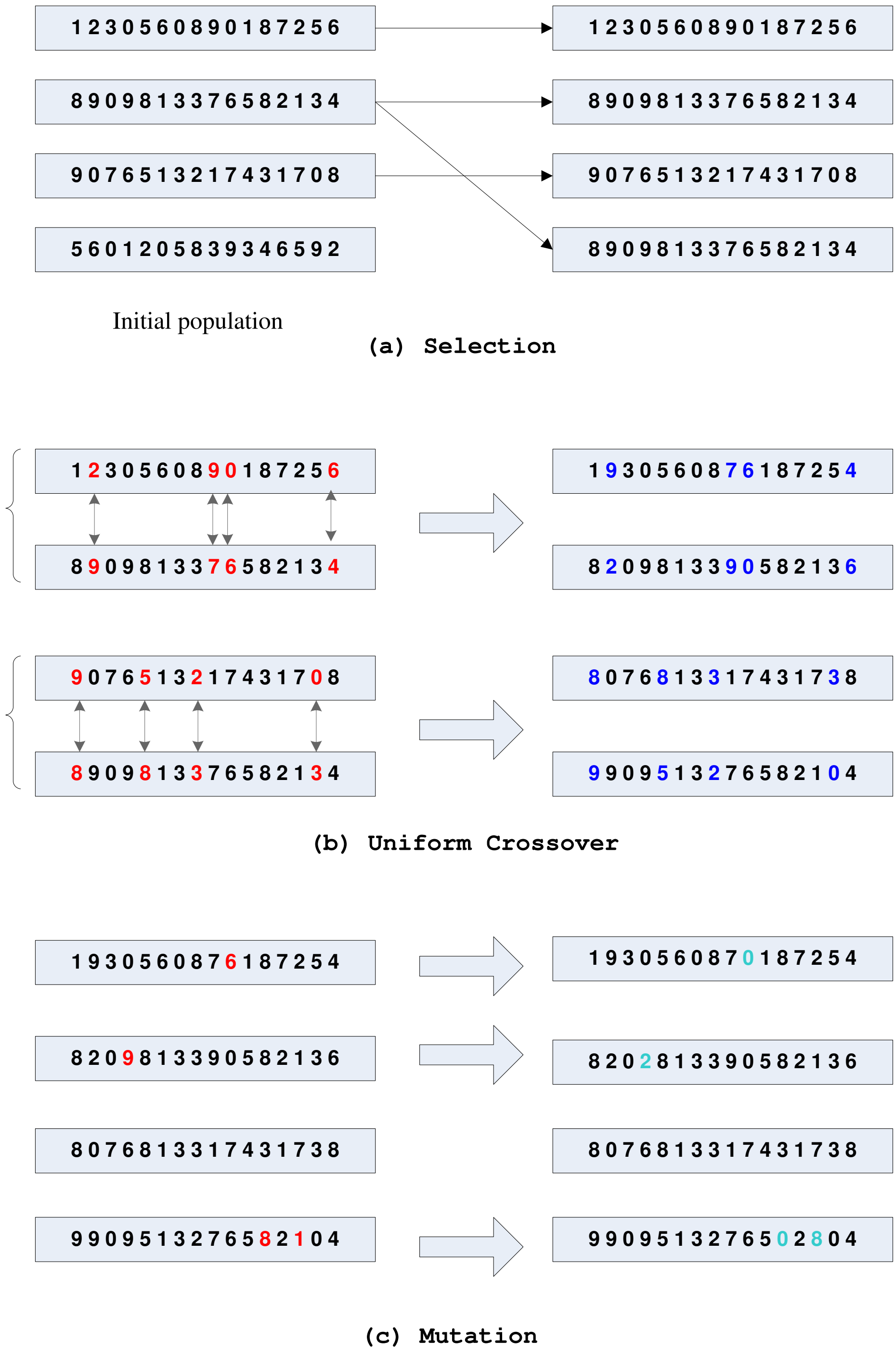
Since the GA operates on a set of multiple solutions, it starts with an initial set of chromosomes. In general, this first population is randomly generated. The total number of chromosomes generated in a pool is called the *population size*, and this size is a problem-dependent parameter that can be tuned. Once the initial population is created, the GA searches for better solutions as the number of generations increases with the *selection*, *crossover*, and *mutation* processes.

- Selection process

From one generation to another generation, the selection process is used to select the parents in order to produce new offspring. First, the quality of each chromosome is determined by the fitness function. There are many criteria in defining the fitness function, but usually it is based on the objective function of an optimization problem. In each generation, the chromosomes with the best fitness values will have a higher chance to be selected as the parents in the crossover operation, or to survive in the next generation. The purpose for selecting the best individuals is to continually improve the fitness of the best solutions. Depending on a specified policy, the fittest chromosomes in a population may be selected to the next round directly; this is called an *elitest strategy.* In addition, even the unfit individuals may also be probabilistically selected in order to maintain diversity. Figure 5.2 (a) illustrates an example of the selection process. Based on some specified fitness function and a policy, the first three chromosomes are probabilistically selected. The second one is selected two times, while the last one is not chosen.

- Crossover operation

In the crossover process, a pair of chromosomes is selected from the population, and only parts of these chromosomes will be exchanged. Such randomness occurs according to a



(a) Selection

(b) Uniform Crossover

(c) Mutation

Figure 5.2: GA Operations

specified *crossover rate*. There are many types of crossover operators, such as, *one-point*, *two-point*, or *uniform crossover*. For example, in one-point crossover, two crossover points on a pair of chromosomes are randomly chosen, and then the portions of their chromosomes are exchanged. Similarly, two-point crossover requires two crossover points, and the chromosome portions between these two points are exchanged. In uniform crossover, which is used in our experiments, the crossover elements on both the chromosomes are uniformly chosen, and then those elements are exchanged. Each parent is equally likely to be chosen. An example of the uniform crossover operation is shown in Figure 5.2 (b).

- Mutation operation

In an individual chromosome, mutation can occur on random positions (see Figure 5.2 (b)). The frequency of mutation occurrence depends on a defined parameter called the *mutation rate*, which is typically much smaller than the crossover rate. For example, the mutation rate in our experiments is 0.06, while the crossover rate is 0.6. However, such a small perturbation in a chromosome could increase the chance of finding good solutions in addition to the crossover routine.

The GA processes mentioned above will be performed in every generation. At the end of each generation, some fit parents and the new offspring will be probabilistically selected for the next generation, thus replacing the old chromosomes. This process continues, and the solutions in each generation will "evolve" to the better solutions until a termination criterion is satisfied, which is usually based on the number of generations, computational time, quality of solution, etc. Special GA functions such as an evaluation function and a penalty function used in the experiments are described as follows.

Evaluation function

The evaluation function used in the experiments is the *rank-based evaluation function* defined in [24]. In this approach, chromosomes are selected based on their fitness values. First, at the beginning of a generation, all chromosomes in the population are ranked in order from the highest to the lowest fitness value: $V_1, V_2, V_3, \ldots, V_{PopSize}$. Then, each chromosome will be given a cumulative probability q_i determined by using the following function:

$$\begin{aligned} q_0 &= 0 \\ q_i &= \sum_{j=1}^{i} Eval(V_j),\ i = 1, 2, \ldots, PopSize \end{aligned} \tag{5.9}$$

where $Eval(.)$ is a *rank-based evaluation function* defined as follows,

$$Eval(V_i) \quad = \quad a(1-a)^{i-1},\ 1, 2, \ldots, PopSize \tag{5.10}$$

The parameter $a \in (0, 1)$ is given, and needs to be tuned according to the population size. For example, if $a = 0.05$ and $PopSize = 60$, the cumulative probability q_i's assigned to the chromosome V_i's are:

$q_0 = 0,\ q_1 = 0.0500,\ q_2 = 0.0975,\ q_3 = 0.1426,\ q_4 = 0.1855,\ q_5 = 0.2262,\ \ldots,\ q_{60} = 0.9539$
Then the $PopSize$ chromosomes will be selected using *roulette wheel selection.* In each round, a random number $r \in (0, q_{PopSize}]$ will be uniformly generated. If $q_{i-1} < r \leq q_i$, chromosome V_i will be selected. The resulting selected chromosomes will then be used in the crossover and the mutation operation, respectively.

Penalty function

It is possible that the chromosomes obtained from the random generation (as in the initial population), or from the genetic operations would be infeasible. However, those infeasible solutions could be useful in a search since they might be closer to the global optimum. With the use of a penalty function, the search can be forced to stay inside or close to the feasible region. In the experiments, we use the adaptive penalty method proposed by Smith and Coit [31]. The penalty objective function $F_p(x)$ for the problem with m constraints is of the form:

$$f_p(x) \quad = \quad f(x) + (F_{feas}(t) - F_{all}(t)) \sum_{i=1}^{m} \left(\frac{d_i}{NFT_i} \right)^{\kappa} \tag{5.11}$$

where the notation is as follows:

- $f(x)$ denotes the unpenalized objective function.
- F_{feas}denotes the unpenalized value of the best feasible solution yet found.

- F_{all} denotes the unpenalized value of the best solution yet found.
- d_i denotes the distance to a feasible solution. This value is the number of constraints violated in our study.
- Parameter κ is a user defined exponent. Normally, the value of 1 or 2 is used.
- NFT_i denotes the "Near-Feasibility Threshold."

As introduced in [31], this is the threshold that the penalty function forces the GA to search for solutions near or within a feasible region. The NFT can be used either in a fixed or dynamic way, with the following general form:

$$NFT = \frac{NFT_o}{1+\Lambda} \tag{5.12}$$

where NFT_o is an initial NFT (e.g., $NFT_o = 1$). If $\Lambda = 0$, the NFT is static. Otherwise, Λ can be used as a dynamic feedback. For example, Λ could be defined as a function of the generation number (e.g., the generation number multiplied by a positive constant). Therefore, as the number of generations increases, the NFT will decrease, resulting in an increase in the penalty.

In our study, violation of bandwidth constraints (i.e., Inequality (4.3) in Chapter 4) is common with the chromosomes that are randomly generated, or are created by the genetic operations. Thus, to follow the penalty function (5.11), the distance d_i will denote the number of bandwidth constraint violations. Thus, the form of the penalty objective function $F_p(x)$ is:

$$f_p(x) = f(x) + (F_{feas}(t) - F_{all}(t)) \left(\frac{d_i}{NFT_i}\right)^{\kappa} \tag{5.13}$$

where $\kappa = 2$ is used. The NFT function uses $\Lambda = \lambda t$, where t is the generation number, and $\lambda \in [0, 1)$ (e.g., $\lambda = 0.1$ is often used).

In the next section, the GA with the special approaches described previously will be used in comparison with the heuristic algorithms.

5.3 NUMERICAL RESULTS

In this section, performance of the heuristic algorithms and how the solutions are affected by parameters are investigated. Based on the mathematical formulation, the factors that could affect the cost and solution time are: 1) the levels of guarantee (α), 2) the link bandwidth bound (W_j), 3) the traffic demand, and 4) the network topology and size. First, due to the lack of real cases for uncertain demand patterns that would be carried in future networks, the parameters of each random demand (i.e, the pair of mean and variance (μ, σ^2)) are generated with characteristics based on the work in [21]. In our experiments, first the variance is uniformly created within the range $[50, 100]$. Then, the maximum value of the mean (μ_{max}) can be set (which is specific to a network). The mean value of each demand is uniformly selected from $[\mu_{min}, \mu_{max}]$. These parameter values are also set based on the network limitation (e.g., link bandwidth). Examples of sets of traffic demands and networks are listed in Appendix A.

Basic experiments for studying the effects of the first three factors using Net23 (Figure 5.1) are shown in Table 5.2. Each experiment set consists of four experiments for different α values based on the notation in Table 5.3. For example, a generic set name is *Net*-`k` set (e.g., the 23-8 set), where *Net* is the name of a network and `k` is the experiment number. This scheme will also be used to study different networks for the effects of topology and size.

For the random case, the α values in the *Net*-k set in Table 5.3 are uniformly generated using the Uniform(0,1) distribution from three levels of link-level guarantees, which are 95%, 99%, and 99.9%. For example, one forth of the network links are assigned with 95%-guaranteed, and the rest are equally assigned with 99%-guaranteed and 99.9%-guaranteed. For the fixed-α cases (i.e., *Net*-k1 - *Net*-k3), all the links in the network are assigned with the same α as shown in Table 5.3.

To see the effects of the same set of demands on a different topology, Net23v2 (Figure 5.3) is used in the study. This topology is created with the same number of nodes and links as Net23. All the same sets of traffic demands for Net23 are reused in the experiments with

Table 5.2: Basic experiment sets on a 23-node network (Net23)

Experiment Set	Bandwidth Bound (capacity unit)	Number of Demands	Network/Demand Variations
23-1	200	16	-
23-3	200	16	similar to 23-1 with smaller mean of demands
23-4	400	16	similar to 23-1 with more link bandwidth
23-5	350	16	-
23-6	400	16	similar to 23-5 with more link bandwidth
23-2	200	16	-
23-7	400	16	similar to 23-2 with more link bandwidth
23-8	400	30	similar to 23-7 with more number of demands
23-9	400	30	similar to 23-8 with a different set of demands
23-100	400	45	similar to 23-7 with more number of demands
23-300	500	45	similar to 23-100 with more link bandwidth
23-500	1000	100	-
23-700	1000	200	similar to 23-500 with more number of demands

Table 5.3: Example of the α values in the Net-k set

Experiment	α
Net-k	random
Net-k1	0.95
Net-k2	0.99
Net-k3	0.999

Net23v2.

To solve the problem with the heuristic approach, the first phase is implemented with AMPL and solved with the CPLEX 9.1 solver with a Brand-and-Bound option. The second phase program is written in C and compiled with the g++ compiler. Both phases are performed on a Sun V240 workstation with 2x1.2GHz UltraSPARC-IIIi CPU's and 2GB of RAM. The results are compared with those of GAs implemented with Microsoft Visual C++ on a PC 2.6GHz Pentium4 CPU and 768 MB of RAM. The GA parameters are obtained by trial-and-error tuning. We consider the population size, the number of generations, and a number

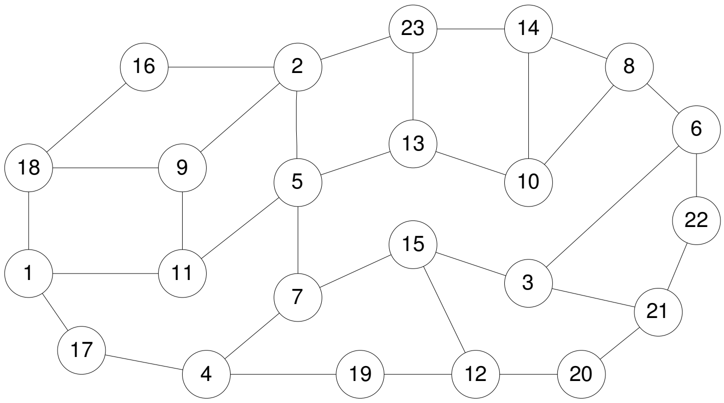


Figure 5.3: Net23v2 (23 nodes, 33 links)

of repetitions that is large enough to generate high-quality solutions within a reasonable amount of computational time. The crossover and mutation rates are fixed at 0.6 and 0.06, respectively. The GA simulation will terminate if there is no improvement after a specified consecutive number of generations (e.g., after 100 generations). The GA program used in comparison is implemented with the population size of 100, and 10 repetitions for each data point. The search will stop if it cannot obtain a better solution after 100 generations since the best solution has been found.

The results are as follows. First we consider the levels of guarantee (α) with different α's based on Table 5.3. As illustrated in Table 5.4, the computational times of the heuristic algorithm are smaller than those of GAs for the same level of solution quality. For the heuristic, although some randomness may occur, computational times tend to be larger for higher levels of guarantee (compare Experiment Net-k1 ($\alpha = 0.95$) with Net-k4 (α=0.999)). The reason is that when α increases, the network need to reserve more bandwidth to guarantee uncertain traffic, thus making the routing problem more difficult to solve due to a higher chance of infeasible solutions.

Note that the heuristic solution times in Table 5.4 are dominated by the times spent in the

Table 5.4: Effect of varying α

Exp.	Heuristic and GA Cost	Heuristic Time (sec)	GA Time (sec)
23-1	4770.00	8.67	45.62
23-11	4516.64	6.30	49.67
23-12	4773.06	8.97	43.77
23-13	5136.36	19.40	48.04
23-2	4528.72	6.59	25.02
23-21	4243.57	5.50	27.56
23-22	4497.47	7.17	29.67
23-23	4801.84	13.89	26.06
23-3	4187.64	5.57	25.43
23-31	3933.08	4.82	26.47
23-32	4185.57	6.12	27.69
23-33	4485.28	6.93	25.98
23-5	4998.01	5.09	25.34
23-51	4655.16	4.83	25.81
23-52	4907.04	5.01	26.42
23-53	5191.93	5.22	26.23

Phase II-Step2 algorithm. For example, in Experiment 23-11, the total time of 6.30 seconds is composed of 1.50 seconds in Phase I and 4.80 seconds in Phase II-Step2. However, the pattern is different for the experiments in Table 5.5, where the bandwidth bounds are increased; this makes the routing problem easier to solve. Next, consider the set 23-4, 23-6, and 23-7, which are the larger-bandwidth versions of the set 23-1, 23-5, and 23-2, respectively. Due to larger bandwidths, changing α's no longer affects the solution times, and the computational times in Phase II do not significantly dominate those in Phase I. For example, in Experiment 23-41 (the larger-bandwidth version of Experiment 23-11), the total computational time is 4.29 seconds. This is composed of 1.45 seconds in Phase I and 2.84 seconds in Phase II-Step2. The results are similar for the rest of the experiments in Table 5.5. The solutions obtained by both the heuristic and GA are the same.

The next parameter that can affect algorithm performance is the traffic demand. Basically,

Table 5.5: When bandwidth bounds (W_j's) are increased.

Exp.	Heuristic and GA Cost	Heuristic Time (sec)	GA Time (sec)
23-4	4732.59	4.21	26.04
23-41	4491.71	4.29	30.92
23-42	4744.20	4.20	28.89
23-43	5025.04	4.23	27.06
23-6	4959.60	4.50	25.42
23-61	4555.55	4.46	26.80
23-62	4793.65	4.60	29.85
23-63	5157.45	4.60	28.46
23-7	4468.97	4.50	27.55
23-71	4188.99	4.31	28.20
23-72	4436.09	4.21	26.33
23-73	4713.30	4.20	30.04

there are two factors related to traffic demand: the demand volume and the number of demand pairs. First, the demand volume can have a similar effect as in previous experiments. Increasing demand volume is similar to increasing α. For example, in Table 5.4 the demands in Set 23-3 are generated with a smaller mean than those in Set 23-1, but used in the same network. This results in smaller computational times on average for the heuristic, and such a pattern is more obvious in the experiments with the same set of demands with Net23v2, as shown in Table 5.6. These results are similar to those of Net23, and the effect is more obvious. For example, the computational times in Experiment set 23v2-4 (whose bandwidth bound is twice as 23v2-1's) significantly decrease.

Besides the demand volume, the number of demands, which is the cardinality of set $\mathcal{D}$ or the length of the solution string, can directly affect computational time. Results are shown in Table 5.7. Note that in the experiment 23-100 - 23-700, the GA parameter settings no longer leads to high-quality solutions. Thus, the additional parameter tuning experiments are needed, and the GA* parameter setting is used. This increases computational level by using the population size of 200, requiring 200 generations before terminating a search if

Table 5.6: Results on Net23v2

Exp.	Heuristic and GA Cost	Heuristic Time (sec)	GA Time (sec)
23v2-1	5231.86	31.18	20.04
23v2-11	4884.58	38.65	14.09
23v2-12	5247.56	30.19	18.89
23v2-13	5566.86	31.50	22.06
23v2-3	4332.68	11.10	31.42
23v2-31	3912.34	7.60	31.42
23v2-32	4156.09	9.09	28.04
23v2-33	4705.19	14.14	33.86
23v2-4	4709.87	4.23	36.42
23v2-41	4450.36	4.38	34.75
23v2-42	4689.31	4.24	40.52
23v2-43	4957.38	4.24	38.54

there is no improvement. This would lead to higher chance to obtain better solutions than the previous GA setting.

As expected, when compared with the previous experiments, it takes longer to solve the experiment set 23-8x, 23-9x, and 23-100 when the number of demands increases. This happens due to 1) the solution space is larger and 2) the total load of the traffic in a network is higher when the total bandwidth is unchanged. In Experiment 23-300, which is the larger-bandwidth version of 23-100, the solution time decreases as expected.

Next, we consider the cases that are affected by the number of demands only, without the effect of the bandwidth bound. Experiment 23-500 (100 demands) and 23-700 (200 demands) are designed such that the network has very large bandwidth bound, and traffic demands are randomly generated with a small value of mean. The results show that the computational times are the largest due to the longest solution strings. Furthermore, the heuristic solution time in Phase I dominates the time in Phase II. For example, in Experiment 23-700, it takes 50.87 seconds in Phase I, but only 8.95 seconds in Phase II. Similar results are shown in Ta-

Table 5.7: Net23: When the number of demands increases.

Exp.	Hueristic Cost	GA Cost	GA* Cost	Heuristic Time	GA Time	GA* Time
23-8	6614.48	same	-	6.09	32.56	-
23-81	6252.18	same	-	6.15	37.61	-
23-82	6581.42	same	-	6.09	40.02	-
23-83	6949.15	6946.39	-	6.25	38.86	-
23-9	7479.58	same	-	7.37	39.02	-
23-91	7115.10	same	-	6.79	42.75	-
23-92	7460.11	same	-	7.47	36.34	-
23-93	7995.49	7985.30	-	8.36	42.75	-
23-100	9950.48	10077.55	9999.87	40.74	47.55	129.19
23-300	9956.10	10060.76	9967.15	8.97	40.66	141.70
23-500	10699.06	11581.59	11268.38	21.00	89.81	319.75
23-700	19501.09	23616.72	22687.17	59.82	287.42	1034.06

Table 5.8: Net23v2: When the number of demands increases.

Exp.	Hueristic Cost	GA Cost	GA* Cost	Heuristic Time	GA Time	GA* Time
23v2-8	8342.29	same	-	16.95	34.89	-
23v2-81	7913.95	7925.38	-	16.74	47.86	-
23v2-83	8301.78	8286.31	-	24.07	27.11	-
23v2-84	8941.94	same	-	30.05	33.09	-
23v2-9	7790.31	same	-	9.65	36.17	-
23v2-94	7380.06	same	-	7.56	36.39	-
23v2-93	7746.41	same	-	8.77	26.41	-
23v2-94	8179.78	same	-	11.80	26.25	-
23v2-100	10838.07	11262.20	11040.68	40.42	37.23	120.97
23v2-300	10499.73	10613.64	10499.73	13.68	45.11	141.69
23v2-500	10042.04	11118.44	10750.96	20.43	122.11	355.69
23v2-700	19169.79	23340.70	22337.56	60.24	272.92	938.56

ble 5.8. When the number of demands increases (e.g., Experiment 23v2-500 and 23v2-700), a longer solution time due to a larger solution space is clearly observed.

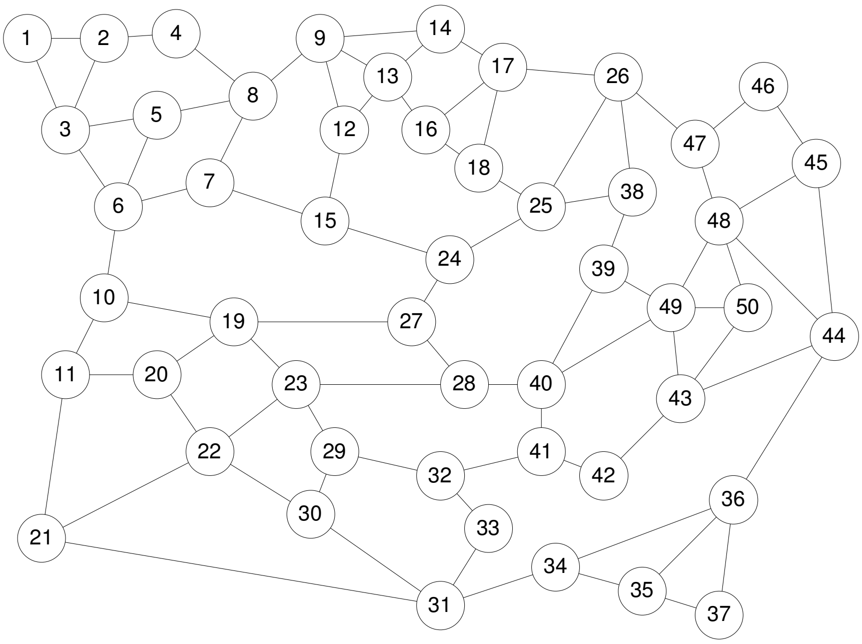


Figure 5.4: Net50 (50 nodes, 82 links)

Finally, the heuristic is tested with a larger network, i.e., a 50-node network Net50 as illustrated in Figure 5.4, and the results are listed in Table 5.9. Additional experiments with GA* are conducted for the last four experiments. As expected, as the network size increases, the solution time also increases due to larger solution space. When the effects of network size and the number of demands are combined (e.g., Experiment 50-100 and 50-200), long computational times are expected. At this point, the GAs are no longer sufficient to obtain the same quality of solutions.

The results for some experiments such as 50-12, 50-73, and 50-9 (with GA*) suggest that the heuristic solution algorithm could be further improved. Although the computational times of the heuristic are small compared with those of GAs for the same level of solution quality, the nonlinear approximation may or may not obtain the optimum solution. In addition, the GA approach could be used to improve the solutions, but it is computational expensive. Thus, if the optimum is desired, additional steps are required at the expense of computation time.

One alternative would be the hybrid between this heuristic and the GA. This means that the GA could be used to explore the solution space with an initial input from the heuristic. Furthermore, a possible future work should consider developing a method for finding the likelihood of path selection (e.g., the One-more-level rule in the heuristic) to further reduce the solution space.

Table 5.9: Results on Net50

Exp.	Number of demands	Bandwidth bound	Heuristic Cost	GA Cost	GA* Cost	Heuristic Time	GA Time	GA* Time
50-1	25	300	6767.31	6779.17	-	22.54	91.70	-
50-11	25	300	6453.27	same	-	20.96	85.38	-
50-12	25	300	6823.90	6808.48	-	21.21	78.72	-
50-13	25	300	7287.01	same	-	24.72	80.86	-
50-7	30	300	9297.43	same	-	24.19	108.31	-
50-71	30	300	8849.35	8871.68	-	15.20	111.44	-
50-72	30	300	9430.03	same	-	23.18	113.06	-
50-73	30	300	10011.91	10000.54	-	27.65	121.51	-
50-8	30	300	8991.40	same	-	23.58	94.63	-
50-81	30	300	8605.82	same	-	14.47	135.06	-
50-82	30	300	9082.13	same	-	24.85	135.69	-
50-83	30	300	9595.56	same	-	14.47	115.06	-
50-9	45	300	8877.89	8974.15	8857.21	23.58	94.63	448.56
50-91	45	300	8384.66	8530.18	8379.43	22.94	218.13	417.84
50-92	45	300	8865.72	9057.64	8854.85	24.85	185.69	411.57
50-93	45	300	9403.83	9505.73	9386.65	25.49	199.66	408.21
50-100	100	1000	10981.13	12487.50	11861.11	48.74	457.56	1496.30
50-200	200	1000	24419.34	28266.59	27462.87	143.80	909.20	3387.55

After the nonlinear link-level guarantee problem can be solved with the heuristic solution algorithm, the cost benefit between the link-level approach and the end-to-end guarantee approach (described by Mathematical Formulation II problem in Section 4.3.2) can be compared. As mentioned in Chapter 4, these two cases are fundamentally different and based on their usages. The end-to-end guarantee approach considers bandwidth allocation for a traffic demand individually, while the link-level approach considers a statistically shared bandwidth allocation for all the demands carried on the same link. Since the benefit of using common bandwidth pooling in the link-level approach becomes clear when there are many number of traffic demands in the network, the comparisons are shown in Table 5.10 as the number of demands is increased from 25 to 200, considering three levels of guarantee in a 50-node network. The calculations are based on the equivalent approximation in Section 4.3.3, which consider the average hop counts in the solutions (ranging from 3.86 to 3.96 hops).

Table 5.10: Comparison of Link-level and End-to-end guarantee designs on Net50

Number of demands	Link-level Experiment	Cost	Equivalent End-to-end Experiment	Cost	Cost Difference
25	50-11(α=0.95)	6453.27	End-to-end level=0.818	6390.81	-62.46
25	50-12(α=0.99)	6823.90	End-to-end level=0.961	7176.22	352.32
25	50-13(α=0.999)	7287.01	End-to-end level=0.996	8094.77	807.76
200	50-201(α=0.95)	23166.96	End-to-end level=0.820	25651.5	2484.54
200	50-202(α=0.99)	24327.69	End-to-end level=0.962	30600.41	6272.72
200	50-203(α=0.999)	25629.88	End-to-end level=0.996	35688.64	10058.76

5.4 SUMMARY

This chapter proposes a heuristic solution to the nonlinear demand uncertainty problem. The solution approach consists of two phases. In the first phase, the nonlinear problem is approximately transformed to a simpler linear model and then solved. The second phase then uses the initial solution from the first phase to further improve the solution. Due to the nature of this nonlinear NP-Complete problem and the need for fast solution algorithms, it is crucial to consider the solution space (which is influenced by a combination of set $\mathcal{A}$, $\mathcal{D}$, and $\mathcal{P}^k$), system parameters, and the tradeoffs between computational time and quality of solutions. Numerical results show that this set of heuristic algorithms perform well compared with Genetic Algorithms in terms of quality of solutions within reasonable amount of time.

6.0 A CHANCE-CONSTRAINED PROGRAMMING BASED SCHEME WITH TRAFFIC CHANGE CONSIDERATIONS

This chapter discusses how the Chance-Constrained Programming based scheme can be practically implemented when traffic change is considered. The main issue concerns how traffic change or perturbation could have an impact on the design of the CCP-based scheme. Requirements and guidelines for a corrective process and network reconfiguration process which could be applied to a two-layer network model are described.

6.1 TRAFFIC CHANGE CONSIDERATIONS

The Chance-Constrained Programming (CCP) based scheme discussed so far assumes that traffic variation could occur within the specified variances or within the uncertainty bounds. However, it is possible that there will be a future event that traffic varies beyond a statistical estimation, with a long fluctuation period. If this situation occurs, the initial CCP solution will no longer be suitable, and must be updated to handle such a change. The updating process, which will be discussed in the next sections, depends on the assumptions on the unknown future traffic and knowledge on traffic information as follows.

- Future traffic

Generally, traffic in the uncertainty model can be classified into two types. In the first type, traffic characteristic rarely changes or traffic volume fluctuates within an estimated bound. This traffic type is assumed to be the majority, and will be referred to as the "main" or "stable" traffic. Clearly, the main type could be handled with the CCP based optimization.

However, the uncertainty model also has flexibility to include the other type of traffic that can arrive after the initial optimization process. This type will be referred to as the "secondary" or "dynamic" type. Thus, there is a need for an additional step or an adaptive process to take into account of such change.

- Knowledge on traffic characteristic

In addition to the future traffic consideration, the other factor that influences the implementation choice is the knowledge on traffic characteristic. In fact, it is possible that sometimes we do not have complete knowledge on some traffic at an optimization time. Therefore, in the initial step of the CCP process, some "guess" values of the unknown parameters may be required initially. After that, whenever such parameters can be obtained or estimated, the original solution will then be adjusted.

Clearly, the above two factors suggests the need for the second step or a "corrective action" in addition to the CCP based optimization step. Before describing the required processes in detail, it is important to see different approaches whose design concepts are related to the assumption of knowledge on demand pattern. In the first approach, when the patterns of traffic at different time periods are completely known or highly predictable, a one-time optimization process can be used. Network resource can be optimized according to the demand pattern in each period. This type of design is referred to as the Multi-Hour network design [6]. There are different optimal solutions obtained for each of time periods. For example, the optimal routing for the 9-12 A.M period is different than that of the 1-5 P.M. period.

Another approach that use a predefined set of demand patterns in a design is a Stochastic Programming with Recourse (as discussed in Chapter 4). Its practical implementation uses a scenario-based method, which is based on optimizing the problem with respect to known (or predicted) scenarios with probability estimate for each demand scenario. Thus, scenarios that represent the future must be selected carefully. An example is the work of Leung and Grover [19, 20] on a capacity planning of survivable networks.

In contrast to the known-pattern approaches as mentioned above, there is a "dynamic" or "non-pattern" approach in a sense that it does not require the knowledge on traffic pattern in advance. For example, in the work of Gençata and Mukherjee [32] on designing a reconfiguration algorithm for a WDM network, the topology is dynamically reconfigured or adapted itself according to the current traffic on links. Such process is based on a periodic measurement on traffic loads and utilization thresholds. The resource, which is the number of lightpaths in this study, can be increased or decreased to satisfy demand based on a utilization goal.

Compared with those models, the CCP based scheme could be viewed as an alternative approach. This is because traffic variation could be handle with statistical guarantee without the need for obtaining exact traffic parameters. A well-calculated bound obtained from the first step could help the network operator not to update traffic routes very often (as in the fully-dynamic approach). In this dissertation, since each traffic demand is broken into the certain and uncertain parts. The uncertain part, which represents traffic variation specified by a variance parameter, is guaranteed based on the uncertain guarantee level. Therefore, there is no need to update the solution often except that the variation occurs beyond the bound for a long period of time, or the bound could be improve for better network utilization. Before performing the CCP re-optimization, the corrective action may be suitable for some cases. The requirement for an integrated scheme, which is the combination of the CCP optimization and the corrective step will be discussed in later sections. First, the scenarios of traffic changes will be described next.

6.2 SCENARIOS OF TRAFFIC CHANGE AND ISSUES ON A CORRECTIVE PROCESS

The previous section suggests the need for the second process, called the *corrective process*, in addition to the CCP optimization process. Assuming that the corrective process can measure traffic periodically, the traffic changes can be classified into two scenarios:

1. An arrival of a new demand.
2. A large increase in traffic volume of the existing demands.

These changes may lead to resource insufficiency if they are significant. For example, some new demand may be blocked when there is no available bandwidth along the routes. Furthermore, although the bandwidth on a link is still enough, using more bandwidth on such link may lead to higher utilization than a preferred level. To response with such issues, an adaptive process or a corrective step is needed in addition to the initial CCP design. The adaptive process should find a new solution to support the change. In fact, the new solution should be correlated to the previous one so that there is little network performance degradation.

The scheme mentioned so far start with the initial optimization process using the CCP. When traffic change is detected, the corrective process will be performed. For example, a basic process could be designed as shown in Figure 6.1. If the change is only an increase in demand volume, the process will check whether it satisfies the bandwidth bounds (i.e., W_j in Chapter 3). Sometimes the bandwidth bound can be relaxed to meet the need if there are available resources. This is similar in the case of a new demand arrival. In both case, if it is not possible to route the traffic on the existing routes, the network operator will simply establish a new route to carry this new traffic, reroute or rearrange some traffic, or re-optimize the network (i.e., by solving the basic uncertainty problem with a new set of demands).

In practice, re-optimization with the CCP approach will be performed as necessary since it is not efficient to do so whenever there is a traffic change. Thus, basic actions of the corrective process are needed. Note that as the network characteristics (e.g., route or bandwidth) change, there is a consequence. First, the levels of guarantee will not be the same as the original values. Second, a new solution, which is used for handling new traffic, may not have the same objective as before (i.e., the minimal cost objective). These issues will be discussed in detail in a later section.

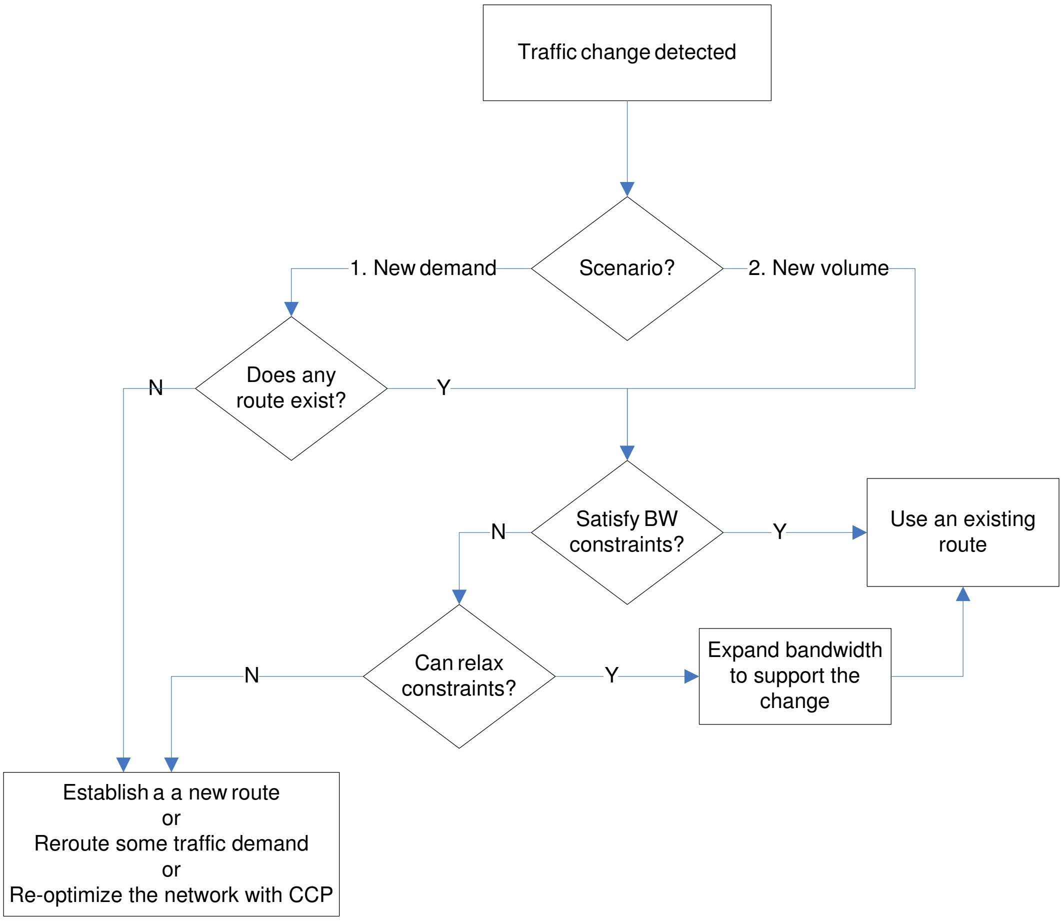


Figure 6.1: A basic corrective process

The approach and processes as described so far are in a context of a one-layer network. The next section will discuss the uncertainty effect on different layers as an important issue in designing the corrective process.

6.3 EFFECT OF THE UNCERTAINTY ON A TWO-LAYER NETWORK

When considering a two-layer network (e.g., an IP-over-WDM network), one important question would be “Which layer should perform the corrective operation when traffic change?” To investigate this, traffic granularity offered at each layer should be considered first since the higher the layer, the finer the granularity. Considering the routing in a two-layer network, the amount of required bandwidth for the uncertainty from the top layer will be passed down to the lower layer to prepare the physical resource to carry traffic. Basically, the lower layer (or a physical layer) will allocate resource by determining a combination of wavelengths used in optical fibers. The process of selecting appropriate wavelengths and fibers will be performed by an algorithm known as *Routing and Wavelength Assignment* (RWA) [10]. The RWA problem is not in the scope of this dissertation, and assume to be further work in the physical layer. In short, the operation at the top layer (i.e., the IP layer) could handle traffic at a very fine level of granularity, whereas, the operation at the bottom or the optical layer can be performed at much more coarser granularity, i.e., at the wavelength or the fiber level. Thus, with the granularity requirement on each network layer, it is clear that the upper layer would be appropriate in performing the CCP optimization process since the resulting uncertainty bounds (which is calculated with the levels of guarantee) should have fine granularity. Otherwise, with coarse granularity, the resulting values would be just loose bounds, thus wasting the network resource.

The uncertainty as discussed so far is seen by the upper layer. Since the lower layer deals with traffic in much coarser granularity, the effect of the uncertainty would be much smaller than the upper layer. Another way is to consider the levels of guarantee perceived at different layers. Such analysis could be used at least to roughly quantify the effect of the uncertainty. To understand the effect, firstly let’s consider how traffic demand could be mapped from the upper layer to the lower layer, as depicted in Figure 6.2. The links in the upper layer (i.e., the virtual links which are obtained by solving the uncertainty problem in Chapter 5) can be viewed as the input demand-pairs of the lower layer. Since the network considered here is based on the overlay model, the lower-layer network has its own separate routing

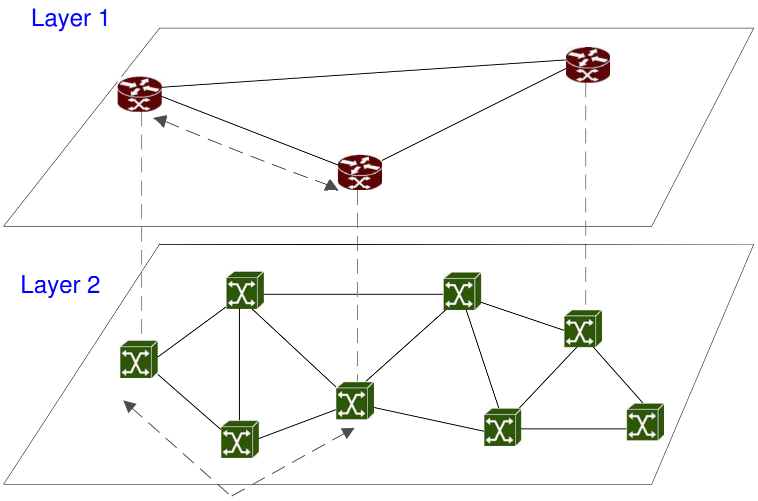


Figure 6.2: Routing in a two-layer network

mechanism. Thus, basically the routing process in the lower-layer network aims to find routes that corresponds to the demands or the set of virtual links from the upper layer. A simple solution could be obtained by solving a linear programming problem with a mathematical formulation described below:

Problem LP2:

$$\min_{y_j,\ z_j} \sum_{j \in \mathcal{A}} C_j\ y_j\ +\ F_j\ z_j \tag{6.1}$$

subject to:

$$\begin{aligned}
\sum_{k \in \mathcal{D}} \sum_{p \in \mathcal{P}^k} \gamma_j^{k,p}\ g^{k,p}\ U_k &\leq My_j\ ,\ \forall j \in \mathcal{A} && (6.2)\\
My_j &\leq B_j\ ,\ \forall j \in \mathcal{A} && (6.3)\\
\sum_{p \in \mathcal{P}^k} g^{k,p} &= 1\ ,\ \forall k \in \mathcal{D} && (6.4)\\
\gamma_j^{k,p}\ g^{k,p} &\leq z_j\ ,\ \forall j \in \mathcal{A},\ \forall k \in \mathcal{D},\ \forall p \in \mathcal{P}^k && (6.5)\\
y_j &\geq 0, integer\ \forall j \in \mathcal{A} && (6.6)\\
g^{k,p},\ \ z_j &\in \{0, 1\} && (6.7)
\end{aligned}$$

Let a network be defined by a graph of the second layer network $(\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is a set of nodes, and $\mathcal{A}$ is a set of links. The notation is as follows:
Let

- $\mathcal{D}$ denote the set of demands. This is the same as the set of virtual links of the upper layer (obtained from the solution of the upper-layer optimization process).
- U_k denote the volume of demand $k \in \mathcal{D}$ (or the bandwidth allocated on the virtual link k of the upper layer). This parameter is given as a solution obtained from the upper layer.
- $\mathcal{P}^k$ denote a predefined set of candidate paths for demand $k \in \mathcal{D}$.
- C_j denote the capacity cost per capacity module for link $j \in \mathcal{A}$.
- F_j denote the fixed operation cost on link $j \in \mathcal{A}$.
- B_j denote the capacity bound on link $j \in \mathcal{A}$.
- M denote the generic number of capacity units per module used in the lower layer.
- $\gamma_j^{k,p}$ denote a binary parameter. It is equal to 1 if path $p \in \mathcal{P}^k$ for demand k uses link j, and 0 otherwise. $\gamma_j^{k,p}$ is given when the set of possible paths $\mathcal{P}^k$ is calculated.
- $g^{k,p}$ denote a decision variable, which is equal to 1 if the flow of demand k selects path p from the predetermined path set $\mathcal{P}^k$, and 0 otherwise.
- y_j denote an integer decision variable, representing the number of capacity modules required to route traffic on link $j \in \mathcal{A}$.
- z_j denote a decision variable. It is equal to 1 if link j is used, and 0 otherwise.

This formulation for routing in the lower layer is simply a minimum cost routing model. In the objective function 6.1 , the total cost consists of the cost of using capacity (variable cost) and the fixed cost incurred from the operational or equipment cost for the selected link. Constraints 6.3 ensures that the amount of capacity required to carry traffic on link j (My_j in 6.2) does not exceed the capacity bound B_j. Constraints 6.4 enforce the non-bifurcated (single path) flow allocation scheme so that the traffic requested from the upper layer can be easily managed in the same flow. Constraints 6.5 adds a fixed cost if link j is used to carry traffic. Once this problem is solved, the resulting solution will be further used by the

RWA algorithm to assign appropriate wavelengths to carry traffic.

Again, the process in the physical layer deals with much coarser traffic granularity, e.g., a multiple of M capacity units as shown in the problem above. To investigate the effect of the uncertainty, for simplicity let consider the effect of one random demand on a link at the upper layer, which will pass the requirement down to the lower layer. Suppose that link j in the lower layer is selected (consider constraints 6.2). Let further assume that on link j only one module is used to carry traffic ($y_j = 1$), then the the total capacity required is M. Therefore, the sum of all the traffic demands, carrying on link j can be written as:

$$\sum_{k \in \mathcal{D}} U_k \quad \leq \quad M \tag{6.8}$$

Consider only the effect of one traffic demand U_i, where $U_i, i \in \mathcal{D}$, denotes the allocated bandwidth on the considered link in the upper layer (e.g., see Figure 6.2). This can be written as

$$U_i \ + \sum_{k \in \mathcal{D},\ k \neq i} U_k \quad \leq \quad M \tag{6.9}$$

Let the rest of U_k be a constant $C = \sum_{k \neq i} U_k$.

$$U_i \quad \leq \quad M \ - \ C \tag{6.10}$$

Since U_i is the value calculated from the deterministic equivalent of the random demand, U_i can be written in the following form,

$$\mu \ + \ \Phi^{-1}(\alpha)\sigma \ \leq \ U_i \ \leq \ M - C \tag{6.11}$$

Since the aggregated demand on the link in the upper layer could be represented as a Gaussian random variable ξ (i.e., $\xi \sim N(\mu, \sigma^2)$), thus

$$P\,(\ U_i \geq \xi\ ) \geq \alpha \tag{6.12}$$

Since $M - C \geq U_i$, the following also holds:

$$P\,(\ M - C \geq \xi\ ) \geq P\,(\ U_i \geq \xi\ ) \geq \alpha \tag{6.13}$$

Therefore, there exists $\alpha^* \geq \alpha$ such that

$$P\,(\ M - C \geq \xi\ ) \geq \alpha^* \geq \alpha \tag{6.14}$$

This means that while link i in the upper layer can handle the demand ξ with the α-guarantee, the link j in the lower layer can handle the same demand with the α^*-guarantee, where $\alpha^* \geq \alpha$. Note that in general the capacity M offered in the lower layer is much greater than the carried traffic. In other words, the effect of the uncertainty in the lower layer is much less than that in the upper layer due to its high bandwidth.

The relation between parameters of the traffic demand (mean and standard deviation), capacity provisioned, and guarantee levels are quantitatively illustrated by the graphs in Figure 6.3 and Figure 6.4.

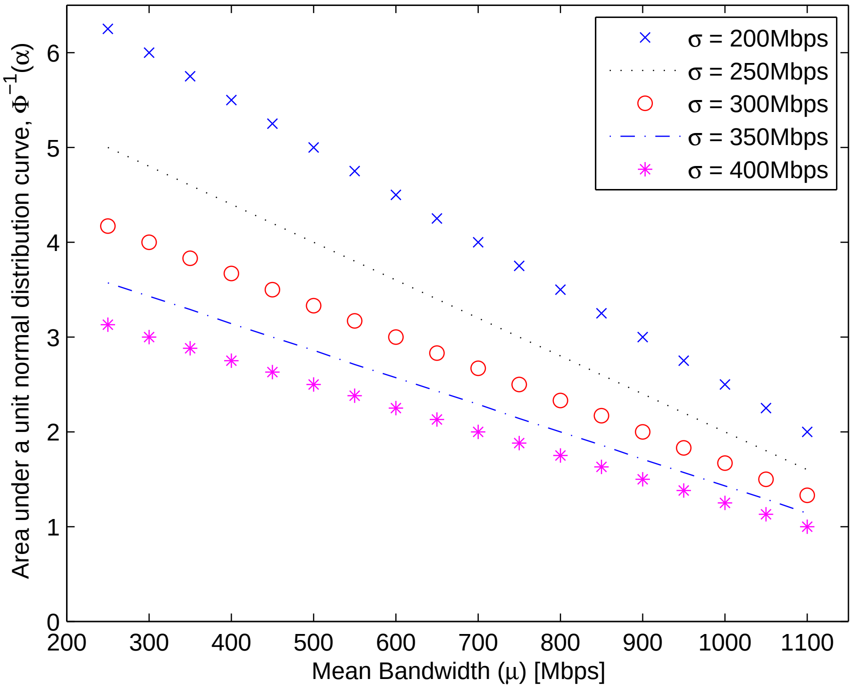


Figure 6.3: Effect of varying traffic demand parameters (I)

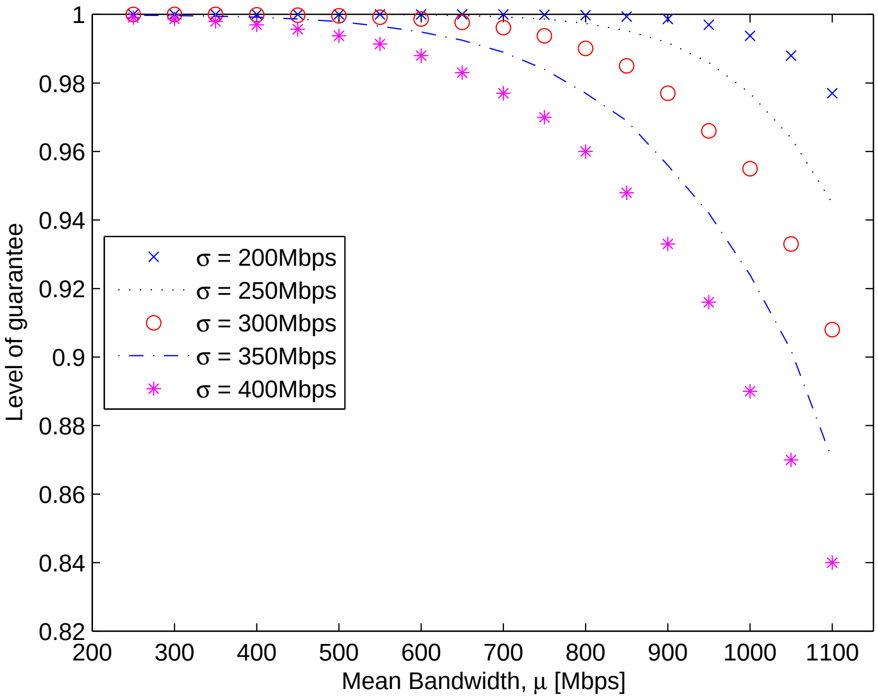


Figure 6.4: Effect of varying traffic demand parameters (II)

These are plotted with the following assumptions. First, let the capacity unit of one wavelength (M) be 2.5 Gbps, and let the capacity of total traffic demands except for the considered demand (C) equal to 1 Gbps. Then, each line of the graph is plotted by varying μ with a fixed σ. The effects of changing parameter μ and parameter σ can be seen on the plots. Figure 6.3 shows the effect of these parameters on the levels of guarantee in terms of area under a unit normal distribution or $\Phi^{-1}(\alpha)$ as appeared in the Chance-constrained deterministic equivalent form or in Inequality (6.11). Figure 6.4 is obtained by determining the levels of guarantee from their inverse transforms ($\Phi^{-1}(.)$) in Figure 6.3. As shown in the figure, when the magnitude of demand uncertainty increases (by increasing μ or σ), the level of guarantee decreases.

In the next section, the practical implementation of the corrective process to be used in case of traffic change will be discussed.

6.4 IMPLEMENTATION OF THE CORRECTIVE PROCESS

The purpose of the corrective process is to adaptively and quickly adjust network resource according to the change of traffic demand with small perturbation to the existing traffic. To investigate the requirements of the corrective process. First, we need to understand the goal of the design. Originally, the objective of the first step process (CPP optimization) is to obtain the minimum cost design that can handle the uncertainty, assuming that traffic variation could be covered by the calculated bounds. When there is a significant change and it is not efficient to re-optimize the whole network, a simple corrective process would be more appropriated option. However, the new solution obtained may not yield the minimum cost as the original solution. Thus, the tradeoffs should be considered.

The basic corrective process shown in Figure 6.1 checks whether it is possible to use the existing routes or could expand the bandwidth to handle the change. Such operation is performed in order to avoid perturbation to the existing traffic. However, if bandwidth expansion is not allowed, it would be more efficient to reroute some existing traffic demands in order to cover the change. This could lead to net work performance degradation such as traffic disruption which could occur during the rerouting process.

Next, the scenarios that the volume of the existing demands has been increased will be discussed. Recall from Chapter 3 that the link bandwidth used to be carry the total demand should be at least:

$$x_j = \sum_{k=1}^{K} \mu_k + \Phi^{-1}(\alpha_j) \sqrt{\sum_{k=1}^{K} \sigma_k^2} \tag{6.15}$$

Now, if one demand has changed its probabilistic profile from $N(\mu_1, \sigma_1^2)$ to $N(\mu_2, \sigma_2^2)$, and the demand volume increases, possible scenarios could be classified as follows:

Case 1: The bandwidth expansion is allowed to cover the change. This case will require no further action. Since the bandwidth is increased, the level of guarantee (α) would decrease or increase depending on the term on the left (i.e., the new bandwidth allocated) and on the right (the mean and the function of variances) of Equation (6.15).

Case 2: The bandwidth is not allowed to expand, but the changed traffic demand can still be carried on its current path. This could be performed by decreasing the level of guarantee from α to α^{**} in order to compensate for the increase in the mean in the first term and the square root of variances in the second term as follows:

$$x_j = \sum_{k=1}^{K} \mu_k^{**} + \Phi^{-1}(\alpha_j^{**})\sqrt{\sum_{k=1}^{K} \sigma_k^2}$$

In other words, the level of guarantee is decreased in order to satisfy the bandwidth constraint when some traffic demand volume is increased. As a result, the uncertain part of the traffic (i.e., the function of variances) will be less supported.

Case 3: If it is impossible to use the existing route, rerouting traffic would be the option. The factors to be considered for the tradeoffs in the rerouting process are described as follows:

1. Additional cost

Obviously, rearranging traffic could yield many feasible solutions whose costs are higher than the optimal one obtained by a re-optimization process. This is because traffic demands may be inefficiently grouped, resulting much more unnecessary amount of network resources. Furthermore, some rerouted traffic may not use the minimum cost path, thus unnecessarily consuming more network resources.

2 Performance factors

There are three main performance factors to be considered:

2.1 Load balancing

A well-balanced network could delay resource exhaustion, with a higher chance that the future demand request will be routed without being blocked. Here we consider the maximum link utilization factor as a metric; this is a ratio of the actual bandwidth requested to the link bandwidth provided. The basic way to include the load balancing feature is to design the network such that the maximum link utilization of all links is minimized.

2.2 Network delay performance

The network delay can be controlled by limiting the number of hops in a rerouting or reconfiguration process. It is preferable to route the traffic within the small number of hop(s) as much as possible to limit network delay.

2.3 Disruption of traffic

Disruption of services of the existing traffic during rerouting traffic in the corrective process is undesirable. Therefore, if traffic rerouting is necessary, new network configurations with small number of changes from the previous configuration are preferable.

With these factors, some desirable features such as, fast computation time, small additional cost, well-balanced load, small network delay, and small number of changes from the previous solution should be included in the design of the corrective process. However, some of these objectives may be conflicting with each other. For example, instead of rerouting traffic on the shortest path, some traffic demands may be rerouted on the longer-hop paths in order to avoid traffic congestion on some link (i.e., routing with a load-balancing goal), thus conflicting the minimum-cost goal. Therefore, the goal depends on the network policy, and is also based on assigned priorities. For example, the requirements could be prioritized as follows:

1. Load-balancing is the main objective.
2. An additional cost must be within a specified cost budget.
3. The new levels of guarantee of the new network configuration should be satisfied.
4. The transition from the original to a new network configuration should require a small number of changes as much as possible.

The choice depends on the network policy. For example, the rerouting process may consider preparing the resource to handle the future variation. As a result, the goal is to determine a configuration that has high possibility to handle traffic change. Therefore, the load balancing is set as the main objective, and the rest of the factors are set as subgoals, which could be in forms of constrains or secondary decision metrics. Priorities of subgoals depend on network policy.

According to these rules, a fast rerouting or reconfiguration process is suggested as follows:
1. Based on the minimum-cost configuration, the network operator generates a set of correlated configurations, called the *Correlated Backup Set*. When the reconfiguration is required, a configuration in this set may be selected. Since they are correlated with each other, changing from one to another configuration is expected to take a small amount of reconfiguration time.
2. To construct the Correlated Backup Set, the algorithm randomly generates a set of solutions or structures (configurations) with the new input (including the demand with increasing volume) with small variations compared to the base structure (the minimum-cost configuration). This can be achieved easily with a mutation-like process (as in the Genetic Algorithm literature) with a small probability of mutation.
3. The resulting solutions are then ranked as follows:
3.1 Select the best least-congested N solutions, and rank them according to their maximum link utilization values.
3.2 Depending on network policies, further ranking rules may be applied. For example,

- If the maximum link utilizations of two solutions are equal, the smaller-cost topology ranks higher. If they have the same cost, the solution with a smaller number of changes ranks higher. Note that another possible policy could consider the number of changes more important than the cost.
- If all of the above metrics are equal, link utilization histograms (which will be explained next) may be used for further evaluation.

Example: Consider the lower-layer network as shown Figure 6.5, and some demand volumes are increasing. The initial solution (before traffic change) is as follows:

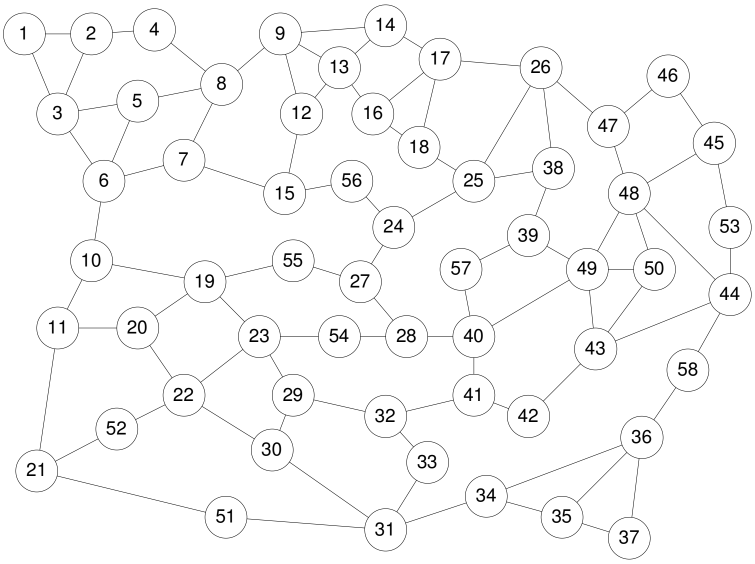


Figure 6.5: Net58 (58 nodes, 90 links)

Path solution(same format as the solution pattern in Chapter 5):

`1 1 4 4 0 1 1 1 1 1 2 0 1 0 0 1 0 0 0 1 0 0 0 0 1 3 1 0 0 0 5 2 0 1 0 0`

Cost: 12392

Utilization histogram: `1 15 5 6 6 6 3 2 0 0`

The utilization histogram shows the number of links that have link utilizations within the ranges: (0.0-0.1], (0.1-0.2], . . . , (0.9-1.0]. For example, there are 15 links with link utilization values in the (0.1-0.2] range.

With the new input, the following 15 solutions for the Correlated Backup Set are selected from the generated 500 solutions with the additional cost budget of 500.00 and probability of mutation equal to 0.2. They are ranked as follows:

Rank: Solution

```
1 :  1 1 4 4 0 1 0 1 1 1 2 0 1 0 0 0 0 0 0 2 0 0 0 0 1 3 1 1 0 0 5 2 1 1 0 0
2 :  1 1 4 4 0 1 0 1 1 1 2 0 1 0 0 1 0 1 0 2 0 0 0 0 2 3 1 0 0 0 5 2 0 1 0 1
3 :  1 1 4 4 0 1 0 1 1 1 2 0 1 0 0 1 0 0 0 1 0 0 1 0 1 4 1 0 0 0 1 2 0 1 0 0
4 :  1 1 4 4 0 1 0 1 1 1 2 0 1 0 0 1 0 0 0 1 0 1 0 0 1 3 1 1 0 0 5 2 0 1 0 0
```

```
5 :  1 1 4 4 0 1 0 1 1 1 2 0 1 0 0 1 0 0 0 1 0 0 1 0 1 4 1 0 0 0 5 2 0 1 0 0
6 :  1 1 3 4 0 1 0 1 1 1 2 0 1 0 0 1 1 0 0 1 0 0 1 0 1 4 1 0 0 0 5 2 0 1 0 0
7 :  1 1 0 4 0 1 0 1 1 1 2 0 1 0 0 1 0 1 0 1 0 1 1 0 2 3 1 0 0 0 5 2 0 1 0 0
8 :  1 1 4 4 0 1 0 1 1 1 2 1 1 0 0 1 0 0 0 1 0 0 1 0 2 3 1 0 0 0 5 2 0 1 0 0
9 :  1 1 4 4 0 1 0 1 1 1 2 0 1 0 0 1 0 0 0 0 1 0 1 0 2 3 1 0 0 0 5 2 0 0 0 0
```

The values of maximum link utilization, cost, the number of route changes (from the minimum structure), and utilization histogram of this set are shown in Table 6.1.

Table 6.1: The Correlated Backup Set

Rank	Max Util.	Cost	No. Changes	Util. Histogram
1	0.652	12892	5	2 16 6 8 8 6 3 0 0 0
2	0.660	12792	5	2 14 6 11 6 5 4 0 0 0
3	0.660	12892	4	3 16 6 8 6 5 5 0 0 0
4	0.664	12695	3	2 15 5 8 5 8 4 0 0 0
5	0.664	12695	3	2 15 7 6 5 6 6 0 0 0
6	0.664	12695	5	1 15 7 8 5 7 4 0 0 0
7	0.664	12857	6	3 14 7 9 6 9 1 0 0 0
8	0.666	12792	4	2 17 5 8 4 6 6 0 0 0
9	0.666	12892	6	2 18 7 6 4 6 6 0 0 0

Note that if the first three factors equal, the utilization histogram may be used in further evaluation. In this example, the demand units and the bandwidth cost for all links are the same, thus, it is possible that the total costs could be equal. For example, consider the forth solution and the fifth solution that have the same maximum link utilization, cost, and the number of changes. Next, we consider the non-zero number of links in the highest range from the utilization histogram (i.e., the (0.6-0.7] range), and the 4th solution is ranked better than the 5th solution (i.e., 4 links vs. 6 links in the utilization (0.6-0.7]). Furthermore, we can get a better view about resource utilization of the network when the maximum link utilization and utilization histogram are used together.

This ranking scheme is used for selecting the first N highly correlated solutions with the best load-balancing characteristics in the total generated solutions. This does not mean that the higher ranked solutions could perform better than the lower ones in terms of ability to handle the future traffic change. However, the reconfiguration process could switch from one configuration to another one within a short period of time, with a small impact on the existing traffic. Note that the uncertainty bounds used in the upper layer optimization process already takes into account the major source of traffic variations, and the capacity in the lower layer is then allocated according to the optimization solution in the upper layer. Therefore, we expect that the reconfiguration in lower layer would not be required very often, except for a significant change that would require a re-optimization of the whole network. Thus, a simple and quick reconfiguration process such as the suggested approach with the Correlated Backup Set would have a benefit. For more intensive calculation, a multi-object programming approach would be required for further needs.

After receiving a resource information update from the upper layer, initially the lower layer would try to carry the new load via the existing connections in order to avoid some undesirable effects (which will be described next) that may arise in a reconfiguration process. For example, in the WDM layer (the lower layer), it will increase the number of wavelengths to support the increasing IP traffic. If there is not enough resource, the following options may be selected:

1. Re-optimize the network by solving Problem LP2 with a new set of input parameters.

2. Reconfigure or reroute some traffic connections while considering an additional cost and network performance factors.

The first choice yields the minimum cost configuration of the lower layer, but the main drawback is that it could lead to many changes from the original configuration. This is because it may require tearing down many connections (i.e.,lightpaths), and then establishing the new ones, resulting in traffic disruption. Furthermore, changing routes in the lower layer increase complexity in resource assignment operations (e.g., a wavelength assignment process). Thus, if it is not necessary to re-optimize the whole network, the second choice would be more appropriate similar to the process in the upper layer.

6.5 SUMMARY

This chapter discusses the practical implementation of the Chance-constrained based optimization scheme. The effect of the uncertainty in the upper layer is greater than that of the lower layer; thus, this will influence the design process. Since some change scenarios would require traffic rerouting or rearranging of network resources to support such changes. Thus, a reconfiguration approach to handling the uncertainty is needed. The guidelines for the processes in the upper and the lower layers, requirements, and a suggested simple reconfiguration process are discussed.

7.0 CONCLUSION

This research provides a methodology for network design under demand uncertainty. The framework begins with the assumptions on requirements of future network. We assume that the traffic distribution of traffic demands carried on links in the network can be approximated by the Gaussian distribution. As a result, it is possible that a traffic demand could be represented in the form of a pair of mean and variance to characterize the random demand. In the network design literature, Stochastic Programming approach is suited to some cases such as a long-term or multi-stage planning design. An implementation based on the Stochastic Programming still requires a few assumptions such as the expected traffic patterns and their associated probabilities of occurrence, even though the future is difficult to predict. Therefore, such assumptions could be avoided with the Chance-Constrained Programming based design approach. This alternative approach handles the uncertainty via the levels of guarantee. We define a new approach such that a traffic demand consists of two portions: the certain part and the uncertain part. The certain part is 100%-guaranteed as in the mean-based reservation approach. The uncertain part is statistically guaranteed, and is derived from the variance of the demand and the level of guarantee, which is based on available network resource or network policy. These two components are used in the deterministic equivalent of the Chance-constraint.

Based on the deterministic equivalent form, an optimization model is then developed. The resulting formulation of the proposed model is a nonlinear multi-commodity flow problem, which requires heuristic solution algorithms to solve. We develop a set of heuristic algorithms to solve the design problem with fast and reasonably good quality. These optimization algorithms can be used in a time-critical environment, and could be applied in solving other similar nonlinear problems. Note that the solutions produced by this set of algorithms are approximated solutions to the nonlinear problem, and some solutions could be further improved depending on time permitted.

Finally, we provide the implementation guidelines for developing a corrective or updating process for handling traffic change. This is crucial because it is possible that the future traffic would vary beyond a statistical estimation, while the Chance-Constrained Programming based optimization assumes that traffic variation could occur within the specified variances or within the uncertainty bounds. As a result, a corrective process is required to handle the change since generally the total re-optimization is not preferable. How the Chance-Constrained Programming based scheme can be practically implemented under traffic variation depends upon the impact of the uncertainty. Basically, the granularity of a network layer will be a crucial factor to determine which layer should perform the corrective operation when traffic change. In addition, if we consider the effect of the uncertainty on the levels of guarantee provided for the traffic, it is clear that the effect of the uncertainty in the lower layer is much less than the effect the upper layer due to its high bandwidth. Thus, the direct impact on the top network layer is to be considered in designing the optimization and corrective strategy.

There are several possible future research studies that could extend this framework:

- Simulation studies on the relation between different traffic patterns and the benefit of the Chance-constraint approximation are needed. The first important topic could be a study using other traffic demand distributions. This is because the Gaussian distribution is assumed only for a large aggregation of traffic demands. The second simulation study should investigate the effectiveness of using uncertainty bound considering different demand patterns. The third study should investigate the performance of the CCP optimization process and the corrective process
- Since the uncertainty models used in this framework are based on a simple graph model, further studies with various scenarios and the input data from measurement are needed.
- A survival network design that considers the demand uncertainty factor. This study

should determine resource planning and statistically guarantee or how much we should reserve resource in case of failure with the uncertain traffic demand.

- Research on methodologies and techniques for dynamic routing and resource allocation under demand uncertainty using a multi-objective programming approach. Since the corrective process used for handling the uncertainty could have multi-objectives, the tradeoffs and multi-objective approach would be needed further study.

APPENDIX

A: NETWORK INFORMATION AND DEMAND INPUTS

Some selected sets of data input are listed in this section.

Table A1: Demand matrix for the set Net23-0 - Net23-04

Demand	Mean	Variance
1-3	46.85	99.42
1-9	42.61	71.17
1-14	39.21	66.70
2-11	44.19	61.30
2-22	44.39	88.02
3-6	33.01	82.03
3-15	52.14	68.99
4-14	41.78	84.04
4-18	52.86	78.39
5-20	44.43	52.96
6-20	37.03	52.51
7-13	55.36	65.25
8-17	50.80	50.75
8-23	59.69	98.54
9-19	41.49	89.44
10-15	32.28	74.92

Table A2: Net23-0 (23 nodes, 33 links)

$Link_j$	c_j	$\Phi^{-1}(\alpha_j)$	W_j	F_j
1-2	1.50	1.282	200	40
1-13	1.25	1.645	200	40
1-22	1.25	1.645	200	40
1-23	1.25	1.645	200	40
2-5	1.50	0.842	200	40
2-9	1.25	1.645	200	40
2-20	1.25	0.842	200	40
2-23	1.25	0.842	200	40
3-16	1.00	0.842	200	40
3-17	1.25	0.842	200	40
3-18	1.00	1.282	200	40
4-7	1.50	1.282	200	40
4-11	1.50	0.842	200	40
4-19	1.50	0.842	200	40
5-9	1.50	1.645	200	40
5-19	1.25	1.282	200	40
6-12	1.00	0.842	200	40
6-14	1.50	1.282	200	40
7-16	1.00	1.282	200	40
8-13	1.50	1.282	200	40
8-21	1.25	1.645	200	40
9-20	1.25	1.645	200	40
10-16	1.25	1.282	200	40
10-17	1.25	1.282	200	40
11-12	1.50	1.282	200	40
12-14	1.50	1.282	200	40
14-20	1.50	1.645	200	40
15-17	1.50	1.645	200	40
15-18	1.00	1.282	200	40
15-22	1.00	1.645	200	40
18-19	1.00	1.645	200	40
18-23	1.00	1.645	200	40
21-22	1.00	0.842	200	40

Table A3: Net23-1 (23 nodes, 33 links)

$Link_j$	c_j	$\Phi^{-1}(\alpha_j)$	W_j	F_j
1-2	1.50	3.090	200	40
1-13	1.25	1.645	200	40
1-22	1.25	1.645	200	40
1-23	1.25	1.645	200	40
2-5	1.50	2.326	200	40
2-9	1.25	1.645	200	40
2-20	1.25	2.326	200	40
2-23	1.25	2.326	200	40
3-16	1.00	2.326	200	40
3-17	1.25	2.326	200	40
3-18	1.00	3.090	200	40
4-7	1.50	3.090	200	40
4-11	1.50	2.326	200	40
4-19	1.50	2.326	200	40
5-9	1.50	1.645	200	40
5-19	1.25	3.090	200	40
6-12	1.00	2.326	200	40
6-14	1.50	3.090	200	40
7-16	1.00	3.090	200	40
8-13	1.50	3.090	200	40
8-21	1.25	1.645	200	40
9-20	1.25	1.645	200	40
10-16	1.25	3.090	200	40
10-17	1.25	3.090	200	40
11-12	1.50	3.090	200	40
12-14	1.50	3.090	200	40
14-20	1.50	1.645	200	40
15-17	1.50	1.645	200	40
15-18	1.00	3.090	200	40
15-22	1.00	1.645	200	40
18-19	1.00	1.645	200	40
18-23	1.00	1.645	200	40
21-22	1.00	2.326	200	40

Table A4: Net23-1v2 (23 nodes, 33 links)

$Link_j$	c_j	$\Phi^{-1}(\alpha_j)$	W_j	F_j
1-11	1.50	3.090	200	40
1-17	1.25	1.645	200	40
1-18	1.25	1.645	200	40
2-5	1.25	1.645	200	40
2-9	1.50	2.326	200	40
2-16	1.25	1.645	200	40
2-23	1.25	2.326	200	40
3-6	1.25	2.326	200	40
3-15	1.00	2.326	200	40
3-21	1.25	2.326	200	40
4-7	1.00	3.090	200	40
4-17	1.50	3.090	200	40
4-19	1.50	2.326	200	40
5-7	1.50	2.326	200	40
5-11	1.50	1.645	200	40
5-13	1.25	3.090	200	40
6-8	1.00	2.326	200	40
6-22	1.50	3.090	200	40
7-15	1.00	3.090	200	40
8-10	1.50	3.090	200	40
8-14	1.25	1.645	200	40
9-11	1.25	1.645	200	40
9-18	1.25	3.090	200	40
10-13	1.25	3.090	200	40
10-14	1.50	3.090	200	40
12-15	1.50	3.090	200	40
12-19	1.50	1.645	200	40
12-20	1.50	1.645	200	40
13-23	1.00	3.090	200	40
14-23	1.00	1.645	200	40
16-18	1.00	1.645	200	40
20-21	1.00	1.645	200	40
21-22	1.00	2.326	200	40

Table A5: Net23-13 (23 nodes, 33 links)

$Link_j$	c_j	$\Phi^{-1}(\alpha_j)$	W_j	F_j
1-2	1.50	3.090	200	40
1-13	1.25	3.090	200	40
1-22	1.25	3.090	200	40
1-23	1.25	3.090	200	40
2-5	1.50	3.090	200	40
2-9	1.25	3.090	200	40
2-20	1.25	3.090	200	40
2-23	1.25	3.090	200	40
3-16	1.00	3.090	200	40
3-17	1.25	3.090	200	40
3-18	1.00	3.090	200	40
4-7	1.50	3.090	200	40
4-11	1.50	3.090	200	40
4-19	1.50	3.090	200	40
5-9	1.50	3.090	200	40
5-19	1.25	3.090	200	40
6-12	1.00	3.090	200	40
6-14	1.50	3.090	200	40
7-16	1.00	3.090	200	40
8-13	1.50	3.090	200	40
8-21	1.25	3.090	200	40
9-20	1.25	3.090	200	40
10-16	1.25	3.090	200	40
10-17	1.25	3.090	200	40
11-12	1.50	3.090	200	40
12-14	1.50	3.090	200	40
14-20	1.50	3.090	200	40
15-17	1.50	3.090	200	40
15-18	1.00	3.090	200	40
15-22	1.00	3.090	200	40
18-19	1.00	3.090	200	40
18-23	1.00	3.090	200	40
21-22	1.00	3.090	200	40

Table A6: Demand matrix for the set Net50-1

Demand	Mean	Variance
1-6	33.76	84.79
1-8	43.33	81.27
1-18	35.31	78.79
1-27	30.78	94.63
2-7	53.4	64.62
2-20	30.66	99.7
3-28	57.56	98.65
6-19	46.14	97.94
6-29	38.33	97.53
6-50	25.01	63.29
7-24	34.05	66.64
7-48	25.16	70.12
8-15	31.07	64.37
11-29	55.49	78.3
14-24	44.75	62.75
15-45	29.46	76.13
21-23	33.2	50.19
21-32	59.78	68.04
23-33	41.97	97
24-45	43.95	96.78
28-36	58.66	98.55
31-39	48.21	86.04
34-36	28.57	93.13
37-44	27.03	68.29
44-50	23.61	66.73

Table A7: Net50-1 (50 nodes, 82 links)

$Link_j$	c_j	$\Phi^{-1}(\alpha_j)$	W_j	F_j
1-2	1.00	3.090	300	40
1-3	1.00	1.645	300	40
2-3	1.50	3.090	300	40
2-4	1.00	1.645	300	40
3-5	1.25	2.326	300	40
3-6	1.25	3.090	300	40
4-8	1.00	1.645	300	40
5-6	1.25	3.090	300	40
5-8	1.00	3.090	300	40
6-7	1.00	1.645	300	40
6-10	1.50	2.326	300	40
7-8	1.25	1.645	300	40
7-15	1.25	1.645	300	40
8-9	1.00	1.645	300	40
9-12	1.50	1.645	300	40
9-13	1.50	2.326	300	40
9-14	1.25	3.090	300	40
10-11	1.50	3.090	300	40
10-19	1.25	2.326	300	50
11-20	1.50	1.645	300	40
11-21	1.50	3.090	300	60
12-13	1.00	1.645	300	40
12-15	1.25	3.090	300	40
13-14	1.25	2.326	300	40
13-16	1.25	3.090	300	40
14-17	1.00	1.645	300	40
15-24	1.00	1.645	300	50
16-17	1.50	2.326	300	40
16-18	1.50	3.090	300	40
17-18	1.50	2.326	300	40
17-26	1.50	1.645	300	40
18-25	1.00	3.090	300	40
19-20	1.25	2.326	300	40
19-23	1.50	1.645	300	40
19-27	1.25	3.090	300	60
20-22	1.00	2.326	300	40
21-22	1.50	2.326	300	70
21-31	1.25	1.645	300	100
22-23	1.50	3.090	300	40

Table A8: Net50-1 (continued)

$Link_j$	c_j	$\Phi^{-1}(\alpha_j)$	W_j	F_j
22-30	1.25	2.326	300	40
23-28	1.25	3.090	300	60
23-29	1.00	3.090	300	40
24-25	1.25	3.090	300	40
24-27	1.25	1.645	300	40
25-26	1.25	3.090	300	50
25-38	1.50	2.326	300	40
26-38	1.25	1.645	300	40
26-47	1.25	3.090	300	40
27-28	1.00	2.326	300	40
28-40	1.50	1.645	300	40
29-30	1.25	1.645	300	40
29-32	1.50	1.645	300	40
30-31	1.00	3.090	300	60
31-33	1.25	3.090	300	40
31-34	1.50	3.090	300	40
32-33	1.00	1.645	300	40
32-41	1.50	2.326	300	40
34-35	1.25	3.090	300	40
34-36	1.50	3.090	300	60
35-36	1.50	3.090	300	40
35-37	1.00	2.326	300	40
36-37	1.00	1.645	300	40
36-44	1.25	1.645	300	70
38-39	1.00	2.326	300	40
39-40	1.00	2.326	300	50
39-49	1.00	1.645	300	40
40-41	1.00	3.090	300	40
40-49	1.00	1.645	300	50
41-42	1.25	1.645	300	40
42-43	1.00	1.645	300	40
43-44	1.00	3.090	300	60
43-49	1.50	1.645	300	40
43-50	1.25	1.645	300	40
44-45	1.25	3.090	300	60
44-48	1.25	1.645	300	60
45-46	1.25	1.645	300	40
45-48	1.00	1.645	300	40
46-47	1.25	1.645	300	40
47-48	1.50	2.326	300	40
48-49	1.50	2.326	300	40
48-50	1.50	2.326	300	40
49-50	1.00	2.326	300	40

Table A9: Net23: Input/Output summary

Experiment	Avg. demand volume (μ)	Avg. demand volume (σ^2)	Avg. BW allocation	Avg. link utilization	Max. link utilization	Avg. hop count
23-1	44.88	74.03	115.60	0.58	0.75	3.06
23-11	44.88	74.03	107.79	0.54	0.75	3.06
23-12	44.88	74.03	115.83	0.58	0.80	3.06
23-13	44.88	74.03	122.33	0.61	0.74	3.13
23-2	43.51	78.04	114.67	0.57	0.79	3.13
23-21	43.51	78.04	105.27	0.53	0.74	3.13
23-22	43.51	78.04	113.52	0.57	0.79	3.13
23-23	43.51	78.04	122.79	0.61	0.76	3.13
23-3	32.27	74.03	105.21	0.53	0.68	3.19
23-31	32.27	74.03	97.27	0.49	0.70	3.19
23-32	32.27	74.03	105.49	0.53	0.76	3.19
23-33	32.27	74.03	107.39	0.54	0.68	3.06
23-4	44.88	74.03	123.87	0.31	0.61	3.19
23-41	44.88	74.03	115.73	0.29	0.54	3.19
23-42	44.88	74.03	123.94	0.31	0.57	3.19
23-43	44.88	74.03	133.16	0.33	0.61	3.19
23-5	42.32	74.17	162.05	0.46	0.76	3.94
23-51	42.32	74.17	148.95	0.43	0.72	3.94
23-52	42.32	74.17	158.62	0.45	0.76	3.94
23-53	42.32	74.17	169.64	0.48	0.76	3.94
23-6	42.32	74.17	172.01	0.43	0.68	4.00
23-61	42.32	74.17	166.41	0.42	0.75	4.00
23-62	42.32	74.17	176.64	0.44	0.79	4.00
23-63	42.32	74.17	180.21	0.45	0.77	4.00
23-7	43.51	78.04	121.38	0.30	0.64	3.19
23-71	43.51	78.04	111.64	0.28	0.61	3.19
23-72	43.51	78.04	120.11	0.30	0.64	3.19
23-73	43.51	78.04	129.61	0.32	0.68	3.19
23-8	44.68	75.62	176.05	0.44	0.72	2.80
23-81	44.68	75.62	164.58	0.41	0.72	2.80
23-82	44.68	75.62	174.90	0.44	0.75	2.80
23-83	44.68	75.62	186.49	0.47	0.79	2.80
23-9	45.02	74.96	206.79	0.52	0.77	3.20
23-91	45.02	74.96	195.27	0.49	0.74	3.20
23-92	45.02	74.96	206.21	0.52	0.77	3.20
23-93	45.02	74.96	203.08	0.51	0.79	3.17
23-100	44.81	73.22	244.89	0.61	0.79	2.93
23-300	44.81	73.22	245.03	0.49	0.61	2.93
23-500	18.63	37.42	234.09	0.23	0.38	3.25
23-700	18.69	37.42	444.97	0.44	0.76	3.38

Table A10: Net23v2: Input/Output summary

Experiment	Avg. demand volume (μ)	Avg. demand volume (σ^2)	Avg. BW allocation	Avg. link utilization	Max. link utilization	Avg. hop count
23v2-1	44.88	74.03	133.64	0.67	0.78	3.69
23v2-11	44.88	74.03	124.07	0.62	0.78	3.63
23v2-12	44.88	74.03	128.68	0.64	0.73	3.69
23v2-13	44.88	74.03	138.39	0.69	0.79	3.69
23v2-3	32.27	74.03	111.90	0.56	0.76	3.44
23v2-31	32.27	74.03	113.48	0.57	0.71	3.50
23v2-32	32.27	74.03	122.55	0.61	0.77	3.50
23v2-33	32.27	74.03	119.01	0.60	0.79	3.50
23v2-4	44.88	74.03	152.56	0.38	0.69	3.56
23v2-41	44.88	74.03	142.68	0.36	0.66	3.56
23v2-42	44.88	74.03	151.97	0.38	0.69	3.56
23v2-43	44.88	74.03	162.39	0.41	0.73	3.56
23v2-8	44.68	75.62	215.99	0.54	0.77	3.50
23v2-81	44.68	75.62	206.93	0.52	0.75	3.53
23v2-82	44.68	75.62	218.32	0.55	0.79	3.53
23v2-83	44.68	75.62	225.71	0.56	0.75	3.57
23v2-9	43.01	77.06	116.76	0.29	0.55	2.38
23v2-91	43.01	77.06	107.99	0.27	0.50	2.38
23v2-92	43.01	77.06	116.48	0.29	0.53	2.38
23v2-93	43.01	77.06	126.03	0.32	0.57	2.38
23v2-100	44.81	73.22	254.82	0.64	0.87	3.16
23v2-300	44.81	73.22	256.95	0.51	0.82	3.09
23v2-500	18.75	36.45	252.20	0.25	0.44	3.16
23v2-700	18.69	37.42	449.17	0.45	0.82	3.37

Table A11: Net50: Input/Output summary

Experiment	Avg. demand volume (μ)	Avg. demand volume (σ^2)	Avg. BW allocation	Avg. link utilization	Max. link utilization	Avg. hop count
50-1	39.17	80.17	138.85	0.46	0.74	4.08
50-11	39.17	80.17	121.61	0.41	0.77	3.92
50-12	39.17	80.17	130.87	0.44	0.78	3.92
50-13	39.17	80.17	138.34	0.46	0.71	3.96
50-7	40.47	77.94	127.71	0.43	0.73	3.93
50-71	40.47	77.94	119.80	0.40	0.76	3.77
50-72	40.47	77.94	126.51	0.42	0.70	3.77
50-73	40.47	77.94	136.56	0.46	0.76	3.77
50-8	44.68	75.62	138.04	0.46	0.81	3.57
50-81	44.68	75.62	130.31	0.43	0.85	3.53
50-82	44.68	75.62	142.50	0.47	0.78	3.53
50-83	44.68	75.62	152.57	0.51	0.83	3.53
50-9	44.02	75.72	120.40	0.40	0.83	2.09
50-91	44.02	75.72	111.78	0.37	0.79	2.09
50-92	44.02	75.72	120.19	0.40	0.73	2.09
50-93	44.02	75.72	129.59	0.43	0.77	2.09
50-100	18.61	37.43	107.61	0.11	0.30	2.71
50-200	18.58	37.96	248.21	0.25	0.68	3.86
50-201	18.58	37.96	233.84	0.23	0.68	3.86
50-202	18.58	37.96	247.04	0.25	0.70	3.86
50-203	18.58	37.96	261.85	0.26	0.73	3.86

BIBLIOGRAPHY

[1] A. Banerjee, J. Drake, J. P. Lang, B. Turner, K. Kompella, and Y. Rekhter, "Generalized Multiprotocol Label Switching: An overview of routing and management enhancements," *IEEE Commun. Mag.*, vol. 39, no. 1, pp. 144–150, Jan. 2001.

[2] B. Davie and Y. Rekhter, *MPLS: Technology and Applications.* Morgan Kaufmann Publishers, 2000.

[3] D. Cavendish, "Evolution of optical transport technologies: from SONET/SDH to WDM," *IEEE Commun. Mag.*, vol. 38, no. 6, pp. 164–172, June 2000.

[4] N. Ghani, S. Dixit, and T.-S. Wang, "On IP-over-WDM integration," *IEEE Commun. Mag.*, vol. 38, no. 3, pp. 72–84, Mar. 2000.

[5] P. Bonenfant and A. Rodriguez-Moral, "Optical data networking," *IEEE Commun. Mag.*, vol. 38, no. 3, pp. 63–70, Mar. 2000.

[6] M. Pióro and D. Medhi, *Routing, Flow, and Capacity Design in Communication and Computer Networks.* Morgan Kaufmann Publishers, 2004, ch. 11, pp. 455–494.

[7] A. Charnes and W. W. Cooper, "Chance-constrained programming," *Management Science*, vol. 6, no. 1, pp. 73–79, Oct. 1959.

[8] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin, *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, 1993.

[9] G. Mohan and C. S. R. Murthy, "Lightpath restoration in WDM optical networks," *IEEE Network*, vol. 14, no. 6, pp. 24–32, Nov./Dec. 2000.

[10] R. Ramaswami and K. N. Sivarajan, *Optical Networks: A Practical Perspective*, 2nd ed. Morgan Kaufmann, 2002.

[11] P. Green, "Progress in optical networking," *IEEE Commun. Mag.*, vol. 39, no. 1, pp. 54–61, Jan. 2001.

[12] D. Awduche and Y. Rekhter, "Multiprotocol Lambda Switching: Combining MPLS traffic engineering control with optical crossconnects," *IEEE Commun. Mag.*, vol. 39, no. 3, pp. 111–116, Mar. 2001.

[13] E. Mannie, “Generalized Multi-Protocol Label Switching (GMPLS) architecture,” RFC 3945, Oct. 2004, work in progress. [Online]. Available: http://www.ietf.org/rfc/rfc3945.txt

[14] J. R. Birge and F. Louveaux, *Introduction to Stochastic Programming*, ser. Springer series in operations research. New York: Springer, 1997.

[15] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios, “Robust optimization of large-scale systems,” *Operations Research*, vol. 43, no. 2, pp. 264–281, Mar./Apr. 1995.

[16] D. Bai, T. Carpenter, and J. Mulvey, “Making a case for robust optimization models,” *Management Science*, vol. 43, no. 7, pp. 895–907, July 1997.

[17] S. Sen, R. D. Doverspike, and S. Cosares, “Network planning with random demand,” *Telecommunications Systems*, pp. 11–30, 1994.

[18] J. Kennington, K. Lewis, E. Olinick, A. Ortynski, and G. Spiride, “Robust solutions for the WDM routing and provisioning problem: Models and algorithms,” *Optical Networks Magazine*, pp. 74–84, Mar./Apr. 2003.

[19] D. Leung and W. D. Grover, “Restorable mesh network design under demand uncertainty: Toward ”future proof” transport investments,” in *Proc. Optical Fiber Communication Conference (OFC’04)*, 2004.

[20] ——, “Capacity planning of survivable mesh-based transport networks under demand uncertainty,” *Photonic Network Communications*, vol. 10, no. 2, pp. 123–140, Sept. 2005.

[21] D. Mitra and Q. Wang, “Stochastic traffic engineering for demand uncertainty and risk-aware network revenue management,” *IEEE/ACM Trans. Networking*, vol. 13, no. 2, pp. 221–233, Apr. 2005.

[22] E. A. Medova, “Chance-constrained stochastic programming for integrated services network management,” *Annals of Operations Research*, vol. 81, pp. 213–229, 1998.

[23] A. Charnes and W. W. Cooper, “Deterministic equivalents for optimizing and satisficing under chance constraints,” *Operations Research*, vol. 11, no. 1, pp. 18–39, Jan./Feb. 1963.

[24] B. Liu, *Uncertain Programming.* New York: John Wiley & Sons, 1999.

[25] J. Kilpi and I.Norros, “Testing the Gaussian approximation of aggregate traffic,” in *Proc. Internet Measurement Workshop*, 2002, pp. 49–61.

[26] R. G. Addie, M. Zukerman, and T. Neame, “Broadband traffic modeling: simple solutions to hard problems,” *IEEE Commun. Mag.*, vol. 36, no. 8, pp. 88–95, Aug. 1998.

[27] M. Zukerman, T. D. Neame, and R. G. Addie, "Internet traffic modeling and future technology implications," in *Proc. INFOCOM'03*, vol. 1, Mar. 2003, pp. 587–596.

[28] Y. Liu, "Spare capacity allocation: Model, analysis, and algorithm," Ph.D. dissertation, University of Pittsburgh, 2001.

[29] T. Telkamp, "Traffic characteristics and network planning." ISMA, Oct 2002.

[30] C. Charnsripinyo, "Design of survivable wireless access networks," Ph.D. dissertation, University of Pittsburgh, 2003.

[31] A. E. Smith and D. W. Coit, *Handbook of Evolutionary Computation.* Institute of Physics Publishing and Oxford University Press, 1997, ch. C5.2.

[32] A. Gençata and B. Mukherjee, "Virtual-topology adaptation for WDM mesh networks under dynamic traffic," *IEEE/ACM Trans. Networking*, vol. 11, no. 2, pp. 236–247, Apr. 2003.