An Efficient Multi-Objective Survivability Scheme for Mapping and Routing of Virtual Functions in Failure Scenarios

Conference Paper · March 2019

CITATION
1

READS
98

5 authors, including:

Diogo Oliveira
Florida State University
30 PUBLICATIONS **53** CITATIONS
SEE PROFILE

Jorge Crichigno
University of South Carolina
80 PUBLICATIONS **484** CITATIONS
SEE PROFILE

Elias Bou-Harb
University of Texas at San Antonio
85 PUBLICATIONS **785** CITATIONS
SEE PROFILE

Mohamed Rahouti
University of South Florida
14 PUBLICATIONS **31** CITATIONS
SEE PROFILE

An Efficient Multi-Objective Survivability Scheme for Mapping and Routing of Virtual Functions in Failure Scenarios

D. Oliveira[1], J. Crichigno[2], E. Bou-Harb[3], M. Rahouti[4], N. Ghani[4]
[1]*Florida State University,* [2]*University of South Carolina,*
[3]*Florida Atlantic University,* [4]*University of South Florida*

***Abstract*—The network function virtualization (NFV) paradigm focuses on increasing manageability and scalability of modern complex heterogeneous networks and network services by decoupling the network functions and hosting devices. However, as new promising solutions become available, the need for availability and reliability techniques grow, particularly for large-scale and interdependent scenarios. Therefore this study proposes a meta-heuristic genetic algorithm scheme to deploy "risk-aware" virtual function mapping and traffic routing to improve the reliability of user services as well as reduce deployment and routing costs. Furthermore, this solution is compared with two other "risk-aware" survivable schemes in order to evaluate its accuracy, i.e., integer linear programming (ILP) and greedy heuristic solutions .**

***Index Terms*—network function virtualization (NFV), survivability, genetic algorithm (GA).**

1. Introduction

The past three decades have seen rapid technological advances 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 gains have led to the deployment of large interconnected datacenters, and the broader emergence of cloud-computing paradigms. However, as these trends have unfolded, a host of management and orchestration challenges have also arisen.

Now most physical networking-layer setups have yielded increased management complexity due to specialized configuration requirements and a high degree of vendor dependency. In response, various organizations began to partner and develop new strategies to simplify network nodes by decoupling the data and control planes outsourcing the control plane to a concentrator, i.e., *software-defined networking* (SDN). Overall, this technology can lower network ossification and reduce operational and capital expenditures (OpEx and CapEx) for carriers.

However, even though data and control plane decoupling presents many advantages, traditional network services deployment still remains a complex and expensive undertaking. Namely, there is a high degree of vendor-dependency as traditional network services are managed and deployed by embedded vendor-proprietary software. Therefore, in order to reduce network services management complexity and improve (re)deployment capabilities, the *network function virtualization* (NFV) paradigm has been further evolved to support deployment of network functions on *commercial-of-the-shelf* (COTS) equipment, i.e., running as software instances. In general, most client networking services include a range of tasks such as firewalls, *deep packet inspection* (DPI) engines, intrusion detection/prevention systems, *network address translation* (NAT) boxes, etc. Now in terms of NFV, these networking services are typically composed of multiple individual *virtual network functions* (VNF). Specifically a VNF can be deployed/implemented across multiple datacenter sites, and a datacenter site can host multiple VNFs.

Overall, as service providers show increasing interest in NFV paradigms, 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 typically considers a multiple set of client requests, where each request is comprised of a set of VNFs, a source node, a destination node and a minimum bandwidth interconnection rate. Hence service providers must efficiently place these VNFs across their datacenters to reduce deployment and routing costs, and also increase service performance and reliability.

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 and/or increase revenues [2]- [6].

Furthermore, survivability concerns are also becoming increasingly important. Here, plural techniques can be implemented in order to improve the resiliency and availability of the virtual functions and network services against failure events. Although some efforts have addressed NFV survivability topics, these studies have mostly focused on single isolated system failures. As a result, the further impact of large-scale disaster events (multiple failures) on NFV-based services remains a key a concern. These occurrences can include natural disasters, *weapons of mass destruction* (WMD) attacks and cascading power outages. Indeed there is a growing need to build more systematic, multi-objective

VNF placement schemes to efficiently provision resources and directly incorporate the randomized nature of disaster events, i.e., "risk-aware" VNF placement under multi-failure scenarios. Moreover, to the best of the authors' knowledge, [1] is the only known work to perform such analysis and deployment. Although the results presented in such effort are promising, the proposed schemes lack in either performance (greedy heuristic) or scalability (*integer linear programming* - ILP). The linear and greedy schemes presented in [1] may not be the best solutions to overcome the challenges imposed by stochastic failures over large-scale .

This work addresses the above challenges and develops a novel multi-objective resource provisioning solutions for NFV infrastructures based on *genetic algorithm* (GA). Specifically, this technique implements function placement and routing strategies and also incorporates stochastic failure models to lower failure risk and improve VNF reliability in large-scale multi-failure scenarios, which is achieved due to the GA's random nature. In addition, many existing VNF placement and/or routing schemes also assume abundant datacenter resources to satisfy all client demands. As such, these methods only focus on minimizing service cost. However the assumption of unconstrained resources may not hold in heavy demand or post-failure scenarios where resource scarcity will be high. As a result, this effort combines two main goals: a) efficiently place and route VNFs, and deploy traffic engineering, in order to minimize costs and maximize overall requests in a constrained environment, b) reduce/mitigate virtual functions downtime by incorporating a pre-fault "risk-aware" meta-heuristic algorithm to the provisioning/placement solution.

This paper is organized as follows. First, a background review of existing work in survivable VNF placement and routing is presented in Section II. Subsequently, Section III presents the overall notation and stochastic multi-failure disaster model. A detailed meta-heuristic formulation for risk-aware VNF provisioning based on genetic algorithm is then presented in Section IV. Moreover, Section V presents some detailed performance results, which are compared with the schemes presented in [1]. Finally a conclusion is presented along with future directions.

2. Related Work

Along these lines, this section overviews some of the latest developments in the resilient and survivable VNF placement and routing problem. Open research challenges are then outlined to motivate this work.

Now a wide range of studies have looked at VNF placement under regular, i.e., working network conditions [2]- [6]. Overall most of these VNF placement schemes have focused on objectives such as performance improvement, cost reduction, energy efficiency, traffic engineering, etc. However, VNF survivability (reliability) is now becoming a major concern given the critical nature of many services, and these studies consider single failure or multi-failure scenarios. Most of these efforts have only addressed isolated single node and link failures. For example, [7] considers the case of a single VNF failure causing a service chain interruption. An extended orchestration architecture is proposed here to dynamically redefine flows and steer (re-route) traffic to establish new paths and reduce downtime. Nevertheless, VNF placement is not considered here. Further provisions for resource limitations and bandwidth constraints are also lacking. As such, this effort only focuses on reactive disaster recovery.

Meanwhile [8] proposes another resilient *service function chaining* (SFC) allocation scheme. First, a greedy heuristic is designed to map the *virtual network functions forwarding graphs* (VNF-FGs) for service chaining requests, thereby yielding resource allocation constraints and VNF interdependence. However a complex (time-consuming) backtracking scheme is then presented to allocate resources, which makes it intractable for a large-scale infrastructure.

Moreover, the work in [9] presents a *joint topology design and mapping* (JTDM) heuristic, termed as closed-loop with critical mapping feedback. This solution builds the network topology and then maps the VNFs in order to minimize the *total bandwidth cost* (TBC). In particular, TBC minimization is achieved by performing function combination. 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.

Overall, the work in [10] takes a slightly different approach and assumes the availability of a-priori probabilistic resource availability levels. However, broader resources constraints, routing costs and traffic engineering concerns are not included.

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, malicious WMD attacks, etc. Except for [1], the existing body of work on network disaster-recovery has only focused on point-to-point connections and *virtual network* (VN) services. The work presented in [1] proposes two multi-objective VNF placement schemes to address the survivability problem under multi-failure scenarios considering a pre-fault awareness. The first one is an ILP optimization scheme while the second one is a greedy heuristic solution and its methods and results are further analyzed and used for comparison.

3. Notation & Failure Model

To be able to deploy a survivable VNF multi-objective placement and routing scheme, some steps are necessary. First, it is imperative to determine the physical network model, the virtual functions demand models and the multi-failure model and their notation. Only then it is possible to determine and test efficient placement solutions.

3.1. Network Model

A graph G=$(V; E)$ is used to define the network infrastructure, where V is the set of nodes and E the set of links. Additionally, a link $(i, j) \in E$ has an associated cost rc^{ij}, used to determine the routing cost, and a bandwidth capacity b^{ij} , used to quantify the link capacity. Also the set of all possible NFs is denoted by F, and the set of datacenters where NFs are implemented and can be provisioned are represented by the subset $D \subseteq V$. Clearly a given datacenter $d \in D$ implements a subset of functions $F_d \subseteq F$. The customizable number of resource types is also denoted by the integer m. For example, m=3 can refer to processor, storage and memory resources. 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 in order to implement a function $i \in F_d$, datacenter $d \in D$ uses $w^i_{d,1}, w^i_{d,2}, ..., w^i_{d,m}$ resources. Also, the setup cost of locating an instance of a function $i \in F_d$ at datacenter d is sc^i_d, and an instance of function i at datacenter d can serve λ^i_d requests. In order to accommodate more requests, multiple instances of function i can also be deployed at datacenter d.

3.2. Virtual Functions Demand Model

Now for a set of requests R arriving from clients, each request $r \in R$ is characterized by a 4-tuple format denoting the source and destination nodes of the flow, the set of requested functions $F_r \subseteq F$, and the minimum required bandwidth capacity, i.e., src_r, dst_r, F_r, b_r. Ideally, the VNF placement solution defines the best (or most efficient) placement locations according to the fitness function (see Eq. 4). Additionally, the overall cost of provisioning all functions F_r associated with request r along a path P_r is given by:

$$d(P_r) = \sum_{d \in D} \sum_{i \in F_r} sc^i_d y^i_d \tag{1}$$

Similarly, the total routing cost for this path is also given by:

$$c(P_r) = \sum_{r \in R} \sum_{(i,j) \in E} rc^{ij} l^{ij}_r \tag{2}$$

Here, $x^i_{r,d}$ indicates whether function i requested by request r is implemented at datacenter d or not, while y^i represents the number of instances of function i at node d. Moreover, $l^{i,j}_r$ is indicates whether link (i, j) is used to route the traffic flow for request r.

3.3. Multi-Failure Model

A realistic probabilistic model is used to specify large-scale disaster events with multiple highly-correlated spatial and temporal link failures. Namely, the set U defines an a-priori set of outage events, $U = \{u_1, u_2, ..., u_N\}$, where each event u_n has an associated occurrence probability, $p(u_n)$. It is also assumed that all events are sufficiently rare and therefore can be treated as independent and mutually-exclusive, $\sum \forall u_n \in Up(u_n) = 1$. Without loss of generality, it is assumed that all outages are non-overlapping in the geographic domain. Now each event has an associated set of vulnerable links, termed as the *shared risk link group* (SRLG). A non-conditional failure probability $\omega(i, j)$ is also defined for each physical link $(i, j) \in E$ in the region of event u_n (with respect to the occurrence of event u_n). Note that this framework can also be extended for conditional failure probabilities with overlapping risk regions.

4. The GA Provisioning Strategy

Overall, a pre-fault "risk-aware" meta-heuristic genetic algorithm is proposed here to reduce failure downtime and maximize the number of satisfied requests, especially on large-scale network sizes, termed as *"risk-aware" genetic algorithm* (RA-GEN). Before detailing the RA-GEN scheme, it is crucial to briefly overview the genetic algorithm approach. As shown in [11], this scheme mimics the biological evolution. Here, from within a population of individuals, a set of pairs are randomly chosen to generate a set of children, where each pair (father and mother) generates a single child. This child's chromosomes sequence is formed by the crossover of the parents' chromosomes. Out of that population of parents, a new population are then created and new individuals are also formed, resulting in diversified individuals with random chromosomes sequence. Note that the chromosomes represent the set of values to be used by the objective function.

4.1. RA-GEN Notation Overview

Figure 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) identifying 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 child 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 generate 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 child populations evaluated

Moreover, the proposed RA-GEN has a specific algorithm (see sub-section 4.2) and therefore specific variables,

which are introduced and summarized in Figure 1. Foremost, the overall population is defined by the set P and consists of pop_size individuals. Each individual $ind \in P$ has a set of chromosomes, where $ind_{r,i}$ is the (r, i) chromosome identifying the datacenter that instantiates function i requested by request r. The fitness function value for an individual is given by FX_{ind}.

Furthermore, the best individual across all populations is defined as $ind_{overall}$. 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, where $c \in C$ is an individual child. The first child is also denoted by c_0. Foremost, T individuals (parents) are chosen from the main population to generate the children, and this subset is denoted by $T_c \subseteq P$, i.e., $|T_c| = |\tau| \leq |P|$. The father (mother) of an individual is also denoted by c_f (c_m). Finally, 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.

4.2. RA-GEN Scheme

The overall pseudocode for the RA-GEN scheme is presented in Figure 2 and consists of two key stages, i.e., initialization and selection. Here the first stage starts by initializing all GA search variables (line 3) 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 source and destination nodes (lines 4-11). 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). 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 modified "risk-aware" Dijkstra scheme, i.e., risk aware *constrained_Dijkstra()*, where the routing path cost is computed based upon link failure probabilities, $\omega(i, j)$, as follows:

$$c'_{ij} = \sum_{(i,j) \in E} (c^{ij} + \frac{b_r}{b_{ij}}) * (1 + \omega(i, j)) \qquad (3)$$

The fitness function value, FX_{ind} (see Eq. 4.2.1), is also computed for each randomized individual, and the one with the highest value is chosen as the best solution (lines 12-13).

After the initial population generation is complete, the second stage is launched to perform selection. Namely, this phase iterates to generate and test a given number of child populations $Npop$. 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 17). The remaining

Figure 2. RA-GEN pseudocode

1: INPUT (Network infrastructure and demands): G(V, E), $c^{ij}, \omega(i,j) \forall (i,j) \in E, R, F, D$
2: OUTPUT: $x^i_{r,d}, y^i_d, l^{ij}$ 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

individuals are then created by inheritance, i.e., crossover and mutation (lines 19-23). 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 τ individuals is selected for each child by randomly selecting τ individuals from population P (line 19). 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). 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). Finally, the child's chromosomes are further adjusted 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 with 16 datacenters and a chromosome assigning datacenter 5 to instantiate a specific function. If this particular chromosome is selected for mutation, then its value is modified, as $16 - 5 = 11$. In order to ensure that the new datacenter has enough resources after inheritance and mutation are complete, the algorithm also checks 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). The overall fitness value FX_c is then evaluated for each child

individual c (line 26), and akin to the parent population stage, the best individual, ind_{best}, from the child population C is selected (line 27). Finally, the best overall individual is 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) generates a total of $Npop$ children populations which are created by inheriting chromosomes from their respective parents. Note that each child population may not necessarily 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.

4.2.1. Objective Function. The accuracy of the a solution is measured by its capability of increasing the number of satisfied functions and reducing the deployment cost, routing cost and link usage, and is defined by the fitness function in Eq. 4:

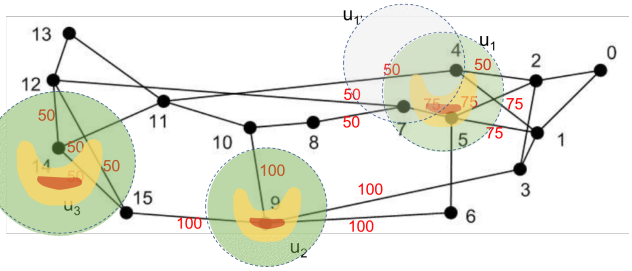
$$FX_{ind} = w_1 \sum_{r\in R}\sum_{i\in F_r}\sum_{d\in D|i\in F_d} x_{r,d}^i - w_2 \sum_{d\in D}\sum_{i\in F_r} c_d^i y_d^i$$
$$\text{- } w_3 \sum_{r\in R}\sum_{(i,j)\in E} c^{ij} l_r^{ij} * (1+\omega(i,j)) - \alpha w_4 \quad (4)$$

Here, 4 terms are multiplied by respective weight factors, $w_1, w_2, w_3 and w_4$. Namely, the first term above represents the total number of satisfied NFs, and the main goal is to maximize this value. Meanwhile the total cost of deploying NFs instances and accepting requests at the various datacenters is given by the second term. Similarly, the third term is the total routing cost. However note that the link failure probabilities $\omega(i,j)$ are also incorporated here. Namely, the routing cost can be increased proportional to the link failure probability. Meanwhile the fourth term represents the maximum overall link load, where α is the highest link usage ratio, i.e., sum of all b_r using a link $(i,j) \in E$ divided by link capacity $b_{i,j}$. Note that the second, third and fourth terms are all negative since maximizing a negative term is equivalent to minimizing it [6].

5. Performance Evaluation

To evaluate the performance and accuracy of the proposed survivable RA-GEN meta-heuristic scheme, its results

Figure 3. Network topology and stressor regions



are compared with the ILP and heuristic schemes introduced by [1], i.e., *"risk-aware" ILP* (RA-ILP) and *"risk-aware" greedy heuristic* (RA-GR). Therefore, the same topology, risk regions, faillure regions and parameters setups are adopted here, as illustrated in Fig. 3. Specifically, this topology is comprised of 16 nodes and 25 links, and three potential stressor (risk) regions are superimposed here, i.e., u_1 is 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 is comprised of 4 links ($l_{9,3}$, $l9,6$, $l_{9,10}$, $l_{9,15}$) and u_3 is comprised of 4 links ($l_{14,11}$, $l_{14,12}$, $l_{14,15}$, $l_{12,15}$). Region $u_{1'}$ is not a risk-region, but an actual failure region, and it is comprised of 6 links ($l_{1,4}$, $l_{2,4}$, $l_{4,11}$, $l_{5,7}$, $l_{7,8}$ and $l_{7,12}$). The associated link failure probabilities are further modeled based upon multi-failure stressor attacks with 3 sub-areas, as shown in green, yellow and red (representing 50, 75 and 100% of outage probabilities, respectively). Note that Figure 3 also shows the corresponding failure probabilities next to each link, i.e., $\omega(i,j)$, for each SRLG region.

Before failure conditions are introduced, the RA-GEN scheme is executed to satisfy request batch sizes varying from 1-60 requests, with each request demanding 4 NFs, i.e., the total number of NFs ranges from 4 to 240. Note that satisfying a batch of requests means being able to place VNFs (of each and all requests) and establish connecting paths among the corresponding datacenters taking the three pre-defined risk regions (u_1, u_2 and u_3) into consideration. Once the VNF placement and routing are done, the testcases are defined and tested. Namely, a multi-failure scenario is randomly selected and triggered by choosing one of the stressor (risk) regions in Fig. 3. In particular, the large u_1 region is chosen here as it includes the largest number of links. Now clearly it is very difficult to predict the location of *a-priori* failure events in advance, especially large-scale disasters. Hence in practice the pre-defined/a-priori risk regions will rarely match the actual failure footprints seen in the field. As a result, two different disaster testcases are evaluated here, i.e., idealistic and realistic. The former assumes 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 that differs from the a-priori risk region u_1 and is shifted slightly to the west, i.e., failure region $u_{1'}$, as shown in Fig. 3.

TABLE 1. 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	$u_{1'}$
Outage Links	$(1,5)$, $(2,4)$, $(2,5)$, $(5,7)$	$(4,11)$, $(5,7)$, $(7,12)$

TABLE 2. 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%

Once the testcases are chosen, the survivability testcases parameters are defined and presented in Table 1, along with the parameter settings for the RA-GEN scheme, given in Table 2. In particular, the GA selection is done over 100 populations, with each having 20 individuals and crossover/mutation rates of 20%.

5.1. Idealistic Scenario

To further gauge the impact of multiple link failures, the post-fault performance of the "risk-aware" genetic algorithm is evaluated and compared with the RA-ILP and RA-GR from [1]. Hence, two different testcases are evaluated here i.e., idealistic and realistic stressor scenarios. Note that according to the link failure probabilities determined by the multi-failure model (see Section 3.3), not all links within a risk region are to fail. Therefore, as per Table 1, the idealistic testcase selects the u_1 region and fails 4 links ($l_{1,5}$, $l_{2,4}$, $l_{2,5}$, $l_{5,7}$). Accordingly, Fig. 4 plots the number of failed requests for the ideal stressor and indicates the lowest survivability with the greedy RA-GR scheme (red), i.e., generally within 20-50% more failures than the other two. By contrast, the RA-GEN method (yellow) is very competitive with the RA-ILP optimization scheme (blue). In fact the meta-heuristic solution even gives lower failures for some batch sizes, i.e., 4, 7, 16, 37, 46 and 55 requests.

5.2. Realistic Scenario

Subsequently, the realistic stressor testcase selects the modified $u_{1'}$ region (Fig. 3) and fails 3 links ($l_{4,11}$, $l_{5,7}$ and $l_{7,12}$). The related post-fault failure results are plotted

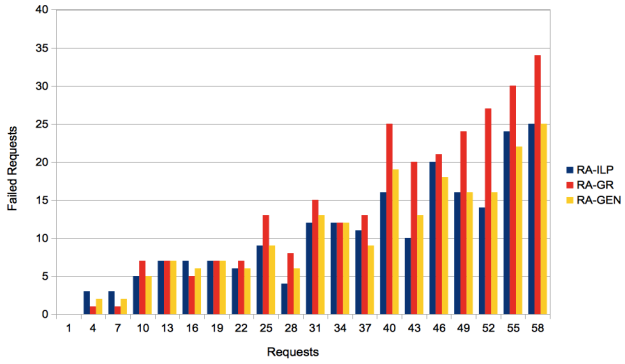


Figure 4. Post-fault performance: idealistic stressor scenario

in Fig. 5 and indicate that the RA-GEN meta-heuristic actually gives improved survivability versus both 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 for large-scale infrastructures. Furthermost, the meta-heuristic GA scheme presents greater reliability improvement compared with the ILP optimization for multi-failure realistic and probabilistic scenarios, most likely due to the algorithm's random nature.

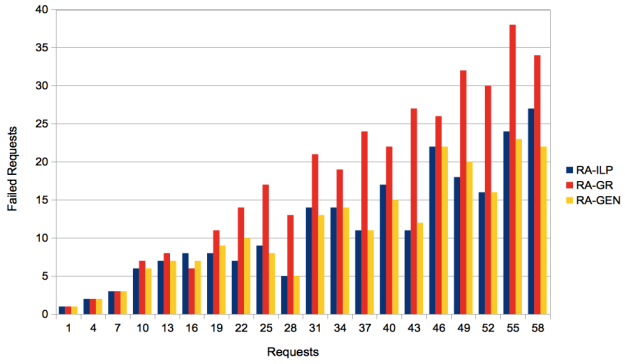


Figure 5. Post-fault performance: realistic stressor scenario

6. Conclusion

This paper presents inspects disaster recover support techniques with the context of NFV-based infrastructures. Overall, a probabilistic multi-failure model is used for a-priori characterization of large-scale disaster events. A novel meta-heuristic algorithm is presented for network function (NF) placement and routing, with a focus in minimizing failures and maximizing the number of satisfied service demands. Detailed reliability analysis results show that the proposed solution can significantly reduce service disruption, especially when applied for realistic scenarios with random characteristics. Furthermost, the proposed scheme has shown improvement when compared with greedy heuristic and linear optimization strategies.

References

[1] D. Oliveira, J. Crichigno, N. Siasi, E. Bou-Harb and N. Ghani, "Joint Mapping and Routing of Virtual Network Functions for Improved Disaster Recovery Support", *IEEE Southeastcon 2018*, Saint Petersburg, FL, April 2018.

[2] H. Moens, et al, "VNF-P: A Model for Efficient Placement of Virtualized Network Functions", *IEEE International Conference on Network and Service Management (CNSM)*, Rio de Janeiro, Brazil, Jan. 2015.

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

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

[5] J. Crichigno, D. Oliveira, M. Pourvali, N. Ghani, D. Torres, "A Routing and Placement Scheme for Network Function Virtualization", *IEEE Telecommunications and Signal Processing (TSP) 2017*, Barcelona, Spain, 2017.

[6] D. Oliveira, J. Crichigno, N. Ghani, "On Sensitive and Weighted Routing and Placement Schemes for Network Function Virtualization", *Infocommunications Journal*, Vol. IX, Issue 4, Dec. 2017.

[7] A. Medhat, et al, "Resilient Orchestration of Service Functions Chains in a NFV Environment", *IEEE Conference on Network Function Virtualization and Soft- ware Defined Networks (NFV-SDN) 2016*, Palo Alto, CA, Nov. 2016.

[8] M. Beck, J. Botero, K. Samelin, "Resilient Allocation of Service Function Chains", *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Paulo Alto, Nov. 2016.

[9] Z. Ye, et al, "Joint Topology Design and Mapping of Service Function Chains for Efficient, Scalable, and Reliable Network Functions Virtualization", *IEEE Network*, Vol. 30, Issue 3, May 2016.

[10] M. Casazza, et al, "Securing Virtual Network Function Placement with High Availability Guarantees", *IFIP Networking Conference and Workshops 2017*, Stockholm, Sweden, June 2017.

[11] D. Oliveira, et al, "Heuristic Methodology for Horizontal Multilayer Soil Stratification", *IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC)*, Florence, Italy, June 2016.