Robustness and Vulnerability Design for Autonomic Management

by

Alireza Bigdeli

A thesis submitted in conformity with the requirements
for the degree of Doctor of Philosophy
Graduate Department of Electrical and Computer Engineering
Department
University of Toronto

Abstract

Robustness and Vulnerability Design

for

Autonomic Management

Alireza Bigdeli

Doctor of Philosophy

Graduate Department of Electrical and Computer Engineering Department

University of Toronto

2012

This thesis presents network design and operations algorithms suitable for use in an autonomic management system for communication networks with emphasis on network robustness. We model a communication network as a weighted graph and we use graph-theoretical metrics such as network criticality and algebraic connectivity to quantify robustness. The management system under consideration is composed of slow and fast control loops, where slow loops manage slow-changing issues of the network and fast loops react to the events or demands that need quick response. Both of control loops drive the process of network management towards the most robust state.

We first examine the topology design of networks. We compare designs obtained using different graph metrics. We consider well-known topology classes including structured and complex networks, and we provide guidelines on the design and simplification of network structures. We also compare robustness properties of several data center topologies. Next, the Robust Survivable Routing (RSR) algorithm is presented to assign working and backup paths to online demands. RSR guarantees 100%

single-link-failure recovery as a path-based survivable routing method. RSR quantifies each path with a value that represents its sensitivity to incremental changes in external traffic and topology by evaluating the variations in network criticality of the network. The path with best robustness (path that causes minimum change in total network criticality) is chosen as primary (secondary) path.

In the last part of this thesis, we consider the design of robust networks with emphasis on minimizing vulnerability to single node and link failures. Our focus in this part is to study the behavior of a communication network in the presence of node/link failures, and to optimize the network to maximize performance in the presence of failures. For this purpose, we propose new vulnerability metrics based on the worst case or the expected value of network criticality or algebraic connectivity when a single node/link failure happens. We show that these vulnerability metrics are convex (or concave) functions of link weights and we propose convex optimization problems to optimize each vulnerability metric. In particular, we convert the optimization problems to SDP formulation which leads to a faster implementation for large networks.

to my wife, my daughter, my mother

and

the memory of my father

Acknowledgements

I would like to express my sincere gratitude to my supervisor Professor Alberto Leon-Garcia for his guidance, advice, and continued support throughout my research. I would like to thank him for his insightful discussions and ideas that shaped my research and led me to the completion of this thesis. I am also much grateful to his patience and willingness throughout my work on this thesis. I am much indebted to his kind sharing of his expertise and research insight, and it has been a distinct privilege for me to work with him.

I wish to thank the honorable members of my committee, Prof. Baochun Li, Prof. Ben Liang, Prof. Ted Szymanski, and Prof. Elvino Sousa for evaluating my thesis and providing their invaluable comments and feedback.

I would like to thank my friend Ali Tizghadam for providing me with the assistance and resources through my entire PhD program.

I would also like to thank the university staff members, especially Ms. Linda Espeut, Mr. Vladimirio Cirillo and Ms. Darlene Gorzo for their generous help and administrative support.

Special thanks to all of my friends at the Network Architecture Lab for their diligent help and support all through my PhD years especially Hadi Bannazadeh, Armin Ghayoori, Nadim Abji, Ivan Hernandez, Houman Rastegarfar, Ramy Farha, Hagop Koulakezian, and Weiwei Lee.

Finally, I would like to express my greatest gratitude to my dear wife Leili, my lovely daughter Ghazal, and my dearest mother. I feel proud and consider myself to be fortunate to have their encouragement and support. To them I dedicate this thesis.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Network management and design is a major challenge in communication networks. While the initial design of a communication network is a complicated task where the network should be designed optimally for multiple related objectives, network management is an ongoing process that allocates network resources to various demands and generates options to modify the initial design of network to adapt to new requirements. While management and evolution of communication networks today is done with human intervention, the rapid growth of communication networks necessitates the eventual autonomous management of these networks. To respond to this need, our solution approach to network management in this thesis is based on autonomics that try to realize a network as a self-managed entity. Self-management of a communication network in the autonomics framework has different aspects of self-configuring, self-healing, self-protecting, and self-optimizing.

The focus of this thesis is to support the design of an autonomic management system through the development of algorithms using graph-theoretical concepts. For this purpose, we model the network with a weighted graph and use several metrics such as network criticality, algebraic connectivity, and vulnerability metrics to quantify

the robustness of the network, and develop algorithms and optimizations to realize a self-managed communication network. In the rest of this chapter we first describe various issues in network management which lead to the need of autonomics. Next we explain the autonomic management as a solution approach for these issues. We then give an introduction to the application of graph theory concepts in the autonomic management of a communication network, and explain the contributions of this thesis.

1.1 Design and Management Issues

Recently carriers have attempted to consolidate all the network needs into a single IP-based network called the Next Generation Network (NGN). An all-IP packet network that handles voice, video and data is an accepted framework for future communication networks for both mobile and fixed communication. On the other hand, large-scale Next Generation Data Centers (NGDCs) are becoming important in providing applications over the Internet. The design and management of the next generation networks for service providers as well as data centers provides interesting challenges that we consider next.

1.1.1 Design Issues

The first problem we face in developing a network is the design of the physical topology. Many constraints and objectives should be considered in the physical topology design. The geographical restrictions and physical location of the nodes as well as budget and traffic demands between different nodes are the first considerations in physical network design. We also should design topologies that are robust to failures and do not lose connectivity easily. For structured networks such as data center net-

works scalability is the next issue that should be addressed. Network planning which is the assignment of the appropriate capacity to each link is the next step after the initial design of the network.

The evolution of the network topology is a continuous process that should be handled in long term. As the clients and applications that utilize the network change over time, the network should adapt to these changes to be able to handle the new requirements and new traffic demands. The evolution of the network can be in the form of dimensioning (changing capacity of links), adding a link or a group of links, and deleting some links (simplification). The management system should monitor all the elements of the network, and make decisions on how to modify the network according to traffic demands during the time. Here we note that design issues in general are long term in nature, and should be performed through control loops that act slowly.

1.1.2 Management Issues

After initial topology design and network planning, there come several issues that should be addressed by the network management in future communication networks. In this part we briefly explain some of these issues and challenges.

-Application Trends and Changing Requirements As the Internet has emerged as a general information network, various applications have appeared with different requirements. While some of the applications like email need reliable delivery of data and timeliness is not important for them, some other applications like telemedicine have timeliness requirements for their data, and delay more than a certain limit is not tolerable for them. The TCP-IP protocol cannot manage time sensitive applications since it is only designed for reliable transfer

of data. Protocols for best effort delivery of data guarantee neither reliability nor timeliness. Therefore more sophisticated methods and algorithms are needed to handle diverse application types over the Internet. Since new applications may expand rapidly, network management should be able to adapt itself to these changes fast enough.

-Resource Management Different requirements and specifications for various applications or clients of the network brings the need to assign guaranteed resources to some of the applications or network clients to deliver the desired Quality of Service (QoS) to them. Resource allocation for different clients should be done based on their Service Level Agreement (SLA), and the rendered service should be monitored and controlled through control loops that are faster in nature than the ones we mentioned before in design section. Bandwidth management is an important aspect of the resource management system.

- Reliability Concerns An important issue in resource management of next generation networks is addressing the reliability concerns in order to satisfy availability requirements for the network clients or applications. The resources allocated to each application should be restored rapidly and intelligently in case of node/link failures. This is an important aspect of self-healing property in a self-managed system.

- QoS Issues Different flows in a communication network have different QoS requirements in terms of delay, jitter, and packet-loss. Future communication networks should have mechanisms to realize the necessary QoS for each traffic type and be able to monitor the QoS that is delivered to each flow.

- Cost of Management While the traditional management of communication net-

works is done with the human intervention through expert network administrators, this method of management is prone to errors, slow, and extremely costly. These problems lead to the overall management cost of 70% of Information Technology (IT) budget [1], and expansion of the network over time adds to the cost and complexity in the network management. Automation of the network management can significantly decrease the Operational Expenses (OPEX) of network management, and increase the network scalability and its ability to recover from failures from both cost and complexity points of view. In addition, more efficient use of network infrastructure leads to decrease in Capital Expenses (CAPEX) in turn.

1.2 Solution Approach: Autonomics

Future communication networks are growing in size and complexity with a variety of clients and applications with different requirements running on them. Managing these networks requires new management solutions. Network Management (NM) for future networks should be able to manage the requirements of all applications and clients and have the ability to adapt itself to the changing requirements of each client over time. NM should also be able to handle faults and failures seamlessly without performance degradation. It should be able to manage monitoring functions and optimize resource utilization to bring down the cost of network operation. In addition, NM should provide a secure environment for clients/applications and let them protect themselves from possible attacks. Finally, NM should be self-managing and do all these management functions intelligently with minimum human intervention.

The best solution to the abovementioned management requirements is autonomic management system. The term autonomic comes from an analogy to the autonomic

central nervous system which adjusts itself to many situations automatically without any external help. Autonomic management system should have four attributes to be considered self-managing. These attributes are self-configuring, self-healing, self-optimizing, and self-protecting.

Self-Configuring - This attribute corresponds to the ability of the NM to configure the network to provide applications/clients with their resource requirements and adapt itself to changes in their needs.

Self-Healing - This attribute refers to the ability of NM to handle faults or internal failures automatically without interruption or degradation in its services.

Self-Optimizing - NM should be able to maximize the usage of its active resources and minimize its cost of operation automatically.

Self-Protecting - A self-protecting NM allows authorized people to access the right data at the right time and can take appropriate actions automatically to make itself less vulnerable to attacks on its run-time infrastructure and business data.

The realization of the attributes of self-managing systems happens through building blocks that collaborate using standard mechanisms such as web services. These building blocks are in the form of control loops, and as the road map of IBM to the autonomics [2] suggests, each control loop should include Monitor, Analyze, Plan, and Execute (MAPE) parts, and is called a MAPE control loop [3]. Here is a brief description of the functionalities of each part of a MAPE control loop:

Monitor - The monitor part provides the mechanisms that collect, aggregate, filter, manage, and report details (metrics and topologies) collected from an element.

Analyze - The analyze part provides the mechanisms that correlate and model complex situations. These mechanisms allow the autonomic manager to learn about the IT environment and help predict future situations.

Plan - The plan part provides the mechanisms that structure the action needed to achieve goals and objectives. The planning mechanism uses policy information to guide its work.

Execute - The execute part provides the mechanisms that control and enforce the execution of a plan through effectors.

These four parts work together to provide the control loop function. In general, a large number of autonomic managers cooperate to manage the whole system.

1.3 Utilizing Graph Theory Concepts

A key factor in design and management of a communication network is the robustness of the network to the environmental changes. Robustness is considered as the ability of the network to adapt itself to the environmental changes and continue to operate efficiently in the presence of these changes. There are three main categories of the environmental changes that may affect the operation of a communication network: changes in network topology or the capacity of links, variations in the community of interest (the set of active source-destination pairs), and changes in the traffic demand for each active source-destination pair. We utilize graph theoretical concepts to define robustness metrics that capture the environmental changes and provide algorithms for design and management of communication networks based on these metrics. The basic metrics of interest in our work are network criticality and algebraic connectivity,

and vulnerability metrics are defined based on the variations of these metrics when a failure happens in the network.

Network criticality is a graph-theoretical metric of network robustness to the variations in network topology and capacity as well as traffic shifts where the network is modeled as a weighted graph [4]. Network criticality is a global metric that is equal to the ratio of the betweenness to weight for any link/node in the network when a random walker traverses the network from a random source to a random destination. Therefore the lower network criticality results in lower betweenness in the network which is more desirable. Network criticality is a convex function of link weights in a weighted graph, and minimizing this metric can introduce a robust network design [4]. Algebraic connectivity is another graph-theoretical metric that quantifies the connectivity of a network [5]. Algebraic connectivity is a concave function of link weights, and its maximization can be considered as a method of robust weight assignment in a communication network to optimize connectivity [6, 7]. We compare algebraic connectivity with network criticality in various contexts in this thesis. The last set of metrics examined in this thesis are vulnerability metrics which are defined as the fluctuations or the expected value of some basic robustness metrics (such as network criticality or algebraic connectivity) when a link or node failure happens in the network. We use these metrics to design networks to have the best robustness properties in case of failures.

1.4 Proposed Solution Overview

The main contribution of this thesis is to provide algorithmic solutions for an autonomic management system for communication networks using graph theoretical concepts and metrics. We consider a management system which is composed of two

types of control loops as suggested in [4]. The first type are long-run control loops that deal with issues that change slowly over time like network topology. The second type of the control loops are short-run control loops that handle issues that change with a fast pace and need quick response from the network management like online routing of network demands or reacting to a failure. We try to maximize the robustness of the network when we design its topology or when we repeatedly modify the topology or the link weights through slow control loops. These methods try to realize the self-optimizing attribute of our autonomic management. We also present a survivable routing solution by which we assign the paths to the various demands and reserve backup paths for them in an optimal way in a fast control loop that reacts to the incoming requests for the network. The backup paths help the management to realize the self-healing aspect of the autonomic management system. Here we present a brief description of the different parts of our proposed solution presented in this thesis in the following chapters.

Robust Topology Design - To tackle this problem we examine the behavior of network criticality and algebraic connectivity in a large number of structured topologies and complex networks, and provide guidelines to compare and contrast different topologies and structures. We also investigate the variations of network criticality and algebraic connectivity when the link and node connectivity changes in a complex network, and provide guidelines to design and simplify complex networks such as ER random, small-world, and scale-free networks that are the suggested models to describe the complex networks. Moreover, we compare the robustness and vulnerability metrics of different topologies of interest for the new generations of data centers such as Virtual Layer 2 (VL2), Portland, and BCube. Some parts of our contribution on network topology design

are presented in [8].

Robust Survivable Routing (RSR) - We present survivable routing algorithm based on the Weighted Random-Walk Path Criticality Routing algorithm (WRW-PCR) WRW-PCR defines link cost to reflect its sensitivity to the changes in traffic demand, network topology, and active source-destination pairs. We assess the criticality of each link and attempt to choose the path with minimum overall criticality. We use shared backup protection to provide survivability in our algorithm. Shared backup protection permits bandwidth sharing among backup paths to save resources while guaranteeing full failure recovery. Routing for shared protection involves identifying working and backup paths that optimize the total bandwidth consumption. The shared backup path protection (SBPP)[9] problem can be divided in two parts. The first part deals with assigning appropriate restoration bandwidth on each link to guarantee 100% single-failure recovery, and the second part is to choose the best backup path for any active (working) path to minimize the total cost. To address the first problem, we consider three variables for each link, the working bandwidth, the backup or restoration bandwidth, and the available bandwidth of the link. When these variables are changing we check the new network condition and apply appropriate changes to continue guaranteeing 100% failure recovery using WRW-PCR. Some parts of our contribution on robust survivable routing is presented in [10].

Design for Minimum Vulnerability - The last method we have examined in this thesis is design for minimum vulnerability [11]. The objective of vulnerability analysis is to study the behavior of network when an unwanted link/node failure happens or an intentional attack discontinues the operation of a part of network.

The important feature of the network here is its ability to operate efficiently and optimally in case of node/link failures, and it is also an aspect of the network robustness. Therefore, the main focus here is robust network design with emphasis on robustness of network when different failures happen. In order to quantify the vulnerability of a communication network to a failure, we consider the changes in the network parameters, such as network criticality and algebraic connectivity, due to the failure as a measure of network vulnerability to that failure.

The first set of our metrics quantify the worst case (maximum possible value) and expected value that network criticality may take after a single node/link failure. We show that these metrics are convex functions of link weights and we introduce and solve convex optimization problems to minimize these metrics. We note that each optimization problem can be useful depending on the nature of the network under study and its possible failures. The second set of our proposed metrics are defined based on the worst case (minimum possible value) and expected value that algebraic connectivity may take after a single node/link failure. We show that these metrics are concave functions of link weights and introduce and solve convex optimization problems related to each metric. Parts of our contribution on different aspects of vulnerability analysis and its applications in network design are presented in [11, 12, 13, 14].

Chapter 2

Related Work

In this chapter we present the literature related to different aspects of our work. We first start with a review of the works related to robustness in a communication network. Next we review two robustness metrics that are compared in this thesis i.e. network criticality and algebraic connectivity. Network criticality is the main robustness metric in the following chapters and its interpretations justify its application as a robustness metric. The works related to algebraic connectivity as another robustness metric of interest are described afterwards. In the next part, we review the works related to robust survivable routing. We first explain different routing methods studied in the literature and explore various approaches to survivable routing. A review of complex networks and their different implementations as the basis for chapter 3 comes next. We then explain the literature around network vulnerability as the basis for chapter 5, and finally the autonomic computing and its architecture are explained.

2.1 Robustness in a Communication Network

Robustness in a communication network is an important aspect of its design and operation, and is studied extensively in the literature. Robustness can be examined from two different points of views: robustness in network design and robustness in routing. In the robust network design, the problem is finding the best topologies for a communication network, and assign suitable capacities to the links to have the maximum robustness in case of various changes in the network like failures or malicious attacks. The robustness in routing, on the other hand, deals with presenting traffic engineering algorithms that are not affected dramatically by environmental changes, specifically variations in the traffic matrix, network topology and source-destination pairs of interest. In this section we focus on the literature related to robustness in network design, and examine robust routing later in section 2.3.1

In [15], the authors study the robustness of network topologies. They study graph-theoretical metrics to find out which network topologies are the most robust. They study the behavior of link and node connectivity as the major metrics that monitor the network robustness. Link (node) connectivity is defined as the minimum possible number of link (node) removals that makes the network disconnected. They argue that node connectivity is the most useful metric to measure robustness. They examine the relationship between node connectivity and the degree of symmetry, and conclude that node similarity and optimal connectivity are two conditions that lead to topological robustness of the networks. Considering the fact that for any graph node connectivity is less or equal to link connectivity and link connectivity is less or equal to minimum node degree of the graph in turn, the optimal connectivity is defined as a condition in which all these three are equal. Node similarity means that for any two nodes of the network there is an automorphism that maps these nodes to

each other. The authors of [15] introduce guidelines on how to construct this kind of graphs.

The authors of [16] present symmetry ratio of a network as an important measure of robustness. Symmetry ratio is defined as the ratio of the number of distinct eigenvalues of the network to its diameter. Symmetry ratio is used to quantify the robustness of the network topology to targeted attacks. The works in [15, 16] have discussed topology, but they have not considered role of the link weights in their works.

[17] presents a design method for backbone networks that is not sensitive to the traffic matrix and supports all the valid traffic matrices even in the presence of a number of link or node failures. The authors suggest using Valiant Load Balancing [18] for traffic routing. In this routing method, traffic from a source to a destination is sent to an intermediate node first, and is routed to the destination from there. The intermediate node is chosen uniformly randomly from all the network nodes, other than source and destination. The problem of this method is that the packets experience a longer delay to their destinations.

In [19] authors have optimized a graph-theoretical robustness metric called network criticality to design a robust network. Network criticality is introduced in [4]. The authors define network criticality as a global measure for the network robustness through combining of betweenness with the weight matrix. Because of the importance of this metric in this thesis, we will discuss this metric and its interpretations later in this chapter in section 2.2.1 in detail. They have considered a maximum budget for designing the network as a linear combination of the link weights, and they have obtained an optimum weight set for the network. They have formulated the optimum weight set for several use cases such as complete graph, hypercube, and tree; and have proposed an application of their work in capacity assignment for a communication

network. This work has not considered the robustness of the network when failures happen.

2.2 Robustness Metrics

Network criticality and algebraic connectivity are two major robustness metrics that are compared in this thesis in various applications. Algebraic connectivity was first introduced by Fiedler [5] [6] as a robustness metric for graphs, while network criticality was first introduced in our research group as a more effective metric to quantify robustness of a graph [4]. In this section, we first describe network criticality and its interpretations that justify its usage as a robustness metric, then we explain algebraic connectivity and the literature around it to explain why we have chosen it as a comparison base to network criticality.

2.2.1 Network Criticality

Consider a weighted, undirected, connected graph, where the link weights show the desirability of a link to contribute in data transfer. The link weights can be related to QoS parameters in a way that larger beneficial QoS parameters (like link capacity or available bandwidth) increase the link weights and larger detrimental QoS parameters (like packet loss) decrease the link weights. We also assume that SLA's can be mapped to the link weights with an appropriate method. Some of these methods are discussed in [4, 20, 21].

Network criticality is a robustness measure defined on a weighted graph to quantify the sensitivity of the graph to the environmental changes. Conceptually, network criticality is related to the definition of random-walk betweenness in graphs. Consider a set of trajectories walked by a random walker, starting at s and terminating when

the walk first arrives at destination d. The random walk betweenness of a node k for the set of trajectories from s to d (denoted by $b_{sk}(d)$) is defined as the average number of visits to node k. It has been proved that for a generic random walk, where the probability of transitioning along a link to a neighbor node is proportional to the weight of that link , quantity $\frac{b_{sk}(d)+b_{dk}(s)}{W_k}$, where W_k is the total sum of the link weights incident to node k, is independent of k [4]. We refer to this quantity as point-to-point network criticality and denote it by τ_{sd}. If we choose to define the link weight as available capacity of a link, then point-to-point network criticality quantifies the risk of sending a stream of packets via trajectories between s and d [4].

The total random walk betweenness of node k is the sum of the contributions for all $s-d$ trajectories. The normalized random walk betweenness of a node (i.e. the node betweenness divided by the node weight) is a global measure on the graph and it is independent of the node location [22]. We refer to this global graph metric as network criticality and we denote it by τ.

Network criticality has some nice properties and interpretations, which explain why it is an important robustness metric on graphs. Consider an equivalent electrical circuit with the same graph as our original network graph, and with link conductances (reciprocal of link resistances) equal to the link weights of the original weighted graph. Point-to-point network criticality τ_{sd} can be interpreted as the resistance distance (effective resistance) [23] between two end nodes s and d, and network criticality will be the total resistance distance seen between different pairs of nodes in the electrical circuit. A high network criticality is an indication of high resistance in the equivalent electrical circuit; therefore, in two networks with the same number of nodes, the one with lower network criticality is better connected, hence better positioned to accommodate network flows.

2.2.2 Algebraic Connectivity

The notion of algebraic connectivity of a graph was first introduced by Fiedler [5]. He defined algebraic connectivity as the second smallest eigenvalue of the Laplacian matrix [24, 25, 26, 27, 28]. We know that all the eigenvalues of the Laplacian matrix are nonnegative and the smallest one is always zero [5]. He proved that algebraic connectivity of graph is a measure of the graph that increases when a link is added to the graph. He also showed that algebraic connectivity is a lower bound to node and link connectivity. Therefore higher algebraic connectivity can result in higher connectivity, and it can be considered as a measure of robustness for a network. He formulated algebraic connectivity for some well known graphs like complete graph, path, ring, star, and hypercube.

While in [5] Fiedler introduced the concept of the algebraic connectivity in an unweighted graph, the problem of maximizing the algebraic connectivity in a weighted graph with a constant sum of link weights is introduced in [6, 7]. In [6] Fiedler names the maximum value of the algebraic connectivity that the graph can take as absolute algebraic connectivity, and derives analytical formulation for the absolute algebraic connectivity of some special graphs like trees. This problem is analyzed in [29] in the context of finding the fastest mixing Markov process (FMMP) problem. Boyd et al. show in this work that algebraic connectivity of a network determines the mixing speed of the Markov process on a weighted graph when the weights show the transition rate between adjacent nodes. They show that the distribution of Markov process converges to the uniform distribution in an exponential form at a rate determined by algebraic connectivity. They consider a maximum for the linear combination of the weights, and maximize algebraic connectivity to reach the maximum speed. They show that FMMP problem is convex optimization problem, and its dual is equivalent

to the maximum variance unfolding (MVU) problem. MVU has the interesting geometric interpretation of choosing a set of points as far as possible, measured by their variance, while limiting the Euclidean distance of each pair of the points.

Algebraic connectivity is also considered as a measure of network stability and robustness in many system problems defined over networks [30, 31, 32, 33]. In all of these works, it is observed that when there is a small perturbation of the system parameters from equilibrium state, the system returns to the equilibrium with a rate that is proportional to the algebraic connectivity. [33] provides a distributed algorithm to maximize algebraic connectivity. None of these works has examined the behavior of algebraic connectivity when failures happen like what we have done in this thesis. We compare the behavior of algebraic connectivity as a topological robustness metric with network criticality in the context of complex networks as well.

2.3 Survivable Routing

The first application discussed in this thesis is robust survivable routing. In this section we first review the literature around routing methods, and works on robust routing. Then we explain the WRW-PCR as the robust routing method we have used in our work, and finally review the works related to survivable routing.

2.3.1 Routing Methods

Routing methods are the algorithms that deal with traffic engineering in order to maximize the network utilization. Traffic engineering algorithms has gained a lot of attention in the last few years [34, 35, 36, 37, 38]. The main focus of the routing algorithms in this thesis are the core networks that use MPLS (Multi Protocol Label Switching) [39] as their routing protocol or the optical networks with full wavelength

conversion. The main task of routing algorithms in these networks is finding a path for each traffic demand in a way that the networks get utilized maximally. There are well known solutions for this purpose including MIRA (Minimum Interference Routing Algorithm)[40], PBR (Profile-Based Routing) [41], and MATE (MPLS Adaptive Traffic Engineering)[42].

MIRA is one of most renowned routing algorithms which considers the effect of a new LSP (Label Switched Path) on the existing LSPs in the network. MIRA computes the maximum possible flow between every source-destination (sd) pair and chooses a new LSP that maximizes a linear combination of these maximums for all sd pairs. The problem of MIRA is its high complexity in addition to its weak performance in some cases as reported in [41]. PBR is the other routing algorithm proposed for MPLS networks. PBR performs the routing in offline and online phases. PBR assumes that the traffic demand for each sd pair is known ahead of time. In offline phase, it uses multicommodity flow assignment optimization to maximally assign capacities on the network links to each class of traffic. In online mode it simply routes the traffic using the assigned capacities in offline mode with shortest path algorithm. PBR has less complexity compared to MIRA, but there are some cases in which its performance is worse than WSP (Widest Shortest Path) as reported in [43]. Finally, MATE is a routing algorithm for MPLS networks that adaptively balances the load along multipaths to lower congestion in the network links using the feedback it gets about the packet loss and delay in destination nodes. MATE is based on a quasi-static routing proposal by Gallagar [44]. MIRA, PBR, and MATE have a better performance than WSP and SWP (Shortest Widest Path) [45]; however low complexity and near to optimal performance of WSP (and SWP) makes it a proper benchmark.

The other important class of routing algorithms are robust routing algorithms.

The problem in robust routing is how to enable the network to support a varying traffic matrix even when circumstances like failures happen in the network. A lot of work has been done for this purpose, and we mention two main categories here. The first category of robust routing is finding a robust solution to minimize the maximum regret, which means the worst case scenario in the network [46, 47, 48, 49, 50, 51]. The other category is of algorithms in the area of robust routing is oblivious routing [52, 53, 54, 55, 56, 57]. In oblivious routing the objective is to optimize the performance in the worst case scenario over all traffic matrices, therefore the routing scheme can support any dynamic changes in the traffic matrix. [58] proposes a framework for robust routing in core networks based on the concepts of "link criticality" and "path criticality". [59] introduces a robust routing algorithm called Weighted Random-Walk Path Criticality algorithm (WRW-PCR) which is based on the optimizing the network criticality as a robustness metric of the network. This algorithm defines link costs in a way that it reflects its sensivity to the changes in traffic demand, network topology and active source-destination pairs.

2.3.2 Survivable Routing Methods

Fast restoration of network resources after a failure has been an important aspect of the backbone networks, and has attracted a lot of attention in the last few years [60, 9, 61]. Survivability of the network deals with the ability of the network to restore its services after a failure without service interruption. There are different techniques of protection or restoration for this purpose. While protection techniques reserve fixed resources for protecting their active resources, restoration techniques provide backup resources dynamically when a failure happens. Therefore, protection techniques are generally faster, but consume more resources as backup. Shared Backup

Path Protection (SBPP) is a protection technique that combines the backup resources to lower the amount of backup resources while keeping the ability to recover quickly from failures. SBPP has different flavors based on the available information of other system routes and their backups.

In [9] the SBPP is discussed within the context of MPLS networks. The authors of [9] consider three service provisioning cases: (1) No Sharing (NS), (2) Partial Sharing of routing Information, and (3) Full Sharing of routing Information (FS). While in NS it is assumed that only the available capacity of each link is known to make decisions on backup path selection, in FS the assumption is that backup capacity used on each link due to any link failure in the network is known. A spare provision matrix (SPM) based SBPP method is also proposed in [62]. The major assumption is that full routing information can be shared throughout the network. Based on this information, a Successive Survivable Routing (SSR) algorithm is proposed, which can provision survivable services in an online fashion.

Meanwhile [63] and [64] study the SBPP problem based on the assumption of full routing information. Two complementary vectors (also called profiles) are specifically maintained by the network control system, which has the information on how much protection capacity on a span is used to protect the working capacity on other spans, and how much working capacity on a span is protected by the spare capacity on other spans. The routing information in [63] and [64] is mainly maintained in an aggregated way, and it is suitable for the networks such as MPLS and optical networks with full wavelength conversion capability, in which the bandwidth of each connection is continuous or each capacity unit need not be explicitly discerned from one another. In this paper we study the behavior of WRW-PCR in both NS and FS cases.

2.4 Complex Networks

Many of the real networks fall in the range of complex networks like the Internet, social networks, transportation networks, infrastructural networks like water, electricity and gas, human brain, etc. Therefore it is very important to study complex networks and learn about their behavior and characteristics. There is not a unique and comprehensive definition for the complex networks, and they are mostly known through their instantiations [65]. Complex networks have caught a lot of attention during the last few years [66, 65, 67, 68]. Traditionally, most of the complex networks have been be modeled as Erdos-Renyi random graphs [69]. However, the observation that many of the real networks do not follow the properties of random graphs, e.g. [70], motivated many researchers to propose new models for complex networks such as small-world [71] and scale-free [72]. In fact, other than regular graphs, Erdos-Renyi random graphs, small-world graphs, and scale-free graphs are the main classes of complex network in the literature.

In addition to the modeling of the complex networks, analysis and study of statistical properties have gained much attention. [73] by Newman, and [65] by Albert and Barabasi are two good detailed and comprehensive references in this regard. There are also a lot of work done on the spectral analysis of the complex networks and its relation to the network robustness [67, 74, 75, 76]. In [74] the topological robustness aspects of algebraic connectivity in the context of complex networks are investigated. The authors sketch the curves for algebraic connectivity vs. node (link) connectivity for different classes of complex networks through extensive simulations. They observe that within a specific class of random graphs algebraic connectivity increases with increasing node connectivity, and it can be considered as a robustness measure to node/link failures. However, although in ER random graphs algebraic connectivity

is a relatively tight lower bound for node connectivity, for small-world and scale-free graphs it is a loose lower bound, and for graphs with the same number of nodes, links and node (link) connectivity there is no clear relation between algebraic connectivity and node (link) connectivity. We will compare the topological behavior of algebraic connectivity and network criticality in the three models of complex networks as well as some structured graphs in chapter 3.

2.5 Vulnerability Analysis

There are many physical real-world events such as human attacks or natural disasters [77, 78] that can disable parts of a communication network from normal operation. Vulnerability analysis deals with finding the vulnerable and weak points of the network, and try to strengthen them in order to make the network less vulnerable to disasters or attacks. Vulnerability analysis of the networks has been examined in several works [79, 80, 81].

In [79], similar to the the well-known "network inhibition problem" [82], the authors have considered a destruction cost for each link of the network and a fixed budget to attack the network. They have also introduced four performance metrics for the network such the maximum flow between a given pair of nodes. The authors try to find circular or linear cuts that optimize their performance measures, and they find the most vulnerable areas of the communication network using these kinds of searches. The drawback of this work is that they have not optimized the network to minimize its vulnerability to the correlated failures of their interest.

[81] examine the vulnerability of the complex networks to the attacks to the nodes or links of the network. They have considered different classes of complex networks, and have investigated which class of complex networks are less vulnerable to the at-

tack to a certain percentage of the network nodes through the degradation of some performance metrics such as the reciprocal of the average path length. Authors of [80] have defined vulnerability functions of a graph in terms of number of nodes and links, and minimum and maximum node degree to compare the vulnerability of different topologies to the random and international attacks. None of these works has considered the effect of link weights in their analysis of the network vulnerability. In this thesis we have a systematic approach to vulnerability analysis by introducing vulnerability metrics based of the worst value or the expected value of basic robustness metric such as network criticality or algebraic connectivity in case of link/node failures.

2.6 Autonomic Management

Autonomic management refers to the characteristic of a network (or in general IT) management system to be able to adapt itself to the changes in system conditions, objectives and policies [2, 83]. The term "autonomic" roots in the similarity of an autonomous system to the central nervous system of human body to adapt itself and react to the environmental changes in an automatic way without need to any external help or a conscious effort. In a similar way, an autonomic management system for an IT infrastructure should be able to adapt itself to its changing business objectives and policies and also should have the ability to protect and heal itself automatically. The autonomic systems are called self-managing systems as well. As we explained before, an autonomic computing system should include four attributes of self-configuring, self-healing, self-optimizing, and self-protecting which is illustrated in Fig. 2.1.

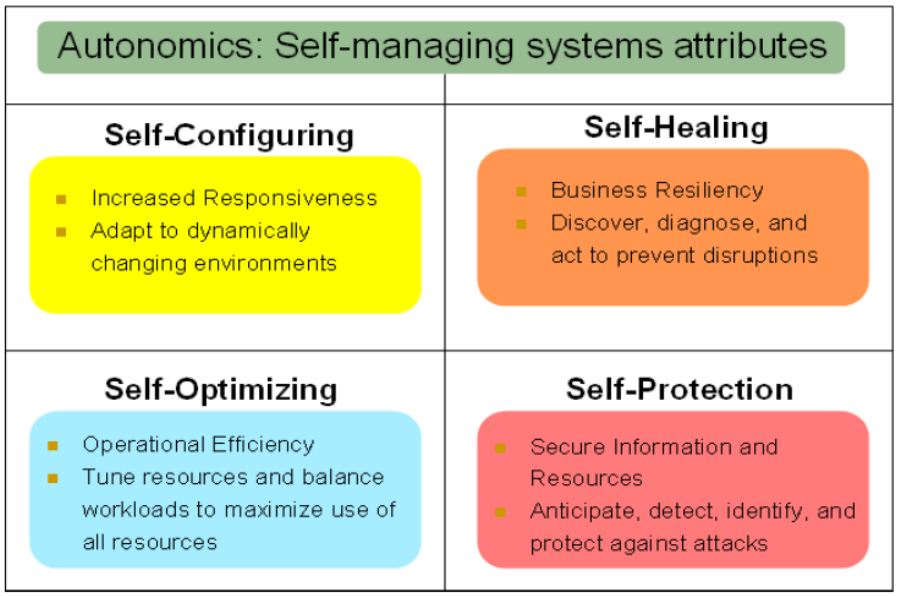


Figure 2.1: Attributes of self-managing systems

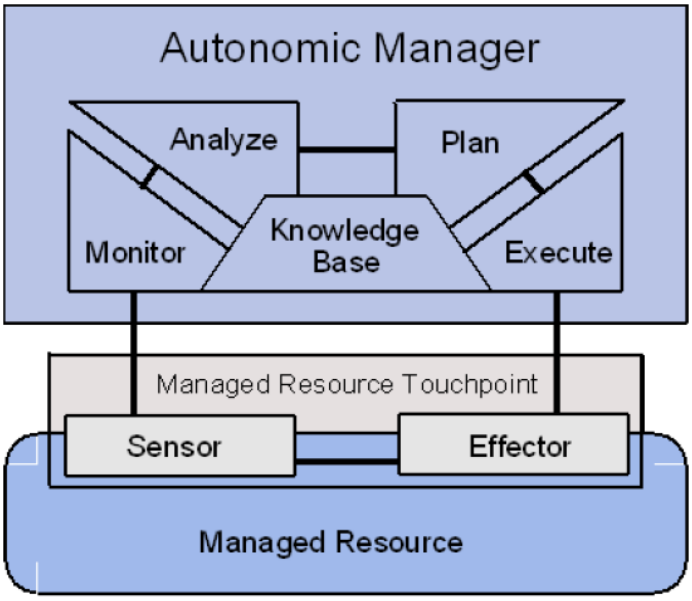


Figure 2.2: The control loop in autonomic management system

2.6.1 Autonomic Computing Architecture

Autonomic computing is implemented through five building blocks: Autonomic managers, Knowledge source, Touchpoints, Manual managers and Enterprise service bus. Autonomic computing architecture is composed of several layers. Managed resources are at the lowest level of this architecture and are connected to the autonomic managers through touchpoints. There are distinct autonomic managers for each attribute of self-management, namely self-configuring, self-healing, self-optimizing, and self-protecting for every managed resource. At the higher level, tthere are autonomic managers that orchestrate autonomic managers at the lower levels based on the system policies. At the highest level, the manual manager has a management interface to the system, and all of the autonomic managers as well as manual manager use the knowledge base to make decisions. All the components of the autonomic architecture are integrated through the enterprise service bus that utilizes technologies such as web services to communicate with all other components [2].

The main building block of the autonomic architecture is autonomic manager. Every autonomic manager implements a control loop as shown in Fig. 2.2 to render its management duty. This control loop has four distinct components of Monitor, Analyze, Plan, and Execute which is called a MAPE loop in brief [3]. The monitor part collects and processes the data from the system details such as various metrics or the topology details from the managed element and reports them to the knowledge base. The analyze block provides tools to correlate the data and model complicated situations in order to learn about the system environment and be able to predict future situations. The plan part uses policy information together with the results of the analyze block to decide about the actions needed to achieve the system objectives. Finally, the execute block provides mechanisms to realize and control the execution of the plans.

2.7 Control Loops for Communication Networks

We consider an autonomic management system for a communication network which is composed of slow and fast control loops as suggested in [4]. We consider robustness as the main property of interest for the communication networks, and conduct the design and management process in a path to the most robust state for the network. Based on the robustness definition, the autonomic management system as a self-managed entity performs the initial topology design and plans it for the maximum robustness, and over time it repeatedly examines the network from the robustness point of view, and applies modifications to the system elements in the direction of maintaining the system in the most robust state. We consider the network criticality as an important choice to quantify robustness of the network, however, based on the design objectives, other robustness metrics can be defined and optimized by the management system. For example, in chapter 5 we have proposed vulnerability metrics as the robustness metrics of interest. The robustness metric of interest can be defined based on the nature or the functionality of the network under study.

2.7.1 Slow Control Loop

The slow control loop implements the functionalities such as topology design, network planning, and high-level performance measurement, and produces suggestions to the manual manager for modifying the topology or link weights in order to adapt the system to the slow changes of the system requirements over time. Fig. 2.3 shows a high-level view of the slow or long-term control loop. The main inputs to the slow control loop are the traffic matrix, Service Level Agreement (SLA), physical constraints, budget constraints, and management policies. While the initial input data are obtained through empirical studies from previous experiences, it gets modified

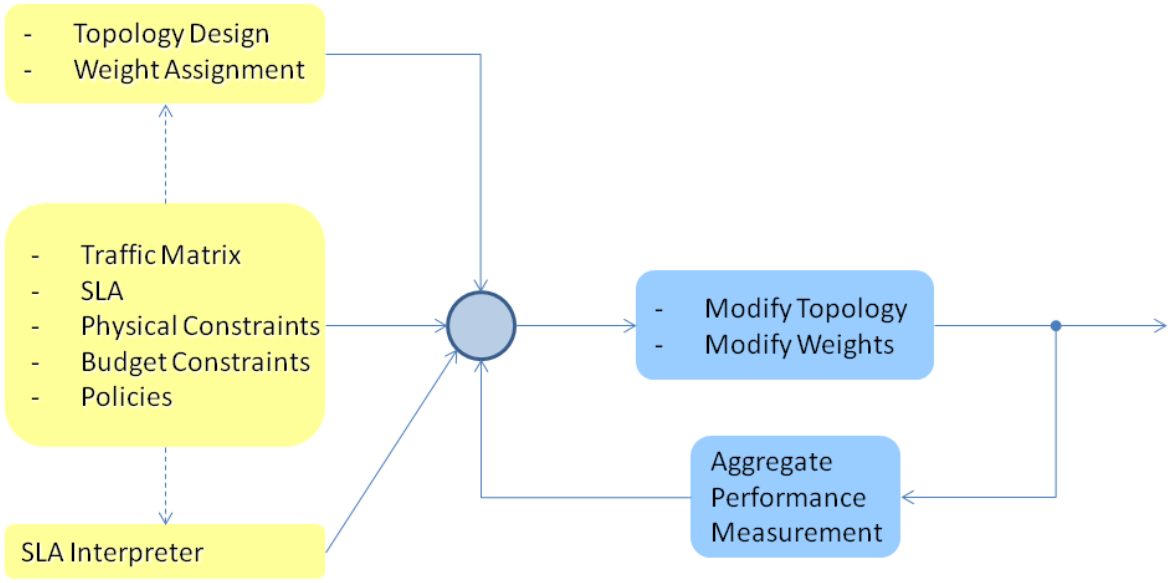


Figure 2.3: Slow or Long-Run Autonomic Control Loop

and evolved over the time using feedback data from the network operation. Fig. 2.3 shows how the initial input data are used to design the initial physical topology of the network.

In the long-run or slow control loop we have the aggregate performance measurement box that performs high-level monitoring of the network to evaluate the performance of the current topology and the assigned link weights. This box combines the feedback data from various parts of the system or the autonomic managers in lower levels. In the central part of the long-run control loop all the input data including the current topology, link weights, and traffic matrix together with the aggregate performance measurement data are filtered appropriately and get analyzed to make decisions on high-level modifications in the system. These modifications can be in the form of adding or removing a node or a link, or changing their specifications, e.g., it can suggest to increase the capacity of a link.

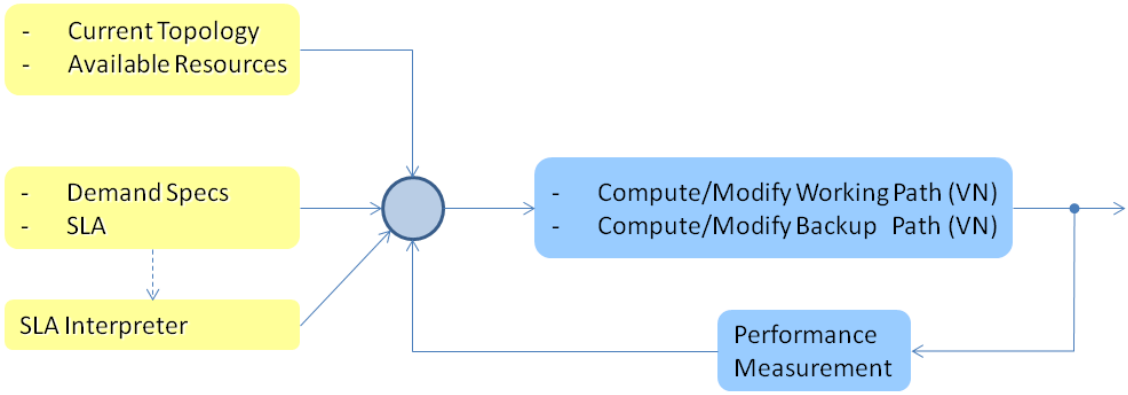


Figure 2.4: Fast or Short-Run Autonomic Control Loop

The important point here is that as the size of the large networks makes it difficult and complicated to manage the whole network as a single entity, generally the network gets divided to Virtual Networks (VNs) that are over-imposed on the same physical (or substrate) network [84, 85]. Different VNs can be assigned to various network clients or applications that have their own resource requirements. With VN approach to the network architecture, every VN has a specific initial topology and would need a separate slow loop to evolve its topology and resource needs based on its (changing) traffic matrix and SLA over time, and then another slow loop at a higher management level collects the information from all of the autonomic managers of the slow loops of the network VNs and orchestrates them by dividing the network resources between various VNs, or produce suggestions to modify the actual physical topology to provide the ability to handle all the VNs' requirements.

2.7.2 Fast Control Loop

Fast or short-run control loops respond to the system management needs that require quick responses and remedies. Fig. 2.4 shows the fast loop associated to a specific

demand. The input to this control loop is the demand specifications and SLA (and its interpretation in term of resources) as well as the current network (or the VN related to the demand) topology and its available resources. The main block in the center of the loop computes the working and backup paths (or VNs) for the demand according to the input data. If it is not possible to provide enough resources for the demand, a report is sent to the network (or VN) slow control loop unit to provide data to help it make decisions based on the policies and the aggregate feedback from other fast control loops. Again there is a performance measurement block that provides feedback to the central box and the required information to modify the working or backup resources assigned to the demand over time.

Chapter 3

Utilizing Graph metrics in Robust Topology Design

Topology design is an important aspect of autonomic management of a communication network. Topology design appears in the implementation of the long-run control loop we introduced in chapter 2 in both physical and logical layers. In physical layer the main issue is to design a physical network with the best robustness and connectivity properties that matches the physical constraints, budget constraints, and management policies as well as make decision on how to modify the network to be the most robust over time. In the logical layer, making decisions on the topology of Virtual Networks (VNs), and how to evolve them over time is an ongoing process.

In this chapter we investigate and compare properties of algebraic connectivity and network criticality, along with node degree and node betweenness, and provide some guidelines for designing and simplifying different kind of networks based on desired connectivity and robustness properties. In the first part, we present the mathematical base related to network criticality and algebraic connectivity. Then we compare and contrast robustness and connectivity properties of some structured

networks, and investigate the usefulness of various metrics in this regard. Next we investigate the behavior of network criticality in different models of complex networks when node connectivity increases and compare the results to those of [74, 75]. We believe that this gives an insight on designing complex networks that are both robust and simple while providing a desired level of connectivity. In the last part, we study the robustness properties of data center topologies as a practical application of our approach.

3.1 Network Criticality

In this section we provide a brief summary of the network model and formulation of network criticality.

3.1.1 Network Model

We model a network with an undirected weighted graph $G = (N, E, W)$ where N is the set of nodes, E is the set of graph links, and W is the weight matrix of the graph. Throughout this thesis we assume that G is a connected graph.

Consider a finite-state irreducible Markov Chain with transition probabilities p_{ij} of transitioning from state i at time t to state j at time $t + 1$ (discrete time). The Markov chain can be represented by a state transition diagram with states as nodes in a graph and edges corresponding to allowable transitions, and labels associated with the edges denoting the transition probabilities. The Markov chain can also be viewed as a random walk on the n-node graph with next-step transition probabilities

p_{ij} according to the following rule:

$$p_{ij} = \begin{cases} \frac{w_{ij}}{\sum_{k \in A(i)} w_{ik}} & if \quad j \in A(i) \\ 0 & otherwise \end{cases} \tag{3.1}$$

where $A(i)$ is the set of adjacent nodes of i and $w_{ik} \geq 0$ is the weight of link (i, k).

We are interested in quantifying the *betweenness* of a node in the random-walk corresponding to a Markov chain. The original definition of random-walk betweenness is given in [86], but here we use a modified version defined in [87]. Consider the set of trajectories that begin at node s and terminate when the walk first arrives at node d, that is, destination node d is an absorbing node. We define the betweenness $b_{sk}(d)$ of node k for the $s - d$ trajectories as the average number of times node k is visited in trajectories from s to d.

Let $B_d = [b_{sk}(d)]$ be the $n \times n$ matrix of betweenness metrics of node k for walks that begin at node s and end at node d. Further, let P_d be the matrix of transition probabilities when the random walk is modified so that state d is an absorbing state. We use $P(i|j)$ to show the truncated $(n-1) \times (n-1)$ matrix that results from removing i^{th} row and j^{th} column of matrix P. It is shown in [87] that:

$$B_d(d|d) = (I - P_d(d|d))^{-1} \tag{3.2}$$

3.1.2 Definition of Network Criticality

We now introduce *network criticality*, the metric that we proposed in [87], to quantify the robustness of a network. We start by defining node/link criticality.

Node criticality is defined as the ratio of the random-walk betweenness of a node to its weight (weight of a node is defined as the sum of the weights of its incident links). Link criticality is similarly defined as the ratio of the betweenness of a link to its weight.

Let $\eta(k)$ be the criticality of node k and η_{ij} be the criticality of link $l = (i, j)$. It is shown in [87] that η_i and η_{ij} can be obtained by the following expressions:

$$\tau_{sd} = l^+_{ss} + l^+_{dd} - 2l^+_{sd} \quad or \quad \tau_{sd} = u^t_{sd} L^+ u_{sd} \tag{3.3}$$

$$\tau = \sum_s \sum_d \tau_{sd}, \quad \hat{\tau} = \frac{1}{n(n-1)}\tau = \frac{2}{n-1} Tr(L^+) \tag{3.4}$$

$$\eta(k) = \frac{b_k}{W_k} = \frac{1}{2}\tau = \frac{n(n-1)}{2}\hat{\tau} \tag{3.5}$$

$$\eta_{ij} = \frac{b_{ij}}{w_{ij}} = \tau = n(n-1)\hat{\tau} \tag{3.6}$$

$$\frac{b_{sk}(d)}{W_k} = l^+_{sk} - l^+_{sd} - l^+_{dk} + l^+_{dd} \tag{3.7}$$

where $L^+ = [l^+_{ij}]$ is the Moore-Penrose inverse of graph Laplacian matrix L, n is the number of nodes, and $u_{ij} = [0 \ ... \ 1 \ ... \ -1 \ ... \ 0]^t$ (1 and -1 are in i^{th} and j^{th} position).

Observation 3.1.1. *Equations 3.3 to 3.6 show that node criticality (η_k) and link criticality (η_{ij}) are independent of the node/link position and only depend on τ (or $\hat{\tau}$) which is a global quantity of the network.*

Definition 3.1.2. *We refer to τ as the* network criticality *and $\hat{\tau}$ as* normalized network criticality*. In this thesis our experiments are based on normalized network criticality.*

One can see that $\hat{\tau}$ is a global quantity on network graph G. Equations 3.5 and 3.6 show that node (link) betweenness consists of a local parameter (weight) and a global metric (network criticality). $\hat{\tau}$ can capture the effect of topology and community of interest via betweenness, and the effect of traffic via weight (by appropriate definition of weight). The higher the betweenness of a node/link, the higher the risk of using the node/link. Furthermore, one can define node/link capacity as the weight of a node/link, then the higher the weight of a node/link, the lower the risk of using the

node/link. Therefore network criticality can quantify the risk of using a node/link in a network which in turn indicates the degree of robustness of the network.

Observation 3.1.3. *It turns out that the value of pair-wise criticality, i.e. τ_{sd} for pair $s-d$, is equal to the effective resistance between nodes s and d when the network is considered as an electrical network with link conductances equal to the link weights [88]. The total effective resistance of the network is also equal to the network criticality τ. In this thesis we use the terms network criticality and effective resistance interchangeably.*

In order to simplify the structure of complex networks, one needs to have a good understanding of connectivity properties of a network as well as its robustness. In this chapter our goal is to investigate $\hat{\tau}$ as a function of the weight matrix (W) and compare it with algebraic connectivity of a graph.

Network criticality can be stated as a function of eigenvalues of the graph Laplacian as follows:

$$\hat{\tau} = \frac{2}{n-1} \sum_{i=2}^{n} \frac{1}{\lambda_i} \tag{3.8}$$

where $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ are eigenvalues of the graph Laplacian L [4]. In [4] it is also proved that $\hat{\tau}$ is an strictly convex function of the link weights. We state this fact as theorem 3.1.1 for further reference.

Theorem 3.1.1. *Network criticality ($\hat{\tau}$) is an strictly convex function of link weights.*

Proof. See [4] for the detailed proof. □

3.2 Network Criticality and Algebraic Connectivity

Algebraic connectivity is defined as the second smallest eigenvalue (λ_2) of the Laplacian matrix of a connected graph. We recall that Laplacian matrix of a graph is a positive-semidefinite matrix, and all its eigenvalues are nonnegative while its smallest eigenvalue is always zero. Algebraic connectivity is a graph metric that increases when a link is added to the graph. Fiedler showed in [5] that algebraic connectivity is the lower bound to node and link connectivity in a graph. Therefore, *the further λ_2 is from zero, the higher the node and link connectivity of a graph.*

In addition, as it is shown in [4], network criticality is bounded by a multiple of the reciprocal of algebraic connectivity:

$$\frac{2}{(n-1)\lambda_2} \leq \hat{\tau} \leq \frac{2}{\lambda_2} \tag{3.9}$$

This means that increasing the connectivity of a network, decreases the upper bound of network criticality, which potentially means more robustness.

An important characteristic of algebraic connectivity is that it is a concave function of link weights. We state this fact as theorem 3.2.1 for further reference.

Theorem 3.2.1. *Algebraic connectivity (λ_2) is a concave function of link weights.*

Proof. See [29] for the proof. □

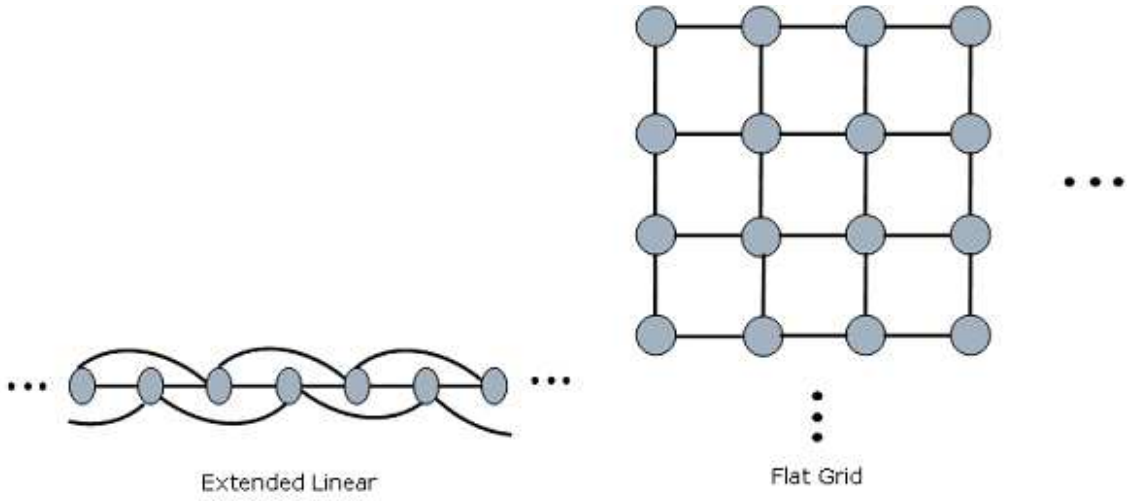


Figure 3.1: Extended Linear Graph (ELG) and Flat Grid Graph (FGG)

3.3 Deterministic Graphs

In this section, we provide an in-depth study of the behavior of algebraic connectivity and network criticality in the structured networks. We use a diverse range of structured topologies like linear graph, grid, Torus, etc. The quantities of interest for us are: network criticality, algebraic connectivity, average degree, and average node betweenness. *In all of the experiments, we assume that all the link weights are equal.* In fact, without loss of generality we assume $w_{ij} = 1 \ \forall (i,j) \in E$. While a complex network is not usually a regular or structured graph, frequently a large number of existing complex networks (for instance the network of proteins) can be built by joining some structured topologies. Therefore, in this section we study the behavior of the metrics of interest in deterministic and structured networks.

3.3.1 Extended Linear Graph (ELG) and Flat Grid Graph (FGG)

In the first experiment we consider extended linear graph (ELG), and flat grid graph (FGG) (see Fig. 3.1). In Fig. 3.2 we compare the behavior of network criticality, algebraic connectivity, average degree, and average node betweenness in ELG and FGG for different network sizes.

Fig. 3.2-a shows the behavior of network criticality for ELG and FGG. The criticality of ELG grows much faster than FGG. Fig. 3.2-b reveals that the algebraic connectivity of FGG is always better than ELG, that is the flat grid has better connectivity but the speed of decreasing algebraic connectivity of the graph is much slower than increasing the network criticality. According to the Fig. 3.2-c, the average node degree of ELG approaches 4, and the average node degree of FGG is between 3.5 and 4.0.

Finally, Fig. 3.2-d shows that the average node betweenness of ELG is higher than that of FGG. The behavior of average node betweenness and network criticality in this experiment are very close, because the variations of node degree for both networks are small, therefore, according to equation 3.5 ,the average node betweenness and network criticality are almost proportional.

This experiment reveals that while the changes in node degree and algebraic connectivity of ELG and FGG are relatively similar, there is a huge change in the behavior of network criticality, which means that network criticality captures some attributes of the graph that cannot be found in node degree and algebraic connectivity.

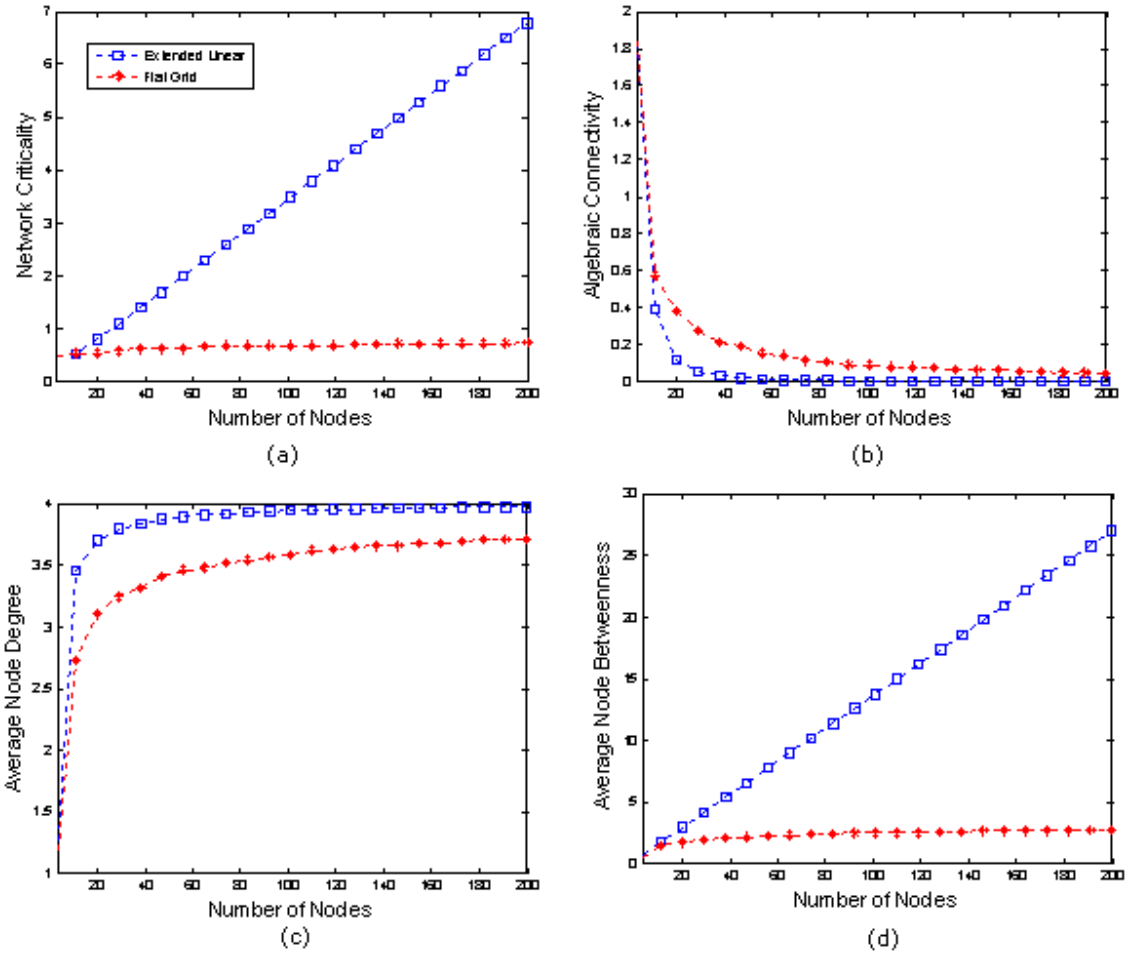


Figure 3.2: Comparison of Extended Linear (ELG) and Flat Grid (FGG) Topologies

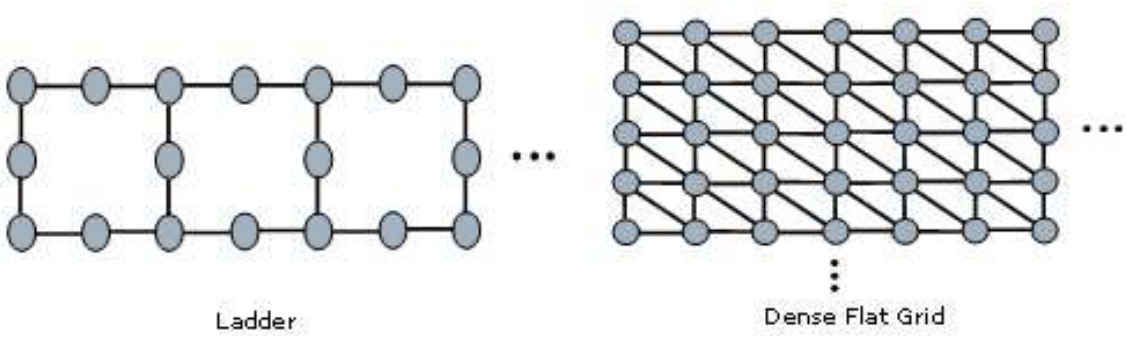


Figure 3.3: Topology of Ladder (LAG) and Dense Flat Grid Graph (DFG)

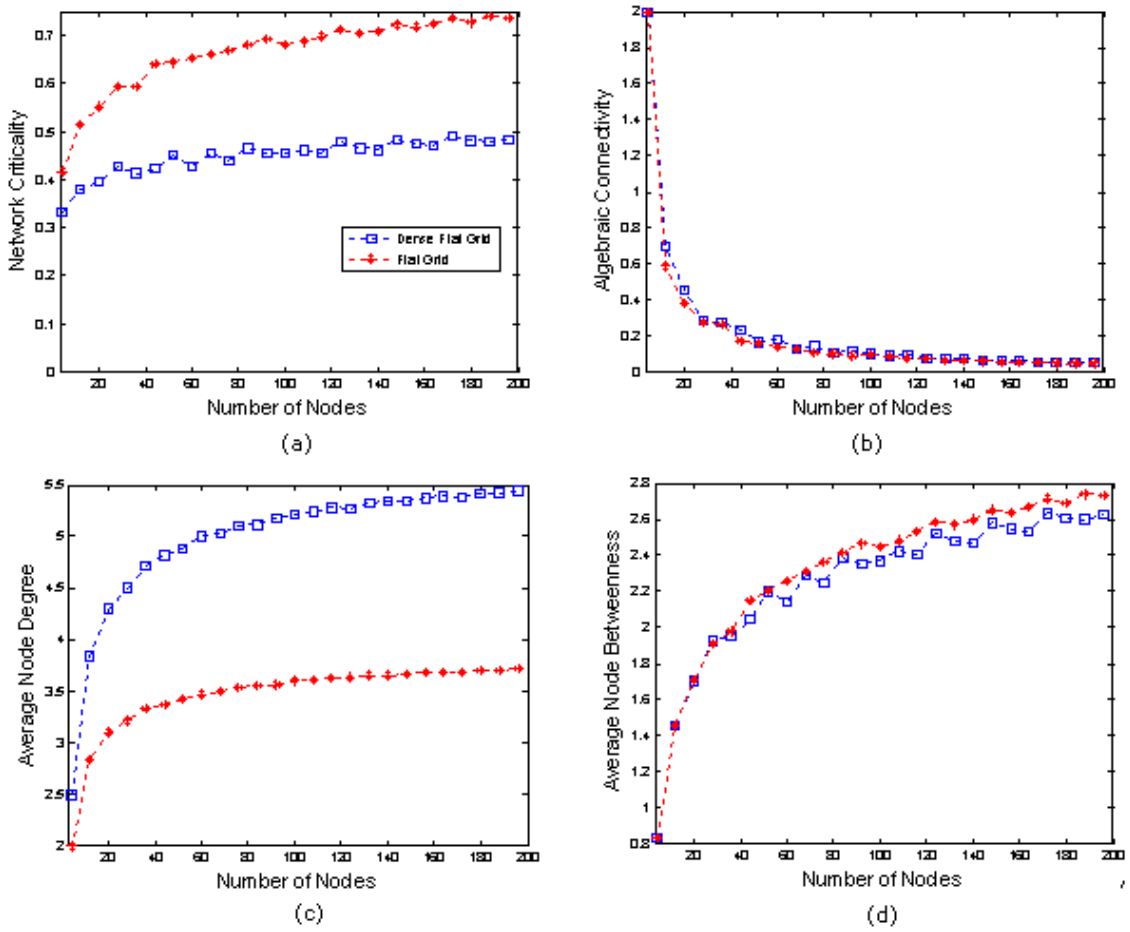


Figure 3.4: Comparison of Flat Grid and Dense Flat Grid Topologies

3.3.2 Ladder Graph (LAG) and Dense Flat Grid Graph (DFG)

Fig. 3.3 shows the ladder and dense flat grid graph (DFG), which are two other topologies of interest. In Fig. 3.4, the behavior of FGG and DFG are compared.

Fig. 3.4-a verifies that the network criticality of DFG is smaller than the network criticality of FGG with the same number of nodes. This is expected because the number of links in DFG is more than FGG. Therefore, if the objective of designing a network is to make it more robust in the sense that robustness is defined in this thesis, we need to use DFG. According to Fig. 3.4-b the algebraic connectivity of FGG and

DFG are very similar for all values of n. This means that if only the connectivity is critical in a network, then it is preferred to use FGG. In other words, if connectivity is the only concern, a DFG can be safely converted to a FGG.

Fig. 3.4-c shows the average node degree of FGG and DFG and Fig. 3.4-d shows the behavior of average node betweenness. According to this figure the average node betweenness for FGG and DFG are very similar. In other words, when the control of average node betweenness is the main objective, then it makes sense to use FGG with the same number of nodes. This will simplify the network significantly.

In the next experiment, we compare the behavior of extended linear graph (ELG) and ladder graph (LAG). According to Fig. 3.5, network criticality of LAG is slightly worse (larger) than the network criticality of ELG, while the algebraic connectivity of ELG and LAG behave similarly for different values of n. Average degree of ELG converges to to 4, while the average node degree of LAG converges to a value less than 2.5. As the total number of links in a graph is approximately the number of nodes times the average node degree, we arrive at the result that when the networks grow in node size, the number of links in ELG is much more than LAG, in other words, a slight improvement in network criticality comes at the expense of a huge increase in the number of links (while the number of nodes in both networks are the same).

Now by looking at the result for average node betweenness, we find that the average node betweenness of ELG grows faster than LAG as n grows. A final result is that in general by changing an ELG-like topology to a LAG, we do not lose too much, as a matter of fact the average node betweenness even decreases, while the network criticality (effective resistance) slightly increases. The average node betweenness has an important role in developing traffic engineering algorithms for communication networks [4].

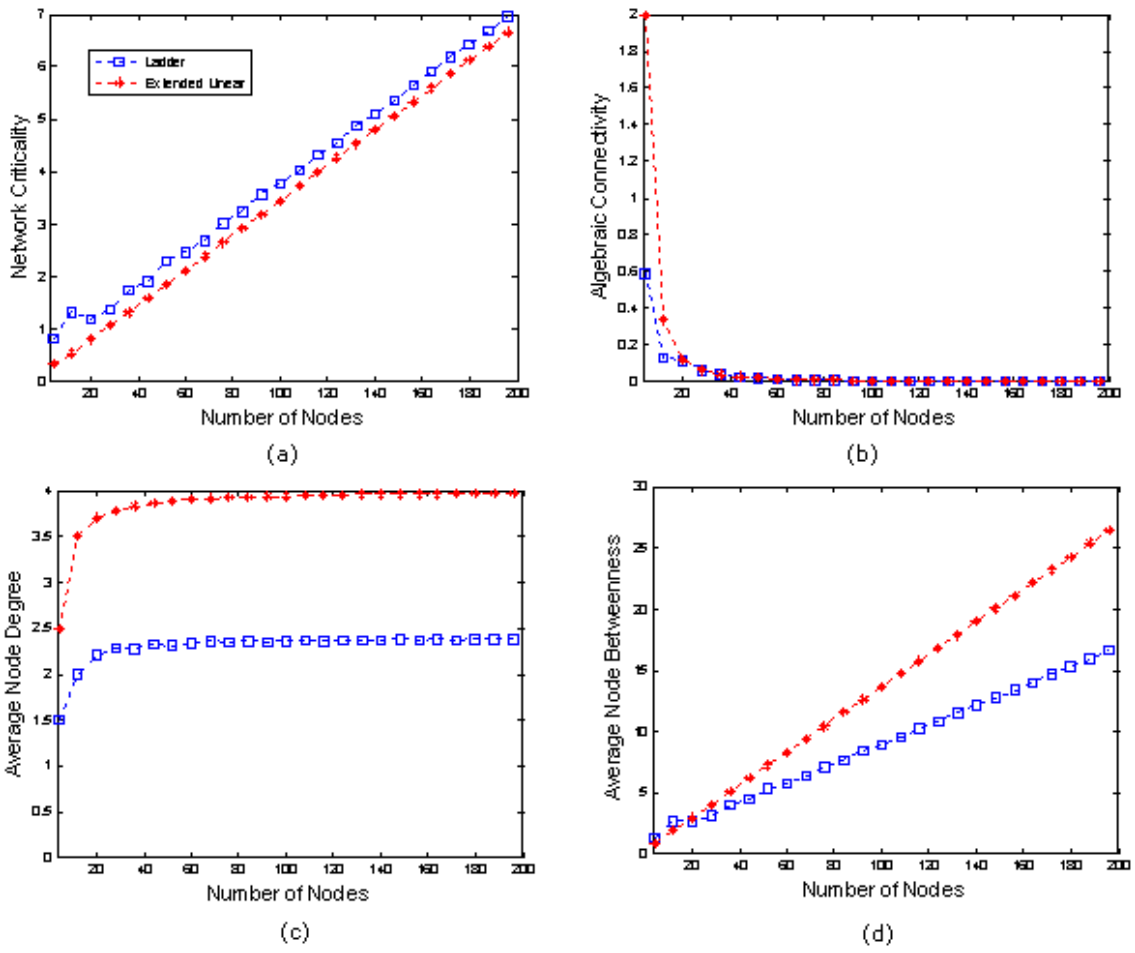
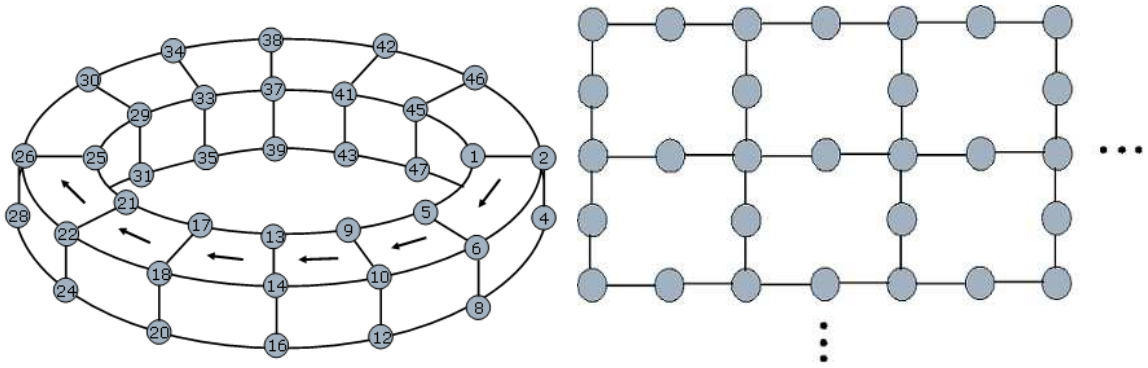


Figure 3.5: Comparison of Ladder (LAG) and Extended Linear Graph (ELG) Topologies

3.3.3 Square Torus Graph (STG) and Sparse Flat Graph (SFG)

Now we study the behavior of square torus graph (STG) and sparse flat graph (SFG) (Fig. 3.6). In Fig. 3.7 the behavior of STG and FGG are compared. Fig. 3.7-a shows that up to a certain number of nodes (around 16 nodes), the network criticality of STG decreases with increasing node number. It also reveals that for networks with less than 90 nodes STG has smaller network criticality than FGG with the same



(a) STG (b) SFG

Figure 3.6: Topology of Square Torus Graph (STG) and Sparse Flat Graph (SFG)

number of nodes. However, for larger networks FGG has better (smaller) network criticality. Fig. 3.7-b shows that algebraic connectivity of STG for networks with less than 80 nodes is better (larger) than that of FGG with the same number of nodes. It also shows that after 80 nodes algebraic connectivity of FGG is slightly better. However, the difference in the behavior of STG and FGG is more evident through their network criticality and average node betweenness.

Fig. 3.7-c shows that average node degree of STG quickly approaches 4, while average node degree of FGG is always less that 4 and goes to 4 very slowly. According to the Fig. 3.7-d, this causes smaller average node betweenness for FGG networks when $80 \leq n \leq 90$, while the network criticality of STG is smaller for the networks with $80 \leq n \leq 90$ as shown in Fig. 3.7-a.

Finally, we study the robustness behavior of SFG and FGG. In Fig. 3.8 we compare the behavior of SFG with FGG. Fig. 3.8-c shows that for networks with large number of nodes, FGG has 45% more links than SFG. This increase in link density leads to a considerable improvement in network criticality, algebraic connectivity, and average node betweenness. Fig. 3.8-a shows that network criticality of FGG is much

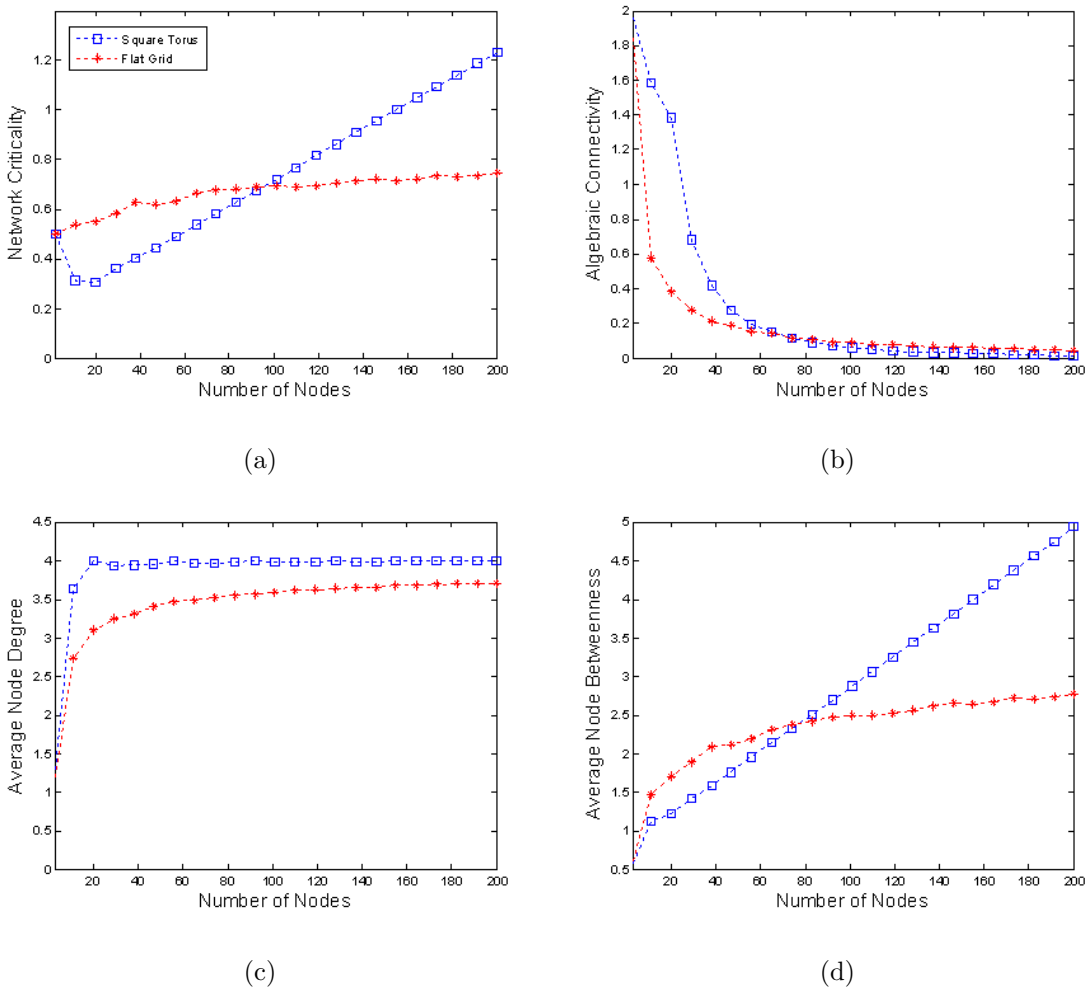


Figure 3.7: Behavior of STG and FGG

smaller than SFG. For $n = 200$, the network criticality of SFG is 1.406 while it is 0.747 for FGG.

There are also some interesting points in the comparison of behavior of FGG, DFG and SFG using Fig. 3.8 and Fig. 3.4. These figures show that moving from SFG to FGG generates more improvement in network metrics compared to moving from FGG to DFG, although in both movements we should increase number of links to the same extent.

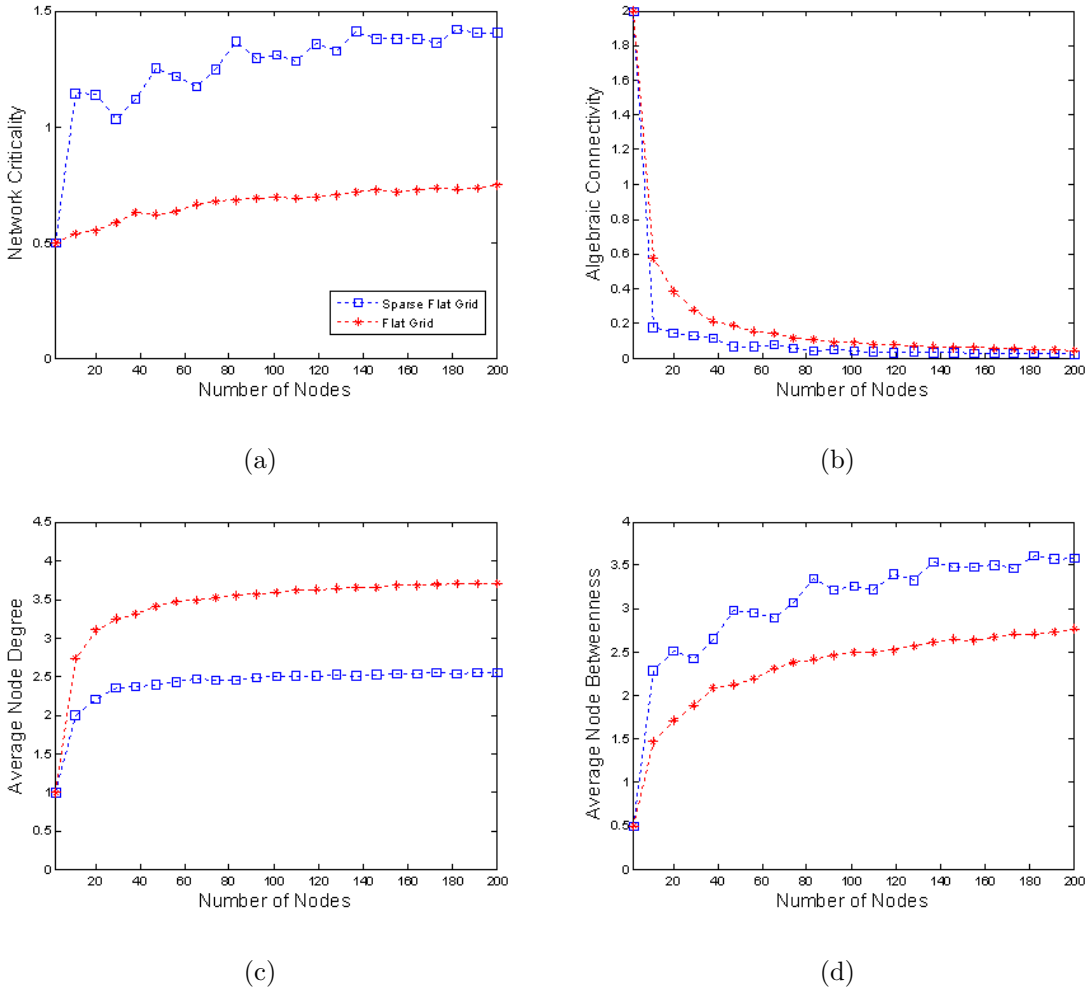


Figure 3.8: Behavior of SFG and FGG

Quantity	SFG	FGG	DFG
Average Degree	2.55	3.71	5.43
Network Criticality	1.406	0.747	0.494

Table 3.1: Numerical Comparison of Robustness in SFG, FGG, and DFG

Table 3.1 shows the average node degree and network criticality for these three topologies when n = 200. According to this table moving from SFG to FGG needs 45% increase in average degree and leads to 47% decrease in network criticality while moving from FGG to DFG needs an increase in average degree by 45% again but network criticality decreases only 34%. In addition, comparison of Fig. 3.8-b with Fig. 3.4-b and Fig. 3.8-d with Fig. 3.4-d show that by moving from SFG to FGG we gain improvement in algebraic connectivity and average node betweenness, while there is almost no gain in these metrics by moving from FGG to DFG.

3.4 Complex Networks

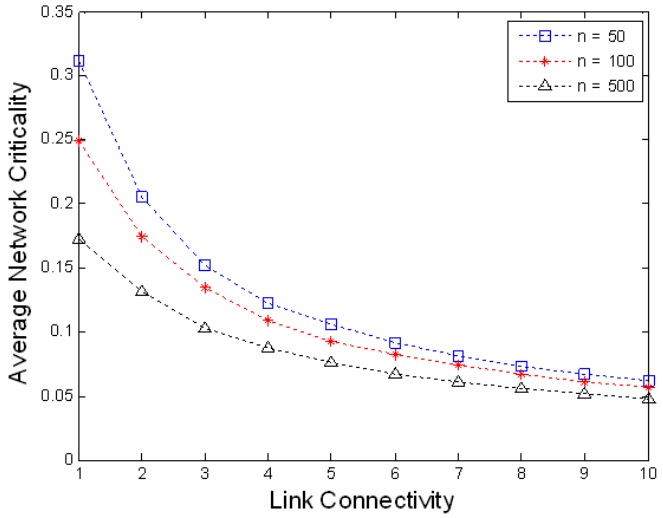
In this section, we investigate the robustness properties of three main categories of complex networks. We examine the behavior of network criticality for each class of random, small-world, and scale-free networks when node/link connectivity changes, and compare it to that of algebraic connectivity.

3.4.1 ER Random Graphs

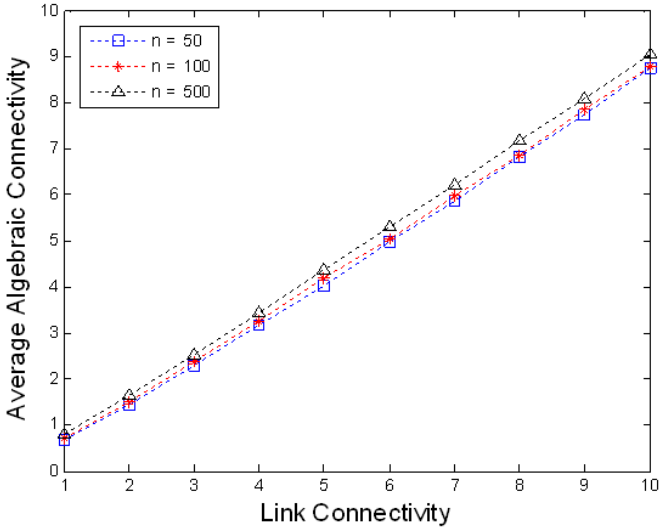
The simplest model for complex networks was first introduced by Errdos-Renyi (ER)[69]. The ER random graph has a simple structure in which each possible link of the network exists with a probability of p. If n is the number of nodes and m is the number of links for such a network then

$$E(m) = p\frac{n(n-1)}{2}$$

where E means the expected value. There is a threshold $p_t = \frac{log(n)}{n}$ for a random graph that if $p > p_t$ then the graph will be almost surely connected [69].



(a)



(b)

Figure 3.9: Average Network Criticality and Algebraic Connectivity for ER Random Networks

We generated 10,000 samples of random graphs for each number of nodes for $n = 50, 100, 500$. We set $p = \alpha p_t$ and selected α as a uniformly random integer $1 \leq \alpha \leq 10$. Then we obtained the link connectivity, algebraic connectivity, and network criticality for each one of the generated samples and calculated the average network criticality and algebraic connectivity for the graphs with the same link connectivity and number of nodes. Fig. 3.9 shows the variations of average network criticality and average algebraic connectivity with link connectivity for different values of n.

Fig. 3.9-a shows that average network criticality decreases with increasing link connectivity (remember that the smaller network criticality is better) and this justifies the usage of network criticality as a robustness metric since networks with higher link connectivity are more robust to link failures and the failure of more links makes them disconnected. Fig. 3.9-b shows the similar result for average algebraic connectivity. This figure shows that average algebraic connectivity increases almost linearly with link connectivity. Comparison of Fig. 3.9-a and 3.9-b reveals that variations of average network criticality and algebraic connectivity with link connectivity are in harmony for ER random graphs in predicting the network robustness. However, as we will see in the next parts, it is not the same for small-world and scale-free networks that are more realistic models for complex networks and the Internet.

3.4.2 Small-World Networks

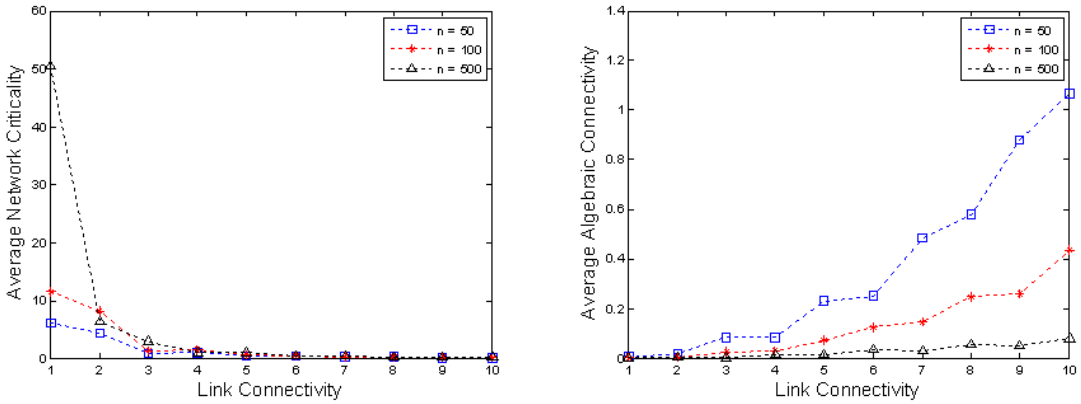
Small-world is a name for a subset of complex networks where despite the huge size of the network, the average path length between any two nodes is relatively small. The small-world model proposed by Watts-Strogatz is the most studied small-world model [71]. Starting from a regular ring lattice, we connect each node with 2s neighbors (s neighbors in each side), then we rewire each link with probability

p. In rewiring process, we keep the first node of each link and reconnect the other end node (clockwise) with probability of p to another node which is chosen randomly from the ring nodes in a way that self-loops and parallel edges are not permitted. For $0.01 \leq p \leq 0.1$ small world characteristic appears (small average path length, large clustering coefficient). If we continue increasing the randomness by increasing p, the graph starts behaving more like a random graph. For $p = 1$ we get a pure random graph (small average path length, small clustering coefficient).

In Fig. 3.10 variations of average network criticality and average algebraic connectivity with link connectivity for different values of n are shown. Fig. 3.10-a and 3.10-b show the behavior of average network criticality and algebraic connectivity for $p = 0.01$, and Figures 3.10-c and 3.10-d correspond to $p = 0.1$. For each combination of n and p, we have generated $10,000$ graphs. The value of s is taken to be uniformly distributed between 1 to 10 units ($1 \leq s \leq 10$). The results shown in Fig. 3.10 are the average values over all generated graphs.

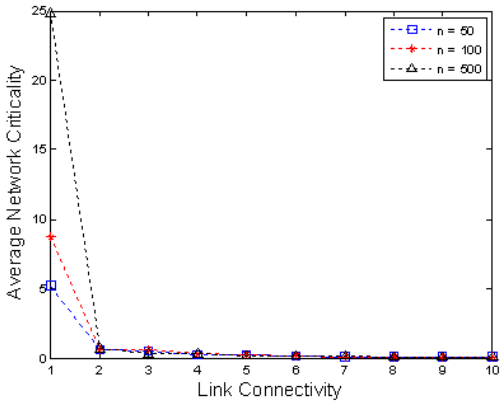
One can see that average algebraic connectivity in both cases ($p = 0.01$ and $p = 0.1$) and for different values of n increases relatively smoothly with link connectivity. However, Fig. 3.10-a and 3.10-c show a significant change in average network criticality when link connectivity increases from 1 to 2. Therefore, if simplifying a small-world network leads to decreasing edge connectivity from 2 to 1, this results in a huge jump (increase) in average network criticality (losing robustness) which is not desirable. This effect is not captured through average algebraic connectivity charts.

Furthermore, when the link connectivity is more than 4, the value of average network criticality does not have a significant change even if the network grows from $n = 50$ to $n = 500$. In other words, for small-world networks, if the connectivity is fixed to a desirable value (this value depends on the probability p), increasing the network size does not cause dramatic change in network criticality.

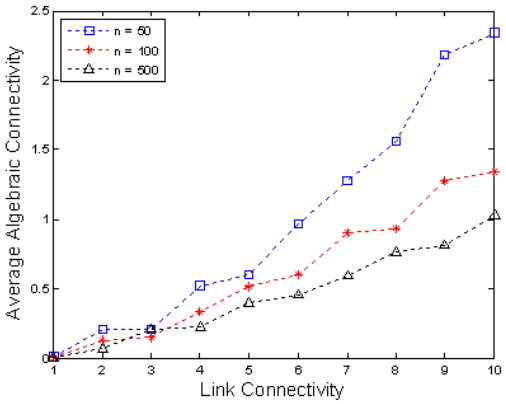


(a) $p = 0.01$

(b) $p = 0.01$



(c) $p = 0.1$

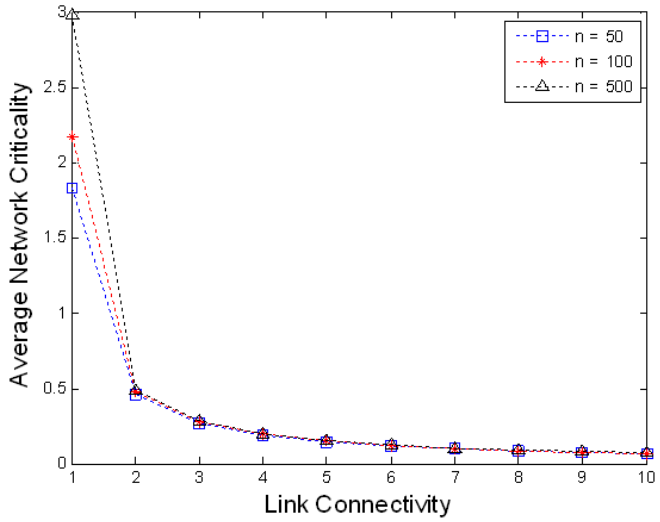


(d) $p = 0.1$

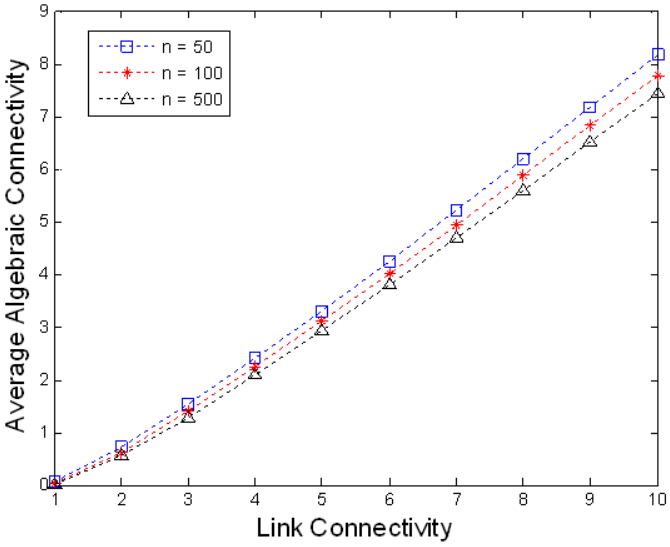
Figure 3.10: Average Network Criticality and Algebraic Connectivity for Small-World Networks with $p = 0.01$ and $p = 0.1$

3.4.3 Scale-Free Networks

Scale-free networks are networks for which node degree has a power-law distribution, while node degree distribution is binomial for ER random graphs. Scale-free networks represent many real-world networks including the Internet [70]. The Barbasi-Albert (BA) scale-free complex networks [72] are the most studied scale-free networks, and are generated through the process of preferential attachment. The degree distribution



(a)



(b)

Figure 3.11: Average Network Criticality and Algebraic Connectivity for Scale-Free Networks

for BA scale-free networks has the form of:

$$P(d = k) = \alpha k^{-\beta}$$

where d is the node degree and $\beta \approx 3$. Most of the nodes of these networks have small node degree while a few of them have large node degree (greater than 10).

Barbasi and Albert proposed the process of preferential attachment to produce scale-free networks, and we followed their method in our work. To generate a BA scale-free network with n nodes we started with a full-mesh of n_0 nodes ($n_0 < n$), and construct the network through $n - n_0$ steps. In each step, we add one new node to the network, and connect it to the previous nodes with k links ($k < n_0$) using preferential attachment. This means that the new node is connected to the previous nodes with a probability proportional to their degrees at that step. The number of links will be:

$$m = \frac{n_0(n_0 - 1)}{2} + k(n - n_0)$$

Depending on the initial number of nodes and k, the generated network will have different number of links. In our experiments, we set $n_0 = 2k+1$ which led to networks with $m = nk$ links. We generated 10,000 networks for each of $n = 50, 100, 500$ nodes, and selected k as a uniformly random integer $1 \leq k \leq 10$.

Fig. 3.11 shows the variations of average network criticality and algebraic connectivity for different values of link connectivity. We observe that average network criticality shows a huge drop when link connectivity increases from 1 to 2, while average algebraic connectivity increases almost linearly against link connectivity, and does not show this effect. We note that the behavior of average network criticality for scale-free networks resembles its behavior for small-world networks, while the behavior of average algebraic connectivity is similar to that of ER random graphs. In other words, Fig. 3.11-a shows that scale-free networks with link connectivity greater than

1 are much more robust, and the algebraic connectivity curve (Fig. 3.11-b) does not show this. In addition, similar to small-world networks, the value of average network criticality does not change significantly when the link connectivity is more than 4 even if the network size increases from $n = 50$ to $n = 500$. Therefore, for scale-free networks, if the connectivity is fixed to a desirable value (this value depends on n_0 and k), increasing the network size does not cause a dramatic change in network criticality.

3.5 Data Center Networks

There has recently been a great interest in data center design. A number of researchers in academia and industry have tried to propose new techniques and architectures for building and maintaining data centers [89, 90, 91]. The focus of all of the recent works on data centers is on virtualization techniques to provide a flexible management architecture on top of the existing physical topologies. Researchers have introduced methods for imitating layer 3 functionalities (routing) in data layer to accelerate the process and decrease the overhead.

In contrast, there is no serious study on existing physical data center topologies and their effectiveness in providing services for data center costumers. Today's data centers mostly use a variation of tree-based topologies with some modifications to increase reliability. For example, VL2 [89] uses a 3-level folded Clos network because it is well suited for a type of oblivious routing mechanism which is used in VL2 to provide load balancing. PortLand [90] also exploits a Fat-Tree to provide path multiplicity. In other words, in the present approaches to the design of data centers, the physical topology is selected from a list of common existing architectures based on some important goals of the design, but there is no explicit design methodology

for the physical topology.

In this section we are concerned about the physical topology of data centers. We investigate a few popular topologies that have been recently used in data centers and compare their topological robustness with the help of some ideas from networks. A data center topology consists of two major parts, containers, and a core network interconnecting different containers. Using the idea of network criticality or effective resistance from graph theory, we will investigate alternative combinations of container-core topologies (for example Tree topology for containers and Hyper-Cube topology for the core network) and study the topological robustness of different combinations. We present an optimization to minimize network criticality and use it as a planning method to allocate optimal capacity to the links of a specific topology. We solve the optimization problem for Fat-Tree, Hyper-Cube, Clos, and Star configurations and compare the optimal network criticality of these topologies when initial conditions are similar.

3.6 Data Centers Topologies

In this section we present a short review of three recent architectures for data centers which have absorbed attention in academia and industry.

3.6.1 Virtual Layer 2 (VL2)

Today's data centers with tens of thousands of servers must achieve the property of agility to assign any server to any service. Existing architectures do not provide enough bisectional throughputs between the servers. Furthermore, services are not isolated from each other's impact. In addition, fragmentation of the address space limits the migration of virtual machines.

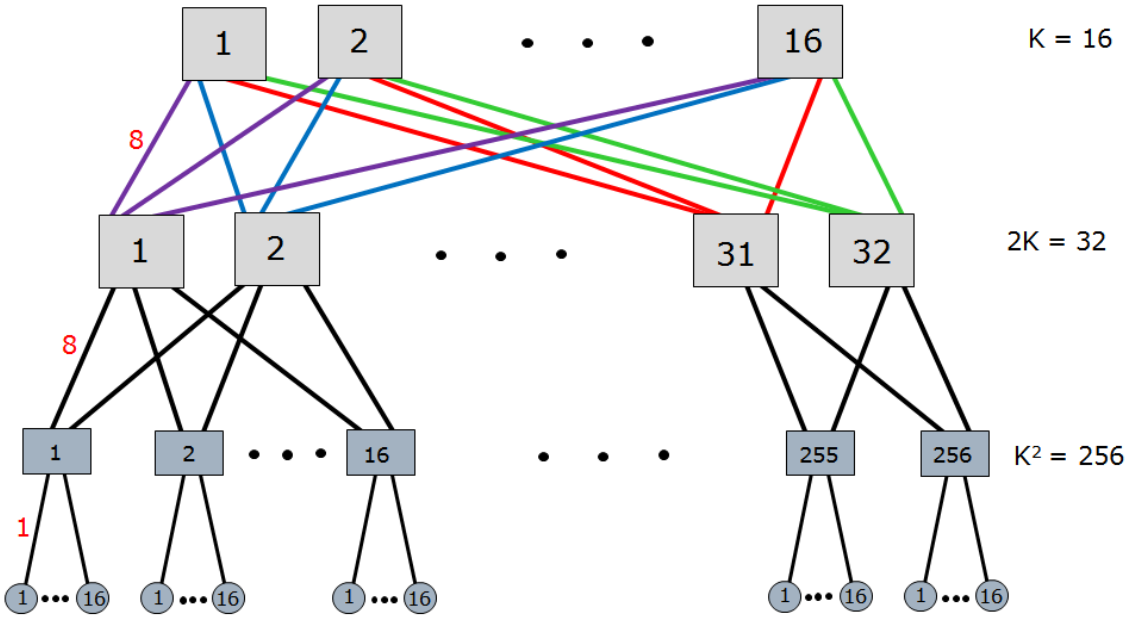


Figure 3.12: Clos Network Used in VL2

One possible solution is to give each service the illusion that all the servers assigned to it are connected by a single Virtual Layer 2 (VL2) non-interfering Ethernet switch [89]. VL2 is built from low-cost switches arranged into a Clos topology [92] and Valiant Load Balancing [17] spreads traffic across all available paths; VL2 uses Location Address and Application-specific Address to separate the host identity and its topology location.

Up to recent years, most of the existing data center topologies were based on Cisco's conventional hierarchical topologies [93]. A major drawback of these topologies is that they have poor bisection bandwidth and are susceptible to major disruptions due to device failures at the highest layers. Instead, VL2 uses a folded Clos architecture with three layers (Intermediate or Core, Aggregation, and Edge), where the links between the Intermediate and Aggregation switches form a complete bipartite graph. Top-of-the -Rack switches connect to two Aggregation switches, and in

general if there is k Intermediate switch at the highest level, there will be $2k$ aggregation switches and k^2 Top of Rack (ToR) switches in the Clos as shown in Fig. 3.12. In this section we focus on the physical topology of VL2 architecture and investigate the robustness properties of Clos networks.

3.6.2 PortLand

Another recent proposal for data center architecture is PortLand [90]. In [90], the authors propose a scalable layer 2 protocol designed for data center environments. The idea is quite simple: it leverages specific knowledge of the baseline topology and growth model of data center networks to assign pseudo MAC addresses to servers based on their location in the topology. The pseudo addresses internally encode topological information of the server and this makes it simple to route traffic in the data center fabric. In addition, the size of forwarding tables at each switch is smaller and scales better than traditional solutions. The big underlying assumption is that data center networks are built as Fat-Tree topologies. Similar to the Clos network topology, Fat-Tree is also divided into three layers: Edge, Aggregation and Intermediate or Core. This hierarchy is chosen as each switch relies upon it to discover its own location in the network: a location discovery protocol is introduced for that purpose and it is shown to support growing the network in a plug-and-play fashion. Edge and aggregation switches are connected with some level of redundancy and are grouped into pods. Each pod is then connected to all of the core switches. A typical topology for PortLand data center (based on Fat-Tree) is shown in Fig. 3.13. The Fat-Tree as a whole is divided into pods that use k-port switches, and each pod supports non-blocking operation among $\frac{k^2}{4}$ hosts.

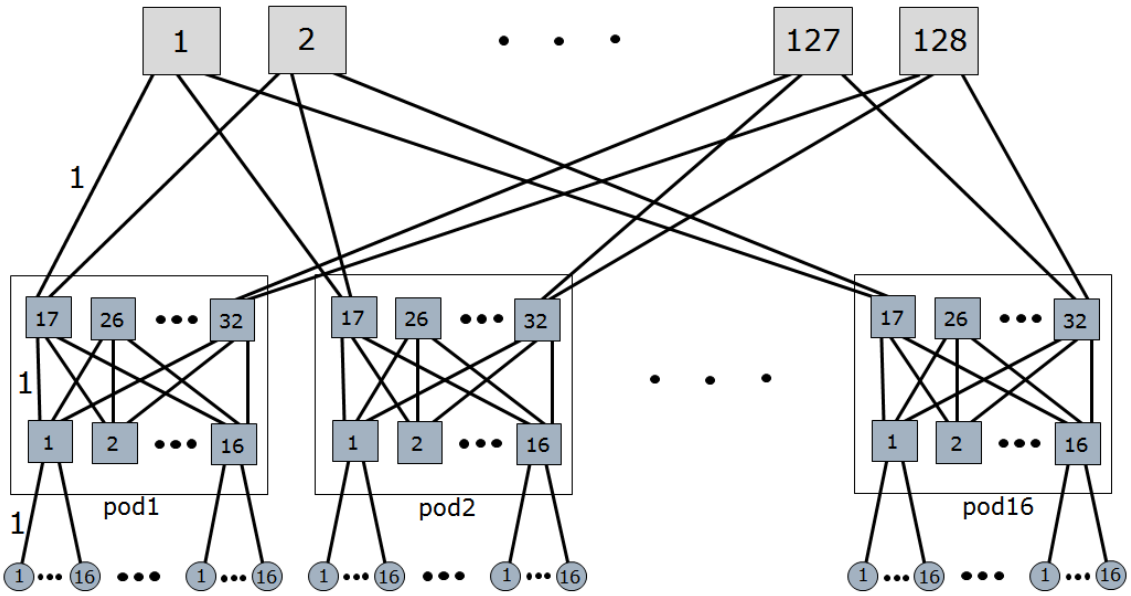
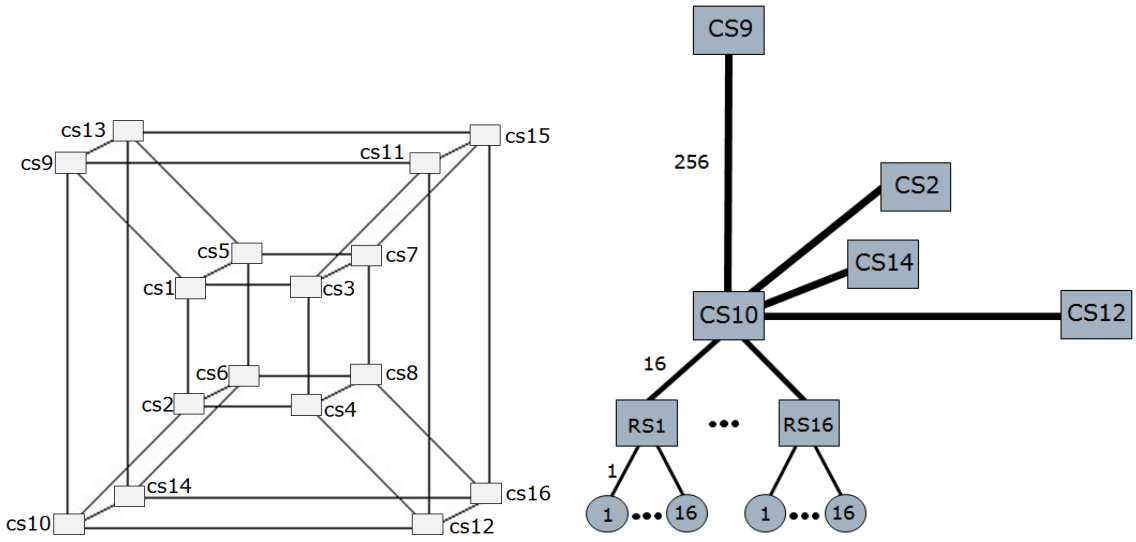


Figure 3.13: Fat-Tree Network used in PortLand

3.6.3 BCube

BCube [91], closely related to the generalized Hyper-Cube, is a high-performance and robust network architecture for a modular data center. Both oblivious and source routing are provided in the protocol suite. [91] presents a new network architecture specifically designed for shipping-container based, modular data centers. At the core of the BCube architecture is its server-centric network structure, where servers with multiple network ports connect to multiple layers of commodity on-the-shelf mini-switches. Servers act as not only end hosts, but also relay nodes for each other. BCube supports various bandwidth-intensive applications by speeding- up one-to-one, one-to-several, and one-to-all traffic patterns, and by providing high network capacity for all-to-all traffic. BCube exhibits graceful performance degradation as the server and/or switch failure rate increases. This property is of special importance for shipping-container data centers, since once the container is sealed and operational,



(a) Hyper-Cube Core (b) Hyper-Cube Node+Container

Figure 3.14: Hyper-Cube Topology

it becomes very difficult to repair or replace its components.

As stated in [91], BCube is a special case of generalized Hyper-Cube. In this work, we are interested in a general Hyper-Cube core with container as tree topologies connected to the nodes of Hyper-Cube core. Fig. 3.14-(a) shows the architecture of a Hyper-Cube on 16 nodes. Each node of the Hyper-Cube is connected to a container with Star architecture as shown in Fig 3.14-(b).

3.7 Optimization of Network Criticality for Data Center Networks

The ultimate goal is to find a method to minimize network criticality. Hence, we consider the minimization of τ under some constraints. We assume a fixed total capacity (or link weights in a more general sense) for the links of the network. In

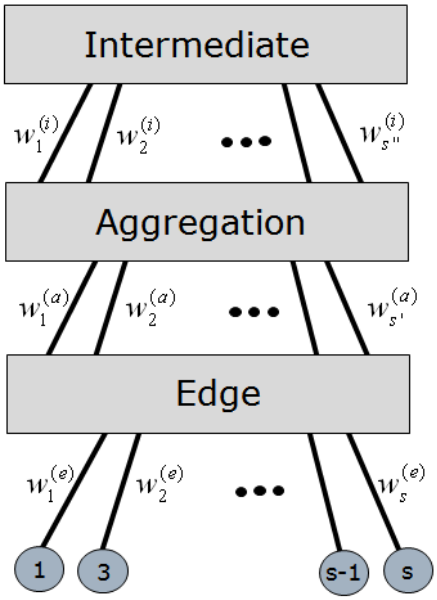


Figure 3.15: A General 3-Layer Form of Data-Center with Weights

order to have a non-blocking service from any server in the data center topology to the other servers, we add another constraint to keep the total bisection capacity fixed in all three layers of our data center topology.

More specifically, consider Fig. 3.15 as the general form of the data center with three major layers, Edge, Aggregation, and Intermediate or Core layers. The rationale behind this model is that in general large scale data centers are organized as a hierarchy of racks and containers where a rack contains a certain number of servers, a container is composed of a number of racks, and data center is made of several containers in turn. See [94] for more details of this structure. Therefore the switches in the Edge layer are switches in (or top of) the racks that are connected to the servers directly, the switches in the Aggregation layer are switches that are used to connect Edge switches inside a container, and the Intermediate or Core switches are used to connect the containers to each other. It can be easily seen that this model conforms

with three switch layers used in VL2 and Portland as well.

We define the following notation to specify the capacities (weights) of different layers. We denote the link capacities (weights) between Edge layer and lowest level with $w_p^{(e)}$, the link capacities (weights) between Aggregation layer and the lower layer with $w_q^{(a)}$, and the link capacities from Intermediate layer to the lower layer with $w_r^{(i)}$. We also denote the cost of having link weights $w_p^{(e)}$, $w_q^{(a)}$, and $w_r^{(i)}$ with $z_p^{(e)}$, $z_q^{(a)}$, and $z_r^{(i)}$ respectively. Then the the first constraint (fixed total weight or capacity) can be written as:

$$\sum_{p=1}^{s} z_p^{(e)} w_p^{(e)} + \sum_{q=1}^{s'} z_q^{(a)} w_q^{(a)} + \sum_{r=1}^{s"} z_r^{(i)} w_r^{(i)} = C$$

where C is the fixed total budget, and s, s', and s" are the number of links from edge, Aggregation, and Intermediate layers to the lower layers respectively. The second constraint regarding the bisection capacity between different layers can be written in the following form:

$$\sum_{p=1}^{s} w_p^{(e)} = \sum_{q=1}^{s'} w_q^{(a)} = \sum_{r=1}^{s"} w_r^{(i)}$$

The optimization problem is then:

$$Minimize \quad \hat{\tau} \tag{3.10}$$

$$\begin{aligned} Subj.\ to \qquad & \textstyle\sum_{p=1}^{s} z_p^{(e)} w_p^{(e)} + \sum_{q=1}^{s'} z_q^{(a)} w_q^{(a)} + \sum_{r=1}^{s"} z_r^{(i)} w_r^{(i)} = C \\ & \textstyle\sum_{p=1}^{s} w_p^{(e)} = \sum_{q=1}^{s'} w_q^{(a)} = \sum_{r=1}^{s"} w_r^{(i)} \\ & w_p^{(e)} \geq 0 \quad 1 \leq p \leq s \\ & w_q^{(a)} \geq 0 \quad 1 \leq q \leq s' \\ & w_r^{(i)} \geq 0 \quad 1 \leq r \leq s" \end{aligned}$$

Optimization problem 3.10 can be converted to a semidefinite programming problem. First we note that

$$\hat{\tau} = \frac{2}{n-1} Tr(L^+) = \frac{2}{n-1} Tr((L + J/n)^{-1}) - \frac{2}{n-1}$$

[4] where I is the identity matrix and J is a matrix with all of its elements equal to 1. Now we consider the matrix $\Theta = \begin{pmatrix} \Gamma & I \\ I & L + \frac{J}{n} \end{pmatrix}$ where Γ is an $n \times n$ variable matrix. The necessary and sufficient condition for positive-semidefiniteness of Θ is that its Schur complement [28] be positive- semidefinite. The Schur complement of Θ is $\Gamma - (L + J/n)^{-1}$ and its positive-semidefiniteness leads to $Tr(\Gamma) \geq Tr((L + J/n)^{-1})$. Considering this inequality, optimization problem 3.10 can be converted to the following semidefinite program:

$$\begin{aligned} & Minimize \quad \tfrac{2}{n-1} Tr(\Gamma) - \tfrac{2}{n-1} \\ Subj.\ to \quad & \textstyle\sum_{p=1}^{s} z_p^{(e)} w_p^{(e)} + \sum_{q=1}^{s'} z_q^{(a)} w_q^{(a)} + \sum_{r=1}^{s"} z_r^{(i)} w_r^{(i)} = C \\ & \textstyle\sum_{p=1}^{s} w_p^{(e)} = \sum_{q=1}^{s'} w_q^{(a)} = \sum_{r=1}^{s"} w_r^{(i)} \\ & \begin{pmatrix} \Gamma & I \\ I & L + \frac{J}{n} \end{pmatrix} \succeq 0 \end{aligned} \tag{3.11}$$

3.8 Evaluation of Data Center Networks

Study of recent data center architectures in section 3.6 reveals that Clos (Fig. 3.12), Fat-Tree (Fig. 3.13), and Hyper-Cube (Fig. 3.14) topologies are popular. We add Star topology as the fourth well-structured topology to this list and investigate the robustness properties of these four different topologies in this section. Fig. 3.16 shows Star topology for constructing a data center. All these topologies fall into the general 3-layer architecture of Fig. 3.15.

In this section we evaluate the behavior of network criticality as a measure of topological robustness in Clos, Fat-Tree, Hyper-Cube and Star. We consider two cases. First, we assume that total number of servers in all topologies are the same

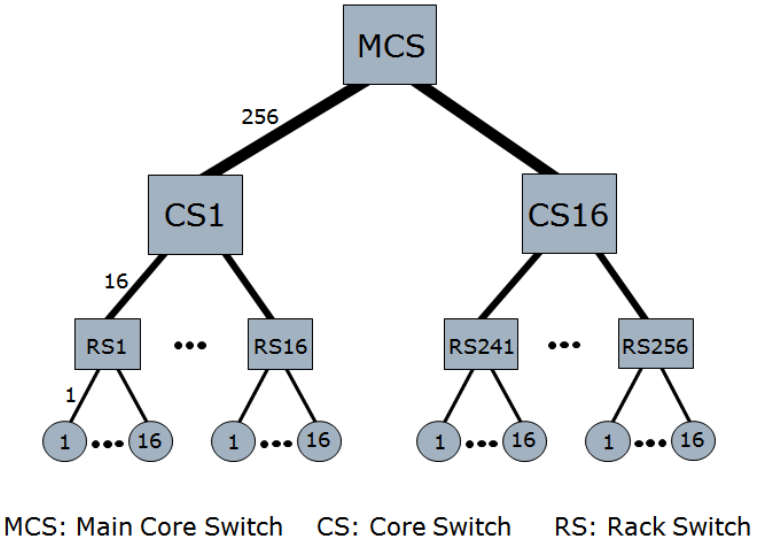


Figure 3.16: Star Network Topology for Data-Centers

and all of them have the same connection capacity to the edge switches. We also assume that we need to have maximum non-blocking traffic from each server to the other servers in all four topologies. Second, we add the constraint of having equal total capacity in all the networks, and find the optimal capacity allocation to minimize network criticality.

3.8.1 Performance Analysis of Data Center Architectures

Suppose we have 4096 servers with 1 GB (gigabytes) connection to the rack switches. Now we construct Aggregation and Intermediate layers in Clos, Fat-Tree, Hyper-Cube and Star in such a way to have non-block traffic from each server to the others. Table 3.2 shows the specification of all these network topologies when there exists a total number of 4096 servers in each one. The third, fourth, and fifth column of the table show the number of nodes (other than server nodes), links (other than leaf nodes), and total capacity in each topology respectively.

	Servers	Nodes	Links	Capacity
Star	4096	273	272	8192
Clos	4096	304	1024	8192
Fat-Tree	4096	640	8192	8192
Hyper-Cube	4096	272	288	12288

Table 3.2: Specifications of different Network Topologies

We observe in Table 3.2 that the number of nodes and links for the Fat-Tree topology is much higher than other topologies since it only uses 1GB links all over the system. The Clos network that uses aggregate links (8GB links in our example) in addition to 1GB links has lower number of nodes and links compared to Fat-Tree but it has remarkably larger number of links compared to Star and Hyper-Cube. The other point is that the total capacity of Hyper-Cube is higher than other topologies. The reason is that for 4096 servers, there should be 16 switches in the core Hyper-Cube, and each one should be connected to 4 other switches with 256GB links in order to maintain the constant bisectional capacity property (see Fig. 3.14-b). This increases the total capacity used in this topology compared to other ones.

Fig. 3.17 shows the comparison of network criticality for different data center topologies. The number of servers was 8, 64, 512, or 4096 in each experiment. We observe in Fig. 3.17 that Fat-Tree topology is the most robust topology (the lowest criticality) while the Star topology is the least robust one, and Clos and Hyper-Cube are in the middle. In fact, Hyper-Cube topology makes the Star more robust by replacing the central node with a Hyper-Cube, and eliminates the problem of single point of failure which is the weak point of the Star topology. Clos on the other hand

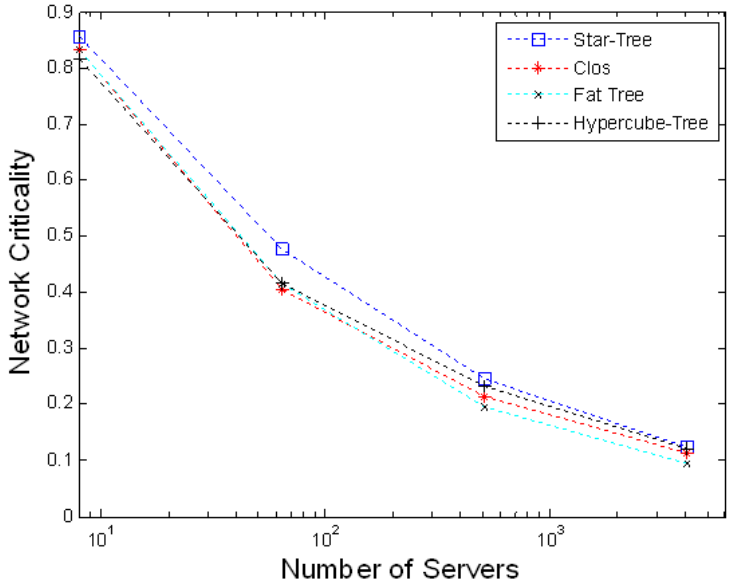


Figure 3.17: Network Criticality for Different Topologies

sacrifices the robustness of Fat-Tree to some extent but lowers the number of the network elements, and makes the network easier to manage.

3.8.2 Capacity Planning for Fixed Total Capacity

In this part, we applied the condition of the the fixed total capacity to our optimizations in addition to the constant bisectional capacity which is needed for the non-blocking routing capacity, and compared the robustness of the resulting networks. Fig. 3.18 shows the comparison of network criticality for various data center networks for 8, 64, 512, and 4096 servers. What we observe here is that similar to the last experiment, Fat-Tree has the lowest network criticality and is the most robust topology. However, the Hyper-Cube for high number of servers (512 and 4096) has become the least robust. The reason is that in previous experiment the total capacity of Hyper-Cube was higher than the other topologies (see Table 3.2) but the condition

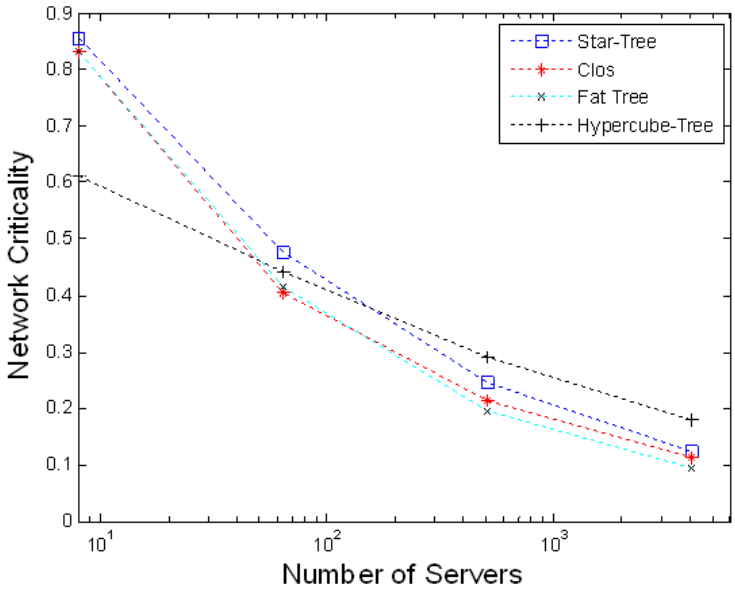


Figure 3.18: Optimal Network Criticality Subject to Having Constant Bisection Capacity and Fixed Total Capacity

of the constant total sum of capacities brings down the capacity of core links and increases the network criticality of the Hyper-Cube network.

Chapter 4

Robust Survivable Routing

The main elements of the autonomic management system under consideration in this thesis are in the form of control loops some of which are fast in nature, while others are slow. In this chapter we present our Robust Survivable Routing (RSR) algorithm that implements a fast control loop for online routing of the incoming demands with the orientation of fulfilling the self-optimizing and self-healing attributes of our autonomic management system.

4.1 Robust Survivable Routing (RSR)

Shared backup protection permits bandwidth sharing among backup paths to save resources while guaranteeing full failure recovery. Routing for shared protection involves identifying working and backup paths that optimize the total bandwidth consumption. The shared backup path protection (SBPP)[9][95][60] problem can be divided in two parts. The first part deals with assigning appropriate restoration bandwidth on each link to guarantee 100% single-failure recovery, and the second part is to choose the best backup path for any active (working) path to minimize the total cost. To

address the first problem, we consider three variables for each link, the working bandwidth, the backup or restoration bandwidth, and the available bandwidth of the link. When these variables are changing we check the new network condition and apply appropriate changes to continue guaranteeing 100% failure recovery. To solve the second problem we use the Weighted Random-Walk Path Criticality Routing algorithm (WRW-PCR) which is proposed in [59].

WRW-PCR defines link cost to reflect its sensitivity to the changes in traffic demand, network topology, and active source-destination pairs. We assess the criticality of each link and attempt to choose the path with minimum overall criticality. This method is examined for networks with bandwidth-guaranteed connections such as MPLS [39] or optical networks with full wavelength conversion capability.

4.1.1 WRW-PCR

Weighted Random-Walk Path Criticality Routing algorithm (WRW-PCR) is a routing algorithm that tries to minimize network criticality to maximize the network robustness. Minimization of network criticality has been investigated in detail in [88, 59] and references therein. Here, we provide a very brief introduction. Assume that there is a per-unit cost z_{ij} for link (i, j) with weight w_{ij}, and assume we have a total budget of C for all the links. One general optimization problem is then:

$$\begin{aligned} & Minimize \quad \hat{\tau} \\ Subject\ to \quad & \textstyle\sum_{(i,j)\in E} w_{ij} z_{ij} \leq C \quad , C\ is\ fixed \\ & w_{ij} \geq 0 \end{aligned} \tag{4.1}$$

Let w_{ij}^* be the optimal weight of link (i, j) and $\hat{\tau}^*$ be the minimum value of $\hat{\tau}$, then for any sub-optimal solution of the convex optimization problem $(\hat{\tau})$, the deviation

from the optimal solution (optimality gap) in optimization problem (4.1) has the upper bound:

$$\frac{\hat{\tau} - \hat{\tau}^*}{\hat{\tau}} \leq 1 + \frac{\hat{\tau}}{C \min_{(i,j)\in E} \frac{1}{z_{ij}} \frac{\partial \hat{\tau}}{\partial w_{ij}}} \tag{4.2}$$

In [59] a path selection method is designed using the upper bound for the optimality gap in inequality (4.2). Without loss of generality, we assume that $z_{ij} = 1$ for all links, then the upper bound becomes

$$1 + \frac{\hat{\tau}/C}{\min_{(i,j)\in E} \frac{\partial \hat{\tau}}{\partial w_{ij}}} = 1 - \frac{\hat{\tau}/C}{\max_{(i,j)\in E} |\frac{\partial \hat{\tau}}{\partial w_{ij}}|}$$

Here we have used the fact that network criticality is a decreasing function of the link weight. Therefore, the upper bound is determined by the link with the highest sensitivity of $\hat{\tau}$ (i.e. the derivative of $\hat{\tau}$) with respect to weight . This suggests that the path selection algorithm should avoid links with high sensitivity. WRW-PCR is based on this fact, and it selects the shortest path (using Dijkstra or other shortest path algorithm) where the link metric is given by

$$1 - \frac{\hat{\tau}/C}{|\frac{\partial \hat{\tau}}{\partial w_{ij}}|}$$

The length of a path is then:

$$\sum (1 - \frac{\hat{\tau}/C}{|\frac{\partial \hat{\tau}}{\partial w_{ij}}|}) = h - (\hat{\tau}/C) \sum \frac{1}{|\frac{\partial \hat{\tau}}{\partial w_{ij}}|}$$

where the sum is over all the links in the path and h is the number of hops in the path. We note that this path selection algorithm can be viewed as a modified minimum hop algorithm using a correction term based on link sensitivities.

4.2 RSR Formulation

Consider a network of nodes and links that specifies a graph $G(V, E)$. All links are directed. We characterize a request for a path by a triple (s_k, d_k, b_k). For request k, s_k specifies the ingress node, d_k specifies the egress node, and b_k is the amount of bandwidth required (or unit bandwidth per wavelength for wavelength routing with identical wavelength capacities). For each request, both active path and backup path needs to be determined. By a path we mean a sequence of the network nodes that starts with s_k, ends with d_k, there is a link from each node to the next node in the sequence, and no node appears more than once. Even though it may not be necessary to protect every path, in our work, we only consider demands that require protection by pre-established backup paths. Since all paths are to be protected, the active path and the backup path cannot share a common link for any path. If we want to protect paths against node failures then the active and backup paths should not share a common node.

4.2.1 RSR Notations and Assumptions

Backup path selection can be considered in different contexts and based on different assumptions. The first factor is the level of protection an algorithm provides. This can be protection against link and/or node failure. In this regard, our algorithms provide protection against single link failure, based on the assumption that nodes are stable enough or are protected against failures through redundancies. Thus we make the following assumptions:

- A request is blocked if one cannot find either active or backup path.
- Every active path must be completely protected against single link failure.

- If a link disjoint path cannot be found, the request is blocked.

The other factor is the level of information that path selection can be based on. In this context, we propose two kinds of algorithms: backup path selection with Full Sharing (FS) of the information of link backup capacities, and backup path selection with No Sharing (NS) of this information. In FS, the information of backup capacity which is used on a link due to failure on all other links of the network is used in computing the backup paths for arriving demands, while in NS, available bandwidth of the links is the only information used for this purpose. In fact, no backup capacity is shared between various demands in NS.

Table 4.1: Notation

Notion	Description
C_{ij}	capacity of link (i, j)
W_{ij}	total bandwidth used in working paths on link (i, j)
R_{ij}	total bandwidth used for backup(reserve) on link (i, j)
F_{ij}	Free capacity on link (i, j)
A_k	active path for demand k
B_k	backup path for demand k
$S(ij, pr)$	Backup capacity used on link (i, j) due to failure of link (p, r)

The notation we use in our algorithms is shown in TABLE 4.1. Clearly, $F_{ij} = C_{ij} - W_{ij} - R_{ij}$ in both algorithms, and $S(ij, pr)$ is used to prevent reserving repetitive capacity on each link related to simultaneous multiple link failures in the FS algorithm. Since we have considered one link failure at a time, we have:

$$R_{ij} = max_{pr}(S(ij, pr)) \quad over\ every\ link\ (p, r) \in\ E \tag{4.3}$$

The reason for this equation is that each link should have backup capacity for the worst case in which the most backup capacity is needed on it.

4.3 RSR Algorithms

In this section, we propose algorithms for FS and NS. In both algorithms, we first compute the active path A_k using WRW-PCR algorithm, and then try to find a link disjoint path B_k for the backup. Finally, we update states of the links on both active and backup paths. The main difference between the two algorithms is in updating link states after finding backup and working paths. In NS, the available bandwidth of the links on both working and backup paths is decreased by b_k while in FS, values of $S(ij, pr)$ are used to combine the backup capacities used on the links of backup path.

4.3.1 NS Algorithm

The pseudo code for NS algorithm is as follows:

1- Wait for the next demand

2- Receive next demand specs (s_k, d_k, b_k)

3- Prune links with residual capacities less than b_k

4- Use WRW-PCR to choose the least critical path between (s_k, d_k)
as active path A_k

5- If no active path is found, block the demand; Go to step 1

6- If active path A_k ***is found, prune every link of it from*** G

7- Use WRW-PCR to find the least critical path between (s_k, d_k)
as backup path B_k

8- If no backup path is found, block the demand; go to step 1

9- If backup B_k ***is found, update link states for links on*** A_k
and B_k ***as follows***
- For every link (i,j) ***in*** A_k ***,*** $W_{ij} = W_{ij} + b_k$ ***;*** $F_{ij} = F_{ij} - b_k$
- For every link (i,j) ***in*** B_k ***,*** $R_{ij} = R_{ij} + b_k$ ***;*** $F_{ij} = F_{ij} - b_k$
10- Go to step 1

In step 9 of the algorithm, after finding working and backup paths, for links on the working path the link working capacity is increased by b_k, while for links on the backup path the link backup capacity is increased by b_k. Free capacity for links on both paths is decreased by b_k.

4.3.2 FS Algorithm

The pseudo code for FS algorithm is as follows:

1- Wait for the next demand
2- Receive next demand specs (s_k, d_k, b_k)
3- Prune Links with residual capacities less than b_k
4- Use WRW-PCR to choose the least critical path between (s_k, d_k)
as active path A_k
5- If no active path is found, block the demand; Go to step 1
6- If active path A_k ***is found, prune every link of it from*** G
7- Add links pruned in step 3
8- For every link (i,j) ***in*** $(E|A_k)$***, compute*** $H_{ij} = max_{pr}(S(ij,pr))$
over all (p,r) ***in*** A_k ***;*** H_{ij} ***is the maximum backup bandwidth***
used on link (i,j) ***due to a failure on*** A_k ***without counting***
the current demand
9- Prune every link (i,j) ***in*** $(E|A_k)$ ***for which*** $C_{ij} - H_{ij} - W_{ij} < b_k$

10- Use WRW-PCR to find the least critical path between (s_k, d_k)
as backup path B_k

11- If no backup path is found, block the demand; go to step 1

12- If backup B_k ***is found, adjust link parameters of links on***
A_k ***and*** B_k ***as follows***

- ***For every link*** (i,j) ***in*** A_k ***,*** $W_{ij} = W_{ij} + b_k$***;***
  $F_{ij} = F_{ij} - b_k$
- ***For every link*** (i,j) ***in*** B_k ***and every link*** (p,r) ***in*** A_k ***,***
  $S(ij,pr) = S(ij,pr) + b_k$
- ***For every link*** (i,j) ***in*** B_k ***update*** R_{ij} ***as***
  $R_{ij} = max_{pr}(S(ij,pr))$ ***over every*** (p,r) ***in*** E

13- Go to step 1

Note that in step 12, after finding working and backup paths, updating of the link parameters of working path links is done similarly to NS algorithm. However, for links on the backup path, first the values of $S(ij,pr)$ are updated so that links on the backup path have enough reserved capacity to support failure of any link on the working path and then their backup capacity is adjusted using equation (3).

4.4 RSR Evaluation

In order to assess the proposed backup method both in NS and FS algorithms, we developed several test scenarios. Our test network is shown in Fig. 4.1 which is borrowed from [96], and it consists of 22 nodes and 45 full-duplex links. We considered two different cases, the static case where each demand stays in the network forever (no departures), and the dynamic case with arrivals and departures based on the specified distributions. Demands in the static scenarios are chosen from a uniform

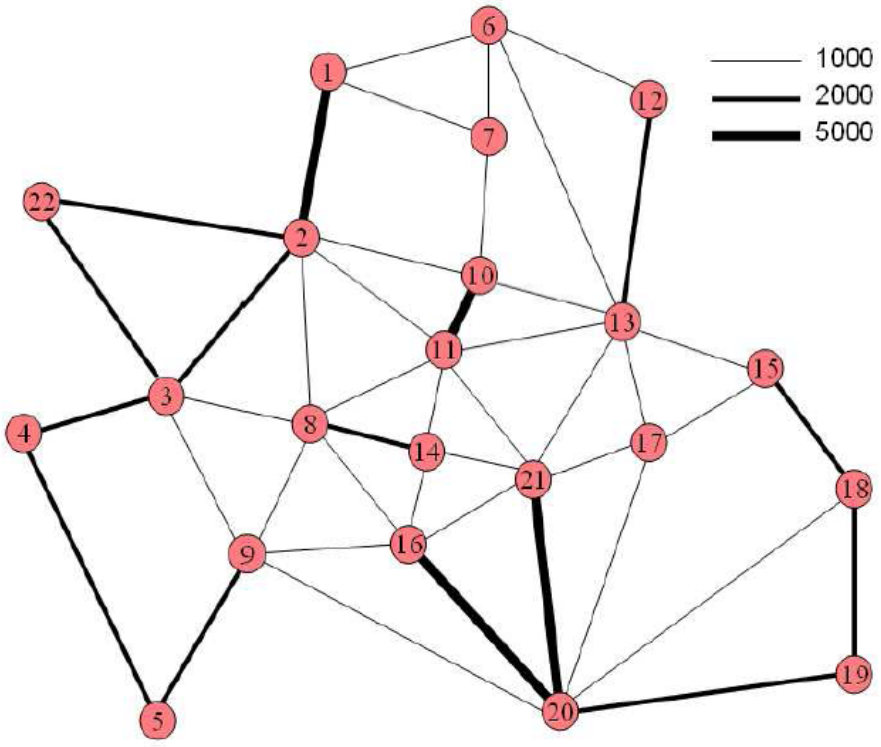


Figure 4.1: Test Network Topology

random number generator which selects integer numbers between 1 to 3 units. In the dynamic case the time of the demands is generated using a Poisson random number generator with rate λ, again from integer numbers between 1 to 3, and the service time is an exponential random variable with mean of μ where λ/μ is fixed in each of the scenarios to an appropriate number.

4.4.1 Static Traffic

In the first scenario, we consider the network of Fig. 4.1, and assume the bandwidth of the thin links is 1000 units while thick links have 2000 and 5000 units of bandwidth. The demands are chosen randomly among all possible source-destination pairs. In Fig. 4.2 we show the percentage of demands rejected using the NS method under

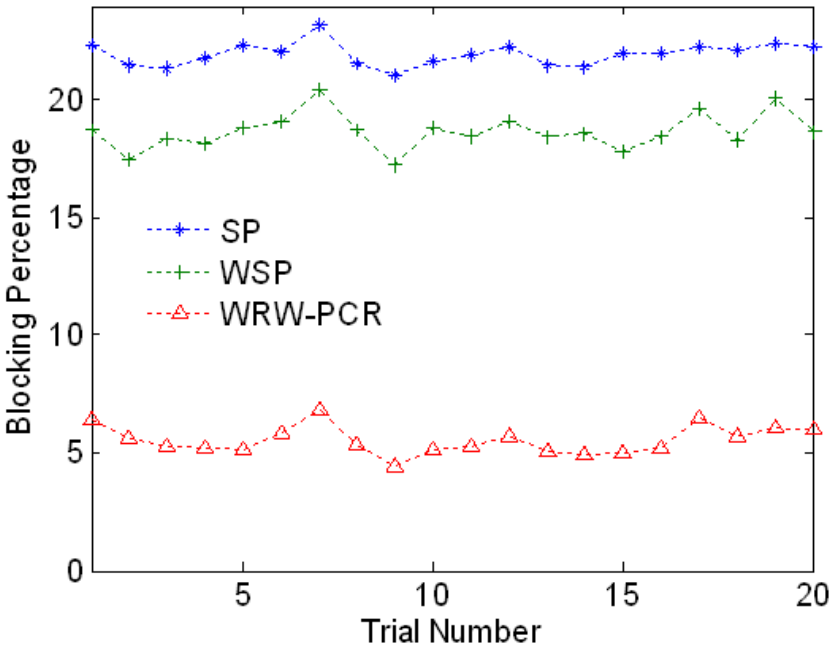


Figure 4.2: NS Static Scenario

three different path selection strategies; WRW-PCR, Shortest Path (SP), and Widest Shortest Path (WSP) [97]. SP is a well-known routing algorithm which is the base of OSPF and WSP is the improved version of it. In our implementation of SP and WSP, we first prune the links without enough available bandwidth to route the demand, and then compute min-hop path(s) between source and destination. In this experiment, we applied 9000 demands to the network each time and measured the blocking percentage in each trial.

Fig. 4.3 shows the result for the same scenario in the case of full sharing. Since in FS the network can accommodate more requests, 13000 requests are applied to the network in each trial. Again WRW-PCR outperforms SP and WSP considerably, while WSP has marginally better performance than SP.

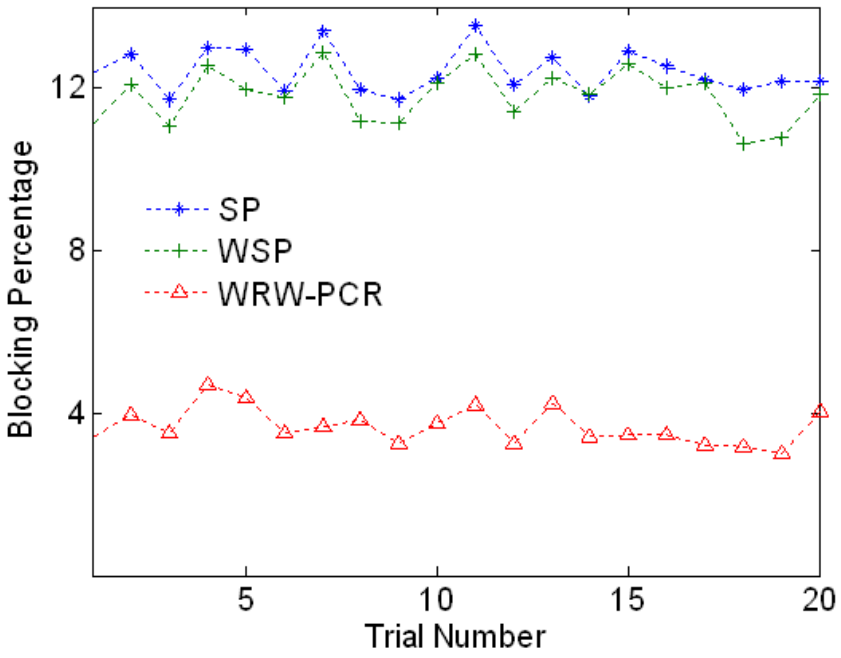


Figure 4.3: FS Static Scenario

4.4.2 Dynamic Traffic

We then conducted a dynamic scenario, where the incoming traffic has a finite lifetime. The traffic obeys the Poisson traffic with $\lambda/\mu = 900$ for the NS scenario and $\lambda/\mu = 1400$ for the FS scenario. We measured the blocking percentage for 10000 demands in each trial. In all the dynamic scenarios the link capacities are scaled down to 100, 200, and 500 units in order to reach the working point of the network more rapidly. The results are shown in Fig. 4.4 and Fig. 4.5. Once again the results indicate that WRW-PCR outperforms SP and WSP in both static and dynamic cases. The reason of WRW-PCR's superior performance is that it looks at more diversified paths to choose the primary and backup paths based on their criticality (PCI) while SP and WSP only look at the shortest path.

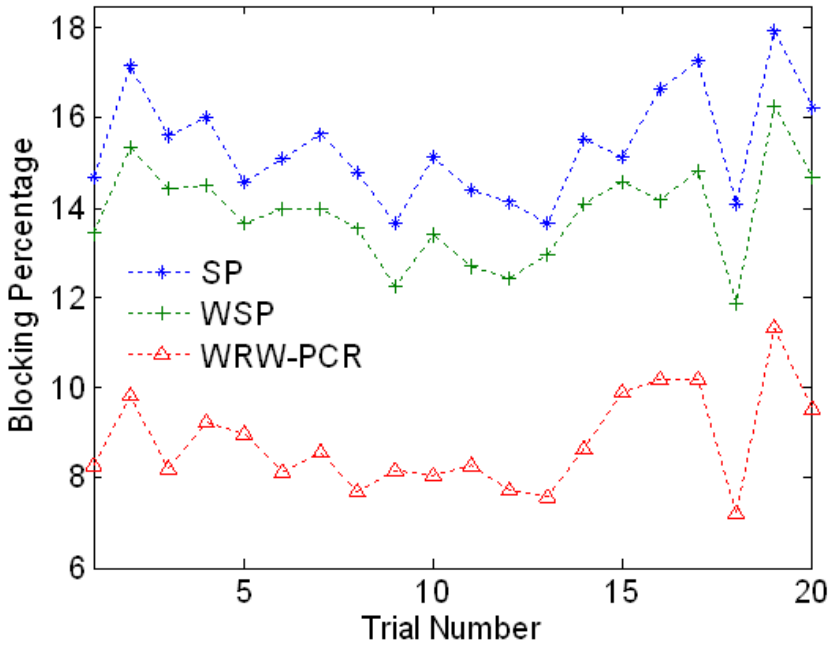


Figure 4.4: NS Dynamic Scenario

4.4.3 No Share vs. Full Share

Finally, we compare the NS and FS methods for backup selection in another scenario. Fig. 4.6 illustrates the comparison of no-share and full-share methods in a static scenario, where we load the network of Fig. 4.1 with 13000 bandwidth requests. As it is seen in this figure, for full-share method 96.5% of requests are accepted in average while the average acceptance rate for no-share method is 67.9% which means an improvement of 42.1 percent in FS compared to NS.

Fig. 4.7 summarizes the results for the second scenario when the traffic requests have finite life time ($\lambda/\mu = 1400$). Again in this case, we see similar results to static case; in FS method 94.4% of requests are accepted in average while the average acceptance rate for NS method is 71.1% which means an improvement of 32.7 percent in FS compared to NS.

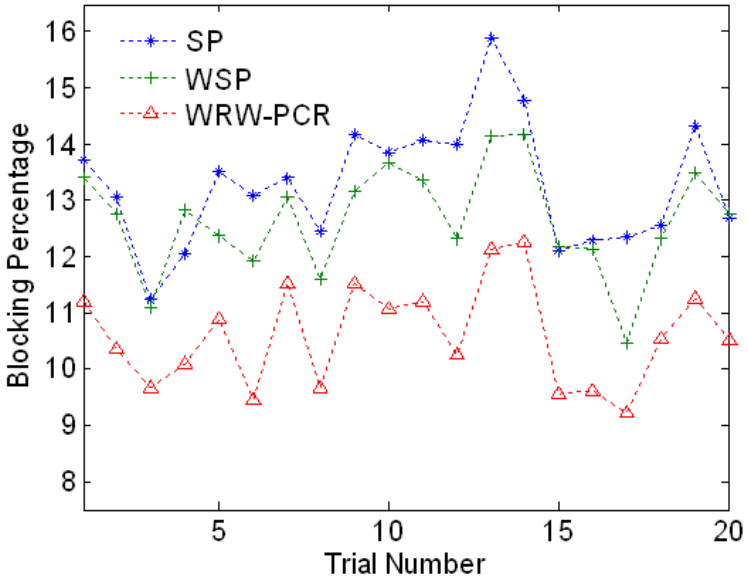


Figure 4.5: FS Dynamic Scenario

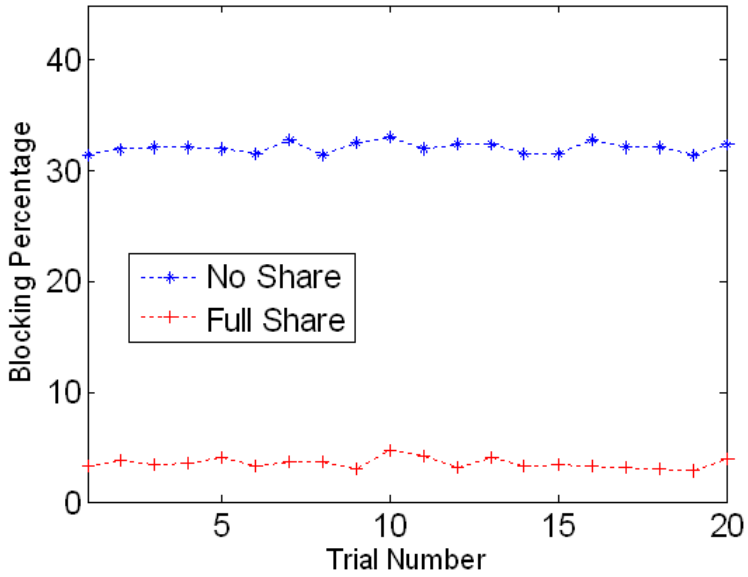


Figure 4.6: Comparison of NS with FS in Static Scenario

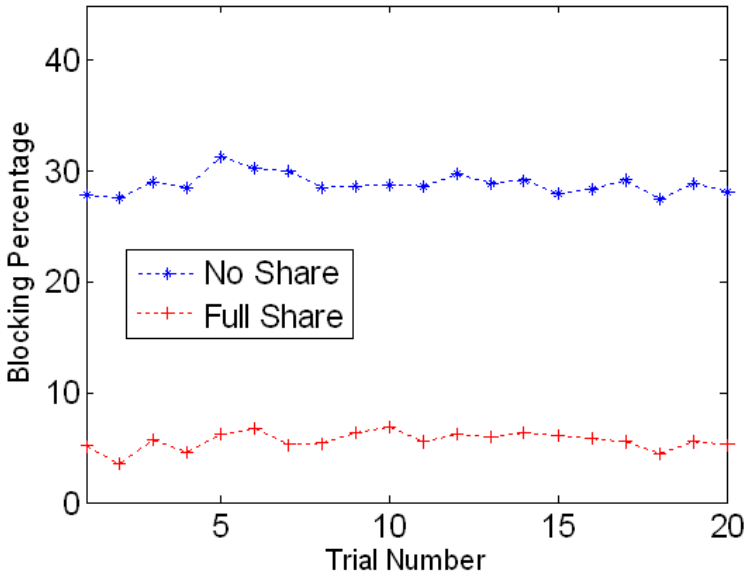


Figure 4.7: Comparison of NS with FS in Dynamic Scenario

Chapter 5

Network Design for Minimum Vulnerability

We introduced our autonomic management system as a combination of slow and fast control loops in chapter 2. As it can be seen in Fig. 2.3, one of the main functionalities of the slow control loop is network planning that means assigning weights to network links to have the most robust network. Robustness to the environmental changes is a key factor in planning a communication network and it is considered as the ability of the network to adapt itself to the environmental changes and continue to operate efficiently in the presence of these changes.

An important aspect of robustness in a communication network is its vulnerability to unwanted node/link failures or any malicious attack to nodes/links of the network. A network is less vulnerable to the failures or attacks if it can handle them in the best way. Vulnerability analysis studies the behavior of the network when a link/node failure discontinues the operation of a part of the network. The important feature of the network here is the ability of the network to operate efficiently and optimally in case of node/link failures. Therefore, the main focus of this chapter is robust

network design with emphasis on the robustness of the network when different failures happen. In order to quantify the vulnerability of a communication network to a failure, we consider the changes in the network parameters such as network criticality and algebraic connectivity due to a failure, as a measure of network vulnerability to that failure.

The methodology followed in this chapter is to optimize the weights of the network to force it to have the best performance when failures happen. For this purpose, we propose new *vulnerability metrics* based on the network criticality and algebraic connectivity of the network when a single node/link failure happens. The first set of our metrics quantify the worst case (maximum possible value) and expected value that network criticality may take after a single node/link failure. We show that these metrics are convex functions of link weights and we introduce and solve convex optimization problems to minimize these metrics. We will note that each optimization problem can be useful depending on the nature of the network under study and its possible failures. The second set of our proposed metrics are defined based on the worst case (minimum possible value) and expected value that algebraic connectivity may take after a single node/link failure. We show that these metrics are concave functions of link weights and introduce and solve convex optimization problems related to each metric.

The main communication networks of interest in this chapter are core networks. Therefore, we apply the proposed optimizations on Rocketfuel networks which are the most trustable publicly available dataset for real ISP networks. We present the results of applying each optimization problem on these networks, and compare the results to introduce practical guidelines on using our metrics and optimizations.

5.1 Vulnerability Analysis

In this section we present the vulnerability analysis of a communication network. We define the vulnerability metrics of the network as functions of the variations of basic metrics in case of failures. We define vulnerability metrics based on two base metrics which are network criticality and algebraic connectivity.

5.1.1 $\hat{\tau}$-based vulnerability metrics

We model a communication network with an undirected weighted graph $G(N, E, W)$ where N and E are the set of nodes and links respectively, and W is the symmetric link weight matrix. After the failure of a link (i, j), the network topology changes to a new network. We assume that G is a connected graph, and it remains connected after a single link or node failure. The new network after the failure of link (i, j) has a new network criticality which we denote it as $\hat{\tau}^{(ij)}$. In [22] it is proved that $\hat{\tau}$ is a monotone decreasing function of link weights:

$$\frac{\partial \hat{\tau}}{\partial w_{ij}} < 0 \tag{5.1}$$

Therefore, failing a link that is equivalent to reducing w_{ij} from a positive value to zero increases the quantity of network criticality and we have:

$$\hat{\tau}^{(ij)} > \hat{\tau} \tag{5.2}$$

for all the links in E. We can sort all of the network links based on their corresponding $\hat{\tau}^{(ij)}$. If the value of $\hat{\tau}^{(ij)}$ corresponding to the link (i, j) is more, the network is more vulnerable to the failure of that link. The link (i, j) is called the most critical

link of the network if its $\hat{\tau}^{(ij)}$ is the largest among all $\hat{\tau}^{(ij)}$'s. In a similar way, we denote the $\hat{\tau}$ of the network after the failure of the node i as $\hat{\tau}^{(i)}$. Once more, we sort all of the network nodes by their $\hat{\tau}^{(i)}$, and we call the network to be more vulnerable to the failure of node i if $\hat{\tau}^{(i)}$ corresponding to that node is more. The most critical node of the network is the node for which $\hat{\tau}^{(i)}$ is the most compared to all of the other nodes.

Now we define four *vulnerability metrics* of the network as the worst case and expected value of $\hat{\tau}^{(ij)}$'s and $\hat{\tau}^{(i)}$'s over all the links/nodes of the network. We denote these vulnerability metrics as $max(\hat{\tau}^{(ij)})$, $E(\hat{\tau}^{(ij)})$, $max(\hat{\tau}^{(i)})$, and $E(\hat{\tau}^{(i)})$. The first metric, $max(\hat{\tau}^{(ij)})$, is defined as the maximum value (worst value) of $\hat{\tau}^{(ij)}$'s over all the links of the network, and is equal to $\hat{\tau}^{(ij)}$ correspond to the most critical link of the network.

$$max(\hat{\tau}^{(ij)}) = max\{\hat{\tau}^{(ij)} | (i,j) \in E\} \tag{5.3}$$

Similarly, we define $max(\hat{\tau}^{(i)})$ as the maximum value (worst value) of $\hat{\tau}^{(i)}$'s over all the nodes. In the following section we explain the details about the probabilistic approach to failures and definition of other two metrics, i.e. $E(\hat{\tau}^{(ij)})$ and $E(\hat{\tau}^{(i)})$.

5.1.2 Probabilistic Approach to Failures

In this section we consider different failure probability for various links and nodes of the network, and define vulnerability metrics which take this into account. The first metric is the expected value of $\hat{\tau}^{(ij)}$ when a single link failure happens which we denote it by $E(\hat{\tau}^{(ij)})$. If we denote the failure probability of link (i,j) in case of a

single link failure with π_{ij}, then:

$$E(\hat{\tau}^{(ij)}) = \sum_{(i,j)\in E} \pi_{ij}\hat{\tau}^{(ij)} \tag{5.4}$$

We define $E(\hat{\tau}^{(i)})$ as the expected value of $\hat{\tau}^{(i)}$'s over all the nodes of the network in a similar way. If we denote the failure probability of node i in case of a single node failure with π_i, then:

$$E(\hat{\tau}^{(i)}) = \sum_{i\in N} \pi_i\hat{\tau}^{(i)} \tag{5.5}$$

We notice that smaller values of these vulnerability metrics make the network less vulnerable to failures corresponding to that metric.

Generally we do not have π_{ij}'s (π_i's) directly and we should formulate them in terms of link (node) failure probabilities. For this purpose, let α_{ij} be the probability of the failure of link (i, j), β_i be the probability of the failure of node i and assume that all the link or node failures are independent events. We denote the event of the single failure of link (i, j) while all the other links and nodes of the network are working normally with F_{ij}, the event of the single failure of node i while all the other nodes and links are working normally with F_i, the event of happening a Single Link (SL) failure in the network with F_{SL} and the event of happening a Single Node (SN)failure in the network with F_{SN}. We note that F_{ij}'s and F_i's are mutually exclusive events since each event is related to the failure of a single link or node while all the other network elements operate normally. Now we can calculate π_{ij}'s (π_i's) in terms of α_{ij}'s and β_i's using the following formulation:

$$Pr(F_{ij}) = \alpha_{ij} \prod_{(k,p)\in E-(i,j)} (1-\alpha_{kp}) \prod_{k\in N} (1-\beta_k) \tag{5.6}$$

$$Pr(F_i) = \beta_i \prod_{(k,p)\in E} (1-\alpha_{kp}) \prod_{k\in N-i} (1-\beta_k)$$

$$Pr(F_{SL}) = \sum_{(k,p)\in E} Pr(F_{kp})$$

$$Pr(F_{SN}) = \sum_{k\in N} Pr(F_k)$$

$$\pi_{ij} = Pr(F_{ij}|F_{SL}) = \frac{Pr(F_{ij}F_{SL})}{Pr(F_{SL})} = \frac{Pr(F_{ij})}{Pr(F_{SL})}$$

$$\pi_i = Pr(F_i|F_{SN}) = \frac{Pr(F_iF_{SN})}{Pr(F_{SN})} = \frac{Pr(F_i)}{Pr(F_{SN})}$$

Here by $Pr(e)$ we mean the probability of the event e. We note that the first two lines of this formulation is based on the assumption that the link and node failures are independent events.

5.1.3 λ_2-based vulnerability metrics

Now we develop similar concepts based on the algebraic connectivity as the base metric of interest. Algebraic connectivity (λ_2) is defined as the second smallest eigenvalue of the Laplacian matrix, where the smallest eigenvalue of the Laplacian matrix is always zero, and $\lambda_2 > 0$ if the graph is connected. In [4] it is shown that λ_2 is a monotone non-decreasing function of link weights:

$$\frac{\partial \lambda_2}{\partial w_{ij}} \geq 0 \tag{5.7}$$

and it can be considered as a connectivity metric for a weighted graph. Now we consider λ_2 of the graph when a link or node fails in the network. We denote the λ_2 of the network when the link (i,j) fails as $\lambda_2^{(ij)}$. Since failing a link is equivalent to

decreasing its weight to zero from a positive value, we have:

$$\lambda_2^{(ij)} \leq \lambda_2 \tag{5.8}$$

We note here that the larger values of λ_2 or $\lambda_2^{(ij)}$ are more desired. Therefore, we can sort all the links of the network based on their $\lambda_2^{(ij)}$, and the link with the least $\lambda_2^{(ij)}$ is called the most λ_2-critical link in the network. In a similar way, we define $\lambda_2^{(i)}$ as the algebraic connectivity of the network when the node i fails. Again, we can sort the network nodes based on their $\lambda_2^{(i)}$'s. The node with the lowest $\lambda_2^{(i)}$ is called the most λ_2-critical node of the network.

Now we define four λ_2-based vulnerability metrics of the network as the worst case and expected value of $\lambda_2^{(ij)}$'s and $\lambda_2^{(i)}$'s when a single link/node failure happens. The first metric is $min(\lambda_2^{(ij)})$ which is the minimum (the worst value) of $\lambda_2^{(ij)}$ over all links of the network. This value corresponds to the value of $\lambda_2^{(ij)}$ for the most λ_2-critical link, and shows the worst value that algebraic connectivity may take after a single link failure. The next metric is $E(\lambda_2^{(ij)})$ which is the expected value of $\lambda_2^{(ij)}$'s when a single link failure happens:

$$E(\lambda_2^{(ij)}) = \sum_{(i,j)\in E} \pi_{ij}\lambda_2^{(ij)} \tag{5.9}$$

We define vulnerability metrics related to a single node failure in a similar way. The first metric is $min(\lambda_2^{(i)})$ which is the worst value that λ_2 may take through a single node failure. This metric is equal to the $\lambda_2^{(i)}$ correspond to the most λ_2-critical node in the network. The next metric is $E(\lambda_2^{(i)})$ which is the expected value of $\lambda_2^{(i)}$'s over all the links of the network:

$$E(\lambda_2^{(i)}) = \sum_{i\in N} \pi_i\lambda_2^{(i)} \tag{5.10}$$

Table 5.1: summary of the definition of the metrics

$\hat{\tau}$	Network Criticality
$max(\hat{\tau}^{(ij)})$	Maximum of $\hat{\tau}$ in case of a single link failure
$E(\hat{\tau}^{(ij)})$	Expected value of $\hat{\tau}$ in case of a single link failure
$max(\hat{\tau}^{(i)})$	Maximum of $\hat{\tau}$ in case of a single node failure
$E(\hat{\tau}^{(i)})$	Expected value of $\hat{\tau}$ in case of a single node failure
λ_2	Algebraic Connectivity
$min(\lambda_2^{(ij)})$	Minimum of λ_2 in case of a single link failure
$E(\lambda_2^{(ij)})$	Expected value of λ_2 in case of a single link failure
$min(\lambda_2^{(i)})$	Minimum of λ_2 in case of a single node failure
$E(\lambda_2^{(i)})$	Expected value of λ_2 in case of a single node failure

Table 5.1 shows a summary of the metrics we defined in this part. We can optimize each of these metrics in planning a communication network. In the following subsections, we first prove that all of the defined metrics are convex or concave functions of link weights. Then we introduce convex optimization problems to optimize each of them, and explain methods to convert all the optimization problems to semidefinite programs (SDPs).

5.1.4 Convex and Concave Vulnerability Metrics

In the previous section we showed that $\hat{\tau}$ is a convex function of link weights, while λ_2 is a concave one. Here we prove that all of the $\hat{\tau}$-based metrics in Table 5.1 are convex functions of link weights and all of the λ_2-based one's are concave.

Proposition 5.1.1. *$\hat{\tau}^{(ij)}$ and $\hat{\tau}^{(i)}$ ($\lambda_2^{(ij)}$ and $\lambda_2^{(i)}$) are convex (concave) functions of*

link weights.

Proof. In theorem 3.1.1 it is proved that $\hat{\tau}$ is a convex function of link weights. Since $\hat{\tau}^{(ij)} = \hat{\tau}|_{w_{ij}=0}$, $\hat{\tau}^{(ij)}$ is also a convex function of link weights. In addition, $\hat{\tau}^{(i)}$ is equal to $\hat{\tau}$ when the weights of all the links incident to node i are set to zero, so $\hat{\tau}^{(i)}$ is also a convex function of link weights. With a similar reasoning, it can be easily proved that $\lambda_2^{(ij)}$ and $\lambda_2^{(i)}$ are concave functions of link weights.

□

Proposition 5.1.2. *$max(\hat{\tau}^{(ij)})$, $E(\hat{\tau}^{(ij)})$, $max(\hat{\tau}^{(i)})$, and $E(\hat{\tau}^{(i)})$ are convex functions of link weights. Furthermore, $min(\lambda_2^{(ij)})$, $E(\lambda_2^{(ij)})$, $min(\lambda_2^{(i)})$, and $E(\lambda_2^{(i)})$ are concave.*

Proof. According to proposition 5.1.1, $\hat{\tau}^{(ij)}$'s are convex functions of link weights. Therefore, $max(\hat{\tau}^{(ij)})$ which is the maximum of a number of convex functions (eq. 5.3) is a convex function of link weights [98]. Moreover, $E(\hat{\tau}^{(ij)})$ is a convex function of link weights since it is a weighted sum of a number of convex functions with non-negative coefficients (eq. 5.4). With a similar reasoning, it can be proved that $max(\hat{\tau}^{(i)})$, and $E(\hat{\tau}^{(i)})$ are convex functions of link weights as well. A similar proof is valid for concavity of λ_2-based metrics.

□

Here we note that although $\hat{\tau}$ is an *strictly* convex function of link weights, it can be easily verified that $\hat{\tau}^{(ij)}$ and $\hat{\tau}^{(i)}$ are not *strictly* convex, and as a result, $\hat{\tau}$-based vulnerability metrics are not *strictly* convex. In addition, non of the λ_2-based metrics (including λ_2 itself) is strictly concave.

5.1.5 Optimizing the Vulnerability Metrics

Consider a weighted graph $G(N, E, W)$ with a symmetric cost matrix $Z = [z_{ij}]$ where z_{ij} is the cost of assigning a unit of weight to the link (i, j). Then the total cost of the planning for the network would be $\sum_{(i,j)\in E} z_{ij}w_{ij}$. If $\Phi(W)$ is a convex metric of the Table 5.1, the following optimization problem minimizes this metric:

$$\begin{aligned} &Minimize \quad \Phi(W) \\ &Subject\ to \quad \sum_{(i,j)\in E} z_{ij}w_{ij} \leq C,\ C\ is\ fixed \\ &\qquad\qquad w_{ij} \geq 0 \qquad \forall (i,j) \in E \end{aligned} \tag{5.11}$$

This is a convex optimization problem since it has a convex objective function and linear constraints, and the optimum value obtained from it is a global optimum. It is also noteworthy that at optimum $\sum_{(i,j)\in E} z_{ij}w_{ij} = C$, and we can relax the inequality $\sum_{(i,j)\in E} z_{ij}w_{ij} \leq C$ to equality.

If $\Phi(W)$ is an strictly convex function of link weights (which happens only when $\Phi(W) = \hat{\tau}$), then the optimum weight matrix W^* is unique. Since the metrics in Table 5.1 are not strictly convex (except for $\hat{\tau}$), the optimum weight set corresponding to the optimum value for (5.11) may not be unique. We pick the weight matrix with the smallest (most desirable) $\hat{\tau}$ from the set of solutions to (5.11) as the most robust solution through a second optimization. If ζ is the optimum value obtained from (5.11), then the second optimization problem is:

$$
\begin{aligned}
&Minimize \quad \hat{\tau} \\
&Subject\ to \quad \Phi(W) \leq \zeta \\
&\qquad \sum_{(i,j)\in E} z_{ij} w_{ij} \leq C \\
&\qquad w_{ij} \geq 0 \qquad \forall (i,j) \in E
\end{aligned}
\tag{5.12}
$$

Since the minimum value that $\Phi(W)$ can take is ζ, the feasible set of (5.12) is the solution set of (5.11). However, as the solution set of a convex optimization problem is a convex set [98], optimization problem (5.12) is a convex optimization problem in turn. There is a practical point here that because of the rounding problem, when optimization (5.12) is solved numerically, inequality: $\Phi(W) \leq \zeta$ should be relaxed to: $\Phi(W) \leq \zeta + \epsilon$, where ϵ is a very small number (e.g. $n \times 10^{-8}$).

We denote the result of optimizations (5.11) and (5.12) based on metric of interest as follows:

mT (**m**inimize $\hat{\boldsymbol{\tau}}$) if $\Phi(W) = \hat{\tau}$;

mMTL (**m**inimize **M**aximum $\hat{\boldsymbol{\tau}}$ in case of **L**ink failures) if $\Phi(W) = max(\hat{\tau}^{(ij)})$;

mMTN (**m**inimize **M**aximum $\hat{\boldsymbol{\tau}}$ in case of **N**ode failures) if $\Phi(W) = max(\hat{\tau}^{(i)})$;

mETL (**m**inimize **E**xpected value of $\hat{\boldsymbol{\tau}}$ in case of **L**ink failures) if $\Phi(W) = E(\hat{\tau}^{(ij)})$;

mETN (**m**inimize **E**xpected value of $\hat{\boldsymbol{\tau}}$ in case of **N**ode failures) if $\Phi(W) = E(\hat{\tau}^{(i)})$.

In order to optimize λ_2-based metrics which are concave functions of link weights, we change the first optimization problem to a maximization problem, and we do a similar second optimization to obtain the most robust weight matrix from the solution set of the first optimization. We have five new optimization problems which we denote as follows:

ML (**M**aximize $\boldsymbol{\lambda_2}$) if $\Phi(W) = \lambda_2$;

MmLL (**M**aximize **m**inimum $\boldsymbol{\lambda_2}$ in case of **L**ink failures) if $\Phi(W) = min(\lambda_2^{(ij)})$;

MmLN (**M**aximize **m**inimum $\boldsymbol{\lambda_2}$ in case of **N**ode failures) if $\Phi(W) = min(\lambda_2^{(i)})$;

MELL (**M**aximize **E**xpected value of $\boldsymbol{\lambda_2}$ in case of **L**ink failures) if $\Phi(W) = E(\lambda_2^{(ij)})$;

MELN (**M**aximize **E**xpected value of $\boldsymbol{\lambda_2}$ in case of **N**ode failures) if $\Phi(W) = E(\lambda_2^{(i)})$.

A summary of our naming convention for the optimization problems is shown in Table 5.2.

5.2 Semidefinite Programming (SDP) Formulation

We can convert all the optimization problems introduced above to SDPs [99, 100] that can be solved numerically faster and more efficiently. Here we show the derivation for a few cases, and the rest can be obtained similarly.

5.2.1 SDP formulation of mT

Similar to what we did to obtain 3.11, the SDP equivalent of mT can be obtained as follows:

$$\begin{aligned} Minimize \quad & \frac{2}{n-1}(Tr(\Gamma) - 1) \\ Subject\ to \quad & \sum_{(i,j)\in E} z_{ij} w_{ij} \leq C \\ & \begin{pmatrix} \Gamma & I \\ I & L + \frac{J}{n} \end{pmatrix} \succeq 0 \\ & Diag(Vec(W)) \succeq 0 \end{aligned} \tag{5.13}$$

where $Diag(Vec(W))$ is a diagonal matrix with all the elements of weight matrix W on its main diagonal and $\overrightarrow{1}$ is a $n \times 1$ vector with all of its elements equal to 1. This optimization problem is a semidefinite program since its objective is a linear function and its constraints are linear matrix inequalities (LMI) [98].

5.2.2 SDP Formulation of mETL

We denote the Laplacian of the remaining graph when link (i, j) fails with $L^{(ij)}$. It is easily seen that $L^{(ij)} = L|_{w_{ij}=0}$ and we can state $\hat{\tau}^{(ij)}$ as:

$$\hat{\tau}^{(ij)} = \frac{2}{n-1} Tr(L^{(ij)+}) = \frac{2}{n-1} Tr(L^{(ij)} + J/n)^{-1} - \frac{2}{n-1}$$

Please see [4] for more explanation about this equation. Now we can rewrite first optimization problem of mETL as follows:

$$\begin{aligned} Minimize \quad & \frac{2}{n-1}\,(t-1) \\ Subject\ to \quad & \sum_{(i,j)\in E} z_{ij} w_{ij} \leq C \\ & Tr(L^{(ij)} + J/n)^{-1} \leq t_{ij} \quad \forall (i,j) \in E \\ & \sum_{(i,j)\in E} \pi_{ij} t_{ij} \leq t \\ & Diag(Vec(W)) \succeq 0 \end{aligned} \tag{5.14}$$

If we consider $\Theta_{ij} = \begin{pmatrix} \Gamma^{(ij)} & I \\ I & L^{(ij)} + \frac{J}{n} \end{pmatrix}$, the Schur complement [28] of Θ_{ij} is $\Gamma^{(ij)} - (L^{(ij)} + J/n)^{-1}$. Θ_{ij} is positive-semidefinite iff its Schur complement is positive-semidefinite, i.e.:

$$\Theta_{ij} = \begin{pmatrix} \Gamma^{(ij)} & I \\ I & L^{(ij)} + \frac{J}{n} \end{pmatrix} \succeq 0 \Leftrightarrow \Gamma^{(ij)} \succeq (L^{(ij)} + J/n)^{-1} \tag{5.15}$$

and the inequality $\Gamma^{(ij)} \succeq (L^{(ij)} + J/n)^{-1}$ results in $Tr(\Gamma^{(ij)}) \geq Tr(L^{(ij)} + J/n)^{-1}$. Therefore we can reformulate (5.14) as the following semidefinite program:

$$
\begin{aligned}
&\textit{Minimize} \quad \frac{2}{n-1}\,(t-1) && && (5.16)\\
&\textit{Subject to} \quad \sum_{(i,j)\in E} z_{ij} w_{ij} \leq C && \\
&\qquad \begin{pmatrix} \Gamma^{(ij)} & I \\ I & L^{(ij)} + \frac{J}{n} \end{pmatrix} \succeq 0 && \forall (i,j) \in E \\
&\qquad Tr(\Gamma^{(ij)}) \leq t_{ij} && \forall (i,j) \in E \\
&\qquad \sum_{(i,j)\in E} \pi_{ij} t_{ij} \leq t && \\
&\qquad Diag(Vec(W)) \succeq 0 &&
\end{aligned}
$$

where $\Gamma^{(ij)}$'s are m variable $n \times n$ matrices, and t_{ij}'s are m real variables. The second optimization problem for mMTL can be converted to an SDP similarly.

5.2.3 SDP formulation of mMTN

The process of converting mMTN to an SDP is a little different since the network will have $n-1$ nodes after a node failure. If we denote the Laplacian, weight matrix , and diagonal matrix of weighted node degrees of the network after failure of the node i with $L^{(i)}$, $W^{(i)}$ and $D^{(i)}$ respectively, then:

$$
\begin{aligned}
&W^{(i)} = I_n^{(i)}\ W\ I_n^{(i)t} \\
&L^{(i)} = D^{(i)} - W^{(i)} \\
&\hat{\tau}^{(i)} = \frac{2}{n-2} Tr(L^{(i)+}) \\
&L^{(i)+} = (L^{(i)} + \frac{J}{(n-1)})^{-1} - \frac{J}{(n-1)}
\end{aligned}
$$

where $I_n^{(i)}$ is an $(n-1) \times n$ matrix obtained by removing ith row from $n \times n$ identity matrix I_n. The SDP formulation of mMTN can be formulated as follows similar to the optimization problem (5.16):

$$
\begin{aligned}
&Minimize \quad \frac{2}{n-2}\,(t-1) && && (5.17)\\
&Subject\ to \quad \sum_{(i,j)\in E} z_{ij} w_{ij} \le C && \\
&\qquad \begin{pmatrix} \Gamma^{(i)} & I_{n-1} \\ I_{n-1} & L^{(i)} + \frac{J}{n-1} \end{pmatrix} \succeq 0 && \forall i \in N \\
&\qquad Tr(\Gamma^{(i)}) \le t && \forall i \in N \\
&\qquad Diag(Vec(W)) \succeq 0 &&
\end{aligned}
$$

where $\Gamma^{(i)}$'s are n variable $(n-1) \times (n-1)$ matrices, and I_{n-1} is $(n-1) \times (n-1)$ identity matrix.

5.2.4 SDP Formulation of ML

In this part we convert optimization ML to an SDP. For this purpose we rewrite the first optimization problem of ML as follows:

$$
\begin{aligned}
&Maximize \quad t && (5.18)\\
&Subject\ to \quad \lambda_2 \ge t \\
&\qquad \sum_{(i,j)\in E} z_{ij} w_{ij} \le C \\
&\qquad w_{ij} \ge 0
\end{aligned}
$$

It can be easily proved that the inequality constraint $\lambda_2 \geq t$ is equivalent to the following:

$$L \succeq t(I - J/n) \tag{5.19}$$

The reason is that the eigenvalues of the matrix $L - t(I - J/n)$ are $0, \lambda_2 - t, ..., \lambda_n - t$ and positive-semidefiniteness of this matrix leads to $\lambda_2 \geq t$ and vice versa. Therefore we can convert (5.18) to the following:

$$\begin{aligned} &Maximize \quad t \\ &Subject\ to \quad \sum_{(i,j)\in E} z_{ij} w_{ij} \leq C \\ &\qquad\qquad L \succeq t(I - J/n) \\ &\qquad\qquad Diag(Vec(W)) \succeq 0 \end{aligned} \tag{5.20}$$

which is an SDP.

5.3 Evaluations

In section 5.1, we introduced 10 metrics for a network when modeled as a weighted graph (see table 5.1), and we introduced 10 optimization problems corresponding to these metrics. In this section we apply these optimizations on different networks to observe the behavior and effects of each optimization problem. We use CVX [101], which is a package for specifying and solving convex programs, to solve the convex optimization problems formulated in the previous section. In order to provide a better sense of results, we start with a simple network and discuss the consequence of applying different optimization problems, and then we study the behavior of real networks (Rocketfuel and Abilene). We also compare our results for optimum weight

Table 5.2: List of the abbreviations for weight assignment methods

Method	Description	Optimized Metric
EW	Equal Weights	-
IW	Initial Weights	-
mT	**m**inimize $\hat{\tau}$	$\hat{\tau}$
mMTL	**m**inimize **M**aximum $\hat{\boldsymbol{\tau}}$ in case of **L**ink failures	$max(\hat{\tau}^{(ij)})$
mETL	**m**inimize **E**xpected value of $\hat{\boldsymbol{\tau}}$ in case of **L**ink failures	$E(\hat{\tau}^{(ij)})$
mMTN	**m**inimize **M**aximum $\hat{\boldsymbol{\tau}}$ in case of **N**ode failures	$max(\hat{\tau}^{(i)})$
mETN	**m**inimize **E**xpected value of $\hat{\boldsymbol{\tau}}$ in case of **N**ode failures	$E(\hat{\tau}^{(i)})$
ML	**M**aximize $\boldsymbol{\lambda_2}$	λ_2
MmLL	**M**aximize **m**inimum $\boldsymbol{\lambda_2}$ in case of **L**ink failures	$min(\lambda_2^{(ij)})$
MELL	**M**aximize **E**xpected value of $\boldsymbol{\lambda_2}$ in case of **L**ink failures	$E(\lambda_2^{(ij)})$
MmLN	**M**aximize **m**inimum $\boldsymbol{\lambda_2}$ in case of **N**ode failures	$min(\lambda_2^{(i)})$
MELN	**M**aximize **E**xpected value of $\boldsymbol{\lambda_2}$ in case of **N**ode failures	$E(\lambda_2^{(i)})$

assignment using optimization problems with two more weight sets. The first weight set is Initial Weights (IW), in which the link weights are proportional to the link capacities of the real networks under study. We use the initial link weights and the link costs z_{ij}'s, to obtain the total cost of the planning for the initial network, i.e. C, using the equation: $C = \sum_{(i,j)\in E} z_{ij} w_{ij}$, and use equal value in all of the optimization problems we apply on the network. The other weight set is Equal Weights (EW), in which we assign equal weight to all the network links. We hold the equation $\sum_{(i,j)\in E} z_{ij} w_{ij} = C$ by assigning the same weight to all of the network links as follows:

$$w_{ij} = C/(\sum_{(i,j)\in E} z_{ij}) \quad \forall (i,j) \in E$$

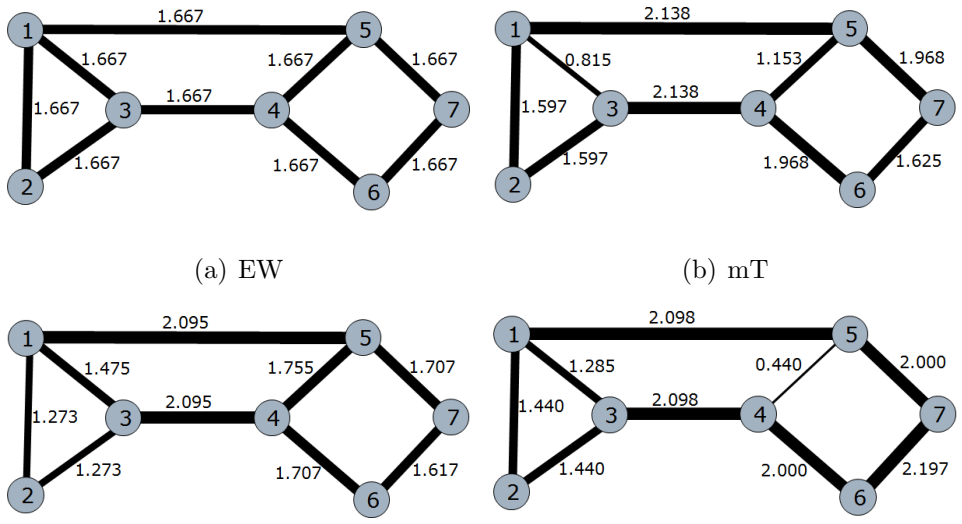
Then there are 12 weight assignments for every network in total. Table 5.2 presents a review of all the proposed methods and the corresponding optimized metric for each of them.

5.3.1 mT, mMTL and mMTN

We first start our evaluations with mT, mMTL, and mMTN methods. We apply these methods on the networks, and observe the effect of each method on the relevant metrics and compare them to IW and EW. Then we compare the routing performance for these methods in the presence of single link/node failures.

Fish Network

The topology of the Fish network for EW method is shown in Fig. 5.1-a. In this part we assume the total cost of planning to be equal to be $C = 30$, and all the links have equal planning cost equal to 1, i.e., $z_{ij} = 1\ \forall (i,j) \in E$. This network has 18 directed links and 7 nodes, and therefore the weight of all the links using EW method is $w_{ij} = \frac{30}{18} = 1.667$ as it is indicated in Fig. 5.1-a. The original Fish network used in



(a) EW

(b) mT

(c) mMTL

(d) mMTN

Figure 5.1: Weight Sets for Fish Network using EW, mT, mMTL, and mMTN Methods

[4] does not include any link between nodes 1 and 5 ; however, we add it here to the fish network to make it more asymmetric. The numerical value for network criticality in this case is: $\hat{\tau} = 0.5854$. Now by calculating the value of network criticality after different single link failures, we find that in case of failure of link $(1, 5)$ or $(3, 4)$ (in both directions) network criticality changes to $\hat{\tau}^{(15)} = \hat{\tau}^{(34)} = max(\hat{\tau}^{(ij)}) = 0.9095$, and since this is the maximum value $max(\hat{\tau}^{(ij)})$ can take in this network, links $(1, 5)$ and $(3, 4)$ are the most critical links of the network. The corresponding value of network criticality for a single node failure is $max(\hat{\tau}^{(i)}) = 1.1200$ which occurs with the failure of the node 5 or 4, i.e., $\hat{\tau}^{(5)} = \hat{\tau}^{(4)} = max(\hat{\tau}^{(i)}) = 1.1200$. Since the failure of a node effectively is equivalent to the failure of the set of links incident to that node, it affects the network more severely, and increases the network criticality more than link failures.

Now we examine the optimized weight sets for mMTL and mMTN methods.

Once more, in both of these optimizations for Fish network, we assume $z_{ij} = 1$ for $\forall(i,j) \in E$ and $C = \sum_{(i,j)\in E} z_{ij} w_{ij} = 30$. Fig. 5.1-b to Fig. 5.1-d illustrate the computed weight sets for mT, mMTL, and mMTN methods for Fish network. According to this figure, the weight matrix is different for each optimization, and as an example, for mMTL method the weight matrix for the network is as follows:

$W_{mMTL} =$

$$\begin{pmatrix} 0 & 1.273 & 1.475 & 0 & 2.095 & 0 & 0 \\ 1.273 & 0 & 1.273 & 0 & 0 & 0 & 0 \\ 1.475 & 1.273 & 0 & 2.095 & 0 & 0 & 0 \\ 0 & 0 & 2.095 & 0 & 1.755 & 1.707 & 0 \\ 2.095 & 0 & 0 & 1.755 & 0 & 0 & 1.707 \\ 0 & 0 & 0 & 1.707 & 0 & 0 & 1.617 \\ 0 & 0 & 0 & 0 & 1.707 & 1.617 & 0 \end{pmatrix}$$

Fig. 5.1-b indicates that to minimize network criticality in mT method, more weight is assigned to links $(3,4)$ and $(1,5)$ since they are more critical, and weights of the links $(1,3)$ and $(4,5)$ are significantly decreased. Table 5.3 shows the network criticality and vulnerability metrics of mT, mMTL, and mMTN methods. For mT method, we have $\hat{\tau} = 0.5600$ which is the best attainable network criticality value for the network, and is 4.33% less than $\hat{\tau}$ in EW case. Comparison of the vulnerability metrics of EW and mT shows that $max(\hat{\tau}^{(i)})$ for mT is also better (less), however, $max(\hat{\tau}^{(ij)})$ is better for EW. This shows that minimizing $\hat{\tau}$ does not neccesarily optimize all vulnerability metrics. The third and fourth rows of the Table 5.3 show that

the best attainable value for $max(\hat{\tau}^{(ij)})$ for Fish network is 0.8807, and takes place with mMTL wieght set. Moreover, the best possible value of $max(\hat{\tau}^{(i)})$ is 0.9945, and corresponds to mMTN weight set. The other interesting point we observe in Table 5.3 is that the changes of $max(\hat{\tau}^{(ij)})$ and $max(\hat{\tau}^{(i)})$ in different methods is more than the changes of $\hat{\tau}$ in various weight assignments.

Table 5.3: Parameters of Fish Network for Various Weight sets

	EW	mT	mMTL	mMTN
$\hat{\tau}$	0.5853	0.5600	0.5762	0.5726
$max(\hat{\tau}^{(ij)})$	0.9095	0.9221	0.8807	0.9861
$max(\hat{\tau}^{(i)})$	1.1200	1.0782	1.1015	0.9945

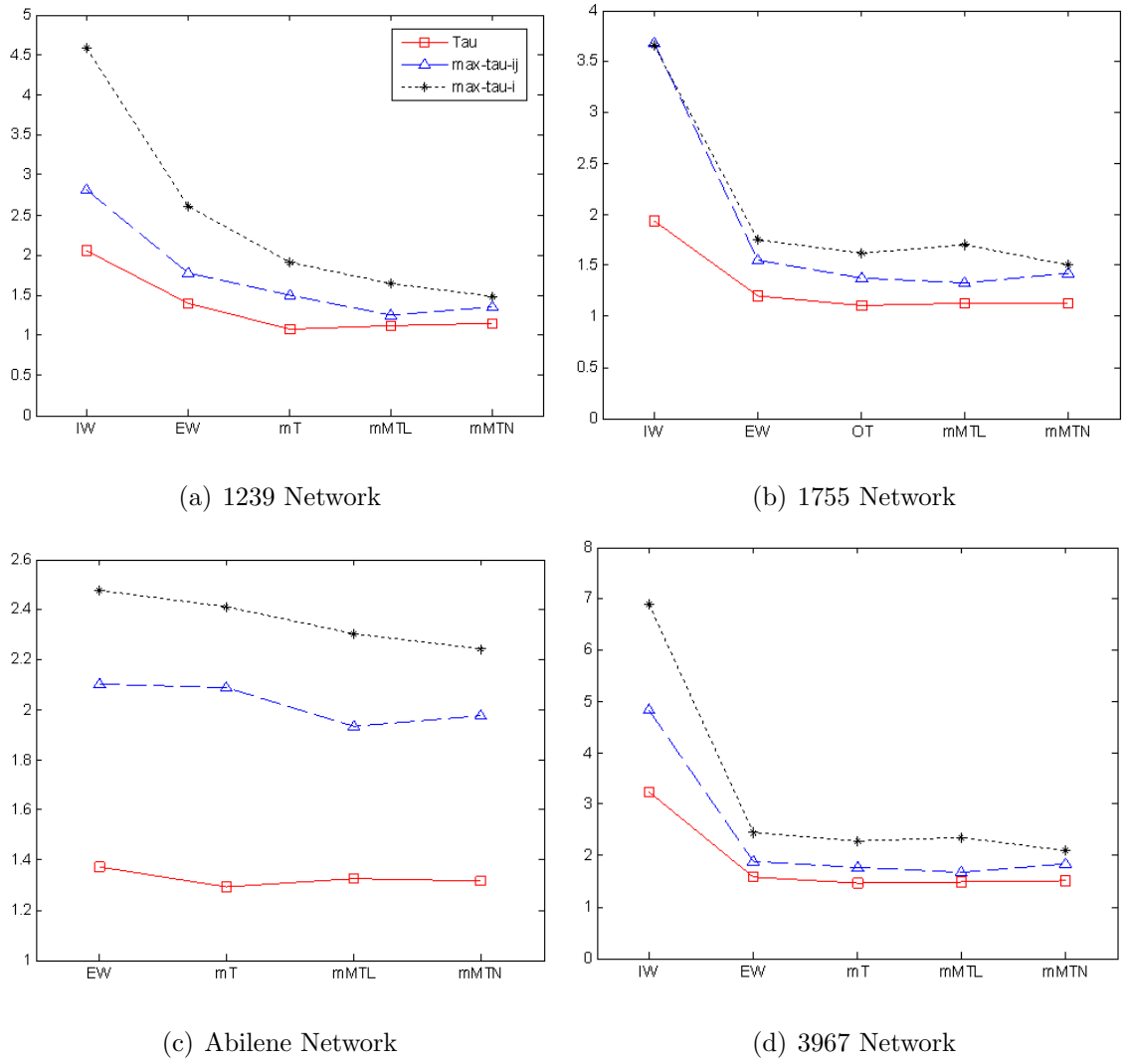
Rocketfuel Networks and Abilene

Table 5.4: Rocketfuel Dataset ISPs

ISP	Routers	Links	Reduced Cities	Reduced Links	Weight per Link	Total Weight
1755	87	322	18	33	0.7822	51.628
3967	79	294	21	36	0.6743	48.551
1239	315	1944	30	69	0.7231	99.784

Table 5.5: Parameters of 1755 Network for Various Weight sets

	EW	IW	mT	mMTL	mMTN
$\hat{\tau}$	1.9408	1.1977	1.1013	1.1239	1.1257
$max(\hat{\tau}^{(ij)})$	3.6802	1.5478	1.3774	1.3277	1.4190
$max(\hat{\tau}^{(i)})$	3.6557	1.7525	1.6188	1.7070	1.4997



(a) 1239 Network

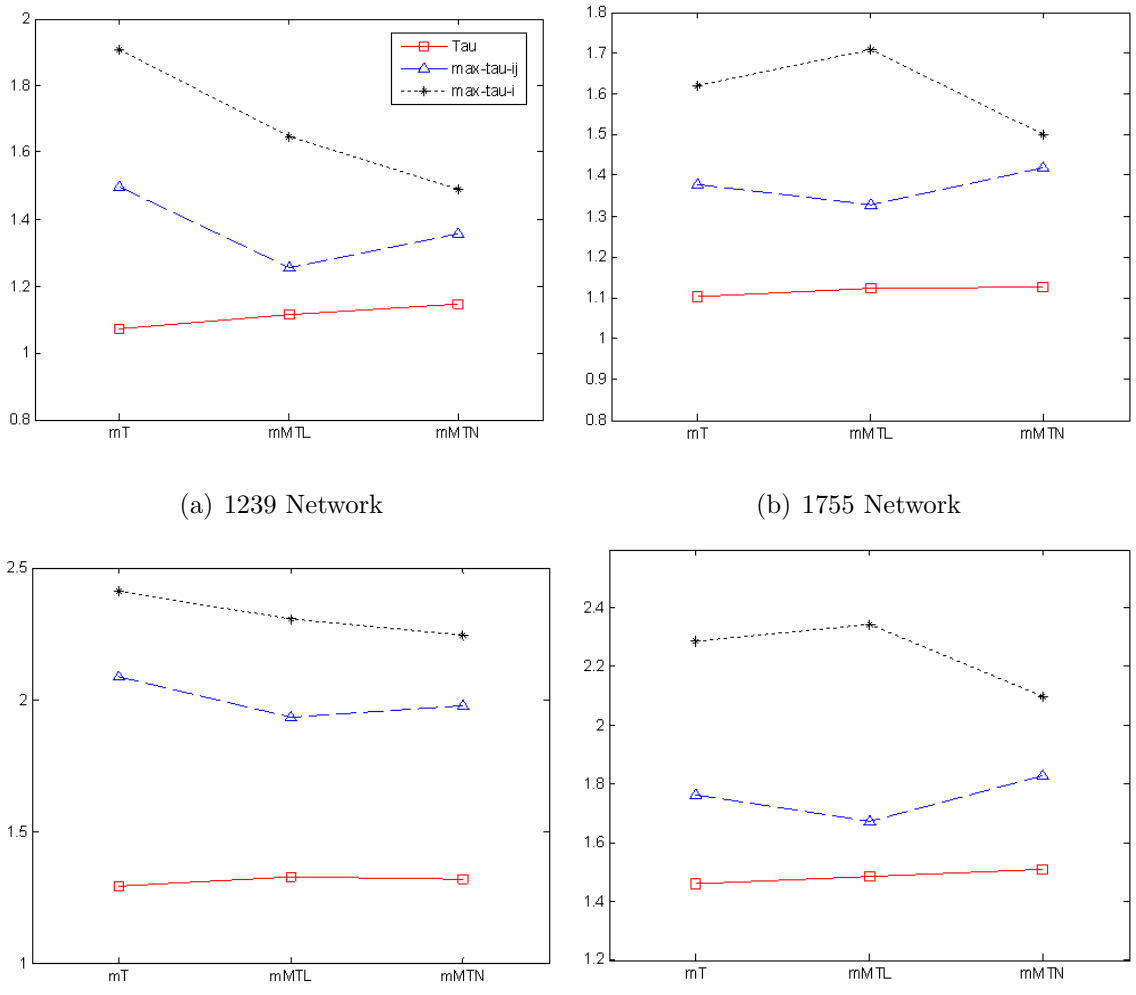
(b) 1755 Network

(c) Abilene Network

(d) 3967 Network

Figure 5.2: Comparison of IW and EW with optimized Weight Sets for Abilene and Rocketfuel Networks

Rocketfuel topologies [102] are the most important ISP networks that their dataset are publicly available. In this part, we apply the first set of our optimizations on Rocketfuel topologies [102]. We obtained the maps of Rocketfuel ISPs from their available dataset and adopted a method from [52] to collapse and simplify the topologies. According to this method, we considered each city of the initial network as a single node



(a) 1239 Network

(b) 1755 Network

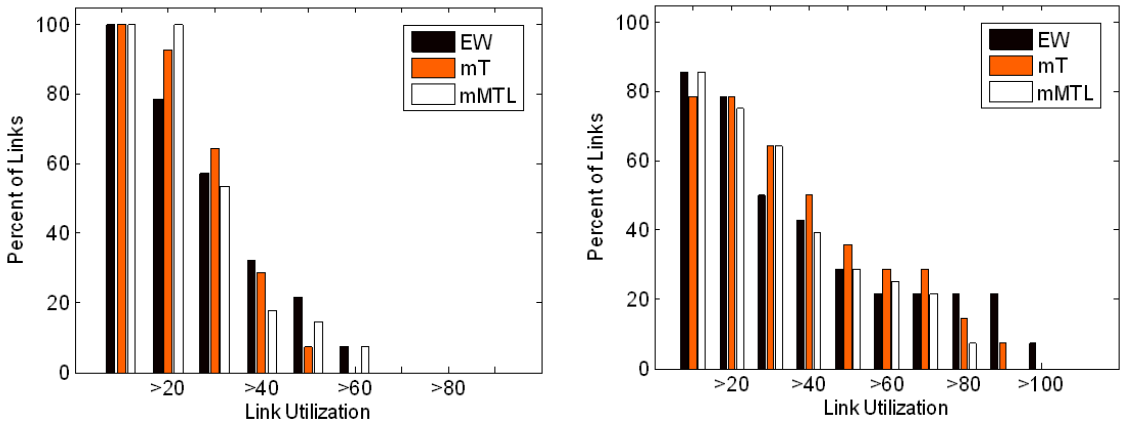
(c) Abilene Network

(d) 3967 Network

Figure 5.3: Parameters of Different Optimized Weight Sets for Abilene and Rocketfuel Networks

of the final network with a single link between nodes in the final network, and summed up the capacity of all the links between cities in the initial network to obtain the total capacity of the equivalent link between corresponding nodes in the final network. There were links between routers inside each city in the initial networks that were eliminated by considering each city as a node of the final network. We also eliminated

leaf nodes of the final network in a few iterations to reach the topologies with nodes of degree two or more. Table 5.4 indicates the final data for the Rocketfuel networks after our processing. We also performed our tests on Abilene [103] network. Abilene includes with 11 undirected links and 9 nodes. Since the networks under experiment in this part are real networks, they already have an initial weight (IW) set, and the weights are considered to be proportional to the link capacity of the real networks. EW weight sets for Rocketfuel networks are different from IW since the initial links have different capacities, however, just for Abilene network, EW and IW are the same as all the links have equal capacities. The link costs ($z'_{ij}s$) are considered to be 1 for all links in these series of experiments.

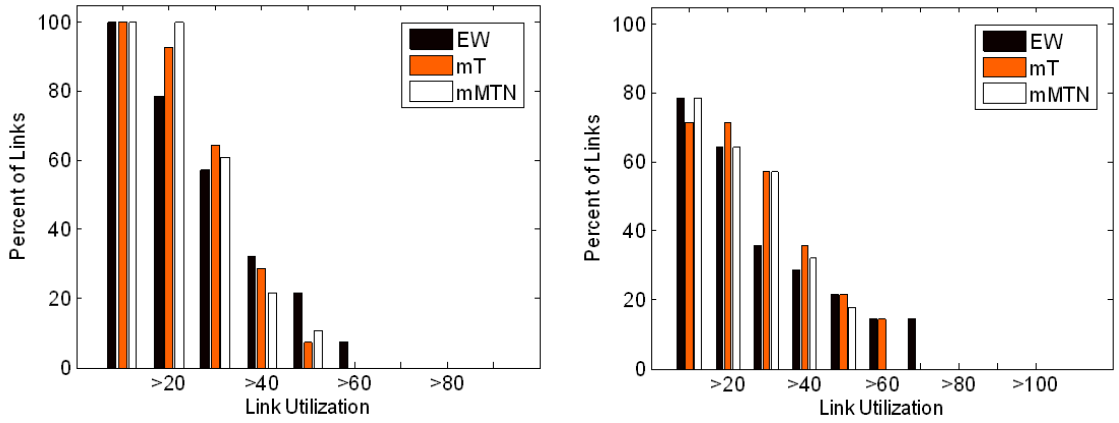


(a) No Failure

(b) Single Link Failure

Figure 5.4: Comparison of Cumulative Link Utilization for Three Weight Sets: EW, mT, and mMTL; Before and After a Link Failure

Network criticality and vulnerability metrics of Rocketfuel networks and Abilene for mT, mMTL, and mMTN methods are shown in Fig. 5.2. As it is easily seen in this figure, there is a large difference between the parameters of the initial weight (IW) and optimized weight assignments for Abilene and Rocketfuel networks. This



(a) No Failure

(b) Single Node Failure

Figure 5.5: Comparison of Cumulative Link Utilization for Three Weight Sets: EW, mT, and mMTN; Before and After a Node Failure

indicates that our proposed optimizations lead to a significant improvement in the vulnerability metrics for these networks. As an example, looking at Table 5.5 shows that the optimal metrics of 1755 network, i.e. $\hat{\tau}$, $max(\hat{\tau}^{(ij)})$, and $max(\hat{\tau}^{(i)})$, improve 42%, 61.4%, and 60% in optimum cases respectively compared to IW case.

Finally, the comparison of the metrics of Rocketfuel networks and Abilene for optimized weight assignments of interest in this section, i.e., mT, mMTL, and mMTN is shown in Fig. 5.3 while IW and EW are excluded. Similar to what we observed about the Fish network, we see that network criticality changes much less than vulnerability metrics $max(\hat{\tau}^{(ij)})$ and $max(\hat{\tau}^{(i)})$ in different optimization methods. Our experiments in the next subsection, confirms that this effect results in an improvement in the traffic routing performance after a failure happens.

Routing Performance

In the previous section we compared the metrics of different weight sets but we did not compare their performance when the traffic is applied to them. Here, we apply traffic to the networks under experiment with various weight sets and examine their routing performance. The dataset of Rocketfuel topologies does not include traffic matrices (TMs), so we synthesized traffic matrices for Rocketfuel topologies using Gravity model [53] [104]. This model is based on the assumption that real networks are designed based on the real expectations of traffic demand. Similar to [53] we assumed that the volume of external outgoing and incoming traffic for each of the nodes in the final Rocketfuel networks is proportional to the total capacity of the links attached to those nodes. Then we divided the external incoming traffic to each node between all nodes of the network proportional to the capacity of their incident links and in this way we calculated the full-mesh TM.

We used IGP-WO method in the TOTEM package [105] to measure the link utilization for the network links. IGP-WO is a tool which optimizes routing weights for intradomain Internet routing protocols like OSPF and IS-IS. It uses Tabu search meta-heuristic method [106] to obtain the routing weights of the links (which are different from link capacities) to utilize network in an optimum way. We set the link capacities equal to the weights we obtained from our optimizations or equal (initial) weights for EW (IW), and let the TOTEM calculate its routing weights and route the full-mesh gravity traffic matrix.

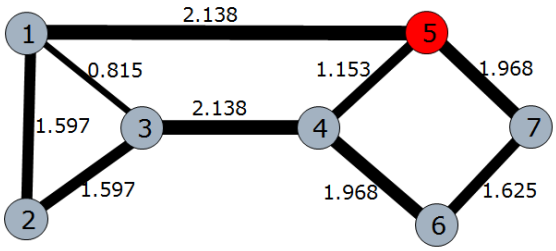
The cumulative distribution of the link utilization for the Abilene network for three different weight sets EW, mT, and mMTL before any failure is indicated in Fig. 5.4-a . As this figure shows link utilization for mT method has the best distribution as no link is utilized more than 60% in this case while there are links with high

utilization for mMTL and EW methods. In the next experiment, we compare the link utilization of these networks after the failure of the most critical link of the network. Again we measured the cumulative distribution of link utilzation for each method and the results are shown in Fig. 5.4-b. In this figure, we can clearly observe that mMTL has a better routing performance than mT and EW after the link failure because there is no link with a higher utilization than 90%, while mT and EW do include such heavy-loaded links.

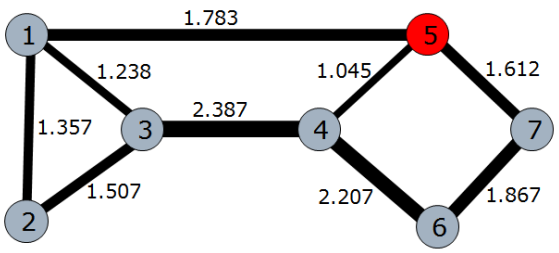
The next series of our tests examine the routing performance of mMTN after a single node failure. For this purpose, we compare the link utilizations of Abilene network for EW, mT, and mMTN methods before and after a node failure. Cumulative distribution of link utilization for each of the three weight sets is shown in Fig. 5.5-a when no failure has happened. We observe in this figure that mT has a better link utilization distribution as it does not have any link with a utilization higher than 60% but mMTN and EW have such links. Now we measure the link utilizations in all three methods after the failure of the most critical node of the network. Cumulative link utilization for each case is indicated in Fig. 5.5-b. We can clearly observe in this figure that the routing performance after the node failure for mMTN is better because there is no link with a utilization higher than 60% for this method; however, there are such highly utilized links for mT and EW methods.

5.3.2 Optimizing Expected Value of $\hat{\tau}$ after failures

In this part we apply mETL and mETN on the networks to observe the effect of them. The difference here is that mETL and mETN consider different failure probability for network nodes or links but two previous optimizations, i.e. mMTN and mMTL, do not. As an example consider Fish network when the failure probability for node



(a) mT



(b) mETN

	$\hat{\tau}$	$E(\hat{\tau}^{(i)})$	$\hat{\tau}^{(5)}$	MCN
mT	0.5606	0.8850	1.0782	4,5
mETN	0.5673	0.8672	0.9648	4

(c)

Figure 5.6: Comparison of mT with mETL for Fish Network When Failure of Node 5 is 3 Times More Probable Compared to the Other Nodes (MCN = Most Critical Node(s))

5 is 3 times more compared to the other nodes (Fig 5.6). As a numerical example, we assume the failure probability of node 5 is $\beta_5 = 3 \times 10^{-6}$, for all other nodes $\beta_i = 10^{-6}$, and for all links $\alpha_{ij} = 10^{-4}$ (see Eq. 5.6). First we look at Fig. 5.6-a which displays the weight assignment for mT. We notice mT has not considered the node failure probabilities, and link weights are equal to what we saw before for mT method (Fig. 5.1-b). In addition, by looking at Fig. 5.6-c we observe that both nodes 4 and 5 are Most Critical Nodes (MCN) of the network for mT, and there is a symmetry for these nodes in this case. Now comparing mETN in Fig. 5.6-b with mT, we observe that METN has given less weights to the links incident to node 5 due to its unreliability, and node 5 is not MCN anymore (Fig. 5.6-c). Comparison of the other parameters of the two methods in Fig. 5.6-c shows that mETN has a better expected value for $\hat{\tau}^{(i)}$ while mT has a better $\hat{\tau}$, which is a natural result of the

objective functions for each optimization problem. However, $\hat{\tau}^{(5)}$ is lower (better) for mETN that shows this method is more robust against failure of the most unreliable node of the network, i.e node 5.

5.3.3 Optimizing λ_2

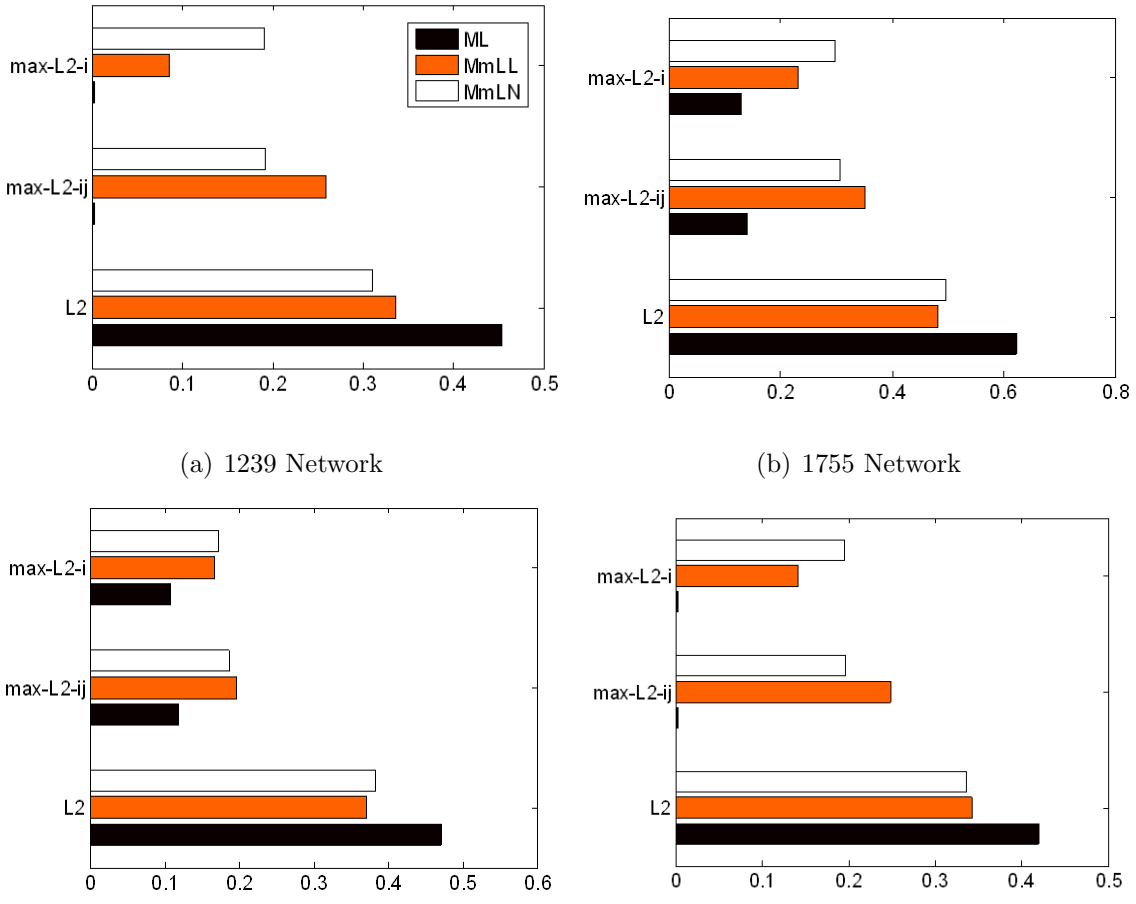
	ML			mT		
	λ_2	$\hat{\tau}$	$\min(\lambda_2^{(ij)})$	λ_2	$\hat{\tau}$	$\min(\lambda_2^{(ij)})$
Fish	1.7678	0.6031	0.5225	1.5057	0.5600	0.6130
Abilene	0.4713	1.5447	0.1189	0.3965	1.2936	0.1677
1755	0.6228	1.4041	0.1390	0.4519	1.1013	0.2832
3967	0.4192	1.9581	0.0	0.3060	1.4609	0.1926
1239	0.4531	1.8429	0.0	0.2799	1.0706	0.1059

Figure 5.7: Comparison of ML with mT for Various Networks

In this part we maximize λ_2 for the networks and compare them with previous optimized graphs. Fig. 5.7 shows a brief comparison between ML (Maximize λ_2) against mT for all the networks under investigation. Comparison of λ_2 and $\hat{\tau}$ for each network in two methods shows that λ_2 is better (larger) for ML while $\hat{\tau}$ is better (smaller) for mT. This is what we naturally expected due to the objective function of each optimization problem. However, comparison of $min(\lambda_2^{(ij)})$ for ML and mT shows that even though ML maximizes λ_2 for the network, ML does not have a good performance in case of failures, and this parameter is better for mT in all of the networks. In extreme cases for 1239 and 3967 networks, $min(\lambda_2^{(ij)}) = 0$ for ML, and it means that there is a link that its failure make the network disconnected. This motivate us to go for maximizing $min(\lambda_2^{(ij)})$ and $min(\lambda_2^{(i)})$ in the next part, i.e. MmLL, and MmLN methods, as ML does not seem to be a robust weight assignment

in case of failures. Here we note that the reason that we may go after algebraic connectivity instead of network criticality, in spite of its sub-optimality, can be the complexity. According to eq. (3.8), calculation of $\hat{\tau}$ is more computationally intensive than λ_2 since we should know all the Laplacian eigenvalues to calculate $\hat{\tau}$, while λ_2 is simply the second smallest one.

5.3.4 Optimizing λ_2-Based Vulnerability Metrics



(a) 1239 Network

(b) 1755 Network

(c) Abilene Network

(d) 3967 Network

Figure 5.8: λ_2-Based Parameters for ML, MmLL, and MmLN

In this part we maximize the λ_2-based vulnerability metrics, i.e. $min(\lambda_2^{(ij)})$ in MmLL and $min(\lambda_2^{(i)})$ in MmLN, and compare the results with ML. Fig. 5.8 shows the comparison of the parameters of Abilene and Rocketfuel networks for optimized weight assignments ML, MmLL, and MmLN. What we observe here is that vulnerability metrics $min(\lambda_2^{(ij)})$ and $min(\lambda_2^{(i)})$ for MmLL and MmLN have a tangible improvement over ML for all of the networks. Specially for 1239 and 3967, we observe that ML has vulnerability metrics equal to zero which means failure of some links or nodes makes the ML network disconnected; however; this problem does not exist for MmLL and MmLN anymore. We also notice that this improvement is obtained by a decrease in λ_2 for MmLL and MmLN compared to ML which is naturally expected due to the objective functions of these optimizations.

Chapter 6

Conclusions

Design and management of communication networks in today's world is a challenging task. While the initial design of a communication network should be able to respond to the initial requirements of the network and match different restrictions such as physical and financial constraints, the management system should be able to adapt itself to the varying needs and applications, and render the desired quality of service for each client according to its service level agreement and provider's policies. In addition, creation and evolution of virtual networks within the existing physical infrastructure is an ongoing responsibility of the management system. All these tasks should be performed with minimum human intervention since it is the main origin of errors and imposes a high expense on the system. In response to this need, we have provided algorithmic solutions in the direction of realizing an autonomic management system for a communication network with emphasis on robustness in design and management. The management system under consideration designs a robust network by optimizing some robustness metrics and continuously optimizes the robustness metrics to keep the network in the most robust state over time.

6.1 Contributions

The main contribution of this thesis is providing algorithms and optimizations to implement different blocks of the autonomic management system under consideration. We present here a brief description of different parts of our proposed solution and explain the contributions of each part.

6.1.1 Robust Topology Design

We tackled the problem of robust topology design as an important aspect of the slow control loop of our proposed autonomic management system. We investigated the behavior of different graph theoretic measures on some well-known graph topologies including structured graphs as well as different classes of complex networks including ER random, small-world, and scale-free networks [8]. More specifically, we examined the effect of network size on network criticality, algebraic connectivity, average degree, and average node betweenness. We observed that in both structured and complex networks, network criticality reveals more information about the network structure compared to algebraic connectivity. However, we found that there is no unique graph metric to satisfy both connectivity and robustness objectives while keeping a reasonable complexity. Each metric captures some attributes of the graph. It turns out that in order to design or simplify a network, we need to study the effect of all these graph metrics and choose the best network topology according to the requirements of the problem at hand.

In addition, we investigated the robustness properties of various data center topologies as an application of our approach to the topology design. We examined the robustness behavior of four different data center topologies, including Clos, Fat-Tree, Hyper-Cube, and Star using the network criticality as the robustness metric and con-

sidering that each topology should provide non-blocking connectivity between all the servers. Considering the fact that network criticality is a strictly convex function of link weights, we proposed a convex optimization problem of minimizing the network criticality under some constraints on the weight matrix. We also found a semi-definite programming representation of this problem which permits us to use available literature on semi-definite programming to solve the optimization problem and find the optimal weights.

6.1.2 Robust Survivable Routing Algorithm

We proposed Robust Survivable Routing (RSR) algorithm [10] as a method to implement the fast loop of our proposed autonomic management system with emphasis on realizing self-optimizing and self-healing attributes of a self-managing system. The main contribution of this method is to present the routing strategy to create the paths both in primary and backup selection. RSR guarantees 100% single-link-failure recovery as a path-based survivable routing method. The main idea is to quantify each path with a value to quantify its sensitivity to the incremental changes in external traffic and topology by evaluating the variations in network criticality of the network. The path with best robustness (path that causes minimum change in total network criticality) is chosen as primary (secondary) path.

We assigned a working path and a link-disjoint backup path to each demand using WRW-PCR algorithm [59], and combined the back-up paths for different active demands using Shared Backup Path Protection (SBPP) techniques. We implemented two methods of No Share (NS) and Full Share (FS) to combine backup resources. We evaluated RSR through extensive simulations and in different scenarios. We conducted experiments in both static and dynamic cases of incoming traffic, and

showed that our algorithm leads to a remarkable lower blocking ratio compared to the methods based on SP and WSP. We also compared NS and FS in both static and dynamic scenarios and showed that the amount of backup resources gets reduced considerably in FS compared to NS.

6.1.3 Design for Minimum Vulnerability

We proposed robust network design with emphasis on minimum vulnerability to single node and link failures [11, 12, 13]. In vulnerability analysis, our focus was to study the behavior of a communication network in case of node/link failures, and optimize the weights of the network to design it to have the best performance when failures happen. For this purpose, we proposed new vulnerability metrics based on the variations of network criticality or algebraic connectivity of the network when a single node/link failure happens. We defined two sets of vulnerability metrics based on two basic metrics, network criticality and algebraic connectivity. The first metric set captures the worst or the expected value of the network criticality after a single link or node failure. We showed that these metrics are convex functions of link weights. The second set quantifies the worst or the expected value of the network algebraic connectivity after a single link/node failure. We proved that the metrics of the second set are concave functions of link weights. We proposed convex optimization problems to optimize each vulnerability metric, and converted the optimization problems to SDP formulation to have a faster implementation for large networks.

We compared and contrasted the vulnerability metrics and their related optimization problems by applying them on some well-known networks including Abilene and ISP topologies from Rocketfuel dataset. We found optimum values for the metrics of these networks and discussed the application of these optimizations in robust net-

work planning. We observed that the robustness metrics of the network after failures can be improved a lot by sacrificing a little bit of network robustness in the normal operation by optimizing $\hat{\tau}$-based vulnerability metrics. We also applied traffic to the networks derived from different optimizations, and showed how optimizing the vulnerability metrics improves the routing performance in case of link or node failures. In addition, we showed through some examples how the probabilistic approach to failures can give different roles to the more reliable nodes or links. We also found that designing the network with optimizing algebraic connectivity did not lead to a robust weight assignment, but the robustness properties improved considerably when we optimized λ_2-based vulnerability metrics.

6.2 Future Work

We have covered many aspects of our proposed autonomic management system for a communication network in this dissertation. In terms of future work, there are many possible research avenues to extend our work. The first area for the future work is to extend the proposed algorithms for working and backup path selection to more general algorithms for assigning multipath or virtual network to the demands in order to improve the performance of the whole system. We have started this by introducing joint optimizations for resources and routes at the same time in [14]. In this work we have introduced algorithms to plan a network for a specific traffic matrix and route the traffic inside the network at the same time. The RSR algorithm can also be extended to cover multiple failures such as dual failures [107] and more.

Our work on vulnerability analysis can be extended in several possible interesting ways since our approach to vulnerability is a systematic approach. The first extension is defining similar vulnerability metrics to monitor the network behavior when more

than one failure is possible in the communication network. The other approach can be defining the vulnerability metrics based on other robustness metrics than those we have done in this thesis. For example, traffic aware network criticality [108] can be a proper basic robustness metric in a communication network. The other research avenue would be to extend the results of deterministic link (node) failure scenario to a set of correlated link (node) failures such as multiple failures that happen in the network as a result of geographical disasters. These kinds of failures can have a specific pattern, e.g., all of the failures can be on a line or inside a circle [79].

While we have examined robust flow assignment and capacity planning in our works, a natural generalization of our work is considering network resources other than the link capacity in dividing the communication network to virtual networks. These resources can be computing power or the memory of a network node. This problem is studied as the virtual network embedding problem [109] in the literature. The similar problem is how to assign the jobs to the network nodes or data center servers based on the available resources on them in order to maximize the network utilization or minimize the power consumption. To tackle these problems one can start with examining the dual of the network in which each link of the initial network becomes a node of the dual network [110]. The job assignment problem, studied in [111, 112] in which the goal is to balance the load among a network of processors, is another good start point.

Bibliography

[1] M. Nicolett, K. Brittain, and P. Adams. Enterprise Management ROI and Cost Reduction in 2003. Technical report, Gartner, November 2002.

[2] IBM. An Architectural Blueprint For Autonomic Computing, April 2003.

[3] IBM. Practical Autonomic Computing: Roadmap to Self Managing Technology, Jan 2006.

[4] A. Tizghadam. *Autonomic core Network Management System.* Phd thesis, University of Toronto, School of Electrical and Computer Engineering, April 2009.

[5] M. Fiedler. Algebraic Connectivity of Graphs. *Czechoslovak Math. Journal*, 23(98):298–305, 1973.

[6] M. Fiedler. Absolute Algebraic Connectivity of Trees. *Linear and Multilinear Algebra*, 26:85–106, 1990.

[7] M. Fiedler. Some Minmax Problems for Graphs. *Discrete Math.*, 121:65–74, 1993.

[8] A. Bigdeli, A. Tizghadam, and A Leon-Garcia. Comparison of network criticality, algebraic connectivity, and other graph metrics. In *First Workshop on Simplifying Complex Networks: SIMPLEX*, Venice,Italy, July 2009.

[9] M. S. Kodialam and T. V. Lakshman. Dynamic Routing of Bandwidth Guaranteed Tunnels with Restoration. In *INFOCOM*, pages 1217–1225. IEEE, 2006.

[10] A. Bigdeli, A. Tizghadam, and A Leon-Garcia. Survivable routing using path criticality. In *International Conference on Computing, Networking and Communications*, Maui, Hawaii, USA, January 2012.

[11] A. Bigdeli, A. Tizghadam, and A. Leon-Garcia. On Capacity Planning for Minimum Vulnerability. In *8th International Workshop on the Design of Reliable Communication Networks (DRCN)*, Krakow, Poland, 2011.

[12] A. Bigdeli, A. Tizghadam, and A. Leon-Garcia. Planning fo Maximum Robustness and Minimum Vulnerability. *Submitted to IEEE Transactions on Network and Service Management*, 2011.

[13] A. Tizghadam, A. Bigdeli, and A. Leon-Garcia. k-Robust Network Design Using Resistance Distance: Case of RocketFuel and Power Grids. *Infocom Workshop on Network Science for Communications (NetSciCom)*, April 2011.

[14] A. Tizghadam, A. Bigdeli, H. Naser, and A. Leon-Garcia. Joint Optimization of Resources and Routes for Minimum Resistance: From Communication Networks to Power Grids. *To appear in Handbook of Optimization in Complex Networks*, 2011.

[15] A. H. Dekker and B. D. Colbert. Network Robustness and Graph Topology. *Australasian Computer Science Conference*, 26:359–368, Jan. 2004.

[16] A. H. Dekker and B. Colberet. The Symmetry Ratio of a Network. In *Australasian symposium on Theory of computing*, volume 41, pages 13–20, Newcastle, Australia, 2005. ACM International Conference Proceeding Series.

[17] R. Zhang-Shen and N. McKeown. Designing a Predictable Internet Backbone with Valiant Load-Balancing. In *Thirteenth International Workshop on Quality of Service (IWQoS)*, Passau, Germany, June 2005.

[18] L. Valiant and G. Brebner. Universal Schemes for Parallel Communication. In *13th Annual Symposium on Theory of Computing*, May 1981.

[19] Ali Tizghadam and Alberto Leon-Garcia. On Robust Network Planning. In *7th International Workshop on the Design of Reliable Communication Networks (DRCN)*, pages 139–146, Washington DC, USA, 2009.

[20] P. Van Mieghem and F. A. Kuipers. Concepts of Exact QoS Routing Algorithms. *IEEE/ACM TRANSACTIONS ON NETWORKING*, 12(5):851–864, October 2004.

[21] J. M. Jaffe. Algorithms for Finding Paths with Multiple Constraints. *Networks*, 14:95–116, 1984.

[22] A. Tizghadam and A. Leon-Garcia. Autonomic Traffic Engineering for Network Robustness. *IEEE Journal of Selected Areas in Communications (J-SAC)*, 28(1):39–50, 2010.

[23] D. J. Klein and M. Randic. Resistance Distance. *Journal of Mathematical Chemistry*, 12(1):81–95, December 1993.

[24] Michael William Newman. *The Laplacian Spectrum of Graphs*. PhD thesis, Department of Mathematics, University of Manitoba, July 2000.

[25] Fan R. K. Chung. *Spectral Graph Theory*. CBMS Regional Conference Series On Mathematics, No. 92. American Mathematical society, 1997.

[26] R. Grone, R. Merris, and V. S. Sunder. The Laplacian Spectrum of a Graph. *SIAM Journal, Matrix Analysis and Applications*, 11(2):218–238, April 1990.

[27] Jiayuan Huang. A Combinatorial View of Graph Laplacians. Technical Report 144, Max Planck Institute for Biological Cybernetics, 2005.

[28] Dennis S. Bernstein. *Matrix Mathematics*. Prinston University Press, 2005.

[29] J. Sun, S. Boyd, L. Xiao, and P. Diaconis. The Fastest Mixing Markov Process On a Graph and a Connection to a Maximum Variance Unfolding Problem. *SIAM Review*, 48(4):681–699, 2006.

[30] R. Olfati-Saber and R. M. Murray. Consensus problems in networks of agents with switching topology and time-delays. *IEEE Trans. Autom.Control*, 49(9):1520–1533, Sep. 2004.

[31] J.A. Fax and R.M. Murray. Information flow and cooperative control of vehicle formations. *IEEE Trans. Autom.Control*, 49(9):1465–1476, Sep. 2004.

[32] A. Jadbabaie, J. Lin, and A. S. Morse. Coordination of Groups of Mobile Autonomous Agents Using Nearest Neighbor Rules. *SIAM Review*, 48(9):988–1001, 2003.

[33] Y. Kim and M. Mesbahi. On maximizing the second smallest eigenvalue of state-dependent graph Laplacian. *IEEE Trans. Autom. Control*, 51(1):116–120, Jan. 2006.

[34] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and Principles of Internet Traffic Engineering. Internet Engineering Task Force, RFC2732, May 2002.

[35] M. Rougan, M. Thorup, and Y. Zhang. Traffic Engineering with Estimated Traffic Matrices. In *Internet Measurement Conference*, pages 248–258, New York, NY, USA, 2003. ACM Press.

[36] M. Rougan, M. Thorup, and Y. Zhang. COPE: Traffic Engineering in Dynamic Networks. In *ACM SIGCOMM*, volume 36, pages 99–110, 2006.

[37] C. Zhang, Y. Liu, W. Gong, J. Kurose, R. Moll, and D. Towsley. On Optimal Routing with Multiple Traffic Matrices. In *IEEE INFOCOM*, 2005.

[38] B. Fortz, J. Rexford, and M. Thorup. Traffic Engineering with Traditional IP Routing Protocols. *IEEE Communications Magazine*, (33), October 2002.

[39] E. Rosen, A. Viswanathan, and R. Callon. Multiprotocol Label Switching Architecture. *RFC 3031 IETF*, January 2001.

[40] K. Kar, M. Kodialam, and T. V. Lakshman. Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Applications. *IEEE Journal on Selected Areas in Communications*, 18(12):2566–2579, Dec. 2000.

[41] S. Suri, M. Waldvogel, and P. R. Warkhede. *Profile-Based Routing: A New Framework for MPLS Traffic Engineering*, chapter Quality of Future Internet Services. Lecture Notes in Computer Science. Springle Verlag, September 2001.

[42] A. Elwalid, C. Jin, S. Low, , and I. Widjaja. MATE: MPLS Adaptive Traffic Engineering. In *Proceedings of IEEE INFOCOM'01*, pages 1300–1309, 2001.

[43] S. Yilmaz and I. Matta. On the Scalability-Performance Tradeoffs in MPLS and IP Routing. In *Proceedings of SPIE ITCOM*, May 2002.

[44] R. G. Gallager. A Minimum Delay Routing Algorithm Using Distributed Computation. *IEEE Transactions on Communications*, 25:73–84, January 1977.

[45] Z. Wang and J. Crowcroft. Quality of Service Routing for Supporting Multimedia Applications. *IEEE Journal in Selected Areas in Communications*, 14(7):1228–1234, 1996.

[46] I. Averbakh. Minmax Regret Linear Resource Allocation Problems. *Operation Research Letters*, 32:174–180, 2004.

[47] A. Ben-Tal, A. Nemirovski, L. Ghaoui, and F. Rendl. *Handbook on Semidefinite Programming*. Kluwer Academic Publishers, 2000.

[48] P. Kouvelis and G. Yu. *Robust Discrete Optimisation and its Applications*. Kluwer Academic Publishers, 1997.

[49] D. Bertsimas and M. Sim. The Price of Robustness. *Operation Research*, 52(1):35–53, January-February 2004.

[50] A. Ben-Tal and A. Nemirovski. Lectures on Modern Convex Optimization: Analysis, Algorithms; Engineering Applications. *SIAM-MPS Series in Optimization*, 2000.

[51] H. Yaman, O. E. Karasan, and M. C. Pinar. The Robust Minimum Spanning Tree Problem with Interval Data. *Operations Research Letters*, 29:31–40, 2001.

[52] D. Applegate and E. Cohen. Making Intra-Domain Routing Robust to Changing and Uncertain Traffic Demands: Understanding Fundamental Tradeoffs. In *SIGCOMM*, pages 313–324, 2003.

[53] D. Applegate, L. Breslau, and E. Cohen. Coping with Network Failures: Routing Strategies for Optimal Demand Oblivious Restoration. In *SIGMETRICS*, pages 270–281, 2004.

[54] Y. Azar, E. Choen, A. Fiat, H. Kaplan, and H. Racke. Optimal Oblivious Routing in Polynomial Time. In *ACM Symposium in Parallelism in Algorithms and Architectures (SPAA)*, 2003.

[55] N. Bansal, A. Blum, S. Chawla, and A. Meyerson. Online Oblivious Routing. In *ACM SIGCOMM*, volume 36, pages 99–110, 2006.

[56] M. S. Kodialam, T. V. Lakshman, and S. Sengupta. Maximum Throughput Routing of Traffic in the Hose Model. In *INFOCOM*, pages 902–911. IEEE, 2009.

[57] Y. Li, J. Harms, and R. Holte. A Simple Method for Balancing Network Utilization and Quality of Routing. In *ICCCN*, San Diego,CA, 2005.

[58] A. Tizghadam and A. Leon-Garcia. A Robust Routing Plan to Optimize Throughput in Core Networks. *ITC20, Elsvier*, pages 117–128, 2007.

[59] A. Tizghadam and A. Leon-Garcia. Betweenness Centrality and Resistance Distance in Communication Networks. *IEEE Network Magazine*, 24(6):10–16, 2010.

[60] W.D. Grover. *Mesh-based Survivable Transport Networks: Options and Strategies for Optical, MPLS, SONET Networking.* Prentice Hall, 2004.

[61] D. Awduche, L. Berger, D. Gan, T. Li, V. Srinivasan, G. Swallow, and D. Gan. RSVP-TE: Extensions to RSVP for LSP Tunnels. RFC3209, December 2001.

[62] Y. Liu, D. Tipper, and P. Siripongwutikorn. Approximating Optimal Spare Capacity Allocation by Successive Survivable Routing. In *INFOCOM*, pages 699–708. IEEE, 2001.

[63] G. Li, D. Wang, C. Kalmanek, and R. Doverspike. Efficient Distributed Path Selection for Shared Restoration Connections. In *INFOCOM*, pages 140–149. IEEE, 2002.

[64] C. Qiao and D. Xu. Distributed Partial Information Management (DPIM) Schemes for Survivable Networks - Part I. In *INFOCOM*, pages 302–311. IEEE, 2002.

[65] R. Albert and A.-L. Barabasi. Statistical Mechanics of Complex Networks. *Reviews of Modern Physics*, 74:47, 2002.

[66] S. Boccalettia, V. Latorab, Y. Morenod, M. Chavezf, and D.U. Hwanga. Complex networks: Structure and dynamics. *Physics Reports*, 424:175 308, 2006.

[67] P.V. Mieghem. *Graph Spectra for Complex Networks*. Cambridge University Press, 2010.

[68] M.E.J. Newman. *Networks: An Introduction*. Oxford University Press, 2010.

[69] Bollabas. *Random Graphs*. Cambridge University Press, 2001.

[70] Michalis Faloutsos, Petros Faloutsos, and Christos Faloutsos. On Power-Law Relationships of the Internet Topology. In *Proceeding of ACM SIGCOMM*, pages 251–262, 1999.

[71] D.J. Watts and S.H. Strogatz. Collective Dynamics of Small-World Networks. *Nature*, (393):440–442, 1999.

[72] A. Barabasi and R. Albert. Emergence of scaling in random networks. *Science*, 286:509–512, 1999.

[73] M.E.J. Newman. The structure and function of complex networks. *SIAM Review*, pages 167–256, 2002.

[74] A. Jamakovic and S. Uhlig. On the Relationship between the Algebraic Connectivity and Graph Robustness to Node and Link Failures. In *3rd EuroNGI Conference, On Next Generation Internet Networks*, pages 96–102, Trondheim, May 2007.

[75] A. Jamakovic and P. Van Mieghem. On the Robustness of Complex Networks by Using the Algebraic Connectivity. In *NETWORKING 2008 Ad Hoc and Sensor Networks, Wireless Networks, Next Generation Internet*, pages 183–194. Springer Berlin / Heidelberg, May 2008.

[76] A. Jamakovic and S. Uhlig. Influence of the network structure on robustness. In *ICON*, pages 278–283. IEEE, 2007.

[77] J.S. Foster Jr, M.E. Gjelde, W.R. Graham, R.J. Hermann, H. M. Kluepfel, G. L. Lawson, G.K. Soper, L.L. Wood Jr, and J.B. Woodardand. Report of the Commission to Assess the Threat to the United States from Electromagnetic Pulse (EMP) Attack . April 2004.

[78] J. Borland. Analyzing the internet collapse. http://www.technologyreview.com /Infotech/20152/?a=f, Feb. 2008.

[79] S. Neumayer, G. Zussman, R. Cohen, and E. Modiano. Assessing the Vulnerability of the Fiber Infrastructure to Disasters. In *INFOCOM*, pages 1566 – 1574. IEEE, 2009.

[80] R. Criado, J. Flores, B. Hernandez-Bermejo, J. Pello, and M. Romance. Effective Measurement of Network Vulnerability Under Random and Intentional Attacks. *Journal of Mathematical Modelling and Algorithms*, 4:307–316, 2005.

[81] P. Holme, B. J. Kim, C. N. Yoon, and S. K. Han. Attack vulnerability of complex networks. *Phys. Rev. E65, 056109*, 2002.

[82] C. A. Phillips. The network inhibition problem. In *STOC'93*. ACM, 1993.

[83] K. Appleby, S. B. Calo, J. R. Giles, and K. W. Lee. Policy-Based Automated Provisioning. *IBM Systems Journal*, 43(1):121–135, 2004.

[84] A. Leon-Garcia and L. Mason. Virtual Network Resource Management for Next-Generation Networks. *IEEE Communications Magazine*, 41(7):102–109, July 2003.

[85] A. Jun and et al. Virtual Network Resources Management: A Divide-and-Conquer Approach for the Control of Future Networks. *IEEE Globecom, Sydney Australia*, 2:1065–1070, November 1998.

[86] M. Newman. A Measure of Betweenness Centrality Based on Random Walks. *arXiv cond-mat/0309045.*, 2003.

[87] A. Tizghadam and A. Leon-Garcia. On Robust Traffic Engineering in Core Networks. In *IEEE GLOBECOM*, December 2008.

[88] A. Ghosh, S. Boyd, and A. Saberi. Minimizing Effective Resistance of a Graph. *SIAM Review, problems and techniques section*, 50(1):37–66, February 2008.

[89] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D.A. Maltz, P. Patel, and S. Sengupta. VL2: a scalable and flexible data center network. In *ACM SIGCOMM*, pages 51–62, 2009.

[90] R.N. Mysore, A. Pamboris, N. Farington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat. PortLand: A Scalable Fault-Tolerant Layer 2 Data Center Network Fabric. In *ACM SIGCOMM*, pages 39–50, 2009.

[91] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu. BCube: a high performance, server-centric network architecture for modular data centers. In *ACM SIGCOMM*, pages 51–62, 2009.

[92] W.J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers, 2004.

[93] Cisco. Data Center: Load Balancing Data Center Services SRND. *Cisco Systems*, 2004.

[94] R.H. Katz. Tech Titans Building Boom. *IEEE Spectrum*, 46(2):40–54, 2009.

[95] G. Shen and W.D. Grover. Survey and performance comparison of dynamic provisioning methods for optical shared backup path protection. In *2nd International Conference on Broadband Networks*, volume 2, pages 1310 – 1319, 2005.

[96] P. Ho and H. T. Mouftah. On Achieving Optimal Survivable Routing for Shared Protection in Survivable Next-Generation Internet. *IEEE Transactions on Reliability*, 53(2), June 2004.

[97] R. Guerin, A. Orda, and D. Williams. QoS Routing Mechanisms and OSPF Extensions. In *Proceedings of IEEE Globecom*, 1997.

[98] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[99] C. Helmberg. Semidefinite Programming for Combinatorial Optimization. *ZIB-Report ZR-00-34, Konrad-Zuse-Zentrum Berlin*, October 2000.

[100] H. Wolkowicz, R. Saigal, and L. Vandenberghe, editors. *Handbook on Semidefinite Programming*. Kluwer, 2000.

[101] M. Grant and S. Boyd. CVX: Matlab Software for Disciplined Convex Programming (Web Page and Software). *http : //stanford.edu/ boyd/cvx*, Sep. 2008.

[102] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. Inferring link weights using end-to-end measurements. In *2nd Internet Measurement Workshop*. ACM, 2002.

[103] http://abilene.internet2.edu/.

[104] M. Roughan, A. Greenberg, C. Kalmanek, M. Rumsewicz, J. Yates, and Y. Zhang. Experience in measuring backbone traffic variability: models, metrics, measurements, and meaning. In *Proceedings of the 2nd Internet Measurement Workshop, ACM*, 2002.

[105] G. Leduca, H. Abrahamssone, S. Balona, S. Besslerb, M. DArienzoh, O. Delcourta, J. Domingo-Pascuald, S. Cerav-Erbasg, I. Gojmeracb, A. Pescaph X. Masipd, B. Quoitinf, S.P. Romanoh, E. Salvadoric, F. Skivea, H.T. Tranb, S. Uhligf, and H. Umitg. An Open Source Traffic Engineering Toolbox. *Computer Communications*, 29(5):593–610, 2006.

[106] B. Fortz and M. Thorup. Increasing Internet Capacity Using Local Search. *Computational Optimization and Applications*, 29:13–84, 2004.

[107] V. Y. Liu and D. Tipper. Spare Capacity Allocation using Shared Backup Path Protection for Dual Link Failures. In *8th International Workshop on the Design of Reliable Communication Networks (DRCN)*, Krakow, Poland, 2011.

[108] A. Tizghadam and A. Leon-Garcia. On Traffic-Aware Betweenness and Network Criticality. *Infocom Workshop on Network Science for Communications (NetSciCom)*, March 2010.

[109] M. Yu, Y. Yi, J. Rexford, and M. Chiang. Rethinking virtual network embedding: Substrate support for path splitting and migration. *ACM SIGCOMM Computer Communication Review*, 38(2), 2008.

[110] S. Winter. Route Specification with a Linear Dual Graph. In *Advances in Spatial Data Handling: Proc. 10th Int. Symp. Spatial Data Handling (SDH 2002)*, pages 329–338. Springer-Verlag, 2002.

[111] J. E. Boillat. Load Balancing and Poisson Equation in a Graph. *Concurrency: Practice and Experience*, 2(4):289–313, December 1990.

[112] J. E. Boillat. Load Balancing Algorithms Based on Gradient Methods and their Analysis through Algebraic Graph Theory. *Journal of Parallel and Distributed Computing*, 68(2):209–220, February 2008.