© 2011 Xiaolan Zhang

RESOURCE MANAGEMENT FOR DYNAMIC OPTICAL NETWORKS

BY

XIAOLAN ZHANG

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Electrical and Computer Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Associate Professor Steven S. Lumetta, Chair
Professor Bruce Hajek
Associate Professor Sanjay J. Patel
Professor Klara Nahrstedt

ABSTRACT

The ability of core networks to manage data transmission of increasing volume and variation is critical for the success of data-intensive and network-centric applications as they grow in both scale and complexity. Traditionally, static optical networks were the dominant transport for medium and long distance communication. However, these networks can no longer meet the needs of tomorrow's applications for higher bandwidth at lower cost. New dynamic optical networks greatly improve the reconfigurability of optical terminal systems and support unprecedented flexibility for high-traffic resource sharing.

However, managing dynamic networks poses challenging problems related to scale and traffic volume. Traditional analytical techniques, which rely heavily on simplification of network topologies and route choices, are insufficient to understand the significant performance differences implied by the subtle path preferences of dynamic routing algorithms.

This dissertation presents an integrated approach to efficient and robust resource management algorithms for on-demand data traffic on dynamic photonic switching networks. First, a resource management framework is proposed to consider both resource dimensioning and routing for optical opaque networks. I develop dimensioning and routing algorithms that are efficient to implement and robust to evolutions of traffic load, network topology, and scale. Second, using Poisson traffic assumptions, the thesis develops an opportunity cost model for analyzing threshold-based dynamic routing algorithms. The model is scalable and provides better congestion management than previous work. Third, applying the dimensioning technique developed in this thesis, I have several important findings on managing dynamically routed optical translucent networks, optical restorable networks, and multiple optical network domains. Finally, the thesis solves a combinatorial optimization provisioning problem for a dynamic wavelength service traffic model on an optical translucent network. This work is the first to evaluate the robustness of optical route rearrangement. New solutions are proposed to design an optimal dynamic service network with rerouting capability.

To Kai-Shen, for his enduring love and support

ACKNOWLEDGMENTS

In the spring of 2009, I walked out of the south-side door of the Coordinated Science Lab, on my way to proctor an early morning final. In front of me, there was the Civil Engineering building. The east side of the building was a whole red-brick wall of five stories tall without a single window on it. Two trees stood about half as tall as the building, with numerous slim branches stretching into the air. The morning sun, peeking through the side of the Siebel Center, cast two giant and vigorous patterns of the trees on the entire wall. How magnificent! I couldn't help wondering how fortunate it was that my humble figure was able to walk on the very same campus with these most brilliant minds. The thesis would have never been accomplished without help from many, just as the sun rays to the trees on the wall.

My adviser, Steven Lumetta, provided extraordinary advising and guided me the way to outstanding research work. Steve is a great teacher and the role model for my academic career. From him, I learned how to conduct system research and communicate effectively in the technical community. I owe Steve for his vision and wisdom that helped me sail through many difficulties. I appreciate the freedom he granted me to explore my own interest in many avenues. It is his patience and confidence in me, along with all his time and efforts, that helped me succeed.

Bruce Hajek provided me many opportunities in teaching and research during my entire Ph.D. study. I want to particularly thank Bruce for his efforts in promoting my work. My thanks also go to Sanjay Patel, Klara Nahrstedt, and Tamer Başar. I will always remember their help on my career advancement. Additionally, I am very grateful for Lila Rhoades's administrative support for many years.

Sun-il Kim, my officemate and a long-term friend, collaborated with me for the first several years of my Ph.D. Sun-il helped me selflessly with simulations and measurements and contributed to many thinking processes prior to this thesis. I am grateful for the companionship of Elizabeth Van Ruitenbeek, Gabriel Jacques da Silva, and Jerry Chiang during my graduate schooling. There are many other friends whom I cannot name all here. My graduate life wouldn't be so memorable without their friendships.

I wish to thank the GRIPhoN project team at AT&T for a successful collaboration. In particular, my thanks go to Robert Doverspike and Angela Chiu for their participation that inspires a big portion of my thesis. I also want to thank James Giles from IBM T. J. Watson Research center, for providing me several industrial opportunities. Jim's mentoring greatly broadened the horizon of my knowledge beyond my thesis. Many other colleagues from these companies helped me greatly, through sharing their expertise and knowledge.

This thesis was supported by the grants from National Science Foundations, an IBM Ph.D. Fellowship, the Information Trust Institute of the University of Illinois at Urbana-Champaign, and the Hewlett-Packard Company through its Adaptive Enterprise Grid Program; equipment donations from Intel; and the CORONET project from Defense Advanced Research Projects Agency.

Finally, I thank my husband, Kai-Shen, for his devotion to me. It is his enduring love and support that kept me moving forward. I thank my parents for their love and generosity, supporting my pursuits and being ready to help all the time. My son, Francis, was born during the writing of this thesis. I am grateful for the joy he brought into my life. Finally, I want to remember my mother-in-law, for her kindness that stays forever in my heart.

Every achievement I made owes the contributions from many others. I take this opportunity to express my deepest gratitude to all my mentors, my friends, and my family. They make me grow stronger and reach higher.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ASPF	Adaptive Shortest Path First Routing
ATM	Asynchronous Transfer Mode
BAL	Balanced basic dimensioning algorithm
B&R	Bridge and Roll
CAR	Congestion Aware Routing
COL	Competitive Online Routing
CSR	Concatenated Shortest path Routing
D/MUX	De/Multiplexing
DBR	Design Based Routing
DPP	Dedicate Path Protection
DWDM	Dense Wavelength Division Multiplexing
E-NNI	External Network-Network Interface
E2E	End-to-End global SPF
ER	Edge Routers
FPLC	Fixed-Paths Least-Congestion
GMPLS	Generalized Multi-protocol Label Switching
GA	Genetic Algorithm
GAF	Genetic Algorithm with Fixed routes
GS	Global Shortest path dimensioning
IP	Internet Protocol

ILP	Integer Linear Programming
IS	Independent Shortest path dimensioning
LC	Liquid Crystal
LER	Label Edge Router
LLD	Least Loaded Routing
LWR	Least Resistant Weighted Routing
LSP	Label Switched Paths
LSR	Label Switched Router
MEAN	Incremental dimensioning algorithm using Mean scaling
MEM	Micro-Electronic-Mechanical
MIR	Minimal Interference Routing
MPLS	Multi-protocol Label Switching
NP	Non-deterministic Polynomial-time
NS	Normalized Shortest path dimensioning
nE2E	Normalized End-to-End global SPF
OOO	Optical-Optical-Optical
OEO	Optical-Electronic-Optical
OPT	Optimization
OT	Optical Transponder
OXC	Optical Cross-Connect
PBR	Profile-Based Routing
PCE	Path Computation Element
QoS	Quality of Service
RA	Single Routing Area
REGEN	Regenerator
RFR	Reduced Flow Routing
ROADM	Reconfigurable Optical Add/Drop Multiplexer

RWA	Route and Wavelength Assignment
RSVP-TE	Resource Reservation Protocol - Traffic Engineering
SA	Simulated Annealing algorithm
SAF	Simulated Annealing algorithm with Fixed routes
SDH	Synchronous Digital Hierarchy
SDR	State Dependent Routing
SD	Incremental dimensioning algorithm using Standard Deviation scaling
SLA	Service Level Agreement
SONET	Synchronous Optical Networking
SPF	Shortest Path Routing
SPG	Shortest Path Graph
SPP	Shared Path Protection
SWP	Shortest Widest Path Routing
TSL	Topological Shortest path Length
TSP	Topological Shortest Path
VPN	Virtual Private Network
WDM	Wavelength Division Multiplexing
WSP	Widest Shortest Path Routing
WSS	Wavelength Selective Switching

LIST OF SYMBOLS

N Set of nodes.

E Set of links.

C_e Current residual capacity of link $e \in E$. C is the set of all links' capacities.

V_e Current available capacity of link $e \in E$.

U_e Current used capacity of link $e \in E$. $U_e = C_e - V_e$.

B_e Basic dimensioning capacity for each link. B is the set of all links' capacities.

X_e Incremental dimensioning capacity for each link. X is the set of all links' capacities.

σ_e Statistic standard deviation of link capacity B_e. σ is the set of all links' standard deviations.

R Set of all end-to-end request pairs. $R \subseteq N \times N$.

λ Mean arrival rate per pair. λ_i is a random arrival rate for a pair.

μ Mean departure rate per pair. The average holding time, $1/\mu$. μ_i is a random departure rate for a pair.

T Traffic matrix.

TSL_r Topological shortest path length for a pair r.

TSP_r Topological shortest paths set for pair r.

SPG_r Topological shortest path graph for pair r. $SPG_r = \cup_{p \in TSP_r} p$.

$f_r(C)$ Maximum flow on SPG_r with link residual capacity C.

$MC_r(C)$ Set of links forming the minimum cut of a pair r with link residual capacity C.

u	Projected traffic load ratio. $u \in (0, 1]$.
l	Offered traffic load ratio. $l > 0$.
ϵ	Traffic evolution factor $\epsilon \in [0, 1]$.
α	Network misdimensioning factor $\alpha \in [0, 1]$.
ξ	Incremental dimensioning scheme in $\{MEAN, SD\}$.
β	Routing scheme in $\{SPF, WSP, LRW, CAR\}$.
$SPF_r(C)$	Set of all shortest available paths of a pair r based on current residual capacity C.
$P_i(C)$	Set of all available paths of a pair r based on current residual capacity C.
γ_p	Path congestion cost.
η	Congestion threshold.
$d_i^{c,l}$	Opportunity cost for a link with load l, total capacity c, and current capacity i.
N'	Set of border nodes in the local network. $N' \subset N$.
S'	Set of foreign border nodes. $S' \subset S, \lvert S' \rvert = \lvert N' \rvert$.
C'_e	Total capacity of link $e \in E$ provisioned for the external traffic.
C^I_b	Total capacity of inter-domain link connecting border nodes $b \in N', b' \in S'$ provisioned for the external traffic.
T'_L	Equivalent external traffic matrix on the local network.
$p_{n,s,b}$	The probability that border b is used to connect a local node $n \in N$ to a foreign node $s \in S$. $\sum_{b \in N'} p_{n,s,b} = 1$.
$\mathcal{V}$	Virtual link set.
O_n	Number of OT ports at node $n \in N$.

p Route. $p = \{V_p, N_p\}$, where V_p is the set of virtual links, and N_p is the set of regen nodes.

l Link (or virtual link) distance.

$\mathcal{W}$ Ordered wavelength set (1 to $|\mathcal{W}|$) in a DWDM fiber.

$W_{e,w}$ Binary ILP variable. $W_{e,w} = 1$ if wavelength $w \in \mathcal{W}$ is allocated on link $e \in \mathcal{E}$; $W_{e,w} = 0$, otherwise.

R_n Nonnegative integer ILP variable for the number of REGENs at node $n \in \mathcal{N}$.

CC Common cost per wavelength per mile.

RC Regenerator cost per device.

$C()$ Cost computation function.

$\mathcal{D}$ Set of demand matrices.

$\mathcal{M}$ Set of maximal demand matrices. $\mathcal{M} \subseteq \mathcal{D}$.

$\mathcal{S}$ Set of reduced demand matrices. $\mathcal{S} \subseteq \mathcal{M}$.

D_s The sth demand matrix. $D_s \in \mathcal{D}$.

$d_{s,t}$ The tth demand of the sth demand matrix. $d_{s,t} \in D_s$.

$N(d)$ The node pair (i, j) for demand d. In a context without confusion, $N(d)$ can represent a unique demand d.

$\mathcal{P}_{(i,j)}$ Set of routes for the node pair (i, j).

$X_{p,d}$ Binary ILP variable. $X_{p,d} = 1$ if route $p \in \mathcal{P}_{N(d)}$ is selected for demand d; $X_{p,d} = 0$, otherwise.

A Route allocation for all demands (in demand-route tuples) without wavelength assignment. $|A|$ is the number of all demands. $A[x : y]$ represents the list of xth to yth demand-route tuples.

$A(d)$ Route allocation for demand d in allocation A.

$\mathcal{A}$ Set of all possible route allocations without wavelength assignment.

A^w Wavelength assignment for route allocation $A \in \mathcal{A}$. A set of solved $X_{p,d}$s.

CHAPTER 1

INTRODUCTION

In recent years, optical networks have evolved toward higher bandwidth, integrated routing planes, and flexible reconfiguration. At the same time, the request for on-demand and high-bandwidth connection services emerged from multiple application domains, such as cloud computing, video content hosting, and enterprise private networks. These applications drive the need for efficient management techniques for dynamic wavelength routed optical networks.

This chapter provides background on the technological advances, a summary of new challenges, and the contributions of this thesis.

1.1 Background

Optical networks have evolved over the years to provide higher data-rate services in a cost-effective manner [1, 2]. Dense Wavelength Division Multiplexing (DWDM), the major technology behind today's backbone networks, has improved over the years to support higher data transmission rates per wavelength, larger numbers of wavelength channels multiplexed on a single fiber, and longer photonic signal transmission distances. These efforts greatly reduced the unit bandwidth cost and made high-bandwidth connections more affordable to a large number of customers.

Higher layer protocols, especially the IP layer, have become more closely integrated with the photonic layer. Traditionally, there were multiple digital layers in between for routing and grooming IP traffic on the optical transport. Synchronous Optical Networking (SONET) and Synchronous Digital Hierarchy (SDH) are typical electronics-optical networks that transport Asynchronous Transfer Mode (ATM) traffic on optical networks. For a long time, ATM networks were the primary method of carrying voice data and

IP traffic. Today, however, the volume of IP traffic has outgrown that of other traffic by several orders of magnitude due to the growing popularity of Internet applications. IP is now the primary technology to support with the photonic layer.

Simplifying network protocol stacks for supporting IP traffic cheaply has become the technology priority for many network providers. Generalized Multi-protocol Label Switching (GMPLS) [3] was developed to provide signaling capability to directly reserve wavelength paths on WDM networks or SONET-like networks. Border Gateway Protocol (BGP) can be built upon GMPLS-over-WDM to carry regional Internet traffic. Further, a carrier-grade Ethernet standard is under development so that WDM networks, together with other transport technologies, can directly transport Ethernet frames. With integrated control planes, the photonic layer can be more seamlessly connected to the user layer, and traffic at higher layers can have a more direct influence on the photonic layer.

Route configuration on photonic networks has become more flexible. In the past, route provisioning at the optical layer required intensive manual work; therefore, backbone networks were managed statically and did not allow frequent connection changes. Today, reconfigurable photonic devices—such as Reconfigurable Optical Add-Drop Multiplexers (ROADM), Micro-ElectroMechanical (MEM) photonic cross-connects, and tunable lasers—are widely used in core photonic networks [4, 5], greatly increasing the dynamic capability and transparency of the optical network. The GRIPhoN research testbed being developed at AT&T Labs can configure an optical route automatically in a few minutes and reroute a photonic path using a bridge-and-roll technique [6].

Recently, practical interest in providing on-demand wavelength connection services for future IP customers has grown [7, 8]. The emergence of relatively short-duration applications that require high data rates for video delivery, health care applications, etc. [9], has increased the dynamic variability of traffic. However, building and upgrading optical networks in response to frequent short-term traffic changes (i.e., hours and minutes) is expensive and impractical. For long-haul optical networks, expensive construction is needed to lay new fibers. Adding more active wavelength channels includes hardware upgrade costs for transponders and switches and management costs for tuning and testing the system. The entire duration may take weeks or even months, which makes it difficult to change as frequently as traffic varies. Therefore, the network provider must have sufficient equipment on hand to route dynamic traffic.

Although the majority of current long-haul optical networks remain statically provisioned, general discussions of dynamic WDM channel provisioning dates back ten years [10]. The study in [11] showed that a dynamic network (called Intelligent Optical Network (ION) by the authors) could potentially save up to 80% capacity relative to the current static optical network (referred to as Optical Transport Network (OTN)) for dynamic demands. The practical interests of providing on-demand wavelength connection services for future IP customers have revived research into online, link-state-based routing and load balancing on optical networks to supplement statically optimized paths. Correspondingly, the underlying physical networks should be provisioned in response to the current and expected growth of dynamic traffic demands while simultaneously allowing robust adaptation to load changes and different network topologies.

1.2 Network Resource Management

Resource management for dynamic systems addresses three issues: traffic load characterization, network dimensioning, and online routing. *Traffic load characterization* involves finding a model to describe the amount of traffic in relation to network resources. *Network resource dimensioning* involves the distribution of equipment at initial setup or the upgrading of hardware on rare occasions. *Online routing* selects and allocates a route when a connection request arrives. As the frequency of demand variations (which change on an hourly or minute-based timescale) outpace the hardware upgrade speeds (which change on a days or weeks timescale) for new installations, it becomes less likely that a system can stay at the optimal point between equipment upgrades. Therefore, both the management of long-term load projection using resource dimensioning and short-term load fluctuation using online routing must occur. The goal is to improve performance and robustness of networks under load variations to reduce the need for resource reallocation and rebalancing.

Dimensioning for online, state-based routing schemes is challenging. Previous studies on traditional circuit-switching networks tried to solve routing and dimensioning problems together by planning an optimal path (or a few alternative paths) for each demand. Many models were proposed to approximate the steady state blocking probability in such networks [12, 13, 14, 15]. Analytical approaches for fixed-path routing [16] can be useful for traditional telephony networks where traffic has a strong correlation to geographic factors; however, these are not appropriate for data networks [17]. New fixed-point approximation techniques for routing online traffic on optical networks is difficult to scale to large mesh networks. To reduce the network state, a preselected path set is still used to approximate state-based online routing, and some routing preferences are ignored [18, 19, 20]. In fact, these approximations do not sufficiently capture the path selection preferences of online routing algorithms, which can make a substantial difference in performance results [20, 21, 22].

1.3 Thesis Contributions

This dissertation develops efficient network resource provisioning solutions (including resource dimensioning and online routing algorithms) through a combination of analysis and simulation. I built event-driven network simulators to study the quantitative relationship between network resource dimensioning and online routing. Traffic load characterization, dimensioning, and routing were considered jointly in order to achieve an optimal solution.

The thesis first presents efficient resource dimensioning algorithms on optically opaque networks for stochastic dynamic traffic. I then propose an analytical model of the opportunity cost for accepting a connection with a dynamic routing algorithm. Based on the model, I develop novel routing algorithms with efficient implementation, and the algorithms are robust to network variables, such as topology and capacity. The resource dimensioning technique can be applied to translucent networks, restorable opaque networks, and multiple opaque network domains.

Further, the thesis pioneers the study of resource provisioning problems for a practical dynamic traffic model with port constraints (called *dynamic wavelength services* by AT&T). I develop several heuristic optimizers to find an optimal network resource provisioning. On problems of current practical interest, my methods achieve an average of 8% overhead above lower bounds of optimal values. I also introduce theoretical analysis of the traffic model and leverage properties that can significantly reduce optimization time. This work is the first to evaluate the benefit of using an optical bridge-and-roll technique (a route changing technique for minimizing signal loss that sets up the new route before removing the old route) for dynamic wavelength services. Despite the benefits, the bridge-and-roll operation requires that the resources for the new route must be available before the old route is torn down. The analysis shows that optimal provisioning cannot always satisfy the bridge-and-roll condition with optimized resources. I derive provable bridge-and-roll safe provisioning technique and measure negligible overhead for ensuring bridge-and-roll rerouting on real-scale network simulations.

The remaining chapters are organized as follows. Chapter 2 provides background information on dynamically routed optical opaque and translucent networks, cost models, Poisson traffic models, dynamic wavelength services models, typical routing algorithms, network topologies, assumptions, and simulation methodologies. Chapter 3 introduces load metrics and dimensioning algorithms for Poisson-loaded opaque networks. In Chapter 4, I propose a novel flow-based online routing algorithm, an opportunity cost model for optimizing threshold-based congestion control routings, and an oracular optimal routing. Chapter 5 describes the performance of the dimensioning algorithms presented in Chapter 3 on translucent networks and evaluates the resource distribution for four different wavelength operating modes. Chapter 6 presents a redimensioning technique for a single-link failure restorable network, and Chapter 7 develops dimensioning algorithms for two separately managed optical networks and discusses fairness issues. I suggest efficient techniques to tackle a challenging resource optimization problem for dynamic wavelength services and analyze the applicability of optical rerouting techniques Chapter 8. Chapter 9 explores related work, while Chapter 10 concludes the thesis.

CHAPTER 2

OPTICAL NETWORK MODELS AND MANAGEMENT

This chapter discusses foundational concepts, including network models, cost models, traffic models, and route management technologies.

2.1 Translucent Networks

In optical network systems, the term “transparent” refers to photonic circuit switching without per-packet electronic processing; all optical switching is also called optical-optical-optical (OOO). The term “opaque” refers to switching that involves optical electronic conversion and regeneration in which optical signals are converted, corrected, and retransmitted. Opaque switching, also called optical-electronic-optical (OEO) switching, was used at all nodes in early optical networks when all-optical switching devices were too expensive and the insertion loss was too high, making long photonic transmission without electronic regeneration impossible. With substantial improvements in photonic devices, today’s transmission systems extend the distance and switching hops between a pair of optical transmitters and receivers.

Given these changes, an emerging model of “translucent” networks has become popular. In this model, optical signals are circuit-switched photonically at some nodes and electronically regenerated at other nodes, as necessary. Reconfigurable Optical Add-Drop Multiplexer (ROADM) networks are a type of translucent network widely used in practice. Dense Wavelength Division Multiplexing (DWDM) technology, harnessing the reconfigurable power of tunable lasers and photonic cross-connects, has emerged as a key component in future dynamic ROADM infrastructures. This section describes the network and cost model of a fully reconfigurable ROADM network architecture. A detailed description of the physical characteristics of the transmission system can be found in [5, 23, 24, 25, 4].

2.1.1 Reconfigurable Optical Add-Drop Multiplexer Networks

In a wavelength division multiplexing system, data packets are modulated into analog optical signals and multiplexed onto a specific wavelength frequency of a DWDM signal. The signal is transmitted through a long-reach fiber system that connects remote switching nodes. DWDM technology remarkably improves the efficiency of optical systems by enabling a fiber link to carry 40 to 80 different wavelength channels at the same time. An optically reconfigurable network consists of ROADM as nodes and DWDM fibers as links. A typical backbone ROADM network is a mesh topology that interconnects nodes at city's level. Each city may have one (or more, depending on the city population) switching node at the carrier's central office. Each fiber link transmits high-power optical signals for hundreds of miles without electronic signal regeneration.

Figure 2.1 shows the architecture of a ROADM node on the GRIPhoN testbed that is being developed at the AT&T Labs Research. All optical links are bidirectional such that each port of the shown devices connects to a pair of fibers, one for each direction. The ROADM node is fully reconfigurable: any wavelength from one DWDM input can be switched to other available DWDM ports or add/drop ports. The ROADM has colorless and steerable add/drop ports, enabling OTs and REGENs to connect to any available add/drop ROADM port and any available wavelength color.

GRIPhoN ROADM nodes achieve reconfigurability using Wavelength Selective Switches (WSS). A WSS can deflect a wavelength channel from one port to the other. Some devices use an array of electronically configurable MicroElectroMechanical (MEM) mirrors as switching components. Newer devices use Liquid Crystal (LC)-based channel blockers that can be controlled to select pass-through wavelength channels. The wavelength channels that need to drop locally are demultiplexed into individual wavelength signals that are carried by one fiber per wavelength channel. Typically, these are low-power optical connections that transmit short-reach signals within tens of meters.

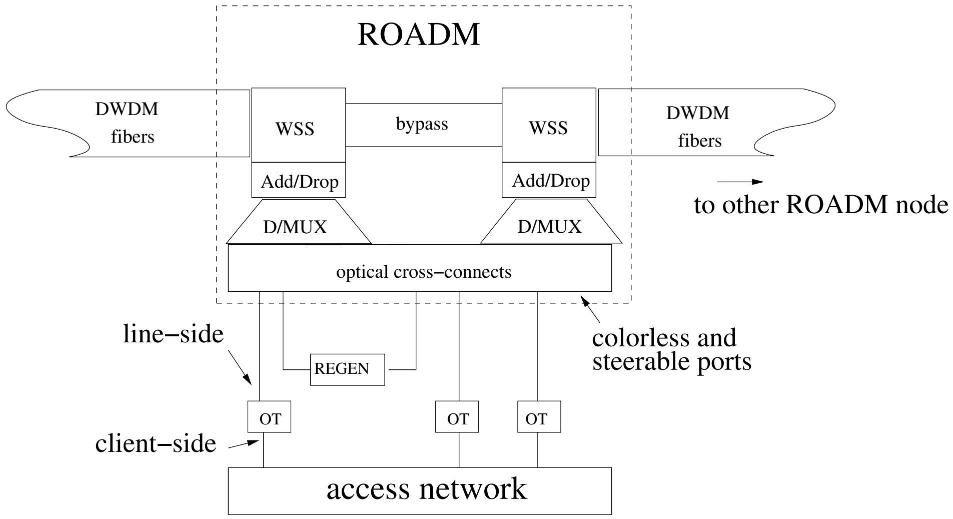


Figure 2.1: A ROADM node architecture.

An optical transponder (OT) has a local interface that connects to the optical ports of access routers and a line-side interface (high-power laser port) that connects the ROADM network. A local signal is electronically transcoded at an OT and sent through its laser for long distance transmission. Today's OTs are wavelength tunable, i.e., an OT can receive from any wavelength channel of the demultiplexed DWDM signal and send to another wavelength channel. A 3R regenerator (REGEN) receives the data from a wavelength channel and resends the same data to another wavelength. Physically, a 3R REGEN is equivalent to a pair of optical OTs whose customer-side ports are short-connected. However, a REGEN lacks customer-side ports, so it cannot be used as an OT.

Traditionally, ROADM nodes are non-steerable or partially steerable because OT/REGENs can connect to only one wavelength channel of one add/drop ports. GRIPhoN ROADM nodes achieve full steerability by using an optical cross-connect. Therefore, the OTs and REGENs can connect to any of the wavelength channels of the D/MUX add/drop ports of any direction. This design provides the opportunity to share OT/REGENs throughout the entire node.

The access network interfaces the ROADM network to high layer protocols, such as SONET/SDH, ATM, IP or carrier-grade Ethernet. In practice, the access network consists of routers co-located with the ROADM node. The routers connect fiber ports that transmit one single-wavelength optical signal per fiber for tens of meters. The access network provides an electronic layer built on top of the photonic layer.

2.1.2 Route Provisioning

A wavelength connection is set up between a pair of OTs installed at source and destination ROADM nodes. The route must be a simple path; no node is visited twice. Figure 2.2 shows an example route. Intermediate ROADM nodes, which route the connection through optical bypass cross-connects, are called “bypass” nodes. If the connection distance exceeds the maximal optical reachability for the network system, some intermediate nodes serve as “regen” nodes and cross-connect the connection through a 3R REGEN. In order to regenerate a signal, the connection is received by a REGEN through a drop port and resent by the same REGEN through an add port. The route segment between a pair of OTs or REGENs is called an “optically transparent segment.” Except for the end nodes, all intermediate nodes in an optically transparent segment are bypass nodes. Therefore, wavelength continuity is enforced within a segment. The wavelength can change from segment to segment using the REGEN.

ROADM networks are managed centrally and completely circuit switched. ROADMs, optical cross-connects, OTs (both line-side and customer-side), and REGENs are all remotely configurable. On the GRIPhoN testbed, wavelength connections can be set up and torn down in a few minutes, which makes it possible to provide a dynamic wavelength service. If a customer owns a few OTs at different ROADM nodes, the network allows the customer to reconfigure the optical network topology freely by creating and removing connections on the reconfigurable ROADM network.

2.1.3 Reachability and Virtual Link Graph

Optical *reachability* is the maximum distance of an optically transparent segment allowed for a system. For 10Gbps or 40Gbps systems, reachability is typically 1500 km (932 mi). A *virtual-link graph* simplifies the routing of optically transparent segments with reachability constraints. The original virtual-link idea was explained as express links in [26]. To create a virtual

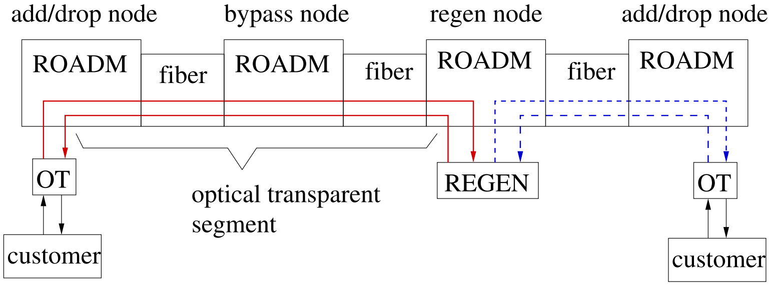


Figure 2.2: Bidirectional wavelength connection. The connection includes two optically transparent segments. The first segment routes on a red wavelength. The second segment routes on a blue wavelength.

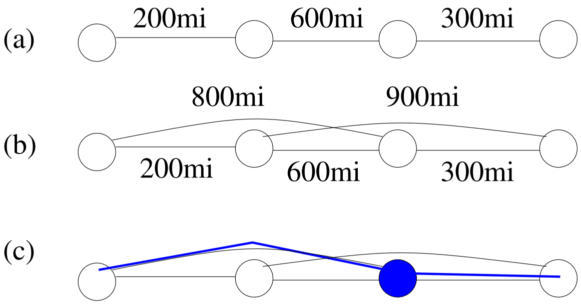
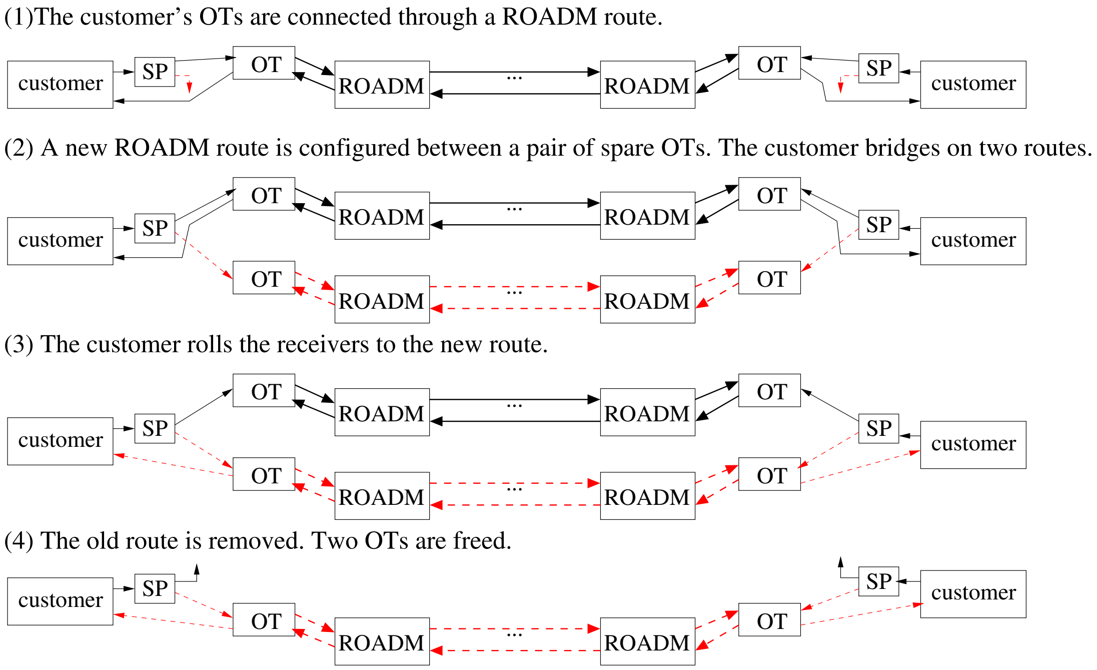


Figure 2.3: (a) Physical graph with link marked with fiber distance (in miles). (b) The corresponding virtual-link graph for a 40Gbps system (reachability of 932 mi). Virtual links are also marked with fiber distance. (c) Example customer route on the virtual-link graph. The route passes two transparent segments. The dashed wavelength is marked for use by all links in the graph. A REGEN at the solid node connects two segments. The cost of the path is $150 + 1100\frac{70}{1000} = 227$.

link graph, a virtual link is created for every reachable optical transparent segment (including each individual physical link). All virtual links that pass the same physical link share the wavelength resources of the physical link. A route can be created on the virtual-link graph. The physical route that maps to the virtual-link route must be a simple path. A REGEN must be used at each node that connects two virtual links. Figure 2.3 shows an example.

2.1.4 Photonic Bridge-and-Roll Rerouting

Optical bridge-and-roll (B&R) rerouting is supported by the GRIPhoN testbed. Figure 2.4 illustrates the B&R process. A new route is set up between two OTs for a customer while the old route is still connected. A 50/50 op-


(1)The customer's OTs are connected through a ROADM route.

(2) A new ROADM route is configured between a pair of spare OTs. The customer bridges on two routes.

(3) The customer rolls the receivers to the new route.

(4) The old route is removed. Two OTs are freed.

Figure 2.4: Illustration of optical B&R operation steps. A 50/50 optical passive splitter (SP) is preinstalled at the customer's transmission side.

tical passive splitter is preinstalled at the customer's transmission port. The customer then has the opportunity to "bridge" two OT routes at any time without interrupting the transmission. Once the customer bridges the old and new routes, the receivers are quickly "rolled" to the OTs of the new route. Finally, the old route is taken down. In order for B&R to work, there must be one spare OT available at each node, and the new route must be resource disjoint from the old route. The GRIPhoN testbed demonstrated that the Ethernet packet loss period is about 8 milliseconds in a photonic layer B&R experiment [6].

2.1.5 Cost Model

The capital expense of a ROADM network consists of the cost of OTs, REGENs, ROADM ports, and DWDM fibers. Since the cost of tunable OTs/REGENs is relatively high and increases linearly with the load of network connections, they are counted as a per-equipment cost. The rest of the equipment, shared by all wavelengths, has to be preinstalled. Therefore, it is modeled as a common cost and prorated in wavelength-miles. The current normalized cost (based on vendor's price) for 40Gbps systems is 100 for an OT, 150 for a REGEN, and 0.07 per wavelength per mile. The cost of the sample route in Figure 2.3(c) is 227. The REGEN cost is nearly double the common cost for this route.

2.2 Opaque Networks

Typically, IP networks can be built directly upon WDM networks. Any link between two IP routers is, in fact, one or more wavelength routes that may have multiple OOO or OEO hops, where no electronic processing occur at the IP layer. IP-WDM network architectures have become increasingly popular as IP traffic (VPN channels) has dominated the usage of backbone networks. The traditional opaque network model can apply to these VPN networks, as optical signals in fact are electronically processed and circuit switched at every VPN node. Wavelength continuity does not impose on the network as a routing constraint.

2.2.1 Generalized Multi-Protocol Label Switching (GMPLS)

A GMPLS network [3] is an opaque network. GMPLS builds an electronic network interface on the lower optical network of the IP layer. Each link on the GMPLS network is an optical lightpath that consists of one or more hops. Its control platform extends traditional MPLS [27] to support routing through various physical infrastructures, including synchronous optical networks (SONET/SDH) and photonic networks (DWDM), aiming at integrating multiple network carriers into a single control structure. As an extension of MPLS, GMPLS also provides signaling for explicit routing for connection-oriented optical networks. The resource reservation protocol - traffic engineering (RSVP-TE) [28] protocol provides resource reservation or non-reservation routing through label switched routers (LSRs), including establishing label switched paths (LSPs), preemption, and loop detection functions. According to [7], GMPLS networks employ dynamic routing and signaling mechanisms so that LSPs can be established and terminated on demand, providing a platform necessary for dynamic routing.

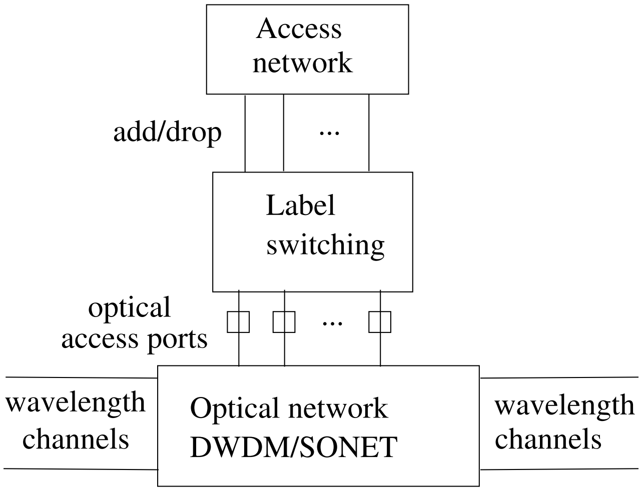


Figure 2.5: A label switching node structure.

Figures 2.5 depicts the node architecture of a GMPLS network. The label switching router (LSR) has optical-electronic interfaces to the underlying optical networks, such as DWDM or SONET. Local access (often implemented as a label edge router (LER)) is an interface to an IP network. LER generates end-to-end routing requests, and LSR forwards traffic. Practically, LER and LSR together can be conceptualized as an electronic switch that integrates multiple network layers and provides routing functionality at the IP layer. However, in some networks, some nodes have either LER or LSR. The nodes without an LER do not generate new traffic. Currently, the industry is also developing standards for carrier-grade Ethernet that allow the optical layer to carry native Ethernet frames. For a GMPLS on a ROADM design, the label switching module accesses the ROADM network through OTs. The label switch processes at the packet level and forwards packets according to the labels. This thesis considers only the type of networks that switch at the data rate of one wavelength channel. It does not discuss electronic traffic grooming or sub-wavelength routing.

2.2.2 Cost Model

This thesis uses a per-wavelength cost model for opaque networks. One unit cost is assigned to each wavelength channel provisioned between any pair of nodes. For wavelength routing, the cost of a route simply counts the number of hops. Such simplification is reasonable because the per-wavelength photonic equipment comprises the majority of the network cost. The percentage continues to grow as the data rate increases. A projection of the capital expense of future 40 Gbps network systems showed that 91% of the cost is at the terminal devices (switching ports, routers, transponders, regenerator cards) [24]. Amongst terminal devices, electronic routing/grooming takes 40%. If we subtract the electronic part, OT and REGEN constitute 83% of the photonic cost.

2.2.3 Online Routing Algorithms

The dynamic routing problem has been extensively studied for circuit-switched ATM networks. Dynamic routing in a mesh network involves selecting an available path on the arrival of an end-to-end connection request, based on the current network capacity. A connection request is initiated by a source-destination node pair. Without loss of generality, we consider only bidirectional connections. For abbreviation, "node pair" or "request pair" refers to a pair of nodes that initiates a connection request.

Adaptive shortest path first (ASPF) routing is a simple solution that selects an available path with the fewest hops. The connection request is dropped if no SPF paths can be found. Many other heuristic solutions have been proposed to improve connection acceptance rates using an estimate of the traffic load. This section introduces a few such online routing algorithms (Algorithm 2.1).

Some algorithms select routes exclusively from the available shortest paths on the residual graph. Widest shortest path first (WSP) routing picks the path of the largest residual capacity at the bottleneck link. The WSP+1, a one-hop relaxed WSP variant, chooses the widest path among all paths within shortest hops plus one. Further ties are broken arbitrarily since the choice would have little impact on the capacity distribution of future residual networks.

Some algorithms select routes from available paths of any length. The SWP algorithm chooses the shortest hop path from all paths with the widest residual bottleneck capacity. Least resistant weighted (LRW) routing [29] chooses the minimum cost path as defined by the sum of the ratios of total capacity to available capacity on each link. Minimal interference routing

Algorithm 2.1: Routing algorithms

```
begin WSP
    foreach $p \in SPF_{s,d}(C)$ do
        $\gamma_p \leftarrow \min_{e \in p} C_e$;
    Pick the widest $path \leftarrow \arg\max_p \gamma_p$;
begin SWP
    foreach $p \in P_{s,d}(C)$ do
        $\gamma_p \leftarrow \min_{e \in p} C_e$;
    Widest path set $Q \cup \arg\max_p \gamma_p$;
    Pick the shortest $path \leftarrow \arg\min_{p \in Q} |p|$;
begin LRW
    foreach $p \in P_{s,d}(C)$ do
        $\gamma_p \leftarrow \sum_{e \in p} \frac{(B_e + X_e)}{C_e}$;
    $path \leftarrow \arg\min_p \gamma_p$;
begin MIR
    foreach  $r \in R$ do
        Compute min-cut link set $MC_r(C)$;
    foreach $p \in P_{s,d}(C)$ do
        $\gamma_p \leftarrow \sum_{e \in p} \sum_{r \neq (s,d): e \in MC_r(C)} \frac{\lambda}{\mu}$;
    $path \leftarrow \arg\min_p \gamma_p$;
begin SDR
    foreach $p \in P_{s,d}(C)$ do
        $\gamma_p \leftarrow \sum_{e \in p} \frac{ErB(C_e, l_e)}{ErB(U_e, l_e)}$;
    $q \leftarrow \arg\min_p \gamma_p$;
    if $\gamma_q > \eta$ then
        Reject the connection;
    else
        Accept the connection on route $q$;
begin COL
    foreach $p \in P_{s,d}(C)$ do
        $\gamma_p \leftarrow \sum_{e \in p} \upsilon^{-\frac{V_e}{C_e}}$;
    $q \leftarrow \arg\min_p \gamma_p$;
    if $\gamma_q > \eta$ then
        Reject the connection;
    else
        Accept the connection on route $q$;
```

(MIR) [30] uses the flow of information to compute network weights for routing. The computation of complete MIR routes is NP-hard. Therefore, an online version of MIR was proposed to approximate the optimal solution while reducing computation complexity. MIR first computes the min-cut link set for the flows of all other node pairs excluding the current one. Then, MIR computes the link cost by summing the min-cuts that involve this link of all other pairs. The link cost is further weighted by the predetermined "importance" of a pair. I use the expected Poisson load for the parameter. Finally, MIR chooses the path with the minimum total weight. The computational time for each request is three orders of magnitude higher than for non-flow-based algorithms.

There are also threshold-based routing algorithms. State-dependent routing (SDR) [31] attempted to compute the best route using a Markov decision process. As the model quickly becomes intractable, a link cost model is proposed to approximate the analytical solution. The cost of each link is computed by $\frac{ErB(C_e,l_e)}{ErB(U_e,l_e)}$, where $ErB(C_e, l_e)$ is the Erlang-B blocking formula, and l_e is the link load estimation. Link load is estimated using a SDRAdapt algorithm [32]. Basically, SDRAdapt periodically measures residual capacity on each link and updates estimated link load in a sliding window fashion. SDRAdapt requires the selection of two parameters: a scan interval δ and a sample number Δ; the authors picked $\delta = 0.5$ and $\Delta = 30$. However, the result is sensitive to these parameters, and better results were obtained using $\delta = 0.05$ and $\Delta = 50$ (Chapter 4). The difference is due to the lack of normalization of timescales between papers.

Gawlick *et al.* [33] proposed a throughput-competitive online routing and admission control algorithm (COL). The original version of the algorithm required knowledge of the actual holding time of each connection. The authors therefore proposed a practical version in which each link cost is computed by $\upsilon^{-\frac{V_e}{C_e}}$. However, the determination of υ is nontrivial, and the result is sensitive to υ [32]. The authors did not provide any effective way to choose υ.

2.3 Typical Network Topologies

As carrier-grade networks often contain proprietary information and are kept away from public access, this thesis uses open topologies (mostly from research projects) that provide characteristics similar to practical backbone networks. Figure 2.6 shows the backbone topologies for simulating the performance of optical opaque network models. NJ LATA provides a good case study for performance evaluation on small networks. The other networks all serve regional telecommunication needs. COST 239 interconnects major cities in Europe. NSFNET was built to connect academic supercomputing centers across the U.S for research purposes.

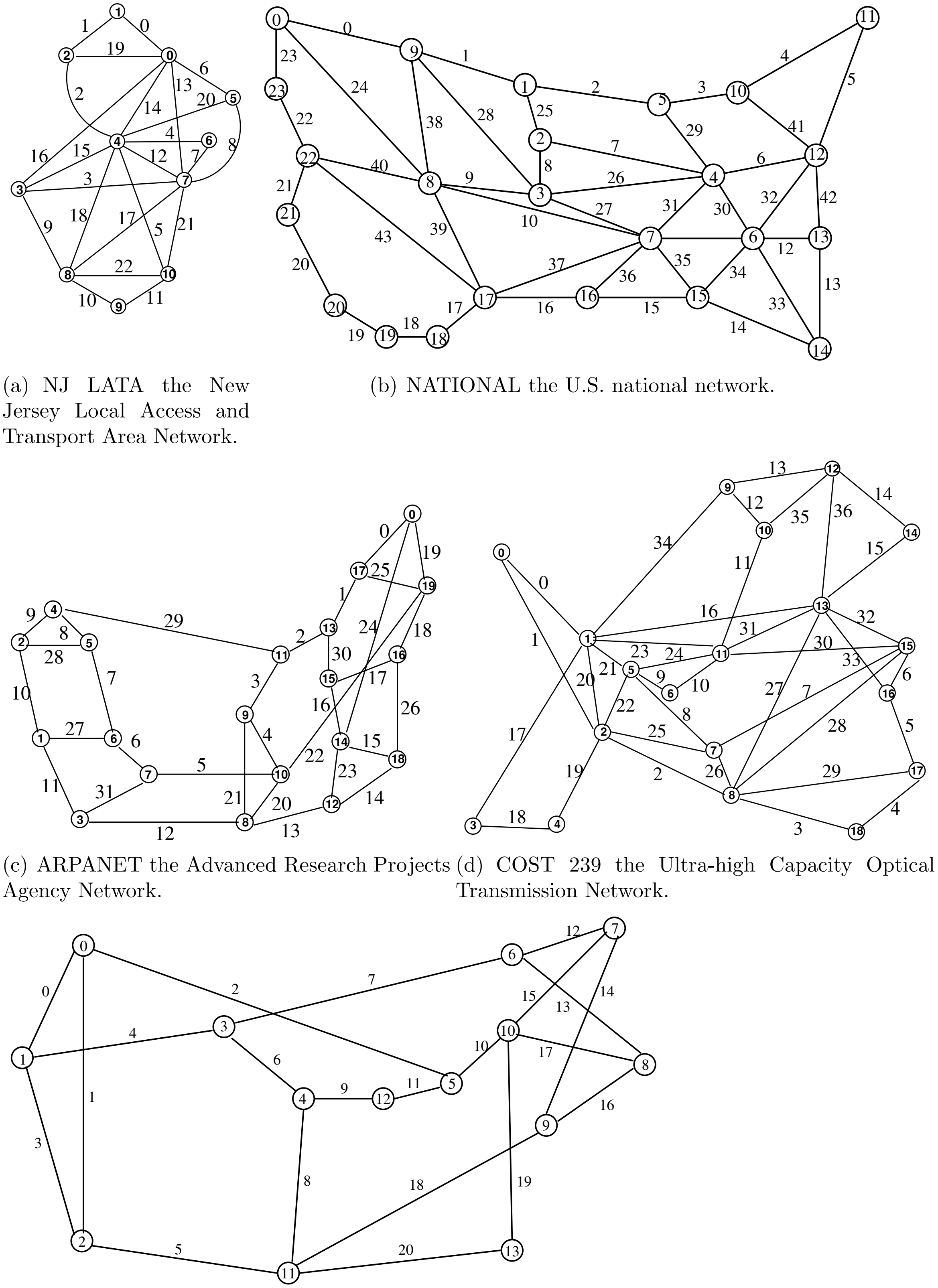
Figures 2.8 and 2.7 show the topology of U.S. CORONET, chosen as the backbone topology for simulating translucent networks. CORONET was created by a Telcordia-AT&T team to mimic a typical large international core network [34]. We use the U.S. contiguous part of CORONET, which consists of 75 nodes and 99 links. Table 2.1 shows the mapping of CORONET nodes to U.S. cities. This thesis targets a 40Gbps ROADM system that has the maximal optical reach of 932 miles. I proportionally scaled down the link distance by 75% of the original CORONET to fit the single hop reach of today's 40Gbps system. Table 2.2 lists the new link distances in miles that are used in this thesis.

2.4 Traffic Models

2.4.1 Poisson Traffic

This thesis uses the Poisson model to generate connection request arrivals, but my algorithms do not depend upon this choice. Rather, the Poisson model is chosen because the burstiness of traffic on backbone networks is usually suppressed by huge amounts of aggregation of higher layer services [35]. The real traffic distribution remains unknown as few practical dynamic optical backbones exist today, and traffic statistics from carriers are often proprietary information.

Unless otherwise specified, all node pairs in a network generate traffic arrivals. Each node pair initiates a connection arrival following a Poisson



(a) NJ LATA the New Jersey Local Access and Transport Area Network.

(b) NATIONAL the U.S. national network.

(c) ARPANET the Advanced Research Projects Agency Network.

(d) COST 239 the Ultra-high Capacity Optical Transmission Network.

(e) NSFNET the National Science Foundation Network

Figure 2.6: Optical opaque network topologies with link and node numbers.

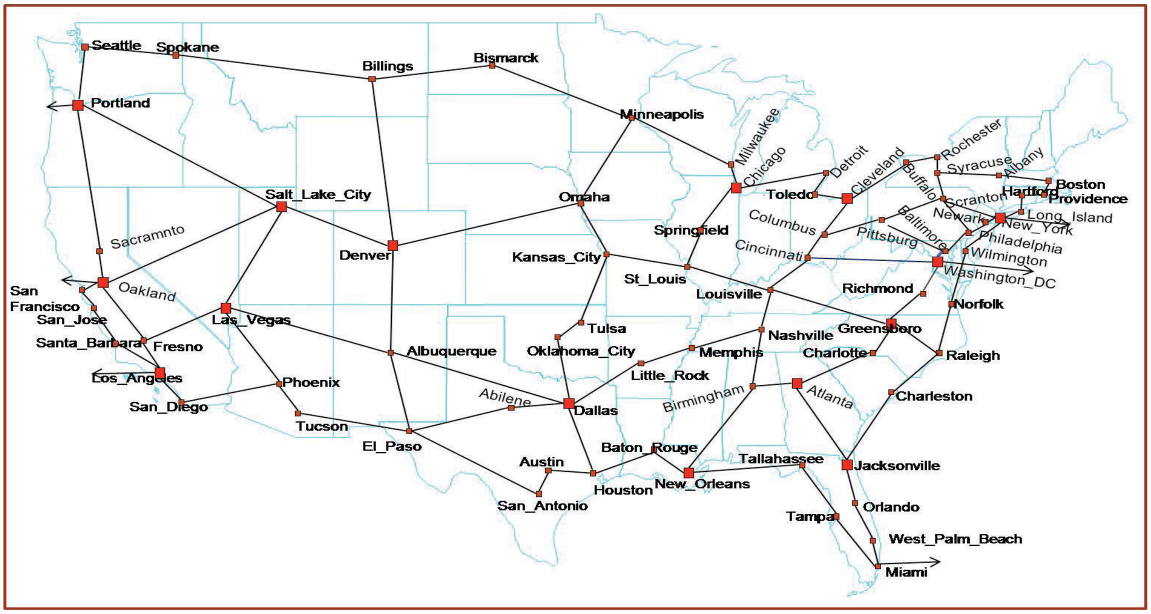


Figure 2.7: U.S. CORONET on the U.S. map.

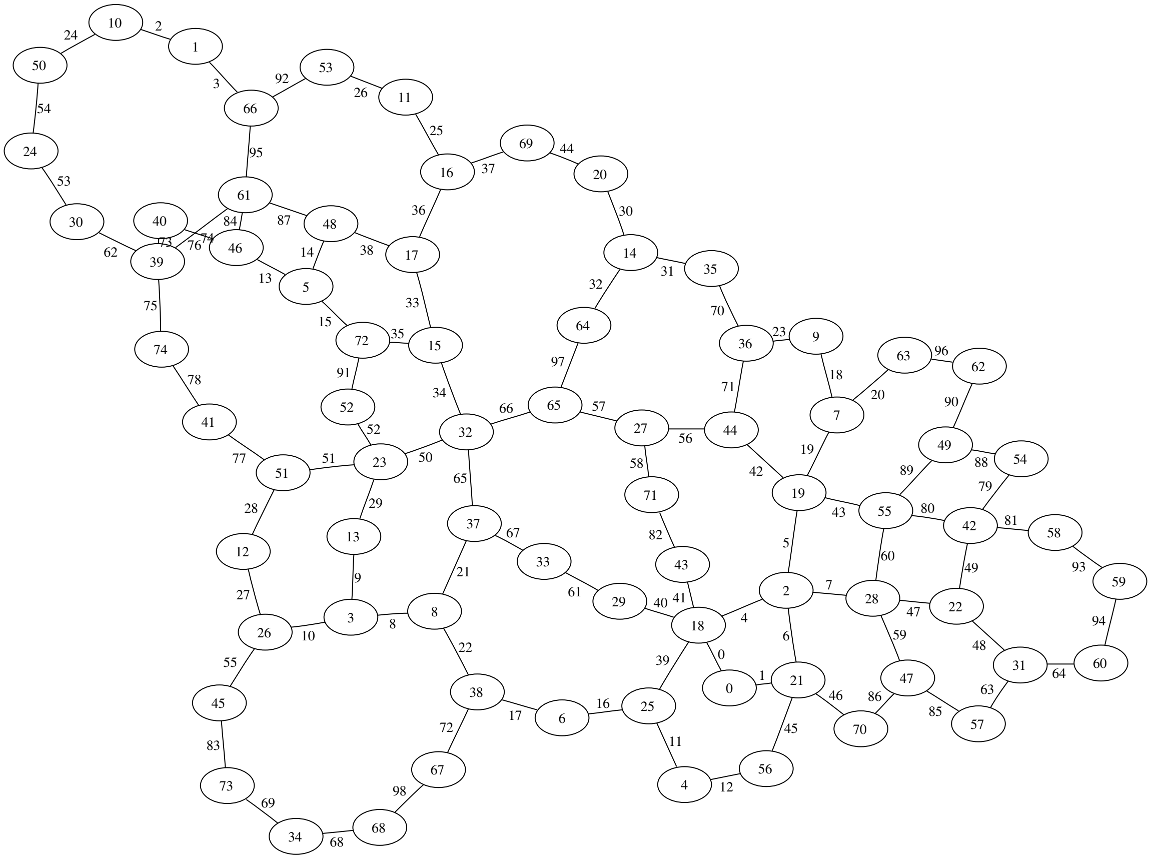


Figure 2.8: U.S. CORONET. Optical translucent network topology with link and node numbers.

Table 2.1: U.S. CORONET node-to-city map.

#	city	#	city	#	city
0	Abilene	25	Houston	50	Providence
1	Albany	26	Jacksonville	51	Raleigh
2	Albuquerque	27	Kansas City	52	Richmond
3	Atlanta	28	Las Vegas	53	Rochester
4	Austin	29	Little Rock	54	Sacramento
5	Baltimore	30	Long Island	55	Salt Lake City
6	Baton Rouge	31	Los Angeles	56	San Antonio
7	Billings	32	Louisville	57	San Diego
8	Birmingham	33	Memphis	58	San Francisco
9	Bismarck	34	Miami	59	San Jose
10	Boston	35	Milwaukee	60	Santa Barbara
11	Buffalo	36	Minneapolis	61	Scranton
12	Charleston	37	Nashville	62	Seattle
13	Charlotte	38	New Orleans	63	Spokane
14	Chicago	39	New York	64	Springfield
15	Cincinnati	40	Newark	65	St Louis
16	Cleveland	41	Norfolk	66	Syracuse
17	Columbus	42	Oakland	67	Tallahassee
18	Dallas	43	Oklahoma City	68	Tampa
19	Denver	44	Omaha	69	Toledo
20	Detroit	45	Orlando	70	Tucson
21	El Paso	46	Philadelphia	71	Tulsa
22	Fresno	47	Phoenix	72	Washington DC
23	Greensboro	48	Pittsburgh	73	West Palm Beach
24	Hartford	49	Portland	74	Wilmington

Table 2.2: U.S. CORONET link distance.

#	mile	#	mile	#	mile	#	mile
0	252.7	25	252.3	50	517.8	75	153.1
1	570.9	26	95.0	51	98.3	76	102.1
2	207.8	27	284.5	52	238.4	77	224.0
3	175.7	28	322.6	53	139.7	78	287.8
4	850.1	29	119.9	54	94.2	79	99.5
5	485.8	30	344.4	55	184.9	80	851.8
6	327.7	31	124.0	56	235.5	81	19.3
7	707.7	32	268.2	57	353.2	82	145.0
8	199.7	33	144.9	58	313.8	83	206.8
9	329.4	34	133.1	59	371.9	84	145.1
10	415.6	35	582.8	60	525.0	85	431.0
11	211.5	36	179.2	61	196.0	86	166.8
12	107.7	37	143.4	62	22.0	87	355.2
13	134.4	38	221.0	63	167.9	88	703.3
14	288.6	39	324.6	64	113.0	89	915.9
15	50.4	40	415.5	65	221.3	90	209.3
16	375.1	41	275.2	66	355.3	91	142.6
17	110.5	42	690.3	67	283.4	92	109.0
18	546.8	43	548.5	68	297.8	93	57.9
19	660.0	44	80.4	69	97.4	94	335.2
20	636.6	45	723.3	70	426.3	95	167.8
21	264.3	46	379.3	71	421.0	96	333.2
22	436.9	47	372.2	72	490.0	97	108.0
23	554.2	48	290.0	73	18.2	98	295.6
24	59.9	49	217.5	74	149.7		

distribution. The thesis does not differentiate connection requests of opposite directions. One arrival rate characterizes the load of each node pair. Existing connections depart in an exponential distribution.

2.4.2 Dynamic Wavelength Service

AT&T's dynamic wavelength service model allows customers to create or remove a wavelength connection on-demand on a dynamic ROADM network. Customers own or lease a set of OT ports to access the network. They can choose to connect between their free OTs at different nodes in arbitrary ways. Every connection occupies one of the customer's OTs at each end of the connection until the connection is released. The routes are preprovisioned at the design phase to guarantee 100% acceptance of valid connection requests. Network resources, such as REGENs and wavelengths, must be preinstalled so the customers can reconfigure their connections at any time and within a few minutes. A network can have many customers. But one customer is not allowed to connect to another customer's OT ports. The demands of each customer are independent. No sharing of network resources between customers is considered.

2.5 Simulation Assumptions

This thesis assumes that the topology of the backbone network is given. In particular, we evaluate simulation results on a few existing backbone topologies. Often, backbone networks are built along railways. The location of cities, costs of construction, and rights-of-way are all important factors in determining where to place fibers. The problem of network topology design is beyond the scope of this thesis.

The network is managed centrally. The route configuration time is shorter than the expected inter-arrival time. Each connection arrival/departure can be processed before the next arrival request. Bidirectional wavelength routes are automatically set up and taken down for each request. Each wavelength route occupies one wavelength channel on each link of both directions along the path until the connection terminates. Based on the routing policy and the current availability of wavelengths, the network either accepts or rejects

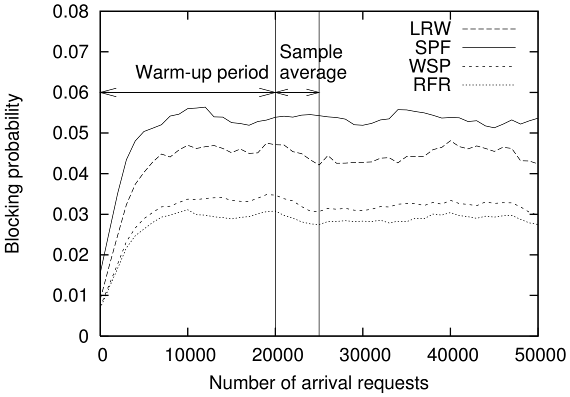


Figure 2.9: Steady state performance on NJ LATA.

a connection request. No queuing/preemption/rerouting is assumed. We use the same route for connection requests from opposite directions. We assume that the traffic load changes slowly relative to connection arrivals and departures. Specifically, request arrival rates and hold times for connections allow sufficient time for signaling such that all new connections are routed based on up-to-date information about residual capacity for all links in the network. The dynamic network system can come sufficiently close to a steady state.

This thesis studies the performance of networks under traffic evolutions. To model various traffic demands, I introduce a traffic matrix space that includes all possible traffic matrices. The traffic matrix space is defined by an average arrival rate and a departure rate for each end-to-end pair. One traffic matrix in the space is generated by randomly selecting an arrival rate for each pair. The selection of an arrival rate follows uniform distribution, normal distribution, or bimodal distribution of the same mean and deviation. One departure rate is used for all connections. Connection arrivals and departures are modeled by Poisson processes. The arrival process follows the arrival rate of each pair from the traffic matrix. The departure process of any connection follows the departure rate. Each connection requests one wavelength capacity.

The blocking probability is sampled by averaging 5000 arrivals at steady state behavior. The simulation is repeatedly run until the 95% confidence interval is within ±5% of the value or the blocking probability is lower than 0.0001. Figure 2.9 illustrates the steady state of a dynamic network. For all the test networks given our traffic load and all the algorithms considered, the system reaches a steady state after 20,000 requests.

CHAPTER 3

DIMENSIONING OPAQUE NETWORKS FOR EVOLVING TRAFFIC

New Internet applications are increasingly generating high-bandwidth, short-lived demands. If network resources are available, establishing a lightpath on-demand takes only a few minutes on today's reconfigurable optical networks. These demands thus create a more variable and unpredictable environment for long-term network planning. At the same time, upgrading backbone networks is expensive and therefore occurs infrequently. Dimensioning network resources to sustain variable traffic demands for the long term requires fewer upgrades to achieve high performance but poses complex challenges.

Two kinds of dimensioning problems for optical opaque networks are proposed and studied in this chapter: (1) *basic dimensioning*, which allocates network resources for a newly built network; and (2) *incremental dimensioning*, which allocates extra resources for future demand growth and variations. I propose new metrics to quantify the traffic load and the traffic pattern evolution for dynamically routed networks. I evaluate performance under load scaling, traffic evolution, and misdimensioning; a dimensioned network can sustain a much higher load while providing the same performance compared to misdimensioned ones. My approach is better adapted to traffic evolution than either a uniform allocation or the asymptotic optimization approach proposed earlier.

3.1 Load Definition

This section introduces network load metrics. Many previous studies still use the traditional Erlang unit to model traffic volume. An Erlang represents the use of a single voice channel; in practice, it has typically been measured by averaging the (instantaneous) number of active channels over the span of an hour. The Erlang unit can describe the average traffic volume on one link.

However, on a dynamic mesh network, the number of calls does not clearly connect to network resources. We can linearly approximate the resources used per call using an average, but the scaling factor must be calculated for a specific topology and a specific traffic pattern on that topology. The Erlang definition is too simplistic to represent network loads adequately.

I propose a new load metric. I model the average network resources used by the expected traffic by the sum of Poisson load weighted by the topological shortest path length for each request pair. Poisson dynamic traffic is assumed (see Section 2.4.1). Let N be the set of nodes and R be the set of all end-to-end request node pairs. Each pair $i \in R$ in the network is characterized by the mean arrival rate λ_i, departure rate μ_i, and capacity demand 1. Let E be the set of links. Each link $e \in E$ is dimensioned with total capacity $B_e + X_e$, where B_e is the capacity to support expected traffic, and X_e is the incremental capacity to support future traffic variations. The X_e portion can be deployed at a later time than the B_e portion. Once both portions are installed, they are treated the same in the total volume $B_e + X_e$. For a dimensioned network to support the expected traffic, the total basic capacity should equal the expected traffic volume, as shown in Equation (3.1).

$$\sum_{e \in E} B_e = \sum_{i \in R} \frac{\lambda_i}{\mu_i} TSL_i \tag{3.1}$$

where TSL_i is the topological shortest path length for request pair $i \in R$. The extra capacity, $X = \sum_{e \in E} X_e$, is dimensioned for future traffic scaling and variations. Therefore, the projected traffic *load ratio*, u, is defined by Equation (3.2).

$$u = \frac{\sum_{e \in E} B_e}{\sum_{e \in E}(B_e + X_e)} \tag{3.2}$$

At routing time, the actual load ratio l can be greater or smaller than the projected load ratio.

3.2 Dimensioning Procedure

The dimensioning algorithm appears in Algorithm 3.1. Basic dimensioning (Lines 1-12) computes basic capacity B, and incremental dimensioning (Lines 13-20) computes extra capacity X according to the type of incremen-

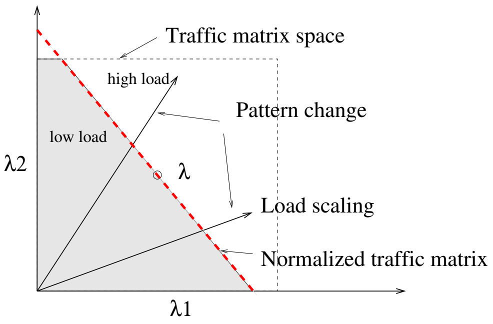


Figure 3.1: Traffic space and evolution model for two node pairs. The arrival rates are noted by λ_1 and λ_2.

tal dimensioning algorithm $\xi \in \{MEAN, SD\}$ used. If the projected load ratio is $u = 1$, $X = 0$, no incremental dimensioning is needed. In general, the simulation-based basic dimensioning algorithm can handle any traffic distribution with a given mean for arrival rate, departure rate, and capacity demand. Algorithm 3.2 shows a simpler analytical version for Poisson traffic that can generate equivalent results. However, the simulation version can be easily extended to handle more complicated cases. For example, if a link/node is physically bounded by a maximum capacity, the routes can avoid these links when they are full.

In our model, the load characteristics of each request pair (λ_i, μ_i) are randomly drawn from a traffic space. The traffic space is characterized by an arrival rate, a departure rate, and a distribution that defines the probability of possible arrival rates. Let λ be the mean arrival rate and μ be the mean departure rate. Any traffic matrix is a point inside the whole space of all possible traffic matrices that can be generated from the distribution. Since the departure rates are assumed the same, the arrival rates are the varying values. Let $T = \{\lambda_i, \mu_i\}^{|R|}$ denote a random projected traffic matrix that is drawn from the space. Figure 3.1 illustrates a traffic space model for two node pairs. Assume that the arrival rate follows a bounded uniform distribution of mean λ. The space is a square. Over time, the real traffic load can vary in both scale and pattern, which is characterized as *traffic evolution* in Section 3.3.

The *average traffic load* is defined in Equation 3.3 such that all arrival rates equal the mean arrival rate λ.

$$load_{avg} = \frac{\lambda}{\mu}\sum_{j\in R} TSL_j \tag{3.3}$$

The load of a random traffic matrix can be either lower or higher than the average traffic load. In practice, estimation of projected traffic matrices is susceptible to measurement errors or obsolete data. In order for the dimensioning algorithm to be robust to traffic load changes, the arrival rates of a projected traffic matrix are adjusted proportionally so the new load equals the average load. Equation (3.4) shows the normalization. Given a traffic matrix, each arrival rate is scaled by a constant.

$$\lambda_i^n = \lambda_i \frac{load_{avg}}{\sum_{j\in R}\frac{\lambda_j}{\mu_j} TSL_j} \tag{3.4}$$

The goal of incremental dimensioning is to allocate extra capacity, perhaps at a later time, to adapt to traffic evolution. The method should increase capacity allocation without interfering with established connections (and resources).

Two incremental dimensioning schemes (MEAN and SD) are proposed. The MEAN scheme increases each link's capacity by an amount proportional to that capacity. MEAN over-dimensioning occurs *implicitly* when the offered traffic load ratio is smaller than the projected load ratio. The result is equivalent to *explicit* MEAN over-dimensioning in that the amount $(1-l)\sum_e B_e$ is the part of the extra capacity that is incrementally dimensioned using the MEAN algorithm.

The other scaling approach, SD, increases each link's capacity proportionally to the statistical standard deviation of the basic dimensioned link capacity. The idea behind SD is that links with larger deviations tend to block more frequently due to traffic variations. It is thus more effective to allocate extra capacity to these links rather than to those with smaller traffic fluctuations.

$ADJUST(x)$ is a procedure that adjusts the simulated capacity of each link into an integer (if it were not already) and makes the summed capacity of all links to the expected traffic load. The result B should satisfy Equa-

Algorithm 3.1: Dimensioning procedures $(B, X) = BAL(T, u, \xi)$.

```
begin Basic dimensioning
    Input: $T$
    Get normalized traffic matrix $T^n$ (using Equation (3.4));
    while System has not reached steady state do
        Draw a random event according to the traffic matrix $T^n$;
        if An arrival event of connection request $j$ then
            Uniformly select a topological shortest path (SPF) $p$;
            forall the Link $e \in p$ do
                $C_e \leftarrow C_e + 1$;
        else
            /* An departure event of connection request $j$ */
            forall the Link $e \in p$ where $p$ is the route of $j$ do
                $C_e \leftarrow C_e - 1$;
    $B \leftarrow ADJUST(C)$ ;    /* Do it only once after incremental
    dimensioning if incremental dimensioning is done immediately
    after basic dimensioning */
begin Incremental dimensioning
    Input: $B, u, \xi$
    Compute extra capacity $X = \frac{1-u}{u} \sum_{e \in E} B_e$;
    Compute statistic deviation $\sigma_e = \sqrt{B_e}$;
    if $\xi = MEAN$ then
        Get extra link capacity $X_e = X \frac{B_e}{\sum_{e \in E} B_e}$;
    else if $\xi = SD$ then
        Get extra link capacity $X_e = X \frac{\sigma_e}{\sum_{e \in E} \sigma_e}$;
    $X \leftarrow ADJUST(X)$;
```

tion (3.1), and X should satisfy Equation (3.2), where the traffic rates are normalized. Since the load of traffic T^n has been normalized, the computed sum of B_es at steady state is the same as the expected load $\frac{\lambda}{\mu} \sum_{i \in R} TSL_i$. The algorithm is presented in Procedure ADJUST(). I first round down the real numbers, and then adjust from the difference to the expected value computed by the equations. The total capacity difference is less than 0.2% of the expected load volume.

Algorithm 3.2: Basic dimensioning with Poisson-independent traffic matrix $T\{\lambda_i, \mu_i\}$.

```
$\forall e \in E, C_e \leftarrow 0$;
foreach request pair $r \in R$ do
    foreach topological shortest path $p \in TSP_i$ do
        foreach link $e \in p$ do
            $C_e \leftarrow C_e + \frac{\lambda_i}{\mu_i |TSP_i|}$;
$B \leftarrow ADJUST(C)$;
```

Algorithm 3.3: Adjusted the simulated capacity ADJUST()

```
Input: Per link capacity $B$
Input: Total capacity $t$
Output: Per link capacity $B$
$\forall e \in E, C_e \leftarrow \lfloor B_e \rfloor$;
Sort $C_e$ ascendantly by $B_e - C_e$ and let $c(0 \ldots |E| - 1)$ be the sorted array;
Computer the difference $d \leftarrow t - \sum_{e \in E} C_e$;
if $d > 0$ then
    $i \leftarrow |E| - 1$;
else
    $i \leftarrow 0$;
while $d \neq 0$ do
    if $d > 0$ then
        $c(i \mod |E|) \leftarrow c(i \mod |E|) + 1$;
        $d \leftarrow d - 1$;
        $i \leftarrow i - 1$;
    else
        $c(i \mod |E|) \leftarrow c(i \mod |E|) - 1$;
        $d \leftarrow d + 1$;
        $i \leftarrow i + 1$;
Copy $c()$ into $B$;
```

3.3 Traffic Evolution Model

The future offered traffic T' may vary from the normalized projected traffic matrix T^n in two ways: load scaling and pattern change. Figure 3.1 shows load scaling and pattern changes. *Load scaling* means that the actual traffic matrix is identical to the projected traffic matrix after normalization. If it is not, the situation is referred to as a *pattern change. Traffic evolution* is a combination of pure load scaling and pattern change. Assume that the new traffic matrix (matrices) T' is drawn from the same distribution. Let λ'_i, μ'_i be the new load characteristic for a pair i in T'. The offered load ratio l_i per pair is the ratio of the actual load to the normalized projected load, as shown in Equation (3.5).

$$l_i = \frac{\lambda'_i}{\mu'_i} \frac{\mu^n_i}{\lambda^n_i} \tag{3.5}$$

Pure load scaling happens when the loads of all connection pairs from the projected traffic matrix increase/decrease in the same ratio ($l_i = l$ for all i). However, if l_i varies with i, the variation indicates a change in traffic pattern. The level of change in the traffic pattern can be modeled as linear evolution of the original traffic (Equation (3.6)). I define ϵ as the traffic variance factor that measures the degree of evolution in a transition from the original traffic

T^n to a new traffic T'. The actual matrix T_ϵ is computed by Equation (3.6) using the matrix sum. Traffic evolution indicates a change of traffic load in most cases.

$$T_\epsilon = (1 - \epsilon)T^n + \epsilon T' \tag{3.6}$$

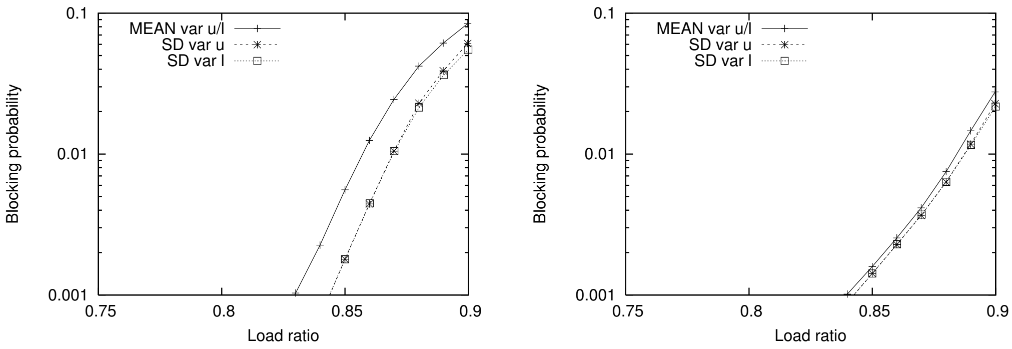
3.4 Performance Study

This section quantitatively measures the benefits of basic dimensioning and incremental dimensioning techniques for actual, random, and expected traffic patterns in the traffic evolution model.

3.4.1 Simulation Setup

The simulator was built according to the network model described in Section 2.5. In my simulation, the arrival rate of each connection ranges from 1 to 10. By default, the selection distribution is uniform. Each connection requests one wavelength channel and has the same departure rate. The departure rate is determined by the network load, which is the product of the projected departure rate and the load ratio. The setup is statistically equivalent to the general traffic model with variable departure rate and capacity demands. I use three well-known networks for the experiments: NJ LATA, COST 239, and ARPANET (Figure 2.6). The total capacity stays the same for each network in all experiments, with an average of 120 wavelengths per link. Network performance does vary with changes in the average link capacity. The current setting provides a interesting range of operating load with reasonable simulation effort.

The dimensioning process $BAL(T, u, \xi)$ is defined by Algorithm 3.1. $RUN(T, B, X, \beta, l)$ is defined as a simulated routing process with a choice of an online routing algorithm $\beta \in \{SPF, WSP, LRW, CAR\}$, given traffic matrix T and capacity resource (B, X). SPF, WSP, and LRW are introduced in Chapter 2. An optimized dynamic routing algorithm, congestion aware routing (CAR) (which is introduced in Chapter 4), is also applied here. The blocking probability is sampled by averaging 5000 arrivals in steady state. For all test networks given the traffic load, the system reaches steady state after 20,000 requests.



(a) ARPANET

(b) COST

Figure 3.2: Comparing projected load scaling (var u with $l = u, \epsilon = 0$) and offered load scaling (var l with $u = 0.85, \epsilon = 0$) using Algorithm 3.4 and SPF routing.

3.4.2 Traffic Load Scaling

I first study performance when complete traffic pattern information is known at the dimensioning stage. The normalized future traffic matrix is the same as the projected traffic matrix. There are two kinds of load scaling scenarios, projected load scaling and offered load scaling (Algorithm 3.4). In *projected load scaling*, the ratio of incremental dimensioning u varies with the projected load ratio, as specified by Equation (3.2). The offered load l is the same as the projected load u. As u increases, the portion of overdimensioned capacity on each link decreases proportionally. In *offered load scaling*, the network is assumed to be dimensioned at a fixed projected load ratio ($u = 0.85$, and l varies from 0.75 to 0.9). However, the real traffic offered scales over the projected traffic with the same ratio defined by Equation (3.5). The results of both experiments are averaged across $m = 100$ random projected traffic matrices. For both load scaling cases, MEAN results are identical because offered load scaling is the same as implicit MEAN over-dimensioning. Figure 3.2 shows that the SD approach improves relative to MEAN on both ARPANET and COST. The performance results of projected load scaling and offered load scaling are the same in the studied load range.

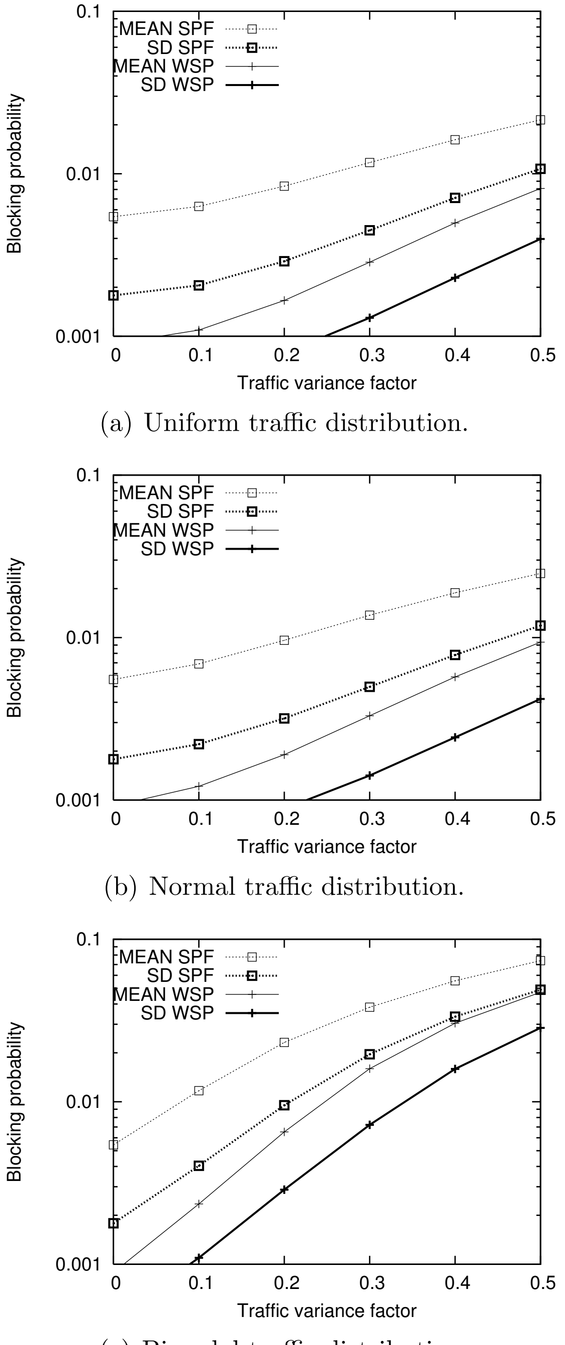
Algorithm 3.4: Test procedure for load scaling with variable projected load ratios ($l = u, \epsilon = 0$), variable offered load ratios ($u = 0.85, \epsilon = 0$), and random evolutionary traffic patterns ($u = l$).

```
foreach projected load ratio $l$ do
    foreach project traffic matrices $T_i$ picked from a sample set of size $m$ do
        Dimensioning $(B, X) = BAL(T_i, u, \xi)$;
        foreach traffic evolution factor $\epsilon$ do
            foreach traffic matrices $T'_j$ picked from another sample set of size $n$
            do /* $n$ is picked so the 95% confidence interval of $p_{i,\epsilon,l}$ is
            within $\pm 5\%$ of the data or it is lower than $0.0001$. */
                Test blocking probability
                $p_{j,i,\epsilon,l} = RUN((1-\epsilon)T_i + \epsilon T'_j, B, X, \beta, l)$;
            Get average blocking $p_{i,\epsilon,l} = \frac{1}{n}\sum_{j=1}^{n} p_{j,i,\epsilon,l}$;
        Get average blocking $p_{\epsilon,l} = \frac{1}{m}\sum_{i=1}^{m} p_{i,\epsilon,l}$;
```

3.4.3 Traffic Pattern Evolution

At a given load, I show the amount of performance degradation when the traffic pattern varies from the projected traffic matrix. I compare incremental dimensioning algorithms, MEAN and SD, on evolutionary traffic patterns so that the load for each pair of connections λ_i/μ_i varies differently. The arrival rate λ_i of each connection pair is picked randomly from three different general distributions: uniform, normal and bimodal. The uniform distribution ranges from 1 to 10. The normal distribution generates a Gaussian random variable in the range of 1 to 10 with mean 5.5 and variance 6.75. The bimodal distribution picks either 2.9 or 8.1 with 50% probability. All three distributions thus have the same mean and variance. The departure rate μ_i is the same for all arrivals.

The test procedure for evolution traffic is shown in Algorithm 3.4. Figures 3.3 and 3.4 show the blocking probability of SPF and WSP using SD and MEAN on ARPANET and COST at load 0.85. The actual traffic is a mix of projected and future traffic in a ratio from 0 to 50 percent, i.e., $\epsilon \in [0, 0.5]$. Again, I randomly choose $m = 100$ projected traffic matrices. On ARPANET, SD improves MEAN by over 50% when $\epsilon = 0.5$. The improvement is smaller on COST.

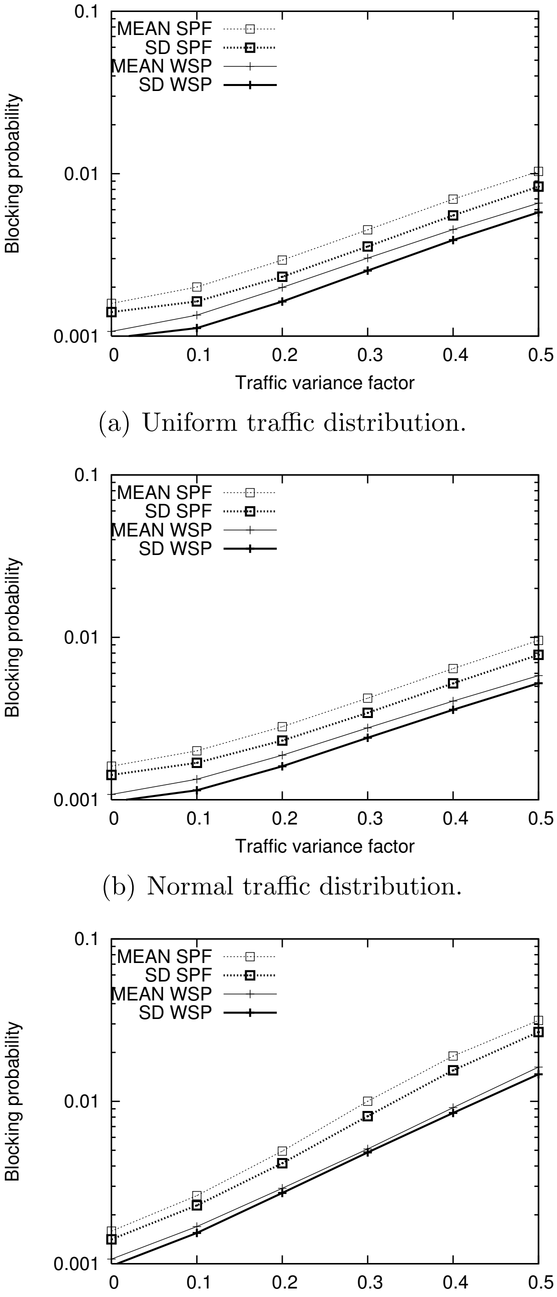


(a) Uniform traffic distribution.

(b) Normal traffic distribution.

(c) Bimodal traffic distribution.

Figure 3.3: Comparison of MEAN and SD on traffic pattern evolution on ARPANET (using Algorithm 3.4 with $u = l = 0.85$).



(a) Uniform traffic distribution.

(b) Normal traffic distribution.

(c) Bimodal traffic distribution.

Figure 3.4: Comparison of MEAN and SD on traffic pattern evolution on COST (using Algorithm 3.4 with $u = l = 0.85$).

Algorithm 3.5: Test procedure for expected traffic evolution and randomized misdimensioning ($l = u$).

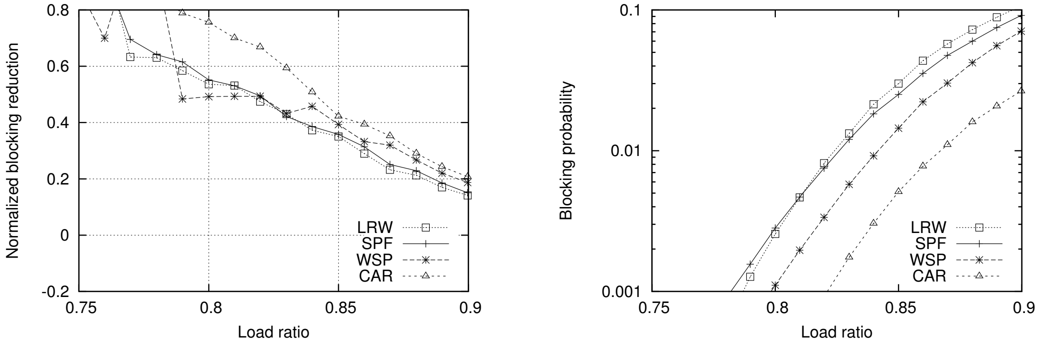
```
foreach projected load ratio $u$ do
    foreach project traffic matrices $T_i$ picked from a sample set of size $m$ do
        Dimensioning $(B, X)_i = BAL(T_i, u, \xi)$;
    Get average link capacity $(\bar{B}, \bar{X}) = \frac{1}{m}\sum_{i=1}^{m}(B, X)_i$;
    foreach $s$ randomly misdimensioned network $(\hat{B}, \hat{X}) = UNBAL(\alpha, \bar{B}, \bar{X})$ do
        foreach traffic matrices $T'_j$ picked from another sample set of size $n$ do
        /* $n$ is picked so the 95% confidence interval of $p_{k,u,\alpha}$ is
        within ±5% of the data or it is lower than 0.0001.  */
            Test blocking probability $p_{j,k,u,\alpha} = RUN(T'_j, \hat{B}, \hat{X}, \beta, l = u)$;
                                  /* Complete traffic evolution $\epsilon = 1$.  */
        Get average blocking $p_{k,u,\alpha} = \frac{1}{n}\sum_{j=1}^{n} p_{j,k,u,\alpha}$;
    Get average blocking $p_{u,\alpha} = \frac{1}{s}\sum_{k=1}^{s} p_{k,u,\alpha}$;
```

3.4.4 Dimensioning for Expected Traffic Patterns

The performance improves by taking a set of expected traffic matrices (instead of one matrix) during dimensioning. The idea is to average the dimensioned capacity of a set of projected traffic matrices. The test procedure is described in Algorithm 3.5. The evolution factor is 100%. Unlike Algorithm 3.4, which uses only one projected traffic matrix at the dimensioning stage for each test, Algorithm 3.5 considers all m projected matrices in one dimensioning step. The network misdimensioning function, $UNBAL(\alpha, \bar{B}, \bar{X})$, is used in Section 3.4.5 to explore my misdimensioning model. Here, I pass the misdimensioning part by setting $\alpha = 0$ so $(\hat{B}, \hat{X}) = (\bar{B}, \bar{X})$ and $s = 1$ and randomly generate $m = 100$ for dimensioning. Many random testing matrices n are used to meet the confidence interval requirement. The arrival rate is uniform from 1 to 10.

Figures 3.5 and 3.6 present the normalized blocking reduction from MEAN to SD and the average blocking probability of SD over all testing matrices, for routing algorithms in set β. The normalized blocking reduction is defined by one minus the ratio of the blocking probability of SD to the blocking probability of MEAN. On ARPANET, SD reduces blocking by 10% to 80% over the load range for all routing algorithms. On COST, the reduction is around 0% to 10%.

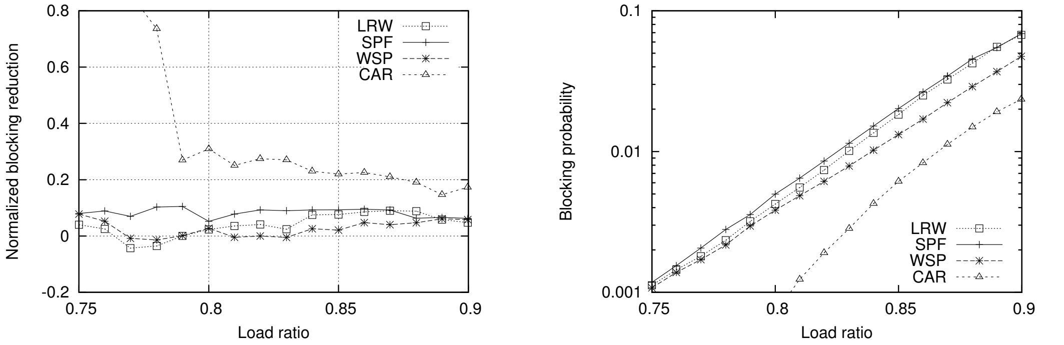
I next investigate why the improvement of SD over MEAN is smaller on COST than on ARPANET. *Mean square difference (MSD)* defines the difference of the dimensioned capacity between MEAN and SD. For m projected



(a) Normalized SD blocking reduction over MEAN.

(b) Blocking probability using SD incremental dimensioning.

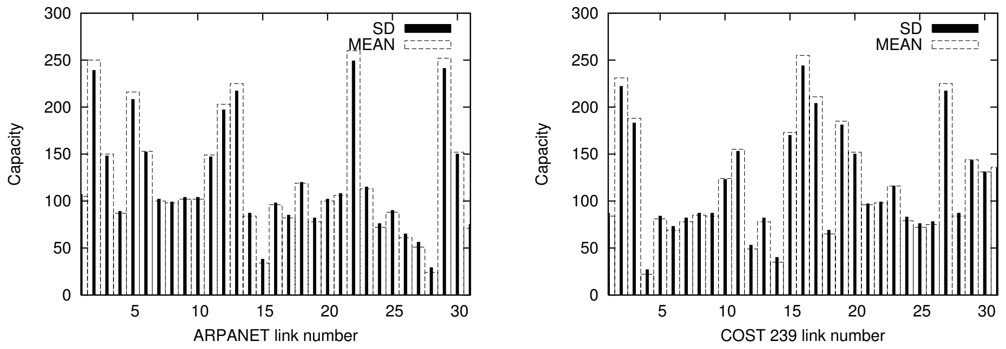
Figure 3.5: Comparison of MEAN and SD on ARPANET dimensioned with 100 random projected traffic matrices, with varying traffic loads and patterns (using Algorithm 3.5, $\alpha = 0$).



(a) Normalized SD blocking reduction over MEAN.

(b) Average blocking probability over all test traffic matrices using SD incremental dimensioning.

Figure 3.6: Comparison of MEAN and SD on COST dimensioned with 100 random projected traffic matrices, with varying traffic loads and patterns (using Algorithm 3.5, $\alpha = 0$).



(a) ARPANET $u = 0.85$.

(b) COST $u = 0.85$.

Figure 3.7: Comparison of total dimensioned capacity with $m = 100$ randomly picked projected traffic matrices.

traffic matrices $T_i, 1 \leq i \leq m$, and projected load ratio u, let the capacity of MEAN incremental dimensioning be:

$$(\bar{B}^M, \bar{X}^M) = \frac{1}{m}\sum_{i=1}^{m} BAL(T_i, u, MEAN)$$

and the capacity of SD incremental dimensioning be:

$$(\bar{B}^S, \bar{X}^S) = \frac{1}{m}\sum_{i=1}^{m} BAL(T_i, u, SD)$$

MSD is defined by Equation (3.7) as follows:

$$\frac{1}{|E|}\sum_{e \in E}(\bar{B}_e^M + \bar{X}_e^M - \bar{B}_e^S - \bar{X}_e^S)^2 \tag{3.7}$$

The SD and MEAN schemes are identical at the limit $u = 1$ since $\bar{B}_e^M = \bar{B}_e^S$ and no incremental capacity exists ($\forall e, X_e = 0$). However, they are not the same with $u < 1$.

For $m = 100$ random projected matrices, Figures 3.7(a) and (b) compare the total dimensioned capacity at projected load ratio $u = 0.85$ between SD and MEAN. Table 3.1(a) compares the MSD between SD and MEAN at various dimensioning loads. Clearly, COST has a smaller MSD than ARPANET for all loads. The improvement of blocking probability for SD relative to MEAN is thus also smaller on COST.

Table 3.1: Mean square capacity difference (MSD) for dimensioning for various projected loads.

(a) SD and MEAN

u	**0.75**	**0.80**	**0.85**	**0.90**	**1.00**
ARPANET	64.81	40.00	23.31	10.00	0
COST	50.22	30.70	18.27	8.32	0

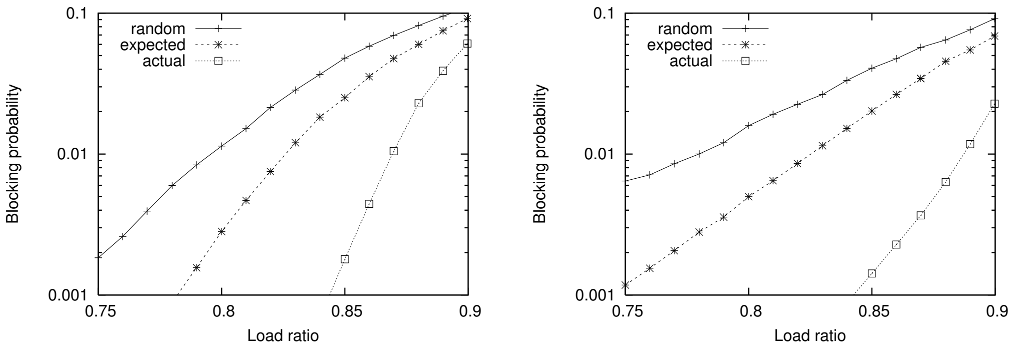
(b) SD and UNI

u	**0.75**	**0.80**	**0.85**	**0.90**	**1.00**
ARPANET	4018.81	4003.69	4001.00	3998.94	3939.19
COST	3401.89	3369.89	3390.59	3400.92	3332.59

Finally, I show the benefit of having prior traffic pattern information during the dimensioning stage. Figure 3.8 compares the blocking probability of SPF using SD with actual, random, and expected traffic patterns. Dimensioning for the actual traffic pattern, the load scaling case, is when the statistical behavior of all connection pairs is known but not the actual scale. The normalized traffic matrix of the real traffic is identical to the projected traffic matrix (Algorithm 3.4 with $l = u, \epsilon = 0$). Dimensioning for the random traffic pattern is the complete traffic evolution case with $l = u, \epsilon = 1$. Dimensioning for the expected traffic pattern uses 100 random projected matrices (Algorithm 3.5 with $m = 100, \alpha = 0$). The n testing traffic matrices, randomly drawn from the original space, are not limited by 100 projected traffic matrices. Both networks show significant performance improvements with more prior knowledge of future traffic patterns. The performance of dimensioning for the actual traffic pattern degrades quickly when traffic evolves into random traffic patterns. If the traffic is expected to evolve rapidly and unpredictably, dimensioning for expected traffic that uses a set of traffic matrices gives better performance.

3.4.5 Robustness

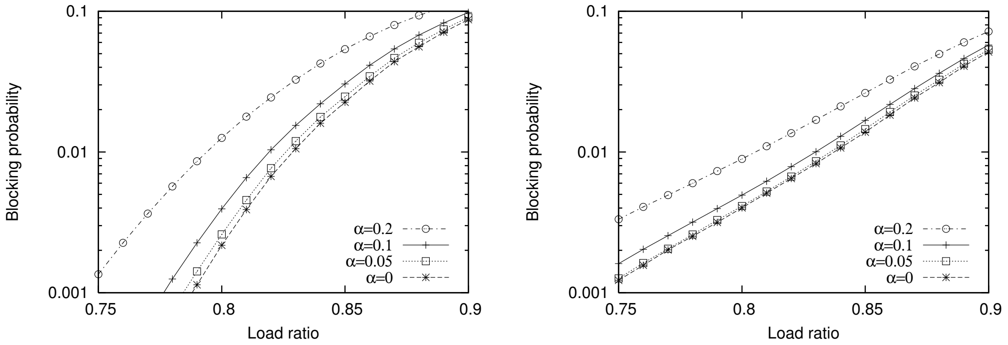
To study the sensitivity of the dimensioning algorithm, I randomly misdimension the network and compare the performance. Noticeable performance degradation starts to shown when each link is misdimensioned by at most 10% (about ±12 wavelengths on average). The misdimensioning is modeled by a *misdimensioning factor* $\alpha \in [0, 1]$. $UNBAL(\alpha, B, X)$ denotes the procedure to randomly misdimension a network. Given α, each dimensioned link e with



(a) ARPANET

(b) COST

Figure 3.8: The performance of SPF on SD dimensioned networks for actual, expected and random traffic patterns.



(a) ARPANET

(b) COST

Figure 3.9: Performance degradation on misdimensioned networks for WSP routing (using Algorithm 3.5).

capacity B_e is re-dimensioned with a random capacity $\hat{B}_e$ that is picked uniformly from the range $[\max(\lfloor B_e(1-\alpha)\rfloor, 0), \lceil B_e(1+\alpha)\rceil]$. The same approach is applied to the extra capacity X_e. Hence, $(B, X) = UNBAL(0, B, X)$. The difference of total capacity is fixed by adjusting each link capacity up/down to guarantee the same load. The network is more likely to be misdimensioned as α increases. Figure 3.9 compares the blocking performance of WSP using MEAN with varying α values. The test was repeated for 100 randomly generated misdimensioned networks. The result becomes worse as the misdimensioning factor increases above 0.1. Similar results were found for SD incremental dimensioning.

3.5 Comparisons

In this section, I show that dimensioned networks achieve significant improvements over undimensioned ones. A comparison with a previous study indicates that dynamic routing with dimensioning achieves better performance than fixed routing with dimensioning.

3.5.1 Uniform Dimensioning

The importance of network dimensioning can be illustrated by comparing the performance of a dimensioned network with a uniform link capacity network. Uniform dimensioning (called UNI) allocates the same capacity to all links. It is 120, so the total capacity is the same as other dimensioning approaches. The performance of WSP with random testing traffic matrices is compared in Figures 3.10 and 3.11, showing that the same routing algorithm can reduce blocking probability by 10 times on a better dimensioned network. Many links tend to be underutilized on UNI networks because the capacity is not distributed according to topology and traffic demand. Table 3.1(b) shows the MSD between a SD dimensioned network and a uniform one. The MSD between SD and UNI is much greater than that between SD and MEAN. Earlier work [36] showed that a quick cut can form on a UNI network even when the offered load is low.

3.5.2 Fixed Dimensioning and Routing

Nayak and Sivarajan [37] proposed a dimensioning method (called ABS here) for dynamic traffic with increasing demands. Their incremental dimensioning algorithm is equivalent to the MEAN scheme. Comparing my methods to ABS on one traffic matrix has been shown in previous work [36]. For evolutionary traffic with $\epsilon = 1$, I further show that my methods outperform ABS on the averaging of $m = 100$ randomly chosen projected traffic matrices and 100% traffic evolution at runtime with n randomly chosen runtime traffic matrices. Figures 3.10 and 3.11 compare blocking and utilization of BAL and ABS using different routing algorithms. SD supports dynamic routing better at most load ranges. ABS and fixed SPF routing tend to underuti-

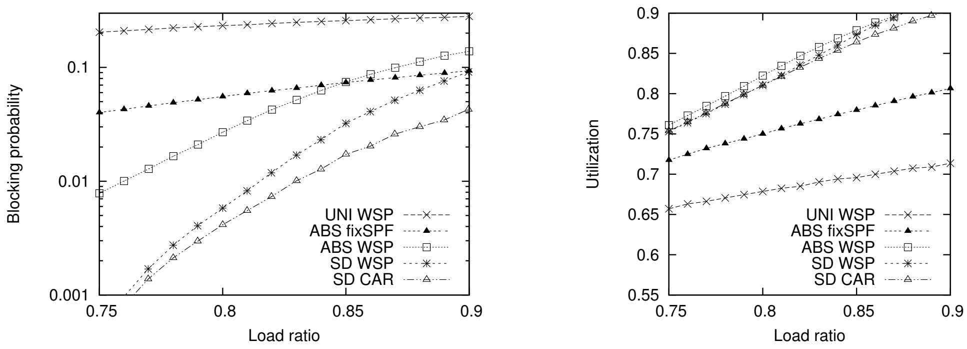


Figure 3.10: Comparison of UNI, ABS, and BAL for random traffic patterns on ARPANET (using Algorithm 3.4 with $l = u, \epsilon = 1$).

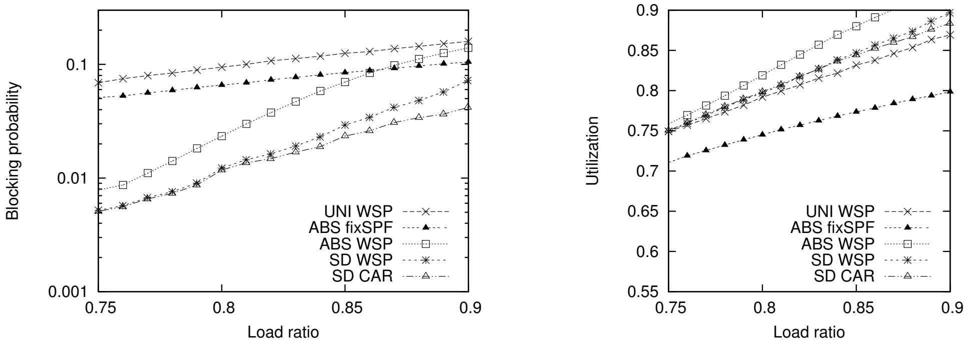


Figure 3.11: Comparison of UNI, ABS, and BAL for random traffic patterns on COST (using Algorithm 3.4 with $l = u, \epsilon = 1$).

lize network resources. Using better dynamic load balancing routing (CAR) improves the results over all load ranges.

3.6 Conclusion

This section introduces simulation-based network wavelength capacity dimensioning algorithms for dynamic optical networks. I propose metrics for traffic load and evolution to quantify traffic variations. Two stages of resource dimensioning are discussed: basic dimensioning and incremental dimensioning. The basic dimensioning algorithm dimensions network resources according to traffic projection. Incremental dimensioning over-dimensions the network for future traffic evolution. Two incremental dimensioning approaches, MEAN and SD, are proposed. The performance of the algorithms are evaluated in terms of blocking probability and robustness.

I show that dynamic routing algorithms, especially flow balancing algorithms, are leveraged by my capacity dimensioning techniques. Compared to fixed routing and dimensioning schemes, my approach provides better results for dynamic traffic. For incremental dimensioning schemes, scaling network capacity by standard deviation outperforms linear scaling in supporting traffic variations. My solution is robust for random mis-dimensioning by an average of 10% of the wavelengths on each link. Dimensioning for the expected traffic pattern is more robust to traffic evolution than dimensioning for one specific traffic pattern.

CHAPTER 4

ONLINE ROUTING AND CONGESTION CONTROL ON DIMENSIONED OPAQUE NETWORKS

Future network platforms must be able to support heterogeneous traffic models and provide low service-blocking rates, while minimizing the cost of expensive network resources. The difficulty of predicting changes in traffic demands and the need for a rapid and robust response to traffic pattern changes have thus generated great interest in online routing algorithms at all levels of the network.

This chapter introduces efficient load balancing routing and admission control mechanisms that are optimized for the dimensioned optical opaque networks introduced in Chapter 3. An online routing algorithm is to choose an available path upon the arrival of a connection request based on the current network capacity state. The connections are bidirectional. Only one wavelength is requested per connection. Each lightpath occupies one wavelength on each link in both directions along the path until the connection terminates. Based on the routing policy and the current availability of wavelengths, the network either accepts or rejects a connection request. No queuing/preemption/rerouting is assumed.

The key question to answer for online routing is to what extent the chosen route will affect the chance of blocking of future arrivals. I start with analyzing the flow reduction of other pairs for each chosen route. I find that choosing longer routes than the available shortest path on the residual graph increases only the chance of blocking for future arrivals (Section 4.1). My study also shows that the impact to other potential connections are local, as the dimensioning algorithm distributes resources regarding pair's topological shortest paths. We need to consider only the highly likely used region (a.k.a. shortest paths reduced graph) of each other arrival pairs to compute the level of congestion. Such reduction greatly simplifies the flow computation and improves the results.

Further, I realize that the available shortest route can become excessively long when the network is congested. Using extra resources beyond what has been dimensioned for can only worsen network congestion. I propose a threshold-based admission algorithm to limit the acceptance of excessively long routes (Section 4.2.2). In order to find the optimal threshold, I develop an opportunity cost model that is able to approximate the expected loss of future arrivals. The model predicts optimal thresholds, and the optimized thresholds remain the same for a set of practical network topologies in various scales. I also find that expensive flow analysis becomes unnecessary when admission control is used. So, a simple load balancing routing algorithm can be applied with admission control to achieve the best blocking performance. I also present the performance of algorithms on misdimensioned networks.

Finally, I build an oracular optimization model to explore the theoretical optimality if a finite sequence of future arrivals is given. I show that computing the optimal solution is too expensive, even with a small set of oracular arrivals, to provide enough capability for understanding any practical benefits.

My study shows that the effect of admission control dominates sophisticated route selections. A simple admission control rule that is scalable and robust is useful for practical networks.

4.1 Reduced Flow Routing Algorithm

This section presents a novel reduced flow routing (RFR) algorithm, which is a flow-based method, providing fast analysis of residual capacity. RFR reduces blocking without knowledge of future arrivals.

4.1.1 Multiple Shortest Paths

Among the many different routing algorithms, SPF-based routing algorithms generally outperform non-SPF counterparts under the same traffic distribution among all possible source-destination request pairs. Some representative comparisons are shown in Section 4.1.3. Given the superior performance and timing benefits of choosing from shortest paths, I was motivated to better un-

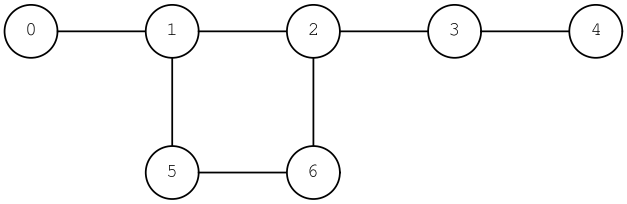


Figure 4.1: Sample graph for SPF ties.

derstand the selection amongst multiple shortest paths and design a routing algorithm based on the trade-offs.

The shortest path between any pair of nodes in a typical network topology is often not unique, even when network capacity constraints are considered. Intuitively, since most topologies are at least two-link-connected so as to be robust to link failures, any node pair sits on some cycle in the network. If the cycle is of even length, multiple paths exist. To illustrate how this property holds under dynamic loading of networks, I counted the number of topological and dynamic shortest paths (i.e., paths based on residual capacity at load 60%) found in routing 25,000 connection requests selected uniformly at random on networks with uniform capacity. Table 4.1 summarizes the results for average tied shortest paths per request and the percentage of requests that have more than one shortest path available. In most networks, over 40% of the dynamic requests have more than one shortest path.

Table 4.1: SPF ties.

Network	**Topological**		**Dynamic load 0.6**	
	# ties per req	**% req**	**# ties per req**	**% req**
NJ LATA	1.80	45	1.74	44
NATIONAL	2.07	47	1.77	40
COST 239	1.80	47	1.64	41
ARPANET	1.48	26	1.20	20

The simplest and most common approach to dealing with this multiplicity is to break ties randomly. Although such an approach is fast, it also increases the chance of blocking future requests. By using slightly more complicated tie-breaking algorithms that consider information about the network topology and traffic load, I can make better decisions and reduce the expected blocking rate.

As an example, consider the problem of routing a connection from node 1 to node 6 on the network of Figure 4.1. Assume that each link has sufficient capacity to route the current request. The two shortest paths are 1-5-6 and 1-2-6. Choosing either of these may interfere with future requests. For example, if I select path 1-5-6, link [1, 5] is on one of the SPF paths of six other node pairs: (0, 5), (0, 6), (1, 5), (2, 5), (3, 5), and (4, 5). Similarly, link [5, 6] may interfere with future requests from five pairs: (0, 6), (2, 5), (3, 5), (4, 5), and (5, 6). Thus, a total of 11 pairs (some counted twice in this simple analysis) may be blocked later if route 1-4-5 is taken. However, links [1, 2] and [2, 6] are part of the SPF paths of 14 other pairs. Assuming uniform traffic for all node pairs, I can argue that path 1-2-6 is more likely to interfere with future requests. Intuitively, a more careful selection of shortest paths among ties may reduce overall blocking. The next section explores this problem and describes my algorithm to address the issue using information on network topology and residual capacity.

4.1.2 Leveraging Residual Capacity

Network residual capacity is the most commonly used information for online algorithms in selecting a path. WSP provides one simple selection method based on the bottleneck residual capacity information. The new RFR algorithm provides fast analysis of residual capacity and reduces blocking.

Algorithm 4.1 explains reduced flow routing. Initially, I compute the SPF reduced graph (SRG_i) for each request pair i on a given network topology. SRG_i is a subgraph consisting of all topologically shortest paths for a pair i. RFR computes the maximum flow $f_j(C, SRG_i)$ on the SRG with current residual capacity C for each request pair other than the current connection request. I compute the flow on SRG rather than on the entire graph because (1) the computation time is greatly reduced, and (2) considering too many links increases the sharing of links from different pairs and cancels out the essential information indicating future network capacity demands, eventually resulting in more blocking. Therefore, we think of SRG in general as a critical part of the graph for a particular pair.

Any occupation on the critical part of the graph by routes of other connections would interfere with the future routing requests issued by this pair.

Algorithm 4.1: Reduced Flow Routing (RFR)

```
begin Initialize SRGs
    foreach $i \in R$ do
        Compute $SRG_i$;
begin RFR
    Input: $i \in R$
    foreach $p \in TSP_i(C)$ do
        foreach $j \neq i \in R$ do
            $a_j \leftarrow f_j(C, SRG_i)$;
            foreach $e \in p$ do
                $\widetilde{C}_e = C_e - 1$;
            foreach $e \notin p$ do
                $\widetilde{C}_e = C_e$;
            $b_j \leftarrow f_j(\widetilde{C}, SRG_i)$;
            $g_j \leftarrow \frac{a_j - b_j}{a_j}$;
        $cost_p \leftarrow \sum_j g_j$;
    $path \leftarrow \arg\min_{p \in TSP_i(C)} cost_p$;
```

Such interference is instantiated by the reduction of max-flow caused by the routing pair. As several SPFs are found for the routing request pair, I select one SPF and temporarily remove the request capacity along the path. The network capacity then becomes $\widetilde{C}$. I again compute the maximum flow for all other pairs on the SRG with modified capacity information. Then, I compute the difference between two flows for each pair and normalize it by the original flow. Considering all other pairs, the total cost for picking up one SPF will be the summation of all other pairs by their normalized flow. For each shortest path, I compute the cost and finally choose the one with the minimal cost. Essentially, I am choosing the path that results in minimum interference on the critical subgraph of the other pairs.

Figure 4.2 shows a sample SRG for pair (1, 10) in NJ LATA with residual capacity indicated on each link. SRG is shown on the right side. RFR computes the maximum flow on the SRG with current residual capacity for each request pair other than the current connection request. The max-flow on SRG for pair (1, 10) is 5. Figure 4.3 shows two SPF choices for request pair (2, 7), path 2-0-7 or 2-4-7. If the requested capacity were $R_{2,7} = 1$, the routing through path 2-4-7 would result in a reduction of 1 capacity on link [2, 4], leading to a reduction of $f_{1,10}(\widetilde{C}, SRG_{2,7})$ by 1. The normalized reduction is 1/5, but path 2-0-7 reduces the capacity on link [0, 7], which is not a critical link that affects the original max-flow of (1, 10). For pair (1, 10), path 2-0-7 is the preferable route for request (2, 7).

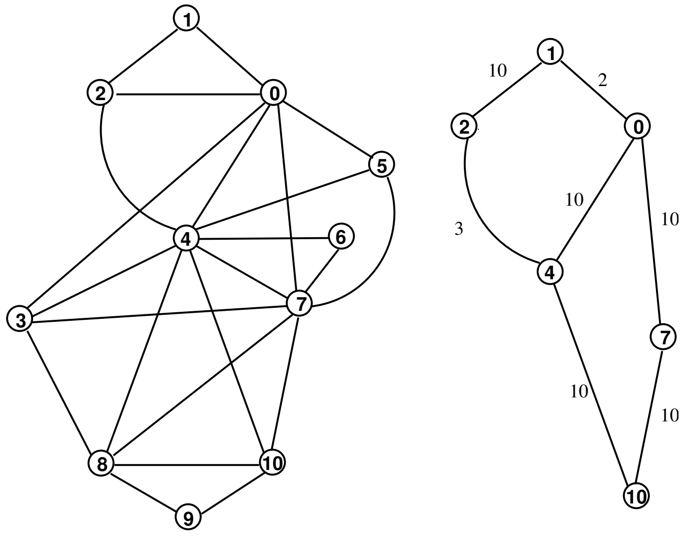


Figure 4.2: NJ LATA and SRG of connection (1, 10).

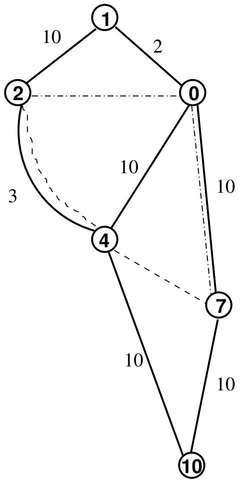


Figure 4.3: Impact of routing connection (2, 7) on the flow of the SRG of connection (1, 10).

If the traffic pattern, like the arrival rate of each pair, is well-known and stable, the computation of cost in the normalized flow could be weighted with each arrival rate. However, I tested both the weighted version (RFRw) and unweighted version, and little difference was found between these two schemes. The unweighted algorithm, presented in Sec.4.1.3, outperforms other online algorithms and occasionally outperforms RFRw, as well. Therefore, my scheme does not necessarily require an anticipated traffic model, which is an essential property for the success of other online algorithms.

4.1.3 Simulation Results

The network is dimensioned using the basic dimensioning algorithm introduced in Algorithm 3.1. The anticipated average link capacity is 120 wavelengths. Each request uniformly demands 1 wavelength channel. The provisioned load is 1. In my experiment, the blocking rate is measured by sampling 5000 arrivals after running 20,000 warm-up requests before the sampling. Each pair has the same arrival rate under the uniform traffic load, it holds a randomly preassigned arrival rate, uniformly selecting from 1 to 10. The average hold time for each pair is determined by the network load. Each experiment will repeat 100 times with different arrivals to obtain the average blocking rate. The sample size I choose is adequate to provide representative results for the system performance aspects I intend to study. I pay special attention to the blocking probability ranging from 0.5% to 8%, because a higher blocking rate indicates network overloading. It also indicates that more than one cut has been formed, which makes it impossible to route a significant portion of requests. Since my goal is to find a better routing algorithm to prevent congestion, any comparison under heavy congestion is undesirable.

Figures 4.4 through 4.7 compare *blocking probabilities* in various online routing algorithms. The horizontal axis represents the actual load that has been normalized by the projected load. MIR, WSP, SWP, and WSP+1 are introduced in Algorithm 2.1. All node pairs in the network are considered as ingress-egress pairs for MIR testing. The α parameter is set to the arrival rate for each pair. SPF uses a random tie-breaking strategy.

The results show that SWP and WSP+1 do not perform as well as other algorithms. MIR presents a higher blocking rate when every pair in the network issues connection requests, compared to when arrival requests are generated by a specific set of predetermined ingress-egress pairs [30]. The result also confirms that selecting paths beyond the shortest available paths, such as what SWP, WSP+1 and MIR do, does not help improving the performance. WSP has a marginally higher blocking rate compared to RFR when the load is low, which increases faster as the network load increases. RFR demonstrates the lowest blocking. RFR has less than 2% blocking compared to WSP and less than 4% compared to SPF when the network is fully loaded. The traffic rate weighted version, RFRw, shows little difference from the nonweighted version, RFR; This means that RFR performs well even without having a priori knowledge of traffic patterns.

Figures 4.4–4.7 also illustrate the *actual network utilization* by offered load. The actual utilization is computed by the sum of average used capacity on each link divided by the total available capacity. The actual utilization for all algorithms exceeds the offered load because the latter is computed using the topological shortest paths length, while real routing scenarios can utilize longer paths when available. The results show that RFR uses the least amount of total capacity compared to other algorithms and provides the best resource balancing for future requests.

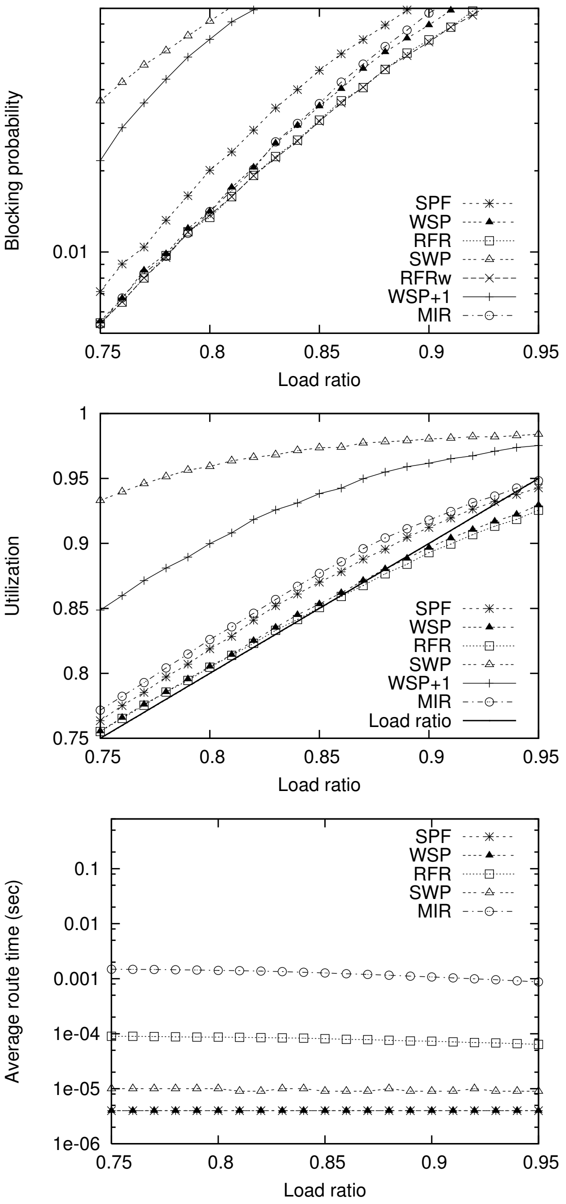


Figure 4.4: Comparison of blocking probability, utilization, and time on NJ LATA.

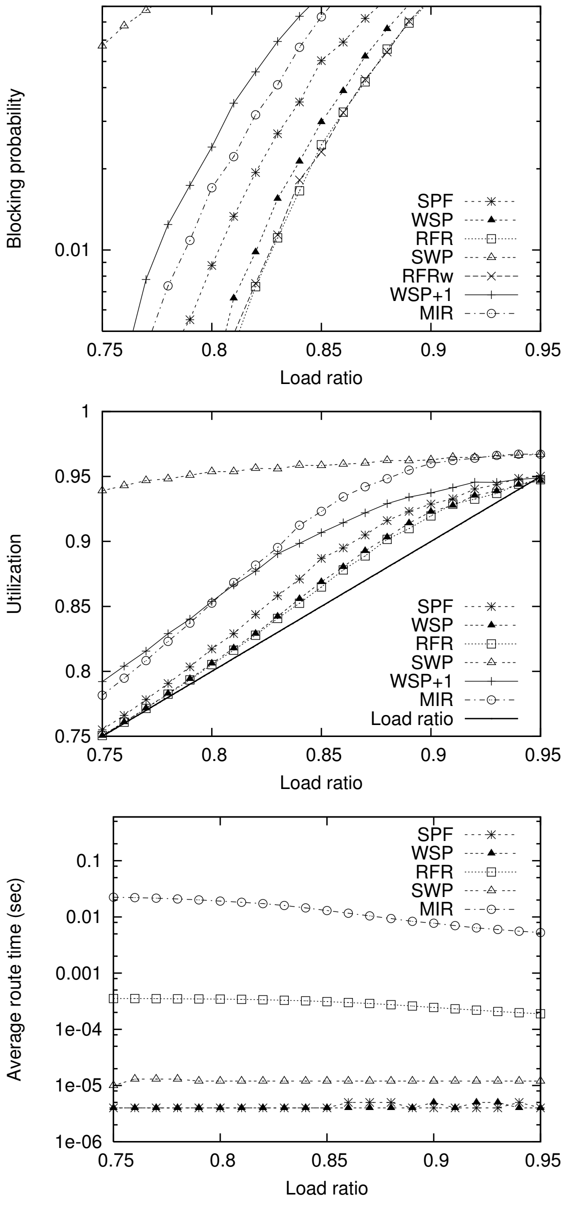


Figure 4.5: Comparison of blocking probability, utilization, and routing time on ARPANET.

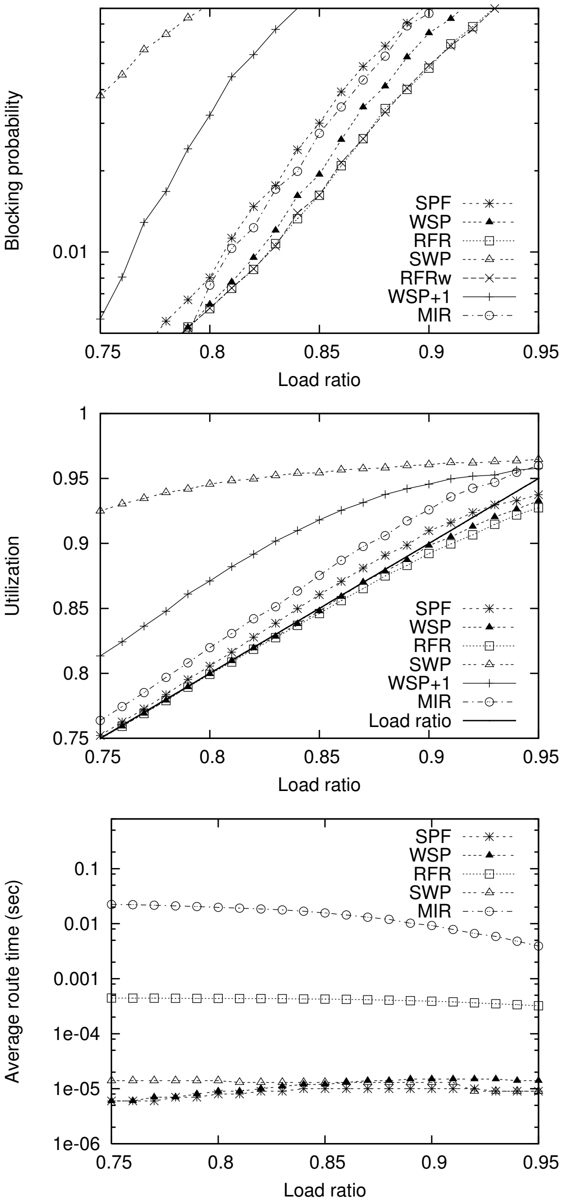


Figure 4.6: Comparison of blocking probability, utilization, and routing time on COST 239.

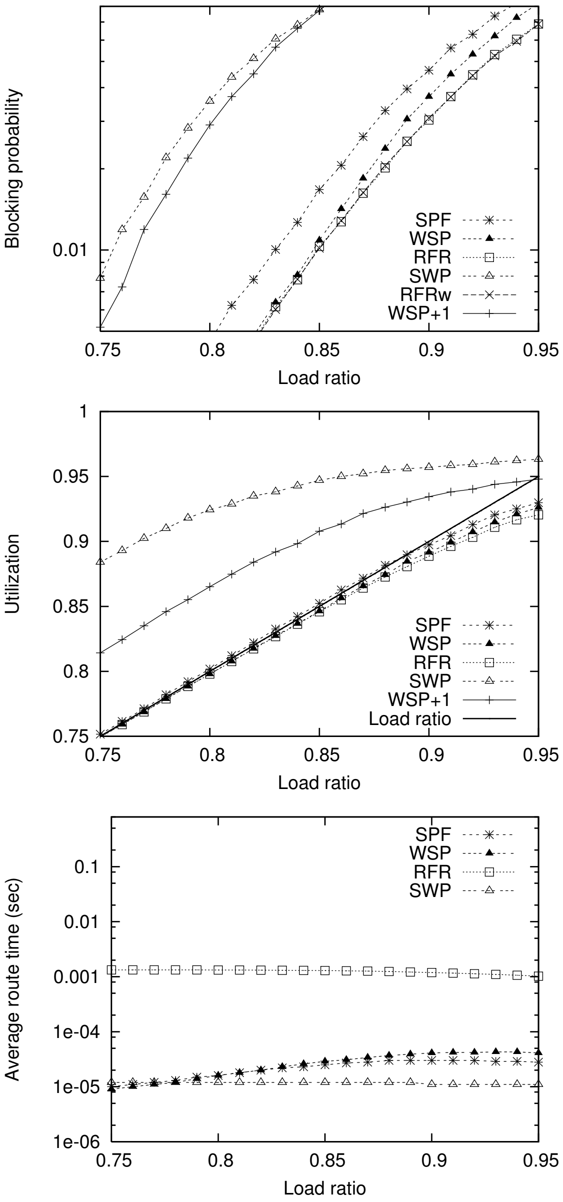


Figure 4.7: Comparison of blocking probability, utilization, and routing time on NATIONAL.

I now compare average *routing times* for SPF, WSP, RFR, SWP and MIR. The relevance of timing depends on real inter-arrival times in the network. If requests arrive on the order of minutes, all five algorithms will be good enough (less than 1 sec to route one request) to meet the deadline. Figures 4.6 and 4.7 show the timing in units of logarithmic scale seconds. For smaller networks, such as NJ LATA, the routing times for SPF and WSP are mostly independent of the traffic load. In larger networks, such as NATIONAL, noticeable increases in routing times are shown because SPF-based algorithms take more time to search for a longer available path, especially when heavily loaded. SWP is affected least by the change in network size. RFR performs 10 times better than MIR. Both RFR and SWP times decrease as the load increases: their routing times are mainly spent choosing among available paths, the number of which decreases when the network becomes congested.

4.2 Opportunity Cost Optimized Congestion Aware Routing

The performance of pure routing mechanisms, such as RFR, is still not ideal when networks are congested. Network congestion increases along with an increase in offered load, causing congestion on many links. Admitting connections with excessive path lengths can increase blocking for future requests. Threshold-based admission control algorithms can respond to network congestion rapidly by controlling the admission of new arrivals. However, these algorithms achieve ideal performance only at the optimal values of their control parameters. These values, identified experimentally in previous studies, are also sensitive to network scaling and topology changes. The cost for simulation can thus be expensive, as it is needed for every change made (such as upgrades) for a large network.

I aim to improve these results in three ways: (1) limit the design to a single control parameter, (2) analytically identify the optimal parameter, and (3) find a consistent optimal parameter value that remains the same for varying network topologies and sizes. To achieve these goals, a good estimation of the network load is needed, which can then be used to relate network congestion to the control parameter. Estimating real-time dynamic network load is challenging because load information is time and location sensitive.

The current network load can evolve from the projected load, and the average load often fails to reflect temporary hot spots created by short-term traffic fluctuations.

This section introduces the design of a threshold-based admission control algorithm that is robust to various load and network scales. The algorithm can be applied to any other online routing algorithms to further reduce blocking. I propose the concept of a “least opportunity cost path,” which provides us with a deeper understanding of dynamic routing algorithms. To approximate the least opportunity cost path using CAR, I develop an opportunity cost model that can analytically identify the optimal threshold parameters for CAR. Simple to compute, robust to different network topologies, and scalable, this model aids in finding a family of improved congestion aware routing algorithms. These algorithms, termed CAR-x, are described in detail in this session. My simulation results show that these improved algorithms are fast to implement and have the most robust thresholds.

4.2.1 Motivation for Limiting Path Length

This section illustrates the relationship between path lengths and network loads for dynamic routed connections. Figure 4.8 presents the distribution of extra hops to the TSL of each connection request among accepted connections routed by ASPF at different offered network loads. When the network is lightly loaded (75%), over 98% of the accepted paths use TSL. At 95% offered load, the percentage of TSL paths drops to 70. TSL is defined on individual connections, so two connections with different path lengths may be grouped into the same TSL+n.

Figure 4.9 shows the notation for constraining hop lengths for connections depending on the traffic load (or the level of congestion). hc-TSL+n denotes a hop-constrained routing in which a connection request is admitted if the available shortest path length is less than or equal to its TSL+n. We see that constraining hop lengths reduces blocking in highly loaded networks but hurts performance when the load is low. Figure 4.9 compares the blocking probabilities of different hop-constraint policies applied on ASPF. I use TSL+n as a hop limit to prevent connection pairs with long TSL from being blocked much more frequently. hc-TSL+n is a similar approach to

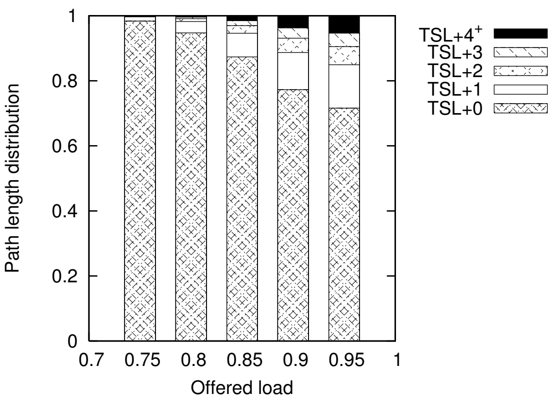


Figure 4.8: Distribution of accepted path lengths on ARPANET.

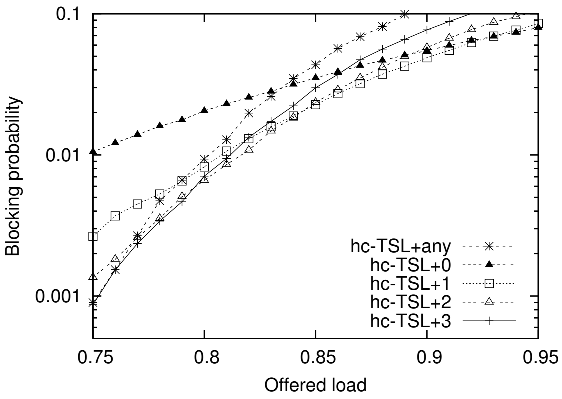


Figure 4.9: ASPF with different hop constraints on ARPANET.

the adaptive length-limit strategy mentioned in [38]. Figure 4.9 shows that constraining path length reduces network congestion when load is high. However, routing without constraint allows more aggressive use of the network and provides the opportunity to admit more connections when load is low. The two extremes are shown at hc-TSL+any and hc-TSL+0. Although it has good low-load performance, hc-TCL+3 provides over 10% blocking at a higher load. Similarly, hc-TCL+1 is attractive at high loads but presents two times more blocking than the best achievable result at low loads. We want to find an adaptive hop-constraint routing policy that determines hop limits dynamically based on the current network state and potentially achieves lower blocking compared to the best nonadaptive routing algorithms across all loads.

Algorithm 4.2: Congestion aware routing (CAR).

```
Input: request pair $r$
Input: current used capacity $U_e, \forall e \in E$
Input: total capacity $C_e, \forall e \in E$
Output: acceptance or rejection
foreach available shortest path p found for a connection generated by request pair r by ASPF routing do
    Compute the sum of congestion ratio;
    $\gamma_p = \frac{1}{|p|} \sum_{e \in p} \frac{U_e}{C_e}$        (4.1)
    /* Path selection metric varies across routing algorithms */
Path $q \leftarrow p$ with min $\gamma_p$ ;        /* Optimality condition (min or max) */
if $|q| = TSL_r$ then /* Always accept a Topological Shortest Path (TSP) */
    Accept the connection
The congestion estimator for path $q$ is $\gamma_q$ ;        /* Different congestion estimators can be used in combination with routing algorithms */
if $\gamma_q > \eta$ then /* Threshold comparison (greater or smaller depending on optimality condition) */
    Reject the connection;
else
    Accept the connection on route $q$;
```

4.2.2 Congestion Aware Routing

Algorithm 4.2 shows the threshold-based congestion estimation and congestion aware routing algorithm (CAR) for each connection r. The degree of congestion on the path of γ_p can be computed by Equation 3. CAR selects a least loaded path from all available shortest paths (other routing algorithms can also be used for this step). If an available path is found, and it has the same length as the TSL of that node pair, we automatically accept it because it does not use any extra resources (links along the paths with high TSL have been planned with more resources when the network is dimensioned, so the routing algorithm should not bias against them). If not, we accept it only if the congestion estimation value γ_p is smaller than a predetermined threshold η. In other words, we accept a request either by no-hop constraint or maximum-hop constraint of TSL+0, depending on the estimated level of congestion. Usually, when a network starts to use longer paths to accept new connections due to traffic load increases, local congestion propagates to the rest of the network quickly. To suppress the spread of congestion, strict admission control should be used on detection of local congestion. That is, when links are highly loaded in a neighborhood, no paths of extra length should be admitted. I have tried also to estimate congestion by looking at different local subnets. But using the path alone already gives the best result.

Algorithms with similar form appeared in earlier work [31, 33]. The difference is that CAR chooses paths exclusively from ASPFs and accepts TSP regardless of the load. The benefit of choosing ASPFs in a dynamic routing environment was discussed in Section 4.1.1. Basically, assigning a longer path when a shorter path is available increases the opportunity cost for the network to accept future arrivals. TSP is always accepted because the connection pair does not use any extra resource than what was planned. Also, the links along the paths of a greater TSL have been planned with more resources when the network is dimensioned. The routing algorithm, then, should not be biased against these paths.

4.2.3 Opportunity Cost Model

I can evaluate dynamic routing algorithms according to the number of blocked requests for a given number of arrivals. The path choice for an accepted request affects the blocking of future arrivals. I define the *opportunity cost* for a path as the expected difference in the number of future blocked arrivals, for an infinite time span, due to the acceptance of a request using the path. The opportunity cost represents the expected future loss for an immediate gain. The *least opportunity cost path* for a request is a path with the minimum opportunity cost. If the least opportunity cost is greater than one, accepting the request is expected to cause more than one future connection to block (beyond those that will block anyway if the current request is rejected). In other words, the initial benefit of accepting the request is eventually lost by the rejection of more than one future request.

The opportunity cost depends on the policies governing routing and admission control. This dependence arises because the two policies may differ in their future decisions. However, calculating the opportunity cost is not always possible. Some policies are sample-path dependent and thus cannot be modeled by memoryless state-transition graphs. For those that can be calculated, computing the opportunity cost is expensive. The number of states in the network for future arrivals/departures grows exponentially as network size increases. The transitions defined by the policy do not necessarily converge. For simulation approaches, possible arrival sample paths also grow exponentially. The difference in blocking for a sample path can be infinite (a two-link case is explained in Section 4.2.3).

However, in practice, good approximations of the least opportunity cost path can be efficiently computed since traffic loads and path length have a strong link to opportunity cost. When the network load is low, the opportunity cost is small because the path length is minimal and the network has enough extra resources to support other connections. The opportunity cost increases when the network load is high and many links have little spare capacity. The available path becomes longer since some links become unavailable.

All existing dynamic routing algorithms can be viewed as approximations of the least opportunity cost path. ASPF assumes that using more links to route a path increases the chance of blocking future connections. WSP assumes that depleting the little remaining available capacity on a link increases the chance of blocking exponentially. Adaptive WSP leverages the result from queuing theory: the blocking probability distribution varies for the total capacity. Flow-based approaches approximate the opportunity cost by calculating the reduced flows for expected future requests. RFR recognizes the importance of locality to improve generic flow-based heuristics. However, the performance of these algorithms degrades when network load increases. Better routing algorithms are needed to optimize the performance for both low and high network loads.

This section uses simple models to approximate the opportunity cost for CAR algorithms and to derive the optimal threshold value. CAR can thus efficiently approximate least opportunity cost routing. I start with a one-link model, using an arbitrary link in a network, with the assumption that all links have independent traffic. Then, a two-link independent model is built on top of the one-link model. It is useful in practice to compute the optimal threshold value and aids in the design of better routing algorithms.

One-Link Model

My simplified one-link model uses a random link in a network with the assumption that all links have independent traffic. I define $t = 0$ to be the decision time when an arrival requests a unit of capacity from the link for a path that is longer than the TSL. Let the maximum capacity for a link be

c. The aggregated arrival rate to this link is λ, and the departure rate for each connection is μ. Let the offered link load be l. These model parameters are scaled to the real network parameters in that the admission control of future arrivals is already incorporated in the parameters. Therefore, all future arrivals to the link are accepted if there is link capacity available. I do not need to track (and make decisions for) future arrivals that use this link as part of the extra hops. Such simplification does not affect the accuracy of my model since I do not rely on global load estimation for congestion control.

When the network has been dimensioned (according to Chapter 3) and each arrival requests one unit capacity, the load follows Equation 4.2. If the link is loaded with i wavelengths at $t = 0$, where $i < c$, the expected amount of future blocking in an arbitrary time interval $(0, t)$ can be formulated by Equation 4.3, where $P_{c,i}(s)$ is the probability that this link is full at time s given total capacity c and starting capacity i. As the time interval increases, $t \to \infty$, and the expected blocking is nondecreasing. Therefore, Equation 4.3 is not bounded.

$$\frac{\lambda}{\mu} = c \cdot l \tag{4.2}$$

$$\int_0^t P_{c,i}(s)\lambda ds \tag{4.3}$$

Regarding the decision on whether to accept or reject the request at time 0, I want to estimate the difference in future blocking if I choose to use this link (accept the request) or not. Given the two choices, I then have two different initial link states. If I decide to take the extra request, the initial capacity state is $i + 1$. If not, the initial state is still i. The expected difference in blocking of two choices, i.e., the opportunity cost, is then $\int_0^t P_{c,i+1}(s)\lambda ds - \int_0^t P_{c,i}(s)\lambda ds$. Since future arrivals are always accepted if there is free capacity, the difference of blocking probability, $P_{c,i+1}(s) - P_{c,i}(s)$, decreases exponentially. Therefore, the opportunity cost will converge as $t \to \infty$ to a finite number, which is denoted as d_i^c in Equation 4.4.

$$d_i^c = \lim_{t\to\infty} \left(\int_0^t P_{c,i+1}(s)\lambda ds - \int_0^t P_{c,i}(s)\lambda ds \right) \tag{4.4}$$

By viewing the problem slightly differently, I show that d_i^c ranges from 0 to 1 for any starting state $i < c$ and capacity c. The initial capacity is i for the case for which the extra-hop arrival is rejected at time 0, and the

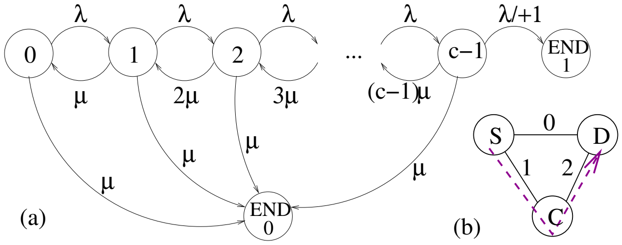


Figure 4.10: (a) Markov chain for one-link model. (b) Two extra links with independent traffic.

initial capacity for the accepting case is $i+1$. Let the link state (wavelengths in use) at time s be $L_i(s)$ with initial state i. $L_{i+1}(s)$ is the link state for the case for which the arrival is accepted. The sequences of further arrivals are the same for both decision branches, and they will be admitted as long as there is unused capacity. Therefore, the number of wavelengths in use for the decision branch L_{i+1} always exceeds the branch of L_i by 1 until the extra link request leaves L_{i+1} or L_i is full. L_i becomes full at exactly the same time as L_{i+1} must reject the first future arrival. The two branches then have equal capacity onwards. Otherwise, the extra-hop arrival in L_{i+1} leaves before L_i becomes full. The expected future blocking probability for two branches is equal after they reach equal capacity, as all connections have equal exponential holding time distributions. I use the term END to describe the state where the two branches reach equal capacity. If an END is caused by the case for which L_{i+1} rejected a request (equivalently, L_i gets full), the difference of blocking probability is 1. Such an END is denoted as *END1*. If an END is caused by the departure of the extra-hop arrival before L_{i+1} gets the first reject, there is zero difference in blocking probability. The END is then denoted *END0*. Equation 4.4 shows the probability that the system ends with END1 given the total capacity and initial state. The probability equals one if the link is already fully loaded, i.e., $i+1=c$. Therefore, the opportunity cost is a binary random variable with expected value d_i^c.

Computing d_i^c directly from $P_{c,i}$ is difficult because $P_{c,i}$ is unbounded. However, I can use a continuous Markov chain to compute the expectation. The transition of states is shown in Figure 4.10(a). The number in each state is the current amount of used link capacity for the case of L_i (L_{i+1} always equals L_i+1). The END1 state is equivalent to the state where L_i is full (also L_{i+1} rejects the first request). As the initial link capacity i ranges from

0 to $c-1$, the system can start at any state except the END states. Again, all connections request one unit capacity, the arrival rate is λ, and the departure rate on the link is $L_i\mu$ as each connection departs at rate μ. From the L_{i+1} state, the extra-hop arrival at $t=0$ may also depart at rate μ. If that happens, the system immediately enters END0. Since $\lambda > 0$ and $\mu > 0$, the system will eventually reach one of the two sink states, END1 or END0. d_i^c from Equation 4.4 can be computed by finding the probability that the system ends at END1 when starting at state i. Trivially, $d_{END1}^c = 1$ and $d_{END0}^c = 0$. The probability of the ending state is computed recursively by Equations 4.5–4.7. These equations balance the in/out probability flows at each state. [1]

$$d_0^c = d_1^c \frac{\lambda}{\lambda+\mu} \tag{4.5}$$

$$d_{c-1}^c = \frac{\lambda}{\lambda + c\mu} + d_{c-2}^c \frac{(c-1)\mu}{\lambda + c\mu} \tag{4.6}$$

For $0 < i < c-1$,

$$d_i^c = d_{i+1}^c \frac{\lambda}{\lambda + (i+1)\mu} + d_{i-1}^c \frac{i\mu}{\lambda + (i+1)\mu} \tag{4.7}$$

Solving these equations by mathematical induction, I get the general form for d_i^c.

$$d_i^c = \frac{\sum_{k=0}^{i} \frac{i!}{(i-k)!}\left(\frac{\mu}{\lambda}\right)^k}{\sum_{k=0}^{c} \frac{c!}{(c-k)!}\left(\frac{\mu}{\lambda}\right)^k} \tag{4.8}$$

Combined with Equation 4.2 to cancel λ/μ, I have the opportunity cost for one link:

$$d_i^{c,l} = \frac{\sum_{k=0}^{i} \frac{i!}{(i-k)!(cl)^k}}{\sum_{k=0}^{c} \frac{c!}{(c-k)!(cl)^k}} \tag{4.9}$$

Equation 4.9 shows that the opportunity cost in future blocking probability on a link depends on total capacity c, initial capacity i, and traffic load l. Figure 4.11 shows how $d_i^{c,l}$ varies with l and i/c when $c = 100$.

Two-Link Independent Model

The one-link model is not yet useful because the opportunity cost is always less than one. I consider two links to find the condition for which the opportunity cost can exceed one. When a TSL path is not available, an online routing algorithm may need two or more additional links to route around

[1] The detailed steps of solving these equations are not shown.

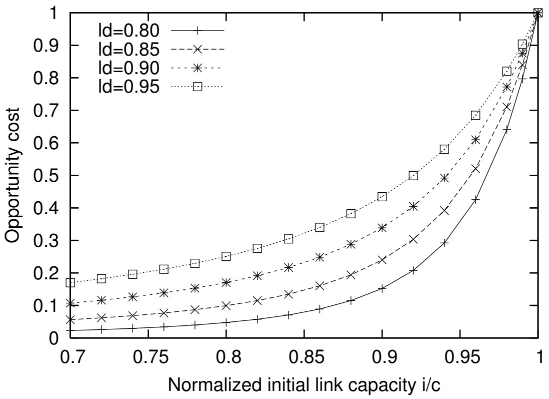


Figure 4.11: The one-link model with various loads at $c = 100$.

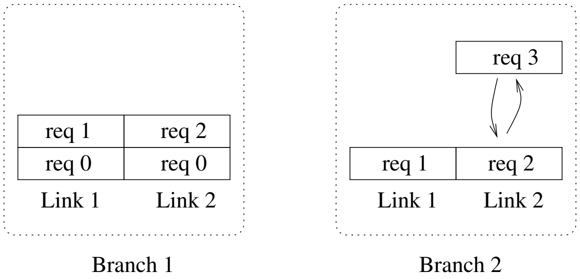


Figure 4.12: An illustration of unbounded expected difference of blocking for two links.

unavailable links. TSL+n means that a path length is equal to its TSL plus n extra hops. Figure 4.10(b) shows an example. Path p1 is the TSL+0 path for the connection S to D. p2 is a TSL+1 path when the direct link S-D is full, and p3 is a TSL+2 path that uses two additional links to route around when links S-D and S-B are unavailable. This section develops a model of two extra links with an independent traffic assumption. The next section discusses the case in which joint traffic is presented across two links. However, the independent traffic model reveals important information that allows us to predict the optimal CAR threshold.

In the two-link case, the opportunity cost is not bounded by one. Figure 4.12 illustrates an unbounded situation. Suppose there are two links, each with capacity $c = 2$. Request 0 is the initial extra two-link request. Later arrivals, Request 1 and Request 2, use the remaining capacity. Future arrivals, such as Request 3, arriving on Link 2 before other connections leave, can increase the number of blocking probability differences between the Request 0 accepted and rejected cases to an arbitrary number. However,

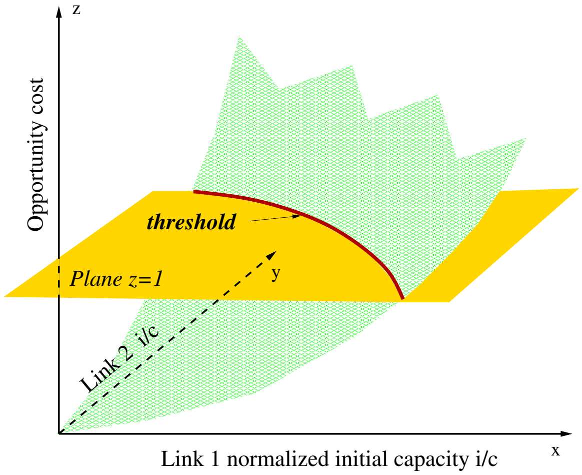


Figure 4.13: Threshold prediction using the two-link independent model.

the probability that an infinite number of arrival requests during the finite holding-time of the current connections is sufficiently small that the expected value remains finite.

Assuming that each link behaves independently, the opportunity cost is equivalent to the sum of $d_i^{c,l}$ for each one-link mode. I can construct a three-dimensional plot with the x- and y-axis representing the normalized initial capacity of each link. Figure 4.13 illustrates the construction. The z-axis is the opportunity cost computed by adding the results of two one-link models to the corresponding traffic load and link capacity. If the sum is greater than 1, accepting the connection at $t = 0$ is expected to cause the link to reject more than one future connection. The threshold curve is shown in Figure 4.13 on the $z = 1$ plane. In this case, the initial request should be rejected. Equation 4.10 defines the contour curve, in set theory, of the 3D graph at load l on plane $z = 1$. Figure 4.14 shows the curves (with legends "mod ld=l") at various loads; these curves show the ideal threshold at load l. The lower the load, the higher the threshold can be to allow aggressive admission. The threshold is lower at higher loads because fewer resources are available for routing excessively long paths. At routing time, based on the current residual graph, we know the operating point on the plane but not the curve that should be used, since I may not know the offered load l. Therefore, the threshold used at routing time must work for all loads.

$$\{(x,y)|d_{xc_1}^{c_1,l} + d_{yc_2}^{c_2,l} = 1, 0 < x < 1, 0 < y < 1\} \tag{4.10}$$

Two-Link Dependent Model

In general there are three types of traffic on two links: traffic using only Link 1, traffic using only Link 2, and traffic using both links. A two-dimensional Markov process suffices to represent the new ending situation but does not terminate until accept/reject cases are loaded with the same amount of two-link traffic and the same amount of one-link traffic on each link. For example, if Link 1 of the accept case carries 50 one-link connections and 20 two-link connections, Link 1 of the initial rejection case should also have 20 two-link connections and 50 one-link connections. Other combinations that occupy the same total resources may diverge due to subsequent events. Unfortunately, the complexity of the 2-D Markov chain grows quickly: the Markov chain of two links of capacity 2 contains 42 states and 203 transitions. So I use simulation to solve this problem. Figure 4.15 shows that the $z = 1$ contour line of the simulation results for various traffic dependency at offered load 0.9. Traffic dependency dp is the fraction of two-link traffic amongst all arrivals. When $dp = 1$, all connections use both links, and the model reduces to a one-link model. When $dp = 0$, only the initial request uses both links, and the model is equivalent to the two-link independent model. The only joint event is the departure of the initial request. The result shows that the threshold drifts upwards in x and y when traffic dependency increases, becoming a point at (1, 1) for $dp = 1$.

In real dynamic mesh networks, the traffic dependency between two links is difficult to track. The flow pattern on the links depends on not only the original dimensioned traffic but also on the usage from other connections as alternative routes. I see that the threshold computed with the independent assumption could be underestimated in a network with high traffic correlation. However, the threshold estimated by the independent model is good enough on typical topologies because long chains of major nodes (which might create dependent traffic) are avoided in backbone network design.

Optimal Threshold Estimation for CAR

Each ideal threshold curve in the 2D graphs is optimal for a fixed and homogeneous global traffic load l. In practice, however, I do not know the exact offered traffic load at the area where the route passes through. It is also expensive to identify the "extra" links during routing. As a simple approach, CAR approximates the opportunity cost by averaging the load of all links. Essentially, it is a linear approximation to the 2D curves. CAR

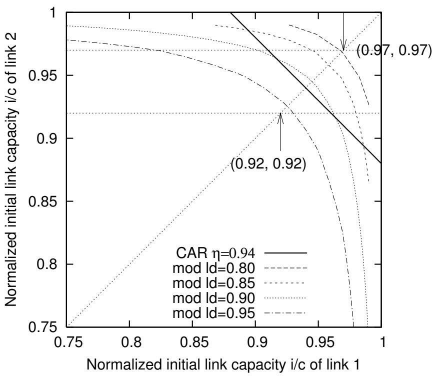


Figure 4.14: Two links with independent traffic ($dp = 0$). $c_1 = c_2 = 100$.

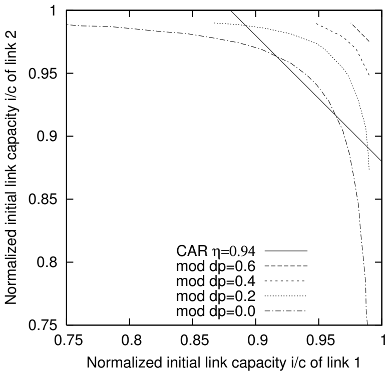


Figure 4.15: Two links with various dependent traffic. $c_1 = c_2 = 100, ld = 0.9$.

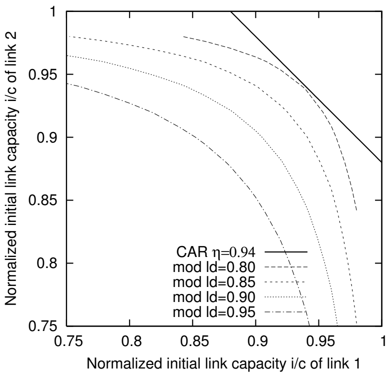


Figure 4.16: Two links with independent traffic ($dp = 0$). $c_1 = c_2 = 50$.

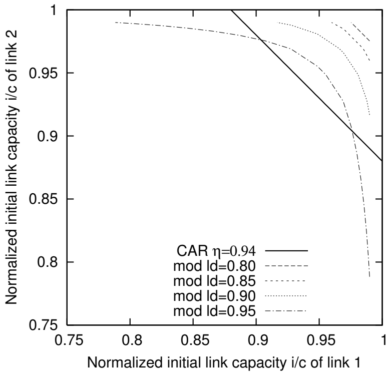


Figure 4.17: Two links with independent traffic ($dp = 0$). $c_1 = c_2 = 200$.

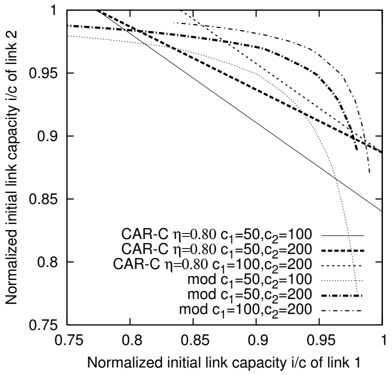


Figure 4.18: Two links with independent traffic and different capacity. $ld = 0.9$.

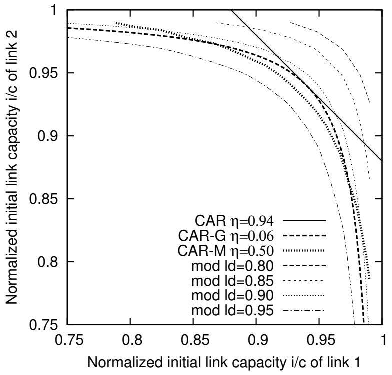


Figure 4.19: Two links with independent traffic ($dp = 0$) and various routing algorithms. $c_1 = c_2 = 100$.

with threshold η can be represented as a line $\frac{1}{2}(x + y) = \eta$ for $\eta = 0.94$ in Figure 4.14.

To identify the optimal η, I assume that the residual capacities on the two links are the same and network offered load fluctuates from 0.8 to 0.95. I draw a line $x = y$ across the plane in Figure 4.14. Then, the contour of load 0.80 intersects the line around (0.97, 0.97), and the contour of load 0.95 intersects it around (0.92, 0.92). Contours for loads less than 0.80 are hardly visible on the plane: admission control is rarely needed on a lightly loaded network. Therefore, the threshold should be a value in the range of [0.92, 0.97] if the network is operating around offered load 0.75 to 0.95. The average threshold is about 0.94. The simulation results in Section 4.2.5 verify the prediction here. The model also indicates that the threshold should increase when the network scales, as shown in Figures 4.16 and 4.17. This prediction is again verified in the simulation results.

The model can analyze two links with different capacities. Figure 4.18 shows an example with link-independent traffic at load 0.9, link capacity $c_1 = 50$ and $c_2 = 100$. The ideal threshold declines when moving close to the axis of the link of smaller total capacity. CAR cannot catch such movement. Other CAR-like algorithms that provide better approximation are introduced in the next section.

4.2.4 Improved Routing and Congestion Estimation

The two-link model inspires better routing algorithms. Figures 4.14–4.17 show the ideal curves a routing algorithm should achieve. The original CAR algorithm is only a linear approximation of the curve. Therefore, I present these better algorithms by fitting the curves: CAR-G, CAR-C, and CAR-M.

All improved algorithms share most of the steps in Algorithm 4.2 except Line 3 (computation of γ_p), Line 4 (picking minimum or maximum γ_p), and Line 8 (comparison criteria). The changes are summarized in Table 4.2.

CAR-G uses a geometric mean of the link availability ratio to compute γ_p in Equation 4.11, Line 3 of Algorithm 4.2. Since it uses the available capacity ratio, the η^G is a maximum threshold. The optimized threshold for CAR-G, 0.06, translates to the threshold equation $((1 - x)(1 - y))^{1/2} = 0.06$. The curve cuts the line of the CAR threshold at 0.94 in Figure 4.19 and shows the best fitting shape of the ideal threshold curves.

CAR-C uses Equation 4.12, which normalizes available capacity by the statistical deviation of the total capacity. Because the total capacity is determined by 100% of the traffic load, the square root of the total capacity approximates the expected standard deviation of dimensioned resources. If the available capacity were smaller than the deviation of the total capacity, transient traffic fluctuation would be more likely to overload the link. Figure 4.18 shows the threshold equation $\frac{1}{2}(\sqrt{c_1}(1-x)+\sqrt{c_2}(1-y)) = 0.8$ for link capacity c_1 and c_2 at the optimal threshold 0.8. CAR-C lines can automatically adjust to different total capacities on each link to match the trend of the ideal curves from the model. The range of optimal threshold values for CAR-C is larger than for CAR since the denominator (square root of the total capacity) in each term is smaller.

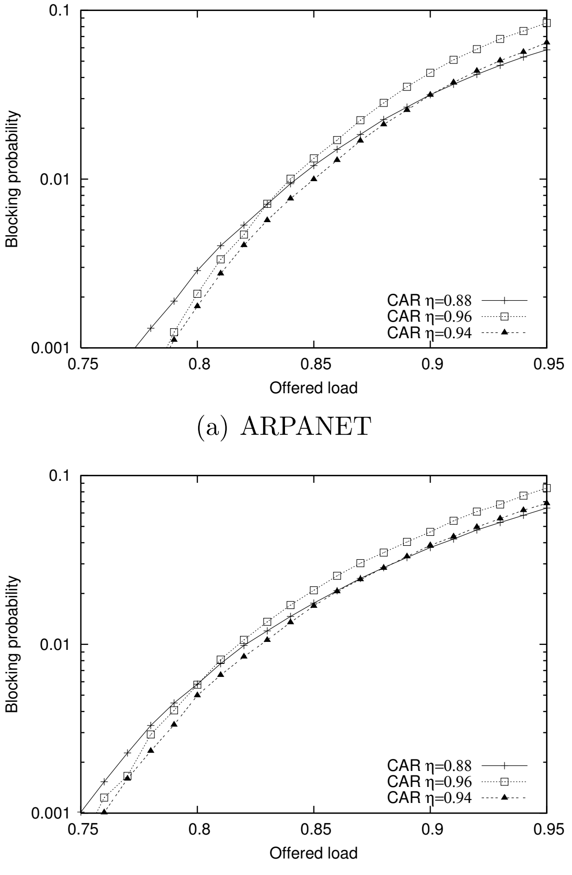
CAR-M is created from the two-link independent model. Equation 4.13 computes the average of the opportunity cost $d_i^{c,l}$ on each link with the estimated load $l = U_e/C_e$, capacity $c = C_e$, and initial capacity $i = U_e$. The optimized simulation threshold $\eta^M = 0.5$ corresponds to the curve $\{(x,y)|\frac{1}{2}(d_{xc_1}^{c_1,x} + d_{yc_2}^{c_2,y}) = 0.5, 0 < x < 1, 0 < y < 1\}$, which is shown in Figure 4.19. Since the computation of γ_p takes the total capacity into account, the threshold is adaptive to capacity scaling.

4.2.5 Simulation Results

Simulation results show that the CAR-x family requires little computation, improves call blocking across a wide range of traffic loads, and provides blocking probabilities that are less dependent on path lengths than those of previous algorithms. The thresholds of CAR-x algorithms can be identified analytically and can thus be robust to changes in topology, capacity, and other factors. A comparison of CAR-x algorithms is shown in Table 4.3, which shows that CAR-M has the most robust threshold with little additional routing time.

Performance on Dimensioned Backbone Mesh Networks

The analytically optimized threshold value for CAR applies to major backbone mesh networks. Figure 4.20 shows that the optimal simulation threshold



(a) ARPANET

(b) NSF

Figure 4.20: Optimal threshold parameter of CAR on major topologies.

Table 4.2: CAR-x algorithm family.

	Line 3	**Line 4**	**Line 8**
CAR	$\gamma_p = \frac{1}{\lvert p \rvert} \sum_{e \in p} \frac{U_e}{C_e}$	$\min \gamma_p$	$\gamma_q > \eta$
CAR-G	$\gamma_p = \left(\prod_{e \in p} \frac{V_e}{C_e} \right)^{1/\lvert p \rvert} \quad (4.11)$	$\max \gamma_p$	$\gamma_q < \eta^G$
CAR-C	$\gamma_p = \frac{1}{\lvert p \rvert} \sum_{e \in p} \frac{V_e}{\sqrt{C_e}} \quad (4.12)$	$\max \gamma_p$	$\gamma_q < \eta^C$
CAR-M	$\gamma_p = \frac{1}{\lvert p \rvert} \sum_{e \in p} d_{U_e}^{C_e, \frac{U_e}{C_e}} \quad (4.13)$	$\min \gamma_p$	$\gamma_q > \eta^M$

for CAR—i.e., the threshold that provides the best improvement in blocking at all loads—falls within a small range around 0.94. Formally proving that an optimal threshold works for all backbone topologies is difficult, as the analytical approach is NP-hard due to inter-path dependence. My results show that the analytical threshold works for many practical networks. More examples are shown in [39].

The algorithms in the CAR-x family perform similarly at their best thresholds in Figures 4.21 and 4.22, also showing that threshold selection can be performed independent of the topology. The CAR-x family is compared to pure route-selection algorithms—ASPF, WSP, and RFR—and previous threshold-based congestion aware routing algorithms—COL and SDR—that are discussed in detail in Section 4.2.6. Clearly, threshold-based routing algorithms outperform pure route-selection algorithms by blocking connections that use excessive resources.

Figures 4.23 and 4.24 show the stability of thresholds in CAR-C and CAR-M under capacity scaling. ARPANET is dimensioned at average link

Table 4.3: Performance comparison of all algorithms.

	CAR	**CAR-G**	**CAR-C**	**CAR-M**	**SDR-orig** [31]	**SDR**	**COL-orig** [33]	**COL**
Lowest blocking (comparable)	yes	yes	yes	yes	no	yes	yes	yes
Threshold robust across major topologies	yes	yes	yes	yes	N/A	no	no	yes
Threshold robust to capacity scaling	no	no	yes	yes	N/A	no	no	no
Routing time relative to ASPF	$\approx 1.5\times$	$\approx 1.5\times$	$\approx 1.5\times$	$\approx 1.5\times$	$> 30\times$	$> 30\times$	$\approx 2\times$	$\approx 2\times$

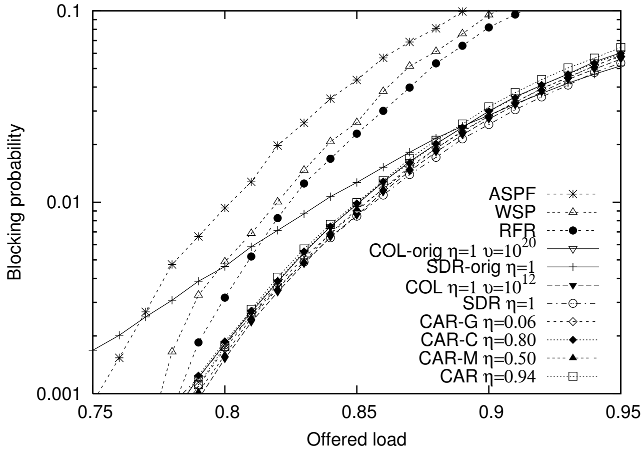


Figure 4.21: Comparison of routing algorithms with optimal parameters on ARPANET. SDR-orig does not have competitive blocking.

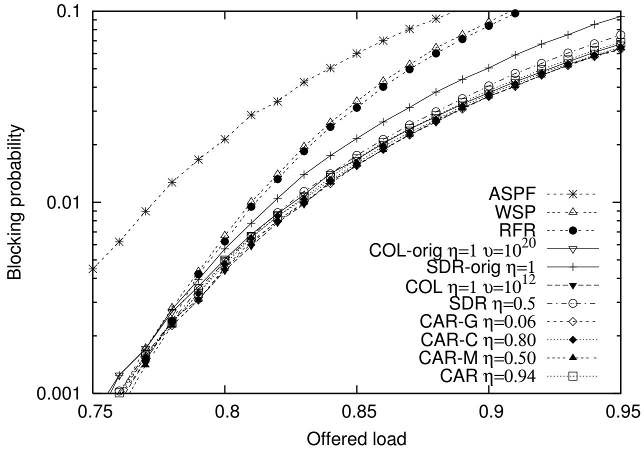


Figure 4.22: Comparison of routing algorithms with optimal parameters on NSF. SDR-orig does not have competitive blocking. The optimal parameter for SDR is changed from ARPANET.

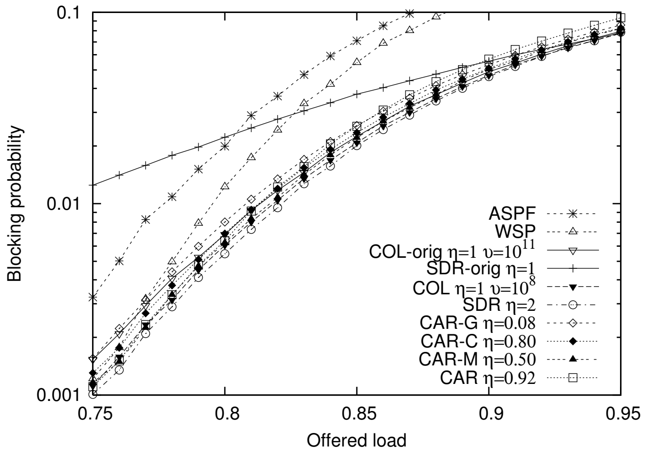


Figure 4.23: Comparison of routing algorithms with optimal parameters on ARPANET dimensioned with average link capacity 60. The optimal parameters for COL-orig, COL, SDR, CAR, CAR-G are changed, relative to capacity 120.

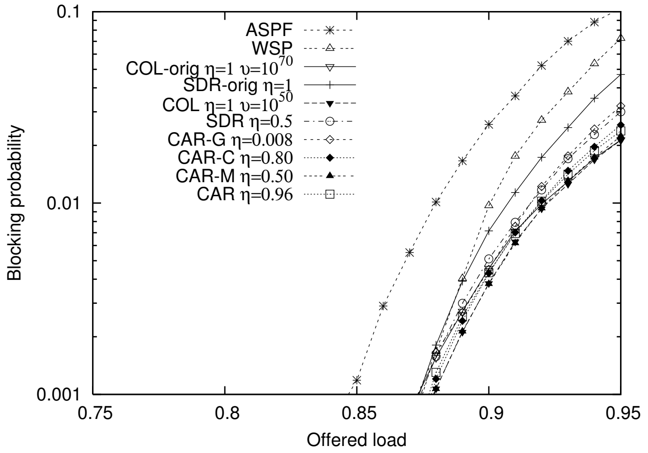


Figure 4.24: Comparison of routing algorithms with optimal parameters on ARPANET dimensioned with average link capacity 600. The optimal parameters for COL-orig, COL, SDR, CAR, CAR-G are changed, relative to capacity 120.

capacity 60 (1/2 of 120, the reference scale) and 600 (5 times 120) respectively. The best threshold values for CAR and CAR-G drift down as average link capacity declines from 120 to 60, which has been predicted by my model in Figures 4.16 and 4.17. On the other hand, the same threshold values hold for CAR-C and CAR-M since they can automatically adjust the curve at different total link capacities.

Comparison of the routing time per connection request is shown in Table 4.3 as a multiplier to the routing time of ASPF. In fact, CAR-x algorithms are more than 10 times faster than RFR, one of the fastest flow-based routing algorithms. These results were obtained from a Linux desktop with a 3.20GHz P4 CPU. CAR (CAR-G/C are similar) and CAR-M are within 10 microseconds of ASPF for the same network. CAR-M is implemented with a fast lookup table that has all $d_i^{c,l}$ values precomputed for each link.

In addition to CAR approaches, I have tried to estimate the least opportunity cost path directly. I warm up the network with 20,000 arrivals using CAR-M. For the 5000 arrivals during the test window, I compute the average difference of blocking for 1000 sample paths, each with 20 future arrivals. These future arrivals are routed by CAR-M, as well. As a result, the per request routing time is about 60-200 times that of CAR-M, but the blocking probability is still 10% higher than CAR-M at the load range. Given our computing power, I do not expect to get better opportunity cost heuristics by simulating more sample points in the state spaces. I conclude that the direct approach is impractical for real networks given the computational cost for each request.

Performance on Misdimensioned Networks

Robustness to misdimensioning is an important aspect of performance for routing algorithms. Although all routing algorithms work poorly on an undimensioned network, how fast they degrade on partially misdimensioned networks is as important as the study on dimensioned networks. This section shows that CAR-G and CAR-M perform better than CAR and CAR-C on misdimensioned networks.

Partial network misdimensioning can happen for many reasons. One possible cause is that the network provider decides to upgrade the capacity for some links but not for others. Some routing algorithms may receive benefits from such upgrades, e.g. WSP. However, the upgrade can unbalance the network and negatively affect other algorithms. If the routing algorithm uses load information for upgraded links, these links appear less loaded than other links and attract more traffic to them. In this case, their surrounding links without upgrades can overload. To illuminate such effects, I create a misdimensioned network by doubling the total capacity of link $(7, 10)$ on ARPANET (Figure 3.8). The rest of the network remains the same. All algorithms are compared in Figure 4.25. The results of WSP and CAR-M on dimensioned ARPANET are also shown for comparison. WSP, CAR-G, and CAR-M show the same performance for all loads on misdimensioned ARPANET. The blocking probability of CAR and CAR-C increase by about 16% at high loads: CAR and CAR-C use a linear sum of link load ratio for all links in the path. When Link (a, b) presents an exceptionally low load, the overall estimated congestion level becomes much smaller than the real network congestion. In contrast, CAR-G and CAR-M are robust because higher loaded links have higher weights than lighter loaded ones in a geometric sum or the sum of opportunity costs. The one-link opportunity cost increases exponentially to the traffic load, as I show in Figure 4.11.

I also study the performance on randomly misdimensioned networks. Each link of ARPANET is re-dimensioned randomly in a uniform range from $\max(\lfloor C_e(1-\alpha)\rfloor, 0)$ to $\lceil C_e(1+\alpha)\rceil$, where C_e is the dimensioned capacity shown in Figure 3.8. The level of misdimensioning is defined by a real number, $\alpha \in [0, 1]$, called the *misdimensioning factor*. The network becomes more misdimensioned as α increases. For a given α, I average the results on different misdimensioned instances until the 95% confidence interval is within ±5% of the data values. Figures 4.26 and 4.27 show that CAR-G and CAR-M degrade more slowly than other CAR-x algorithms. Since the network is misdimensioned on a much larger scale compared to the previous setup, the average blocking is always worse than it is on the dimensioned network.

4.2.6 Comparison to Previous Work

I compare CAR-x to two previous threshold-based approaches, SDR and COL, described in Algorithm 2.1. I use both the original SDR and COL algorithms (denoted as SDR-orig and COL-orig, respectively) as well as my improved versions (SDR and COL). SDR-orig uses only a fixed threshold

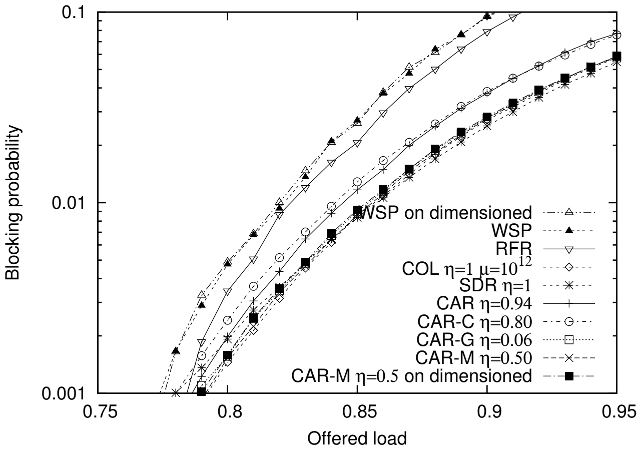


Figure 4.25: Comparison of routing algorithms on misdimensioned ARPANET with total capacity of link $(7, 10)$ doubled.

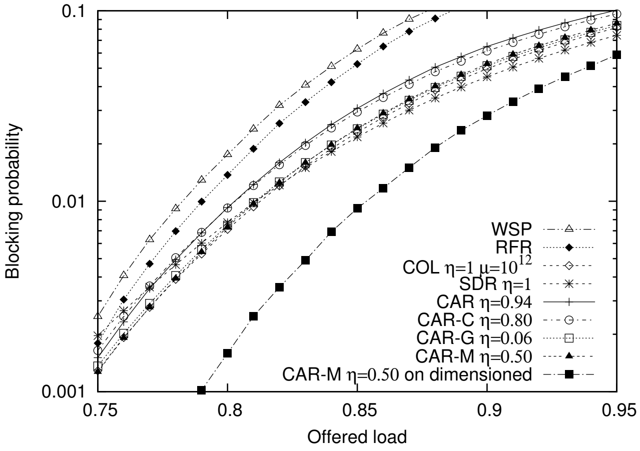


Figure 4.26: Comparison of routing algorithms on randomly misdimensioned ARPANET with $\alpha = 0.2$.

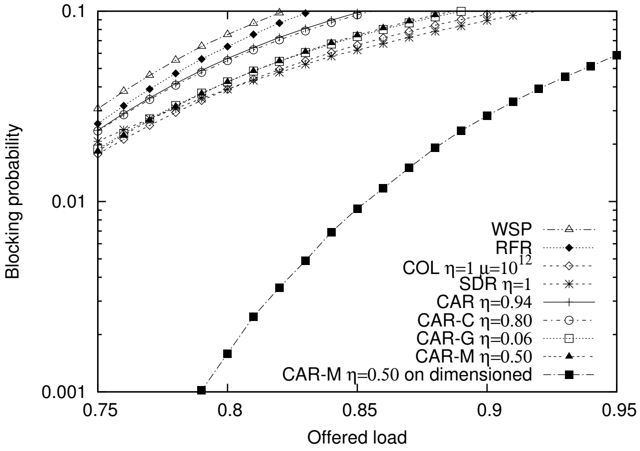


Figure 4.27: Comparison of routing algorithms on randomly misdimensioned ARPANET with $\alpha = 0.4$.

Table 4.4: SDR and COL algorithms.

	Line 3	Line 4	Lines 5-6	Line 8
SDR-orig [31]	$\gamma_p = \sum_{e \in p} \frac{B(C_e, l_e)}{B(U_e, l_e)} \quad (4.14)$	$\min \gamma_p$	remove	$\gamma_q < \eta = 1$
SDR	$\gamma_p = \sum_{e \in p} \frac{B(C_e, l_e)}{B(U_e, l_e)}$	$\min \gamma_p$	no change	$\gamma_q < \eta$
COL-orig [33]	$\gamma_p = \sum_{e \in p} \upsilon^{-\frac{V_e}{C_e}} \quad (4.15)$	$\min \gamma_p$	remove	$\gamma_q < \eta = 1$
COL	$\gamma_p = \sum_{e \in p} \upsilon^{-\frac{V_e}{C_e}}$	$\min \gamma_p$	no change	$\gamma_q < \eta = 1$

$\eta = 1$. My version always accepts TSL paths and tries to optimize the threshold value for the best blocking performance. Table 4.4 summarizes the changes to the CAR algorithm to implement SDR-orig, SDR, COL-orig, and COL.

The comparisons with CAR-M on the dimensioned topologies are shown in Figures 4.21 and 4.22. The optimized υ for COL-orig is 10^{20}, and for COL is 10^{12}, on ARPANET. On other topologies, however, the optimal parameters for COL-orig and SDR are changed. SDR-orig also presents a higher blocking probability at low loads. On scaled ARPANET (Figures 4.23 and 4.24), COL-orig, COL, and SDR require re-optimized parameters. At their optimal threshold values, COL and SDR perform the same as CAR-M. In practice, however, optimizing SDR and COL is challenging. SDR has three parameters to optimize: the threshold, the scan interval, and the sample number (the latter two are not shown in the figures). For COL, the best μ value increases exponentially with the network capacity.

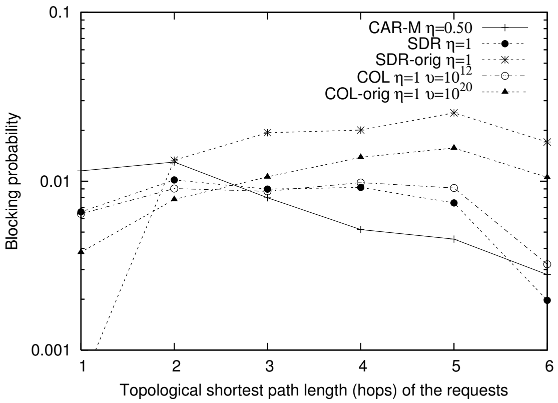


Figure 4.28: Blocking probability at each TSL group of connection on ARPANET at offered load 0.85.

I show that SDR-orig and COL-orig favor shorter connections. The blocking probability at each TSL group is compared in Figure 4.28. SDR-orig represents a deliberate choice to always accept the single-link path. This choice may be appropriate in a network dominated by telephony traffic, which tends to have high geographic locality [17]. Preference to shorter connections (Figure 4.21) may not be acceptable in a network dominated by other types of data traffic.

Improved SDR and COL are robust to network misdimensioning. Both SDR and COL perform well on misdimensioned ARPANET with link (a, b) capacity doubled (Figure 4.25). When I randomly change link capacity by 20 percent of the dimensioned total (Figure 4.26), CAR-M and COL show an advantage over SDR at low loads. As the network becomes more misdimensioned (with 40% changes in Figure 4.27), SDR and COL show slightly better results compared to CAR-M.

Table 4.3 compares routing times. Both original and improved versions are similar in timing. COL is comparable to CAR. Routing time for SDR is much slower, even without accounting for the overhead of updating link load at 1/20 the interval of the average holding time.

4.3 Oracular Optimal Routing

Long-term optimization of dynamic traffic demands is neither feasible nor meaningful as a point of comparison. The precomputed results are suboptimal because the amount of future knowledge is limited. Since rerouting is not allowed for existing connections, in practice, the optimization of a sequence of oracular future arrivals is always placed on a residual network with some existing connections. In particular, given information about a number of future requests (as well as the current state of the network and the termination times for active connections) the oracle selects routes for future requests in a way that minimizes the number of those requests that cannot be routed. This oracular result thus puts a lower bound on the blocking rate achieved by any algorithm operating in the window considered. The oracular window size is selected so as to make the problem of finding the oracular optimum tractable.

I now describe the oracle optimization problem in more detail and present an ILP formulation that allows us to solve it. Optimization begins at a specific simulation time after the network has reached a steady state. In addition to information about the current state of the network and hold time information for all active connections, the optimizer receives a number of future requests with arrival and departure information. (This is the "oracle" part; no real algorithm can have such information.) Figure 4.29 shows an example of six oracle requests and active connection information that should be recorded. (The routes used by each connection active at the start of the optimization window must also be recorded, of course, but are not readily shown by a timeline.)

Let E be the set of graph links, and let R be the set of oracle request pairs, including the arrival and departure timestamps. Let t be a discrete timeline ordered by the arrival time of requests. Therefore, a set of requests sorted by arrival time can be indexed by t. Let r^t be the request that arrives at time t, and let d_{r^t} be the scheduled departure time for request r^t. Let $C_{e,t}$ be the residual capacity on link e at time t, at which point any capacity freed by the termination of active connections before time t has been considered in the network resource constraints. Let P_{r^t} be the set of all available paths for request r based on the network's residual capacity at time t. Let q_{r^t} be the requested capacity of pair r^t. Let Q_e represent the set of paths that traverses edge $e \in E$. Denote by y_{r^t} a binary variable representing the acceptance of request r^t. A solution with $r^t = 0$ implies that r^t is not taken, while a value of 1 means taken. The variable $x_{r^t,p}$ is a positive integer representing the occupancy of path $p \in P_{r^t}$. Again, 0 means not taken, while any other number represents the number of wavelength channels assigned to that path. The ILP formulation is stated as follows.

$$\text{Maximize } \sum_{r^t} y_{r^t} q_{r^t}$$

$$\forall r^t \in R, \sum_{p \in P_{r^t}} x_{r^t,p} \geq y_{r^t} q_{r^t} \tag{4.16}$$

$$\forall r^t \in R, \forall e \in E,$$

$$\sum_{p \in Q_e \cap P_{r^t}} x_{r^t,p} + \sum_{r^\tau : \tau < t, d_{r^\tau} \geq t} \sum_{p \in Q_e \cap P_{r^\tau}} x_{r^\tau,p} \leq C_{e,t} \tag{4.17}$$

$$x_{r^t,p} \in Z^+ \tag{4.18}$$

$$y_{r^t} \in \{0, 1\} \tag{4.19}$$

Equation (4.16) ensures that for each accepted request (right side), enough paths have been allocated to handle the request (left side). Equation (4.17) constrains the channels utilized at any point t in time and at any edge e to the capacity available in the network. The right side of the equation is simply the capacity available in the link e at time t, considering both the initial capacity of the link and the capacity dedicated to connections that were active at the start of the optimization period and have not yet terminated. The terms on the left side include the channels to be provided for request t, as well as the channels provided for other connections allocated before t and terminating after t. The last two equations constrain the values of the integer variables that specify how requests are accepted and routed. As a general form, the ILP solution may assign multiples to one request that maximize the utilization of network resources. However, I do not consider a request capacity greater than one here; thus, no splitting of capacity would occur on multiple available paths. Equivalently in the ILP formulation, the x variables are binary and qs are uniformly one.

Finally, I compare RFR and CAR with the oracular optimal result (OPT) in Figure 4.30. Due to the limitation of computational resources for larger networks, we reduce the per link capacity to 48. Dynamic arrival rate ranges from 1 to 10. Each request asks for one unit capacity. No splitting of paths can occur during optimal routing. The experiment is on the NJ LATA network, and I use SPF to run the first 10,000 arrivals on the network to reach

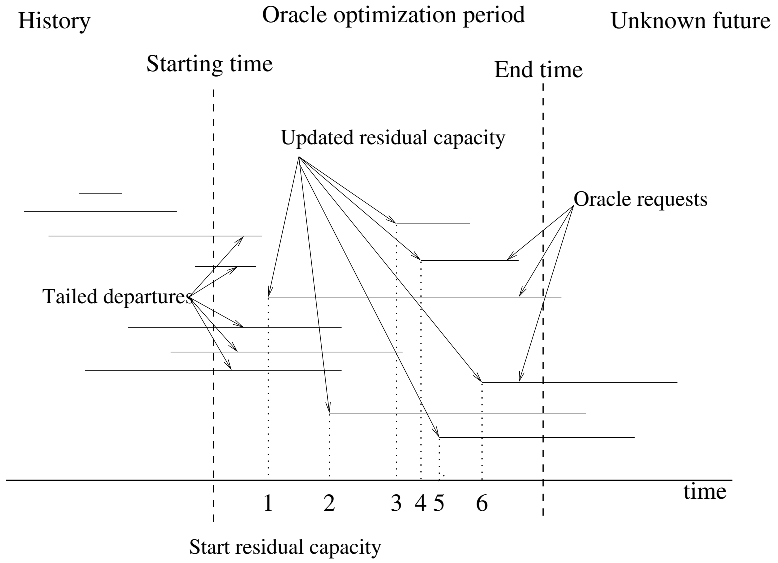


Figure 4.29: Illustration of Oracle requests.

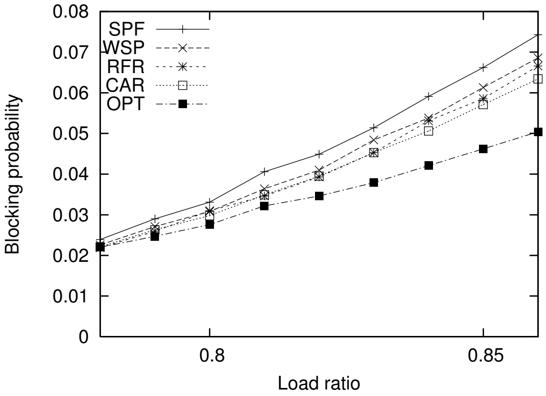


Figure 4.30: Comparison of blocking rates on NJ LATA and OPT.

and obtain a steady state. Then, the next 200 requests are routed by SPF, WSP, RFR, and OPT, respectively. Each data point is an average of 1000 experiments. In this experiment, starting with the same initial network state, CAR is the closest online algorithm to OPT. As the network load increases, the gap between RFR and OPT increases, showing that oracular knowledge becomes increasingly important to reduce blocking in congested network states. Without the aid of oracular knowledge, CAR provides the lowest blocking rate.

4.4 Conclusion

I propose a reduced flow online routing algorithm that provides the lowest blocking and network resource usage, compared to commonly used online algorithms. RFR is most robust to changes in traffic load and shows timing advantages relative to other flow-based algorithms. I also improve online routing with threshold-based admission controlled mechanisms. Using an opportunity cost model, I efficiently estimate the optimal threshold value for the threshold. The model enables me to find several improved congestion estimation routing algorithms. Compared to other algorithms found in previous studies, CAR-M is fast and the optimal threshold value can be identified analytically and is robust to changes in important network parameters, such as topology and capacity. I also discuss an oracular optimization model, showing that the long-term optimization of a network is practically impossible. The results show that good performance is achieved more effectively through deciding whether to admit an available route altogether versus which route to use.

CHAPTER 5

DIMENSIONING DYNAMIC TRANSLUCENT NETWORKS

This chapter discusses the problem of dimensioning resources for dynamic translucent networks. In particular, I consider the Reconfigurable Optical Add-Drop Multiplexer (ROADM) network model, which is introduced in Section 2.1.1. For these networks, optical transponders (OTs), 3R regenerators (REGENs), and wavelengths are disjoint network resources that must be considered separately. A previous study [40] addressed only the REGEN placement problem for this type of network. However, OTs, REGENs, and wavelengths can all affect overall blocking probability, so understanding the dynamics between blocking probability and each resource type is a more important problem. With more variables, dimensioning translucent networks becomes more challenging than dimensioning opaque ones.

I choose CORONET as a targeted design for an efficient resource dimensioning algorithm. Poisson dynamic traffic and four different wavelength operating modes with different levels of wavelength channel restrictions are anticipated for the network. I propose a dimensioning algorithm for OTs, REGENs, and wavelengths and evaluate the performance and cost under the four operating modes. My study shows that bounding the upper limit of the set of usable wavelength achieves the most balanced resource usage.

5.1 Resource Dimensioning Algorithm

On CORONET, the minimal cost route between any pair of nodes is unique, but the placement of REGENs can vary. I use a virtual-link graph model to ease the routing process. A virtual link is created for every reachable optical transparent segment; REGEN is needed to connect two virtual links. The dimensioning algorithm uses a simulation approach that estimates the resource demands on each node and link for a given traffic matrix, as shown in Algorithm 5.1. Let N be the set of nodes, E be the set of physical links, and V be the set of virtual links. Let W be the set of wavelength channels. The connection request set is denoted by $D = \{(s, d)|s \neq d \in N\}$, and λ_r is the arrival rate for a node pair $r \in D$. A route on the virtual-link graph is denoted by $p = \{V_p, N_p\}$, where V_p is the set of virtual links, and N_p is the set of regen nodes. $p(e)$ is the wavelength assigned on physical link $e \in E$. At node n, O_n is the dimensioned number of OTs, while R_n is the dimensioned number of REGENs. $C_{w,e}$ indicates whether wavelength channel w is dimensioned at physical link e.

Simulation runs routing of 20,000 Poisson arrival calls until steady state. Initially, I assume that all resources are available. An arrival is accepted on the preassigned min-cost path if there are enough OT, REGEN, and wavelength resources. Each connection requires two OTs at each of the end nodes, one wavelength on each link, and one REGEN between two virtual links. I use first-fit wavelength assignment. If the preassigned min-cost path is not available, I find another available min-cost path. Each link has maximally 80 wavelength channels, and the maximal number of OTs and REGENs is unbounded. The dimensioned network is acquired by averaging the final used resources of 1000 random runs and rounding each resource allocation to the nearest integer. At the dimensioning stage, I assume the maximal network capacity can support 100% traffic load so no blocking occurs. At the routing stage, the actual traffic follows the same distribution as the dimensioned traffic. Since only dimensioned resources are available for use, call blocking can occur. Each arrival first tries the preassigned min-cost path. If the path is not available, I accept the connection on an alternative min-cost path that is available in the residual graph. If there are not enough resources on any path, the call is rejected. As with dimensioning, first-fit wavelength assignment is used.

5.2 Wavelength Operating Modes

Once OTs and REGENs are installed, reallocating them is expensive and infrequent. Once a DWDM system is installed on a fiber pair, all wavelength channels on that fiber can potentially be used. Depending on the operating or charging mode, a network carrier may use the channels more flexibly. I antic-

Algorithm 5.1: Dimensioning for an independent Poisson traffic matrix.

```
foreach request pair $r \in D$ do
    Find a minimal cost path on the virtual-link graph $spf_r$;
Overall counter $ctr \leftarrow 0$;
Generate a random traffic matrix $\lambda_r \in \mathsf{UNIF}(1,5), \forall r \in D$;
while $ctr < 1,000$ do
    /* Set maximal resources */
    $\forall e \in E, \forall w \in W, C'_{w,e} \leftarrow 1$;
    $\forall n \in N, R'_n \leftarrow MAX$;
    $\forall n \in N, O'_n \leftarrow MAX$;
    Arrival counter $actr \leftarrow 0$;
    while $actr < 20,000$ before reaching steady state do
        Generate a new Poisson arrival or departure request $j$;
        if $j$ is an arrival of pair $r(s,d)$ then
            /* Allocate resources on path $p_j$ */
            $actr \leftarrow actr + 1$;
            if $spf_r$ is free then
                Route $j$ on $p_j \leftarrow spf_r$;
            else
                /* Assume enough resources */
                Route $j$ on an available min-cost path $p_j$ (firstly found and using first-fit) ;
            foreach virtual link $v \in V_{p_j}$ do
                foreach physical link $e \in v$ do
                    $C'_{p_j(e),e} \leftarrow 0$;
            foreach node $n \in N_{p_j}$ do
                $R'_n \leftarrow R'_n - 1$;
            $O'_s \leftarrow O'_s - 1$;
            $O'_d \leftarrow O'_d - 1$;
        else        /* Release resources for departure request on path $p_j$ */
            foreach virtual link $v \in V_{p_r}$ do
                foreach physical link $e \in v$ do
                    $C'_{p_r(e),e} \leftarrow 1$;
            foreach node $n \in N_{p_r}$ do
                $R'_n \leftarrow R'_n + 1$;
            $O'_s \leftarrow O'_s + 1$;
            $O'_d \leftarrow O'_d + 1$;
    $\forall e \in E, \forall w \in W, C_{w,e} \leftarrow \frac{ctr}{ctr+1} C_{w,e} + \frac{1}{ctr+1}(1 - C'_{w,e})$;
    $\forall n \in N, R_n \leftarrow \frac{ctr}{ctr+1} R_n + \frac{1}{ctr+1}(MAX - R'_n)$;
    $\forall n \in N, O_n \leftarrow \frac{ctr}{ctr+1} O_n + \frac{1}{ctr+1}(MAX - O'_n)$;
    $ctr \leftarrow ctr + 1$;
/* Round to the nearest integers */
$\forall e \in E, \forall w \in W, C_{w,e} \leftarrow round(C_{w,e})$;
$\forall n \in N, O_n \leftarrow round(O_n)$;
$\forall n \in N, R_n \leftarrow round(R_n)$;
```

ipate four different modes for operating wavelength channels. In the *TIGHT* mode, only the wavelength channels in the dimensioned network are available for use. This mode represents networks that partition wavelength channels between dynamic traffic and statically provisioned traffic. Each wavelength channel in this mode is restricted to its predefined use. The *BOUND* mode is a variation of TIGHT that allows the use of all channels with ranks under the maximum-dimensioned channel rank on each link. Since I use first-fit wavelength assignment for both resource dimensioning and routing, lower-ranked channels are more likely to be packed than higher-ranked channels. Therefore, *BOUND* can lower the blocking probability with little additional cost. The next mode, *LOOSE*, bounds only the number of channels to be used on each link but allows the choice of which specific channels to use at routing time. This mode reflects networks with more flexible channel allocations that nevertheless require the maximum number of channels used at any time not to exceed the number dimensioned. Finally, the *FREE* mode allows free use of all channels on dimensioned fibers. Dedicated dimensioned networks are likely to operate in this mode because fibers are not shared with other types of traffic.

5.3 Simulation Results

As noted, the simulation uses CORONET and assumes a 40Gbps system with an optical reach of 932 miles. Each fiber pair contains 80 wavelength channels. I use 100 for the normalized cost of an OT, 150 for a REGEN, and 0.07 for the cost per λ-channel-mile. Using the 15 large cities shown in Figure 5.1, I allow connection requests to be generated between any pair. For 100% traffic load, the arrival rate for each pair is uniformly chosen from 1 to 5. The departure rate is 1. Connections arrive at an average rate of one per hour and last for an average of three hours.

Figure 5.2 shows the blocking probability of the four operating modes from load 0.5 to 0.8. The traffic load is defined by proportionally reducing the holding time for each connection. Each data point is an average of simulating 100 random traffic matrices. As expected, the FREE mode yields the lowest blocking. I also study the contributions of each of the three types of resources to the total blocking probability and attribute a blocked call to OT blocking if

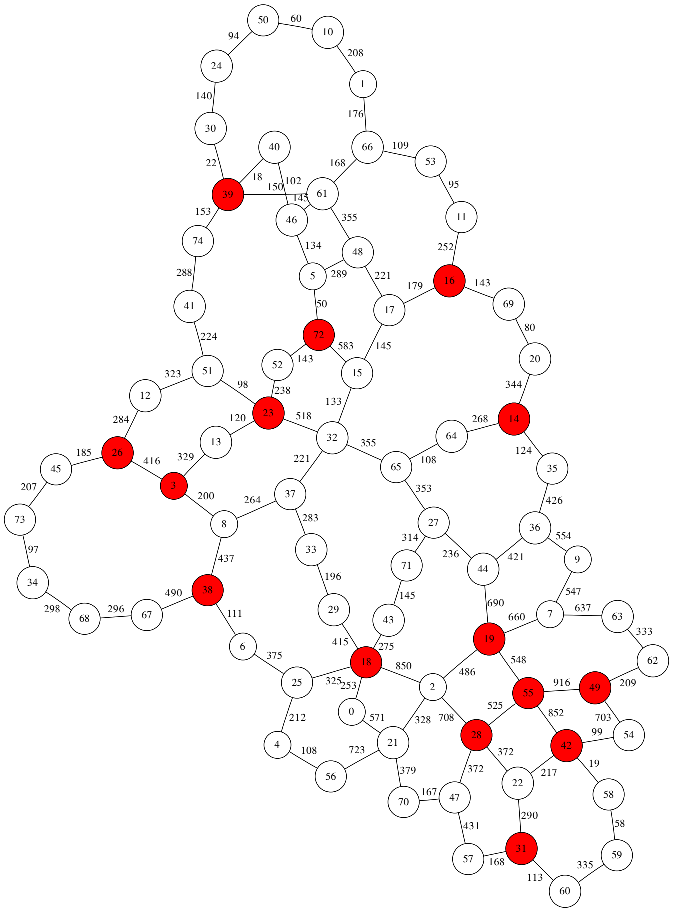
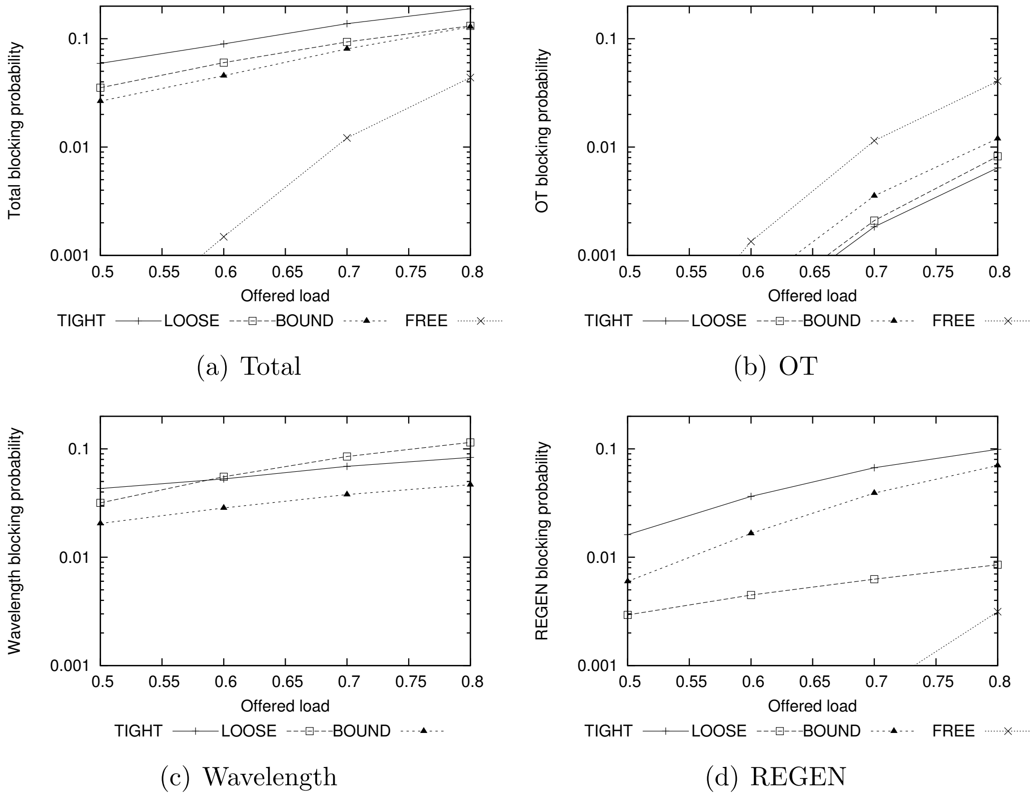


Figure 5.1: U.S. CORONET with 15 large cities highlighted. Links are labeled with distance in miles.

no OTs are available at the end nodes. Otherwise, if a blocked call would have been accepted given enough REGENs (only nodes previously dimensioned with some REGENs), I attribute the call to wavelength blocking. Other blocked calls count as REGEN blocking.

The total blocking probability and the distribution of blocking in the three resource categories can change with shifts in wavelength constraints for different operating modes. For the TIGHT mode, REGENs and wavelengths are the major constraining factors. OTs become the major constraining factor in the FREE mode since wavelength constraints are completely removed (and the need for REGENs is also reduced). The LOOSE mode shows a higher blocking on wavelengths because it provides more freedom of choice on wavelength channel than BOUND and TIGHT modes, and thus more blocked calls are attributed to wavelength blocking versus REGEN blocking.



(a) Total

(b) OT

(c) Wavelength

(d) REGEN

Figure 5.2: Blocking probability on CORONET for random dynamic connection requests from 15 large cities.

The BOUND mode costs only an average 0.88% dimensioning overhead relative to the TIGHT mode. But the FREE mode costs an average of 40.37% dimensioning overhead. With little overhead, BOUND provides the best balancing of resources and the best overall performance relative to TIGHT and LOOSE modes.

5.4 Conclusion

This chapter proposes an efficient resource dimensioning algorithm that allocates OTs, REGENs, and wavelengths for dynamic ROADM networks. I study the performance of a carrier-grade ROADM network under four modes of wavelength channel operation. My results provide network carriers with useful guidance for balancing resource investments on the network.

CHAPTER 6

DIMENSIONING DYNAMIC OPAQUE NETWORKS FOR LINK FAILURE RESTORATION

Failures in optical networks can cause the loss of enormous amounts of data. For critical applications that require rapid service recovery, varied protection schemes have been proposed. However, many applications can tolerate short-term outages and can be exempt from the added expense of providing protection. These applications can be classified under best-effort traffic, which is rerouted when there is a failure. This process can take seconds to minutes (in contrast to protected traffic with rapid recovery, which is 50 to 100 microseconds). Differentiating traffic based on the level of survivability on IP/GMPLS networks has been studied to take advantage of the cost saving opportunities that different types of traffic offer [41, 42].

This chapter explores the potential of network restoration by applying the dimensioning techniques from Chapter 3. These best-effort traffic techniques take more minutes to find a new path, so we do not need to provision the backup path ahead of time. However, before the failure is repaired, the network should be able to restore most of the connections and operate at normal performance with new arrivals. The key question concerns how much extra capacity is needed for the network to sustain good performance after a single link failure.

I start with quantifying the short-term impact of single-link failures by computing the percentage of reroutable connections. The percentage of reroutable connections starts to drop steeply at network load 0.75 and beyond. I then quantify the longer-term impact of link failures, which I show is critical to network performance. My results show that performance can be severely limited for traffic on certain nodes. Under failure conditions, some nodes experience blocking probability 100 times higher than the one under normal operation. I then quantify the costs associated with restoring the network. The thesis applies a redimensioning technique that effectively rebalances the network and addresses the issue of fairness among nodes.

The redimensioned network in fact requires the amount of resources close to shared path protection.

6.1 The Cost of Protection

I use ARPANET to illustrate the ideas and performance results covered in this chapter, but other well-known networks such as NJ LATA and NSFNET were also evaluated.[1] Poisson processes are used to model call arrivals and hold times. Each connection requests a uniform 1-unit capacity and has the same departure rate. The departure rate is determined by the network load (i.e., the product of the projected departure rate and the load ratio). The arrival rates are uniformly distributed between 1 to 10, and the load ratio is varied at some fraction of the projected traffic load. Each request demands the same capacity and hold time, and SPF is used.

Table 6.1 reports the total number of linecards that drive all wavelength capacity for ARPANET with and without protection for a 100% projected load. The original network is dimensioned with an average of 1000 randomly chosen, fully loaded, projected traffic matrices from the same uniform distribution. Since the number of linecards is the true network resource that drives the wavelengths, we use linecard capacity for network cost. The number of linecards is computed by doubling the number of wavelength channels dimensioned. For protected networks, I apply a similar dimensioning algorithm for the same dynamic traffic until steady state. The only difference is that I route two paths, a primary path and a backup path, for each arrival. The backup path is path disjoint to the primary one, so any existing connection is protected under any single link failure. The wavelength capacity is also converted into the number of linecards. The dedicated path protection (DPP) always uses the shortest backup path. The wavelength capacity used by both the primary and backup path is removed on the departure of the arrival. Since the backup path can be longer than the primary path, DPP can cost more than 100% of the original capacity. For shared path protection (SPP), I use a greedy algorithm to find the best shared backup path for each arrival. A new shared backup path may share the wavelength resources of some existing backup paths if their primary paths will not fail at

[1] These results are omitted for brevity, given that they exhibit similar trends.

Table 6.1: Protection capacity overhead (at 100% expected load).

Protection Scheme	**#linecards**	**% overhead**
no protection (reroute)	7702	0%
DPP	19202	149%
SPP(N-hop backup)	11848	53%
SPP(shortest backup)	12424	61%
Restoration and Redimensioning		
Worst case	11683	52%
Mobile linecards	8556	11%

the same time. The N-hop scenario finds any backup path of the lowest cost. The shortest length finds only the lowest cost amongst the shortest paths. Therefore, shared channels are removed if and only if no existing connection is using the channel. Any channel used by a primary path is removed upon the departure of the arrival. The overhead of the SPP is smaller than DPP because of sharing.

6.2 Impact of Link Failures on Restoration

6.2.1 Reroutability

Even though the class of traffic in question is considered noncritical, and thus has no protection assigned, it is important to guarantee stability in performance. For this traffic, rerouting is used to restore the broken connections. Rerouting is to find a new available route for a broken connection and it is initiated as soon as the end points detect path failure. Rerouting does not require capacity reservation and does not depend on failure localization. However, as there is no extra capacity is reserved, one can reasonably expect to see less than 100% recovery. The metric used to measure the immediate impact of a link failure is the *reroutability ratio*. This measure is obtained by taking the ratio of the number of connections that can be successfully rerouted to the total number of connections affected by the failure, when the system is in steady states. The reroutability ratio is obtained by averaging 100 runs of randomly chosen traffic. Figure 6.1 shows this reroutability under a dimensioned network. Starting with a half load (where all connections can be rerouted), most of the connections at load ratios below 0.75 can be

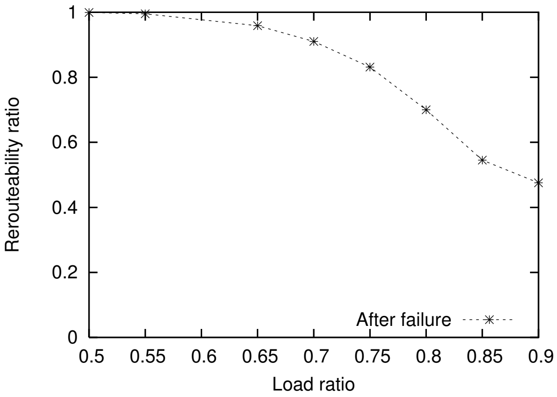


Figure 6.1: Reroutability after a one-link failure on dimensioned ARPANET. Each data point is an average of single failure cases for all links.

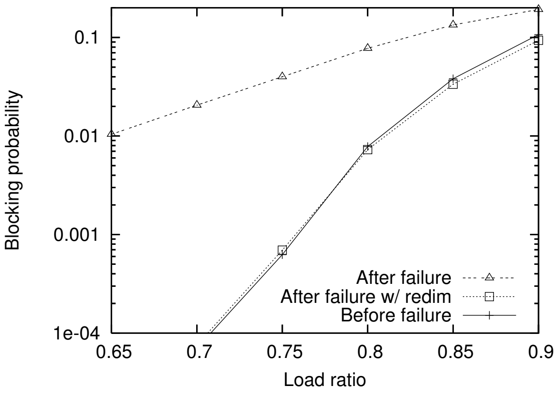
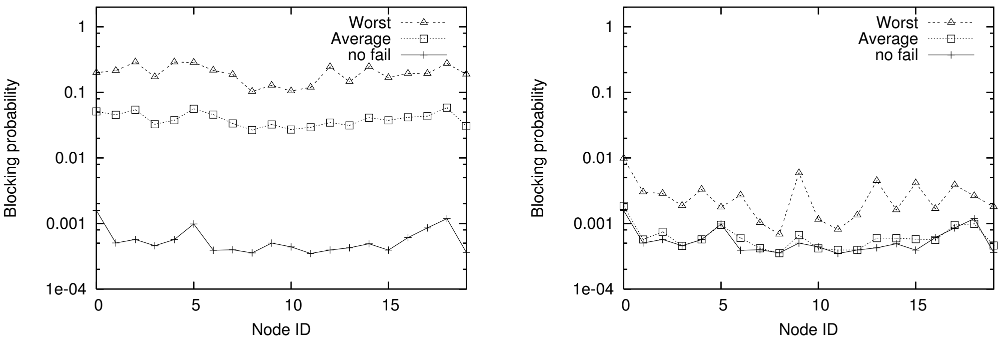


Figure 6.2: Comparison of overall steady-state blocking of a one-link failure on ARPANET with and without redimensioning. Each data point is an average of all single link failure scenarios.

rerouted. However, the next section shows that the real impact on performance occurs after the rerouting stage, before the failure can be physically repaired.

6.2.2 Steady-State Blocking

Until the failure is physically repaired, the network is left operating under suboptimal conditions. Due to the failed link, many of the connections are also forced to use (topologically) non-shortest paths, as well as a limited number of path choices, which exacerbates the impact of the failures. The steady-state blocking probabilities of the network before and after a link failure at different load ratios are shown in Figure 6.2. The numbers for



(a) Before redimensioning

(b) After redimensioning

Figure 6.3: The steady-state blocking of each node on ARPANET at load ratio 0.75.

after-failure results are averaged over all single link failure scenarios. The average blocking performance under a link failure may seem reasonable up to relatively high load ratios (for example, 0.77, where blocking is around 5%). However, when the blocking probability for each node is computed, there is a detrimental impact in terms of fairness. Figure 6.3 shows each network node's blocking probability under normal operation and after failure along with the worst-case blocking for each node. At a load ratio of 0.75, requests may be blocked over 30% of the time (a load ratio of 0.75 was chosen for demonstration purposes because the post-failure blocking remains reasonable, at slightly below 5%). At a load ratio of 0.85 (around 3% blocking without failures, as shown in Figure 6.2), a link failure causes some nodes to drop about a third of their traffic, with the rest of the network dropping around 13% on average.

Repairing an optical link can take a few days to a week. The process is much slower than the anticipated arrival rate of dynamic traffic, which is one in tens of minutes or an hour. Before the failure is repaired, the dynamic network is left in extreme congestion if no additional resources are provisioned.

6.3 Redimensioning and Discussion

In order to rebalance network resources when a failed link occurs, the network is redimensioned with the failed link removed for the same expected 100% traffic load using the basic dimensioning algorithm (Algorithm 3.1).

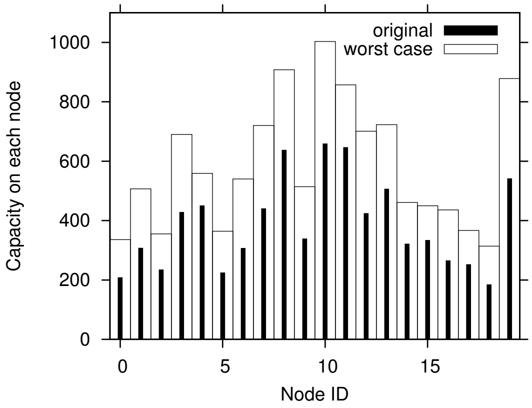


Figure 6.4: Distribution of dimensioned linecards over nodes.

Each single link failure scenario gives a different distribution of linecards on nodes. Practically, each node should prepare enough linecards for a quick installation whenever the worst-case failure occurs. Therefore, the cost of a redimensioned network is the sum of worst-case linecards on each node. Figure 6.4 shows the original capacity and the worst-case capacity after redimensioning over all link failures. The overhead is a result of an increase in many topological shortest paths with a link missing. The total worst-case capacity is 11,683, which is 52% over the original capacity. The overhead of redimensioning is close to the overhead of N-hop backup SPP. Further, if the linecards are mobile (in which the spare linecards at one location can be shipped and installed at another location timely), the total required capacity is reduced to the maximum total capacity over all failure cases, which is 8556 (only 11% overhead). However, such mobility is not expected to happen in near future optical networks.

6.4 Conclusion

I measure the impact of link failures on dynamically routed networks beyond the recovery stage and showed significant degradation in longer-term performance. I then illustrate that a network redimensioning algorithm can be used to restore network balance, but it still requires an overhead of linecards that is close to that of SPP.

CHAPTER 7

DIMENSIONING MULTIPLE DYNAMIC OPAQUE NETWORK DOMAINS

Network domains vary for many different reasons that range from management decisions to geographic locations to vendor-specific component technologies. *Multi-domain networks* can be defined as the joining of peer individual networks (e.g., single routing areas (RA)) through a set of connected border nodes (external network-network interface (E-NNI) as defined in OIF or edge routers (ER) in GMPLS networks). Domains can also be defined hierarchically, corresponding to multiple layers of the optical network (DWDM, SONET/SDH, IP routers, etc) that historically have been implemented in many carrier networks [2].

The ability to dynamically establish connections across the network becomes increasingly important in supporting applications that require high bandwidth on demand. Given the diverse geographic locations of end-users and service providers, many connection requests will require dynamic provisioning across multiple network domains with varying protocols and standards (as well as ownership and management policies). Recent progress in new standards development and multi-domain control plane design improves support for multi-domain services at the optical layer. However, it has also opened the door to new problems that arise in designing algorithms and evaluating performance of multi-domain dimensioning, routing, and management [43, 44]. Many studies have been devoted to designing control and signaling platforms to enable inter-domain path computation. [45, 46] proposed path computation schemes with various QoS constraints in a multi-domain context. A network service plane has been developed to integrate diverse transport network systems [47]. A path computation element (PCE) framework has been extended to support cross-domain shortest path selection [48]. The authors in [49] proposed a framework for a wavelength path establishment mechanism using a ranking database. However, resource provisioning, which is critical to dynamic network performance, has not been considered.

This chapter quantifies the performance of three inter-domain network dimensioning approaches paired with appropriate routing algorithms. I propose a fairness measure to capture the penalty of inter-domain traffic on each network. Using the two basic routing schemes presented by [50], end-to-end global shortest path routing and source-initiated concatenated shortest path routing (equivalent to a single node for two domain case)–I illustrate potential fairness issues in peer-viewed domains. A normalized (and more fair) routing scheme is introduced for limited information sharing. The impact of network scaling and traffic load deviations is shown. My study of multi-domain provisioning for dynamic traffic motivates a new direction in network design in which fairness in network operating costs and benefits is considered.

7.1 Problem Description

In Chapter 3, I saw that an unbalanced network can artificially obscure the differences in performance between various routing algorithms and that a dynamically routed network must be dimensioned properly in order to take full advantage of the overall capacity set in place. When considering a multi-domain network, not only should each domain be dimensioned for its own intra-domain (internal or local) traffic but also it should provide support for inter-domain (external) traffic. When considering network domains that are owned and managed independently, resource usage can become more complicated. I show that one poorly dimensioned domain can cause performance degradation in other well-dimensioned domains. In addition, fluctuations in both internal and external traffic can also affect the performance of external/internal traffic. Therefore, multi-domain dimensioning cannot be solved by simply dimensioning single domains independently. It requires a better understanding of the interaction between network domains and consideration of the impact that levels of shared information (for dimensioning and routing) has on overall performance.

Routing through multiple domains, especially when the domains are bound by conflicting economic interests, raises questions about fairness that require careful attention. A simple example of a conflict between two domains is

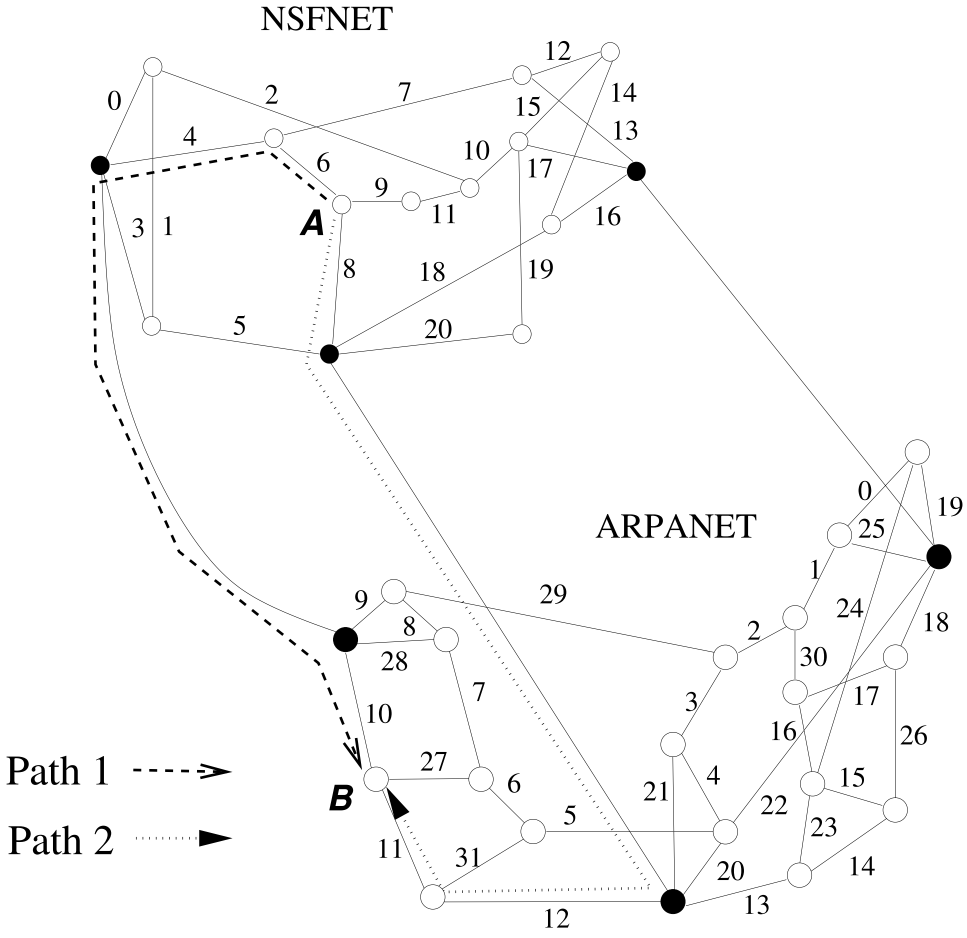


Figure 7.1: NSFNET-ARPANET joint topology (link number is shown on each link).

shown here. Figure 7.1 is a two-domain network connecting NSFNET and ARPANET. Each network has three border nodes (marked in solid color) connected to the borders of the other domain. Each domain may have subdomains, but for the purpose of this study I use only main domains with different ownership. Between the node pair (A, B), we have two choices of global shortest paths, denoted path 1 and path 2. Without loss of generality, if each link has the same cost, path 1 favors ARPANET because fewer resources are used on the ARPANET side. However, path 2 favors NSFNET. When both paths are available, which network should bear the extra cost to establish a connection from A to B? This question not only requires a closer look at performance tradeoffs, but also it motivates cost models for dimensioning and routing policies that consider the notion of fairness.

7.2 Multi-Domain Network Dimensioning

A dynamic network domain must be dimensioned for both inter-domain and intra-domain traffic. Inter-domain traffic is routed through its border nodes: the selection of border nodes is thus critical to the performance and fairness for both domains. Such routing introduces additional loads between internal nodes and border nodes in each network domain. This section discusses general multi-domain dimensioning algorithms. For illustration purposes, all discussions are presented for two-domain scenarios, but my results can be readily extended to an arbitrary number of domains using standard information sharing/hiding techniques, e.g., topology aggregation or virtual topology abstraction.

I assume that every node in each domain can initiate Poisson arrivals to all other nodes in the same domain and to all other nodes in the other domains (equivalent to a full-mesh demand). Domains are joined through their border nodes, on which each border node is connected to a border in the other domain. In most cases, the number of border nodes in both domains is the same. A link connects each pair of border nodes (as shown in Figure 7.1). In practice, it is possible for some border node pairs to be located in the same building/site. In such cases, they may be viewed logically as a single node with finite capacity (equal to the inter-domain link capacity). In fact, the mapping of border nodes between two domains may also not be one-to-one; however, such variations do not affect how dimensioning and routing techniques are designed.

Dimensioning for external traffic for each domain can be treated as dimensioning for additional traffic between each pair of internal nodes to border nodes. I separate the capacity of external and internal traffic loads for the purpose of analysis: in practice, the wavelength resources are shared and no distinction in usage is made. I also define the capacity C_b^I for the inter-domain links that connect border node pairs. Each dynamic connection pair (internal or external) has arrival/departure/capacity rates defined.

The intra-domain traffic matrix is defined as it is for a single domain (Chapter 3). Let $T = \{(\lambda, \mu)\}^R$ be the traffic matrix, where λ is the arrival rate, μ is the departure rate for all request pairs in all end-to-end request pairs R. All traffic across domains, i.e., source and destination nodes are from different domains, is external traffic. Let N be the set of nodes in this domain and S be the set of nodes in the other domain. The external traffic matrix is defined by $T' = \{(\lambda, \mu)\}^{N \times S}$, where the same arrival and departure rates, λ and μ, apply for each external request pair. On a given domain, all external traffic comes from its border nodes. Let N' be the set of border nodes. The equivalent external traffic on the local network is modeled by a traffic matrix, T'_L, in which the arrival and departure rates between all nodes $n \in N$ and all borders $b \in N'$ are defined and the elements are indexed by (n, b). The traffic load between any external node s and local node n can come from any one of the borders. Let $p_{n,s,b}$ be the probability that traffic between n and s comes from border b, such that $\sum_{b \in N'} p_{n,s,b} = 1$. Therefore, the equivalent external arrival rate for n and b is an aggregated arrival rate of all external nodes, weighted by the probability, $p_{n,s,b}$. Equation 7.1 presents the equivalent external traffic matrix on the local network. The entire traffic matrix is then the sum of T and T'_L for the same pair of nodes.

$$T'_L = \{(\sum_{s \in S} \lambda p_{n,s,b}, \sum_{s \in S} \mu p_{n,s,b})\}^{N \times N'} \tag{7.1}$$

The *projected load* of a network is the amount of traffic that the total given network capacity can support without being overloaded. Equation 7.2 defines the projected load for internal traffic. It is the ratio of average traffic load (stochastic arrival/departure rate times the topological shortest path lengths) to the total available network capacity. Topological Shortest Length (TSL) is the minimum number of hops for a connection in an empty network; obviously, available shortest paths selected on residual networks can be longer than their TSL. For external traffic, the average amount of resources used for each connection is the average shortest path lengths to all border nodes. Equation 7.3 defines the external traffic load. All dimensioning techniques use the same load metrics for fair comparison.

$$proj\_load_{int} = \frac{\frac{\lambda}{\mu} \sum_{i \in R} TSL_i}{\sum_{e \in E} C_e} \tag{7.2}$$

$$proj\_load_{ext} = \frac{|N||S| \frac{\lambda}{\mu} \sum_{b \in N'} \frac{TSL_{(n,b)}}{|N'|}}{\sum_{e \in E} C'_e} \tag{7.3}$$

Note that dimensioning a network domain to support traffic to the other domain can be the same as dimensioning a single network with an estimate of external traffic distribution on borders. I use the basic dimensioning algorithm (Algorithm 3.2) to dimension each network separately, using the sum of T and T'_L. The external load (so as to the total capacity) on each domain network remains the same regardless of dimensioning algorithms. Using a similar dimensioning approach, the capacity of inter-domain links is determined according to Equation 7.4.

$$C_b^I = \left\lfloor \frac{\lambda}{\mu} \sum_{n \in N} \sum_{s \in S} p_{n,s,b} \right\rfloor \tag{7.4}$$

7.3 Several Routing and Dimensioning Techniques

This section discusses three dimensioning techniques that estimate the traffic loads to each border node (i.e., $p_{n,s,p}$). This depends on the amount of information shared across domain borders and the inter-domain routing agreement.

Three dynamic routing algorithms are used here. The first two are the same as the algorithms found in [50]. In **source-initiated concatenated shortest path routing (CSR)**, the requesting node chooses the closest border node and uses the shortest path. Starting from this border node, the path in the other domain is selected. No information other than the destination is shared between border nodes. If the downstream domain cannot find a path to the destination through the selected border or the inter-border link is full, the call is rejected.[1] **End-to-end global shortest path routing (E2E)** chooses the border nodes that result in available shortest global paths. Network domains do not have to disclose detailed topology and link load information to other domains. For each connection request, they need broadcast only the available (reachable) shortest path distance from source to each border for the source domain, from each border to all other borders for the intermediate domains, and from each border to the destination for the destination domain. Then, the global shortest path can be found by choosing the combination that yields the shortest length global path. If a domain has no available paths or if two domains cannot join paths due to resource

[1] I omit crankback path selection because paths found by other shared routing algorithms present similar properties to those found by fail-and-retry mechanisms in two-domain cases.

unavailability, the call is rejected. Equation 7.5 shows the computation of shortest paths between connected border pairs in two-domain networks. The border pair is picked with minimum $l^{E2E}_{b,b'}$. **Normalized global shortest path routing (nE2E)** is similar to E2E, but each reported distance is normalized by the average TSL of each respective domain before being advertised and summed. Equation 7.6 shows the computation of nE2E path length. The pair of border nodes is picked with minimum normalized distance $l^{nE2E}_{b,b'}$. nE2E is important because E2E favors the border node selection for the larger network when two domains are of different sizes: since the the larger domain is more likely to have longer paths to the border, its shortest path length can dominate the length of the global shortest path and force the smaller domain to pick unfavorable paths.

For example, assume available SPF lengths from a source node to three borders are 5, 7, and 11 hops in the larger domain A (average TSL=7). In the smaller domain B (average TSL=2), three corresponding borders to the destination node can be 3, 2, and 1 hops away. In E2E, the path with 5 hops in A and 3 hops in B is chosen. In nE2E, the path with 7 hops in A and 2 hops in B is chosen. Although the path picked by nE2E is longer, it is more fair and yields better overall performance for both internal and external traffic on the network, if the network is also dimensioned for nE2E.

$$l^{E2E}_{b,b'} = TSL_{(n,b)} + TSL_{(s,b')} \tag{7.5}$$

$$l^{nE2E}_{b,b'} = \frac{|R|TSL_{(n,b)}}{\sum_{i\in R} TSL_i} + \frac{|S|TSL_{(s,b')}}{\sum_{i\in S\times S} TSL_i} \tag{7.6}$$

I now describe the computation of $p_{n,s,b}$ in three dimensioning algorithms. Once the distribution is computed, I can use the basic dimensioning algorithm (Algorithm 3.2) to dimension each network separately. **Independent shortest path dimensioning (IS)** is used for two networks that share only node information.[2] In this case, an internal node has no idea which border node(s) external calls will come from. Therefore, external traffic is split uniformly across all border nodes, in this case, $p_{n,s,b} = \frac{1}{|N'|}$. **Global shortest path dimensioning (GS)** allocates wavelength resources using least-cost

[2] Node information is the minimal amount of information needed to support full bandwidth-on-demand services; no request is ever made without knowing which destination node to connect to.

Algorithm 7.1: Computation of $p_{n,s,b}$ for GS/NS.

```
foreach inter-domain connection pair $(n, s)$ do
    foreach border node pair $(b, b')$ from each domain do
        Compute the total path length $l^{E2E}_{(b,b')}$ (or $l^{nE2E}_{b,b'}$ for NS);
    Identify the subset of borders $B \subseteq N'$ of minimal total path length;
    foreach shortest path border nodes $b \in N'$ do
        $p_{n,s,b} = \frac{1}{|B|}$ for $b \in B$, $p_{n,s,b} = 0$ for $b \notin B$;
```

routes crossing two domains (assuming E2E). This approach is used when two networks are willing to share path lengths at the border node (the appropriate E-NNI interface) to pick the global shortest path. The traffic rate to each border node is then weighted by the likelihood that each border node will be chosen using E2E. Algorithm 7.1 shows the computation of $p_{n,s,b}$ for each inter-domain connection pair (n, s). **Normalized shortest path dimensioning (NS)** is similar to GS except for the metric used to compute the concatenated shortest path length. Algorithm 7.1 shows the procedure to compute $p_{n,s,b}$ for NS, assuming the use of nE2E.

7.4 Fairness Measurement

I propose a *penalty ratio* to measure the fairness of resource usage on each network domain for shared routing. The penalty incurred by each network, computed by Equation 7.7, sums the normalized number of extra hops used for each inter-domain routing. For example, if the TSLs of an internal node to three borders are 1, 4, 10, and the selected path uses the border that is 4 hops away, the penalty for that connection is 3. The total penalty for the dimensioned network is the weighted sum of penalties to all border nodes, given the traffic load distribution. Then, the penalty ratio of two networks is the ratio of the penalty of each domain.

$$\begin{aligned} penalty = \frac{\lambda}{\mu} & \sum_{n \in N \setminus N', s \in S, b \in N'} p_{n,s,b} \frac{TSL_{(n,b)} - \min_{n,k \in N'} TSL_{(n,k)}}{\min_{n,k \in N'} TSL_{(n,k)}} \\ & + \frac{\lambda}{\mu} \sum_{n \in N', s \in S, b \in N'} p_{n,s,b} TSL_{(n,b)} \end{aligned} \tag{7.7}$$

Table 7.1: NSFNET wavelength capacity (total 1491).

link#	**IS**	**GS**	**NS**
0	60	65	74
1	31	38	37
2	75	68	70
3	79	62	61
4	108	94	91
5	126	114	112
6	65	57	52
7	104	88	79
8	80	105	119
9	43	57	66
10	92	96	97
11	44	45	42
12	36	38	38
13	84	91	94
14	34	45	48
15	21	31	34
16	82	62	48
17	90	113	117
18	122	97	86
19	49	49	47
20	66	76	79

Table 7.2: ARPANET wavelength capacity (total 2496)

link#	**IS**	**GS**	**NS**
0	26	26	26
1	71	67	64
2	144	116	118
3	93	93	96
4	48	48	51
5	123	114	115
6	89	78	81
7	55	53	51
8	29	40	37
9	116	95	92
10	118	80	75
11	127	81	90
12	145	126	125
13	135	166	165
14	49	59	60
15	10	19	17
16	41	52	51
17	42	56	51
18	67	111	106
19	50	82	75
20	100	91	101
21	68	99	96
22	205	153	172
23	63	80	80
24	38	48	48
25	65	91	84
26	34	45	46
27	36	39	37
28	31	60	55
29	167	119	124
30	65	62	61
31	46	47	46

7.5 Simulation Results and Discussion

I ran simulations on joint NSFNET-ARPANET networks (Figure 7.1) with uniform arrival rates and departure rates for Poisson traffic. Table 7.1 shows the dimensioned network capacity for NSF, while Table 7.2 shows it for ARPANET. For routing, I ran multiple random arrival sequences until the 95% confidence interval fell within ±5% of the results. In my network configuration, the computed penalty ratio for NSFNET to ARPANET is 2.46 using GS and 0.99 using NS. Therefore, NS-nE2E is a fairer dimensioning and routing scheme compared to GS-E2E since it is closer to one.

Figures 7.2–7.4 show the inter-domain call blocking probabilities and intra-domain blocking on NSFNET and ARPANET, respectively, as the inter-domain load changes. The results show that a significant reduction in blocking can be achieved using shared information dimensioning (GS and NS) compared to IS. For example, an over 90% reduction in call blocking can be seen at load 0.95. However, applying shared routing on an independently dimensioned network instead of independent routing (CSR) does not improve the result significantly. Therefore, the networks must be jointly dimensioned in order to fully benefit from shared routing algorithms.

In terms of resource allocation, NSFNET has a higher external/internal capacity ratio (2.82) compared to ARPANET (1.38) because it is a smaller network. In other words, NSFNET has to pay more for capacity to sustain the inter-domain traffic in proportion to its own size. Internal traffic blocking is greatly improved using a fair routing scheme (nE2E) on NSFNET without affecting the performance of (Figure 7.3). ARPANET Figures 7.5–7.7 show the changes in blocking performance on the three types of traffic as the NSFNET is scaled (Figures 7.8–7.10 show ARPANET scaling). Note that using shared dimensioning, a network cannot under-dimension below 0.95 of the desired traffic load without hurting the performance of other traffic, both internally and externally. However, using independent dimensioning, the internal blocking of one network actually increases when the other network is overdimensioned (Figures 7.7 and 7.9).

Because an overdimensioned network can sustain more inter-domain traffic, it puts more pressure on the internal traffic of the other network. As the capacity of one network further increases, the blocking probability of the other network becomes stable because the inter-domain links become

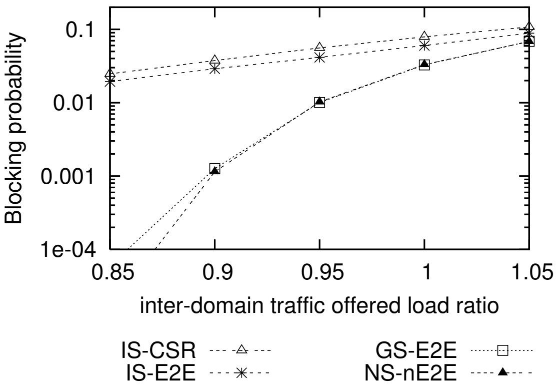


Figure 7.2: Inter-domain blocking (internal offered load 1.0).

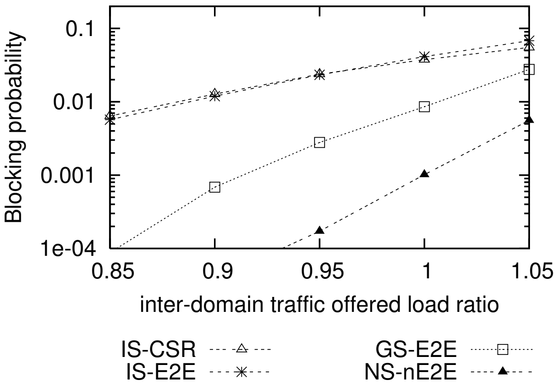


Figure 7.3: NSFNET internal blocking (internal offered load 1.0).

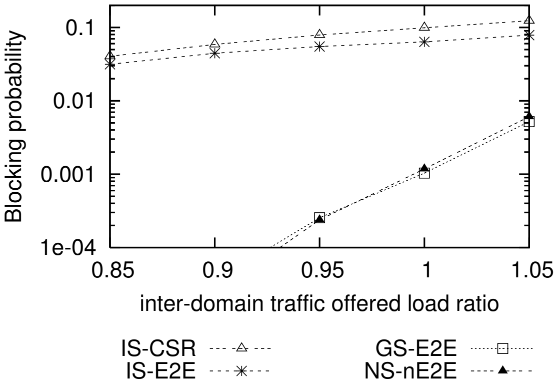


Figure 7.4: ARPANET internal blocking (internal offered load 1.0).

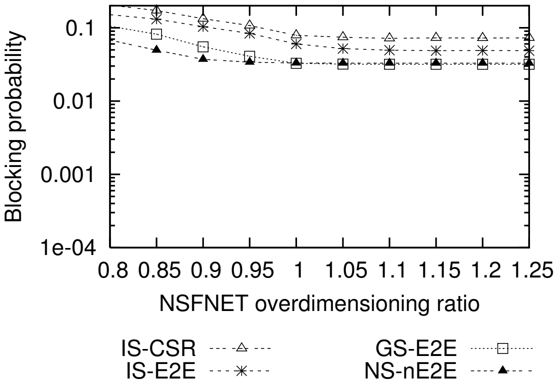


Figure 7.5: Inter-domain blocking (all offered load 1.0).

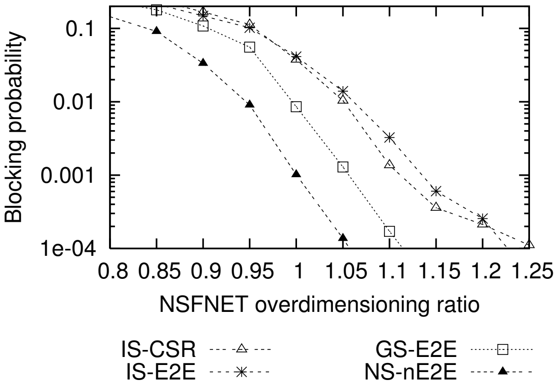


Figure 7.6: NSFNET internal blocking (all offered load 1.0).

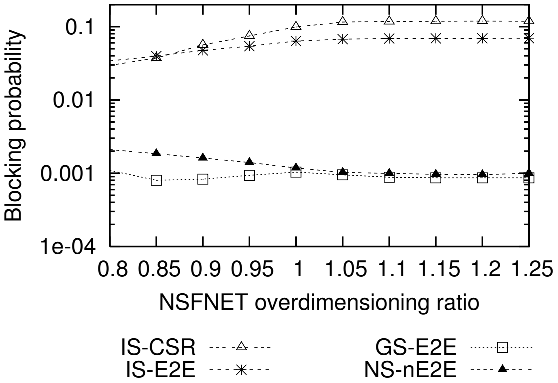


Figure 7.7: ARPANET internal blocking (all offered load 1.0).

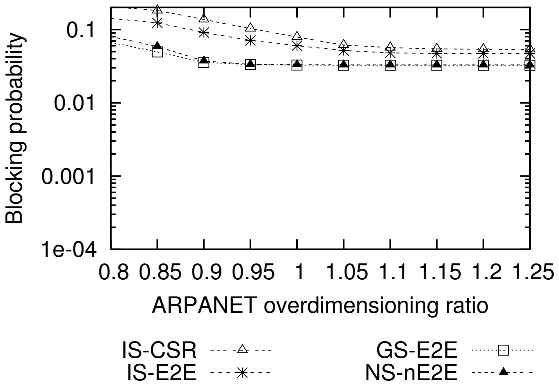


Figure 7.8: Inter-domain blocking (all offered load 1.0).

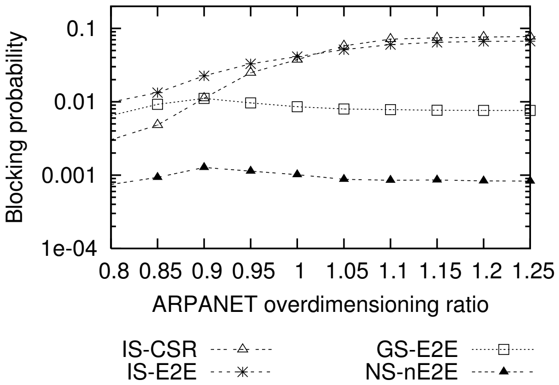


Figure 7.9: NSFNET internal blocking (all offered load 1.0).

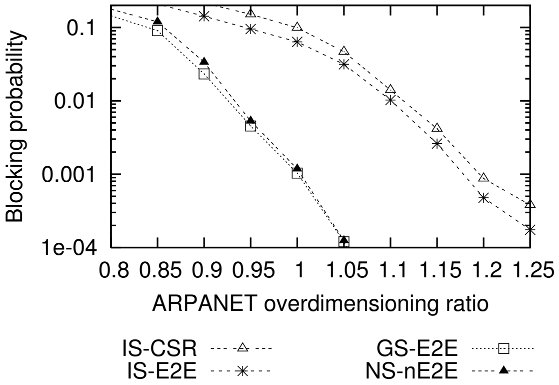


Figure 7.10: ARPANET internal blocking (all offered load 1.0).

saturated and limit the volume of inter-domain traffic. When there is no capacity limit on inter-domain links[3], Figures 7.11–7.16 show similar trends; but much lower blocking can be achieved for shared dimension algorithms. This is because unlimited inter-domain links prevent external traffic routing through unfavorable paths (much longer than TSL) to a nonoptimal border node that has available capacity.

7.6 Conclusion

I quantify the performance and fairness of various routing and dimensioning schemes in a multi-domain environment with simple abstract levels of information being shared between the domains. I propose the metrics to measure the loads of multi-domain traffic and the fairness of dimensioning and routing schemes. My results show that joint provisioning—for both dimensioning and routing—and playing fair are crucial to the performance of both networks for both internal and external traffic. I also show that scaling of one network domain can affect the performance of the other network, especially if they are independently dimensioned. These results motivate the need to gain a deeper understanding of the interaction between network domains of varying sizes, especially to maximize the overall performance and fairness across domains that are independently owned.

[3] Typical cross-domain links, like cross-ocean cables, can have sufficiently more capacity allocated.

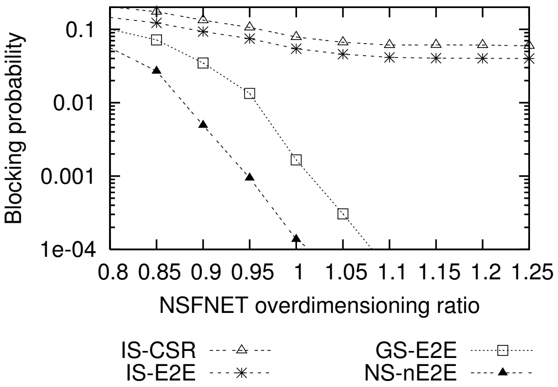


Figure 7.11: Inter-domain blocking (all offered load 1.0, $C_b^I = \infty$).

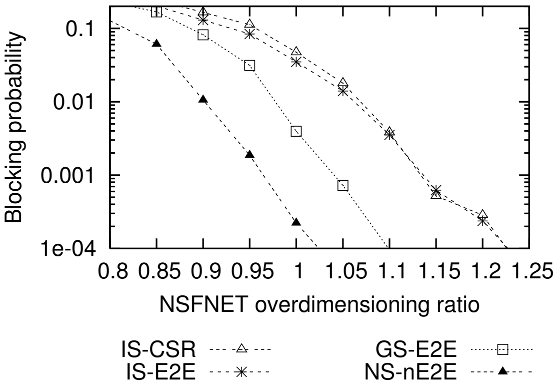


Figure 7.12: NSFNET internal blocking (all offered load 1.0, $C_b^I = \infty$).

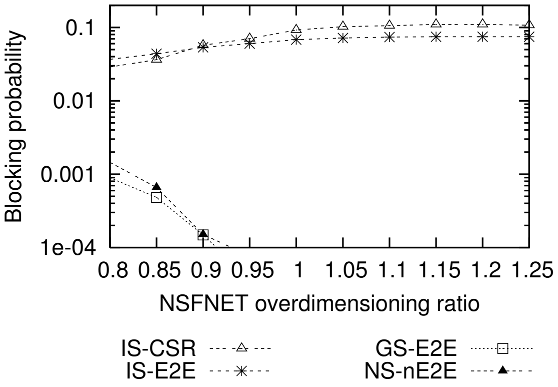


Figure 7.13: ARPANET internal blocking (all offered load 1.0, $C_b^I = \infty$).

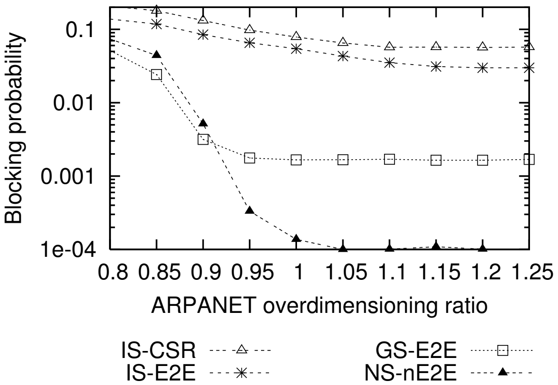


Figure 7.14: Inter-domain blocking (all offered load 1.0, $C_b^I = \infty$).

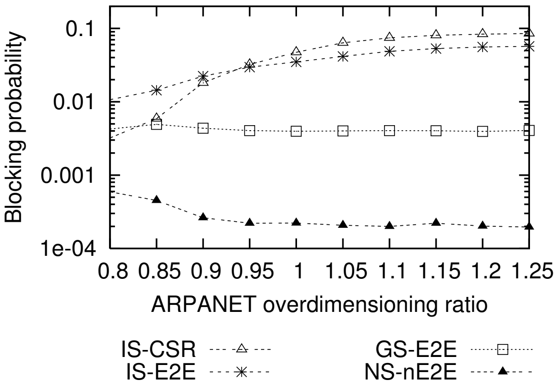


Figure 7.15: NSFNET internal blocking (all offered load 1.0, $C_b^I = \infty$).

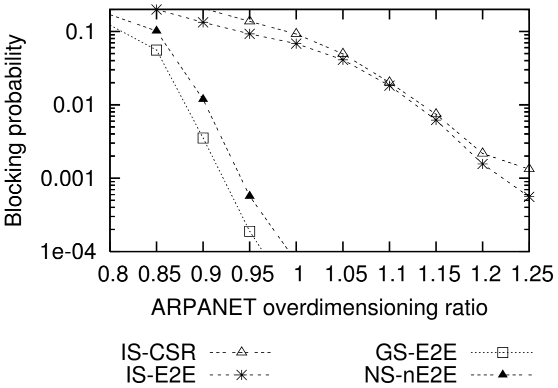


Figure 7.16: ARPANET internal blocking (all offered load 1.0, $C_b^I = \infty$).

CHAPTER 8

DIMENSIONING AND ROUTING FOR DYNAMIC WAVELENGTH SERVICES ON TRANSLUCENT NETWORKS

There is a need for rapid, on-demand connection provisioning for large customers, such as national banks or chain retailers, who manage high-rate private line networks spreading over multiple cities. Carriers already offer bandwidth-on-demand services at lower data rates, such as AT&T's Hosh VPN [51] and Optical Mesh Service (OMS) [52]. Expanded service models have been proposed for high data-rate, bandwidth-on-demand private networks at the photonic layer [53]. With these *dynamic wavelength services*, a customer can purchase/lease photonic access interfaces at multiple locations and set up photonic connections to support high (wavelength) rate private lines between pairs of available access interfaces. Carriers must preinstall enough network resources to support arbitrary changes of customer connections.

One goal of a carrier is to minimize installed network resources needed to support dynamic wavelength services. However, finding an optimal network resource dimensioning solution for even one customer requires prohibitive computing power. Given a fixed set of access interfaces, a customer can interconnect them in a large number of potential patterns. The carrier has the opportunity to route each connection over many possible routes in the backbone network. The number of possible routes grows exponentially in network scale. Furthermore, other provisioning variables, such as wavelength selection and placement of intermediate regenerators, further increase the size of the problem. Even if provisioning is done at the design phase, one must reduce the problem space to enable near-optimal solutions to be found. In the future, customers may even add or remove a few ports gradually, either monthly or weekly. Fast provisioning algorithms are important for network carriers to serve a large number of customers timely and efficiently.

On the other hand, many dynamic wavelength service applications are also mission critical, including private networks for military, financial, and medical use. These networks often require additional reliability and robustness [54]. As the dynamic wavelength service allows customers to change their connections over time, taking down and restoring a connection can cost tera-bytes of data loss on a 10 Gbps link with current system technology. Such loss is intolerable for many critical applications. Therefore, the carrier must provision its network to be *nonvolatile*, that is, to ensure that existing connections are not affected by demand changes. I find that the optimal provisioning solution is generally volatile, even allowing one photonic bridge-and-roll (B&R) operation per connection. A nonvolatile solution is needed and proposed for the dynamic wavelength service to successfully serve these applications.

This chapter discusses the dimensioning and routing problem for a practical dynamic traffic model, called the *dynamic wavelength service*, on a reconfigurable optical add/drop multiplexers (ROADM) network, a type of translucent optical network. First, I introduce efficient algorithms to solve the resource optimization problem for a single customer. In particular, I propose a demand-matrix reduction technique that can greatly simplify the problem, as well as an efficient lower bound computation. Next, I develop heuristic optimization algorithms based on simulated annealing and genetic algorithms and a few heuristics for search space reduction. Also shown there is an scalable and fast greedy approach to the problem. I study the robustness of the dimensioned network by analyzing the volatility of the optimal provisioning solution for two types of traffic modes: *individual demand changing mode* and *maximum demand changing mode*. A sufficient condition for a guaranteed *nonvolatile* resource provisioning is proposed.

8.1 Problem Description

This section formulates an integer linear programming (ILP) problem for resource optimization. Finding an optimal solution is impractical given the size and structure of the problem.

In a ROADM network of a set of nodes N, the customer has a fixed number of OT ports at each node, denoted by O_n, where $n \in N$. The customer can connect nodes freely if spare ports are available.

Denote E for the set of physical DWDM links. The set of virtual links is denoted by $\mathcal{V}$. A *virtual link*, $v \in \mathcal{V}$, is a set of physical links corresponding to a simple path. $\mathcal{V}$ includes virtual links representing individual physical links ($\{e \in E\}$), as well as paths within an optical reach limit ($\{e_1, e_2, e_3\}$). Each physical link is associated with a distance l_e in miles. Each virtual link is also associated with a distance value that is the sum of distance values of all member physical links (Equation 8.1).

$$l_v = \sum_{e \in v} l_e \tag{8.1}$$

A route p includes a set of virtual links, along with wavelength assignment information (p_v) and a set of regen nodes (p_n). The cardinality of p_v is the number of transparent segments in the path. If a route has three segments, the route is represented by $p = \{V_p = \{(v_1, w_1), (v_2, w_2), (v_3, w_3)\}, N_p = \{n_1, n_2\}\}$, where $v_1 \cap v_2 = v_1 \cap v_3 = v_2 \cap v_3 = \emptyset, n_1 \neq n_2$ given simple path constraints. The route cost $C(p)$ is associated with a common cost rate CC and a REGEN cost rate RC in Equation 8.2.

$$C(p) = \mathsf{CC} \sum_{(v,w) \in V_p} l_v + \mathsf{RC}|p_n| \tag{8.2}$$

The definition of the customer's traffic demands is the following.

Definition 1 (Demand)**.** *A demand is a unique OT-to-OT bidirectional connection between a pair of nodes.*

If there is more than one connection between a node pair, each connection is treated as a distinct demand.

Definition 2 (Port constraint)**.** *A port constraint is the maximal number of connections allowed at each node in the network.*

Port constraints thus ensure that the number of ports at one node is less than or equal to the sum of ports at all other nodes.

$$\forall i, O_i \leq \sum_{j \in N, j \neq i} O_j \tag{8.3}$$

Definition 3 (Demand matrix)**.** *A demand matrix is a set of demands within the port constraint of a customer.*

Definition 4 (Routable)**.** *A demand is routable if at least one route is available in the network to connect the demand.*

Definition 5 (Demand matrix satisfactory)**.** *A demand matrix is satisfied if all member demands can be routed concurrently.*

Using the dynamic wavelength service, the OT cost is fixed per customer. Only REGEN and wavelengths are subject for optimization.

Definition 6 (Dimensioned network)**.** *A dimensioned network is a network with enough wavelength capacity and REGEN devices to satisfy all demand matrices.*

A dimensioned network is described by $(W_{e,w}, R_n), \forall e \in E, w \in \mathcal{W}, n \in N$. $\mathcal{W}$ is the set of wavelengths available in a fiber. $W_{e,w}$ is a binary variable that indicates whether wavelength w on physical link e is allocated. R_n is a nonnegative integer representing the number of allocated REGENs at node n. In this paper, I assume that the number of wavelengths and REGENs allocated is always smaller than the capacity.

A demand matrix is a possible set of concurrent connections that may be requested by the customer. Sharing of network resources is allowed between demands of different demand matrices since they are different sets of concurrent connection requests. However, no resource sharing is allowed within a demand matrix. Let $N(d)$ be the node pair of a demand d. $\mathcal{P}_{N(d)}$ is the set of possible routes for that pair regardless of resource availability. On a dimensioned network, for each demand matrix $D \in \mathcal{D}$, a route must be available for each demand $d \in D$. A customer can have many possible dimensioned networks.

The resource optimization problem is to find a route allocation for each demand that produces the minimal-cost dimensioned network. The cost objective function for the network is defined by Equation 8.4.

$$\text{minimize } \mathsf{CC} \sum_{e \in E} \sum_{w \in \mathcal{W}} W_{e,w} l_e + \mathsf{RC} \sum_{n \in N} R_n \tag{8.4}$$

Let the binary value $X_{p,d}$ be the indicator for whether a route $p \in \mathcal{P}_{N(d)}$ is chosen for demand d. One route is selected for each demand.

$$\forall D \in \mathcal{D}, \forall d \in D, \sum_{p \in \mathcal{P}_{N(d)}} X_{p,d} = 1 \tag{8.5}$$

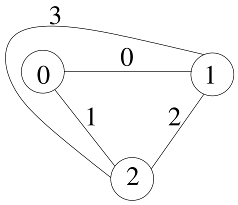


Figure 8.1: Virtual-link graph of a triangular network with node and link numbers.

Equation 8.6 prevents wavelength sharing within the same demand matrix and ensures that only allocated wavelengths are used to route demands. $I_{\{\}}$ is an indicator function, which is 1 if the condition is met or 0 if not.

$$\forall e \in E, \forall w \in \mathcal{W}, \forall D \in \mathcal{D},$$
$$\sum_{d \in D} \sum_{p \in \mathcal{P}_{N(d)}} I_{\{(v,w) \in V_p, e \in v\}} X_{p,d} \leq W_{e,w} \tag{8.6}$$

Equation 8.7 prevents REGEN sharing within the same demand matrix.

$$\forall n \in N, \forall D \in \mathcal{D}, \sum_{d \in D} \sum_{p \in \mathcal{P}_{N(d)}} I_{\{n \in N_p\}} X_{p,d} \leq R_n \tag{8.7}$$

I use Figure 8.1 to illustrate an example. The customer has one OT port at each node. The port constraints are $O_0 = O_1 = O_2 = 1$. Assume $l_0 = 400, l_1 = 400, l_2 = 900$, and $l_3 = 800$ (in miles). There are three demand matrices. Each demand matrix contains one demand. They are $d_{0,0} = (0,1), d_{1,0} = (0,2)$, and $d_{2,0} = (1,2)$. The optimal solution is $W_{0,1} = W_{1,1} = 1, W_{i,j} = 0 otherwise$. Route $\{(0,1), \emptyset\}$ is assigned to $d_{0,0}$. Route $\{(1,1), \emptyset\}$ is assigned to $d_{1,0}$. Route $\{(3,1), \emptyset\}$ is assigned to $d_{2,0}$. No REGEN is required. All demands can share the same wavelength channel.

8.2 Differentiation to the Hose Model for VPN Traffic

Dynamic wavelength service shares some similarity with the "hose model" that was proposed for characterizing VPN traffic [51]. The hose model defines the total input and output bandwidth (which can be different) of each access node and allows variable node-to-node pipe bandwidth. Kumar *et al.* [55] proposed the optimal tree solution for the hose model as the tree topology

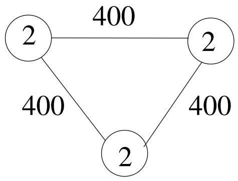


Figure 8.2: An example that the optimal solutions may not be a tree. Physical graph of a triangular network with node port numbers and link distances in miles. All links are 400 miles. Assume that reachability is 932 miles. If all nodes have two ports, the optimal solution is one wavelength per link, which is not a tree.

presents several advantages for the IP network. Dynamic wavelength service is an analogy of hose model at the photonic layer. The input and output bandwidths are the same since wavelength routes are bidirectional. However, the tree solution can not be readily applied to our problem for the following reasons. First, as with VPNs, the optimal solution does not have to be a tree. Figure 8.2 shows an example. Some benefits of the tree network, such as simplified label switching does not apply to the photonic layer. Second, Kumar's polynomial time tree construction problem for symmetric bandwidth (equal I/O bandwidths for all nodes) does not directly apply to my problem with wavelength continuity and REGEN placement problems. Figures 8.3 and 8.4 show examples that the lowest-cost tree is not necessarily the lowest cost after wavelength assignment or REGEN placement, even if Kumar's original algorithms are extended for weighted graphs.

8.3 Optimization Techniques

I next introduce demand-matrix reduction and problem decomposition techniques. The input of the optimization problem, i.e., the number of demand matrices, can be reduced using my demand-matrix reduction algorithm.

8.3.1 Demand-Matrix Reduction

The number of demand matrices grows rapidly in terms of the number of ports. Table 8.1 presents the numerical values of the demand matrices for a few representative constraints. Figures 8.5 and 8.6 show that the total number of demand matrices grows exponentially in terms of the number of

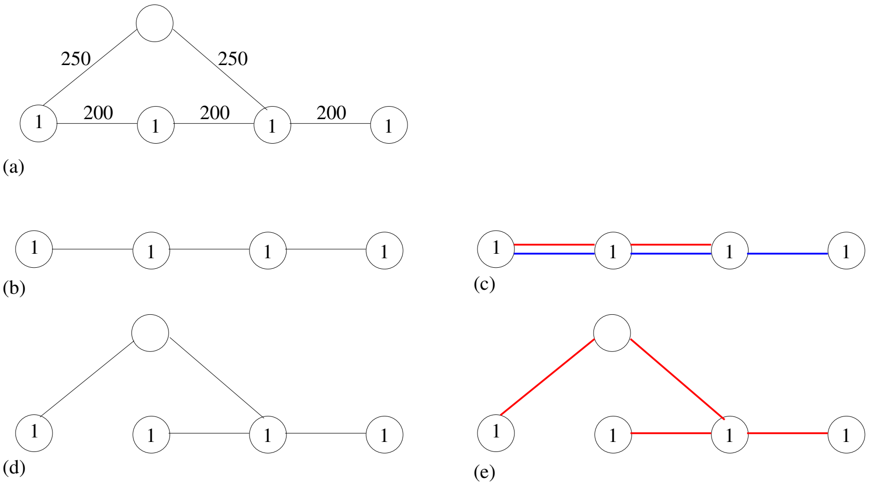


Figure 8.3: An example showing that Kumar's algorithm does not apply to the wavelength assignment problem. (a) Physical graph of a network. The nodes are marked with the number of ports available and links are marked with distance in miles. Assume that reachability is 932 miles. (b) The lowest-cost tree created applying Kumar's algorithm. (c) Minimal wavelength assignment requires $5 \times 200 = 1000$ miles. (d) An alternative tree. (e) Minimal wavelength assignment requires $2 \times 200 + 2 \times 250 = 900$ miles, which is smaller than the wavelength cost of (c).

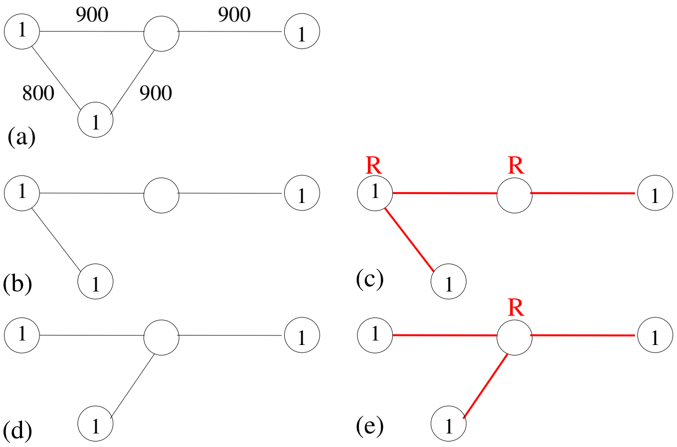


Figure 8.4: An example showing that Kumar's algorithm does not apply to the REGEN placement problem. (a)Physical graph of a network. The nodes are marked with the number of ports available and links are marked with distance in miles. Assume that reachability is 932 miles. (b) The lowest-cost tree created applying Kumar's algorithm. (c) Minimal wavelength and REGEN placement requires 2 REGENs and 3 wavelengths. Total costs $0.07(2 \times 900 + 800) + 2 \times 150 = 482$. (d) An alternative tree. (e) Minimal wavelength and REGEN placement requires 1 REGEN and 3 wavelengths. Total costs $0.07(3 \times 900) + 150 = 339$, which is smaller than the wavelength cost of (c).

ports. Figure 8.7 shows that the number also grows exponentially in terms of the number of nodes. I now develop techniques to reduce the number of demand matrices used for optimization.

Port Number Bound

Figure 8.8 shows growth in the number of demand matrices versus one node's port count, with other nodes' ports staying fixed. The number of demand matrices converge to a fixed number, 2609, when $O_3 \geq 15$. The reason is obvious: the 16th port cannot find a peer port to connect to if all other 15 ports are connected. In practice, no customer will buy a 16th port.

Maximal Demand Matrix

The solutions to some demand matrices will apply to a large number of other demand matrices. For example, assume the port constraint for three nodes is $O_0 = 2, O_1 = O_2 = 1$. If a demand matrix $D = \{(0,1),(0,2)\}$ is satisfied, demand matrices $D_1 = \{(0,1)\}, D_2 = \{(0,2)\}$ that are subsets of D must also be satisfied. Therefore, I do not need to include D_1 or D_2 if D is included in the optimization. I define a maximal demand matrix as the following.

Definition 7 (Maximal demand matrix)**.** *A maximal demand matrix is a demand matrix to which no more demands can be added without removing an existing demand.*

Maximal demand matrices satisfy a "same node rule" for spare OTs. The proof is easy: if two nodes have spare OTs, at least one more new demand can be created by connecting their spare ports.

Lemma 1 (Same node rule)**.** *$\forall M \in \mathcal{M}$, at most one node that has spare OTs can exist in M.*

The number of maximal demand matrices is often smaller than the total number by two orders of magnitude, as shown in Figures 8.5–8.8. Although the growth of maximal demand matrices is much slower than the total number, it is still exponential in the number of ports for some cases.

Reduced Demand Matrix

I can further reduce the number of demand matrices by using the spare OTs in one demand matrix as REGENs. Figure 8.9 shows that D_1 can be reduced into D_2. The red demand in D_1 can be covered by the red demands in D_2. The blue demand in D_1 can be covered by the blue demand in D_2. Therefore,

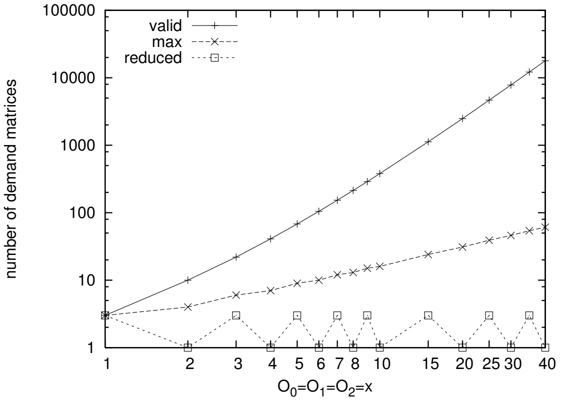


Figure 8.5: Semi-log plot for demand-matrix counts for three nodes with the same number of ports: $O_0 = O_1 = O_2 = x$. The total number of demand matrices and maximal demand matrices grows exponentially in their number of ports. The number of reduced demand matrices follows Theorem 3.

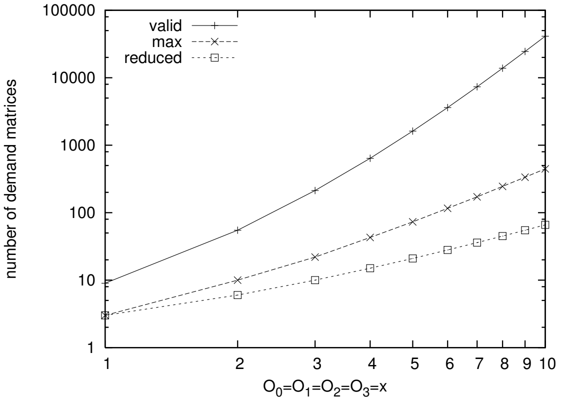


Figure 8.6: Semi-log plot for demand-matrix counts for four nodes with the same number of ports: $O_0 = O_1 = O_2 = O_3 = x$. All numbers grow exponentially in their number of ports.

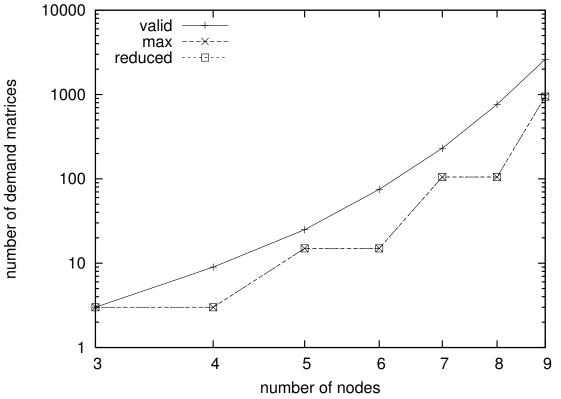


Figure 8.7: Semi-log plot for demand-matrix counts for x nodes with one port each. $\forall x, O_x = 1$. The numbers grow exponentially in their number of nodes.

Table 8.1: Comparison of demand-matrix counts.

O_0	O_1	O_2	O_3	**total**	**max**	**reduced**
1	1	1	0	3	3	3
2	2	2	0	10	4	1
3	3	3	0	22	6	3
4	4	4	0	41	7	1
5	5	5	0	68	9	3
6	6	6	0	105	10	1
7	7	7	0	153	12	3
8	8	8	0	214	13	1
9	9	9	0	289	15	3
10	10	10	0	380	16	1
15	15	15	0	1123	24	3
20	20	20	0	2485	31	1
25	25	25	0	4653	39	3
30	30	30	0	7815	46	1
35	35	35	0	12158	54	3
40	40	40	0	17870	61	1
5	5	5	0	68	9	3
5	5	5	1	239	21	3
5	5	5	2	512	42	21
5	5	5	3	863	52	10
5	5	5	4	1247	79	55
5	5	5	5	1619	73	21
5	5	5	6	1933	100	66
5	5	5	7	2179	88	15
5	5	5	8	2353	100	45
5	5	5	9	2469	86	10
5	5	5	10	2538	89	28
5	5	5	11	2577	77	6
5	5	5	12	2596	77	15
5	5	5	13	2605	71	3
5	5	5	14	2608	71	6
5	5	5	15	2609	69	1
2	1	1	0	4	2	1
3	2	2	0	12	4	3
4	3	3	0	26	5	1
5	4	4	0	47	7	3
3	2	2	1	39	8	3

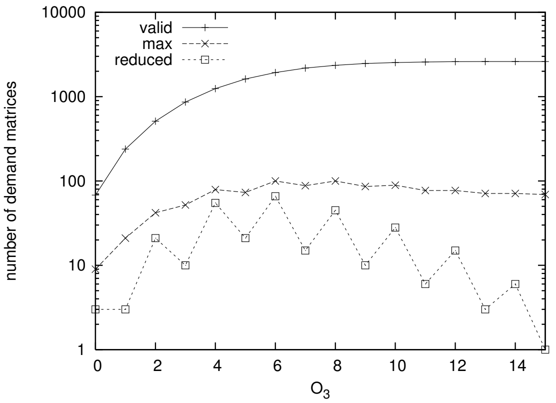


Figure 8.8: Log-linear plot for demand-matrices counts for four nodes with $O_0 = O_1 = O_2 = 5, O_3 = x$. The numbers stay the same for $O_3 \geq O_0 + O_1 + O_2$.

if D_2 is satisfied, D_1 can be satisfied by connecting the blue demand on the route of the blue demand in D_2 and connecting the red demand on the routes of the two red demands in D_2. The routes for the red demands are connected at node 2 by pairing the spare two OTs that are not used in D_1. Since I pair the two OTs as a REGEN, the two routes can be connected regardless of total distance and wavelength colors. Therefore, D_1 is also satisfied.

The condition for reducing one maximal demand matrix to another is explained in the following.

Definition 8 (Reduction)**.** *Let $A \in \mathcal{M}$ be a maximal demand with spare OTs at node n^o, and let $B \in \mathcal{M}$ be another maximal demand matrix. The demands of the same node pairs in A and B, denoted by $A \cap B$, are automatically reduced. The demands left to be reduced in A are denoted by $\alpha = A \setminus (A \cap B)$. The demands left in B are denoted by $\beta = B \setminus (A \cap B)$. If β contains all the demands required to reduce all demands in α by using OTs at node n^o, s.t. $\bigcup_{(i,j)\in\alpha}\{(i,n^o)\bigcup(j,n^o)\}^{|\Omega_{(i,j)}|} \subseteq \beta$, where $\Omega_{(i,j)} = \{d \in \alpha | N(d) = (i,j)\}$ is the number of demands of pair (i,j) in α, A can be reduced into B.*

Then, I have a necessary and sufficient condition to identify reduced demand matrices.

Theorem 1. *A demand matrix is a reduced demand matrix if and only if the demand matrix has one extra OT port for constraints with an odd total number of ports and zero extra ports for constraints with an even total number of ports.*

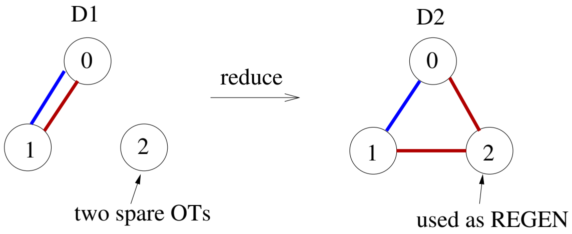


Figure 8.9: Reduce maximal demand matrices. The port numbers on the three nodes are $O_0 = O_1 = O_2 = 2$. The maximal demand matrix $D_1 = \{(0,1),(0,1)\}$ is illustrated graphically by two links between node 0 and 1. The maximal demand matrix $D_2 = \{(0,1),(0,2),(1,2)\}$ is also illustrated graphically. D_1 is reduced into D_2 by mapping the blue demand in D_1 to the blue demand in D_2 and mapping the red demand in D_1 to the red demands in D_2. The two spare OTs at node 2 for demand D_1 are connected as a REGEN to make the connection for the red demand. In fact, all other maximal demands for the given port constraint can be reduced to D_2.

Proof. For necessity, I first show that a reduced demand matrix has no more than one extra port. If a node n has two or more extra ports but no more reductions can be made, all existing demands must connect to that node. Therefore, $O_n \geq \sum_{i \neq n} O_i$, which violates Equation 8.3. With an even total number of ports, having one extra port is impossible, so no extra port exists. With an odd total number of ports, all demand matrices have at least one extra port, so exactly one extra port exists. For sufficiency, with fewer than two extra ports, a demand matrix cannot be further reduced, so the matrix must be a reduced demand matrix. □

If a reduced demand matrix finds a route allocation, all demand matrices that can be reduced into this one can use the same allocation. It proves Theorem 2 that I need consider only reduced demand matrices for resource dimensioning.

Theorem 2 (Satisfactory). *If demand matrix A can reduce into B, demand matrix A must be satisfied if demand matrix B is satisfied.*

In fact, the reduction idea leverages spare OTs in some demand matrices as REGENs in order to shrink both the total resource cost and the optimization space. With this technique, fewer demand matrices are considered for optimization, but the optimal result stays the same.

Reduction shrinks the number of demand matrices by another order of magnitude relative to the number of maximal demand matrices (see Figures 8.5–8.8).

The number of reduced demand matrices forms a periodic pattern in some cases (see Figures 8.5 and 8.8). In particular, for three nodes, I have the following theorem. Let $\mathcal{S}$ be the set of reduced demand matrices.

Theorem 3. *Assume the sum of ports in a port constraint of three nodes $O_0 + O_1 + O_2$ is even, $|\mathcal{S}| = 1$. If $O_0 + O_1 + O_2$ is odd, $|\mathcal{S}| = 3$.*

Proof. For three nodes, if there is an even number of OT ports, s.t. $O_0 + O_1 + O_2 = 2k$ where $k \geq 0$, a demand matrix can maximally have k demands. Let x be the number of demands between nodes 0 and 1. Let y be the number of demands between nodes 0 and 2. Let z be the number of demands between nodes 1 and 2. If these demands consist of a maximal demand matrix and contain exactly k demands, x, y, z must satisfy the following equalities, $x + y = O_0, x + z = O_1, y + z = O_2$, such that all ports are used. Solving the equation set, the maximal demand matrix S is solved: $x = \frac{1}{2}(O_0 + O_1 - O_2), y = \frac{1}{2}(O_0 + O_2 - O_1), z = \frac{1}{2}(O_1 + O_2 - O_0)$. The solution is unique. It can next be shown that all other maximal demand matrices can be reduced into this demand matrix S. If a maximal demand matrix contains $l < k$ demands, there are $2(k - l)$ spare OTs in the graph. In order to satisfy the condition of a maximal demand matrix, these spare OTs must be located at one node. If node 0 has $2(k - l)$ spare OTs, I can route $(k - l)$ demands between nodes 1 and 2 to the demand route of (0, 1) and (0, 2). Then, all spare OTs are used, and the reduced demand matrix is the same as S because of uniqueness. Figure 8.9 is such an example: D_2 is the only reduced demand matrix for $O_0 = O_1 = O_2 = 2$.

If there is an odd number of ports, the total number is $O_0 + O_1 + O_2 = 2k + 1 = 2(k - 1) + 3$ where $k \geq 1$. The demand matrix can maximally have k demands, and one spare OT is left. If one port at each node is removed, the total number becomes even, which is $2(k - 1)$. According to the previous discussion, a unique reduced demand matrix S' that contains $k - 1$ demands can be found for the even portion of ports. Then, the remaining three ports at one node each can create the last demand. The last demand thus combines with S' to form a reduced demand matrix S for all ports. Since there are three ways to create the last demand, there are three reduced demand matrices.

□

The fact that only one demand matrix needs to be considered for an even port count is extremely useful for the optimization. With only one demand matrix, no sharing needs to be considered for optimization. The lower bound computed by `LowBound` (discussed in Section 8.4.1) for this case is thus the optimal result. A huge amount of computation time can be saved for this special case. However, I have not found useful reduction patterns for more than three nodes.

Figure 8.7 shows that the number of reduced demand matrices is still exponential in the number of nodes. Specifically, Theorem 4 says that the growth is at least on the order of a factorial.

Theorem 4. *Consider $n \geq 3$ nodes with $O_i = 1$ for all $0 \leq i \leq n-1$. Let a_n be the number of reduced demand matrices for n nodes. Then, $a_n = n(n-2)(n-4)\cdots 1$ for n odd and $a_n = (n-1)(n-3)\cdots 1$ for n even.*

Proof. For an odd number of nodes n, there are n ways to choose one node out of n. After this node is chosen, all other nodes can be paired. There is a total of

$$\frac{\binom{n-1}{2}\binom{n-3}{2}\cdots 1}{(\frac{n-1}{2})!}$$

ways of pairing. Canceling $\frac{n-1}{2}, \frac{n-3}{2}, \ldots$ with the denominator, the total is $(n-2)(n-4)\cdots 1$. Combined with the choices of the leftover node, the number of reduced demand matrices is $n(n-2)(n-4)\cdots 1$. In the same way, if n is even, there are $(n-1)(n-3)\cdots 1$ ways of pairing up. □

8.3.2 Problem Decomposition

To solve an optimization problem, I must devise a plan to walk though the solution space. For this particular optimization, I decompose the entire problem into a wavelength assignment problem and a route allocation problem. Basically, I first choose a virtual-link graph route without wavelength assignments for every demand. The aggregation of all demand-route pairs is called a *route allocation*. Let $\mathcal{A}$ be the set of all route allocations without wavelength assignments. For each allocation, I find the minimum cost wavelength assignment A^w. A^w is, in fact, a set of solved $X_{p,d}$s. The cost of A^w,

Algorithm 8.1: Problem decomposition.

```
foreach Possible allocation $A \in \mathcal{A}$ do
    Compute the optimal wavelength assignment $A^w$ with minimal cost $C(A^w)$;
Pick the route allocation that generates the lowest wavelength assignment cost
$A_{min} \leftarrow \arg\min_{A \in \mathcal{A}} C(A^w)$;
```

a.k.a $C(A^w)$, can be computed by Equation 8.4, where $W_{e,w}$ and R_n equal the left-hand sides of Equation 8.7 and Equation 8.6 with the solved $X_{p,d}$. Finally, I choose the route allocation that generates the minimal cost.

Algorithm 8.1 shows the decomposition. Heuristic algorithms in later sections are built on this algorithm. There are many benefits from this decomposition. I can focus on smaller problems and develop better optimization techniques for each. The decomposition also helps us understand the problem structure better and trim the space more efficiently later for heuristic optimizations.

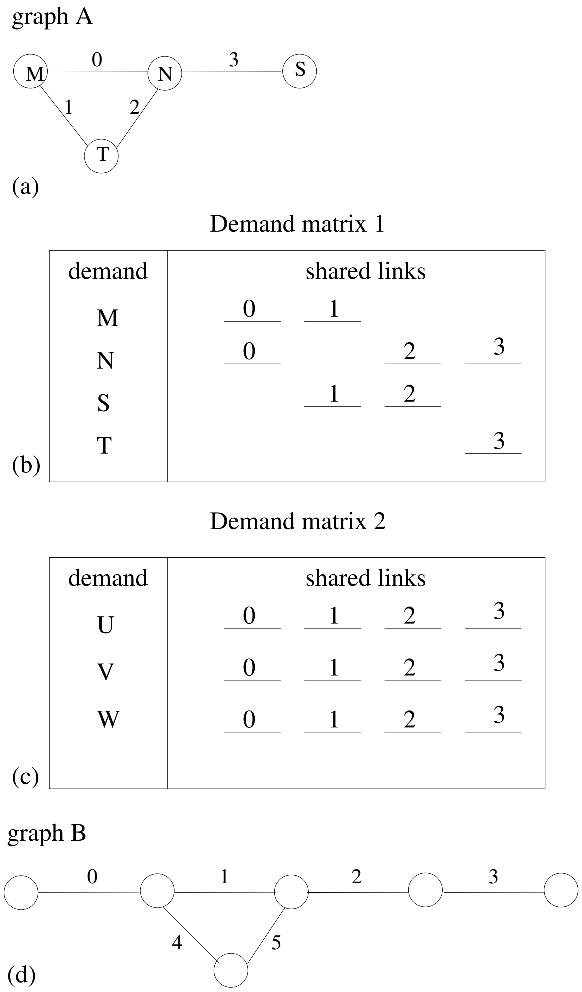
Wavelength Assignment

Given the set of route allocation for all demands, finding the minimal cost wavelength assignment is an NP-hard problem. The proof uses a reduction from the known NP-complete k-color decision problem and shows that the minimal cost wavelength assignment problem is at least as hard as the k-color problem.

Theorem 5. *The optimal wavelength assignment problem is NP-hard.*

Proof. The decision problem of k-color is phrased as the following. Given a graph and k distinct colors, can each node in the graph be labeled by a color in k such that any adjacent nodes have different colors? The effort of proving NP-completeness is to reduce the k-color problem into a decision wavelength assignment problem, which includes create demands, demand matrices, and routes.

For clarity, let me refer the original k-color graph as graph A and the reduced network graph as graph B. The entire reduction is illustrated in Figure 8.10. First, I create Demand matrix 1. For a given graph A, each node is mapped into a demand in Demand matrix 1. For each link in graph A, create a physical link in graph B. Then, allocate this link into the wavelength routes of the two demands that map to the adjacent nodes of the link in graph A. In order to create a realistic graph B and continuous routes for each demand, additional links may be arbitrarily added into demand routes,



Demand matrix 1

demand	shared links			
M	0	1		
N	0		2	3
S		1	2	
T				3

Demand matrix 2

demand	shared links			
U	0	1	2	3
V	0	1	2	3
W	0	1	2	3

Figure 8.10: Illustration of reduction of a 3-color problem to a wavelength assignment problem. (a) The original 3-color graph A. (b) Create one demand per node for Demand matrix 1. Assign each link in A into the routes of demands that are adjacent to the link. For example, link 0 of A is assigned to demand 0 and 1. (c) Create Demand matrix 2 that has 3 demands, each is assigned to a route includes all shared links in Demand matrix 1. The number of demands created matches the number of colors. (d) The physical link graph B that is created based on graph A. Links 4 and 5 are not shared and do not map to any link in graph A. Demand route (physical links) allocation is as the following: Demand M (0, 1), Demand N (0, 4, 5, 2, 3), Demand S (1, 2), Demand T (3), Demand U (0, 1, 2, 3), Demand V (0, 1, 2, 3), Demand W (0, 1, 2, 3). Note that each route is on a virtual link that includes all listed physical links. As Demand matrix 2 requires at least 3 wavelengths on all shared links, the min-cost wavelength assignment problem is whether 3 wavelengths on Links 0, 1, 2, and 3 are sufficient for Demand matrix 1.

given that these links do not map to any of the links in A and must belong to only one demand. Therefore, only these links that have a mapping in A are shared by demands.

Second, I create Demand matrix 2. Demand matrix 2 includes k demands. Each demand is allocated with a route that includes all shared physical links appearing in Demand matrix 1.

Further, for each demand, construct a virtual link that at least includes all the links that map to A. (Assume that each link in B has the same distance and the distance is small enough that the entire route of any simple route in B is reachable.) This constraint ensures that all physical links included in the virtual link must be assigned to the same wavelength. In this way, whenever two demands share the same physical link, the virtual link that includes the physical link can not be assigned to the same wavelength as they are in the same demand matrix. The demand routes in the second demand matrix are also constraint to virtual links. The setup of Demand matrix 2 guarantees that at least k wavelengths on each shared physical links must be assigned. The smaller number of wavelengths used, the smaller cost of wavelength resources are allocated on graph B as all links in B are of the same distance. For a min-cost assignment, Demand matrix 1 should maximize sharing to the allocated k wavelengths and avoid extra wavelengths.

Then, the reduced wavelength assignment problem is whether k wavelengths is sufficient on these links in B that have a mapping in A. The original k colors maps to k wavelengths. These links in B that do not map to A are not shared, so their wavelength assignment does not affect the optimal value. For each demand, the physical links that have a mapping in A have to be assigned to the same wavelength, and the wavelength is different from that of any other demand that shares physical links with this demand. If there is an assignment that does not exceed k wavelengths, the original graph A can be k-colored. The wavelength that is assigned to each demand for their shared links is the color that the corresponding node in A is labeled. The solution guarantees that adjacent nodes in A are not labeled in the same color. The reduction is polynomial time as one needs only to go through all nodes and links in A once. Once the k-wavelength decision problem is solved (a.k.a. the min-cost assignment problem with two demand matrices), mapping the solution to the k-color problem is trivial.

□

Route Allocation

The route space is difficult to trim while retaining optimality due to the complexity of sharing potential between demand matrices. To find an optimal route allocation, I must try to combine routes for all demands. Example 1 illustrates this complexity.

Example 1. *Using Figure 8.1, assume this time that* $l_0 = 800, l_1 = 800$*, and* $l_2 = 800$ *(in miles). Then,* $l_3 = 1600$ *is unreachable. Let the customer's new port constraint be* $O_0 = 2, O_1 = 1, O_2 = 1$*. Trivially, the optimal solution is still a treelike structure, which is* $W_{0,1} = W_{1,1} = 1$*, or* $W_{i,j} = 0, otherwise$*. To connect the demand between nodes (1, 2), the two spare customer OTs at node 0 is used as a REGEN. However, when the customer ports increase to* $O_0 = 3, O_1 = 2, O_2 = 2$*, the optimal solution is no longer a tree, which is* $W_{0,1} = W_{1,1} = W_{2,1} = 1$*, or* $W_{i,j} = 0, otherwise$*. For the two demands between (1, 2), one uses link 2 and the other uses a route through links 1 and 0. In general, a large number of routes must be considered to find an optimal solution.*

8.4 Lower Bound Computation and Heuristic Optimization

Structurally, the problem is a discrete combinatorial optimization problem. Finding an optimal solution is challenging because of the problem size. The number of possible routes per connection grows exponentially in terms of network size. In general, the number of demand matrices after reduction may also grow exponentially in terms of ports and nodes used by the customer. The permutation space for wavelength assignments grows exponentially in demand.

I now propose a useful lower bound and introduce the heuristic algorithms for the wavelength assignment and route allocation problems. I also discuss a few space-limiting heuristics that help to solve the problem faster. Finally, a greedy algorithm that provides fast computation for large problems is developed.

8.4.1 Lower Bound

A lower bound that is easy to compute is useful in practice. For this problem, I can compute the minimal resource cost $C_{min}(D)$ for each demand matrix D independently. The maximum of $C_{min}(D)$ over $D \in \mathcal{D}$ is the necessary cost to support all Ds and is thus a lower bound for the problem. Since the demands of the same demand matrix cannot share wavelength capacity, the total cost for a demand is at least the sum of the cost of each route in each demand. To find the minimal cost of each D, I simply choose the least-cost route for each of its demands.

Finding the maximum cost demand matrix can be considered as an ILP problem. Let P be the set of all possible node pairs. Let c_i be the minimum cost route for pair $i \in P$. Let x_i be the integer variable indicating the number of demands created for the pair i. The maximum cost demand matrix is constructed by finding $x_i, \forall i \in P$ such that

$$\text{maximize} \sum_{i \in P} x_i c_i \tag{8.8}$$

The sum of demands for each node must observe the port constraint.

$$\forall n \in N, \sum_{i \in P, n \in i} x_i \leq O_n \tag{8.9}$$

The problem space is exponential, as I must check every possible combination of ports (effectively finding all possible demand matrices). Therefore, checking the cost of all reduced demand matrices is a better solution. The cost of the combined routes through a third node (using OTs for a REGEN as needed) is always greater than or equal to the least-cost route between the source and destination nodes. Therefore, checking only the reduced demand matrices gives the same lower bound as checking all demand matrices. Procedure `LowBound` shows an algorithm. However, the number of reduced demand matrices still grows exponentially although at a much smaller rate. I can further reduce the average time complexity of the algorithm by combining ILP and `LowBound`. The idea is to first solve a relaxed problem (with real coefficients), which is polynomial time. Then, I have a maximum cost with nonintegral optimal $\hat{x}_i$ values. The maximum cost is essentially an upper bound for the optimal integer solutions. I can use the upper bound to enu-

Procedure LowBound–Compute the lower bound C_{lb} of the network resource cost.

```
Input: Reduced demand matrix set $\mathcal{S}$
Output: The lower bound $Clb$
foreach reduced demand matrix $D \in \mathcal{S}$ do
    Initialize $C_{min}(D) \leftarrow 0$;
    foreach demand $d \in D$ do
        Get the least-cost path cost $C_{min}(d) \leftarrow \min_{p \in \mathcal{P}_{N(d)}} C(p)$;
        Add the cost $C_{min}(D) \leftarrow C_{min}(D) + C_{min}(d)$;
The lower bound $C_{lb} \leftarrow \max_{D \in \mathcal{S}} C_{min}(D)$;
```

merate each reduced demand matrix in `LowBound`. If the cost of a demand matrix equals the upper bound, the demand matrix is the "maximum cost demand matrix" and I do not need to check the rest. However, if the upper bound is loose, I must still check all demand matrices.

Computing the lower bound is polynomial time in the number of reduced demand matrices and network size. Although the number of reduced demand matrices grows exponentially in some cases, the complexity has been greatly reduced compared to the original optimization problem.

8.4.2 Wavelength Assignment

I developed a simulated annealing version to solve the problem heuristically. Trying the combination of all wavelengths for each route is not an efficient approach. Only a few wavelengths are needed on each link when the traffic demand is low. Instead, I propose a heuristic wavelength assignment algorithm. The idea is to use a "first-fit" wavelength assignment strategy to an ordered list of virtual link and demand tuples. For each virtual link in the route to which the demand is allocated, a tuple is created. The tuple is a wavelength assignment unit. We use virtual links because a route can have different wavelengths on each virtual-link segment. Each permutation of the order of tuples determines wavelength assignment priority. Without loss of generality, I assign the wavelength to tuples in left-to-right order. The wavelength assignment for a permutation is explained in `AssignWavelength`.

Figure 8.11 illustrates an example. According to the route allocation, the list contains four tuples. If the wavelength assignment follows order t_1, t_2, t_3, t_4, `AssignWavelength` gives the result: $t_1 = (v_1, d_{0,0}, 1), t_2 = (v_2, d_{0,0}, 1), t_3 = (v_3, d_{0,1}, 2), t_4 = (v_4, d_{1,0}, 1)$. The solution is suboptimal be-

Procedure AssignWavelength–Assign wavelengths to a permutation.

```
Input: Virtual link and demand list $L$
Input: List permutation $l$
Output: Wavelength assignment $A_l^w$
foreach element $(v, d, \emptyset) \in L$ iterated in the order of $l$ do
    foreach wavelength $w$ from 1 to $|\mathcal{W}|$ do    /* Maximize sharing by using the same iteration order */
        if $w$ on all physical link $e \in v$ is yet not assigned to any other demand in the same demand matrix then /* Demands in the same demand matrix cannot share wavelengths on the same physical link. */
            Assign wavelength $A_l^w \leftarrow A_l^w \cup \{(v, d, w)\}$;
            break;
```

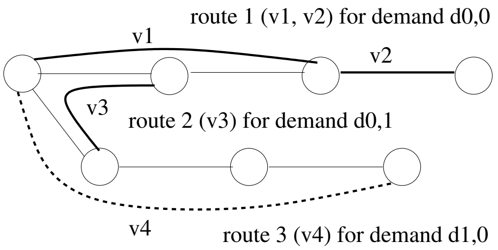

route 1 (v1, v2) for demand d0,0
v1
v2
v3
route 2 (v3) for demand d0,1
v4
route 3 (v4) for demand d1,0

Figure 8.11: Wavelength assignment for the routes of three demands. Relevant virtual links are marked with numbers. Route 1 uses two virtual links, v_1 and v_2. Route 2 uses virtual link v_3. Route 3 uses virtual link v_4. The problem contains three demands, denoted by $d_{s,t}$, from two demand matrices. Route 1 is allocated to demand $d_{0,0}$. Route 2 is allocated to demand $d_{0,1}$. Route 3 is allocated to demand $d_{1,0}$. Four tuples with empty wavelength assignments are:
$t_1 = (v_1, d_{0,0}, \emptyset), t_2 = (v_2, d_{0,0}, \emptyset), t_3 = (v_3, d_{0,1}, \emptyset)$, and $t_4 = (v_4, d_{1,0}, \emptyset)$. t_1, t_3 must not share wavelengths because they share both demand matrix and physical link.

cause t_3 and t_4 use different wavelengths, which occurs because of the first-fit rule. However, if I assign t_3 first using the permutation t_3, t_1, t_2, t_4, the algorithm finds an optimal result.

I define a permutation of the virtual-link-demand list as a solution. A neighbor solution is created by swapping any two virtual-link-demand tuples. Depending on the temperature and the new cost, I decide whether or not to move to the neighbor for the next iteration. Procedure `SaWaveAssign` explains the details. The temperature-dropping schedule is determined experimentally. In my experience, the solution is not sensitive to minor changes in schedule.

The initial solution uses a permutation sorted by the count of physical links. The link distance is used to break ties. The heuristic gives longer links higher priority to assign wavelength. It increases the probability of sharing for many cases but is not optimal for all. Assume the link distance relationship in Figure 8.11 to be $l_{v_4} > l_{v_1} > l_{v_3} > l_{v_2}$. The heuristic link assignment order is t_4, t_1, t_3, t_2, which is not optimal.

8.4.3 Route Allocation

I introduce two algorithms for route allocation. The algorithms use `SaWaveAssign` for the wavelength assignment subproblem and generate the final heuristic optimization results.

Simulated Annealing

The simulated annealing algorithm starts with allocating least-cost routes to all demands. A neighbor solution is generated by randomly changing the route of a random demand drawn uniformly over all demands. The optimal wavelength assignment cost of the allocation defines the cost of the allocation. Again, depending on the temperature and the new cost, I decide whether or not to move to the neighbor for the next iteration. Procedure `SaOptimize` explains the details. I use the same temperature-drop schedule as in `SaWaveAssign`.

Genetic Algorithm

If many demands require longer routes for optimal solutions, the searching speed of simulated annealing can be too slow since only one route is changed in each step. In contrast, a genetic algorithm can explore more demand-route combinations by generating a group of route allocations in each generation. These allocations are evaluated and merged to create the next generation.

Procedure `GaOptimize`, shows the algorithm. Each demand-route tuple represents a *gene*. The route set for the demand represents all possible choices for this gene. A route allocation of all demands represents a *chromosome*. A chromosome also maps to a solution. A set of chromosomes forms a *generation*. In my implementation, a generation is represented by a list of solution and cost tuples. The first generation is a set of randomly chosen allocations. Further generations are created by mating chromosomes from the previous generation. The mating candidate is chosen using the `RouletteWheelPick`

Procedure SaWaveAssign–Simulated annealing wavelength assignment for route allocation.

```
Input: Allocation $A$ for all demands
Input: Optimization steps $k$
Output: Optimal wavelength assignment $A^w$
Empty list $L \leftarrow \emptyset$;
foreach demand $d \in D \in \mathcal{S}$ do
    Get the route allocation $p \leftarrow A(d)$;
    foreach virtual link $v \in p$ do
        Create a link-demand tuple $(v, d, \emptyset)$;
        $L \leftarrow L \cup \{(v, d, \emptyset)\}$;
/* Initial wavelength assignment */
Get permutation $l$ by sorting $L$ in descending order of number of physical links
and distance in $v$;
$A_l^w \leftarrow$ AssignWavelength$(L, l)$;
/* Compute the lower bound */
Get lower bound $C_{lb} \leftarrow$ LowBound$()$;
Get initial cost $c \leftarrow C(A_l^w)$;
Initialize minimum cost $c_m \leftarrow c$;
/* Temperature schedule */
Set start temperature $T \leftarrow 10c$;   /* Start temperature is set 10 times the
initial cost. */
Set minimal temperature $T_m \leftarrow 0.0001$; /* It is extremely unlikely to move
to a higher cost neighbor at this temperature. */
Linear temperature drop step $\Delta T \leftarrow \frac{T - T_m}{k}$;
while $k \geq 0$ and $c > C_{lb}$ do
    Create a new permutation $l'$ by swapping the order of two uniformly chosen
    tuples in $l$;
    $A_{l'}^w \leftarrow$ AssignWavelength$(L, l')$;
    New cost $c' \leftarrow C(A_{l'}^w)$;
    if $c' < c_m$ then
        $c \leftarrow c'$;
        $l \leftarrow l'$;
        /* Save the new minimum */
        $c_m \leftarrow c'$;
        $A^w \leftarrow A_{l'}^w$;
    else
        Get a uniform random real number $x \in [0, 1)$;
        if $e^{-\frac{c'-c}{T}} > x$ then   /* Move to the neighbor. If $c' < c$, always
        move. */
            $c \leftarrow c'$;
            $l \leftarrow l'$;
    Update temperature $T \leftarrow T - \Delta T$;
    Update steps $k \leftarrow k - 1$;
```

Procedure SaOptimize–Simulated annealing route allocation optimization.

```
Input: Reduced demand matrix set $\mathcal{S}$
Input: Optimization steps $k$
Input: Optimization steps for wavelength assignment $k'$
Output: Optimal allocation and wavelength assignment $A^w$
/* Compute the lower bound */
Get lower bound $C_{lb} \leftarrow$ LowBound();
/* Initial path allocation */
foreach demand $d \in D \in \mathcal{S}$ do
    $A(d) \leftarrow \arg\min_{p \in \mathcal{P}_{N(d)}} C(p)$;
Get the wavelength assignment $A^w \leftarrow$ SaWaveAssign($A, k'$);
Get initial cost $c \leftarrow C(A^w)$;
Initialize minimum cost $c_m \leftarrow c$;
/* Temperature schedule */
Set start temperature $T \leftarrow 10c$;   /* Start temperature is set 10 times the initial cost. */
Set minimal temperature $T_m \leftarrow 0.0001$; /* It is extremely unlikely to move to a higher cost neighbor at this temperature. */
Linear temperature drop step $\Delta T \leftarrow \frac{T - T_m}{k}$;
while $k \geq 0$ and $c > C_{lb}$ do
    Uniformly choose a demand $d' \in D \in \mathcal{S}$;
    Uniformly choose a new route $p \neq A(d') \in \mathcal{P}_{N(d')}$;
    Get a new allocation $A'$ by $A'(d') \leftarrow p, A'(d) \leftarrow A(d), \forall d \neq d'$;
    Get the wavelength assignment $A'^w \leftarrow$ SaWaveAssign($A', k'$);
    New cost $c' \leftarrow C(A'^w)$;
    if $c' < c_m$ then
        $c \leftarrow c'$;
        $A \leftarrow A'$;
        /* Save the new found minimum */
        $c_m \leftarrow c'$;
        $A^w \leftarrow A'^w$;
    else
        Get a uniform random real number $x \in [0, 1)$;
        if $e^{-\frac{c'-c}{T}} > x$ then     /* Move to the neighbor. If $c' < c$, always move. */
            $c \leftarrow c'$;
            $A \leftarrow A'$;
    Update temperature $T \leftarrow T - \Delta T$;
    Update steps $k \leftarrow k - 1$;
```

Procedure GaOptimize–Genetic algorithm route allocation optimization.

```
Input: Reduced demand matrix set $\mathcal{S}$
Input: Generations $k$
Input: Population per generation $s$
Input: Optimization steps for wavelength assignment $k'$
Output: Optimal allocation and wavelength assignment $A^w$
Crossover rate $\alpha \leftarrow 0.6$;
Mutation rate $\beta \leftarrow 0.001$;
/* Compute the lower bound                                          */
Get lower bound $C_{lb} \leftarrow$ LowBound();
Initialize minimum cost $c_m \leftarrow \infty$;
/* Generate a random generation of $s$ allocations                  */
for $i$ from 1 to $s$ do
    foreach demand $d \in D \in \mathcal{S}$ do
        $A_i(d) \leftarrow p$ where $p$ is uniform randomly chosen in $\mathcal{P}_{N(d)}$;
    Get the wavelength assignment $A_i^w \leftarrow$ SaWaveAssign($A_i, k'$);
    Compute cost $c_i \leftarrow C(A_i^w)$;
    if $c_i < c_m$ then                          /* Record the minimum cost */
        $c_m \leftarrow c_i$;
        $A^w \leftarrow A_i^w$;
while $k > 0$ and $c_m > C_{lb}$ do
    /* Generate a new generation of $s$ allocations                 */
    for $i$ from 1 to $s$ do
        /* Mating algorithm                                         */
        Get the father $m \leftarrow$ RouletteWheelPick($\{(A_i^w, c_i)\}^s$);
        Get the mother $n \leftarrow$ RouletteWheelPick($\{(A_i^w, c_i)\}^s$);
        Get a uniform random real number $r \in [0, 1)$;
        if $r > \alpha$ then
            Get a uniform random integer $a \in (1, |A_m|)$;
            Create a child by merge two parents $A_i' \leftarrow A_m[1 : a] + A_n[a + 1 : |A_m|]$;
        else
            Copy the father to the child $A_i \leftarrow A_m$;
        /* Mutation algorithm                                       */
        foreach demand $d \in D \in \mathcal{S}$ do
            Get a uniform random real number $r \in [0, 1)$;
            if $r < \beta$ then
                $A_i'(d) \leftarrow p$ where $p$ is uniformly chosen in $\mathcal{P}_{N(d)}$;
        Get the wavelength assignment $A_i'^w \leftarrow$ SaWaveAssign($A_i', k'$);
        Compute cost $c_i' \leftarrow C(A_i'^w)$;
        if $c_i' < c_m$ then                     /* Record the minimum cost */
            $c_m \leftarrow c_i'$;
            $A^w \leftarrow A_i'^w$;
    Update the generation $\{A_i^w, c_i\}^s \leftarrow \{A_i'^w, c_i'\}^s$;
    Update steps $k \leftarrow k - 1$;
```

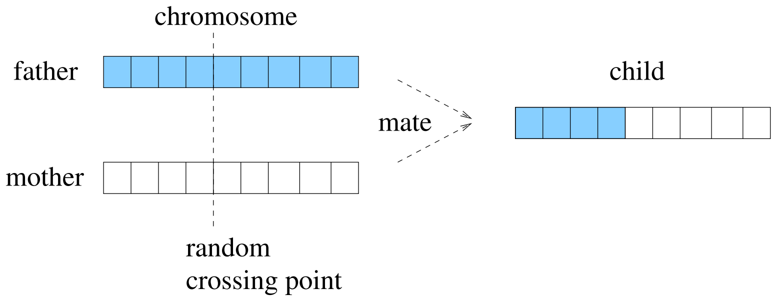


Figure 8.12: Illustration of mating algorithm used in `GaOptimize`.

Procedure RouletteWheelPick–Pick chromosomes from a generation.

```
Input: Generation $\{(A_i^w, c_i)\}^s$
Output: Picked candidate $t$
Generate a uniform random real value $r \in [0, 1)$;
Initialize the cumulative probability density $P_1 \leftarrow 0$;
for $i$ from 1 to $s$ do
    Cumulative probability density $P_i \leftarrow P_i + \frac{1}{c_i \sum_{0 \leq j \leq s} \frac{1}{c_j}}$;
    if $r \leq P_i$ then                              /* Pick the candidate. */
        $t \leftarrow i$;
        return;
```

algorithm. The choosing probability is proportional to the inverse of the cost value of the solution. Therefore, low-cost solutions are more likely to pass to the next generation, while high-cost solutions still have some chance to get chosen. Two chromosomes mate in a probability of a *crossover* rate. Mating combines a random upper half of the father's chromosome with the random lower half of the mother's chromosome.

Figure 8.12 illustrates the mating algorithm. If two chromosomes do not mate, the father's chromosome is used. Each gene of a child is subjected to mutation according to a *mutation* rate. If a gene is selected for mutation, a random route is reselected for the demand. In the algorithm, the crossover rate is set to 0.6. The mutation rate is set to 0.001. The algorithm is not sensitive to minor changes of these numbers.

8.4.4 Route Space Reduction

Even though I use efficient heuristic optimization techniques, the number of routes for each demand can be large. This section introduces space-limiting techniques that help to find good results fast.

Limiting Route Cost

Except in extreme cases, expensive routes are unlikely to be included in an optimal solution. However, searching too few routes can reduce the quality of the results. On CORONET, our testing network, I find it to be a good tradeoff of efficiency and quality to limit the search space of routes for each demand to within 1.4 times the cost of the demand's least-cost route.

Fixing Route per Node Pair

Even if two demands have the same end nodes, they can use different routes. In many cases, these demands must use different routes in an optimal solution. For example, the red and blue demands in Figure 8.9 use different routes in the optimal solution. Demands that have been reduced to other demands, such as the red demand in D_1 that reduces to two red demands in D_2, need not be considered by the optimizer. Demand matrix reduction has already taken care of the diverse routes of many demands by removing a large portion of demand matrices. However, amongst the reduced demand matrices, some demands still need to use different routes to achieve optimal results. Example 1 in Section 8.3.2 shows that two demands between nodes (1,2) should use two different routes. In theory, I may need to select distinct routes per demand to achieve an optimal result.

However, in practice, optimizing routes per demand can be extremely expensive on a larger network, where the nodes are more distant and route choices are many. If I use only one route per node pair, the search space is greatly reduced. In fact, I find that the optimizer can find a good enough solution much faster by using a fixed route per node pair.

Versions of `SaOptimize` and `GaOptimize` were developed to leverage this heuristic. In `SaOptimize`, the only change occurs in Lines 11-13. Instead of randomly changing the route of a demand, we randomly change the route of a node pair. During initialization, all demands of the same pair are allocated with the same least-cost route. The `GaOptimize` algorithm retains its structure except for the changing concept of genes and chromosomes. Demands can no longer choose their routes as freely as before. Node pairs are the unit for route mapping. I define a new tuple, called pair-route, which maps a route for each node pair. Each pair-route tuple represents a gene. The route set for the pair represents all possible choices for this gene. A route allocation of all pairs represents a chromosome. Since the route for each demand is determined by its node pair, a chromosome also maps to a solution. A set of

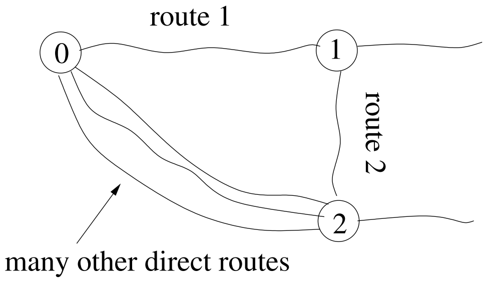


Figure 8.13: Motivation for two-stage optimization. Demand (0,2) should share the route 1 and 2 to achieve an optimal result. However, there are too many other routes between (0, 2), making it hard for the optimizer to choose route 1 and 2.

chromosomes forms a generation. Extending `SaOptimize` and `GaOptimize` to support fixed routes per pair is straightforward, so I omit the algorithms in this paper.

Two-Stage Optimization

The huge space of candidate routes makes convergence to an optimal solution difficult. Figure 8.13 illustrates the situation. The figure shows part of the network. Ideally, I want demand (0, 2) to share route 1 for demand (0, 1) and route 2 for demand (1, 2) to achieve the optimal result. However, there are hundreds of candidate routes (on the virtual-link graph) for demand (0, 2). The chance that demand (0, 2) merges with route 1 plus route 2 is small. To solve the problem, I use a two-stage optimization technique, explained in Algorithm 8.2. The algorithm uses `GaOptimize`, but any optimizer I discussed previously can be used. In fact, I use those supporting fixed routes in my results.

The two-stage optimization finds a reasonable solution in the first round. Then, the graph is reduced to include only the physical nodes and links that appear in the solution. The second round tries to find a better solution based on the reduced graph. During the second stage, the optimizer may find better routes, as is the case in Figure 8.13, since many possible routes are ruled out. Also, I find that the second stage can run much faster by reducing the number of generations to two-thirds of the number used in the first stage and reducing the population by half.

Algorithm 8.2: Two-stage genetic algorithm optimization.

```
Input: Reduced demand matrix set $\mathcal{S}$
Input: Generations $k$
Input: Population per generation $s$
Input: Optimization steps for wavelength assignment $k'$
Output: Optimal allocation and wavelength assignment $A^w$
Run the optimizer $A_1^w \leftarrow \texttt{GaOptimize}(d, k, s, k')$;
Reduce the virtual-link graph to only include the nodes $\mathcal{N}' \leftarrow \{n | n \in A_1\}$ and
    physical links $\mathcal{E}' \leftarrow \{e | e \in A_1\}$;
Run the optimizer $A_2^w \leftarrow \texttt{GaOptimize}(d, \lfloor \frac{2}{3}k \rfloor, \lfloor \frac{1}{2}s \rfloor, k')$ on the new graph;
Pick the minimum $A^w \leftarrow \min(A_1^w, A_2^w)$;
```

8.4.5 Greedy Algorithm

Procedure `Greedy` is to greedily allocate the resources of all demands that maximally reuse the resources that have already been allocated. Note that the demands of different demand matrices can share resources but the demands of the same demand matrix cannot. I allocate each demand matrix one-by-one in an arbitrary order. In the beginning, there is no resource allocated, so the routes allocated for the demands in the first demand matrix are their least-cost paths, using default costs for wavelength channels and REGENs. After the first demand matrix, I have a network on which some resources have been already provisioned.

Starting with the second demand matrix, the resources that have been provisioned previously are marked with zero cost. Then, the route for each newly allocated demand attempts to maximally reuse the wavelength and REGENs that are used by other provisioned demand matrices. The demands of the same demand matrix must pick non-overlaping paths to avoid resource sharing. This is done by assuming maximally available resources on the residual network at the beginning of each demand matrix. After a route is picked for a demand, the resources used by the route are removed from the residual network and so forth. The resources used by the demand matrix are then combined with the currently provisioned graph. I use "union" operation so the resources that were there previously are not doubly counted. During routing, if there are multiple available wavelength channels of the same cost, the first-fit wavelength assignment algorithm is used. The network is provisioned after all demand matrices have been allocated. The algorithm is linear in network sizes for a given number of demands. Although I am aware that the possible number of demands can still grow exponentially in terms of the

Procedure Greedy–Greedy provisioning algorithm.

```
Initialize empty resource graph $G \leftarrow \emptyset$;
foreach reduced demand matrix $D \in \mathcal{S}$ do
    Initialize residual resource graph with full capacity
    $RG \leftarrow \cup_{\forall e \in E, w \in W}(e, w) \cup_{\forall r \in R_n, n \in N} r$;
    foreach $(e, w) \in RG$ do
        if $(e, w) \in G$ then
            $cost_{e,w} \leftarrow 0$;
        else
            $cost_{e,w} \leftarrow l_e \times \mathsf{CC}$;
    foreach $(n, r) \in RG$ do
        if $(n, r) \in G$ then
            $cost_{n,r} \leftarrow 0$;
        else
            $cost_{n,r} \leftarrow \mathsf{RC}$;
    foreach demand $d \in D$ do
        Find the least-cost path available on the residual graph $p \in RG$;
        $RG \leftarrow RG \setminus p$;
        demand route $A(d) \leftarrow p$;
    Get used resource graph $UG \leftarrow RG^c$;
    Combine resource graphs $G \leftarrow G \cup UG$;
```

number of ports, the complexity of the algorithm has been greatly reduced by eliminating the combinatorial optimization part.

8.5 Volatility Analysis

This section studies the robustness of the network through analyzing the volatility of the optimized networks. First, I define the concept of individual and group nonvolatility and show that the optimal dimensioning and routing allocation cannot guarantee a nonvolatile solution. The analytical results are used to find additional conditions that guarantee individual and group nonvolatility.

As customers move from one demand matrix to another, they may add new connections, remove old connections, and retain some connections. If the designated route for an existing connection in the old demand matrix differs from the designated route for the same connection in the new demand matrix, the connection must be rerouted. If I tear down the old route and set up a new route, customers can experience a few minutes of interruption (hundreds to thousands gigabytes of data loss on 10Gbps links), which is unacceptable for mission-critical applications. To minimize the loss, I can take advantage of B&R, which incurs less than 50 milliseconds interruption (SONET restoration threshold time). However, B&R requires certain resources to be available.

Definition 9 (B&R condition)**.** *Each node must have a spare OT. All customer access equipment has an SP installed at the transmission side. To reroute a connection using B&R, the wavelength and REGEN resources for the new route must be available before the old route is taken down.*

In this paper, I assume that the network is provisioned with spare OTs and SPs to perform B&R (the cost of these OTs and SPs is fixed and not counted). I assume that each customer connection can tolerate the loss due to one B&R operation during the interval between two demand matrix changes. Now, I must define connection non-interruption. If I must reroute a connection and the B&R condition for that route is not met, the connection must be interrupted by removing the old route and creating a new route.

Definition 10 (Non-interruption)**.** *A connection is not interrupted if the allocated route is not changed or is changed to another route using one B&R operation for any demand matrix change.*

The demands of one node pair may be assigned different routes in an optimal solution. In such cases, rerouting is inevitable. In Figure 8.14, B&R suffices for all changes, so the red demand must reroute.

There are two possible modes in which a customer may use their network. For the first mode, the *individual demand changing mode*, demand change is defined by an arrival of a new demand or a departure of an existing one. The arrivals and departures are random events with some probability. Assume that no two events happen concurrently. Individual demand changing mode applies to the case for which each OT-OT pair is managed independently, and an established OT-OT connection should not be interrupted unless the connection departs. Each event corresponds to a change of demand matrix, and any demand matrix can be reached through a series of events. Figure 8.14 shows an example of the mode. The red connection is different from the black connection in (b) even if it ends up using the same route in (d) and (e) as the black connection.

The second mode, the *maximum demand changing mode*, assumes that the customer uses only maximal demand matrices at any time. A *maximal*

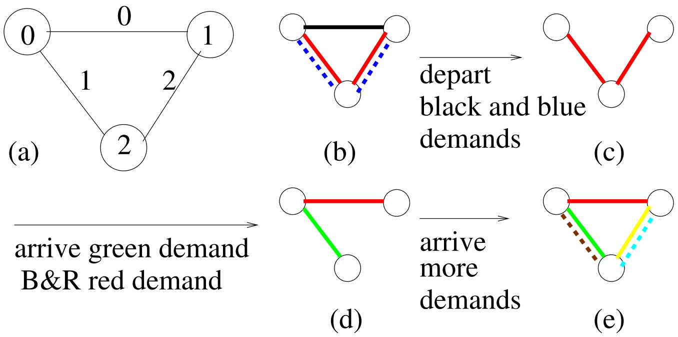


Figure 8.14: Example for demand rerouting using eager B&R. (a) The virtual-link graph for the three-node network with port constraints $O_0 = 3$, $O_1 = 3$, and $O_2 = 4$. Assume that all 2-hop routes need a REGEN. An optimal solution allocates wavelength 1 on link 0, and wavelengths 1 and 2 on both link 1 and link 2. All 2-hop routes use spare OTs as REGENs. (b) The customer starts with a maximal demand matrix: three demands from node pair (0, 1). The routes of demands are marked with different colors. Red and blue routes are regenerated at node 2 using two spare OTs. A solid line indicates wavelength 1. A dashed line indicates wavelength 2. (c) The black and blue demands depart. (d) The green demand for (0, 2) arrives. The red demand must be rerouted to the correct route on link 0. (e) Three more demands arrive to form another maximal demand matrix.

demand matrix is one for which no more demands can be added without removing an existing demand. A non-maximal demand matrix means resource idling. Maximum demand changing mode applies to customers who manage their entire private network; using partial network resources is unlikely for these customers. Therefore, the change of demands must be from one maximal demand matrix to another. In this mode, the demands of the same node pair in two maximal demand matrices are considered the "same" demands. The same demands are retained and should not be interrupted during reconfiguration. The customer in Figure 8.14 would change directly from (b) to (e); steps (c) and (d) are not allowed. The black route will be retained in (e) in place of the red one. I define individual and group nonvolatility corresponding to these application modes.

Definition 11 (Individual nonvolatility)**.** *Using individual demand changing mode, a new arrival demand can always be accepted without interrupting any retained connection.*

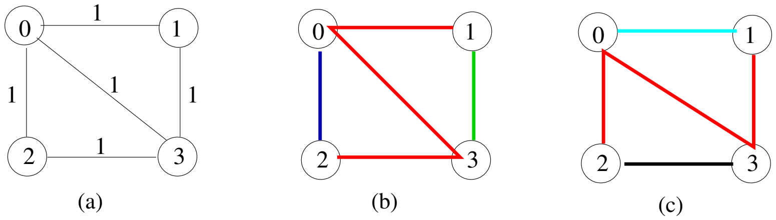


Figure 8.15: The proof of Theorem 6. (a) The physical graph for a four-node network with port constraint $O_0 = O_3 = 1, O_1 = O_2 = 2$. Assume that all routes are reachable, so no REGEN is needed in the solution. An optimal solution requires one wavelength channel (channel 1) on all links, as shown in the figure. (b) The route allocation for demands: blue (0, 2), red (1, 2), and green (1, 3). (c) The route allocation for demands: Cyan (0, 1), red (1, 2), black (2, 3). The red routes of (1, 2) are self-locked with wavelength resources and cannot reroute using B&R, so the allocation is volatile.

Definition 12 (Group nonvolatility)**.** *Using maximum demand changing mode, the connections shared between the initial and final demands are not interrupted when switching from one maximal demand matrix to another.*

Even if a maximal demand matrix change can be decomposed into a sequence of individual demand arrival events, individual nonvolatility does not necessarily infer group nonvolatility: individual nonvolatility cannot guarantee one B&R per retained connection for the entire maximal demand matrix change. The inference is not necessarily true in the other direction as well. Studying the relationship between individual and group nonvolatilities is the goal for future work.

Theorem 6. *An optimal route provisioning solution may satisfy neither individual nor group nonvolatility.*

Figure 8.15 shows the proof of Theorem 6 using an example that fails to meet nonvolatility conditions. The red route is self-locked with wavelength resources, so neither individual nor group nonvolatility holds if the demand (1, 2) needs to be retained. In general, an optimal solution is not free from resource contention that prevents a B&R. Constructing an optimal solution that satisfies nonvolatility is challenging for many cases, but I yet have no proof or counterproof for the existence of nonvolatile optimal solutions for all problems. The problem thus becomes a future area of study.

8.6 Sufficient Nonvolatile Conditions

Rerouting across demands is the major cause of volatility in optimal solutions. A sufficient nonvolatile condition is to limit the possibility of demand rerouting. If the demands of each node pair must be chosen from a fixed pool of resource disjoint routes, where the selection priority of routes is strictly defined for the order of demands appearing in the demand matrix, I can guarantee individual and group nonvolatility. More specifically, if there are at most three demands between nodes 0 and 1, I create three candidate routes in the pool $\{p_0, p_1, p_2\}$. If a demand matrix contains only one demand (0,1), route p_0 must be selected. If a demand matrix contains two demands (0,1), routes p_0, p_1 must be selected. If a demand matrix contains three demands (0,1), all routes are selected. With the strict route selection order, group nonvolatility is easily satisfied, since the possibility of rerouting has been completely removed.

For individual nonvolatility, each arrival demand selects its route in order, and eager B&R is applied to reroute retained connections on each arrival. If p_0 is available when a (0,1) demand arrives, the demand selects p_0. If p_1 is available but p_0 is not, the demand selects p_1. On each new arrival, every retained demand must check if the current route is appropriate. A demand needs to be rerouted to the appropriate route using B&R. The way that we reroute demands on each arrival is called *eager B&R*. Eager B&R can avoid the resource deadlock that can arise on a heavily loaded network. In Figure 8.14, assume that the route selection order for node pair (0, 1) is $\{p_0, p_1, p_2\}$, where the virtual-link-wavelength set (v, w) of routes are $V_{p_0} = \{(0,1)\}, V_{p_1} = \{(1,1),(2,1)\}$, and $V_{p_2} = \{(1,2),(2,2)\}$. The route selection order for node pair (0, 2) is $\{p_3, p_4, p_5\}$, where the virtual-link-wavelength set (v, w) of routes is $V_{p_3} = \{(1,1)\}, V_{p_4} = \{(1,2)\}$, and $V_{p_5} = \{(0,1),(2,1)\}$. The red demand uses route p_1 in (b), indicating that it is a later arrival than the black one. The red demand does not need to reroute at (c) because there is no new arrival. The red demand must B&R to the high-priority route p_0 on the arrival of the green demand at (d) since the allocated route for the green demand, p_3, conflicts with p_1 on wavelength 1.

Formally speaking, the following constraint (Equation 8.10) is added into the ILP described in Section 8.1 to guarantee individual and group nonvolatility (Theorem 7):

$$\begin{aligned}
&\forall d_1 = \{(i,j), s_1, t\}, \forall d_2 = \{(i,j), s_2 \neq s_1, t\}, \\
&\forall p \in \mathcal{P}_{(i,j)}, X_{p,d_1} = X_{p,d_2} \qquad (8.10)
\end{aligned}$$

Each demand d is described by a node pair $N_d = (i,j)$, demand matrix s, and the order t of the node pair's demands in the demand matrix. For example, in Figure 8.14(b), the black demand has $t = 0$, the red demand has $t = 1$, and the blue demand has $t = 2$. Note that the demand order t for a node pair in a demand matrix indicates route priority in the individual demand changing mode. The same OT-OT demand connection can change its order when the demand matrix changes. Again in Figure 8.14, the red demand has order $t = 1$ in (b), but the order is switched to $t = 0$ in (d). However, for maximum demand changing mode, the order always stays the same.

Theorem 7. *The optimal provisioning solution computed with Equation 8.10 guarantees individual and group nonvolatility.*

Proof. A demand d of node pair N_d with order t will reroute only to a route p' that is previously used by another demand d' of the same node pair with order $t' < t$. After d' departs, the appropriate route for d becomes p'. The legal route for d should further change to p'' if another demand of pair N_d depart with orders $t'' < t'$. Without loss of generality, I assume the legal route for d is p' when the next new arrival comes. Because the appropriate route was occupied by a recently departed demand and no new demand has arrived to use more network resources, ever since, demand d can reroute to p' using one B&R operation. Any other inappropriate route can be fixed by one B&R for the same reason. Since the network is provisioned with the condition specified by Equation 8.10, and all retained connections are on appropriate routes, the new arrival can be accepted without contending with any retained connection. Therefore, the route ordering scheme and eager B&R guarantee individual nonvolatility. Group nonvolatility also holds since there is no need of B&R on any transition. □

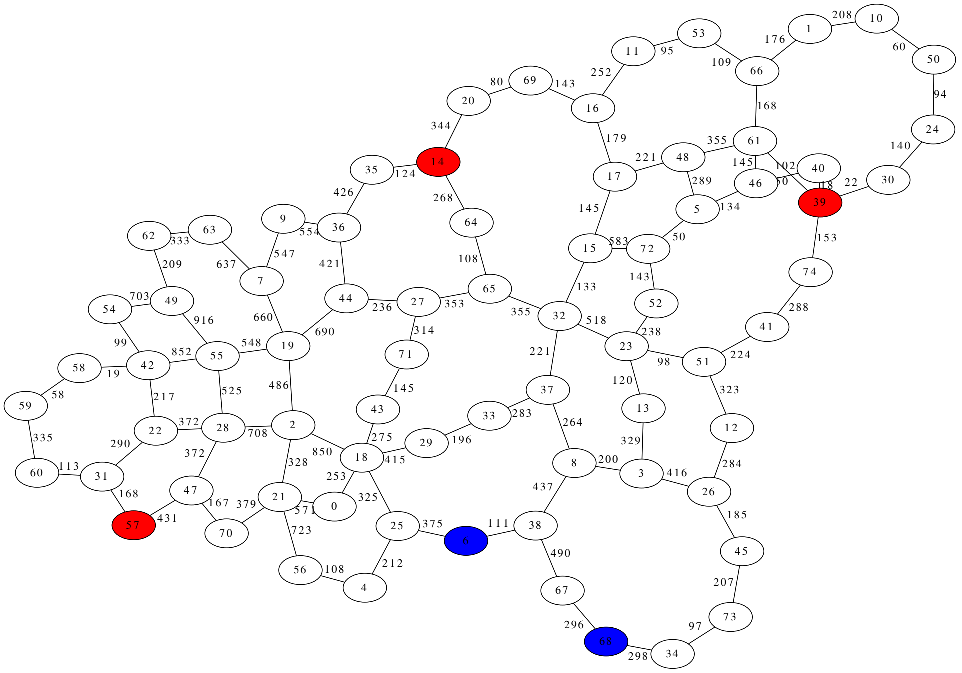


Figure 8.16: U.S. CORONET topology. Nodes are labeled with numbers. Links are labeled with distance in miles. Geographic information has been removed for clarity. Node 14, 39, 57 are marked in red. Other nodes referred to by Figure 8.20 are marked in blue.

8.7 Numerical Results

CORONET was created by the Telcordia-AT&T team to mimic a typical large international core network [34]. I use the U.S. contiguous part of CORONET, which consists of 75 nodes and 99 links. Each node maps to a U.S. city. Figure 8.16 shows the topology of the network; node numbers and link distance are marked. I assume a 40 Gbps ROADM system for the network. Each DWDM link has 80 wavelengths. The photonic transparent reachability is 932 miles. Each REGEN costs 150, and the common cost is 0.07 per mile per wavelength.

Three cities are chosen as base cities for a customer. They are node 14 (Chicago), node 39 (New York City), and node 57 (San Diego). Assume that the default port constraint is $O_{14} = 2, O_{39} = 3, O_{57} = 2$. Only the reduced demand matrices are used by the optimizer.

I tested five optimization algorithms: simulated annealing (SA), simulated annealing with fixed routes (SAF), genetic algorithm (GA), genetic algorithm with fixed routes (GAF), and two-stage genetic algorithm with fixed routes

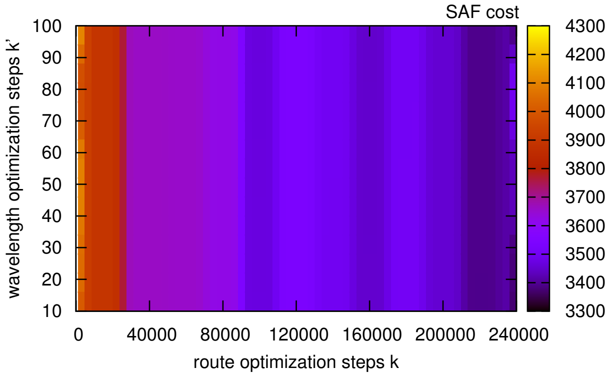


Figure 8.17: SAF cost color map for $O_{14} = 2, O_{39} = 3, O_{57} = 2$. $C_{lb} = 2292.84$. Each data point is an independent optimization process.

(GAF2). They are all compared with the lower bound (low). The simulation is run on an AMD Opteron 64bit machine with a 2.2GHz CPU, 1MB cache memory, and 7GB main memory.

First, I study the performance of the simulated annealing optimizers by varying the number of optimization steps. Figure 8.17 shows the color map of the resulting cost of SAF for the customer traffic demands $O_{14} = 2, O_{39} = 3$, and $O_{57} = 2$. The trends for SA are similar. The lower bound computed is 3216.32. Each data point is a complete new optimization with renewed seeds for random number generators. Therefore, larger steps do not always yield better solutions than smaller steps. The solution improves with an increasing number of steps in the route allocation. However, the benefit of increasing steps for wavelength assignment is not present. Then, I choose $k' = 50$ for the steps for wavelength assignments for the rest of the paper.

I next study the performance of the genetic algorithm by varying the number of generations k and population per generation s. Figure 8.18 shows the color map of the optimal cost of GAF. The lower bound stays the same for the same customer demand. Better solutions are found with an increasing number of generations and population. Using $k = 80$ and $s = 2000$, the values are good enough.

Heuristic optimization methods get closer to an optimal value as the running time increases, but the optimal value may never be reached in a finite length of time. Instead of comparing the runtime, I compare the optimized results given the same amount of computing work. I compare the performance

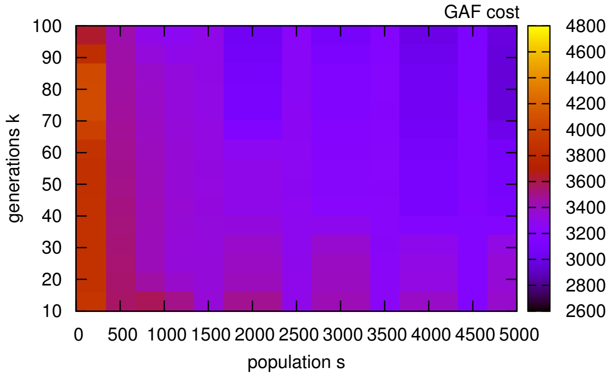
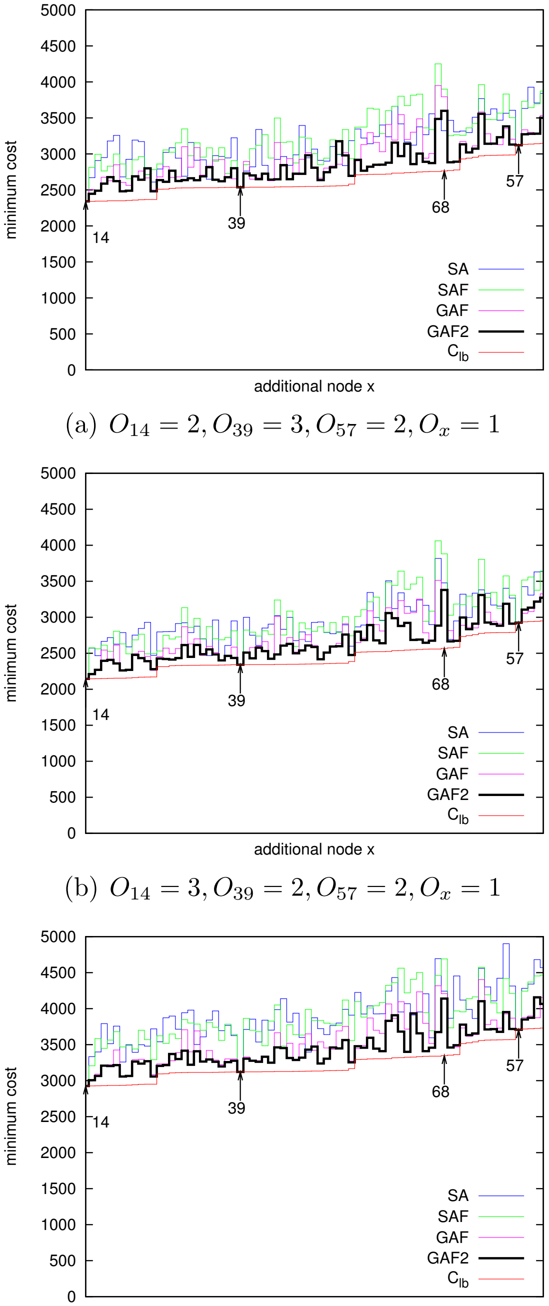


Figure 8.18: GAF cost color map for $O_{14} = 2$, $O_{39} = 3$, $O_{57} = 2$. $C_{lb} = 2292.84$. Each data point is an independent optimization process.

of all algorithms using a four-node demand constraint $O_{14} = 2$, $O_{39} = 3$, $O_{57} = 2$, $O_x = 1, x \in N$. The fourth node is an arbitrary node in the network. If $x \in \{14, 39, 57\}$, there are still 3 nodes, except one has an additional port. I choose the parameters $k = 240,000, k' = 50$ for SA/SAF and $s = 2000$, $k = 80$, $k' = 50$ for GA/GAF so every test case walks the same number of samples in the space and runs approximately for an hour (a smaller number is picked for GAF2 for each stage, so it takes about the same amount of time). The runtime for the lower bounds is on the scale of a few seconds.

The comparison is shown in Figure 8.19(a). The result of GA is not shown because GAF and GAF2 outperform GA for all cases. GAF2 can find a better solution than other algorithms for most cases. GAF is slightly better than SA and SAF. SA and SAF are about the same. Figure 8.19(b) shows results for $O_{14} = 2$, $O_{39} = 3$, $O_{57} = 2$, $O_x = 1, x \in N$, and Figure 8.19(c) shows results for $O_{14} = 2$, $O_{39} = 2$, $O_{57} = 3$, $O_x = 1, x \in N$. All the nodes are sorted by increasing order of the lower bound. The results for $x \in \{14, 39, 57\}$ and the node of the highest $(GAF2 - C_{lb})$ value are annotated. If $x \in \{14, 39, 57\}$, the total number of ports is even for the three nodes and the optimal result equals the lower bound. On average, GAF2 requires 8% additional cost of the lower bound. GAF is 11%, SA is 20%, SAF is 21%, and GA is 27% over the lower bound. The worst-case nodes are those geographically far from the original three nodes. Because of a larger route space due to the distance, the optimizer is less likely to land on a solution as good as others. The amount



(a) $O_{14} = 2, O_{39} = 3, O_{57} = 2, O_x = 1$

(b) $O_{14} = 3, O_{39} = 2, O_{57} = 2, O_x = 1$

(c) $O_{14} = 2, O_{39} = 2, O_{57} = 3, O_x = 1$

Figure 8.19: The performance of the heuristic optimization algorithms. The additional port is added to any one of the nodes in the network. Each data value is a random optimization run. SA/SAF $k = 240,000, k' = 50$, GAF/GAF2 $k = 2000, s = 80, k' = 50$. The nodes are sorted in increasing order of their lower bounds.

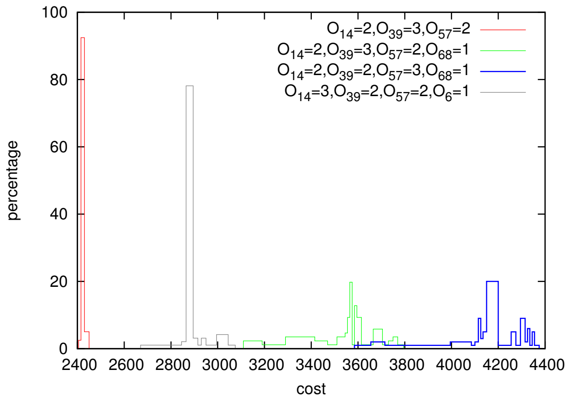


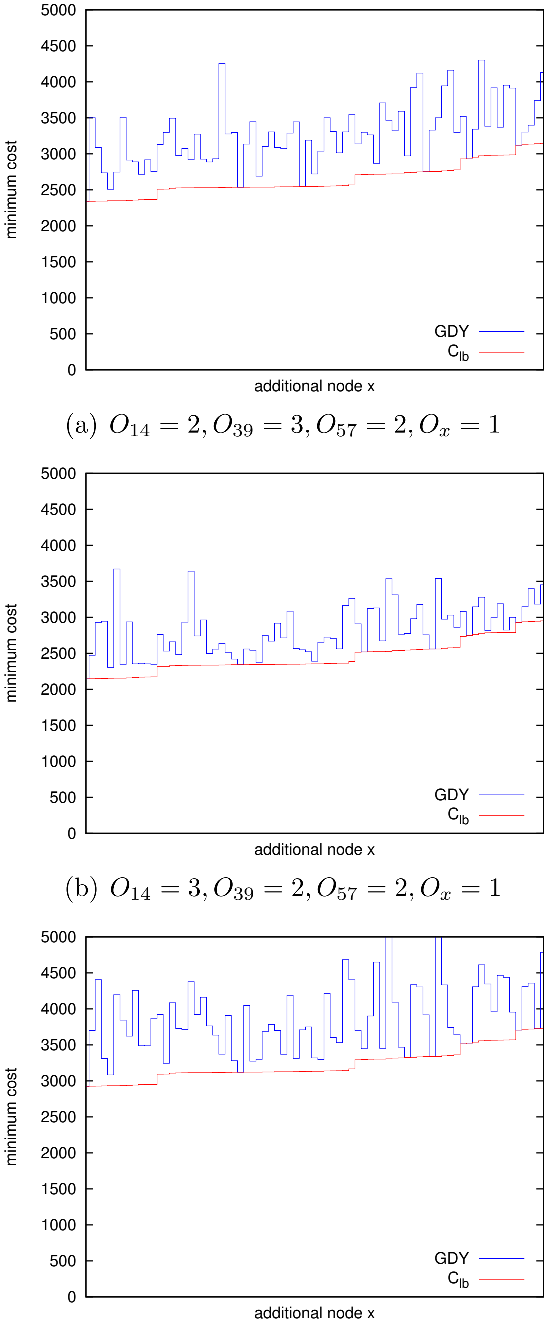
Figure 8.20: The cost distribution of 100 repeated random GAF2 runs for some cases. The standard deviations are within 4% of the mean values. The additional nodes are marked in Figure 8.16.

of resources that cannot be shared by demand matrices is also likely to be higher. Node 68, which is the farthest of the original three nodes, requires the highest additional cost for all experiments.

To understand the randomness of the optimizer, I compute the standard deviation for some cases with 100 repeated random GAF2 runs. The results are shown in Figure 8.20. The deviations are all within 4% of the mean value. Therefore, the cases where other algorithms are better than GAF2 in Figures 8.19(a)–8.19(c) can be attributed to deviations.

I then compare the performance of the greedy algorithm (GDY) with the optimized result (OPT) in Figures 8.21(a)–8.21(c). Unlike the hour run of the optimization algorithms, the GDY result can be obtained within a second. On average, GDY incurs an overhead of 19% of the lower bound.

Figure 8.22 shows that the average cost difference between the optimal nonvolatile solution and the volatile one is within the error range of the heuristic optimizer. I run for representative customer demands: there are four to five geographically remote nodes with two to five OTs at each node. Some large cities—such as New York, Chicago, Boston, Seattle, San Diego, and Raleigh—are chosen with a higher probability. Both simulations use GAF2 and run for the same number of generations and steps (the longest one runs for approximately four days). No significant overhead for applying the nonvolatile condition is incurred on the network.



(a) $O_{14} = 2, O_{39} = 3, O_{57} = 2, O_x = 1$

(b) $O_{14} = 3, O_{39} = 2, O_{57} = 2, O_x = 1$

(c) $O_{14} = 2, O_{39} = 2, O_{57} = 3, O_x = 1$

Figure 8.21: Performance of the greedy algorithm. The additional port is added to any one of the nodes in the network. The nodes are sorted in increasing order of their lower bound results.

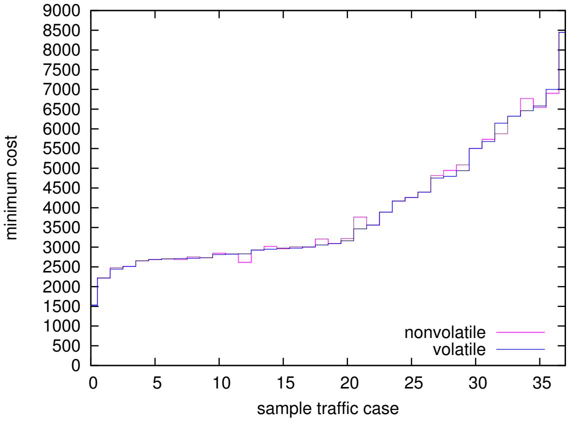


Figure 8.22: Comparison of nonvolatile and volatile heuristic optimization results.

8.8 Conclusion

In this chapter, I formulate a resource optimization problem for dynamic wavelength services. I propose a demand matrix reduction algorithm to reduce the input demand matrices for the optimizer by an order of magnitude. However, I find that the number of reduced demand matrices still grows as a factorial as the number of ports increases. I solve the optimization problem by decomposing it into a wavelength assignment problem and a route allocation problem. For the former, I show that the optimization problem is NP-hard and then introduce a simulated annealing version. For the latter, I show a simulated annealing version and a genetic algorithm version. The use of fixed routes and two-stage optimization techniques lets me trim the solution space. Since finding an optimal solution is impractical on a realistic backbone network due to the problem size, I propose a lower bound and a greedy algorithm that can be computed quickly. On a network of realistic size, the optimizer solver using the genetic algorithm with space trimming techniques (GAF2) obtains solutions within 8% of the lower bound. My greedy approach can achieve 19% overhead at about 0.02% of the runtime of GAF2. The greedy algorithm reduces the provisioning cost for network carriers that want to expand their on-demand dynamic wavelength services to a large number of customers on a national-scale backbone network.

Further, I define the volatility problem that can occur when dynamic wavelength services are used for mission-critical networks in which connection interruptions are not allowed. Network volatility is analyzed under the assumption that the underlying photonic layer can perform photonic bridge-and-roll to reroute a connection. At most, one B&R operation is allowed for each connection during a demand change. I conclude that the optimal resource optimization solution does not guarantee a nonvolatile network. A sufficient condition for nonvolatile solutions is proposed, and I show that the nonvolatile solution incurs no additional cost relative to the volatile one.

CHAPTER 9

RELATED WORK

9.1 Network Dimensioning

Resource dimensioning generally focuses on three main topics: physical wavelength channel provisioning on opaque networks (opto-electronic-opto (OEO) or fully wavelength convertible networks) [37, 56, 36], placement of electronic/optical devices (wavelength converters, regenerators, or transponders) on transparent/translucent networks [57, 58, 59, 60, 61, 62, 40, 63, 64], and virtual topology design on a transparent WDM network [65, 66, 67, 68, 69, 70]. The work in [71] discussed a combined planning for both fiber resources and converters. However, many of the results were still applied to static or dynamic traffic models with fixed-path routing or fixed alternate routing. Dimensioning fiber/equipment resources on topologies that support dynamic traffic with complicated link-state-based routing algorithms still has not been sufficiently studied.

Regarding dimensioning an opaque network, Nayak and Sivarajan [37] proposed an asymptotic routing and dimensioning approach based on absorption probability analysis of a linear traffic growth model. The authors in [56] then proposed a time-dependent blocking probability approach to further reduce network capacity. Their work studied transient network behavior, starting from zero initial traffic, with the assumption that the network would be periodically redimensioned and could be reconstructed in a timely manner to respond to traffic change. Further, the authors in [36] studied the dimensioning problem for dynamic traffic using a system perspective; they proposed a heuristic basic dimensioning algorithm for boot-strapping network design and a few incremental dimensioning algorithms for future network growth. The study compared the new approaches to previous studies that showed that using a more flexible path-selection algorithm for dynamic demands on a well-dimensioned network can greatly improve blocking performance.

Previous work on dimensioning optical transparent/translucent networks with wavelength continuity constraints and physical impairments [72] mostly focused on placement of wavelength converters and regenerators. The authors of [57] investigated the usefulness of wavelength converters on varying topologies, traffic loads, and available wavelengths per fibers. Yates [58] provided a comprehensive modeling and performance study on wavelength converter placement in dynamic networks. This analysis showed that the performance improvement strongly depended on the wavelength assignment schemes and wavelength channel allocation. For static traffic demands, the light path provisioning problem is usually solved by formulating a mixed-integer Linear Programming (ILP) solution. The paper [60] proposed an ILP solution to provision wavelength fiber resources with or without wavelength converters. The authors in [71] formulated another ILP solution to allocate wavelength channels, optical cross-connects and wavelength converters on a physical topology dimensioned for a given traffic demand. They decomposed the large problem into dimensioning and routing subproblems and wavelength assignment subproblems to alleviate the computational hurdle. The authors in [64] studied a network coverage problem by minimizing switching nodes and transceivers on a topology.

Many have studied the regenerator placement problem on a given WDM network. The authors in [59] proposed several heuristic regenerator placement algorithms for dynamic traffic. The authors in [61] proposed an ILP formulation to compute optical regenerator placement for static demands. In [62], they studied the reduction needed in the cost of electro-optical equipment to increase the reachability of the network for a given traffic demand. Further, a more thorough discussion about regenerator based translucent optical network models is discussed in [40]; heuristic regenerator assignment algorithms were proposed with traffic demand predictions, where routing was done by heuristic online RWA algorithms.

Some studies address wavelength allocation for transparent networks but with limited problem scale. Baroni and Bayvel [73] studied the number of required wavelengths for a static and uniform demand on a given topology. The paper modelled the physical topological connectivity and derived the lowest bound of the wavelength number for each network. A heuristic RWA was also proposed to compute the minimal required wavelength number. A dimensioning solution with a small number of wavelengths per link was discussed in [74]. The authors in [63] compared the lowest theoretical bound of the capacity requirement for optical packet switching, optical burst switching, and optical flow switching networks regardless of routing algorithms.

Another important study related to WDM network dimensioning is the design and reconfiguration of virtual topologies (also called "logical topology/layer" in some studies). A virtual topology consists of all-optical lightpaths, reserved from the physical WDM topology, that provide a layer of abstraction for IP customers to route and traffic engineer without consideration of wavelength continuity and physical reachability [65, 66, 67]. Many logical layers can share one or more WDM lower-level networks. Virtual topology design is particularly useful for static optical network models, where reserving/releasing a bypass lightpath is time-consuming and expensive. As the lower-level WDM network becomes more dynamic, virtual topologies can be more frequently reconfigured according to traffic demand changes [68, 69, 75]. The authors in [68] proposed an online, logical topology reconfiguration based on live traffic measured on a daily basis. For a different dynamic traffic model, a subwavelength grooming resource optimization problem for multiple possible static traffic matrices was introduced in [76]. In contrast, dimensioning the physical network as I do is increasingly important: upper layer traffic has a more direct impact on the WDM network, with increasing support for on-demand, high-bandwidth services.

9.2 Online Routing

Research on routing and wavelength assignment (RWA) is closely related to network dimensioning, especially for static traffic models. ILP is commonly applied to optimize resource utilization when selecting paths for a set of traffic demands. However, the ILP can grow intractable, and optimizing for large networks over a long period of time is infeasible in practice. Offline routing algorithms are also infeasible when future information is simply unavailable. For dynamic traffic models, efficient heuristic algorithms are used to route each arrival connection request. Online RWA is usually treated independently of network dimensioning (assuming given network resources).

The routing and wavelength assignment problem for a given traffic model on translucent networks has long been known to be NP-hard[77], but many heuristic algorithms were proposed [78, 79, 80, 18]. A previous study also suggested that routing algorithms have a higher impact on performance than wavelength assignment[81].

Routing algorithms have been studied extensively for many years in the network community. Shortest path first (SPF) routing has been widely adopted for online routing because it is efficient to adopt and responsive to changes in network traffic demands. Many variants of SPF with additional resource constraints have been developed to improve network load balancing and reduce congestion (often referred to as Least Loaded Routing (LLD)). Data traffic engineering techniques, such as trunk reservation (TR), may also be applied to further improve load balancing and accommodate multiclass data flows.

In the past, LLD in circuit-switched networks was studied extensively in the context of ATM broadband networks [82, 83], WDM networks [38, 18, 29, 84, 85, 86, 22] and general broadband networks [87, 88, 19, 20]. The set of candidate paths can be preselected or dynamically discovered. Preselected paths allow a simplified signaling process for path setup. Dynamically selected paths offer flexibility and tolerance to traffic fluctuations. Many LLD algorithms select a path with maximum residual capacity on bottleneck links. The widest-shortest path (WSP) [89, 87, 90] and shortest-widest path (SWP)[91] algorithms are examples. In [87] and [38], variants of WSP were proposed with hop-extension limits. SWP chooses the shortest one among those paths with maximum residual capacity. It has been shown that SWP is inferior to WSP in dynamic wavelength routed networks [90]. Some others, like the fixed-paths least-congestion (FPLC) algorithm for wavelength continuity networks [18], use the sum of free capacity along the path. Minimal interference routing (MIR) [30], a flow-based algorithm, attempts to select the path that is least likely to interfere with future requests from other node pairs. An improved reduced-flow-based routing algorithm (RFR) was proposed in [22]. The work also compared many routing algorithms on dimensioned networks. Instead of looking at the absolute residual bandwidth, the authors in [84] proposed a convex function that uses a fraction of wavelength utilization to compute the link congestion cost. In [29], the authors proposed a Least Resistance Weight (LRW) routing that uses the available capacity normalized by the maximum link capacity in the network. The authors in [85] proposed another congestion estimation using the inverse of available link capacity. The admission control technique is also used for network congestion management. Krishnan *et al.* proposed admission control mechanisms, called state-dependent routing (SDR), for dynamic circuit-routed networks[31]. Later, Gawlick *et al.* [33] proposed another admission control mechanism, called throughput-competitive online routing (COL). The performance of these algorithms was compared in [32]. However, optimizing the parameters for these algorithms has not been addressed. Combined admission control and routing was studied in [92]. For WDM networks, wavelength continuity, physical impairments or survivability requirements may constrain the choice of routes [72, 93, 94]. Specific wavelength assignment techniques can reduce the impact of crosstalk to increase the reachability of some routes[95]. These techniques are not mature in near-future devices, so I do not include them in our model.

Combinations of online and offline approaches have been suggested in [7] and [96]. The adaptive design based routing (Adaptive DBR) scheme [7] takes advantage of expected network traffic demands to select and provision candidate routes for each pair optimally at the offline stage. The online stage of DBR then assigns either a precomputed route or dynamically computed available route with trunk reservation. This method utilizes the currently monitored load to compensate for the variations from the static traffic demand estimates used in offline optimization. Profile-based routing (PBR) [96] uses the expected traffic demands to perform offline computation of the threshold of trunk reservation for each class of connections, and SPF is then used at the online stage.

Analytical models [18, 19, 20] have been proposed to approximate the blocking probability with state-dependent routing for preselected adaptive routing. Some theoretical analysis for RWA can also be found in [18]. In [21], the author used reduced load approximation to analyze the blocking performance of dynamic traffic grooming. However, due to the complexity of the space and dependencies for state-dependent routing algorithms, analytical models are still insufficient to catch the performance of complex and flexible dynamic routing algorithms.

There have been many studies on traffic rerouting at the optical layer. On a given resource provisioned network, connection disruption on traffic changes cannot always be avoided with rerouting [97]. Bridge-and-roll techniques were applied to WDM connections to expedite rerouting of existing lightpaths [98]. However, rerouting or preemption [88, 99] should be carefully engineered to avoid excessive service interruption on existing high-speed connections so it is generally avoided in the WDM layer. Minimizing the number of disrupted connections is a known NP-hard problem and has been studied in [100]. Prioritizing connections for non-interruption was then discussed in [101]. On the design of electronic layer topologies, an incremental topology-configuration approach was proposed by [102] for guaranteed non-interruption during reconfiguration. In contrast with previous studies, my work approaches the problem from the resource provisioning perspective. Rerouting is managed by the carrier, and the number of rerouting operations is limited due to the small amount of interruption allowed in a realistic WDM network. Dynamic wavelength service traffic model is port-constrained, which is also different from other studies.

New traffic models, such as scheduled traffic or hybrid traffic, also call for further research on routing algorithms [103, 104, 105]. Many studies have been devoted to wavelength continuity routing and subwavelength grooming on virtual topologies. Different definitions of Quality-of-Service (QoS) and Service Level Agreements (SLAs) for each class of traffic further complicate the problem [89, 106, 107, 108, 109, 110, 111].

9.3 Multiple Network Domains

Performance studies with different levels of information sharing have been the main research focus. Sharing complete network state information across multiple network domains can be impractical for two reasons: (1) the scale of the aggregated domain exceeds the signaling capacity to flood all link states, and (2) information (e.g., complete network topology) is often considered proprietary/private. With regard to scalability, [112] proposed an efficient information aggregation and updating system for large-scale networks. [113] is a theoretical study on the minimum information needed between network domains to route within a tolerable error rate. Topology aggregation techniques have been used to reduce the amount of information shared between domains, often utilizing virtual topologies to provide more compact and abstract information. [50] compared the performance of global shortest path, concatenated SPF (similar to source routing in other studies), and single-node aggregation/hierarchical routing (equivalent to stitched path routing). They also proposed three different border selection criteria for source routing: random selection, closest border, and least loaded path to border. [114] further proposed a quantitative study of hierarchical routing performance by using simple node (same as single node), full-mesh and symmetric star aggregation schemes.

The study of routing algorithms generally falls into three categories depending upon the level of information sharing. (1) End-to-end global shortest path routing. This is complete information sharing, though not necessarily needed. (2) Source-initiated-concatenated shortest path routing, which searches for a path domain by domain starting from the source (minimum information sharing). Crankback signaling may be used to reattempt a failed search in a segment. (3) Hierarchical/hybrid routing, with various topology aggregation techniques and hence varying amounts of information shared, depending on topology exposure. Generally, performance improves as more information is shared, but the scalability problem increases, as does the inter-domain problem.

9.4 Heuristic Optimization

Combinatorial optimization problems can quickly become unmanageable when the number of elements to be chosen is large. Simulated annealing [115, 116] is a practical heuristic optimization tool to find a good solution given a certain amount of time. The algorithm simulates the random walk of atoms in a metal that is heated and slowly cooled, in order to explore the solution space. Genetic algorithms [117] apply the idea of natural selection to find the optimal combination of elements. These algorithms have been found useful for discrete and unstructured problems. However, for each specific problem, solution elements must be carefully structured and algorithm parameters tuned to achieve the best performance. A genetic algorithm heuristic for routing and dimensioning fully wavelength convertible networks of Poisson dynamic traffic was proposed by [118].

CHAPTER 10

CONCLUSION

Managing large-scale dynamic networks is a hard problem. Network state in analytical models can quickly grow, and it becomes impossible to find a equilibrium solution for routing algorithms that heavily utilize residual graph information. To find a practical solution, I explored the directions that can provide the necessary information to compute at runtime without expensive analysis. In the beginning, I looked at the problem space of Poisson traffic on optical opaque networks. Optical signals must be electronically regenerated at every node. The Poisson traffic model aims for the most flexible on-demand connection services, with blocking probability, for a large number of customers. Traditional WDM networks are opaque. Today, optical reach is extended beyond a single hop, but the opaque network model is widely applied to generalized label switching networks that carry IP traffic directly on a WDM backbone.

Initially, I computed flow reductions for each route option. Then, I realized that the result really depends on how the network is dimensioned. However, there are few previous research discussions about network dimensioning. The whole thing motivated me to approach the routing problem and the dimensioning problem altogether. The effect of the combined approach is phenomenal. As the dimensioning algorithm places resources at the shortest routes of pairs, the amount of useful information for routing algorithms becomes highly localized. In reduced flow routing, I need only to compute the flow interference on the region of affected shortest routes. My reduced flow routing thus improved previous work remarkably in both blocking probability and computation time. Further, I found that the performance of online routing is penalized at high loads because new arrivals are more likely to select excessively long routes. In order to prevent long routes, I used admission control algorithms. I developed an opportunity cost model, trying to predicting the right threshold. Using admission control, I can achieve the optimal solution at both high and low loads. I found that the effect of admission control dominates that of route selection. Even with random shortest routing, near-optimal solutions can be achieved with admission control. For managing dynamic networks, we should really focus on admission control instead of routing.

As optical reach extends beyond hundreds of miles, WDM routes require fewer regeneration sites and become more transparent. A translucent network model is applied to newer type of WDM networks, which allow wavelength routes to bypass switching nodes photonically if the transmission distance is within the maximal photonic reachability. I attempted to apply the dimensioning algorithms for Poisson traffic to translucent networks. But the cost model is more complicated compared to the opaque model, and I have yet to go beyond identifying the bottleneck network resources under a few anticipated operating modes.

I developed an interest in another dynamic traffic model, called "dynamic wavelength services" while I was collaborating with AT&T Labs Research. The main missing part was how to determine the cost of each route that was used to find the cost distribution of demand matrices. I formulated the problem and tried to optimize the cost. I chose a genetic algorithm since it generally provides good performance for discrete combinatorial problems. Further, I considered how the bridge-and-roll rerouting technology developed by the GRIPhoN project could be applied to dynamic wavelength services. I found that finding an optimal dimensioning in which bridge-and-roll can be always applied to migrate inappropriate routes is extremely challenging.

There are many areas worth exploring beyond this thesis. Dynamic traffic imposes another challenge for survivable optical network provisioning. I touched on the dimensioning problem for restoration but remain far from sufficiently understanding the implications of applying admission control to shared backup path protections or *Stream* protections [119]. There are avenues of thought on selecting the right pair of primary and backup routes. Also, protection for dynamic wavelength services is an interesting problem. Backup resources are shared not only from different demand matrices but also between the same demand matrix. These constraints introduce new variables into the optimization problems and volatility analysis.

To summarize, this thesis contributes to the network management problem in five areas:

1. This thesis presents a joint resource dimensioning and routing approach for Poisson traffic on opaque networks. As a result, the implementation of each dimensioning and routing algorithm is much simplified, for which I check only the residual capacity along the route. My algorithms provide better performance than previous work. I propose basic dimensioning algorithms for initial network resource allocation and incremental dimensioning algorithms for network scaling. The performance is evaluated with a traffic evolution model, where the traffic load ranges from completely expected to completely unknown. My solution is more robust than previous solutions to inaccurate traffic load estimation and long-term pattern evolutions.

2. I develop an opportunity cost model to optimize the connection request acceptance rate on dimensioned networks. Using the model, I design threshold-based routing and admission control algorithms that are efficient to implement and for which the optimal threshold is analytically determined and remains roughly constant for a wide range of traffic loads, network topologies, and scales.

3. The dimensioning technique is applied to several novel problems. I study the resource bottleneck problem for dynamic translucent networks. I identify the bottleneck resources for four different wavelength operating modes. Then, I evaluate the resource requirements for dynamic restorable networks. I develop a redimensioning algorithm to maintain comparable performance after a single-link failure and find that the actual restoration overhead is close to that for shared path protection. Finally, I study the fairness problem for multiple optical domains. I dimension joint-domain networks and studied routing and resource usage between two domains. I show that a certain joint dimensioning and routing is important to achieve operating fairness.

4. This thesis is the first to formulate and solve a combinatorial dimensioning and routing optimization problem for a practical dynamic traffic model (called “dynamic wavelength services”) on optical translucent networks. I reduce the problem size by exploring several properties of the traffic model and approach the problem using a combination of meta-heuristic optimization techniques. Further, I propose a quick greedy algorithm for scalable problems.

5. For mission-critical applications, I evaluate a photonic bridge-and-roll rerouting technique on optimized dynamic wavelength service networks. I

find that some transient arrival sequences will cause bridge-and-roll to fail. I solve the problem by developing a provable sufficient bridge-and-roll safe condition.

REFERENCES

[1] S. McCreary and K. Claffy, "Trends in wide area IP traffic patterns - A view from Ames Internet Exchange," in *The 13th ITC Specialist Seminar on Internet Traffic Measurement and Modelling*, Monterey, CA, 2000.

[2] R. Doverspike and P. Magill, *Optical Fiber Telecommunications.* London, UK: Elsevier Inc., 2008, vol. B: Systems and Networks, ch. Commercial optical networks, overlay networks, and services, pp. 511–560.

[3] L. Berger, "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description," RFC 3471, Jan. 2003.

[4] J. Strand and A. Chiu, "Realizing the advantages of optical reconfigurability and restoration with integrated optical cross-connects," *Journal of Lightwave Technology*, vol. 21, no. 11, pp. 2871–2882, Nov. 2003.

[5] M. D. Feuer, D. C. Kilper, and S. L. Woodward, *Optical Fiber Telecommunications.* London, UK: Elsevier Inc., 2008, vol. B: Systems and Networks, ch. ROADMs and their system applications, pp. 293–344.

[6] X. J. Zhang, M. Birk, A. Chiu, R. Doverspike, M. D. Feuer, P. Magill, E. Mavrogiorgis, J. Pastor, S. L. Woodward, and J. Yates, "Bridge-and-roll demonstration in GRIPhoN (globally reconfigurable intelligent photonic network)," in *Optical Fiber Communication Conference and Exposition (NFOEC'10)*, San Diego, California, USA, Mar. 2010, pp. 1–3.

[7] A. Elwalid, D. Mitra, I. Saniee, and I. Widjaja, "Routing and protection in GMPLS networks: From shortest paths to optimized designs," *IEEE Journal on Lightwave Technology*, vol. 21, no. 11, pp. 2828–2838, 2003.

[8] Y. Xin, L. Battestilli, and G. Karmous-Edwards, "Generic network services to support emerging grid applications," in *IEEE International Conference on Broadband Communication, Networks, and Systems (BROADNETS)*, Raleigh, North Carolina, USA, Sep. 2007.

[9] S. Spadaro, "Traffic engineering in IP over optical transport networks for metropolitan and wide area environments," Ph.D. dissertation, Universitat Politècnica de Catalunya, Dec. 2004.

[10] N. Ghani and S. Dixit, "Channel provisioning for higher-layer protocols in WDM networks," in *All-Optical Networking 1999: Architecture, Control, and Management Issues*, J. M. Senior, C. Qiao, and S. Dixit, Eds. SPIE, 1999, vol. 3843, no. 1, pp. 22–32.

[11] K. Casier, S. Verbrugge, D. Colle, I. Lievens, A. Groebbens, M. Pickavet, and P. Demeester, "Dimensioning studies for transparent optical backbone networks," in *Transparent Optical Networks, 2005, Proceedings of 2005 7th International Conference*, vol. 1, July 2005, pp. 252–255.

[12] F. P. Kelly, "Blocking probabilities in large circuit-switched networks," *Advances in Applied Probability*, vol. 18, pp. 473–505, 1986.

[13] B. R. Hurley, C. J. R. Seidl, and W. F. Sewell, "A survey of dynamic routing methods for circuit-switched traffic," *IEEE Communications Magazine*, vol. 25, no. 9, pp. 13–21, 1987.

[14] A. Girard, *Routing and Dimensioning in Circuit-Switched Networks.* Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1990.

[15] A. Girard and B. Liau, "Dimensioning of adaptively routed networks," *IEEE/ACM Trans. Netw.*, vol. 1, no. 4, pp. 460–468, 1993.

[16] F. P. Kelly, "Loss networks," *The Annals of Applied Probability*, vol. 1, no. 3, pp. 319–378, 1991. [Online]. Available: http://www.jstor.org/stable/2959742

[17] A. Dwivedi and R. Wagner, "Traffic model for USA long-distance optical network," in *Optical Fiber Communication Conference, 2000*, vol. 1, 2000, pp. 156–158.

[18] L. Li and A. K. Somani, "Dynamic wavelength routing techniques and their performance analyses," in *Optical WDM Networks*, ser. The Kluwer International Series in Engineering and Computer Science, K. M. Sivalingam and S. Subramaniam, Eds. Springer, 2002, vol. 554, pp. 247–272.

[19] Y. Lee and J. M. Tien, "Static and dynamic approaches to modeling end-to-end routing in circuit-switched networks," *IEEE/ACM Transactions on Networking*, vol. 10, no. 5, pp. 693–705, 2002.

[20] M. Liu and J. Baras, “Fixed point approximation for multirate multihop loss networks with state-dependent routing,” *IEEE/ACM Transactions on Networking*, vol. 12, no. 2, pp. 361–374, Apr. 2004.

[21] C. Xin, “Blocking analysis of dynamic traffic grooming in mesh WDM optical networks,” *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, pp. 721–733, June 2007.

[22] X. J. Zhang, S. Kim, and S. S. Lumetta, “Reduced flow routing: Leveraging residual capacity to reduce blocking in GMPLS networks,” in *Fourth International Conference on Broadband Communications, Networks, and Systems (BroadNets'07)*, Raleigh, North Carolina, USA, Sep. 2007.

[23] R. Ramaswami and K. N. Sivarajan, *Optical Networks: A Practical Perspective.* San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1998.

[24] A. Saleh and J. Simmons, “Evolution toward the next-generation core optical network,” *Journal of Lightwave Technology*, vol. 24, no. 9, pp. 3303–3321, Sept. 2006.

[25] T. E. Stern and K. Bala, *Multiwavelength Optical Networks: A Layered Approach.* Prentice Hall, 2002.

[26] G. Li, A. Chiu, R. Doverspike, D. Xu, and D. Wang, “Efficient routing in heterogeneous core DWDM networks,” in *NFOEC2010*, San Diego, CA, Mar. 2010.

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

[28] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, and G. Swallow, “RSVP-TE: Extensions to RSVP for LSP Tunnels,” RFC 3209, Dec. 2001.

[29] B. Wen and K. Sivalingam, “Routing, wavelength and time-slot assignment in time division multiplexed wavelength-routed optical WDM networks,” in *Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2002)*, vol. 3, 2002, pp. 1442–1450.

[30] M. S. Kodialam and T. V. Lakshman, “Minimum interference routing with applications to MPLS traffic engineering,” in *INFOCOM*, 2000, pp. 884–893.

[31] K. R. Krishnan and T. J. Ott, “State-dependent routing for telephone traffic: Theory and results,” in *25th IEEE Conference on Decision and Control*, vol. 25, Dec. 1986, pp. 2124–2128.

[32] L. Zhang, M. Andrews, W. Aiello, S. Bhatt, and K. Krishnan, “A performance comparison of competitive on-line routing and state-dependent routing,” in *Global Telecommunications Conference. GLOBECOM ’97, IEEE*, vol. 3, Nov 1997, pp. 1813–1819.

[33] R. Gawlick, A. Kamath, S. Plotkin, and K. G. Ramakrishnan, “Routing and admission control in general topology networks,” Stanford University, Stanford, CA, USA, Tech. Rep. TR-95-1548, 1995.

[34] A. L. Chiu, G. Choudhury, G. Clapp, R. Doverspike, J. W. Gannett, J. G. Klincewicz, G. Li, R. A. Skoog, J. Strand, A. von Lehmen, and D. Xu, “Network Design and Architectures for Highly Dynamic Next-Generation IP-Over-Optical Long Distance Networks,” *Journal of Lightwave Technology*, vol. 27, pp. 1878–1890, June 2009.

[35] Z. Jing, H. Sun, and L. Li, “Performance evaluation of packet discard schemes in atm switches in heterogeneous traffic environment,” in *ICATM ’99. 2nd International Conference on ATM*, 1999, pp. 482–491.

[36] X. J. Zhang, S. Kim, and S. S. Lumetta, “Resource dimensioning in WDM networks under state-based routing schemes,” in *Fourth International Conference on Broadband Communications, Networks, and Systems (BroadNets’07)*, Raleigh, North Carolina, USA, Sep. 2007.

[37] T. K. Nayak and K. N. Sivarajan, “Routing and dimensioning of optical networks under traffic growth models: an asymptotic analysis,” *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 8, pp. 1241–1253, Oct. 2003.

[38] J. Späth, “Dynamic routing and resource allocation in WDM transport networks,” *Computer Networks*, vol. 32, pp. 519–538, May 1999.

[39] X. J. Zhang, S. Kim, and S. S. Lumetta, “Efficient and robust congestion estimation for dynamic WDM networks,” University of Illinois at Urbana–Champaign, Tech. Rep. UILU-ENG-09-2212, 2009.

[40] X. Yang and B. Ramamurthy, “Sparse regeneration in translucent wavelength-routed optical networks: Architecture, network design and wavelength routing,” *Photonic Network Communications*, vol. 10, no. 1, pp. 39–53, July 2005.

[41] A. Autenrieth and A. Kirstadter, “Engineering end-to-end ip resilience using resilience-differentiated qos,” *Communications Magazine, IEEE*, vol. 40, no. 1, pp. 50–57, Jan. 2002.

[42] D. Choi, D. Taylor, and R. Seibel, “Priority-based optical network protection and restoration with application to dod networks,” in *Military*

Communications Conference, 2003. MILCOM 2003. IEEE, vol. 1, Oct. 2003, pp. 298–303.

[43] N. Ghani, Q. Liu, A. Gumaste, D. Benhaddou, N. Rao, and T. Lehman, “Multidomain optical networks: issues and challenges - control plane design in multidomain/multilayer optical networks,” *Communications Magazine, IEEE*, vol. 46, no. 6, pp. 78–87, June 2008.

[44] D. King, Y. Lee, H. Xu, and A. Farrel, “Path computation architectures overview of multi-domain optical networks based on ITU-T ASON and IETF PCE,” in *Network Operations and Management Symposium Workshops. NOMS Workshops 2008. IEEE*, April 2008, pp. 219–226.

[45] X. Yang and B. Ramamurthy, “Inter-domain dynamic routing in multilayer optical transport networks,” in *Global Telecommunications Conference, 2003. GLOBECOM ’03. IEEE*, vol. 5, Dec. 2003, pp. 2623–2627.

[46] T. Saad, H. Mouftah, and A. Nouroozifar, “Constraint-based routing across multi-domain optical wdm networks,” in *Electrical and Computer Engineering, 2004. Canadian Conference on*, vol. 4, May 2004, pp. 2065–2068.

[47] S. Figuerola, J. Garcia, A. Sanchez, C. de Waal, and A. Willner, “The network service plane: An approach for inter-domain network reservations,” in *Transparent Optical Networks, 2008. ICTON 2008. 10th Anniversary International Conference on*, vol. 1, June 2008, pp. 13–15.

[48] J. Vasseur, R. Zhang, N. Bitar, and J. Roux, “A Backward Recursive PCE-based Computation (BRPC) procedure to compute shortest interdomain Traffic Engineering Switched paths,” 2007. [Online]. Available: http://tools.ietf.org/html/rfc5441.

[49] T. Tachibana and H. Harai, “Lightpath establishment without wavelength conversion based on aggressive rank accounting in multi-domain wdm networks,” *Lightwave Technology, Journal of*, vol. 26, no. 12, pp. 1577–1585, June 2008.

[50] K. Zhu, H. Zhu, and B. Mukherjee, “Traffic engineering in multigranularity heterogeneous optical WDM mesh networks through dynamic traffic grooming,” *Network, IEEE*, vol. 17, no. 2, pp. 8–15, Mar/Apr 2003.

[51] D. P. Goyal, N. G. Duffield, P. Goyal, A. Greenberg, P. Mishra, K. K. Ramakrishnan, and J. E. Van Der Merwe, “A flexible model for resource management in virtual private networks,” in *Proc. ACM SIGCOMM*, 1999, pp. 95–108.

[52] K. Oikonomou and R. Sinha, "Network design and cost analysis of optical vpns," in *OFC 2006 - Optical Fiber Communication Conference and Exhibit*, Mar. 2006.

[53] R. Doverspike, "Practical aspects of bandwidth-on-demand in optical networks," in *Panel on Emerging Networks, Service Provider Summit, Optical Fiber Communication Conference (OFC)*, Anaheim, CA, Mar. 2007.

[54] S. Kim, X. J. Zhang, and S. S. Lumetta, "Minimizing protection cost for high-speed recovery of mission critical traffic in optical mesh networks," in *Military Communications Conference (MILCOM'07)*, Orlando, Florida, USA, Oct. 2007.

[55] A. Kumar, R. Rastogi, A. Silberschatz, and B. Yener, "Algorithms for provisioning virtual private networks in the hose model," *IEEE/ACM Trans. Netw.*, vol. 10, pp. 565–578, August 2002. [Online]. Available: http://dx.doi.org/10.1109/TNET.2002.802141

[56] W. Tian, "A dynamic modeling and dimensioning approach for all-optical networks," in *BROADNETS*, 2006.

[57] S. Subramaniam, M. Azizoglu, and A. Somani, "All-optical networks with sparse wavelength conversion," *Networking, IEEE/ACM Transactions on*, vol. 4, no. 4, pp. 544–557, Aug. 1996.

[58] J. Yates, "Performance analysis of dynamically-reconfigurable wavelength division multiplexed networks," Ph.D. dissertation, The University of Melbourne, Nov. 1997.

[59] S.-W. Kim and S.-W. Seo, "Regenerator placement algorithms for connection establishment in all-optical networks," in *IEE Proceedings Communications*, vol. 148, no. 1, Feb 2001, pp. 25–30.

[60] M. Tornatore, G. Maier, and A. Pattavina, "Wdm network optimization by ilp based on source formulation," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, June 2002, pp. 1813–1821.

[61] E. Yetginer and E. Karasan, "Regenerator placement and traffic engineering with restoration in GMPLS networks," *Photonic Network Communications*, vol. 6, pp. 139–149, Sep. 2003.

[62] A. Morea, H. Nakajima, L. Chacon, E. Le Rouzic, B. Decocq, and J.-P. Sebille, "Impact of the reach of WDM systems and traffic volume on the network resources and cost of translucent optical transport networks," in *Sixth International Conference on Transparent Optical Networks*, vol. 1, July 2004, pp. 65–68.

[63] G. Weichenberg, V. Chan, and M. Medard, "On the capacity of optical networks: A framework for comparing different transport architectures," *Selected Areas in Communications, IEEE Journal on*, vol. 25, no. 6, pp. 84–101, August 2007.

[64] B. Chen and F. Tobagi, "Network topology design to optimize link and switching costs," in *Communications, 2007. ICC '07. IEEE International Conference on*, June 2007, pp. 2450–2456.

[65] G. N. Rouskas and M. H. Ammar, "Dynamic reconfiguration in multihop WDM networks," *Journal of High Speed Networks*, vol. 4, pp. 221–238, 1995.

[66] A. Farago, S. Blaabjerg, L. Ast, G. Gordos, and T. Henk, "A new degree of freedom in atm network dimensioning: optimizing the logical configuration," *Selected Areas in Communications, IEEE Journal on*, vol. 13, no. 7, pp. 1199–1206, Sep. 1995.

[67] R. Ramaswami and K. Sivarajan, "Design of logical topologies for wavelength-routed optical networks," *IEEE Journal on Selected Areas in Communications*, vol. 14, no. 5, p. 840, Jun 1996.

[68] A. Gencata and B. Mukherjee, "Virtual-topology adaptation for WDM mesh networks under dynamic traffic," in *IEEE Infocom*, 2002.

[69] H. Levy, T. Mendelson, and G. Goren, "Dynamic allocation of resources to virtual path agents," *Networking, IEEE/ACM Transactions on*, vol. 12, no. 4, pp. 746–758, Aug. 2004.

[70] C. Xin and B. Wang, "Logical topology design for dynamic traffic grooming in mesh WDM optical networks," in *IEEE International Conference on Communications. ICC 2005*, vol. 3, May 2005, pp. 1792–1796.

[71] A. Zymolka, A. Koster, and R. Wessäly, "Transparent optical network design with sparse wavelength conversion," in *Proceedings of ONDM 2003. The 7th IFIP Working Conference on Optical Network Design & Modelling*, Budapest, Hungary, February 2003, pp. 61–80.

[72] J. Strand, A. L. Chiu, and R. Tkach, "Issues for routing in the optical layer," *IEEE Communications Magazine*, vol. 39, no. 2, pp. 81–87, Feb. 2001.

[73] S. Baroni and P. Bayvel, "Wavelength requirements in arbitrarily connected wavelength-routed optical networks," *Lightwave Technology, Journal of*, vol. 15, no. 2, pp. 242–251, Feb 1997.

[74] A. Zapata, A. Leiva, R. Vallejos, and M. Aravena, “Static vs. dynamic WDM optical networks under single-cable failure conditions,” in *13th Conference on Optical Network Design and Modeling (ONDM’09)*, Braunschweig, Germany, Feb. 2009.

[75] S. Sinha and C. S. R. Murthy, “Information theoretic approach to traffic adaptive WDM networks,” *IEEE/ACM Trans. Netw.*, vol. 13, no. 4, pp. 881–894, 2005.

[76] N. Srinivas and C. S. R. Murthy, “Design and dimensioning of a WDM mesh network to groom dynamically varying traffic,” *Photonic Network Communication*, vol. 7, no. 2, pp. 179–191, Mar. 2004. [Online]. Available: http://www.springerlink.com/content/u3211j6185863120

[77] H. Zang, J. P. Jue, and B. Mukherjee, “A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks,” *Optical Networks Magazine*, vol. 1, pp. 47–60, 2000.

[78] A. Birman and A. Kershenbaum, “Routing and wavelength assignment methods in single-hop all-optical networks with blocking,” *INFOCOM ’95. Fourteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Bringing Information to People. Proceedings. IEEE*, vol. 2, pp. 431–438, Apr 1995.

[79] X. Zhang and C. Qiao, “Wavelength assignment for dynamic traffic in multi-fiber WDM networks,” in *7th International Conference on Computer Communications and Networks*, Oct 1998, pp. 479–485.

[80] C. Assi, A. Shami, M. Ali, R. Kurtz, and D. Guo, “Optical networking and real-time provisioning: an integrated vision for the next-generation internet,” *Network, IEEE*, vol. 15, no. 4, pp. 36–45, Jul/Aug 2001.

[81] H. Zang, J. Jue, L. Sahasrabuddhe, R. Ramamurthy, and B. Mukherjee, “Dynamic lightpath establishment in wavelength routed WDM networks,” *Communications Magazine, IEEE*, vol. 39, no. 9, pp. 100–108, Sep 2001.

[82] H.-W. Chu and D. H. K. Tsang, “Modified least loaded routing in virtual path based atm networks,” *Telecommunication Systems*, vol. 7, no. 1, pp. 45–57, June 1997.

[83] D. Medhi and I. Sukiman, “Admission control and dynamic routing schemes for wide-area broadband networks: Their interaction and network performance,” in *Proc. of IFIP/IEEE Conf. on Broadband Communications*, Montreal, Canada, Apr. 1996, pp. 99–110.

[84] B. Zhou and H. T. Mouftah, “Adaptive least loaded routing for multi-fiber WDM networks using approximate congestion information,” in *IEEE International Conference on Communications, 2002. ICC’02*, vol. 5, 2002, pp. 2725–2749.

[85] R. Mewanou and S. Pierre, “Dynamic routing algorithms in all-optical networks,” in *Electrical and Computer Engineering, 2003. IEEE CCECE 2003. Canadian Conference on*, vol. 2, May 2003, pp. 773–776.

[86] X. Tian, X. Qi, Q. Ma, and X. Zhang, “Study on a distributed wavelength routing algorithm in WDM optical transport networks,” *Photonic Network Communications*, vol. 11, no. 3, pp. 271–276, May 2006.

[87] A. Shaikh, J. Rexford, and K. G. Shin, “Load-sensitive routing of long-lived IP flows,” in *SIGCOMM ’99: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, 1999, pp. 215–226.

[88] B. Szviatovszky, A. Szentesi, and A. Juttner, “Minimizing re-routing in MPLS networks with preemption-aware constraint-based routing,” *Computer Communications*, vol. 25, pp. 1076–1083, 2002.

[89] R. Guerin, A. Orda, and D. Williams, “QoS routing mechanisms and ospf extensions,” in *2nd Global Internet Miniconference*, 1997.

[90] S. Xu and S. Kaoru, “Comparison between shortest path first routing and dynamic least load routing in WDM networks,” *Technical Report of IEICE.*, vol. 101, pp. 11–16, 2001.

[91] Z. Wang and J. Crowcroft, “Quality-of-service routing for supporting multimedia applications,” *IEEE Journal of Selected Areas in Communications*, vol. 14, no. 7, pp. 1228–1234, 1996.

[92] X. J. Zhang, S. Kim, and S. S. Lumetta, “Opportunity cost analysis of online routing algorithms for dynamic wavelength routed mesh networks,” in *Optical Fiber Communication Conference and Exposition (OFC’10)*, San Diego, California, USA, Mar. 2010.

[93] J. He, M. Brandt-Pearce, and S. Subramaniam, “QoS-aware wavelength assignment with BER and latency constraints for all-optical networks,” *IEEE/OSA Journal of Lightwave Technology*, no. 5, pp. 462–474, 2009.

[94] Y. Pointurier, M. Brandt-Pearce, S. Subramaniam, and B. Xu, “Cross-layer adaptive routing and wavelength assignment in all-optical networks,” *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 6, pp. 32–44, Aug. 2008.

[95] J. He, “RWA algorithm design and performance analysis for all-optical networks subject to physical impairments,” Ph.D. dissertation, University of Virginia, May 2008.

[96] S. Suri, M. Waldvogel, and P. R. Warkhede, “Profile-based routing: A new framework for MPLS traffic engineering,” in *Quality of Future Internet Services*, ser. Lecture Notes in Computer Science, F. Boavida, Ed. Berlin: Springer Verlag, Sep. 2001, no. 2156, pp. 138–157, an earlier version is available as Washington University Computer Science Technical Report WUCS-00-21, July 2000.

[97] N. Jose and A. Somani, “Connection rerouting/network reconfiguration,” in *Design of Reliable Communication Networks, 2003. (DRCN 2003). Proceedings. Fourth International Workshop on*, Oct. 2003, pp. 23–30.

[98] S. Fortune and L. Zhang, “The effect of bridge-and-roll on minimizing wavelength conversion for dynamic traffic,” in *Proceedings of ECOC*, 2008.

[99] P. Iovanna, R. Sabella, and M. Settembre, “A traffic engineering system for multilayer networks based on the GMPLS paradigm,” *Network, IEEE*, vol. 17, no. 2, pp. 28–37, Mar/Apr 2003.

[100] F. Solano, “Analyzing two different objectives of the WDM lightpath reconfiguration problem,” in *IEEE Global Communications Conference (GLOBECOM’09)*, Honolulu, Hawaii, USA, Dec. 2009.

[101] D. Coudert, F. Huc, D. Mazauric, N. Nisse, and J.-S. Sereni, “Reconfiguration of the routing in WDM networks with two classes of services,” in *Optical Network Design and Modeling, 2009. ONDM 2009. International Conference on*, Feb. 2009, pp. 1–6.

[102] M. Saad and Z.-Q. Luo, “Reconfiguration with no service disruption in multifiber WDM networks,” *Lightwave Technology, Journal of*, vol. 23, no. 10, pp. 3092–3104, Oct. 2005.

[103] J. Kuri, N. Puech, and M. Gagnaire, “Routing and grooming of scheduled lightpath demands in a multi-granularity switching network: a mathematical model,” in *Optical Network Design and Modeling, 2005. Conference on*, July 2005, pp. 73–83.

[104] B. Wang and A. Deshmukh, “An all hops optimal algorithm for dynamic routing of sliding scheduled traffic demands,” *Communications Letters, IEEE*, vol. 9, no. 10, pp. 936–938, Oct. 2005.

[105] T. D. Wallace and A. Shami, "Connection management algorithm for advance lightpath reservation in wdm networks," in *Broadband Communications, Networks and Systems, 2007. BROADNETS 2007. Fourth International Conference on*, Sep. 2007, pp. 837–844.

[106] A. Jukan and H. van As, "Quality-of-service routing in optical networks," in *Integrated Optics and Optical Fibre Communications, 11th International Conference on, and 23rd European Conference on Optical Communications (Conf. Publ. No.: 448)*, vol. 3, Sep 1997, pp. 160–163.

[107] G. Apostolopoulos, R. Guérin, S. Kamat, and S. K. Tripathi, "Quality of service based routing: a performance perspective," *SIGCOMM Comput. Commun. Rev.*, vol. 28, no. 4, pp. 17–28, 1998.

[108] S. Ramesh, G. N. Rouskas, and H. G. Perros, "Computing blocking probabilities in multiclass wavelength routing networks," *ACM Trans. Model. Comput. Simul.*, vol. 10, no. 2, pp. 87–103, 2000.

[109] J. Marzo, E. Calle, C. Scoglio, and T. Anjah, "Qos online routing and MPLS multilevel protection: a survey," *Communications Magazine, IEEE*, vol. 41, no. 10, pp. 126–132, Oct 2003.

[110] P. Iovanna, R. Sabella, and P. D'Aprile, "GMPLS traffic engineering system to dynamically manage several classes of services: performance analysis," in *Optical Network Design and Modeling, IEEE Conference on*, Feb 2005, pp. 483–491.

[111] X. Yang, T. Lehman, C. Tracy, J. Sobieski, S. Gong, P. Torab, and B. Jabbari, "Policy-based resource management and service provisioning in GMPLS networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications*, Apr. 2006, pp. 1–12.

[112] J. Szigeti, I. Ballok, and T. Cinkler, "Efficiency of information update strategies for automatically switched multi-domain optical networks," in *Transparent Optical Networks, 2005, Proceedings of 2005 7th International Conference*, vol. 1, July 2005, pp. 445–454.

[113] G. Liu, C. Ji, and V. Chan, "On the scalability of network management information for inter-domain light-path assessment," *Networking, IEEE/ACM Transactions on*, vol. 13, no. 1, pp. 160–172, Feb. 2005.

[114] G. Maier, C. Busca, and A. Pattavina, "Multi-domain routing techniques with topology aggregation in ason networks," in *Optical Network Design and Modeling, 2008. ONDM 2008. International Conference on*, March 2008, pp. 1–6.

[115] S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[116] R. Carr, “Simulated annealing,” MathWorld—A Wolfram Web Resource, created by Eric W. Weisstein, 2011. [Online]. Available: http://mathworld.wolfram.com/SimulatedAnnealing.html.

[117] H.-P. Schwefel, *Numerical Optimization of Computer Models.* New York, NY, USA: John Wiley & Sons, Inc., 1981.

[118] I. de Miguel, R. Vallejos, A. Beghelli, and R. J. Durán, “Genetic algorithm for joint routing and dimensioning of dynamic wdm networks,” *J. Opt. Commun. Netw.*, vol. 1, no. 7, pp. 608–621, 2009. [Online]. Available: http://jocn.osa.org/abstract.cfm?URI=JOCN-1-7-608

[119] S. Kim and S. S. Lumetta, “Capacity-efficient protection with fast recovery in optically transparent mesh networks,” in *BROADNETS*, 2004, pp. 290–299.

AUTHOR'S BIOGRAPHY

Xiaolan (Joy) Zhang was born in Shanghai, China. She is currently a Ph.D. candidate of Electrical and Computer Engineering at the University of Illinois at Urbana-Champaign. She received a B.S. in information engineering from Shanghai Jiao Tong University in 2003 and a M.S. in computer engineering from Purdue University at Indianapolis in 2005. She works closely with colleagues from AT&T Labs Research for developing traffic management technologies on reconfigurable optical add/drop multiplexer (ROADM) networks. She also worked for IBM T.J. Watson Research Center every summer during the year 2007-2010 for developing a distributed stream data processing system. She is a recipient of an IBM Ph.D. Fellowship and a Rambus Fellowship in Computer Engineering. Her research area includes optical networks, performance profiling, modeling, and security. She lives in Champaign, Illinois, with her husband and their son.