April 2018

Multi-Objective Resource Provisioning in Network Function Virtualization Infrastructures

Diogo Oliveira
University of South Florida, diogoo@mail.usf.edu

Multi-Objective Resource Provisioning in Network Function Virtualization Infrastructures

by

Diogo Oliveira

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Department of Electrical Engineering
College of Engineering
University of South Florida

Co-Major Professor: Nasir Ghani, Ph.D.
Co-Major Professor: Jorge Crichigno, Ph.D.
Richard Gitlin, Sc.D.
Stephen Saddow, Ph.D.
Kaiqi Xiong, Ph.D.

Date of Approval:
March 20, 2018

Keywords: Virtual Network Function,
Placement Problem, Risk-Aware

Dedication

To my wife, Elisa Oliveira, for all the support, braveness, understanding and love.

To my daughters, Ana Clara and Emily Oliveira, for the unconditional love.

Acknowledgments

I would like to acknowledge and dedicate my gratitude to the following people who made the completion of this work possible: Most specially, my wife, Elisa, and my daughters, Ana Clara and Emily, for their unconditional love and support.

My supervisor, Dr. Nasir Ghani, for his continuous help and advisement through my Ph.D. study.

My former co-supervisor and committee member, Dr. Stephen Saddow, for believing in me and giving me the chance to join the University of South Florida as a visiting scholar.

My co-supervisor, Dr. Jorge Crichigno, for the contribution and continuous assessment through my research.

My dissertation committee members, Dr. Richard Gitlin and Dr. Kaiqi Xiong, for their valuable suggestions and advice on this dissertation work.

My former and current research colleagues, Dr. Hao Bai, Dr. Mahsa Pourval, M.E. Amanda Gonzalez, M.E. Farooq Shaikh, M.E. Andrea Wright, Dr. Mohammed Jasim, M.E. Nazlee Siasi and Mr. Akhil Arra, for their collaboration.

My former research colleagues, Dr. Tom Lehman and Dr. Xi Yang for the contribution during our collaborative research.

God, for giving me strength, capability and perseverance.

My father, for being such an inspiration as a person and professional.

My mother, my sister, and my in-laws for being there for myself and my family.

Table of Contents

List of Tables

List of Figures

Abstract

Network function virtualization (NFV) and software-defined networking (SDN) are two recent networking paradigms that strive to increase manageability, scalability, programmability and dynamism. The former decouples network functions and hosting devices, while the latter decouples the data and control planes. As more and more service providers adopt these new paradigms, there is a growing need to address multi-failure conditions, particularly those arising from large-scale disaster events. Overall, addressing the *virtual network function* (VNF) placement and routing problem is crucial to deploy NFV survivability. In particular, many studies have inspected non-survivable VNF provisioning, however no known work have proposed survivable/resilient solutions for multi-failure scenarios.

In light of the above, this work proposes and deploys a survivable multi-objective provisioning solution for NFV infrastructures. Overall, this study initially proposes multi-objective solutions to efficiently solve the VNF mapping/placement and routing problem. In particular, a *integer linear programming* (ILP) optimization and a greedy heuristic methods try to maximize the requests acceptance rate while minimizing costs and implementing *traffic engineering* (TE) load-balancing. Next, these schemes are expanded to perform "risk-aware" virtual function mapping and traffic routing in order to improve the reliability of user

services. Furthermore, additionally to the ILP optimization and greedy heuristic schemes, a metaheuristic *genetic algorithm* (GA) is also introduced, which is more suitable for large-scale networks. Overall, these solutions are then tested in idealistic and realistic stressor scenarios in order to evaluate their performance, accuracy and reliability.

Chapter 1 Introduction

This dissertation presents a multi-objective resources provisioning in network function virtualization infrastructures, which focuses on virtual network function placement and routing with a focus on survivability under large-scale disaster conditions. This initial chapter starts by introducing some of the key developments in this space and then presents the key motivations for the work. The main contributions of the research are then presented in a high-level manner along with an overview of the remainder of the thesis.

1.1 Background Overview

The past three decades have seen rapid technological achievements in the networking and information technology (IT) space. These developments have occurred not only in computational power, but also in terms of data rate transmission and storage as well. In turn, these advances have led to much lower acquisition, deployment and maintenance costs. These improvements have led to the adoption of large-scale datacenters interconnection, and the broader emergence of cloud-computing paradigms. However, as these trends have unfolded, a host of management and orchestration ossification challenges have also arisen.

Now traditional physical network devices, e.g., such as switches and routers, have operated with their own specialized internal and standalone control plane configuration sys-

tems; i.e., vendor-proprietary. These setups led to increase management complexity due to individual configuration requirements and a high degree of vendor dependency. As a result, many carriers began to find it very difficult to build and offer cost-effective servers using legacy devices. In response, various organizations initiated efforts to simplify network node designs by decoupling the control and data planes, i.e., *software-defined networking* (SDN). The main premise behind SDN outsource the control plane to a concentrator, namely SDN controller. The controller is responsible for receiving the flow rules via northbound interface and pushing such rules into the proper network nodes via southbound interface. Overall, SDN strives to reduce network ossification as well as reducing operational and capital expenses (OpEx and CapEx) by reducing management complexity and improving dynamism. Additionally, the adoption of SDN nodes leverages vendor-independency.

Now, although decoupling the control and data planes delivers the aforementioned advantages, traditional network services deployment still remains a complex and expensive under-taking. Moreover there is a high degree of vendor-dependency as traditional network services are managed and deployed by embedded vendor-proprietary systems. Therefore, in order to reduce network services management complexity and to improve (re)deployment capabilities, *network function virtualization* (NFV) paradigm has been evolved to support deployment of network functions on *commercial-of-the-shelf* (COTS) equipment as a software instances.

In general, typical client networking services can include a range of offerings such as firewalls, *deep packet inspection* (DPI) engines, intrusion detection/prevention systems, *network address translation* (NAT) boxes, etc. Now in terms of NFV, many of these services are typically composed of multiple *virtual network functions* (VNF). Specifically VNF can be deployed/implemented across multiple datacenter sites, and a datacenter site can host multiple VNFs. Additionally, many client services may impose a strict interdependent relationship between multiple atomic VNFs, i.e., *service function chaining* (SFC). For example, a firewall service can be comprised of two VNFs to filter packets and balance traffic (load balancer). Fig. Figure 1.1 also illustrates a generic set of three SFC demands, i.e., SFC1 (red) consists of VNFs 1 and 3, SFC2 consists of VNFs 1, 2 and 3 and SFC3 consists of VNFs 2 and 3. In an event of a failure of Datacenter D, services demanding VNF1 are steered to Datacenter D, which deploys a backup instance of VNF1.

Although it is not mandatory, in general most deployments use an embedded virtual network as the *virtual network function infrastructure* (VNFI) owing the advantages offered by this approach. In particular, it is possible to map and deploy VNFs in a substrate infrastructure comprised of nodes that support the VNF paradigm. However a virtual network guarantees a more flexible and manageable infrastructure. Specifically, Fig. Figure 1.2 shows a VNFI comprised of a virtual network embedded into a substrate/physical network, where the substrate nodes are COTS devices hosting virtual nodes that instantiate the VNFs illustrated in the NFV layer.

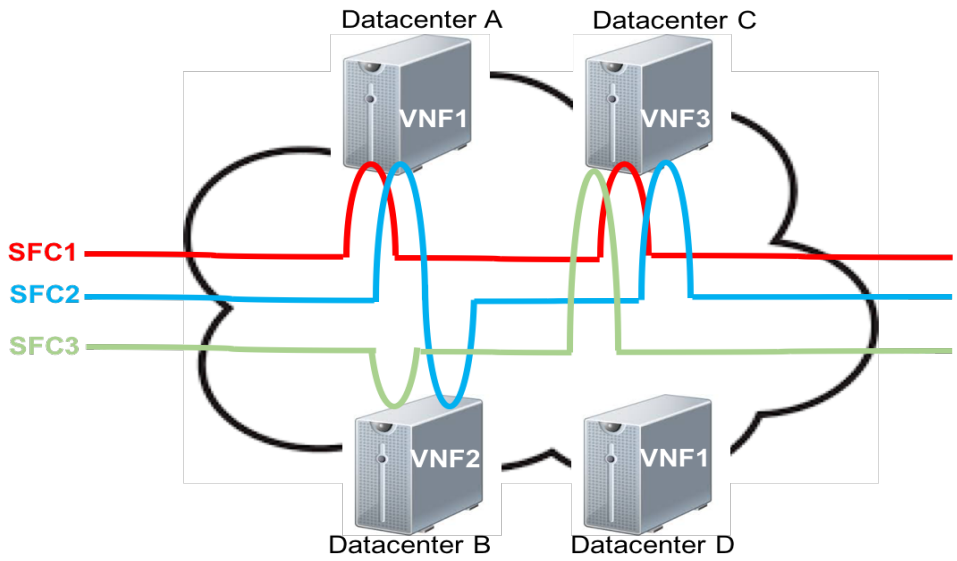


Figure 1.1: Service Function Chaining.

However as service providers show increasing interest in NFV, many further questions and challenges are starting to arise, i.e., such as management and orchestration, security and privacy, performance, and function placement, among others. In particular, the latter placement problem must consider a multiple set of client requests, where each request is comprised of a set of VNFs, a source, a destination and a minimum bandwidth rate. Hence, the service providers must efficiently "place" these VNFs across their datacenters to reduce deployment and routing costs, and also increase service performance and reliability.

Now given the immense interest and focus on network virtualization, the VNF placement problem has been well-studied in recent years. Specifically, researchers have proposed a host of schemes to minimize costs, increase revenues or increase reliability [BA01]-[DO02]. Additional studies have also looked at survivable VNF placement using backup VNF re-

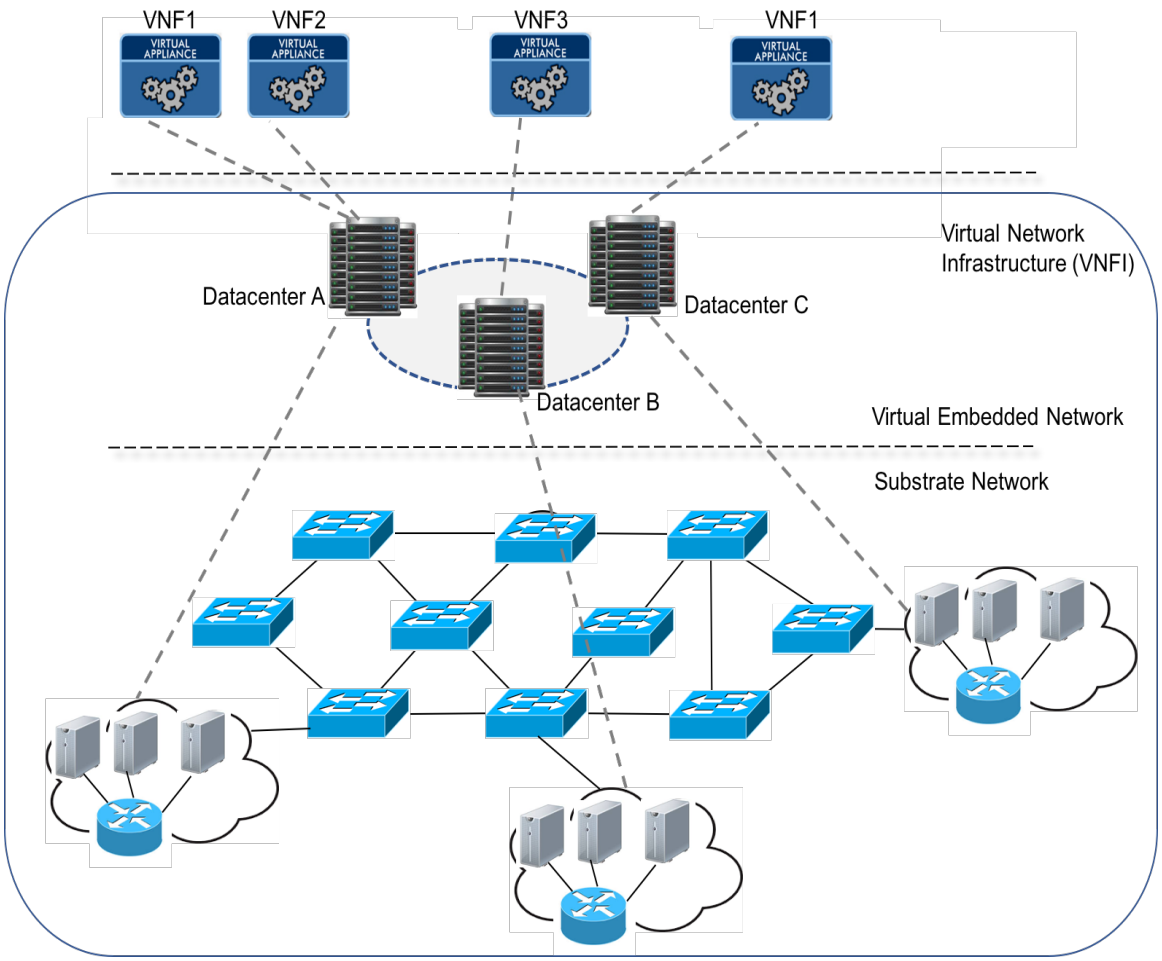


Figure 1.2: Network Function Virtualization overview.

source provisioning [MB01][ZY01][WD01]. In general, these studies have used a wide range of techniques, including optimization, graph-theoretic heuristics, meta-heuristics, and various approximation strategies, see [ZY01] for details.

1.2 Motivations

The current body of research work on virtual network functions placement and/or routing generally focuses on minimizing cost and improving performance. Namely, incoming

requests are provisioned based upon the current available resource levels in the network. However, as more providers start to deploy NFV-based services, related survivability issues are becoming increasingly important. Now traditionally, network operators have relied upon dedicated, expensive hardware systems to deliver a "carrier-grade" reliability for their high-end clients. Clearly, achieving similar performance over commodity-based servers (implementing NFV) is a much more challenging task as such systems have generally lower reliability levels. Hence operators have to derive effective strategies to achieve acceptable survivability support in emerging NFV service setups.

Although some efforts have addressed NFV survivability topics, these solutions mostly focus on single isolated system failures. As a result, the further impact of large-scale disaster events (multiple failures) on NFV-based services becomes a concern. These occurrences can include events such as natural disasters, malicious attacks, and cascading power outages.

Now the only known work on "risk-aware" disaster-based NFV provisioning is presented in [JF01], which outlines the probabilistic availability of a path based on the actual temporal availability of physical resources. Indeed there is a growing need to build more systematic multi-objective solutions to efficiently provision resources and directly incorporate the randomized nature of disaster events into the NFV placement process., i.e., "risk-aware" VNF placement on multi-failure scenarios. This request forms the key motivation for this dissertation research.

1.3 Problem Statement

This dissertation addresses the above challenges and develops a novel set of multi-objective resources provisioning in NFV infrastructures. Specifically, these techniques implement placement and routing strategies and also incorporate stochastic failure models to lower failure risk and improve VNF reliability. In addition, many existing VNF placement and/or routing schemes assume abundant datacenter resources to satisfy all client demands, therefore they only focus on minimizing cost, i.e., unconstrained resource levels. However, this assumption may not hold in heavy demand or in post-failure scenarios where resource scarcity will be high. As a result, this dissertation also looks at constrained VNF placement to minimize costs and maximize the overall VNF placement.

1.4 Proposed Work and Contributions

This dissertation studies the VNF placement problem within the context of disaster recovery, i.e., large failure events causing multiple correlated system failures. The key contributions of this effort include the following:

1) New *integer linear programming* (ILP) optimization models and improved greedy heuristics to minimize deployment and routing costs and maximizing the number of satisfied VNFs.

2) New "risk-aware" probabilistic VNF placement problem along with associated optimization model and greedy heuristics for risk mitigation.

3) Novel meta-heuristic strategies for VNF placement based upon genetic algorithms, including variants for regular cost minimization and risk mitigation.

The remainder of this dissertation is organized as follows. First Chapter 2 presents a detailed survey of existing VNFs placement techniques as well as some rated survivability NFV schemes. Next Chapter 3 details some new joint routing and placement schemes for NFV to conceptually maximize the number of satisfied requests and also reduce costs. Chapter 4 then extends this work by introducing novel "risk-aware" routing and VNF placement schemes assuming probabilistic multi-failure models, i.e., including ILP-based and greedy heuristic methods. Finally, Chapter 5 presents further genetic algorithm methodology for "risk-aware" provisioning, focusing on large-scale networks. Conclusions and directions for future work are then presented in Chapter 6 to conclude this dissertation.

Chapter 2 Background and Related Work

The overall area of NFV has received notable attention in recent years, specifically due to the contemporary nature of this paradigm and its various challenges. Along these lines, this chapter overviews some of the latest developments in this field and then introduces the VNF placement and routing problem. A range of associated provisioning schemes are then surveyed, including survivability-based methods. Open research challenges are then outlined to motivate the thesis research.

2.1 Network Function Virtualization Overview

The *European Telecommunications Standards Institute* (ETSI) selected seven major telecom operators to form the *Industry Specification Group* for NFV (ISG-NFV) in 2012. This community now exceeds 300 companies and published its initial specifications between late 2013 and 2014, termed as Release 1 [ETSI01]. Now given the relatively high complexity of the NFV paradigm, the ISG-NFV also defined several *working groups* (WG) to handle specific NFV specification areas:

- Interfaces and Architecture (IFA): Specifies the overall NFV architecture and its interfacing requirements to support interoperability

- Evolution and Ecosystem (EVE): Specifies the interface between NFV and other standards or technologies that may relate to NFV. As an example, this WG has defined various NFV-SDN interaction requirements and use cases.
- Solutions (SOL): Focuses on specifying and developing protocols to support NFV interoperability.
- Reliability (REL): Focuses on improving the reliability and availability of NFV systems.
- Security (SEC): Addresses security-related concerns for NFV, e.g., such as southbound, northbound and deployment security.
- Testing and Implementation (TST): Focuses on deploying NFV testcases in order to test and demonstrate the capabilities of this technology. Hence this WG is crucial for driving NFV adoption by operators.

Therefore it is critical to identify specific NFV topics being addressed by the WGs and identify the ones that are of higher interest to users [ETSI01]. Accordingly, those topics are briefly detailed here as:

- Architecture: Analyze and proposes improvements to the NFV architecture
- MANO: Focus on physical and virtual resources requirements and management, network functions correlation, lifecycle control, policies implementation and management
- Use Cases: Develop state-of-the-art NFV deployment cases
- Placement: Address the VNF location problem by analyzing and/or proposing novel VNF placement models

- Security: Address NFV-related security concerns and solutions
- Performance: Correlate efforts to target NFV performance evaluation

Now, many studies have addressed a wide range of NFV-related topics. For example, [BH01] surveys the state-of-the-art in NFV technologies, detailing its requirements, architectural framework, use cases, challenges and potential solutions strategies. Specifically, this work addresses performance issues, i.e., throughput limitation and latency, that may arise due to the instantiation of services in a software. In addition, *service function chaining* (SFC) concerns are also addressed the possibility of service failure from a single VNF outage or misplacement. Finally, this study also analyses security concerns involving the interface between the NFV and *network function virtualization infrastructure* (NFVI) layers, and highlights the impact of VNF mapping with regards to performance, security and availability.

As mentioned in Section 1.1, evene though the SDN and NFV paradigms are not interdependent, their combination is still very beneficial. As a result, some efforts have also addressed this integration, i.e., termed as *software defined network function virtualization* (SDNFV). For example, [YL01] details the close association between these two methodologies and highlights VNF placement traffic steering considerations. Potential future applications of SDNFV are also detailed.

Finally, a broader inspection of the NFV paradigm is also presented in [RM01]. Specifically this work details the NFV architecture and discusses a range of NFV topics,

i.e., MANO, energy efficiency, performance, resource allocation, security, privacy and trust. The importance of VNF placement here is also stressed.

2.2 VNF Placement

In general, VNF placement has a major impact on many of the detailed NFV topic areas, e.g., such as performance, security and management. As a result, various research studies have defined addressed this particular problem in a detailed manner. In particular, the service provider's network, whether physical or virtual, is typically modeled as a graph with a set of nodes and links. This infrastructure yields incoming client requests which are composed of source and destination nodes, a set of interconnecting VNFs and a requisite bandwidth requests. Furthermore, each requested VNF has a set of minimum physical requirements, e.g., processor, memory, storage, etc. Accordingly, the VNF placement and/or routing algorithms try to provision each request by assigning a path across a set of datacenters that instantiate (support) the desired set of VNFs. Overall, Figure 2.1 shows a VNF placement example for 2 service requests. Here, the virtual infrastructure layer features a virtual network with 10 nodes and 15 links. The values inside the nodes correspond to the amount of available physical datacenter resources, whereas the values next to the associated link correspond to the link capacity. Now, Request A requires a path from node 1 to node 8 crossing over datacenters supporting functions *a, b, c* with a minimum bandwidth capacity of 50 units. Similarly, Request B demands a path from node 5 to node 6 crossing over datacenters instantiating functions *d, e, f* with minimum bandwidth capacity of 100

units. Also, each function here requires a certain amount of datacenter resources, as denoted by the number next to each requested VNF. Furthermore the grey dashed lines illustrate some potential VNF mappings, whereas the blue and green dashed lines illustrate potential connection path satisfying requests A and B, respectively.

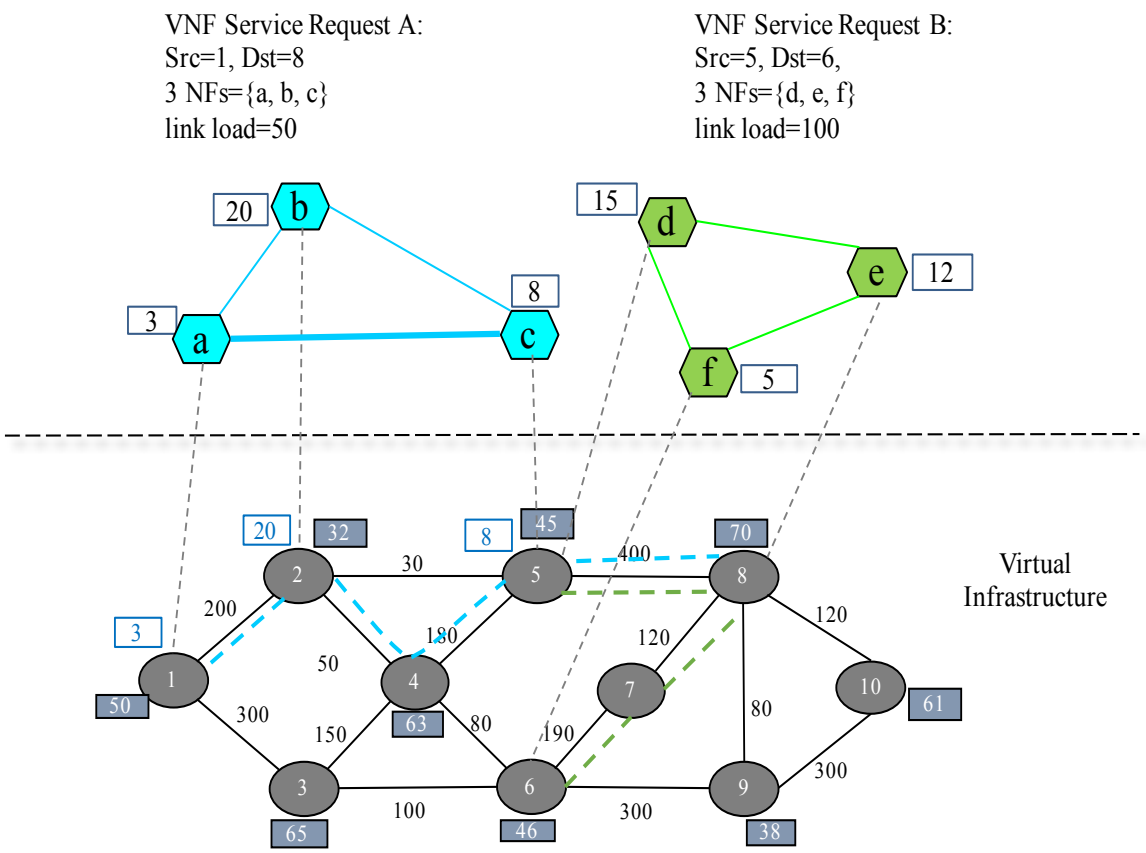


Figure 2.1: VNF placement overview

Overall researchers have studied a range of algorithmic schemes for VNF placement and/or routing. Most of these methods try to achieve a given objective, e.g., such as minimize certain quantities (such as resource usage, physical cost, link load, redeployment delay),

maximize other quantities (such as resource, carried load, energy efficiency, and availability/reliability), or achieve a tradeoff. Accordinglyl, Figure 2.2 presents a high-level taxonomy of some of the existing VNF placement and/or routing schemes. Those solutions are now surveyed further.

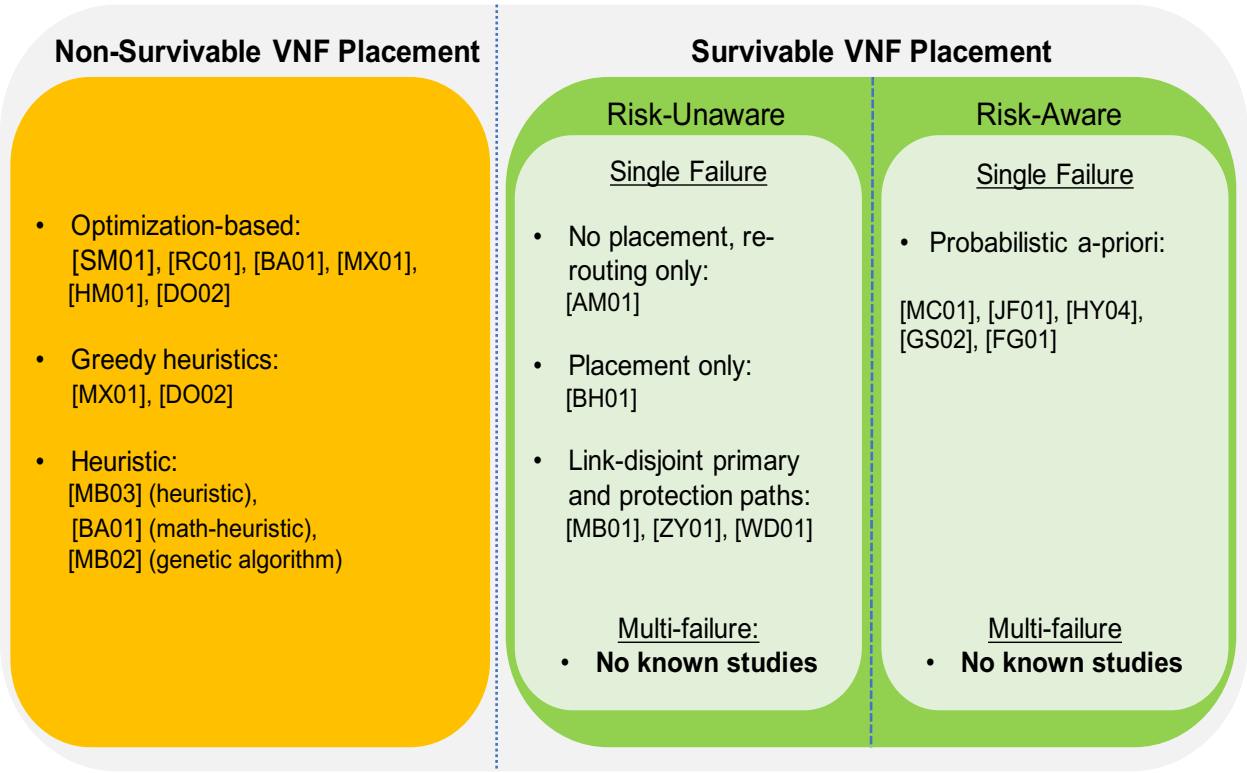


Figure 2.2: A summary of VNF placement and survivability efforts

Foremost, a wide range of studies have looked at VNF placement under regular, i.e., working network conditions, i.e., [SM01], [HM01], [MB03], [BA01], [RC01], [MX01], [MB02], [JC01] and [DO02]. For example, [SM01] presents one of the first studies on VNF placement optimization to maximize the remaining traffic data rate, minimize the number of used network nodes and minimize the total latency over all paths. Moreover, the proposed model

considers some specific constraints, e.g., , an upper-bound number of instances of each VNF, a maximum tolerable latency between endpoints, a pre-defined requests acceptance rate per instance, etc. The solution also develops a chain classifier module to classify flows according to pre-defined chains and steer traffic according to SFC demands. Furthermore a *mixed integer quadratically constrained program* (MIQCP) scheme is also used to provision a VNF chain in the first stage and embed it into the underlying topology in the second stage. This two-stage approach is necessary due to the linear nature of the adopted scheme. Moreover, the objective function tries to either maximize the transmission data rate, minimize the overall number of assigned nodes or minimize latency. Nevertheless, this scheme does not combine any of these three provisioning objectives.

Meanwhile [RC01] proposes two ILP-based optimization schemes to minimize VNF mapping cost, i.e., uncapacitated and capacited. Specifically, mapping cost is defined as the sum of the setup cost of each function at a specific node and the distance between the nodes. Now whereas the uncapacited scheme assumes that a datacenter has unlimited instance constraints while the capacited scheme, akin to [SM01], assumes a finite and pre-determined number of instances of a VNF. Overall, this solution addresses two key conccerns, i.e., facility location problem and *generalized assignment problem* (GAP), governed by the neighboring distance and setup costs, respectively. However, multiple approximation techniques are implemented and no routing costs, bandwidth capacity constraints nor traffic engineering constraints are taken into account. The proposed ILP schemes are compared versus a greedy

heuristic algorithm based on performance ratio. The performance ratio is defined by the ratio between the mapping cost and the number of flows supported by a function instance. Results indicate that the former reduces this ratio by at least 60%. Also the authors show that the performance rate significantly increases when function capacity is reduced, showing that as more allocations are necessary, more accurately the ILP scheme has performed. Thus, the authors show that although the location problem is NP-hard, ILP schemes perform better than greedy heuristics as complexity increases.

Furthermore [MB03] presents a heuristic scheme to compose and embed VNF chains (SFCs) and minimize bandwidth utilization in a reasonable amount of time, termed as CoordVNF. Specifically, the solution defines a maximum path length parameter to limit the distance between a mapped node and the corresponding substrate nodes, i.e., and achieve some level of flexibility. The CoordVNF scheme is then compared to the MIQCP scheme from [SM01] for maximum path length parameters ranging from 1-5 hops. Overall results show that this method gives significantly faster embedding and service chaining, i.e., on the order of milliseconds vesus order of seconds. This joint performance gain occurs due to the heuristic nature of the scheme, which jointly performs both chaining and embedding in a single stage, i.e., whereas the MIQCP algorithm uses a two-stage approach.

Additionally, [BA01] introduces a *mixed integer linear programming* (MILP) scheme to optimally place VNF datacenters over an NFVI and optimally assign requested functions (to VNF datacenters) to build the chains. Specifically, the authors try minimize resources

usage (node and link) and use two different objective functions. Namely, the first technique, termed as TE, defines two separate objective functions, i.e., one to minimize node load and the otther to minimize link load. Meanwhile, the second technique minimizes both node and link loads in the same objective function, termed as TE-NFV.. Additionally, the authors also extend their work in [VM01] and build a heuristic algorithm to handle larger, more realistic operational scenarios. Overall, results show that the TE-NFV scheme reduces the number of VNFs instantiated as compared to the single TE minimization scheme. In addition, VNF setup cost is also 70% lower here. Now these results indicate that the TE scheme gives slightly lower link utilization (about 5% less). However placing VNFs traffic engineering yields substantial setup cost while delay remains the same for both.

Now some studies have also addressed VNF placement for specific solutions, i.e., deploying VNFs to implement a single and specific service. For example [MX01] focuses on heterogeneous NFVI Layer 1 networks and proposes a scheme to minimize NF placement cost in hybrid networks composed of optical and electronic network elements. The goal here is to reduce the *optical-to-electronic-to-optical* (O-E-O) and *electronic-to-optical-to-electronic* (EOE) conversion delay costs by mapping VNFs from the same chain to a reduced number of datacenters. The problem is modeled as a *binary integer linear programming* (BIP) optimization scheme, and a greedy heuristic scheme is also presented to improve applicability (scalability) in more complex scenarios. These two methods are compared with a greedy first-hit heuristic scheme, and results confirm that both of the proposed schemes

drastically reduce the number of required E-O-E and O-E-O conversions (as well as the number of datacenters).

Akin to the above, [MB02] also presents a placement heuristic to minimize the deployment cost of a single service in a SDN network, i.e., *deep packet inspection* (DPI) service. Overall, this effort tries to minimize the number of inspection engines deployed and their loads, as well as improve overall efficiency (traffic inspection). Accordingly, a metaheuristic *genetic algorithm* (GA) solution is proposed owing to its improved scalability versus linear solvers. This algorithm uses evolutionary strategies based upon previous results to narrow down the search space, and the authors implement all critical GA steps here, i.e., including random initial population, selection, crossover and mutation. Moreover a greedy heuristic algorithm is also devised to compute a constrained *shortest path* (SP) between the two endpoints. Furthermore, several different scenarios are evaluated for two key parameters, i.e., cost of a DPI engine and maximum bandwidth usage per link. Overall results show a clear tradeoff between the number of DPI engines and the network load, e.g., the overall deployment cost can be reduced by up to 58% for a lower DPI cost. Finally Gebert et al [SG01] also present a heuristic algorithm to efficiently place VNFs in a service provider NFVI and optimize cellular coverage for large events.

2.3 NFV Survivability

Overall most VNF placement schemes have focused on objectives such as performance improvement, cost reduction, energy efficiency, traffic engineering, etc. However, VNF sur-

vivability (reliability) is now becoming a major concern given the critical nature of many services. In many cases service providers will still be expected to provide "nines" reliability, regardless of if services are being provisioned over commodity systems or legacy carrier-grade platforms.

Now [BH01] presents a high-level look at VNF resiliency and outlines a range of requirements, i.e., failure management, state management/synchronization, VNF migration, and handling larger correlated failures. However this work does not present any specific solution strategies. Instead, only a handful of efforts have addressed survivable VNF placement to improve the resiliency of services (mostly for single failures). Specifically, these methodologies have pursued a range of routing, redeployment and pre-provisioned (protection) schemes. A review of the existing VNF survivable schemes is now presented, see also Figure 2.2.

2.3.1 Single Failure Scenarios

Most VNF mapping studies have only addressed survivability concerns for isolated single node and link failures. For example, [AM01] considers the case of a single VNF failure, i.e., datacenter outage, causing a service chain interruption. A extended orchestration architecture is then proposed to dynamically redefine flows and steer (re-route) traffic to establish new paths and reduce downtime. Nevertheless, VNF placement is not considered here. Further considerations for resource limitations and bandwidth constraints are also lacking. As such, this effort only focuses on reactive disaster recovery.

Meanwhile [MB01] proposes another resilient SFC allocation scheme. First, a greedy heuristic algorithm is designed to map the *virtual network functions forwarding graphs* (VNF-FGs) for service chaining requests, thereby yielding resource allocation constraints and VNFs interdependence. Two different protection strategies are then defined, ie., *link resilience* and *VNF resilience*. The former implements link-disjoint path protection between all neighboring datacenters, whereas the latter defines alternate datacenters for each NF, i.e., more than one NF per datacenter allowed. A complex (time-consuming) back-tracking scheme is then presented to allocate resources. Specifically, the solution randomly choses a datacenter that is able to instantiate the first NF belonging to a service chain, and then performs a breadth search to find a datacenter capable of instantiating the next function. Resources are then allocated at these datacenters. However, if a datacenter cannot be assigned for a specific NF (according to the service chain specifications), all resources are deallocated and another search is initiated. However, runtime performance is only evaluated for a small topology and a small number of requests, resulting in very few backtracking searches. Furthermore, no resiliency results are presented either.

Meanwhile, the work in [ZY01] presents a *joint topology design and mapping* (JTDM) solution, which uses a heuristic scheme termed as *closed-loop with critical mapping feedback*. This solution builds the network topology (NFVI) and then maps the VNFs in order to minimize the *total bandwidth cost* (TBC). In particular, TBC minimization is achieved by performing function combination, i.e., deploying two or more NFs in the same datacenter to

reduce bandwidth load. Furthermore, this solution also incorporates reliability concerns by computing node- and link-disjoint protection service chains. In particular, two protection schemes are considered here, i.e., dedicated and shared [HL01]. Results show a clear tradeoff between reliability and efficiency as expected, with dedicated protection giving higher (lower) reliability (efficiency) than shared protection.

Furthermore, the work in [MC01] takes a slightly different approach and assumes the availability of a-priori probabilistic resource availability levels. An optimization-based MILP scheme and VNF provisioning heuristic are then developed to incorporate these availability levels, i.e., service paths is computed as the product of availability levels of all resources satisfying the request. This study also assumes that the NFVI is comprised of a large number of nodes with limited resources distributed across multiple datacenters, and that each node can only host one VNF instance. Overall, the MILP can only handle a small number of demands, whereas the heuristic is capable of fielding more realistic scenarios. Associated results here confirm excessively high computation times with the MILP approach even for smaller testcases. However, although a-priori probabilities are considered here, broader resources constraints, routing costs and traffic engineering concerns are not studied.

Akin to the above, [JF01] presents another polynomial-time greedy heuristic scheme to collect the actual temporal availability of physical resources. This information is then used to compute the probabilistic availability of a path and further resolve a primary-protection path pair. Additionally this scheme also tries to increase the acceptance rate of online re-

quests by minimizing physical resources consumption, i.e., map NFs according to availability and resource usage levels. Overall results show that the proposed scheme gives notable better resilience and resource efficiency versus traditional greedy primary-backup protection schemes (k-shortest paths), e.g., 27% shorter path lengths (link usage).

Finally, [WD01] introduces a *cost-efficient redundancy algorithm* (CERA) heuristic to efficiently place primary (working) VNFs along with redundant/backup ones. The authors assume that most studies do not consider each NF as part of an interdependent end-to-end service connection, i.e., service chain. In particular, the protection mapping takes into account the NFVI resources by using a *cost-aware importance measure* (CIM) scheme to minimize resource allocation of backup nodes, which tries to guarantee improved performance and therefore higher reliability to the SFC backup paths. Finally, the proposed scheme is compared with two other schemes, i.e., one which tries to minimize backup physical resources costs and another which tries to achieve maximum reliability per iteration. Results are presented for four different metrics, i.e., backup allocation cost, number of physical nodes within a backup path, cost-effective ratio and number of accepted requests. Findings confirm that the CERA scheme outperforms the others for all scenarios. However no failover and disaster-recovery results are presented.

2.3.2 Multi-Failure Scenarios

As noted earlier, most existing VNF provisioning solutions are only designed to handle isolated single failures. As such, these methods will be largely ineffective against large-scale

failure events/stressors, such as natural disasters, power outages, WMD attacks, etc. In fact, there are no known studies on VNF placement and routing within the context of multi-failure disasters. Moreover, the existing body of work on network disaster-recovery has only focused on point-to-point connections and *virtual network* (VN) services. Consider some details here.

In general, disaster events yield a large number of infrastructure node/link failures with a high degree of temporal and spatial correlation. As a result most solutions here define an *a-priori* set of "risk regions" to model probabilistic disaster events (also termed as stressors). Namely, here each region has an associated probability as well as failure sub-graph, i.e., vulnerable subset of datacenter nodes and links. Along these lines, [HL01] presents one of the first studies on "risk-aware" connection routing. Namely, a *probabilistic shared risk link group* (p-SRLG) model is introduced to specify a-priori failure risk regions with conditional node and link failure probabilities. Leveraging this, an advanced *integer non-linear programming* (INLP) formulation is then presented to minimize single connection path and disjoint primary-backup path pair failure risks. Further approximation and linear relaxation techniques are also developed to improve scalability. However, this framework does not incorporate any resource efficiency concerns and hence may lead to longer paths.

To improve resource efficiency, [DO01] proposes a graph-based heuristic method to jointly incorporate both risk and resource usage concerns. In particular, the authors assume that all requests arrive in a dynamic manner and are composed of a source node, destination

node and a required link capacity. Also, a set of probabilistic independent failure events (p-SRLG) are also assumed here, as per [HL01], and associated with a plural number of vulnerable links. The heuristic scheme analyzes the topology graph and prunes all links with insufficient capacity, creating a secondary topology graph. Adaptive weights are then assigned to the remaining links according to their respective failure probabilities and k-shorest paths are computed for the primary source-destination route. Link-disjoint protection paths are computed for each of these k paths and the path-pair with lowest failure probability is chosen, characterizing a "risk-aware" weighted link-disjoint path pair definition. Finally, the proposed risk-aware joint protection path and load balancing scheme is compared with three other schemes that do not implement risk-awareness and one that does in regards to failure rate, protection failure rate, bandwidth blocking rate and average route length. Overall, results confirm that the proposed method yields much lower failure rates versus "non-risk-aware" schemes (about 75% less) and yields a minimal increase in average route lengths. Hence this study shows that combining risk-awareness, and traffic engineering can yield substantial gains with regards to survivability.

More recently, follow-on studies have also applied the above-detailed multi-failure "risk-region" models for VN services. For example, [HY04] and [GS02] proposes two heuristic schemes to achieve backup virtual node and link provisioning, i.e., termed as supplemental and incremental. The former computes mappings for each risk region and then combines them using resource sharing on common nodes and links. Meanwhile the latter adds backup

virtual nodes/links as needed to a base mapping in order to protect against each potential risk region. However, these schemes generally yield very high resource inefficiency, and hence further work in [FG01] proposes improved optimization and heuristics-based strategies that work by grouping the failure regions, see related reference for details.

2.4 Open Challenges

Despite the aforementioned contributions, in general there are no known studies on disaster-aware provisioning of NFV-capable infrastructures. This is a major concern since large stressor events can cause multiple failures and resulting widespread services disruption across the whole network, i.e., due to the high degree of virtual function multiplexing being done. In light of the above, there is a pressing need to study NFV provisioning within the context of multi-failure disasters, i.e., network function placement and routing. These solutions should incorporate a-priori risk (vulnerability) information and also take into account resource efficiency concerns.

Furthermore, as noted in Section 2.2, most regular VNF mapping schemes perform NF placement under the assumption that datacenters have infinite resources to satisfy all demands. However, under heavy load scenarios and/or post-fault conditions, resource constraints (scarcity) will likely arise. As a result, there is also a further need to incorporate datacenter node and link bandwidth constraints into the VNF placement and routing process.

Chapter 3 Joint VNF Placement and Routing with Traffic Engineering [1]

As previously mentioned, an increasing number of studies have looked at VNF placement and routing to improve service deployment performance, efficiency and reliability. However most of these efforts have tried to minimize placement costs under the assumption of unlimited network resources to satisfy all requests. As a result, maximizing the number of satisfied NFs is usually secondary. To address more realistic scenarios, this chapter addresses more realistic resource-limited scenarios, e.g., such as those arising during heavy load intervals or after resources failures/outages.

Foremost, an optimization-based scheme is presented for VNF placement and routing, i.e., termed as the *multi-objective minimized link load ILP* (MLL-ILP) scheme. This solution pursues several objectives in a weighted manner in order to provide an adaptive solution, i.e., maximize the number of satisfied requests, reduce infrastructure deployment costs, reduce routing costs, and also improve load-balancing, overall this multi-objective approach allows providers to tailor the outputs to meet their desired needs/tradeoffs. For example if a datacenter has limited physical resources but sufficient link and routing capacity, deployment costs may be given higher weighting. On the other hand, if link capacities are constrained and path hops and delay times are high, lower routing costs may be more favorable. Also, the

ability to implement load balancing is becoming increasingly important, i.e., as it can benefit service performance in cloud computing and in general, lead to increased revenues. Hence the proposed solutions also incorporate links load levels to improve the VNF provisioning process. Finally, owing to ILP complexity, a further polynomial-time heuristic scheme is also presented to improve scalability, i.e., termed as the *multi-objective minimized link load greedy heuristic* (MLL-GR) scheme. These methods are also analyzed and compared for sample VNF deployment scenarios. Overall this work provides a good basis to develop subsequent disaster-ware VNF schemes.

3.1 Notation Overview

The requisite notation is introduced first. Consider a network infrastructure topology represented by a graph $G = (V, E)$, where V is the set of nodes and E the set of links. Each link $(i, j) \in E$ has an associated cost c^{ij} and capacity b^{ij}, which quantifies the cost of using that link and its maximum bandwidth capacity, respectively. Meanwhile, the subset $D \subseteq V$ represents the set of datacenters implementing the NFs, and the complete set of NFs is denoted by F. Hence a given datacenter $d \in D$ only implements a subset of functions $F_d \subseteq F$. Furthermore, the set of requests is given by the set R. Namely each request $r \in R$ is characterized by a 4-tuple $(src_r,\ dst_r,\ F_r,\ b_r)$, which denotes the source and destination nodes of the flow, the set of requested functions $F_r \subseteq F$, and the required (minimum) bandwidth interconnection capacity, respectively.

Now it is assumed that the infrastructure has a total of m different VNF resource types, i.e., resource dimensionality. For example, m=3 can denote processor, storage and memory. In order to model resource constraints, it is also assumed that a datacenter $d \in D$ has a finite amount of resources $W_d = \{w_{d,1}, w_{d,2}, ..., w_{d,m}\}$. Hence a datacenter $d \in D$ implements a function $i \in F_d$ must employ $w^i_{d,1}, w^i_{d,2}, ..., w^i_{d,m}$ resources. Furthermore, this resource requirement is also datacenter-dependent, which reflects the fact that some datacenters may be designed/specialized to implement certain especific functions. Finally, the setup cost of locating/placing an instance of a function $i \in F_d$ at datacenter d is given by c^i_d, and an instance of function i at datacenter d can serve up to λ^i_d requests. Hence in order to accommodate more requests, multiple instances of function i must be deployed at datacenter d, with each consuming additional resources and imposing further setup cost. All of the afformentioned variables are also summarized in Table 3.1, and several other variables are also defined in Section 3.2.

3.2 Multi-Objective Minimized Link Load ILP (MLL-ILP) Model

A detailed MLL-ILP optimization formulation is now presented to achieve several key objectives, i.e., minimize the number of satisfied NFs, minimize deployment cost, minimize routing cost and minimize the maximum overall link load. Consider some requisite ILP-related variable definitions first:

- $x^i_{r,d} \in \{0,1\} \quad r \in R, i \in F_r, d \in D | i \in F_d$: Indicates whether function $i \in F_r$ requested by request $r \in R$ is implemented at datacenter $d \in D$, i.e., binary

Table 3.1: List of variables

Variable	**Description**
V	Set of nodes
v	Node $\in V$
E	Set of links
(i, j)	Link $\in E$ connecting nodes i and j
c^{ij}	Link (i, j) setup cost
b^{ij}	Link (i, j) bandwidth
$D \subseteq V$	Set of datacenters
d	Datacenter $\in D$
F	Set of all functions
i	Function $\in F$
R	Set of requests
r	Request $\in R$
F_r	Set of functions requested by request r
b_r	Minimum load required by request r
m	Resource dimensionality

- $y_d^i \in \mathbb{Z}^+ \quad d \in D, i \in F_d$: Represents the number of instances of function $i \in F$ at node $d \in D$
- $l_r^{i,j} \in \{0, 1\} \quad r \in R, (i, j) \in E$: Indicates whether link $(i, j) \in E$ is used to route the traffic flow for request $r \in R$, i.e., binary
- $0 \leq \alpha \leq 1$: Represents the highest overall link usage ratio, i.e., sum of all b_r using a link $(i, j) \in E$ divided by link capacity $b_{i,j}$, i.e., fractional quantity

Based upon the above, the MLL-ILP objective function is defined as:

$$\begin{aligned} \max \text{ F} \quad = \quad & w_1 \sum_{r \in R} \sum_{i \in F_r} \sum_{d \in D | i \in F_d} x^i_{r,d} - w_2 \sum_{d \in D} \sum_{i \in F_d} c^i_d y^i_d \\ & - w_3 \sum_{r \in R} \sum_{(i,j) \in E} c^{ij} l^{ij}_r - \alpha w_4 \end{aligned} \tag{3.1}$$

subject to:

$$\sum_{d \in D} x^i_{r,d} \quad \leq \quad 1 \quad r \in R, i \in F_r \tag{3.2}$$

$$x^i_{r,d} \quad \leq \quad y^i_d \quad r \in R, i \in F_r, d \in D | i \in F_d \tag{3.3}$$

$$\sum_{i \in F_d} w^i_{d,j} y^i_d \quad \leq \quad w_{d,j} \quad d \in D, r \in R, j \in \{1, 2, ..., m\} \tag{3.4}$$

$$\sum_{r \in R} x^i_{r,d} \quad \leq \quad \lambda^i_d y^i_d \quad d \in D, i \in F_d \tag{3.5}$$

$$\sum_{j:(i,j) \in E} l^{ij}_r - \sum_{j:(j,i) \in E} l^{ji}_r = \begin{cases} -1; i = dst_r, src_r \neq dst_r \\ 1; i = src_r, src_r \neq dst_r \\ 0; \text{ otherwise. } i \in V, r \in R \end{cases} \tag{3.6}$$

$$\sum_{(d,j) \in E} l^{dj}_r \quad \geq \quad x^i_{r,d} \quad r \in R, i \in F_r, d \in D | i \in F_d \tag{3.7}$$

$$\sum_{r \in R} l^{i,j}_r b_r \quad \leq \quad \alpha b_{i,j} \quad \{i, j\} \in E \tag{3.8}$$

Overall, the objective function in Eq. 3.1 consists of a series of terms scaled by their respective weighting factors, i.e., w_1, w_2, w_3 and w_4. In particular, the first term in Eq. 3.1 represents the total number of requested NFs, whereas the second term is the total cost to setup/deploy NFs at the various datacenters (also called deployment cost). Meanwhile, the third term is the total routing cost, and the fourth term represents the maximum overall link load. Note that the second, third and fourth terms are negative since maximizing a negative term is equivalent to minimizing it [JC02],[JC03]. Furthermore, the setup cost is directly related to the number of satisfied functions and number of instances of each function. Thus the second term (setup cost) depends upon the number of NFs per request and on how many datacenters are used to host these NFs. On the other hand, the third and fourth terms (routing and maximum link load costs) depend upon the number of requests and number of links used. These latter two terms are independent of the number of NFs and NF instances. Now one of the key goals of the MLL-ILP scheme is to avoid link overload or at least minimize link load. Therefore the fourth term in Eq. 3.1 introduces a link load minimization function variable α, which is linked to Eq. 3.8 (detailed later).

Meanwhile, Eqs. 3.2-3.8 represent the model constraints. Namely, Eq. 3.2 ensures that a function i requested by request r is serviced by at most one datacenter d. Since one of the objectives of the MLL-ILP scheme is to maximize the sum of all variables $x^i_{r,d}$ (first term of Eq. 3.1), the optimal solution will drive the constraint in Eq. 3.2 to equality. Meanwhile Eq. 3.3 mandates that function i should be located at datacenter d if request r is assigned

to function i at datacenter d. Also, Eq. 3.4 ensures that the aggregate amount of type j resources used by all functions instantiated at datacenter d should be limited by the total amount of such resources at this datacenter, i.e., $w_{d,j} \in W_d, j \in \{1, 2, ..., m\}$. Similarly, Eq. 3.5 states that the total number of requests for function i served by datacenter d should be bounded by the number of instances of i at d times the capacity λ_d^i of an instance i. Additionally, Eq. 3.6 represents the necessary flow conservation constraint. Also, Eq. 3.7 guarantees that if a function i is requested by a request r and is placed at datacenter d, then the demand traffic flow must be routed through that datacenter. Finally, Eq. 3.8 relates to load-balancing and ensures that the sum of all links (i, j) used by a traffic flow (associated with request r) times the requested load b_r demanded by request r is bounded by the product of the link capacity $b_{i,j}$ times α. As a result, traffic flows are associated with links that have reduced load profiles (higher available capacity).

3.3 MLL-ILP Complexity

Since the MLL-ILP objective function and constraints are linear and all variables are either integer or binary, the optimization problem is NP-hard [TC01]. Hence the complexity of this model can be judged by the total number of variables that the scheme utilizes. In particular, the total number of variables $x_{r,d}^i$, y_d^i and $l_r^{i,j}$ (Eqs. **??**-**??**) is upper-bounded by $|R||F||D|$, $|F||D|$ and $|R||E|$, respectively.Based upon this, the upper-bounds for the number of constraints, Eqs. 3.2-3.5, are also $|R||F|$, $|R||F||D|$, $|R||D||W_d|$ and $|F||D|$, respectively.

Meanwhile, the upper-bounds for the number of constraints in Eqs. 3.6 and 3.7 are $|V||R|$ and $|R||F||D|$. Finally, the upper-bound on the number of constraints in Eq. 3.8 is $|V||R| + 1$.

From the above, the upper-bound for the total number of variables is dominated by the product $|R||F||D|$. Now in general, it is very difficult to pre-specify limits for the number of requests or number of functions to ensure ILP convergence. However for small-to-medium sized network topologies, the proposed ILP can be solved in a reasonable amount of time, i.e., less than a few hundred nodes.

3.4 MLL-GR Heuristic

Although the MLL-ILP approach provides an optimal VNF placement solution, its run-time performance can be a concern in complex scenarios with increased variable costs and high resource dimensionality. As a result, heuristic schemes can be developed to build more scalable, albeit non-optimal, solutions. Along these lines, a greedy graph-based heuristic algorithm is also presented to solve the NF placement problem in larger networks, as shown in Figure 3.1. Overall, this algorithm implements a two-stage solution. Namely, given a graph and input parameters (akin to MLL-ILP; source, destination, set of NFs and minimum link capacity), the algorithm returns the VNF mappings, associated paths, number of function instances in each datacenter and the highest link load. Before presenting this scheme, however, some further variable definitions are introduced here as follows.

- K: Number of datacenters that satisfies function i requested by request r
- d_k: Datacenter that satisfy function i requested by request r
- $y^i_{d_k}$: Total number of instances of function i in datacenter d_k
- $\lambda^i_{d_k}$: Maximum number of clients per instance of function i in datacenter d_k
- $D(r)$: Set of datacenters serving request r
- $C(r)$: Subset of $D(r)$ that tracks the datacenters serving request r connected to their respective neighbors within the source-destination path

Overall, the first stage in Figure 3.1 (lines 4-15) focuses on NF placement at specific datacenters to reduce deployment cost. Namely, for each function i requested by $r \in R$, the algorithm selects the datacenter d_k that implements i with the lowest setup cost and at the same time has sufficient resources to host upcoming requests (line 8). Once a NF is placed, the resources at datacenter d_k are updated accordingly. Namely if there is an instance of NF i at datacenter d_k with enough instance capacity to satisfy request r, the remaining capacity is simply decremented by 1. Otherwise, another instance of NF i is created at cost $c^i_{d_k}$, where each newly-created instance can serve $\lambda^i_{d_k} - 1$ requests. Hence either way, the available resource levels at datacenter d_k, i.e., $w_{d_k,i}$ $(1 \leq j \leq m)$ are reduced by $w^i_{d_k,j}$ (line 9, Figure 3.1). Note that serving a request does not incur any additional cost, i.e., only new instantiations of NF i do. Additionally, for each new instance of NF i, the total number of instances at datacenter $y^i_{d_k}$ is also incremented by 1 (line 10, Figure 3.1) and x^i_{r,d_k} is set to 1 as well (line 11, Figure 3.1).

To conclude NF placement (first stage), datacenter d_k is also added to a set variable, $D(r)$, which tracks the subset of datacenters serving request r.

Meanwhile, the second stage in Figure 3.1 (lines 14-27) computes the shortest path between the source src_r and destination dst_r, that passes through all datacenters $d_k \in D(r)$. Initially, the algorithm starts by connecting the source node src_r to the first datacenter d_1 in $D(r)$. In particular, this connection route is computed using the constrained Dijkstra's shortest path algorithm (line 20, Figure 3.1) and only considers links in $G(V, E)$ with sufficient capacity to support the requested b_r. The routing cost here is defined by the sum of the setup cost of all links associated with a traffic flow and each respective link capacity/request load ratio, as follow:

$$\sum_{(i,j)\in E} c^{ij} + b_r/b_{ij} \tag{3.9}$$

All variables $l_r^{i,j}$ along the path are then set to 1 (line 21, Figure 3.1), and datacenter d_1 is also added (line 22, Figure 3.1) to another set variable $C(r)$. Namely, this subset tracks the datacenters serving request r that have already been connected to their respective neighbors within the src_r and dst_r path. Now if there is another datacenter $j \in D$ in the path between the source and destination, it is also added to $C(r)$ (line 23, Figure 3.1), i.e., in order to avoid duplicated/overlapped routes for datacenters yet to be analyzed in upcoming iterations. Before returning to the beginning of the loop, the temporary src variable is also replaced by the destination datacenter temporary dst (line 24), which is initially set to datacenter d_k (line 18).

The algorithm (second stage) then loops to process the next datacenter serving the request. Again, a shortest-path is computed between the previous destination and the following datacenter $d_k \in D(r)$, i.e., in the second iteration a path is computed between d_1 and d_2. In the two final steps, a shortest path is computed between the last datacenter and the destination *dst* (line 30), and all links $l_r^{i,j}$ within that traffic flow are set to 1 (line 27). Now, note that the greedy heuristic scheme is not an optimization algorithm, which means that it does not loop searching for lower solutions. It is a one-time iteration method, and by the end of the only iteration, costs are calculated for each and all requests.

3.5 MLL-GR Complexity

Consider the computational complexity of the MLL-GR heuristic. Conceptually, this algorithm tries to lower runtime overhead by finding the first acceptable solution. In particular, the first stage selects a datacenter, d, that can implement function $i \in F_r$ with the lowest cost (line 8, Figure 3.1). Note that NF placement here is done in a serialized manner, i.e., each NF is placed independent of requirements of the following NF. However, the second stage poses higher complexity as it runs the constrained Dijkstra shortest-path algorithm within a double loop. In particular, this stage is invoked a total of $|R||D|$ times, $D(r) \leq D$. Now assuming a binary heap Dijkstra implementation complexity of $O(|E|log|V|)$ [TC01], the overall complexity of the MLL-GR heuristic scheme is dominated by the second stage and is given by $O(|R||D||E|log|D|)$.

1: INPUT: G(V, E), $c^{ij} \forall (i,j) \in E, R, F, D$
2: OUTPUT: $x^i_{r,d}, y^i_d, l^{ij}_r$ values
3: set $x^i_{r,d} = 0, y^i_d = 0, l^{ij}_r = 0$ for all $r \in R, i \in F_r, d \in D, (i,j) \in E$
{BEGIN FIRST STAGE}
4: **for** all $r \in R$
5: $D(r) = \{\}$
6: $k = 1$
7: **for** all $i \in F_r$
8: d_k = datacenter that implements i at minimum cost and has enough resources to serve an additional request
9: update resources of d_k
10: update $y^i_{d_k}$
11: set $x^i_{r,d_k} = 1$
12: $D(r) = D(r) \cup d_k$
13: $k = k + 1$
{END FIRST STAGE}
{BEGIN SECOND STAGE}
14: **for** all $r \in R$
15: $src = src_r$
16: $C(r) = \{src\}$
17: **for** $k = 1$ to $|D(r)|$
18: $dst = d_k$
19: **if** $d_k \notin C(r)$
20: $SP = constrained\_Dijkstra(src, dst)$
21: set $l^{ij}_r = 1$ for all link $(i,j) \in SP$
22: $C(r) = C(r) \cup d_k$
23: $C(r) \cup j$, for all datacenter $j \in SP, j \in D(r)$
24: $src = dst$
25: $dst = dst_r$
26: $SP = constrained\_Dijkstra(src, dst)$
27: set $l^{ij}_r = 1$ for all $(i,j) \in SP$
{END SECOND STAGE}
28: **return** NF mappings, NF instances in each datacenter, links within a path

Figure 3.1: MLL-GR heuristic algorithm

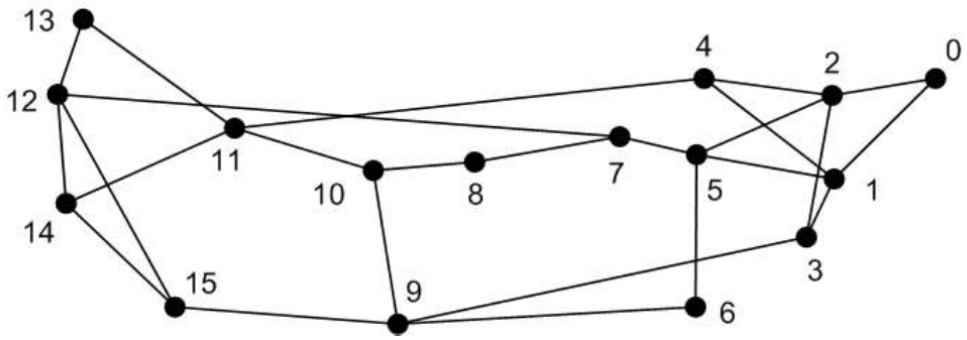


Figure 3.2: NSF network topology.

3.6 Performance Evaluation

The proposed VNF routing and placement solutions (MLL-ILP and MLL-GR) are now analyzed for a sample deployment scenario. Specifically, the optimization model is solved using the *lpsolve* solver API running on a 64-bit Windows machine with an Intel(R) Core(TM) i5 CPU (1.4GHz) and 2 gigabytes of RAM. In order to evaluate the proposed NF placement schemes, a sample 16 node/25-link network topology is chosen here, as shown in Figure 3.2. This setup reflects a large backbone facility, and it is further assumed that each node is also a datacenter i.e., in addition to providing routing/switching capabilities ($|V|$=$|D|$=16). The datacenter resource dimensionality is also set to m=3, i.e., representing processor, memory and storage.

Overall, three different testcase scenarios are defined and tested here. In particular, the first testcase performs sensitivity analysis in order to fine tune the various weighting factors in the objective function, Eq. 3.1 (for use in the second and third testcases). Meanwhile,

Table 3.2: Testcases parameters

Parameter	First Testcase	Second Testcase	Third Testcase
Link	1,000	1,000	10,000
Resources ($w_{d,1}$,$w_{d,2}$,$w_{d,3}$)	500	500	5,000
Weight w_1	1,000		
Weights w_2	1/10	1	1
Weights w_3	10/1	1	1
Weights w_4	10/1000	1,000	1,000
Function set	f0,f1,f2,f3,f4		
Required Resources ($w^i_{d,j}$)	$30 \leq w^i_{d,j} \leq 70$		
Function Setup Cost	50		
Instance Capacity	2		
Link Setup Cost	20		

the second and third testcases are used to evaluate the proposed schemes for varying physical and links resource levels. Specifically, the second testcase is termed as an *under-resourced* scenario and uses lower values for both datacenter and connectivity resources. Meanwhile the third testcase is termed as a *highly-resourced* scenario and emulates settings with higher resources levels. In light of the above, all testcases have three different datacenter resources levels, as shown in Table 3.2, i.e., W_d=$\{w_{d,1}, w_{d,2}, w_{d,3}\}$. Namely, the first and second testcases use values of $w_{d,1}$=$w_{d,2}$=$w_{d,3}$=500 units, respectively. Additionally, all link capacities are set to 1,000 units here, i.e., $b^{i,j} \leq 1,000$ for all $(i,j) \in E$. Meanwhile, the third testcase sets all datacenter resource levels to 5,000 units. The corresponding link capacity is also increased to 10,000 units here. The various other test parameters are equivalent across all testcases. Namely, the function set F is limited to five types, i.e., F=$\{f_0, f_1, ..., f_4\}$ and it is also assumed that all functions are implemented at each datacenter. i.e., $F_d = F$. The

amount of resources $w^i_{d,j}$ needed to implement a function $i \in F_d$ is also uniformly distributed between $30 \leq w^i_{d,j} \leq 70$ units. Meanwhile the setup cost of placing an instance of function $i \in F_d$ at datacenter $d \in D$ is set to $c^i_d = 50$, and the instance capacity λ^i_d of a function i at datacenter d is set to 2. Finally, each request r requires four NFs, namely F_r is randomly selected from F ($F_r \subseteq F$).

3.6.1 Weighting Factors Selection (First Testcase)

Proper selection of weighting factors in the objective function, Eq. 3.1, is crucial for effective NF placement and routing. As a result the first testcase evaluates the impact/sensitivity of these values in maximizing the number of satisfied requests (w_1) and minimizing the deployment, routing costs and link load costs (w_2, w_3 and w_4). Now recall that the first term in Eq. 3.1 represents the number of satisfied NFs, which is typically a small value. Also the other terms represent costs with negative values (minimization). Hence it is imperative to assign a relatively higher value to w_1. Meanwhile, the deployment cost (second term) is related to the number of satisfied NFs i and their respective costs. Hence deploying multiple NFs at the same datacenter reduces the deployment cost of each instance capacity, λ^i_d, at a datacenter. On the other hand, the routing cost (third term) and the link load cost (fourth term) are associated with the number of links used for each traffic flow. Although both of these values are related to the number of hops, minimizing link load cost can result in longer paths. Clearly, this trade-off needs to be analyzed further.

In general, deployment and routing costs are associated with a large number of variables in the mLL-ILP model, i.e., up to $|D||F|$ and $|R||E|$, respectively. Hence increasing one of the associated (minimization) terms in Eq. 3.1 can lower the overall cost. However this approach may also increase the other costs. Accordingly, Figure 3.3(a) and Figure 3.3(b) show the trade-off between deployment and routing costs. Namely, Figure 3.3(a) compares the deployment cost for different weight values of w_2 and w_3. Overall, a larger routing cost weight (w_3=10) gives increased deployment costs, whereas a larger deployment cost weight (w_2=10) results in lower deployment cost. Meanwhile, Figure 3.3(b) also indicates that assigning higher weights to routing cost reduces this component at the expense of higher deployment cost.

Finally, consider the link load minimization term (fourth term) in Eq. 3.1. This term only has a single fractional value α, which is computed as the ratio of between all requests b_r using a specific link $(i, j) \in E$ and its corresponding link capacity $b_{i,j}$. Note that the fourth term is directly correlated with the third term, i.e., routing cost, since both use variables l_d^{ij} and b_r. Hence w_4 can be used as a balancing factor to select between one of the other two minimization terms, i.e., deployment or routing cost. Namely, minimizing α reduces routing cost and increases deployment cost. Furthermore, increasing w_4 decreases routing cost and increases deployment cost. Finally, w_4 also provides increased sensitivity due to its fractional value (greater range).

Now in order to demonstrate the effect of varying w_4, an empirical methodology is deployed here using two different values, i.e., w_4=10 and w_4=1,000. Specifically, tests are performed for a varying number of input batch requests, ranging from 1 to 20 (20 placement solutions) with the remaining weights set to w_1=1,000, w_2=1 and w_3=1. Here, Figure 3.4(a) compares the average deployment cost for both values of w_4, i.e., computed as deployment cost/number of requests. Overall, these results show that the two costs are very similar. However, the average routing costs shown in Figure 3.4(b) show notable differences. Namely, the w_4=10 value gives anywhere from 2 to 4 times higher values than w_4=1,000, i.e., computed as routing cost/number of requests.

Overall, the maximum link load variable α can be used to modify NF placements according to service provider needs. For example, some may prefer placing NFs to reduce deployment costs due to physical datacenter resources limitations. Meanwhile others may prefer to reduce routing cost due to network link transmission constraints, i.e., lower link capacity, increased delays, etc. Based upon the above sensitivity analysis, the respective weighting factors are set to w_1=w_4=1,000 and w_2=w_3=1 in the remaining testcases.

3.7 Under-Resourced Scenarios (Second Testcase)

The second testcase considers under-resourced settings and assumes the same set of input parameters defined in Section 3.6. In particular, up to 30 arriving batch requests are processed here to measure the impact of reduced datacenter and link level resources, as shown in Table 3.2. For comparison purposes, "standardized" non-load-balancing versions of

the respective optimization and heuristic schemes (i.e., MLL-IP and MLL-GR) are devised and also tested here. In particular, the *joint routing and placement ILP* (JRP-ILP) method removes the fourth term in the objection function, i.e., $w_4\alpha$ in Eq. 3.1. Similarly, the *joint routing and placement greedy heuristic* (JRP-GR) scheme implements the algorithm shown in Figure 3.1 but removes the fourth term $w_4\alpha$.

Overall Figure 3.5(a) plots the number of satisfied NFs for the second testcase and shows that all four schemes begin to drop demands after 24 requests, i.e., due to resource limitations. Hence in the subsequent plots, results are only shown for up to 23 requests. However, note that both the JRP-ILP and JRP-GR schemes (yellow and green lines) also fail to satisfy the 15^{th} request due to their inability to balance link loads. In particular, these methods yield higher link congestion, leading to overload and rate blocking on selected links.

Meanwhile, Figure 3.5(b) also compares the deployment cost for this testcase. Clearly, all schemes yield very similar values, albeit the JRP-ILP and JRP-GR methods yield zero cost for request 15 since they cannot satisfy the requested NFs here. Furthermore, the MLL-ILP (blue) and JRP-ILP (yellow) schemes also exhibit negligibly higher deployment cost (by about 5%) for request 19 and beyond. However, the routing costs are shown in Figure 3.6(a) and clearly show a huge separation between the optimized ILP models (blue and yellow) and greedy heuristic schemes (red and green). Namely, the latter methods yield about twice as much routing costs than their respective optimization-based counterparts. Now in general, the routing cost is directly related to the number of links used to route the connections.

Along these lines, Figure 3.6(b) also plots the number of links associated with the deployed traffic flows. Clearly these results mirror those in Figure 3.6(a).

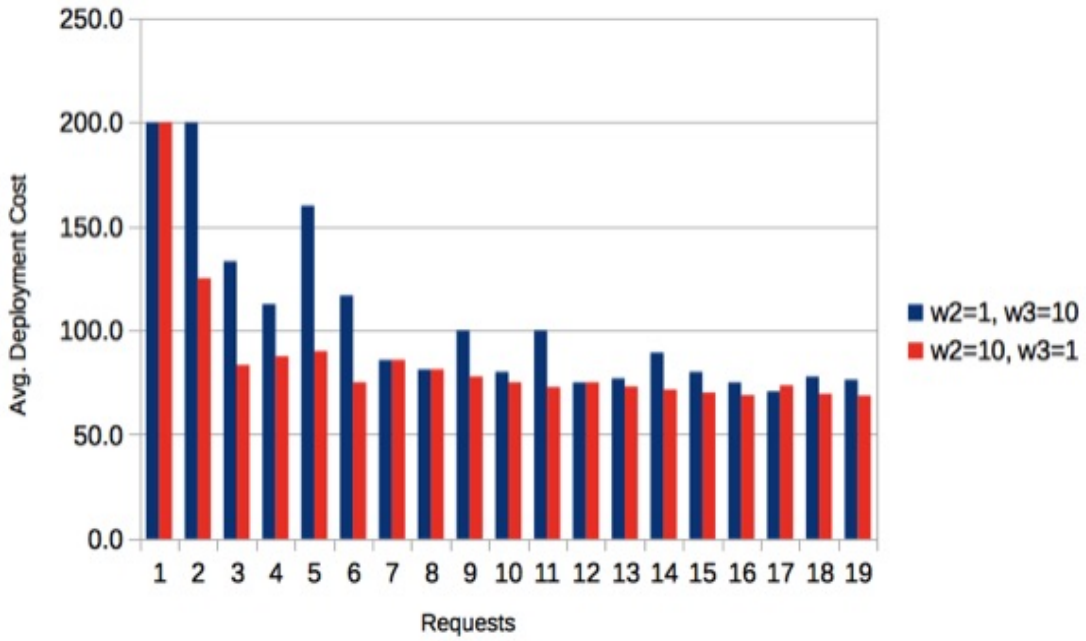
The findings in Figure 3.6(a) and (b) also indicate that the routing costs and number of links for the MLL-ILP scheme start to stabilize after 19 requests, whereas the JRP-ILP scheme maintains linear growth. Now one may postulate that the MLL-ILP scheme should use more links than the JRP-ILP scheme since it implements link load minimization. However, the MLL-ILP solution tries to minimize routing cost, which is directly related to link setup cost. Hence this scheme is more effective in minimizing routing cost and thereby the number of links as well.

3.8 Highly-Resourced Scenarios (Third Testcase)

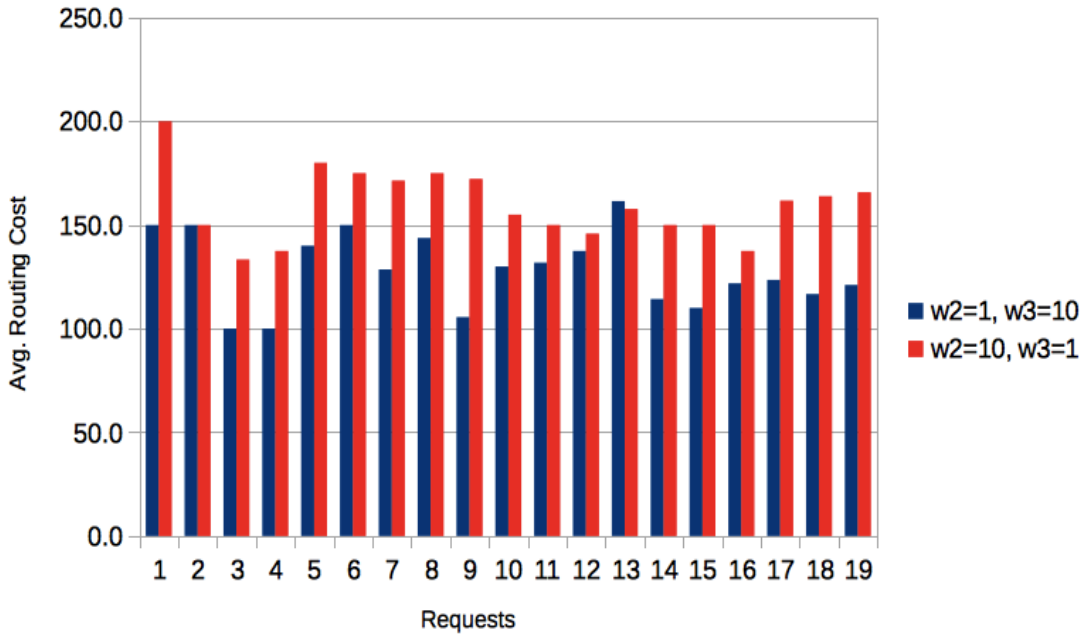
Most service providers deploy highly-resourced nodes in order to handle demand elasticity and achieve rapid service scalability. Along these lines, the third testcase is designed to evaluate these scenarios, see Table 3.2. Foremost, initial tests are done to determine if all four schemes can solve the NF placement and routing problem for all requests in both types of network scenarios, e.g., to differentiate between the MLL and STD schemes. Namely, Figure 3.7(a) compares the two ILP schemes with regards to the number of satisfied NFs for request batch sizes ranging from 1-40 requests. These results show that the JRP-ILP scheme is unable to satisfy requests 11, 13 and 38, unlike the MLL-ILP scheme which successfully provisions all requests. Equivalent results for the greedy heuristics in Figure 3.7(b) also indicate that the non-load-balancing JRP-GR scheme fails to satisfy the above-noted requests

as well (unlike the JRP-GR scheme). These findings clearly indicate that incorporating link load can help reduce demand blocking rates even in such high-resourced testcases, i.e., approximately 5-7.5% in this case.

In light of the above, further tests are also done with only the MLL-based schemes. Namely, Figure 3.8(a) plots the overall deployment costs and confirms similar performance between the optimization-based (MLL-ILP) and heuristic (mLL-GR) schemes. In particular, both methods can satisfy up to 160 requested NFs across 40 demands, i.e., 4 NFs per request. For example, the average overall deployment costs across all requests are 1,649.3 and 1,569.2 for the MLL-ILP and MLL-GR schemes, respectively (MLL-ILP deployment cost is 5% higher than MLL-GR). However, routing costs are shown in Figure 3.8(b) and indicate that the heuristic MLL-GR scheme gives much higher values than its optimization-based counterpart, i.e., about 128% higher (average overall routing costs are 982 and 2,221.5 for MLL-ILP and MLL-GR, respectively). Similarly, the MLL-GR heursitic also uses an average of 111 links to establish all traffic flows, whereas the MLL-ILP scheme only uses 48.6 links, as shown in Figure 3.9 Additional tests (not shown) are also done with larger batch sizes of up to 60 requests, and the findings confirm successful placement of all NFs with the same relative performance between the two methods.

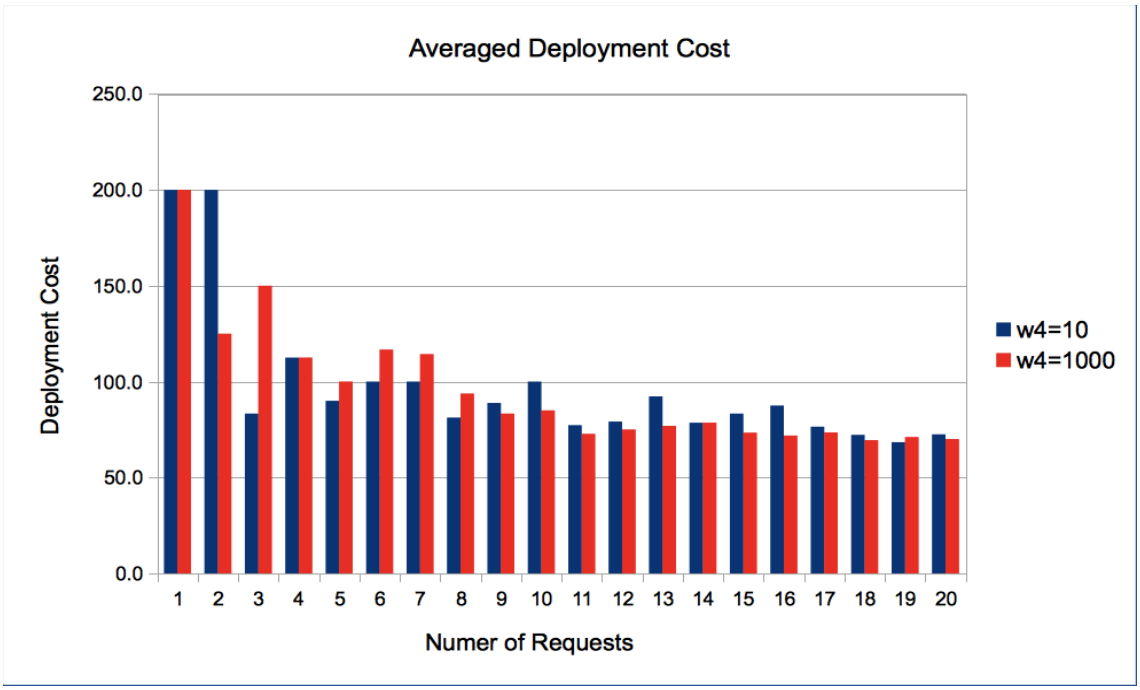


(a)

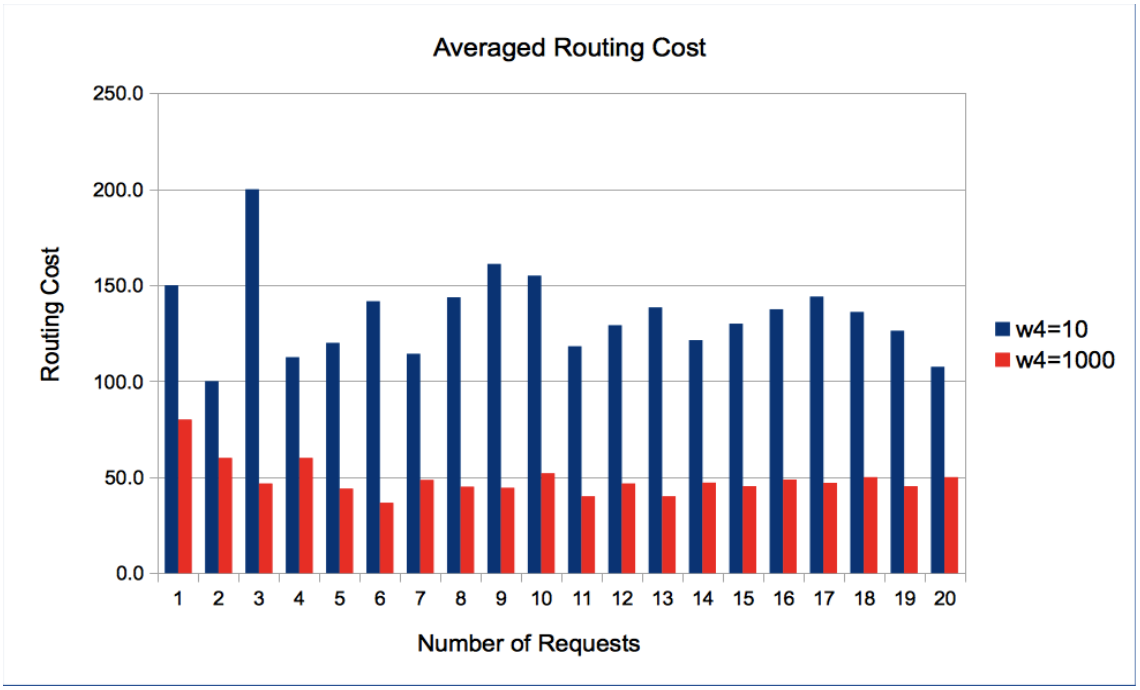


(b)

Figure 3.3: First testcase: a) average deployment cost for different w_2 and w_3 weights b) average routing cost for different w_2 and w_3 weights

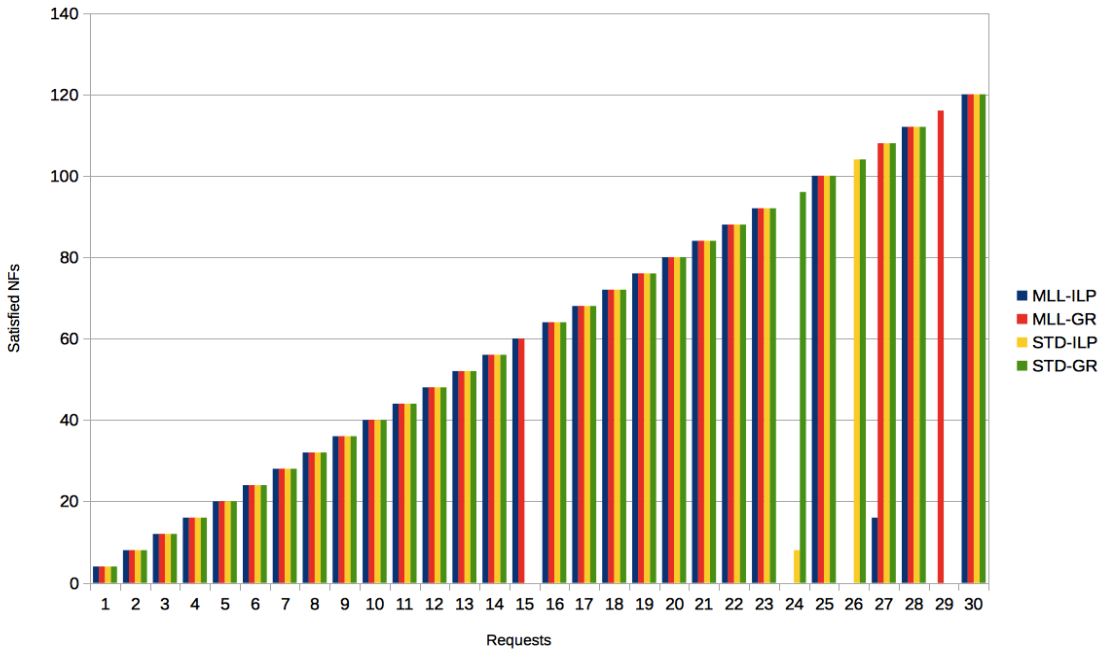


(a)

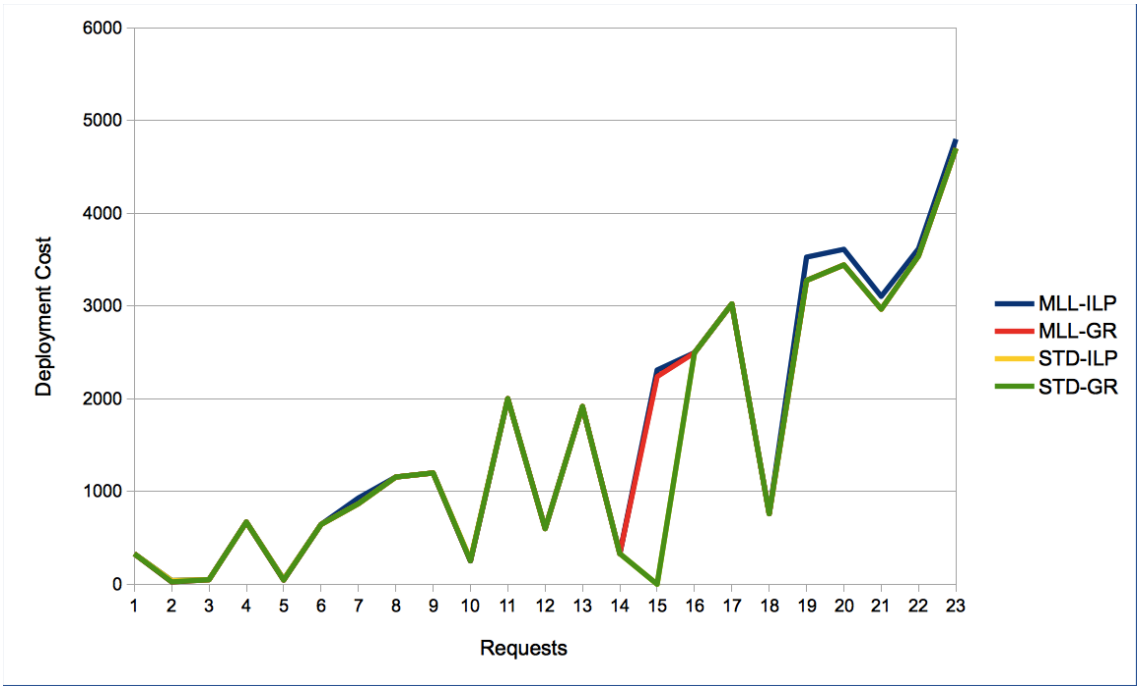


(b)

Figure 3.4: First testcase: a) average deployment cost and b) average routing cost

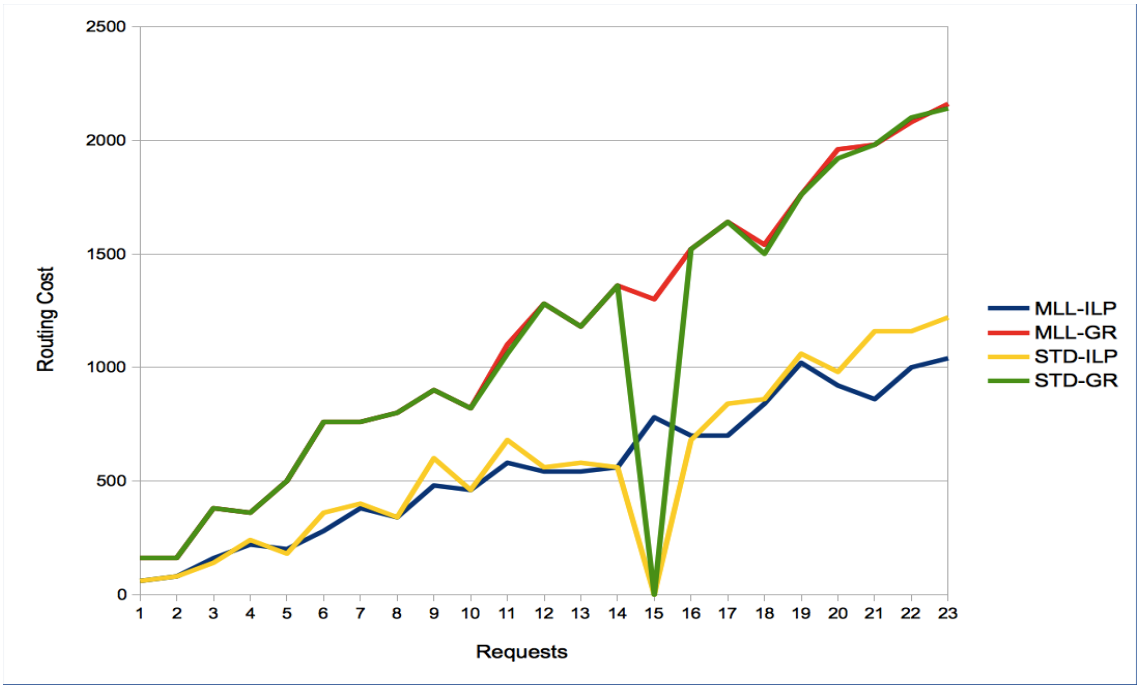


(a)

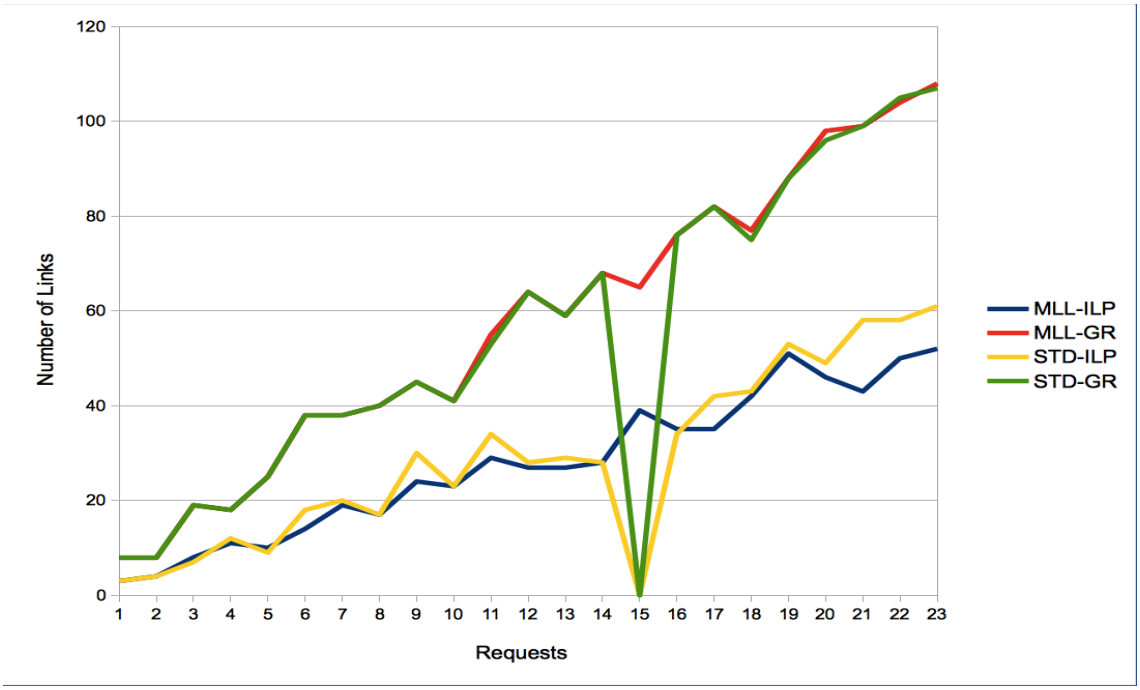


(b)

Figure 3.5: Second testcase: a) satisfied NFs, and b) deployment cost

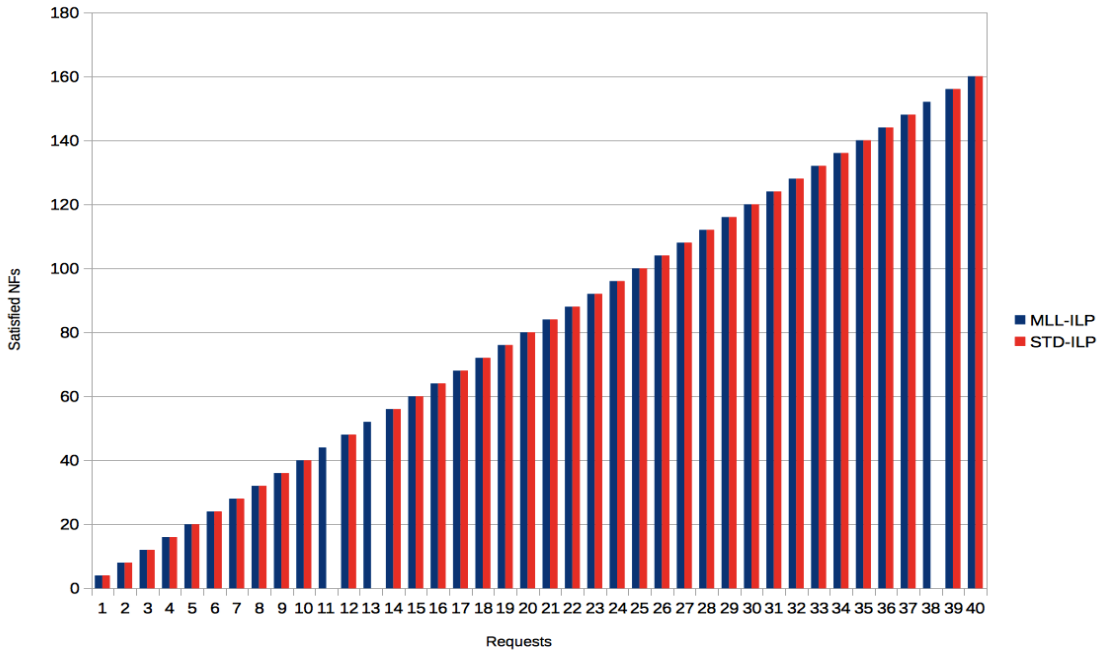


(a)

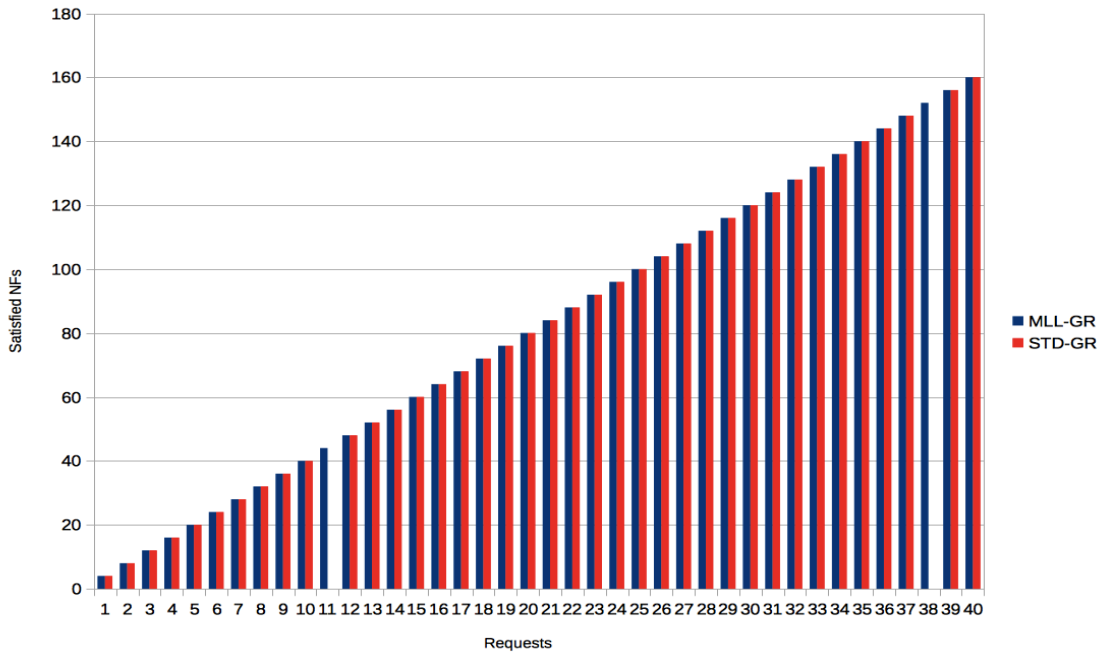


(b)

Figure 3.6: Second testcase: a) routing cost, and b) number of links

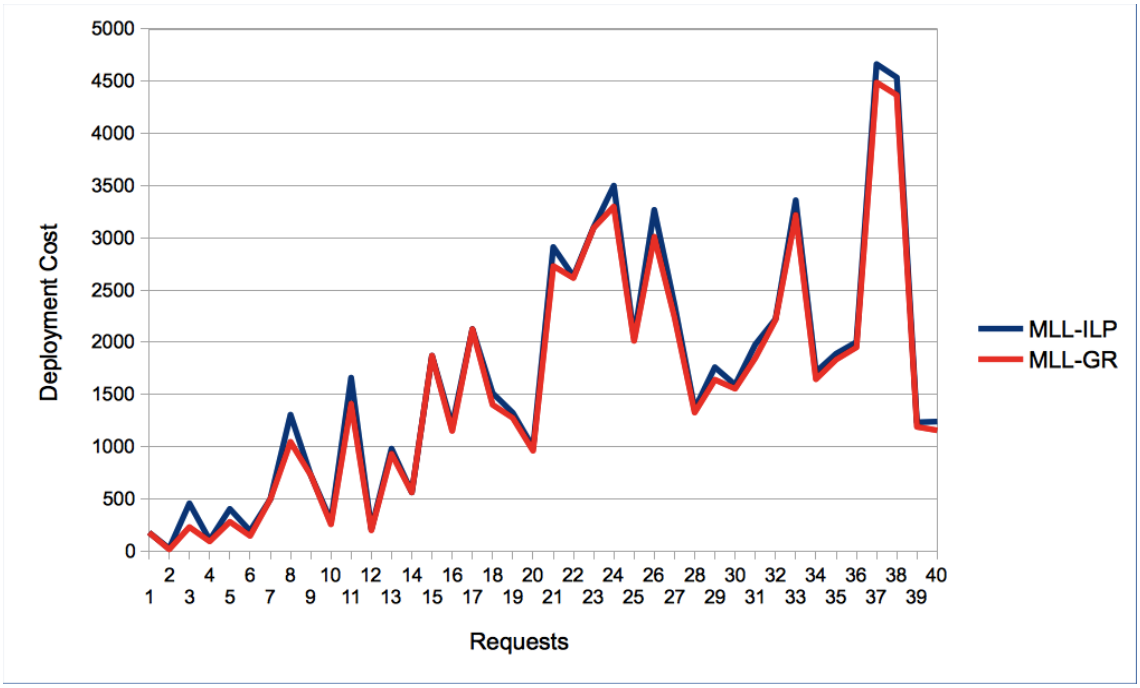


(a)

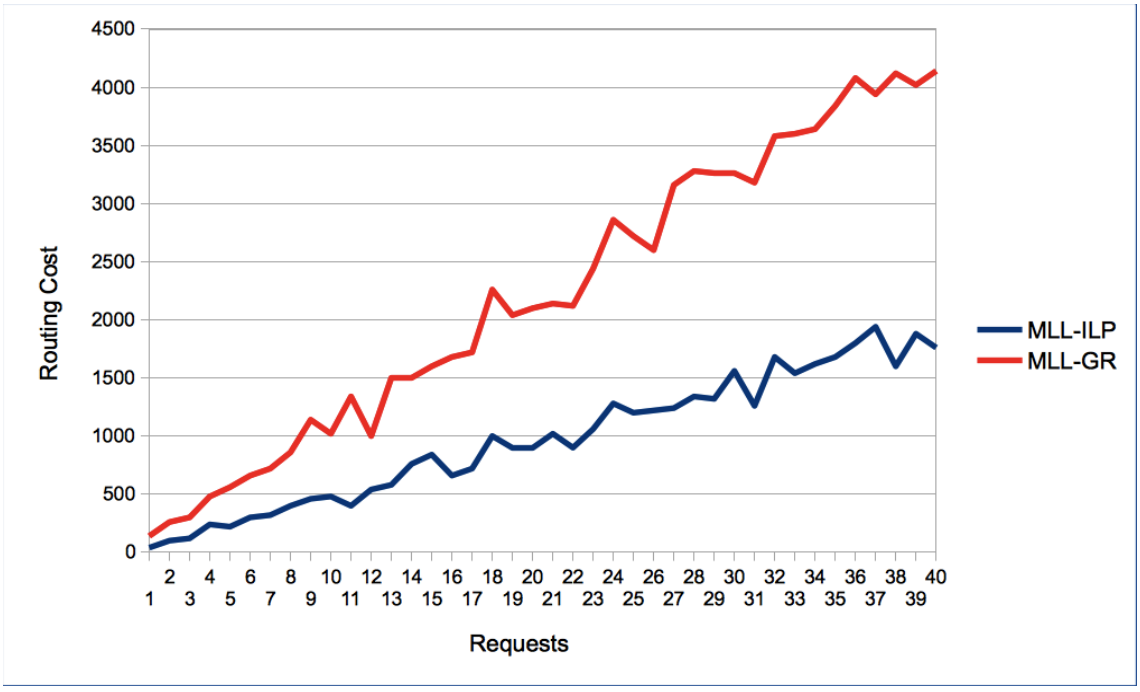


(b)

Figure 3.7: Third testcase: satisfied NFs: a) MLL-ILP and JRP-ILP, b) MLL-GR and JRP-GR



(a)



(b)

Figure 3.8: Third testcase: MLL schemes a) deployment cost, and b) routing cost

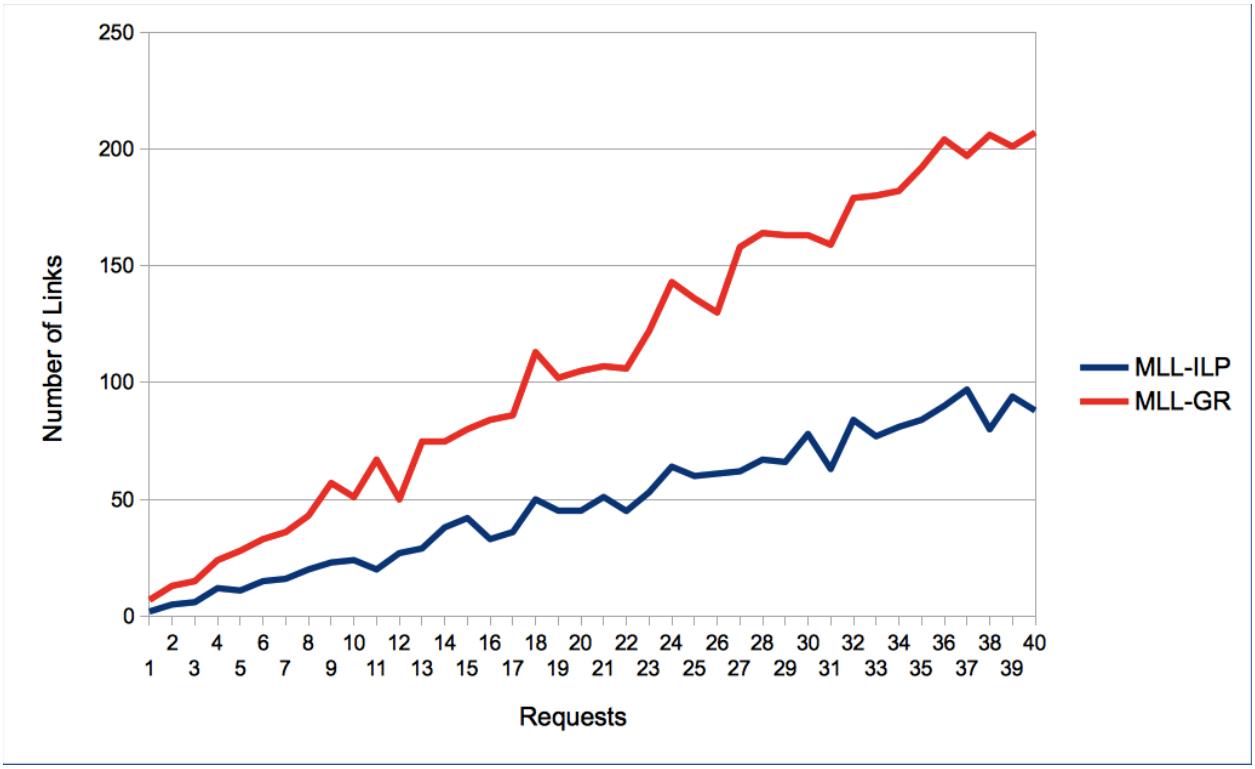


Figure 3.9: Third testcase: number of links - MLL-ILP and MLL-GR.

Chapter 4 Survivable Joint VNF Placement and Routing [1]

As noted in Section 2.3, there are no known studies on disaster-aware VNF provisioning, i.e., to handle multiple failures in large NFV-capable infrastructures These conditions can include power outages, *weapons of mass destruction* (WMD) attacks and a whole range of natural disasters. Given the critical nature of many business services, it is vital to address these concerns for emerging NFV-based services, i.e., network function placement and routing. Ideally, these solutions should incorporate some a-priori knowledge of potential failure (risk) regions in conjunction with cost reduction and resource efficiency concerns.

In light of the above, this chapter presents the first comprehensive modeling of multi-failure events/scenarios. Specifically, an expanded multi-objective ILP optimization scheme is presented for VNF placement and routing, termed as *"risk-aware" ILP* (RA-ILP). This solution tries to improve overall survivability as well as maximize the number of satisfied NFs and simultaneously minimize deployment costs, routing costs and link load. Furthermore, a greedy heuristic scheme is also introduced with the same objectives in order to improve scalability for larger, more realistic network scenarios, i.e., termed as *"risk-aware" greedy heuristic* (RA-GR) scheme. Finally, detailed testcase scenarios are evaluated to verify the survivability and cost performance of these two methods. Further details are now presented.

4.1 Notation Overview and Failure Model

The requested notation is introduced first. Overall, many of the variable definitions introduced in Section 3.1 are re-used here, i.e., for physical network, infrastructure, resources and NFV-related demands. Furthermore, a realistic *probabilistic* model is also presented here to capture large-scale disaster events yielding multiple highly-correlated spatial and temporal link failures. In particular, the failure model developed in [HL01] and further re-used in [FG01], is re-used here. Foremost, a set of pre-defined (a-priori) stressors is defined, detailed by set U. This set is comprised of a number of failure events $u \in U$, where each has a random, independent occurrence probability $p(u_n)$. Without loss of generality here, it is assumed that all stressor events are sufficiently rare and hence treated as mutually-exclusive, i.e., $\sum_{\forall u_n \in \mathbf{U}} p(u_n) = 1$.

Furthermore, each event u_n also has an associate risk region defined by a set of vulnerable links, i.e., *probabilistic shared risk link group* (p-SRLG or simply SRLG) [FG01]. Again, without loss of generality, it is assumed that each event in U is geographically non-overlapping, i.e., a given link in $G(V, E)$ can only be located in one particular SRLG. Finally, a conditional failure probability is also defined for each link a given region, i.e., $\omega(i, j)$, with respect to occurrence of stressor u_n. All these conditional link failure probability here are assumed to be independent [HL01].

Figure 4.1 shows a sample VNF placement example with three failure regions, U=$\{u_1, u_2, u_3\}$, based upon the earlier configuration in Figure 2.1. These regions are physically

Table 4.1: List of variables

Variable	Description
set U	Set of a-priori potential stressor events, where $U=\{u_1, u_2, ..., u_N\}$
u_n	Stressor event, i.e., $u \in U$
$p(u_n)$	Occurrence probability of failure event u_n
$\omega(i,j)$	Unconditional failure probability associated with link $(i,j) \in \mathbf{E}$ in the region of stressor event $u_n \in U$, $0 \leq \omega(i,j) \leq 100$.

disjoint and have specific failure probabilities associated with all links falling within each stressor (SRLG) region, i.e., $\omega(i,j)$, where $i,j \in E$. Accordingly, the proposed "risk-aware" VNF placement and routing schemes should now take into account these failure probabilities in order to improve reliability.

Overall, the above probabilistic model can be used to represent various types of multi-failure stressor events, i.e., power outage cascades, WMD attacks, etc. Now earlier studies in [SG02] have detailed the impact of large-scale nuclear EMP attacks, a particular form of WMD stressor. Namely, this work specifies the overall geometry of the resulting failure area caused by space-based detonations occurring at varying *heights of burst* (HoB) and warhead yield. The overall fallout is composed of a set of sub-areas with varying damage intensities, as shown in Figure 4.2. Based upon this, it is possible to further define the failure probabilities of unshielded electric components (network nodes and fiber optic links) in the different sub-areas. Hence the proposed testcases here specifically incorporate nuclear EMP stressor types.

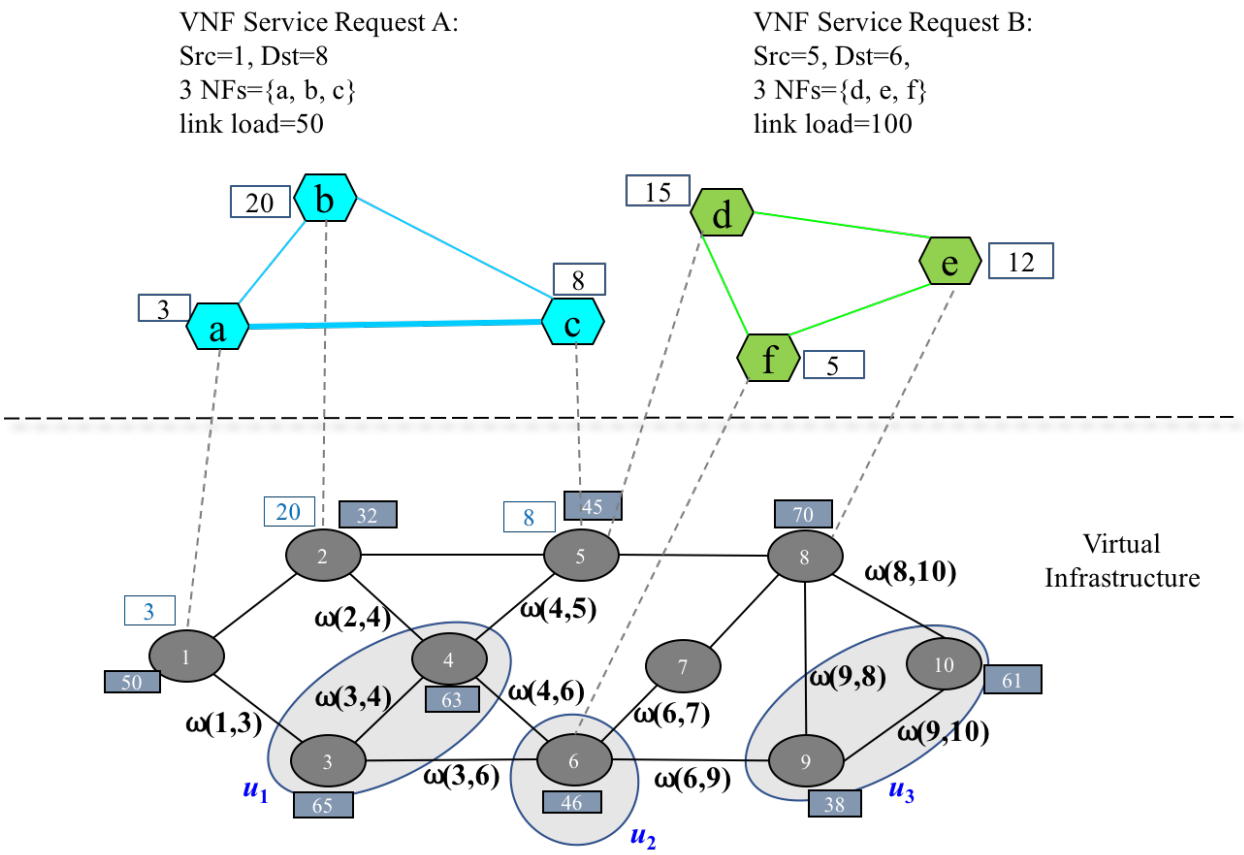


Figure 4.1: VNF request model in a probabilistic multi-failure scenario.

4.2 "Risk-Aware" Network Function Placement and Routing

As noted in Section 2.3, to the best of the authors' knowledge, there are no known studies on NFV provisioning under multiple correlated failures. Accordingly, an expanded ILP formulation is now developed to incorporate a-priori stressor events into the VNF placement and routing process to reduce downtime (due to failures) and maximize the number of satisfied requests, i.e., RA-ILP scheme. Next, a greedy heuristic scheme is also presented to overcome some of the scalability limitations of the optimization model, i.e., termed as

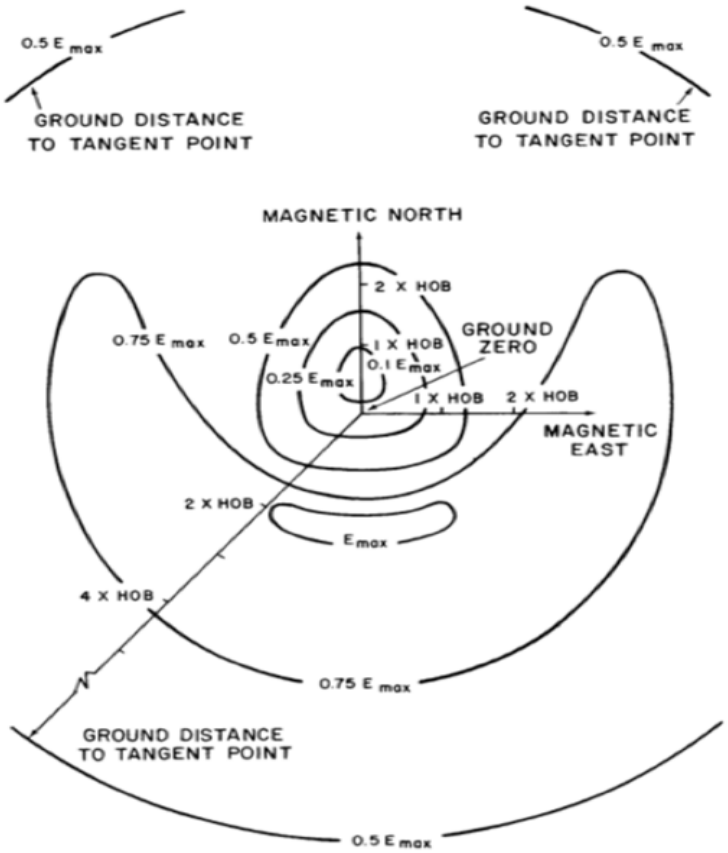


Figure 4.2: Electric fields for low-altitude EMP attacks, from [SG02].

RA-GR scheme.

4.2.1 "Risk-Aware" Optimization Model (RA-ILP)

The proposed "risk-aware" VNF routing and placement optimization scheme extends the MLL-ILP optimization model presented in Section 3.2 by directly incorporating link failure probabilities to the routing costs. In particular, a revised objective function is defined

as follows:

$$\begin{aligned} \max \text{ F} \quad = \quad & w_1 \sum_{r \in R} \sum_{i \in F_r} \sum_{d \in D | i \in F_d} x^i_{r,d} - w_2 \sum_{d \in D} \sum_{i \in F_d} c^i_d y^i_d \\ & - w_3 \sum_{r \in R} \sum_{(i,j) \in E} c^{ij} l^{ij}_r * (1 + \omega(i,j)) - \alpha w_4 \end{aligned} \tag{4.1}$$

subject to:

$$\sum_{d \in D} x^i_{r,d} \quad \leq \quad 1 \quad r \in R, i \in F_r \tag{4.2}$$

$$x^i_{r,d} \quad \leq \quad y^i_d \quad r \in R, i \in F_r, d \in D | i \in F_d \tag{4.3}$$

$$\sum_{i \in F_d} w^i_{d,j} y^i_d \quad \leq \quad w_{d,j} \quad d \in D, r \in R, j \in \{1, 2, ..., m\} \tag{4.4}$$

$$\sum_{r \in R} x^i_{r,d} \quad \leq \quad \lambda^i_d y^i_d \quad d \in D, i \in F_d \tag{4.5}$$

$$\sum_{j:(i,j) \in E} l^{ij}_r - \sum_{j:(j,i) \in E} l^{ji}_r = \begin{cases} -1; i = dst_r, src_r \neq dst_r \\ 1; i = src_r, src_r \neq dst_r \\ 0; \text{ otherwise. } i \in V, r \in R \end{cases} \tag{4.6}$$

$$\sum_{(d,j) \in E} l^{dj}_r \quad \geq \quad x^i_{r,d} \quad r \in R, i \in F_r, d \in D | i \in F_d \tag{4.7}$$

$$\sum_{r \in R} l^{i,j}_r b_r \quad \leq \quad \alpha b_{i,j} \quad \{i, j\} \in E \tag{4.8}$$

Specifically, the above objective function is very similar to that in Eq. 3.1. In particular, the first two terms are identical here, and focus on maximizing the number of satisfied demands and minimizing total deployment costs, respectively. However the third term in Eq. 4.1 modifies the total routing cost by scaling it in proportion to the link failure probability. Hence this term will favor lower risk paths. Meanwhile, the fourth term is also the same as per Eq. 3.1, i.e., for load-balancing purposes. Finally, all of the associated model constraints are also the same as those in the non-survivable ILP model presented in Section 3.2.

Furthermore, akin to the ILP optimization scheme, the "risk-aware" RA-GR heuristic scheme is also developed by extending the MLL-GR method in Section Figure 3.1. Namely, the previous algorithm in Figure 3.1 now uses a modified Dijkstra shortest-path computation approach. In particular, the link routing cost (weight) is now re-defined. In particular the routing cost for a path is now defined as a modified version of Eq. 3.9, which directly correlates link failure probability with static cost and load-balancing, as follows:

$$c'_{ij} = \sum_{(i,j)\in E} (c^{ij} + \frac{b_r}{b_{ij}}) * (1 + \omega(i,j)) \tag{4.9}$$

Specifically, the cost of using each link is now increased in proportion to its risk vulnerability, i.e., to favor less failure-prone routes. Hence the overall pseudocode description for the RA-GR scheme is identical to that for the non-survivable MLL-GR scheme shown in Figure 3.1, with the exception of the constrained function call (line 20, Figure 3.1). In particular, this procedure is now replaced by the "risk-aware" Dijkstra, termed as *"risk-aware"_constrained_Dijkstra*.

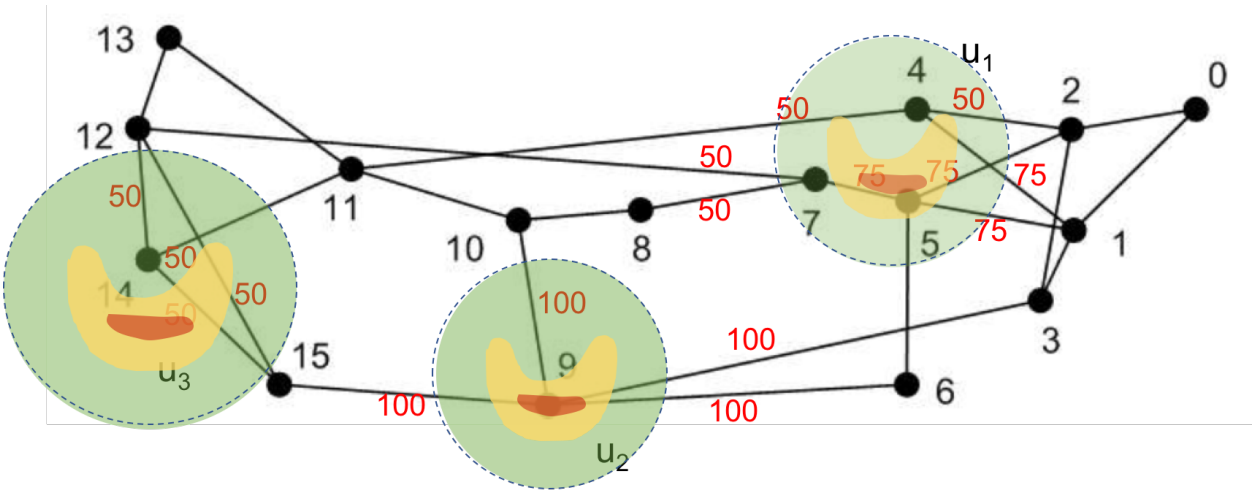


Figure 4.3: NSF network topology and the respective failure regions.

4.3 Performance Evaluation

The proposed risk-aware NF placement and routing optimization and greedy schemes are evaluated using the same infrastructure topology from Chapter 3, i.e., as shown in Figure 3.2. However, three potential stressor (risk) regions $u_n \in U$ are now superimposed here, as shown in Figure 4.3, i.e., u_1 comprised of links 9 links ($l_{4,1}$, $l_{4,2}$, $l_{4,11}$, $l_{5,1}$, $l_{5,2}$, $l_{5,6}$, $l_{5,7}$, $l_{7,8}$, $l_{7,12}$), u_2 comprised of 4 links ($l_{9,3}$, $l_{9,6}$, $l_{9,10}$, $l_{9,15}$), u_3 comprised of 4 links ($l_{14,11}$, $l_{14,12}$, $l_{14,15}$, $l_{12,15}$). The associated link failure probabilities are further modeled based upon EMP stressor attacks with 3 sub-areas as shown in green, yellow and red (representing 50, 75 and 100% of outage probability, respectively). Note that Figure 4.3 also shows the corresponding failure probability values next to each link, i.e., $\omega(i, j)$, for each SRLG region (note that these values are only used by the RA-ILP and RA-GR schemes). All survivability-related testcases also use the same set of parameters defined for the highly-resourced scenario in

Section 3.4, see Table 4.2. Furthermore, the earlier detailed (non-survivable) optimization and heuristic VNF placement and routing schemes from Chapter 3 are also tested here for comparison purposes, i.e., MLL-ILP and MLL-GR.

Table 4.2: Multi-failure testcases parameters

Parameter	**Idealistic Scenario**	**Realistic Scenario**
Link Resources	10,000	10,000
Node Resources ($w_{d,1}$,$w_{d,2}$,$w_{d,3}$)	$5,000$	$5,000$
Weight w_1	$1,000$	
Weights w_2	1	
Weights w_3	1	
Weights w_4	$1,000$	
Function set	f_0, f_1, f_2, f_3, f_4	
Required Resources ($w^i_{d,j}$)	$30 \leq w^i_{d,j} \leq 70$	
Function Setup Cost	50	
Instance Capacity	2	
Link Setup Cost	20	
Outage Event	u_1	Modified u_1
Outage Links	$(1,5)$, $(2,4)$, $(2,5)$, $l(5,7)$	$(4,11)$, $(5,7)$, $(7,12)$

4.3.1 Pre-Fault Performance

Initial tests are done to gauge the (pre-fault) provisioning costs of the survivable "risk-aware" schemes. Specifically, all schemes are compared here in terms of the number of satisfied NFs, mapping/deployment and routing costs (under working non-failure conditions). The request batch sizes are varied from 1-60 requests, with each request demanding 4 NFs, i.e., the total number of NFs ranges from 4 to 240. Now carefully note that all of the schemes try to maximize the number of satisfied NFs. As a result, the first performance test analyzes the ability to place and route all sets of requests, and the findings confirm that

all schemes are successful here. Subsequently, further analysis is done to compare overall cost-related performances. In particular, the deployment and routing costs are shown first for all schemes in Figure 4.5. Foremost, Figure 4.5(a) confirms that deployment costs are mostly similar. However the results in Figure 4.5(b) indicate that the routing costs with the greedy heuristic schemes (RA-GR and MLL-GR) are much higher than their optimization counterparts. Specifically, each greedy heuristic method yields over twice the cost of its ILP optimization counterpart, i.e., RA-GR scheme is about 2.3 times higher than the RA-ILP and MLL-GR schemes is about 2.3 times higher than the MLL-ILP scheme. Furthermore, the same plot also shows that the routing costs for the ILP optimization schemes are very similar. In fact, the costs of the RA-ILP method range between 0-4% higher than MLL-ILP, which shows that survivable schemes yield slightly longer paths in order to avoid paths with higher failure probabilities.

4.3.2 Post-Fault Performance

Post-fault performance of the "risk-aware" schemes is now evaluated in order to gauge the impact of a set of link failures. Namely, a multi-failure (EMP disaster) scenario is randomly selected and triggered by choosing one of the stressor (risk) regions in Figure 4.3. In particular, the large u_1 region is chosen here as it covers the most number of links. Now clearly, it is very difficult to predict the location/impact of a-priori failure events in advance, especially large-scale disasters. Hence in practice, pre-defined/a-priori risk regions will rarely match the actual failure footprints seen in the field. As a result, two different disaster

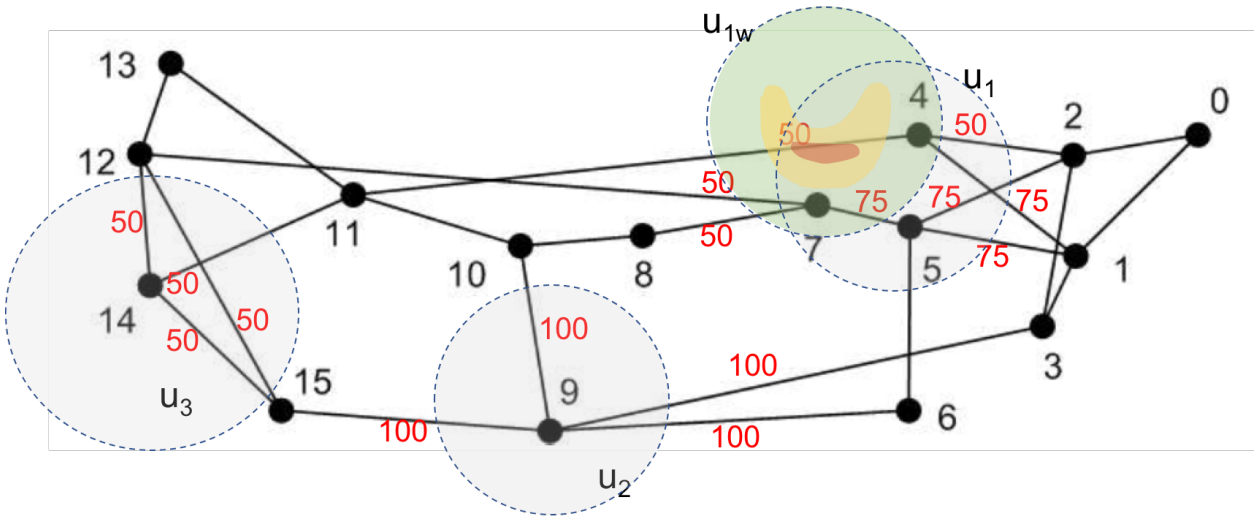


Figure 4.4: NSF network topology and the respective failure regions.

testcases are evaluated here, i.e., idealistic and realistic. The former assures perfect/exact knowledge of failure regions and only (randomly) fails links within the chosen pre-defined stressor region, i.e., u_1. Meanwhile the latter assumes an attack epicenter shifted slightly to the west, i.e., resulting in some failed links falling outside the pre-defined stressor region u_1, as shown in Figure 4.4.

Results are presented for the idealistic stressor scenario first. In particular, 4 links in region u_1 are failed, as noted in Table 4.3, i.e., approximately 16% link failures out of a total of 25 links. Now due to computational scalability limitations, the optimization schemes here can only handle smaller networks. As a result, their associated performances are presented separately from the greedy heuristic schemes. In particular, Figure 4.6(a) plots the number of requests that cannot be satisfied after the stressor event for the two optimization schemes, i.e., failure rates for MLL-ILP (non-survivable) and RA-ILP. These results show that the

risk-aware RA-ILP scheme reduces overall failures between 10-20% for all input batch sizes. Similarly, Figure 4.6(b) plots failed demand requests for the two greedy heuristic schemes. These results show a much larger separation, with the "risk-aware" RA-GR scheme giving up to 50% lower failure rates.

Now, carefully note the results in Figure 4.6 assume that a single link failure disrupts an entire request path. However, the accuracy of the proposed risk-aware models can also be gauged further by computing a link failure ratio, defined as

$$fr_{rn} = \frac{100 fl_{rn}}{total_{rn}} \tag{4.10}$$

where fl_{rn} is the number of failed links and $total_{fn}$ is the total number of links within a request path. In particular, this ratio provides a measure on the required level/complexity to (re)establish a failover path. Overall, the results in Figure 4.7(a) shows that the failure ratio follows the same pattern as that for the request failures, Figure 4.6. For example, the MLL-ILP optimization scheme incurs slightly higher link failure ratios than the RA-ILP scheme, i.e., 25%. However, non-survivable MLL-GR scheme introduces much higher failure ratio than the RA-GR method, i.e., up to 75%.

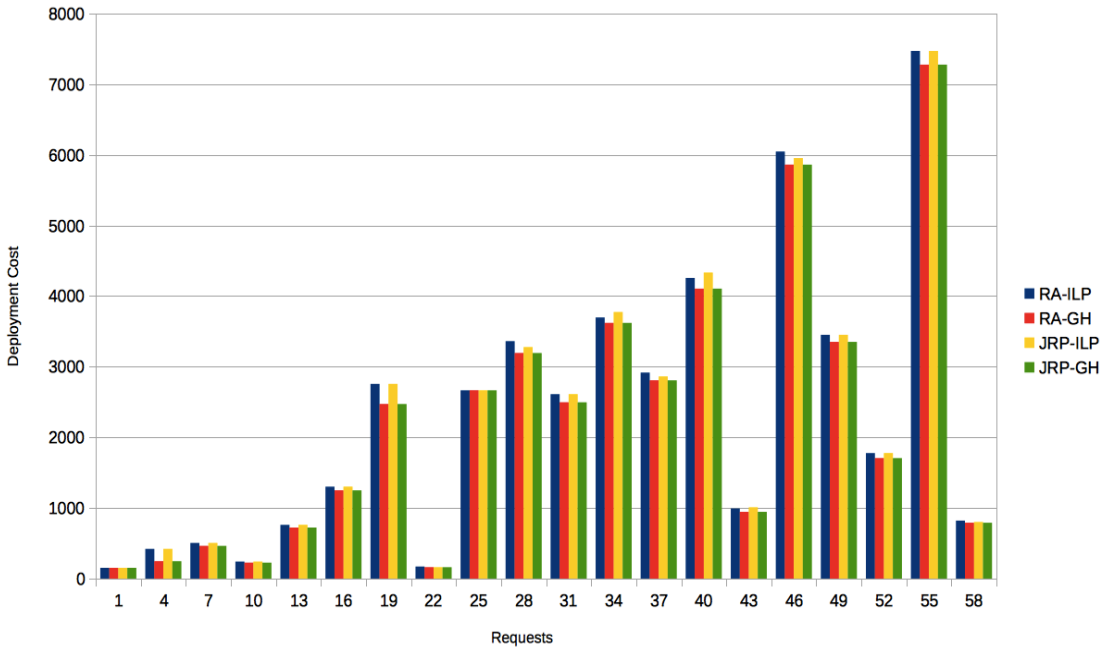
Meanwhile, as noted earlier, the realistic stressor scenario shifts the location of the actual outage event slightly west, as shown in Figure 4.4. As a result, 3 links are now failed, ie., 12% of total. Overall, these results mirror the findings in Figure 4.6. For example, Figure 4.8(a) compares the request failures for both of the ILP optimization schemes, and shows that the risk-aware RA-ILP scheme is less effective in reducing failure rates over the

MLL-ILP scheme. In particular, both schemes give almost identical outages here, with the exception of a few batch sizes for which the RA-ILP performs slightly better, i.e., up to 15%. These results show that the lack of accuracy in a-priori risk regions has a notable impact on ILP-based survivability schemes. The corresponding results for the heuristic schemes in Figure 4.8(b) also illustrate much closer failure rates between the non-survivable (MLL-GR) and survivable (RA-GR) schemes. However, the risk-aware RA-GR scheme gives notably better survivability, i.e., 23% less failures.

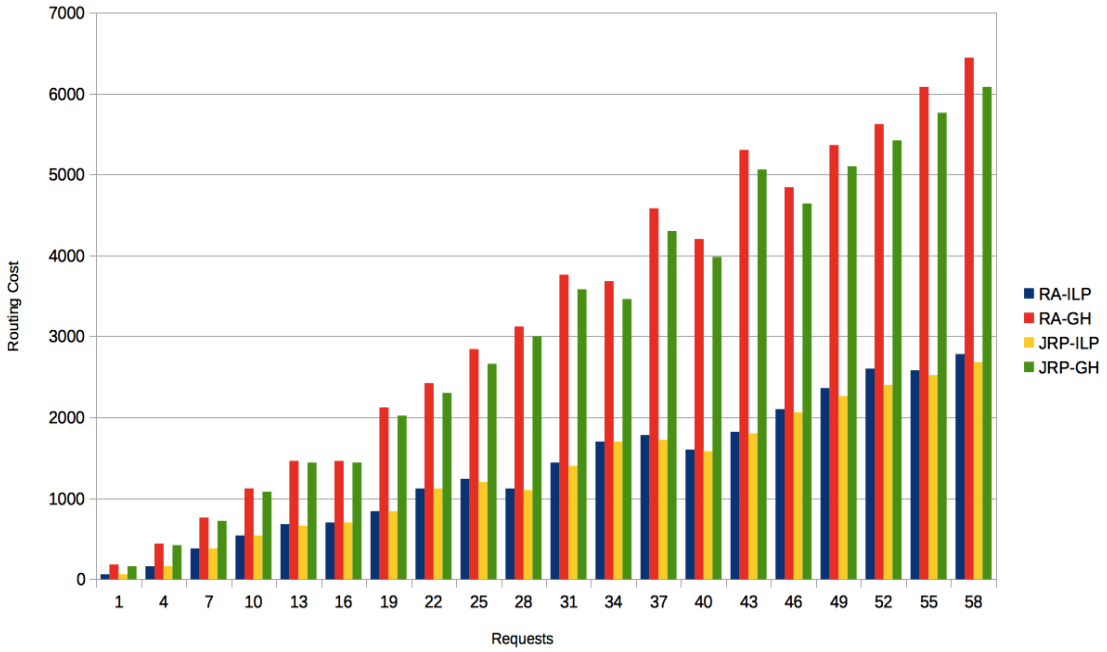
Finally, failure ratio (Eq. 4.11) results are also plotted in Figure 4.9. Unlike the request failure rates, here some improvement is still observed with the "risk-aware" strategies.

$$fr_{rn} = fl_{rn} * 100/total_{rn} \tag{4.11}$$

Clearly, introducing failure risk information into the VNF placement and routing process is very beneficial here, since deployment and routing costs have no considerable variation when this model is adopted. However, the number of discontinued services and links in a multi-failure scenario are considerably lower.

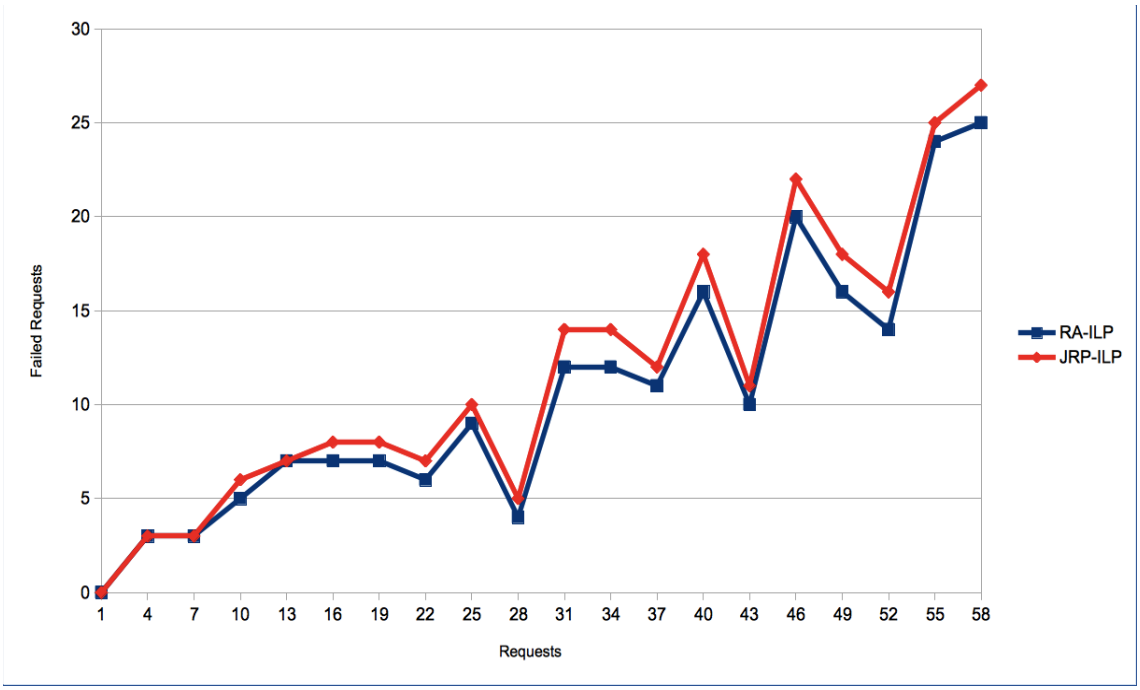


(a)

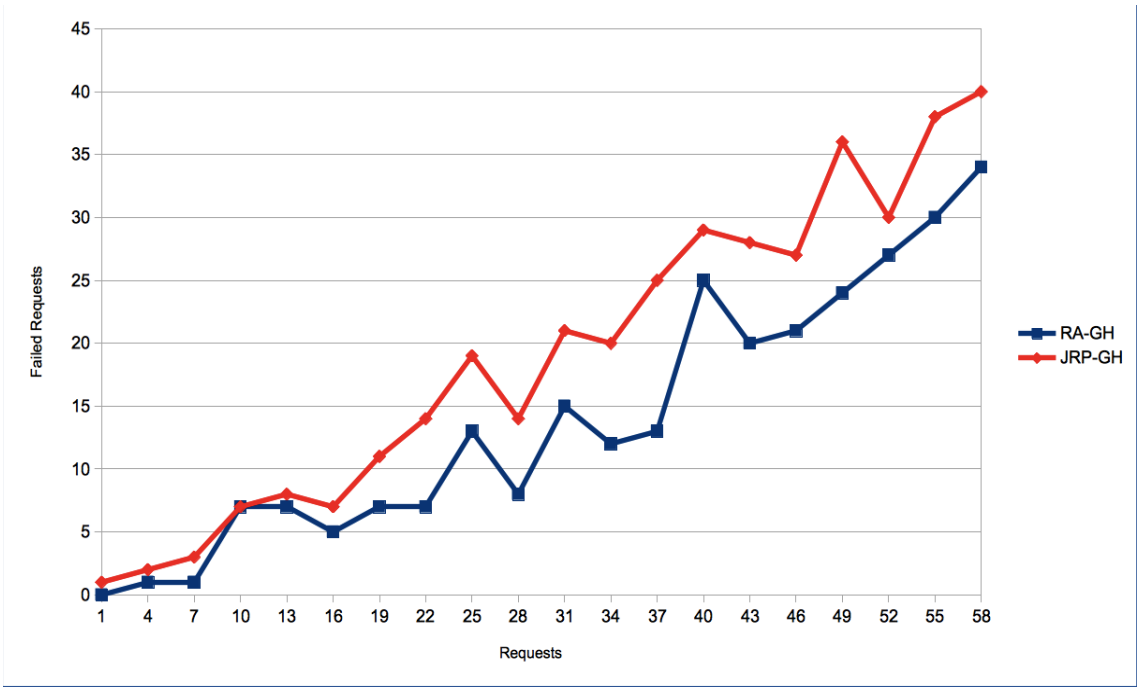


(b)

Figure 4.5: a) deployment costs, and b) routing costs

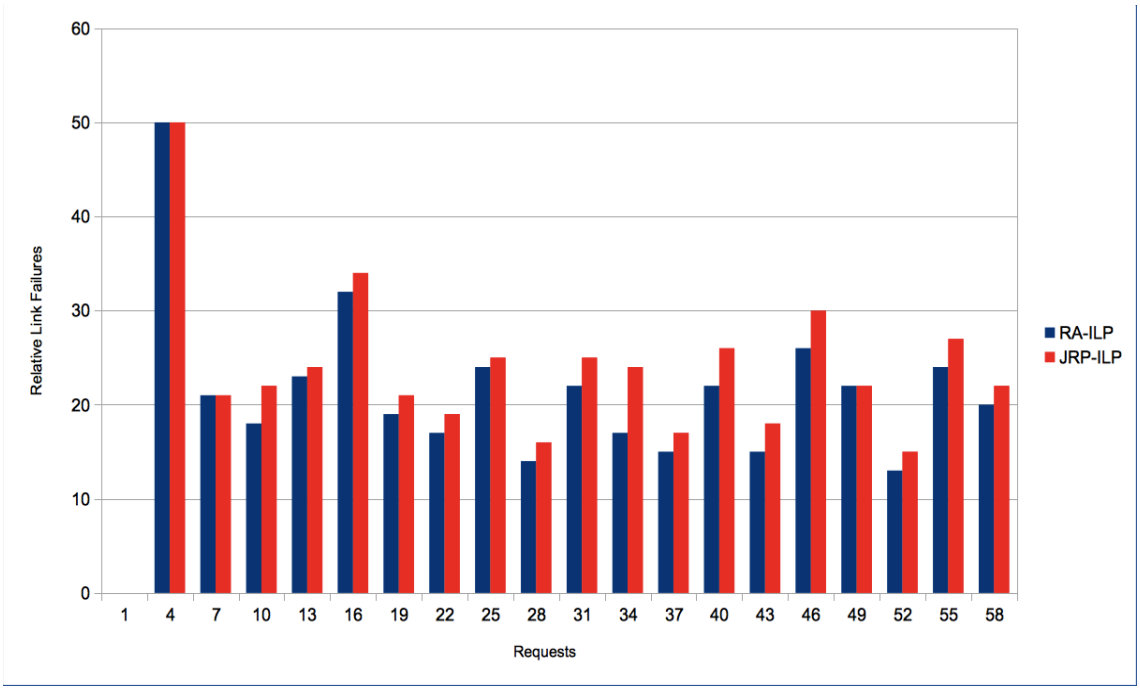


(a)

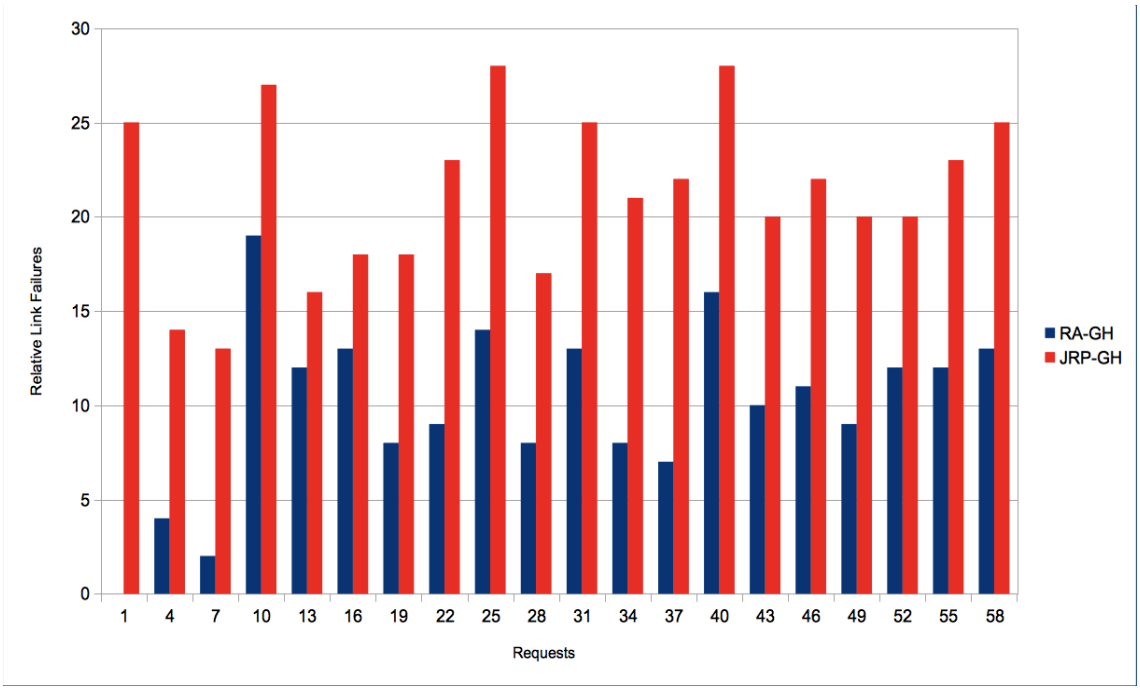


(b)

Figure 4.6: Requests assignment failures for the idealistic stressor scenario) RA-ILP and MLL-ILP, and b) RA-GR and MLL-GR

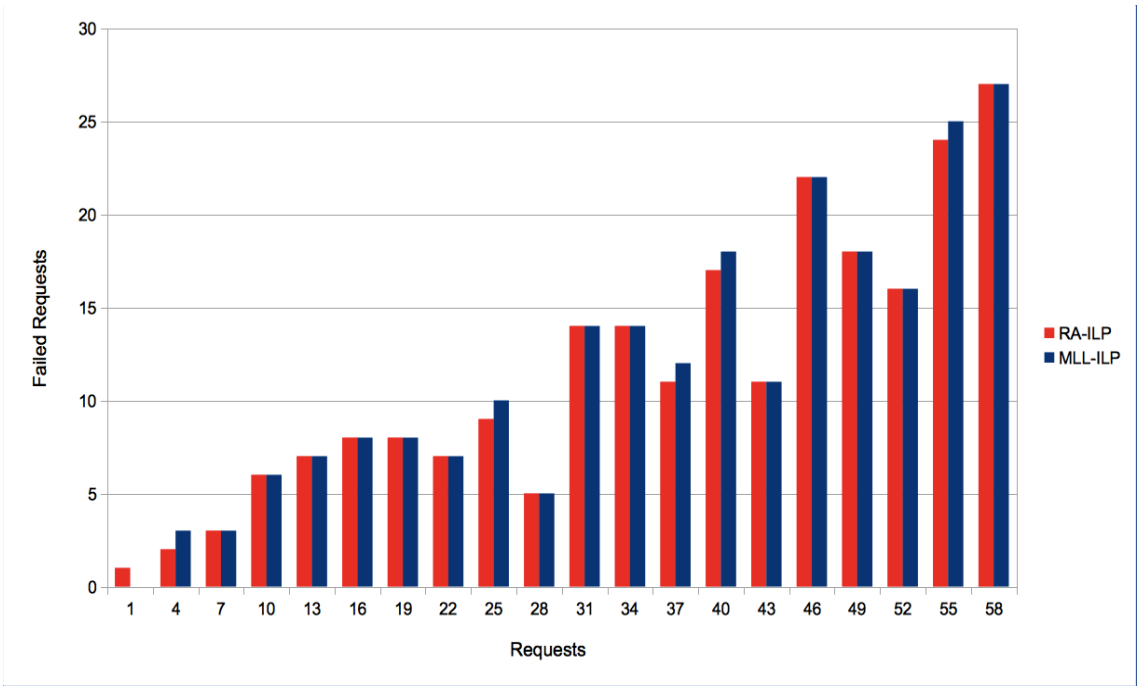


(a)

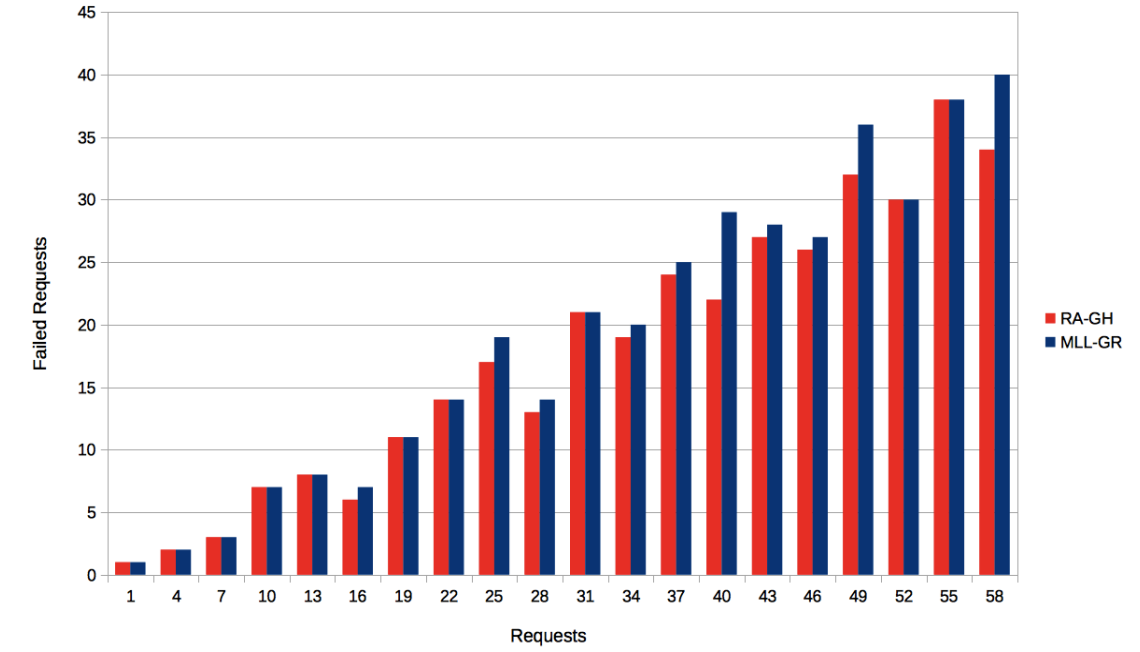


(b)

Figure 4.7: Link failure rates for the idealistic stressor scenario) RA-ILP and MLL-ILP, and b) RA-GR and MLL-GR

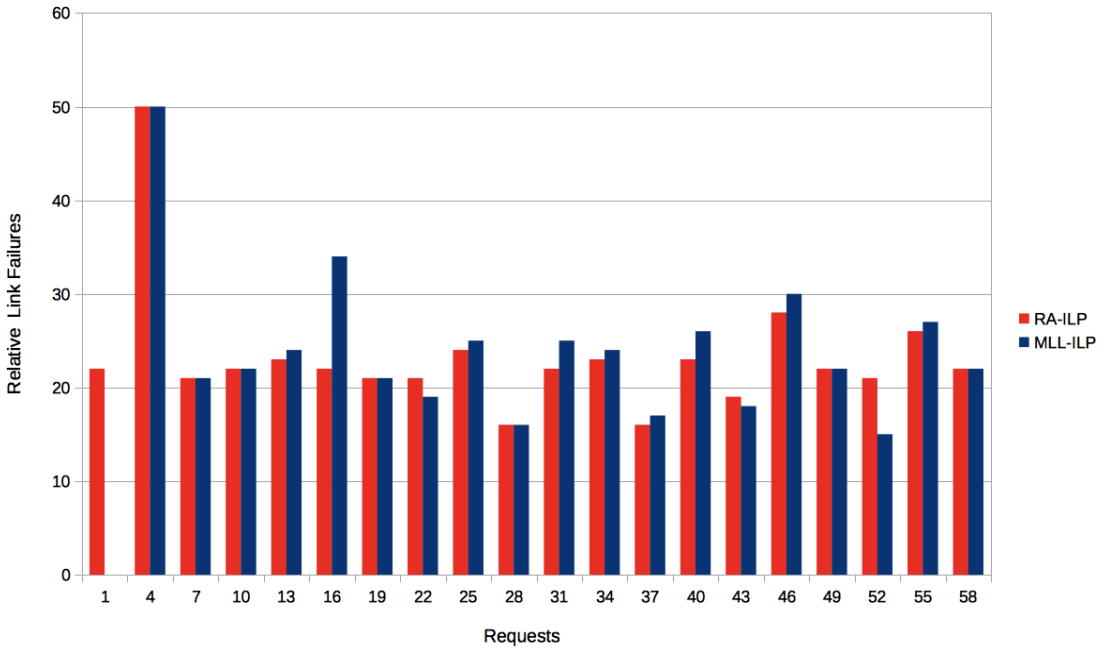


(a)

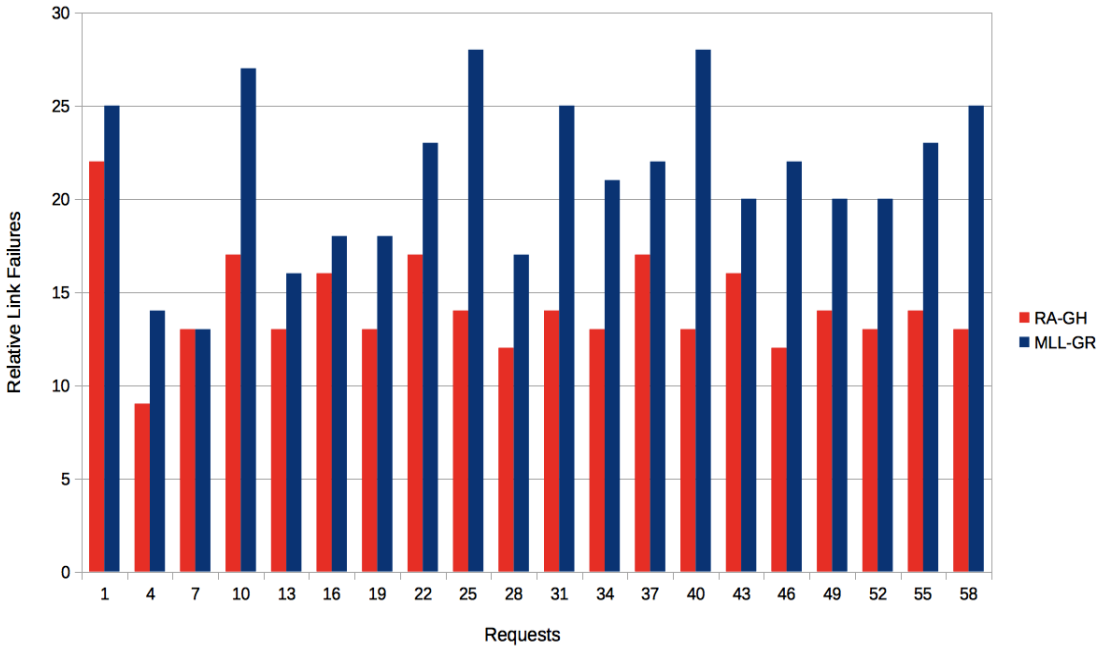


(b)

Figure 4.8: Requests assignment failures for the realistic stressor scenario a) RA-ILP and MLL-ILP, and b) RA-GR and MLL-GR



(a)



(b)

Figure 4.9: Link failure rates for the realistic stressor scenario a) RA-ILP and MLL-ILP, and b) RA-GR and MLL-GR

Chapter 5 Multi-Objective Metaheuristic Scheme for Large-Scale Networks [1]

Overall, the (MLL-ILP) optimization scheme presented in Chapter 3 is quite effective in addressing a range of NFV provisioning objectives for network operators. However as noted previously, VNF provisioning is a NP-hard problem which poses increased computational complexity as the network size increases, i.e., intractable for larger realistic scenarios. Moreover, this complexity is further exacerbated for the case of NFV survivability presented Chapter 4. As a result, the RA-ILP optimization scheme may not significantly reduce request blocking rates as compared to its non-survivable counterpart.

To address these concerns, this chapter presents a novel metaheuristic solution for survivable VNF placement and routing. This scheme uses a *genetic algorithm* (GA) approach to tackle larger network sizes and is termed as the *risk-aware genetic algorithm* (RA-GEN) scheme [DO03][DO04]. Akin to the earlier-detailed optimization (RA-ILP) and heuristic (RA-GR) methods, this GA-based approach also supports multiple provisioning objectives. The proposed solution also tested and compared for a range of deployment scenarios.

5.1 Overview of Genetic Algorithm Metaheuristic

Before presenting the metaheuristic VNF provisioning scheme, it is instructive to review the high-level nature of the GA search algorithm. In particular, this methodology

is based upon the principle of natural selection/evolution, which states that living beings will naturally evolve through a continual recombination and mutation of their chromosomes. Namely, in the GA approach an individual represents an overall solution and is composed of a set of chromosomes. Furthermore, each chromosome here represents a possible solution to a given problem, i.e., value of variable(s). As a result, an associated fitness value is also defined to measure how efficient its solution is. Finally a set of individuals is combined to build a population in which the process of natural selection is applied [JM01].

A high-level view of the GA method is shown in Figure 5.1 and consists of a series of steps. Foremost, the solution starts by creating a population, P, composed of a set of N individuals, i.e., $n \in P$ (*Initialize Population*, Figure 5.1). Each individual here is initialized by randomly creating and assigning values to its j chromosomes, where each chromosome represents a possible specific solution, i.e., usually binary. Next, the fitness value (*Compute Fitness*, Figure 5.1) is computed for each individual and if a satisfactory result is found, this individual is selected as the best solution and the algorithm is terminated. However, if the initially-created individuals in P do not efficiently solve the problem, new individuals are further created. Specifically, two individuals (parents) are selected from the original population (*Select Parents*, Figure 5.1) and a child individual is created and initialized by inheriting the parental chromosomes (*Crossover*, Figure 5.1). In particular, this inheritance is controlled by a *crossover rate* which defines how many chromosomes should be inherited from both the father and the mother. Furthermore, a mutation can also be

generated (*Mutation*, Figure 5.1) based upon a *mutation rate*, which defines whether any chromosomes values are changed or not. Accordingly, the fitness value is (re)computed for each new child, and this process is repeated until a termination condition is reached, i.e., either in terms of the number of iterations, appropriate fitness values, etc. Carefully note that due to the random nature of the GA approach, it is not possible to run the algorithm until an global optimal solution is found (although it is possible to find such a solution here).

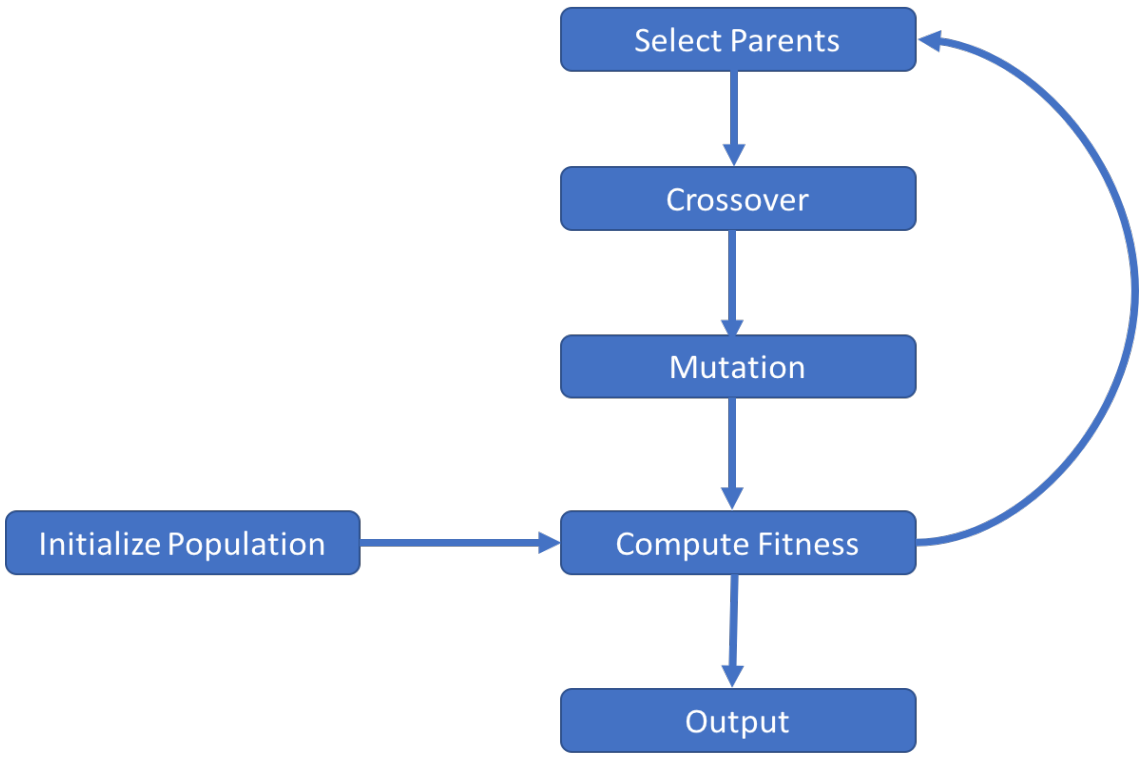


Figure 5.1: Genetic algorithm (GA) flow chart

5.2 Genetic Algorithm Approach for NFV Provisioning

The GA approach is now applied to the survivable multi-objective VNF provisioning problem, i.e., VNF placement, routing, load-balancing and link survivability. This meta-heuristic scheme (RA-GEN) requires some specializations and adaptations to the generic high-level GA solution outlined in Section 5.1. Full details are now presented.

5.2.1 Notation Overview

Before detailing the RA-GEN solution, the related notation and variables are first defined. Overall, this framework re-uses the same notation from Chapter 3 to represent the overall graph and input requests. Some additional variables are also introduced for GA, operation and these are summarized in Table Table 5.1. Foremost, the overall population set is defined by the set P and consists of pop_size individuals. Furthermore, each individual $ind \in P$ has a set of chromosomes, where $ind_{r,i}$ is the (r, i) chromosome addressing the datacenter that instantiates function i requested by request r. The fitness function for an individual is also given by FX_{ind}.

Now, with regards to the population, $ind_{overall}$ is defined as the best overall individual across all populations, i.e., in terms of the fitness function. Similarly, ind_{best} is defined as the best individual in a given population. Meanwhile, the set of children generated by the parents in a population P is given by C, and $c \in C$ is an individual child. The first child is also denoted by c_0. Finally, T individuals (parents) are chosen from the main population to generate the children, and this subset is given by $T_c \subseteq P$, i.e., $|T_c|=|\tau| \leq |P|$. The father

(mother) of an individual is also given by c_f (c_m), see Table 5.1. Also, SP_{ind_r} and SP_{c_r} represent the shortest paths computed by individual $ind_r \in P$ and $c \in C$, respectively, to satisfy request r.

Table 5.1: List of variables

Variable	**Description**
set P	Population with N individuals, where P=$\{1, 2, ..., n\}$
ind	Individual, i.e., $ind \in P$
$ind_{r,i}$	Individuals' chromosome (r, i) addressing the datacenter that instantiates function i requested by request r
$d(ind_{r,i})$	Datacenter d defined by individual ind to instantiate function i requested by request r
SP_{ind_r}	Shortest path computed by individual ind to satisfy request r
FX_{ind}	Fitness value of individual ind
$ind_{overall}$	Best overall individual across all populations
ind_{best}	Best individual in a given population P
C	Child population generated from parent population P
c	Child individual $c \in C$
c_0	First individual c of a children population C
τ	Tournament size
$rate_m$	Mutation rate
$rate_c$	Crossover rate
T_c	Subset of τ individuals selected from parent population P ($T_c \subseteq P$) to generated children
c_f	Father individual selected from T_c
c_m	Mother individual selected from T_c
SP_{c_r}	Shortest path computed by individual c to satisfy request r
pop_size	Population size
$Npop$	Number of children populations evaluated

5.2.2 "Risk-Aware" Genetic Algorithm (RA-GEN) Scheme

The overall pseudocode for the RA-GEN scheme is presented in Figure 5.2 and consists of two key stages, i.e., initialization and selection. Here the first stage starts by ini-

tializing all GA search variables (line 3, Figure 5.2) and generating an initial population of randomly-selected individuals, P. Specifically, each individual randomly assigns datacenters to instantiate each function i (requested by each request r) and also computes the shortest path between the request source and destination nodes (lines 4-11, Figure 5.2). As a result, each individual *ind* has $|R| \cdot |F|$ chromosomes, where $|R|$ is the number of requests and $|F|$ is the number of functions per request. Furthermore, each individual chromosome stores the datacenter being randomly assigned for function i requested by request r, and it is assumed that this datacenter must be able to instantiate the specific function (line 7, Figure 5.2). Finally, a connection path is also provisioned between the source and destination nodes by routing through all the datacenters supporting the (randomly mapped) VNFs. Note that all path computation here is done using a "risk-aware" Dijkstra scheme, i.e., *risk_aware_constrained_Dijkstra*, where link weights are computed based upon their failure probability, $\omega(i,j)$, as follows:

$$c'_{ij} = \sum_{(i,j)\in E} (c^{ij} + \frac{b_r}{b_{ij}}) * (1 + \omega(i,j)) \tag{5.1}$$

akin to Eq. 4.9, (Chapter 4). The fitness function value FX_{ind} is also computed for each randomized individual, and the one with the highest value is chosen as the best solution (lines 12-13, Figure 5.2). After the initial population is complete, the second stage is launched to perform selection. Namely, this phase iterates to generate and test a given number of child populations (*Npop* in total). Now in order to ensure that the best solution from the parent population P is included in the child population, the first child c_0 is chosen as the best

individual in P, i.e., $c_0 = ind_{best}$ (line 16, Figure 5.2). The remaining individuals are then created by inheritance, i.e., crossover and mutation (lines 17-22, Figure 5.2). In particular, each child, c, is generated by running a tournament stage, where potential individuals "compete" to become parents. Namely, a subset T_c of tournament size individuals is selected for each child by randomly selecting τ individuals from population P (line 19, Figure 5.2). The best and second best individuals from T are then chosen as the parents, i.e., father c_f and mother c_m (lines 20-21, Figure 5.2). Next, the crossover rate, $rate_c$, is used to weight the chromosome inheritance between the two parents. For example, if the crossover rate is 20%, then 80% of the chromosomes are inherited from the father and 20% from the mother (line 22, Figure 5.2). Finally, the child's chromosomes are further mutated according to the mutation rate $rate_m$. Specifically, a random value is generated and compared with the mutation rate, and if it is lower, then the chromosome is replaced by the total number of datacenters D minus the current chromosome value $c_{r,i}$. For example, consider a network composed of 16 datacenters, with a chromosome assigning datacenter 5 to instantiate a specific function. If this particular chromosome is selected for mutation, then its value is modified (16-5), i.e., changed to 11. In order to ensure the new datacenter has enough resources, after inheritance and mutation are complete, the algorithm must also check to make sure that this newly-assigned datacenter (chromosome value) can instantiate function i requested by request r, otherwise these steps are repeated.

Finally, each child individual, c, computes the "risk-aware" shortest path SP_{c_r} for each request r (line 25, Figure 5.2). The overall fitness value FX_c is then evaluated for each child individual c (line 26, Figure 5.2), and akin to the parent population stage, the best individual, ind_{best}, from the child population C is selected (line 27, Figure 5.2). Finally, the best overall individual is then selected across all ($Npop$) iterations as the final VNF mapping and routing selection. Namely, this selection is done by comparing the best individual in each child population iteration, ind_{best}, versus the currently known best overall individual, $ind_{overall}$. If the former is better (higher fitness value) than the latter, the best child individual is appropriately updated.

Overall, the second stage (lines 15-31, Figure 5.2) generates and evaluates a number ($Npop$) of child populations which are created by inheriting chromosomes from their respective parents. Note that each child population may not be derived from the best individuals in the parent population. However the child individuals still improve upon each iteration since the their parents are efficiently selected during the tournament stage. In summary, the best overall individual, $ind_{overall}$, returns the complete VNF demand mapping and its set of routes, i.e., including each datacenter d instantiating function i (requested by request r), the number of instances y_d^i of function i in datacenter d, the links l^{ij} routing the paths and the maximum overall link load α.

5.3 Performance Evaluation

The proposed RA-GEN metaheuristic scheme is evaluated using the same infrastructure topology and probabilistic stressor regions as shown in Figure 4.3. All testcase parameters are presented in Table 5.2 (same as those in Table 4.2), and the parameter settings for the RA-GEN scheme are also given in Table 5.3. In particular, GA selection is done over 100 populations, with each having 20 individuals and crossover/mutation rates of 20%. Overall, the goal here is to compare the RA-GEN metaheuristic solution with the methods presented in Chapter 4, i.e., RA-ILP and RA-GR schemes. Hence the same testcases used in Chapter 4 are also adopted here, i.e., idealistic and realistic stressor scenarios.

Table 5.2: Multi-failure testcases parameters

Parameter	**Idealistic Scenario**	**Realistic Scenario**
Link Resources	10,000	10,000
Node Resources ($w_{d,1}$,$w_{d,2}$,$w_{d,3}$)	$5,000$	$5,000$
Weight w_1	$1,000$	
Weights w_2	1	
Weights w_3	1	
Weights w_4	$1,000$	
Function set	f_0, f_1, f_2, f_3, f_4	
Required Resources ($w^i_{d,j}$)	$30 \leq w^i_{d,j} \leq 70$	
Function Setup Cost	50	
Instance Capacity	2	
Link Setup Cost	20	
Outage Event	u_1	Modified u_1
Outage Links	$(1,5)$, $(2,4)$, $(2,5)$, $l(5,7)$	$(4,11)$, $(5,7)$, $(7,12)$

Table 5.3: GA-related parameters (RA-GEN scheme)

Variable	Description	Value
pop_size	Population size	20
$loop$	Number of iterations	100
τ	Tournament Size	4
$crossover$	Crossover rate	20%
$mutation$	Mutation rate	20%

5.3.1 Pre-Fault Performance

Akin to Section 4.3.1, initial tests are done to commensurate and compare the pre-failure provisioning costs of the metaheuristic RA-GEN scheme. Namely, varying incoming batch sizes (with 1-60 requests) are tested here, with each request demanding 4 NFs. Overall, Figure 5.3(a) plots the results for setup (deployment) costs, and these findings show relatively close performance between all three schemes. However, the associated routing costs are also plotted in Figure 5.3(b) and indicate that the GA-based metaheuristic gives notably better performance than the greedy heuristic, i.e., 50% lower routing costs across all batch sizes. Nevertheless, the RA-ILP optimization model still outperforms the RA-GEN scheme here, reducing routing costs by up to 35%.

5.3.2 Post-Fault Performance

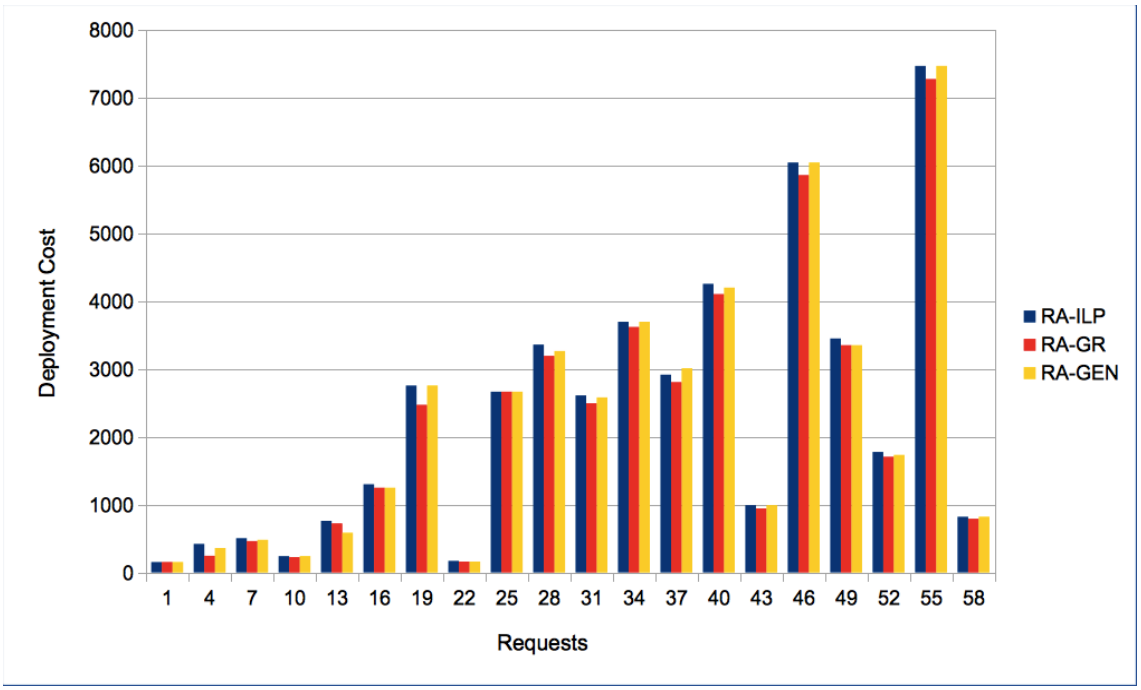
To further gauge the impact of multiple link failures, the post-fault performance of all "risk-aware" schemes is also evaluated. Akin to Section 4.3, two different testcases are evaluated here i.e., idealistic and realistic stressors scenarios. As per Table 5.3, the former testcase selects the u_1 region and fails 4 links here ($l_{1,5}$, $l_{2,4}$, $l_{2,5}$, $l_{5,7}$). Accordingly,

Figure 5.4(a) plots the number of failed requests for the ideal stressor and indicates the higherst amount of failures with the greedy RA-GR scheme. By contrast, the RA-GEN method is very competitive with the RA-ILP optimization scheme here, i.e., generally within 20% of the number of failures in for most batch sizes. In fact, the metaheuristic solution even gives lower failures for some batch sizes, i.e., 4, 7, 16, 37, 46 and 55 requests.

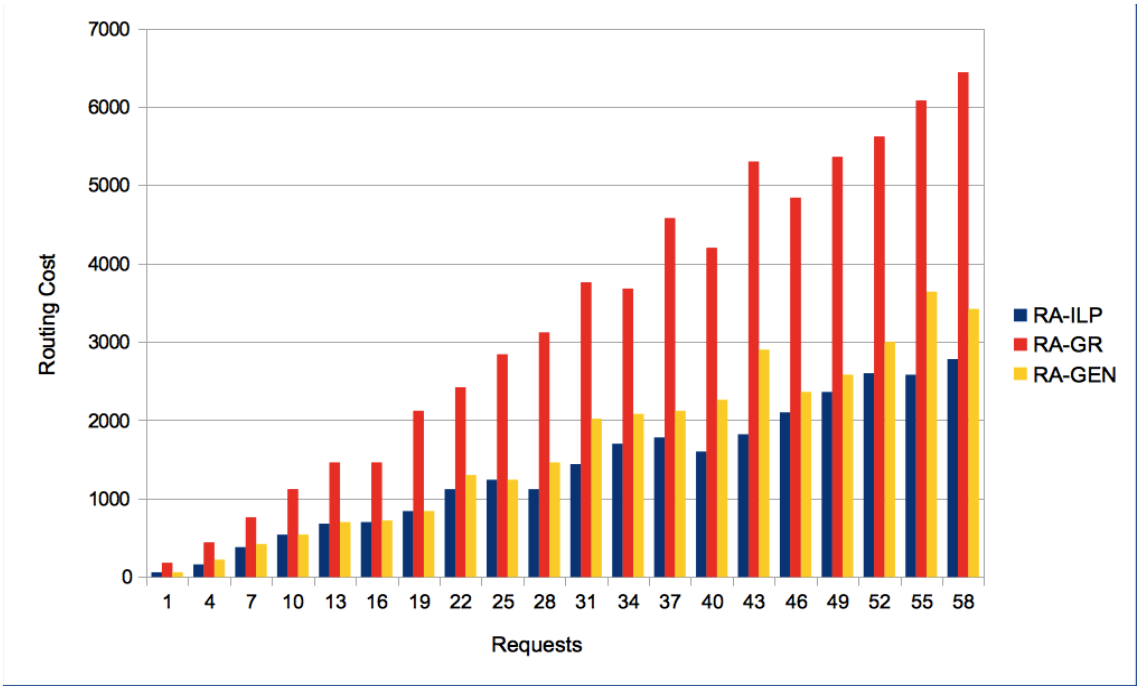
Meanwhile the realistic stressor testcase selects the u_{1_w} region (Figure 4.4) and fails 3 links ($l_{4,11}$, $l_{5,7}$ and $l_{7,12}$). The related post-fault failure results are plotted in Figure 5.4(b) and indicate that the RA-GEN metaheuristic actually gives improved survivability than both of the other RA-ILP and RA-GR schemes. Most notably, failed requests are up to 20% lower than the optimization scheme, a very notable result.

1: INPUT (Network infrastructure and demands): G(V, E), $c^{ij}, \omega(i,j) \forall (i,j) \in E, R, F, D$
2: OUTPUT: $x^i_{r,d}, y^i_d, l^{ij}_r$ and SP_{c_r} values from best overall individual $ind_{overall}$
3: set $ind_{overall} = MAX\_VALUE, x^i_{r,d} = 0, y^i_d = 0, l^{ij}_r = 0$ for all $r \in R, i \in F_r, d \in D, (i,j) \in E$
{BEGINNING OF INITIALIZATION STAGE}
4: **for** all $ind \in P$
5: **for** all $r \in R$
6: **for** all $i \in F_r$
7: $ind_{r,i}$ =random identification of a datacenter that implements i and has enough resources to serve an additional request
8: update resources of $d(ind_{r,i})$
9: update $y^i_{d(ind_{r,i})}$
10: set $x^i_{r,d(ind_{r,i})} = 1$
11: $SP_{ind_r} = risk\_aware\_constrained\_Dijkstra(src_r, dst_r)$
12: $compute(FX_{ind})$
{END OF INITIALIZATION STAGE}
{BEGINNING OF SELECTION STAGE}
13: $ind_{overall}$ =best $ind \in P$
14: ind_{best} =best $ind \in P$
15: **for** $loop \leq Npop$
16: $C = P$
17: $c_0 = ind_{best}$
18: **for** all $c \in C$
19: T_c=randomly pick τ individuals $ind \in P$
20: c_f =best T_c
21: c_m =second best T_c
22: $c = crossover(c_f, c_m)$
23: $c = mutation(c)$
24: **for** all $r \in R$
25: $SP_{c_r} = risk\_aware\_constrained\_Dijkstra(src_r, dst_r)$
26: $compute(FX_c)$
27: ind_{best} =best $c \in C$
28: $P = C$
{Select best individual across all iterations and populations}
29: **if** $ind_{best} < ind_{overall}$
30: $ind_{overall}$=ind_{best}
31: $loop = loop + 1$
{END OF SELECTION STAGE}
32: **return** NF mappings, NF instances in each datacenter, links within a path

Figure 5.2: RA-GEN metaheuristic algorithm

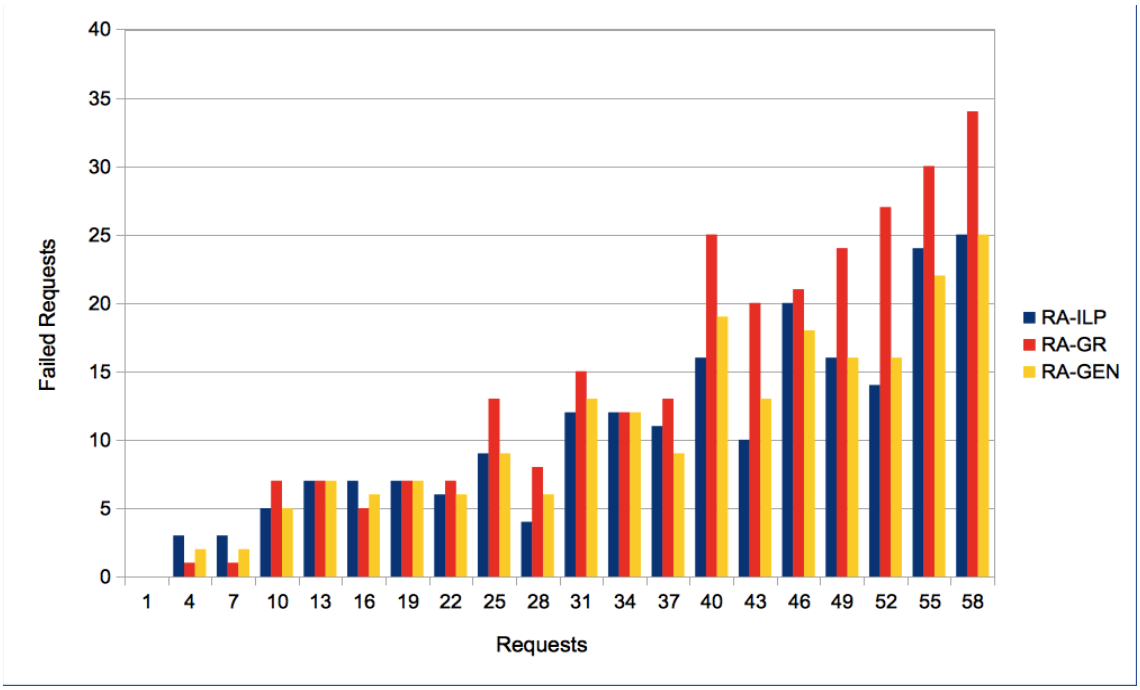


(a)

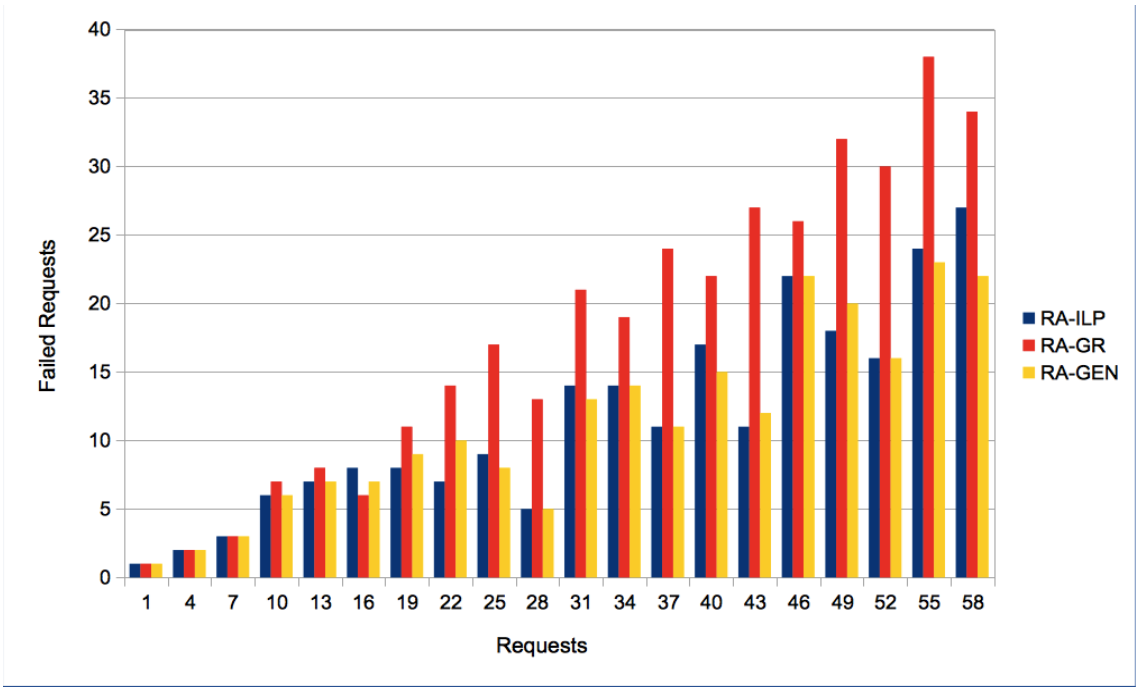


(b)

Figure 5.3: a) deployment costs, and b) routing costs



(a)



(b)

Figure 5.4: Requests assignment failures for the a) idealistic stressor scenario, and b) realistic stressor scenario

Chapter 6 Conclusions and Future Work

This dissertation research focuses on the study of *network function virtualization* (NFV) provisioning and presents a detailed study of survivable multi-objective *virtual network function* (VNF) placement/mapping and routing schemes. First, Chapter 2 presents an overview of the NFV paradigm along with a survey of related work on VNF placement and routing, i.e., including studies on both single and multi-failure scenarios. Subsequently Chapter 3 presents an *integer linear programming* (ILP) optimization model for the efficient multi-objective provisioning in NFV infrastructures. This solution implements both virtual function placement and routing, and a further greedy heuristic method is also proposed for comparison purposes. Both of these solutions are used to verify the importance of *traffic engineering* (TE) load-balancing in (physical) resource-limited scenarios. Next, Chapter 4 extends these optimization and heuristic schemes to further implement "risk-aware" NFV provisioning in multi-failure scenarios. However, VNF placement and routing is a NP-hard problem, and associated optimization schemes can become intractable for a large number of variables. Therefore, Chapter 5 also introduces a more scalable metaheuristic *genetic algorithm* (GA) approach.

6.1 Summary of Research Findings

This dissertation starts out by looking at the problem of VNF placement and routing in NFV infrastructures. To date most related studies have assumed unlimited amounts of physical resources at the infrastructure layer, and only focused on single provisioning objectives. However, clearly these assumptions are not realistic for all operational scenarios. Also to the best of the author's knowledge, there is no known work on multi-objective VNF provisioning.

In light of the above, Chapter 3 introduces a novel ILP optimization solution to jointly maximize the number of satisfied requests, minimize deployment cost, minimize routing cost and implement load-balancing, i.e., termed as the *minimized link load ILP* (MLL-ILP) scheme. In particular, a weighting factor is used in the objective function to allow service providers to tune their service to meet difficult provisioning objective. Additionally, a more scalable polynomial-time greedy heuristic scheme is also proposed, i.e., termed as the *minimized link load greedy heuristic* (MLL-GR) scheme. Although both solutions use the same objective function, the heuristic method is sub-optimal and does not perform as well. Finally, these two methods are compared against each other and also with standardized ILP optimization and greedy heuristic schemes that do not implement any load-balancing support, i.e., termed as *joint routing and placement ILP* (JRP-ILP) and *joint routing and placement greedy heuristic* (JRP-GR). The overall results indicate that:

- The load-balancing factor plays a crucial role in maximizing the number of accepted requests in resource-constrained settings. In particular, results show that the multi-objective MLL-ILP and MLL-GR schemes can provision all input batch request sizes for the evaluated scenarios, whereas the JRP-ILP and JRP-GR schemes experience blocking for some instances.
- All schemes give very similar deployment costs. However the MLL-ILP optimization method achieves notably lower routing costs as compared to the JRP-ILP solution which does not implement load-balancing, i.e., by approximately 20%.
- The proposed MLL-ILP optimization solution also gives much lower routing costs as compared to its greedy heuristic counterpart (MLL-GR) scheme, i.e., by up to 50%. This reduction is due to improved, globalized VNF placement and path selection.

Furthermore, survivability/reliability is also fast becoming a major concern for NFV-based services. Now as noted in the survey in Chapter 2, some studies have addressed related concerns, albeit only for single node or link failures. Indeed, none of these efforts have considered more advanced "risk-aware" techniques to proactively handle large-scale disaster events, i.e., a-priori protection. In light of the above, Chapter 4 proposes two further survivable "risk-aware" multi-objective provisioning schemes for NFV infrastructures. Specifically, the objective function from Chapter 3 is updated to incorporate probabilistic stressor events, and the optimization-based MLL-ILP and heuristic-based MLL-GR solutions are expanded to build two "risk-aware" schemes, i.e., termed as *"risk-aware" ILP* (RA-ILP) and *"risk-*

aware" greedy heuristic (RA-GR). The performance of these solutions is then evaluated, both in terms of pre-fault provisioning efficiency and post-fault restoration/recovery. In particular, the latter tests evaluate two stressor scenarios, idealistic and realistic. The former triggers a large failure region which exactly matches a predefined (a-priori) stressor, whereas the latter generates a unique stressor footprint which differs from the pre-specified stressor events. The overall results show:

- "Risk-aware" provisioning can generate longer connection paths since links with higher failure probability increase routing cost. As a result, the RA-ILP and RA-GR schemes yield higher routing costs as compared to their non-survivable counterparts, i.e., MLL-ILP and MLL-GR.
- For the case of exactly matching failure events (idealistic stressor scenario), the "risk-aware" schemes give a fewer of post-fault demand failure as compared to their non-survivable counterparts, i.e., up to 15% and 25% for the optimization and heuristic schemes, respectively.
- For the case of non-matching failed links (realistic stressor scenario), the "risk-aware" schemes still outperform their non-survivable counterparts. For example, the heuristic RA-GR scheme gives up to 25% less post-failure demand failures as compared to the MLL-GR solution. However, the relative improvement with the optimization-based RA-ILP scheme is lower owing to discrepances. between the predicted (modeled) and actual failures.

Overall, ILP-based optimization solutions can deliver lower costs versus sub-optimal greedy heuristic methods. However, the computational times for ILP solvers can become intractable as the network complexity increases. As a result, there is a further need to develop more scalable solutions that can also deliver good performance. Along these lines, a novel meta-heuristic genetic algorithm scheme for multi-objective NFV provisioning and survivability is also proposed in Chapter 4, i.e., termed as *"risk-aware" genetic algorithm* (RA-GEN) scheme. This solution is further evaluated against its counterpart optimization and heuristic schemes in Chapter 4 (RA-ILP, RA-GR schemes). Overall findings indicate:

- Deployment costs are very similar across all three schemes. However, the RA-ILP solution gives substantially lower routing costs than the heuristic and metaheuristic models.
- For the case of exactly-matching failure events (idealistic stressor scenarios), the RA-GEN scheme gives slightly lower post-fault demand failures then the RA-ILP scheme in a few batch request sizes. Moreover, this solution can also scale to handle larger networks.
- For the case of non-matching failure events (realistic stressor scenario), the RA-GEN scheme yields substantially fewer post-fault demand failures, i.e., increased reliability versus the RA-ILP scheme, i.e., up to 20% less failures.

In conclusion, this dissertation proposes and analyzes a range multi-objective provisioning schemes for NFV infrastructures. This is an open area with little/no existing work to date.

Most notably, three survivable "risk-aware" schemes are proposed and compared, based upon optimization, heuristic and metaheuristic methodologies, i.e., RA-ILP, RA-GR and RA-GEN schemes. Results show that the RA-ILP optimization scheme gives lower costs (better performance), i.e., particularly routing costs. However, the RA-GEN metaheuristic has much lower computational-complexity and also provides a higher level of survivability, especially for realistic stressor scenarios.

6.1.1 Future Work

To the best of the author's knowledge, this work presents the first solution for multi-failure NFV provisioning. As such, it provides a very solid basis to conduct further exploratory research. In particular, new efforts can look at developing post-fault restoration schemes to further improve VNF recovery performance. Further research can also extend the survivability schemes to consider *service function chaining* (SFCs requests). Namely, virtual services will usually be associated with a sequence of VNFs, which must be maintained after failure events. Finally, there is a pressing need to address performance analysis during services migration/redeployment post-failures, i.e., disaster recovery and redeployment.

References

[AM01] A. Medhat, *et al,* "Resilient Orchestration of Service Functions Chains in a NFV Environment". *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN) 2016*. IEEE. 2016.

[BA01] B. Addis, *et al,* "Virtual Network Functions Placement and Routing Optimization". *IEEE International Conference on Cloud Networking*. IEEE. 2015.

[BH01] B. Han, *et al,* "Network Function Virtualization: Challenges and Opportunities for Innovations". *IEEE Communications Magazine* 53.2 (2015), pp. 90–97.

[DO01] O. Diaz, *et al,* "Network Survivability for Multiple Probabilistic Failures". *IEEE/ACM Transactions on Networking* 16.8 (2012), pp. 1320–1323.

[DO02] D. Oliveira, "On Sensitive and Weighted Routing and Placement Schemes for Network Function Virtualization". *Infocommunications Journal* IX.4 (2017).

[DO03] D. Oliveira, *et al,* "Heuristic Methodology for Horizontal Multilayer Soil Stratification". *IEEE International Conference on Environment and Electrical Engineering*. IEEE. 2016.

[DO04] D. Oliveira, *et al,* "Multilayer Soil Parameters Estimation Optimization Using Genetic Algorithms". *International Conference on Grounding and Earthing & International Conference on Lightning Physics and Effects*. 2014.

[ETSI01] Network Function Virtualization Specifications. http://www.etsi.org/technologies-clusters/technologies/nfv.

[FG01] F. Gu, *et al,* "Survivable Cloud Network Mapping for Disaster Recovery Support". *IEEE Transactions on Computers* 64.2 (2014), pp. 2353–2366.

[GS02] W. Ding, *et al,* "Efficient Algorithms for Survivable Virtual Network Embedding". *Asia Communications and Photonics Conference and Exhibition (ACP)*. 2010.

[HL01] H. Lee, *et al,* "Diverse Routing in Networks with Probabilistic Failures". *IEEE/ACM Transactions on Networking* 18.6 (2010), pp. 1895–1907.

[HM01] H. Moens, *et al,* "VNF-P: A Model for Efficient Placement of Virtualized Network Functions". *IEEE International Conference on Network and Service Management (CNSM)*. IEEE. 2014.

[HY04] H. Yu, "Survivable Virtual Infrastructure Mapping in a Federated Computing and Networking System under Single Regional Failures". *IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2010.

[JC01] J. Crichigno, "Joint Routing and Placement of Virtual Network Funcations". *IEEE Telecommunications and Signal Processing (TSP) 2017*. IEEE. 2017.

[JC02] Jorge Crichigno, *et al,* "Dynamic routing optimization in WDM networks". *IEEE Globecom Conference*. IEEE. 2010.

[JC03] Jorge Crichigno, *et al,* "Throughput optimization and traffic engineering in WDM networks considering multiple metrics". *IEEE ICC Conference*. IEEE. 2010.

[JF01] J. Fan, *et al,* "Availability-aware Mapping of Service Function Chains". *IEEE Conference on Computer Communications (INFOCOM) 2017*. IEEE. 2017.

[JM01] J. McCall, "Genetic algorithms for modelling and optimisation". *Journal of Computational and Applied Mathematics* 184.1 (2005), pp. 205–222.

[MB01] M. Beck, "Resilient Allocation of Service Function Chains". *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN) 2016*. IEEE. 2016.

[MB02] M. Bouet, *et al,* "Cost-Based Placement of Virtualized Deep Packet Inspection Functions in SDN". *Military Communications Conference (MILCOMM)*. 2013.

[MB03] M. Beck, *et al,* "Coordinated Allocation of Service Function Chains". *IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2015.

[MC01] M. Casazza, *et al,* "Securing Virtual Network Function Placement with High Availability Guarantees". *IFIP Networking Conference (IFIP Networking) and Workshops 2017*. 2017.

[MX01] M. Xia, *et al,* "Network Function Placement for NFV Chaining in Packet/Optical Datacenters". *IEEE/OSA Journal of Lightwave Technology* 33.8 (2015).

[RC01] R. Cohen, *et al,* "Near Optimal Placement of Virtual Network Functions". *IEEE International Conference on Computer Communications*. IEEE. 2015.

[RM01] R. Mijumbi, *et al,* "Network Function Virtualization: State-of-the-art and Research Challenges". *IEEE Communications Surveys & Tutorials* 18.1 (2016), pp. 236–262.

[SG01] S. Gebert, *et al,* "Demonstrating the Optimal Placement of Virtualized Cellular Network Functions in Case of Large Crowd Events". *ACM SIGCOMM Conference*. 2014.

[SG02] S. Gladstone, *The Effects of Nuclear Weapons*. 1st ed. Vol. 4. 10. An optional note. The address: Knowledge Publications, Nov. 2006. ISBN: 160322016X.

[SM01] S. Mehraghdam, *et al,* "Specifying and placing chains of virtual network functions". *IEEE International Conference in Cloud Networking (CloudNet)*. IEEE. 2014.

[TC01] T. Cormen, *et al, Introduction to Algorithms*. 2nd ed. McGraw Hill, 2001.

[VM01] V. Maniezzo, *et al,* "Hybridizing Metaheuristics and Mathematical Programming". *Annals of Information Systems* 10 (2009).

[WD01] W. Ding, *et al,* "Enhancing the Reliability of Services in NFV with the Cost-Efficient Redundancy Scheme". *IEEE International Conference on Communications (ICC) 2017*. IEEE. 2017.

[YL01] Y. Li, "Software-Defined Network Function Virtualization: A Survey". *IEEE Access* 3 (2015), pp. 2542–2553.

[ZY01] Z. Ye, *et al,* "Joint Topology Design and Mapping of Service Function Chains for Efficient, Scalable, and Reliable Network Functions Virtualization". *IEEE Network* 30.3 (2016).

Appendix A: Glossary

BIP	Binary Integer Linear Program
COTS	Commercial-Off-The-Shelf
DPI	Deep Packet Inspection
ETSI	European Telecommunications Standards Institute
GA	Genetic Algorithm
IDS	Intrusion Detection System
ILP	Integer Linear Program
INLP	Integer Non-Linear Program
IPS	Intrusion Prevention System
ISG	Industry Specification Group
JRP-GR	Joint Routing and Placement Greedy Heuristic
JRP-ILP	Joint Routing and Placement ILP
MILP	Mixed Integer Linear Program
MIQCP	Mixed Integer Quadratically Constrained Program
MLL-GR	Minimized Link Load Greedy Heuristic
MLL-ILP	Minimized Link Load ILP
NAT	Network Address Translation
NFV	Network Function Virtualization
RA-GEN	Risk-aware Genetic Algorithm

RA-GR	Risk-Aware Greedy Heuristic
RA-ILP	Risk-aware ILP
SDN	Software-Defined Networking
SDNFV	Software-Defined Network Function Virtualization
SFC	Service Function Chaining
SP	Shortest Path
SRLG	Share Risk Link Group
SRRG	Share Risk Resource Group
VNE	Virtual Network Embedding
VNF	Virtual Network Function
VNF-FG	Virtual Network Function Forwarding Graph
WG	Working Group