Customizable Mapping of Virtualized Network Services in Multi-datacenter Environments Based on Genetic Metaheuristics

Vinicius Fulber-Garcia (vfulbergarcia@gmail.com)
Federal University of Paraná
Marcelo Luizelli
Universidade Federal do Pampa
Carlos Paula dos Santos
Universidade Federal de Santa Maria
Eduardo Spinosa
Federal University of Paraná
Elias Duarte Jr.
Federal University of Paraná

Research Article

Keywords: Network Functions Virtualization, Service Function Chain, Deployment, Embedding, Mapping, Genetic Algorithm

Posted Date: March 15th, 2023

DOI: https://doi.org/10.21203/rs.3.rs-2671034/v1

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Journal of Network and Systems Management on August 1st, 2023. See the published version at https://doi.org/10.1007/s10922-023-09760-1.

Customizable Mapping of Virtualized Network Services in Multi-datacenter Environments Based on Genetic Metaheuristics

Vinicius Fulber-Garcia* · Marcelo C. Luizelli · Carlos R. Paula dos Santos · Eduardo J. Spinosa · Elias P. Duarte Jr.

Received: ????/??/?? / Accepted: ????/??/??

Abstract One of the major challenges of the Network Functions Virtualization (NFV) paradigm is to properly deploy functions and services across the network. In particular, current solutions for multi-domain service mapping present several restrictions in terms of the choice of optimization models and metrics. This lack of flexibility ultimately leads to sub-optimized mappings that do not meet the (often conflicting) requirements of all the parties involved in the deployment process (*e.g.*, network operators, clients, providers). This work proposes GeSeMa (Genetic Service Mapping), a new intelligent mapping solution based on genetic algorithms. GeSeMa enables flexible configuration of the evaluation setup, which is used to generate candidate mappings. The solution allows the specification of arbitrary optimization metrics, constraints, and different evaluation policies. A genetic algorithm processes mapping requests and iteratively creates/evolves candidate mappings. We evaluate GeSeMa through comprehensive case studies, including a comparison with other classic and state-of-the-art alternatives.

Keywords Network Functions Virtualization · Service Function Chain · Deployment · Embedding · Mapping · Genetic Algorithm

Vinicius Fulber-Garcia, Eduardo J. Spinosa, Elias P. Duarte Junior
Federal University of Paraná
E-mail: {vfgarcia, spinosa, elias}@inf.ufpr.br

Marcelo C. Luizelli
Federal University of Pampa
E-mail: marceloluizelli@unipampa.edu.br

Carlos R. P. dos Santos
Federal University of Santa Maria
E-mail: csantos@inf.ufsm.br

1 Introduction

Network Functions Virtualization (NFV) is driving a paradigm shift in telecommunications. NFV allows network functions that have been traditionally implemented as physical appliances in hardware to be implemented as software that runs on virtual machines [1, 2]. Examples include intrusion detection systems, load balancers, traffic filters, deep packet inspection tools, among many others [3–6]. Hardware appliances, despite presenting high-performance, incur on high capital and operational expenditures, low scalability, low mobility, and complex management. On the other hand, Virtualized Network Functions (VNF) are executed on a software plane that runs on commercial off-the-shelf hardware, thus presenting much lower costs in terms of development, deployment, and maintenance.

Individual VNFs are the building blocks of Service Function Chains (SFC) [7], which are compositions of multiple network functions connected on a service topology. The deployment of service topologies on virtualized environments comprises the execution of a number of tasks [1]: composition, embedding, and scheduling. Recent works have proposed solutions to tackle prominent challenges and improve the results of each of those deployment tasks [8–12].

Informally, the problem of mapping a network virtualization service across multiple domains consists of defining where the network functions that make up the service will be instantiated and executed. Different domains may have restrictions on the number of services they run and the resource requirements of the respective functions. In addition, the policies the domain adopts together with business rules adopted by each domain also have an impact on which alternatives are feasible and their costs. There are network functions that are native to certain domains, to which they must necessarily be mapped. For other functions, there is a choice of where they should be executed, which depends not only on equalizing the resources required with policies and resources available in the domains. It also depends on the topology of the virtualized service and the topology of the multi-domain network to which it will be mapped. In this case, the objective is typically to reduce the amount of traffic transferred between domains as flows are forwarded through the network service. Furthermore, other criteria that can be defined for each mapping process in particular, such as maximizing the number of users, and maximizing or minimizing the number of domains used to host the service. It should also be taken into account that the mapping objectives usually change according to the very nature of the service being mapped, the type of environment in which they operate, and also the network technologies involved, such as 5G or earlier cellular networks or even IoT or vehicular networks.

Traditional solutions for mapping VNFs are based on evaluation setups that are often static in terms of the set of optimization metrics they employ, as well as objectives and weights, lacking the flexibility required to customize their execution [1, 13, 14]. Thus, the requirements of the multiple stakeholders (*i.e.*, clients, providers, and network operators) are hardly met. A static

strategy often leads to poor results, leaving stakeholders to adapt their needs to whatever setup is available in the solutions they are using. These limitations are particularly critical when service topologies are mapped on multiple domain environments [15,10,16,17]. These solutions only allow stakeholders to make simple adjustments (*e.g.*, adjusting weights of existing optimization metrics), but to the best of our knowledge, no current solution allows arbitrary optimization metrics and objectives to be defined.

It is important to note that, besides the evaluation setup, other features can impact the mapping of virtualized network services on multiple domains. Examples of some of those features include: the characteristics of the network infrastructure (such as whether it is private or public, or whether encrypted connections and specific communication protocols are employed); the topology (*e.g.*, fat tree, three-tier, fully-connected, arbitrary); the type of service (*e.g.*, traffic filtering, cache, security); and the structure of service topologies (*e.g.*, linear or branched). Frequently, mapping solutions also neglect (completely or partially) these features, which affects their flexibility and further limits their applicability.

In this work, we propose a new multi-domain mapping solution, called Genetic Service Mapping (GeSeMa). GeSeMa allows the evaluation setup to be customized, providing high flexibility to adapt to different needs of the multiple stakeholders and takes multiple features into consideration. To do that, the stakeholders describe their needs and other service features on a standard request document. GeSeMa then uses a multi-objective optimization metaheuristic based on genetic algorithms to find mapping candidates in feasible time. We evaluate GeSeMa through comprehensive case studies, including a comparison with other alternative classic strategies and state-of-the-art solutions [16].

The rest of this work is organized as follows. Section 2 presents preliminary definitions and related work. GeSeMa is presented in Section 3. Evaluation results are in Section 4, with comparisons with other classic and state-of-the-art solutions. Finally, Section 5 concludes the paper and presents future work.

2 Background and Related Work

This section is organized in three parts. Subsection 2.1 presents basic concepts related to NFV technology and service deployment. Subsection 2.2 gives a brief overview of genetic algorithms. Subsection 2.3 describes related work on multi-domain mapping of virtualized network services.

2.1 NFV & Service Deployment

Traditional network infrastructures rely on dedicated hardware – called physical appliances – to execute a myriad of functions, such as network traffic routing, shaping, and balancing [18]. The Network Functions Virtualization

(NFV) [19] paradigm allows the implementation of network functions using virtualization technologies. In comparison with hardware alternatives, NFV technology reduces costs and increases flexibility [20]. A Virtualized Network Function (VNF) is the basic NFV unit which processes network traffic applying some specific functionality. Furthermore, complex network services can be created through the connection of multiple virtualized network functions in a service topology [21,22] forming a Service Function Chain (SFC) [7].

The instantiation of virtualized network services on the network substrate involves a series of tasks that are collectively known as the service deployment process. In particular, the NFV Resource Allocation (NFV-RA) [1] encompasses the most prominent deployment tasks, which are treated as optimization problems: (i) **composition** of service topologies; (ii) **embedding** of service topologies on virtualized environments; and (iii) **scheduling** network functions on virtualization servers. The deployment tasks directly impact the performance of virtualized network services, being a crucial task to properly execute a myriad of applications, from security to dependable services [23,4,6]. Challenges regarding the NFV-RA tasks have been widely explored in recent works [8–12].

The embedding tasks take into account the resources available on the virtualized environment. Three of these embedding techniques are defined next: mapping, selection, and placement. **Mapping** is dedicated to splitting and mapping service topologies on multiple domains. **Selection** allows network functions that have already been deployed to be shared among different services. Finally, **placement** corresponds to the allocation of network services to virtualization servers. These techniques also take into account the needs of stakeholders (*i.e.*, network providers, operators, and clients). These needs are typically specified as policies and Service Level Agreements (SLAs). The embedding task has also been considered to be a business opportunity for Network-as-a-Service platforms [24]. In this work, we propose a flexible and customizable embedding solution based on multi-objective genetic algorithms for the mapping technique.

2.2 Genetic Algorithms

Genetic algorithms are inspired by Darwin's theory of evolution and have been used to solve a myriad of optimization problems [25–27]. These algorithms evaluate individuals described by chromosomes (typically a vector representing a solution to the given problem). Chromosomes contain at least one gene, where a gene is a vector position. Each gene can be seen as a sub-problem and contains an allele. An allele, in turn, is a value that solves the sub-problem. Generations of individuals are submitted to the evolution processes, and are subject to operations such as crossover and mutation. The purpose of the evolutionary operations is to create new generations with individuals that are more fitted to solve the problem. Crossover operations select two or more individuals and partially combine their chromosomes to create a new descend-

ing individual. Mutation operations select random genes of an individual and replace their alleles (randomly or not). Note that genetic algorithms are metaheuristics and do not guarantee the globally optimal result. Furthermore, genetic algorithms execute stochastic operations. Thus, multiple executions of the same metaheuristic can produce different results for the same input. In particular, genetic algorithms are suitable to solve problems with large search spaces, providing results in workable execution time.

Two main classes distinguish genetic algorithms when they are used to solve optimization problems: mono-objective and multi-objective. Mono-objective algorithms optimize a single metric, while multi-objective algorithms optimize multiple metrics through a cost-benefit relationship — typically using Pareto frontiers. Individuals in a common frontier do not have a domination relationship (*i.e.*, they have at least one metric that presents better results compared to the other individuals in the same frontier). Optimization problems can also have constraints that invalidate portions of the search space. Different mechanisms can be used to tackle these constraints, for example, non-randomization of the initial population, the definition of immutable genes, and discarding invalid individuals. In this work, the problem of mapping virtual services on multiple domains is modeled as a multi-objective optimization problem. We use the Nondominated Sorting Genetic Algorithm II (NSGAII) [28] and Strength Pareto Evolutionary Algorithm 2 (SPEA2) [29] to implement the proposed solution.

2.3 Related Work

There are often multiple possible mappings of a given virtualized network service on a multi-domain environment. However, the performance of those distinct mappings varies when different policies, constraints, and optimization metrics are employed [1]. In this way, mapping solutions must evaluate the multiple alternatives to guarantee, for instance, the QoS (Quality of Service) and QoE (Quality of Experience) of the final results. Multi-domain mapping solutions can be organized in two main classes: (i) centralized and (ii) distributed. Centralized solutions execute on a single processing unit. These solutions receive as input information about the network service to embed and the available domains, and return candidate mappings that optimize a given evaluation setup. Distributed solutions send messages with embedding requests to the available domains. The domains evaluate the embedding requests, decide which functions they will host, and forward requests for the remaining functions to neighbor domains. Centralized solutions are described in [30,31, 10,16], and distributed solutions are presented in [32,15].

Dietrich *et. al.* [30] propose a solution to allocate virtualized network functions across a set of multiple providers infrastructures. The solution optimizes the multi-domain mapping relying on four static metrics: (i) minimization of financial costs; (ii) minimization of the number of different providers and domains; (iii) minimization of resource usage; and (iv) maximization of suit-

ability weights. In [31], an orchestration platform for multi-domain network services, called TeNOR, is proposed. TeNOR includes a multi-domain mapping solution which recovers information about financial costs, transmission delays, and resource usage to evaluate and optimizes (with a minimization objective) the candidate mappings. Finally, in [10], a multi-domain mapping strategy is proposed that considers hybrid scenarios where private and public domains provide optical network resources. The objective of that solution is to minimize financial costs (encouraging the allocation of functions in private domains) and the usage of frequency slots of the optical channels connecting the domains.

The solution proposed in [15] consists of a multi-domain mapping technique based on a vertex-centric algorithm. The solution triggers rounds of message exchanges among providers to find candidate mappings iteratively. The mapping algorithm uses a mechanism to avoid the concentration of the entire service on a single provider. This mechanism limits the number of network functions allocated in each round. This solution does not optimize any metric, returning to the user a set of candidate mappings that fulfill the allocation and instantiation constraints of the requesting service. With a method similar to [15], DistNSE [32] finds candidate mappings by exchanging messages among providers. This solution evaluates two optimization metrics: minimization of financial costs and stabilization of inter-domain load. However, DistNSE does not limit the number of network functions allocated on each provider.

In [16] a multi-domain mapping technique based on a mono-objective genetic algorithm is proposed. The objective of that solution is to allocate the network functions of a network service chain on a multi-domain environment based on a single indicator (E). This indicator represents multiple domain metrics, such as link availability, bandwidth, the number of network functions that each domain can host, among others. Besides mapping the network service across the multiple domains, the proposed solution also provides a backup schema for the mapped network functions. The backup schema intends to improve the overall service reliability. It is however possible map the main functions without the backup.

The solution proposed in [17] employs a mono-objective genetic algorithm to map virtualized network services on physical substrate nodes. The solution aims to optimize the usage of computing and networking resources by the network services. In this way, the authors propose an objective function that minimizes the residual capacity of nodes to host functions and links to handle their communication, given the mapped services. The objective function consists of the minimization of a single indicator called C_m. This indicator typically gets the best results when all the network functions are mapped to a single substrate node.

Table 1 summarizes features of the related works described above. Despite the fact that all these solutions evaluate multiple optimization metrics, they do not enable stakeholders to customize the evaluation setup (*i.e.*, it is not possible to define/select neither the metrics employed by the optimization process, nor the objectives/weights). This lack of customization makes it diffi-

Table 1 Summary of Mapping Solution Characteristics

	Mapping Solution (Reference)						
	[30]	[31]	[10]	[32]	[15]	[16]	[17]
Class	Centralized	Centralized	Centralized	Distributed	Distributed	Centralized	Centralized
Heuristic	Subgraph Mapping	N/A	K-Cut	Time to Live	N/A	Genetic Metaheuristic	Genetic Metaheuristic
Optimization Metrics	Financial Costs Number of Providers Computational Resources Adequacy (Weights)	Financial Costs Inter-domain Delay Computational Resources	Financial Costs Frequency Slots	Financial Costs Load Balancing	N/A	Links Availability Bandwidth Availability Domain Reliability Function Support	Bandwidth Efficiency Processing Efficiency
Customization of Evaluation Setup	✗	✗	✗	✗	✗	✗	✗
Support of Domain Dependencies	✗	✓	✓	✗	✗	✗	✗

cult to model and evaluate policies that are closely related to the deployment process (*e.g.*, maximum delay, maximum geographical distance). Furthermore, solutions in [31] and [10] present limitations in terms of the specification of domain dependencies (*i.e.* they do not allow the specification of which functions should be allocated to which particular domains). Thus, for example, these solutions are not suitable to embed hybrid services (*i.e.*, those in which physical network functions coexist with virtualized network functions along a service topology) in multi-domain environments.

Finally, the GA+LCB [16] solution provides limited support for the definition of domain dependencies – which can be specified only for the first and last network functions of a chain. Besides employing a mono-objective genetic algorithm, that solution does not enable the users to tune typical genetic features such as setting the crossover probability and choosing the selector algorithm. Similar to the GA+LCB, the solution proposed in [17] employs a mono-objective genetic algorithm that enables the users only to tune genetic features, such as crossover and mutation rates. However, this algorithm does not allow the user to change its objective function nor define domain dependencies of particular network functions in a service. It is possible to state that none of the presented solutions supports a customized evaluation setup or is based on multi-objective genetic algorithms to map services on multiple domains. In the present work, we argue that multi-objective genetic algorithms can support customizable evaluation setups while finding good candidate mappings in workable times for complex virtual network service mapping problems.

Genetic algorithms also have been used to solve other problems related to NFV deployment. In [33], [34], [35] and [36], solutions based on genetic algorithms are presented for network function placement. In particular, [33] uses a Multi-Objective Genetic Algorithm (MOGA) and NSGAII to optimize the allocation of network functions on virtualization servers taking two objectives into account: the minimization of the concentration of traffic on some connections while balancing the load across the multiple servers. The genetic algorithm proposed in [34] minimizes the amount of traffic between servers in different datacenters. Similarly, in [35], NSGAII is used to do service placement minimizing two optimization metrics: the number of virtualization servers used, and the concentration of traffic on the connection channels. Finally, in [36] the authors propose a placement solution that maximizes the average usage of available servers and minimizes the number of servers used to provide a single service. In the context of network function scheduling, an NSGAII-based solution is proposed [37] to minimize both server resource consumption

and the processing time of network functions running on those servers. Despite the fact that genetic algorithms are employed by those solutions, none executes multi-objective multi-domain mapping of virtualized network services. Furthermore, they also do not enable stakeholders to customize the evaluation setup (including optimization metrics, objectives, and weights).

3 Genetic Service Mapping

Genetic Service Mapping (GeSeMa) employs genetic algorithms to map virtualized network services across multiple administrative domains. GeSeMa is a flexible and customizable NFV mapping solution, enabling stakeholders to define service and network topologies, function and domain dependencies, and the evaluation setup (optimization metrics, objectives, weights, and constraints). This custom information is specified in a request document written in the YAML Ain't Markup Language (YAML). This section starts with a description of GeSeMa's request model and next presents the genetic strategy for multi-domain service mapping.

3.1 GeSeMa's Request Model

GeSeMa's request model, depicted in Figure 1, presents three main objects that define (i) the service topology and the network functions (`SERVICE`); (ii) the optimization metrics and objectives (`METRICS`); and (iii) the domains and their characteristics (`DOMAINS`). A string specified according to the rules of the Service ChAin Grammar (SCAG) [9] represents the service topology in the `SERVICE` object. Furthermore, for each network function defined in the service topology, there is a corresponding entry in the `FUNCTIONS` sub-object. This entry, identified by the function ID, specifies the minimum resource requirements, including memory, virtualized processing cores, and virtualized network interfaces, all defined as integer values.

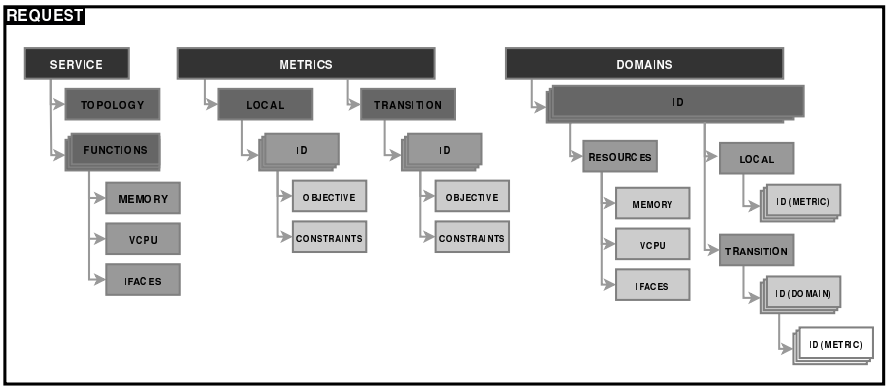

Fig. 1 GeSeMa Request Model

The METRICS object defines metrics and objectives used by the genetic algorithms of GeSeMa to search, evaluate, and optimize candidate mappings. Metrics are of two categories: *local* or *transition*. Local metrics are used to evaluate the allocation of network functions to domains, which correspond to the vertices of a graph representing the infrastructure on which the service is to be mapped. Local metrics include for instance the financial cost to allocate a function, the domain load, among others. Transition metrics are related to inter-domain connections – which correspond to the edges of the infrastructure graph. Examples of transition metrics include: delay, distance in hops, and geographical distance. The metrics and their categories are defined in the request model using LOCAL and TRANSITION sub-objects, respectively. Each of these sub-objects can define multiple metrics. A metric must be uniquely identified (by its ID), besides having two mandatory attributes: OBJECTIVE and CONSTRAINTS. The objective attribute shows the evaluation criteria for a particular metric, which can be either MAXIMIZATION or MINIMIZATION. The last attribute (CONSTRAINTS) consists of a list of strings each of which refers to the constraints of an optimization metric. Constraints define acceptance thresholds for the evaluation results of optimization metrics. In order to check results with respect to thresholds, relational operators ("<", ">", "<=", ">=", "==" and "! =") are employed to compare numerical values with thresholds.

Finally, the DOMAINS object defines the physical and virtual environments available and their transitions (connections). The domains attribute is represented by a directed graph $G = (V, E)$. The set of vertices V corresponds to the set of domains, and the set of edges E represents the logical connections between domains. The model keeps information about LOCAL metrics of each domain (vertex) and TRANSITION metrics associated with the edges. A particular domain is thus defined with three sub-objects: RESOURCES, LOCAL, and TRANSITION. The RESOURCES sub-object contains information about memory (MEMORY), virtual processing cores (VCPU), and virtual network interfaces (IFACES) made available by the domain. The LOCAL and TRANSITION sub-objects, in turn, define the metrics associated with domains and their connections obtained either with benchmarking or from catalogs; this is used by the optimization process. These sub-objects are also related to the METRICS object, and there must be a correspondence between metric identifiers and benchmark identifiers for both the LOCAL and TRANSITION sub-objects. In special, each entry of the TRANSITION sub-object determines to which domain the transition corresponds (using the domain unique identifier) and then defines the values of the optimization metrics for the transition.

3.2 The Proposed Genetic Multi-domain Mapping Method

GeSeMa executes two well-known genetic algorithms: NSGAII [28] and SPEA2 [29]. Note that the system can be extended, so that other algorithms can be included. The stakeholders can choose the genetic algorithm taking into account their characteristics, features of the requested service and the domains, plus

the evaluation setup provided. The genetic algorithms model the virtualized service mapping problem as follows:

1. **Individuals**: an individual's chromosome is modeled as a vector with $N > 1$ genes (*i.e.*, positions), where each gene corresponds to a network function of the service topology (*i.e.*, each function is mapped to a position in the vector). Genes contain alleles, represented by integer values in the range $[0, M-1]$ which correspond to the $M > 0$ domains available to map the network functions to. Note that, in GeSeMa, a valid individual is a candidate mapping.
2. **Population**: the initial population is created randomly or using a greedy-based mechanism. The creation of the initial population must guarantee the non-violation of function to domain dependencies, if there is any (*i.e.*, for instance, if a domain must host some function, the index corresponding to the specific domain is fixed to the allele of the constrained gene). To start with, the greedy mechanism sets a valid allele to the first gene and executes a greedy heuristic to define the other genes. If the population size gets greater than the number of possible alleles in the first gene, surplus individuals are created randomly. The population size $P > 0$ is a parameter defined by the stakeholders.
3. **Objectives and constraints**: GeSeMa evaluates objectives (with the evaluation setup) and constraints (*e.g.*, policies, network topology, computational resources, and dependencies) for all individuals of each generation. We use a taboo list to keep invalid individuals and avoid re-evaluations in case of new occurrences. Furthermore, mechanisms to recover specific types of invalid individuals are also defined (they will be discussed later).
4. **Selection**: the selection chooses individuals of a generation to crossover. GeSeMa uses a tournament mechanism that randomizes I individuals and returns the one that is the most fitted among them (*i.e.*, the one on the best Pareto frontier). The tournament size $I > 1$ is defined by the stakeholders.
5. **Crossover**: GeSeMa provides four crossover operators: Simulated Binary Crossover, Half Uniform Crossover, Partially Mapped Crossover, and Subtour Selection Crossover. The choice of a crossover operator should take into account their particularities and service request features. The crossover ratio (*i.e.*, operator application probability) is also defined by the stakeholders.
6. **Mutation**: the proposed solution employs two mutation operators: replacement and swap. Replacement chooses a random gene and replaces its allele by a new random value. Swap chooses two random genes and exchanges their alleles. Genes with domain constraints are never mutated. Similar to crossover, the stakeholders can define the mutation ratio.

GeSeMa executes two main procedures: (i) validation and configuration of the genetic algorithm; and (ii) creation and evolution of the population. The first procedure uses the model specified in Subsection 3.1 to validate the provided service request, thus mapping high-level structures to iterable elements (*i.e.*, dictionaries and lists). Next, the procedure checks previously

defined genetic parameters (*i.e.*, population size, tournament size, crossover operator/ratio, mutation operator/ratio, and number of generations) and, if valid, configures the genetic algorithm. Finally, the procedure generates a set of software elements employed for the creation and evolution of individuals by the second procedure.

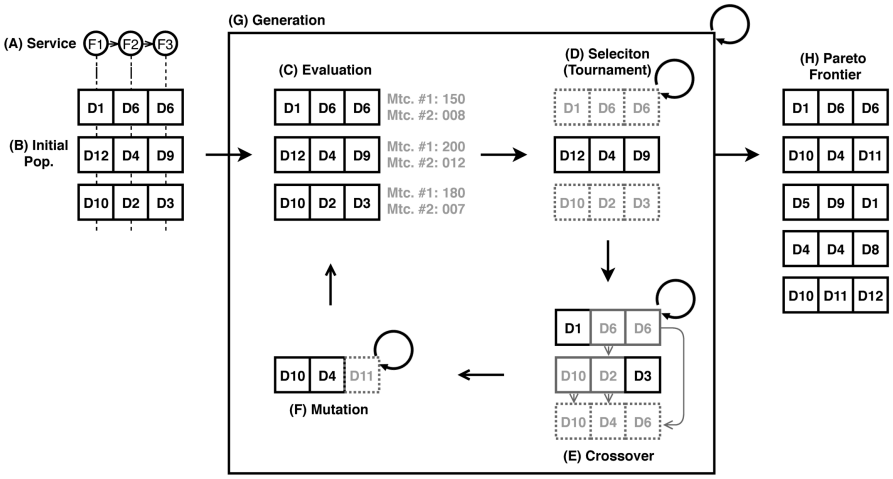

Fig. 2 Summary of the GeSeMa Workflow

Figure 2 summarizes the second procedure of GeSeMa. At first, the network service, encoded as a string according to the SCAG grammar, is converted to a format which is processed by the genetic algorithms (Figure 2: A and B). The initial population is generated with valid individuals in terms of the network topology (network domain transitions) and domain dependencies (constrained network functions pinned to their respective domains). Next, the individuals are evaluated (Figure 2: C) considering the availability of computational resources in the chosen domains and other constraints. In this way, each candidate is evaluated iteratively gene by gene for all metrics. Results of all genes are aggregated to define the overall result for each metric. Finally, GeSeMa executes selections (Figure 2: D) in addition to the crossover and mutation genetic operations (Figure 2: E and F, respectively) to evolve the population. All the stages depicted in Figure 2 C, D, E, and F represent the processing done to create a generation of individuals (Figure 2: G). Finally, after each generation has been created, the genetic algorithm saves the best fitted results (local Pareto frontier) to reuse in future generations. After a predetermined number of generations, GeSeMa returns the last Pareto frontier found as the final result (Figure 2: H).

In particular, the evaluation stage (Figure 2: C) produces information that is relevant for the next stages. The evaluation mechanism executes an iterative process, processing the chromosome of an individual, gene by gene. Local

optimization metrics are computed with the current gene's allele. Transition optimization metrics, in turn, are processed when a domain transition occurs. The transition metrics use the current gene's allele and the alleles of previously related genes. Besides the allele, for each gene there is a so called relation array with indexes of previously related genes (*i.e.*, previous network functions that have a connection with a particular network function in the requested service topology). In this way, linear chromosomes can represent branched service topologies. The set of partial evaluation results (*i.e.*, by gene/allele) are jointly processed, and the individuals are classified in terms of Pareto frontiers.

A taboo list includes all the invalid individuals. In case new invalid individuals occur, if they have already been included in the taboo list, three actions are possible: (i) discard individual (standard action); (ii) replace the individual by a new random individual (in case policies or network topology constraints are violated); or (iii) reduce domain redundancy (in case computational resources constraints are violated). The evaluation process does not consider domain dependencies since they are never violated, given the individuals of the initial population and the specification of constrained genes as static and non-mutable. Note that after GeSeMa returns the Pareto frontier, the user – which can be an automated system that requested the service mapping – has to determine which of the returned candidate mappings will be effectively adopted. This selection has to be done based on local priorities/needs that are defined individually, case-by-case.

4 Experimental Evaluation

In this section we present an evaluation of GeSeMa with two case studies[1]. The first case study is presented in Subsection 4.1 and consists of the mapping of a hierarchical cache service on a network consisting of 114 domains. This case study allows the evaluation of GeSeMa's convergence and execution time, among other metrics. In this case study, we also compare GeSeMa with other service mapping solutions that are based on classic heuristics. In the second case study (Subsection 4.2), we compare GeSeMa with the GA+LCB solution. In this case study, we employ the same network topology (consisting of 114 domains) on which we map a generic service chain that consists of 9 network functions. GeSeMa was configured with the same constraints of GA+LCB, and both were executed in the same evaluation setup.

4.1 Multimedia Hierarchical Cache Mapping

We evaluated GeSeMa through a case study in which a hierarchical multimedia cache is mapped onto a topology that corresponds to the Amazon AWS network, consists of 114 domains [38]. The service connects n individual cache

[1] The implementation is available at https://github.com/ViniGarcia/NFV-FLERAS

functions on a linear topology. The first function is the master cache that receives and spreads content updates from a multimedia server. The other cache functions both receive and make requests for content updates from/to their respective predecessor cache. In this way content is spread hierarchically across the service topology. The following constraints apply. Each domain can host at most one cache. The stakeholders objective is to maximize a local metric and a transition metric: the density of users of the multimedia service of the domain (local) and geographical distance between neighboring caches (transition). The objective is to map caches to regions that are geographically distant to each other and which have high user density.

The network topology is fully connected, in the sense that each node can communicate directly with any other, without having to employ intermediaries, thus the topology can be represented by a complete graph. Note that we could have employed an arbitrary topology (instead of a complete graph), however we opted to employ the complete graph in order to make the search space more challenging for GeSeMa. Each domain x (vertex) has an associated value for user density v_x^{du} (local metric). Each connection between two domains (edge) x and y has a value that corresponds to their geographical distance v_{xy}^{gd} (transition metric). In a preliminary step, we tested multiple genetic configurations to determine the operators and ratio of both crossover and mutation, tournament size, initial population mode, and population size.

For this case study we employed the SPEA2 algorithm; the results for NSGA2 were slightly inferior. SPEA2 was configured with the following parameters: SBX crossover ratio of 30%, replacement mutation ratio of 5%, selection by binary tournament, random initial population, and population size of 50 individuals. The service topologies consisted of 7, 9, and 11 caches. The tests were executed on a machine based on an Intel Core I5-3330 (3.0GHz) CPU with 8GB RAM (DDR3, 1600MHz), running Ubuntu 16.04, and Python 3.5.2. We chose service sizes (*i.e.*, the number of network functions) that could not be computed using exhaustive search in feasible time (on the same machine). All the files, programs, and scripts used are available at https://github.com/ViniGarcia/NFV-FLERAS/tree/GesemaExperiments. Experiments were executed 30 times with a confidence level of 95%.

The first experiment consists of a convergence test. The purpose of this experiment is to validate the feasibility of GeSeMa on the exploration and exploitation of the search space, thus converging to a Pareto frontier (despite of that being the global best frontier or not). Service mappings evolve for an undetermined number of generations, stopping when no modification occurs in the Pareto frontier after a set of 1500 generations. In case a modification has occurred, a new set of 1500 generations is executed. The results of this experiment are shown in Figures 3, 4, and 5 for the service topologies with 7, 9, and 11 functions, respectively. The lines represent the mean of the relative Pareto frontiers of best-fitted candidates at a particular generation, error bars show the best and worst Pareto frontiers found among the best candidates of the same generation.

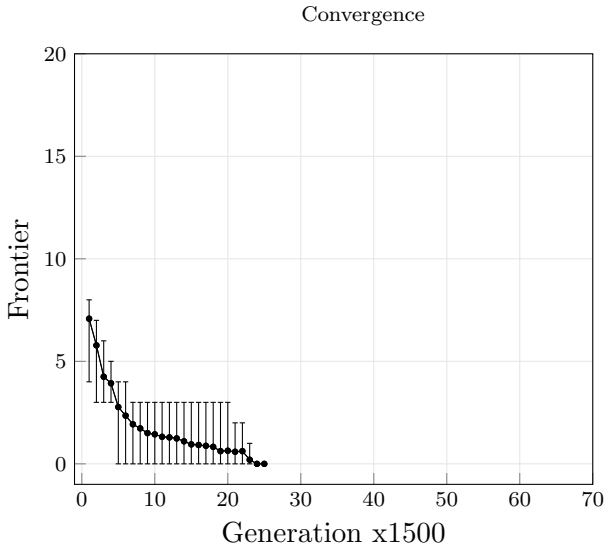

Fig. 3 GeSeMa's Convergence (7 VNFs)

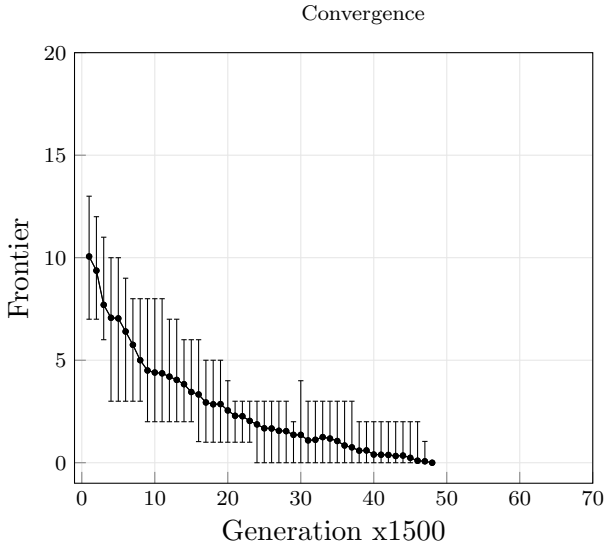

Fig. 4 GeSeMa's Convergence (9 VNFs)

The relative frontiers are computed as follows: (i) a predefined number of generations (in our case, 1500 generations) are evaluated by a preconfigured evolutionary algorithm in round r; (ii) the frontiers are computed and the fittest individuals from round r (hereby called "Pareto frontier r") are obtained and saved; (iii) the Pareto frontier r is merged with the Pareto frontier $r-1$, creating set $s_{[r-1,r]}$; (iv) the relative frontiers of $s_{[r-1,r]}$ are computed and its

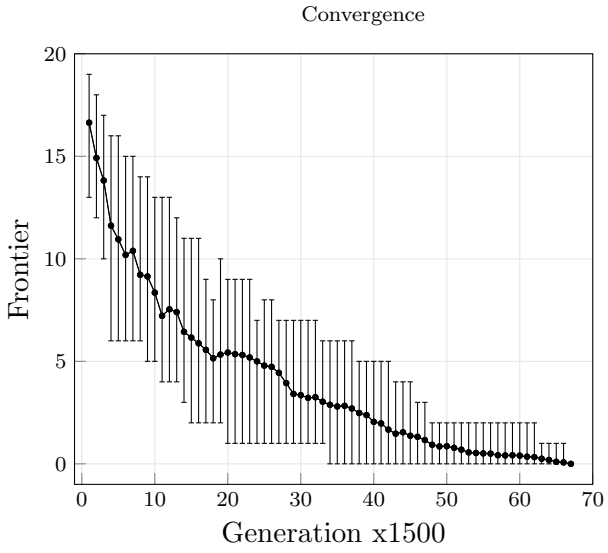

Fig. 5 GeSeMa's Convergence (11 VNFs)

Pareto frontier identified; (v.i) if no individual that appears exclusively in the Pareto frontier r occurs in the Pareto frontier of $s_{[r-1,r]}$, we consider that the algorithm has converged, and the next step - (vi) - is executed; (v.ii) if there is at least one individual that appears both in Pareto frontier r and in the Pareto frontier of $s_{[r-1,r]}$ (but not in frontier $r-1$), we consider that the population is still evolving, and steps (i), (ii), (iii), (iv), and (v) are repeated; (vi) the Pareto frontiers from each round ($[1;r]$) are merged in another in set $s_{[1,r]}$; (vii) the relative frontiers of $s_{[1,r]}$ are computed; and finally, (viii) information about the relative frontiers they appear in $s_{[1,r]}$ are assigned to each individual in the Pareto frontiers of rounds $[1;r]$.

In the first experiment, the execution of GeSeMa with the previously described configuration resulted in a positive correlation between the number of generations required to make the genetic algorithm converge and the problem complexity. This is a consequence of the large number of suboptimal candidates generated in the beginning of the execution of GeSeMa (exploration is more significant than exploitation in that phase), which are replaced in few generations. Thus, typically, the larger the search space (number of domains available) the more complex the problem is (number of VNFs in the service topology), and more exploration is needed to find appropriate regions of the search space to exploit. For the same reason, the number of frontier transitions increases as the search space and the problem complexity do. In the experiment, the mapping of the service topology with 7 functions converged after 36000 generations over 8 frontiers; with 9 functions it converged after 72000 generations over 13 frontiers; and with 11 functions it converged after 100500 generations over 19 frontiers.

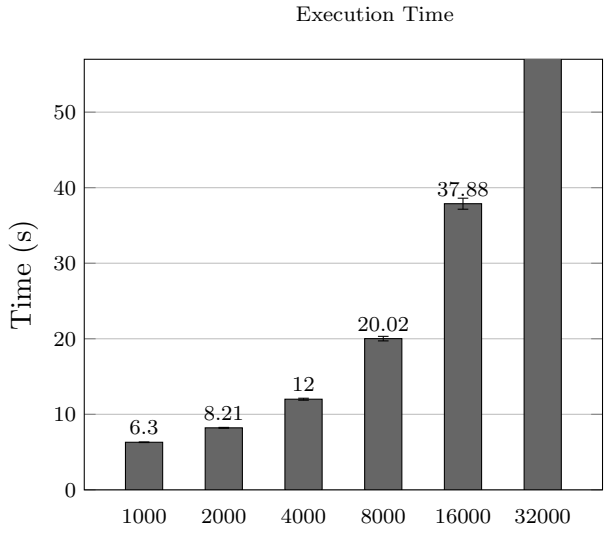

Fig. 6 GeSeMa Execution Time (7 VNFs)

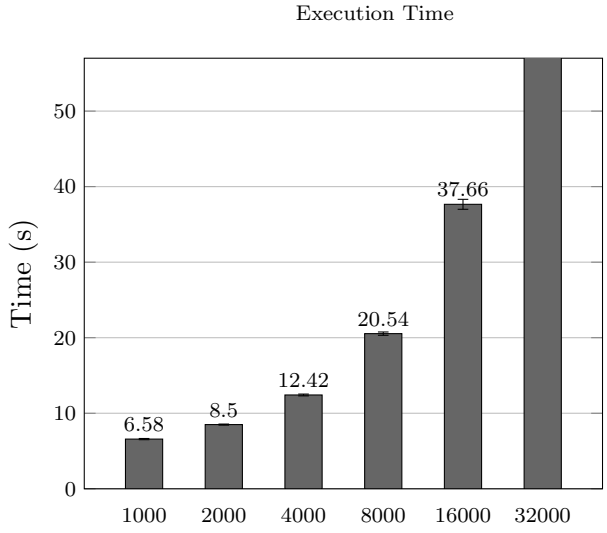

Fig. 7 GeSeMa Execution Time (9 VNFs)

The second experiment shows the execution time (in seconds) of GeSeMa as the number of generations increases. This experiment was executed for the topologies with 7 (Figure 6), 9 (Figure 7), and 11 (Figure 8) functions. The results reveal a positive correlation of the execution time and the number of generations. However, we also noted that the execution time presents little variation as the service topology sizes vary for the same number of generations. This is a consequence of the constraint that specifies that each domain

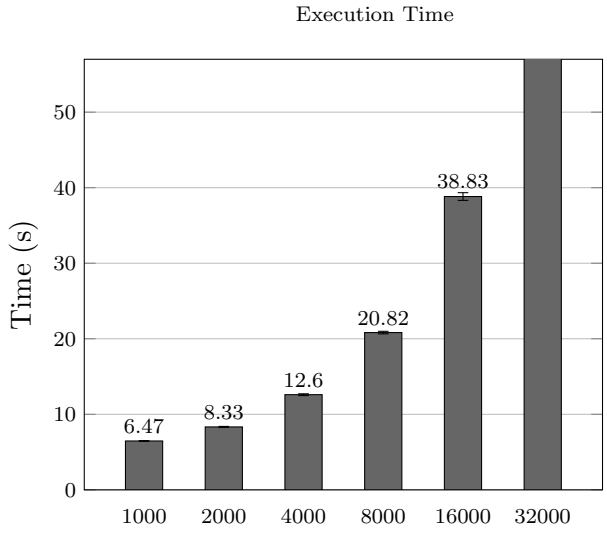

Fig. 8 GeSeMa Execution Time (11 VNFs)

hosts at most a single cache. Thus, the probability of creating invalid candidates increases as the number of chromosomes does. However, these invalid candidates are discarded before they are evaluated, which reduces the impact on the execution time. But, despite maintaining the execution time stable, this makes it more difficult to improve the candidates and also delays the convergence of the genetic algorithm.

The third experiment modifies the evaluation setup employed to map the network service. These modifications do not imply any changes of the algorithm. All the modifications are done in the service request document through the `METRICS` object. We considered three evaluation setups: (i) the standard setup with the maximization of both the user density and the geographic distance between neighboring caches; (ii) a mono-objective setup with the maximization of the user density; and (iii) a mono-objective setup with the maximization of the geographic distance between neighboring caches. Each evaluation setup was submitted to GeSeMa with the previous configurations. In this experiment, our halting criteria is based on time, with the main loop of GeSeMa being executed for 10 seconds.

Table 2 Modifications On The Evaluation Setup

	Maximization		
	User Density Geographical Distance	**User Density**	**Geographical Distance**
User Density	129261	141176	156314.20
Geographical Distance	144831.67	(42313.69, 141176)	(156314.20, 106576)
User Density	–	150153	–
Geographical Distance	–	–	162025.35

The results of the third experiment are shown in Table 2. The first column shows results for the multi-objective setup with the best cost-benefit relation and the same weight (i.e. equal to 0.5) for each particular optimization metric. The second column shows the best candidate in the Pareto frontier for the multi-objective optimization problem, but only considering the user density. Furthermore, we present the best result for the mono-objective user density evaluation in the second row of the same column. The last column shows the best candidate in the Pareto frontier for the multi-objective optimization problem, taking into account only the geographical distance in the first row. We also present the best result for the third evaluation setup (mono-objective geographical distance) in the third row of the third column. In the second and third columns, for the multi-objective evaluation setup row, data between parenthesis represent the results for both metrics employed to evaluate the candidate: (geographical distance, user density).

The results demonstrate the capacity of GeSeMa to deal with different evaluation setups by just modifying the service request document. For sure, the most specific the evaluation setup is, the best the resulting mapping candidates tend to be. So, in the experiment, if only the user density is relevant for some specific problem, it is better to employ mono-objective optimization. The same occurs for the geographic distance between neighboring caches. However, multi-objective optimizations are the best options when the evaluation setup relies on multiple metrics. Thus, later it is possible to evaluate the returned Pareto frontier according to some criteria, such as using weighs.

The next experiments compare results from GeSeMa with two alternative mapping solutions based on heuristic search: random search and k-stochastic greedy search. The random search randomizes the domains that will allocate the network functions of a given service topology. This solution must guarantee the creation of valid candidates regarding domain dependencies and network topology constraints. The stochastic k-greedy search randomizes $k \geq 1$ domains that can possibly allocate a network function of a service topology. It uses a greedy strategy to find the best local option taking into account the evaluation setup. Similar to the random search, the stochastic k-greedy algorithm must also guarantee that only valid candidates are created, with respect to domain dependencies and network topology constraints. Both solutions were adapted to process the same service request document used by GeSeMa, thus enabling the customization of their evaluation setups and a fair comparison with GeSeMa.

We configured all the solutions to execute their main loop for 10 seconds. The main loop consists of the creation and evaluation of individuals. Thus, we did not take into account in the execution time other internal routines, such as request processing, graph instantiation, and output recording. Other relevant configurations are the following. For GeSeMa running the SPEA2 algorithm: we employed a SBX crossover ratio of 30%, a replacement mutation ratio of 5%, selection was done with a binary tournament, the random initial population, and population size was of 50 individuals. In the Figures, results for the Classic random search is labeled as “Random”; stochastic k-greedy search with $k = 2$

and $k = 4$ are labeled with "S2-Greedy" and "S4-Greedy", respectively. Each execution of the mapping solutions returns the local Pareto frontier, which is afterwards used to compute the relative Pareto frontiers (*i.e.*, candidates in the relative Pareto frontier dominate all the other candidates in the local Pareto frontiers found in any execution of any algorithm).

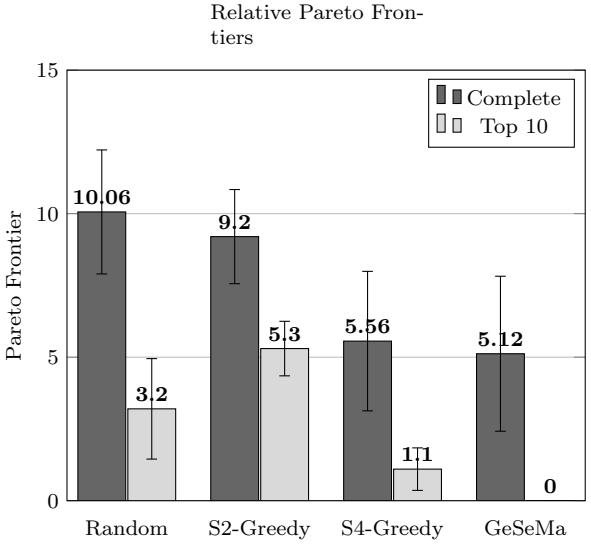

Fig. 9 Frontiers Comparison (7 VNFs)

The fourth experiment presents the mean of the relative Pareto frontiers from the candidates returned by the mapping solutions. Figures 9, 10, and 11 show these results for the topologies with, respectively, 7, 9 and 11 functions. Grey bars show results obtained for all the candidates in the relative frontiers (case "complete"), while white bars show results for the top ten candidates of the relative frontiers of each solution (case "top 10"). GeSeMa applied for mapping services consisting of 7 and 9 network functions presented the best relative Pareto frontier mean for both the "complete" and "top 10" cases. For mapping 11 functions, GeSeMa presented a worse result for the "complete" case in comparison with S4-Greedy. However, for 11 functions, GeSeMa still reaches better results than S4-greedy in the "top 10" case. The degradation of the GeSeMa in the "complete" case occurs due to the higher number of candidates in the Pareto frontiers (373 candidates) in comparison with the number of candidates of S4-Greedy (200 candidates). Thus, the best candidates from GeSeMa are better than the best candidates from S4-Greedy (see "top 10" case), but GeSeMa returns more results that include candidates less fitted than the ones returned from S4-Greedy. The total number of candidates recovered from each mapping solution during the experiments are presented in Table 3. Another relevant observation is that within the top ten candi-

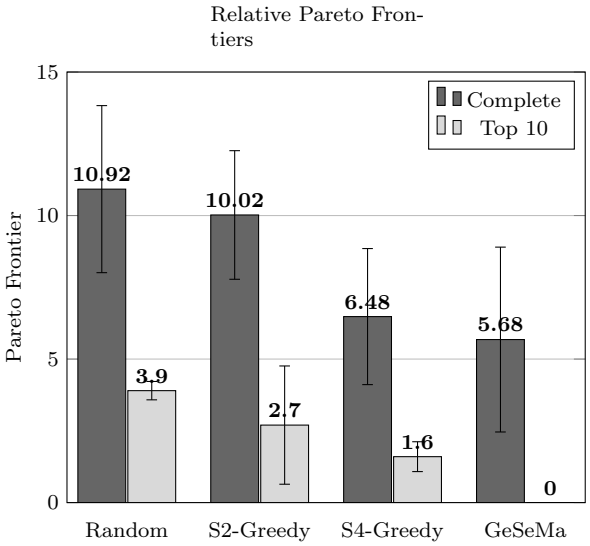

Fig. 10 Frontiers Comparison (9 VNFs)

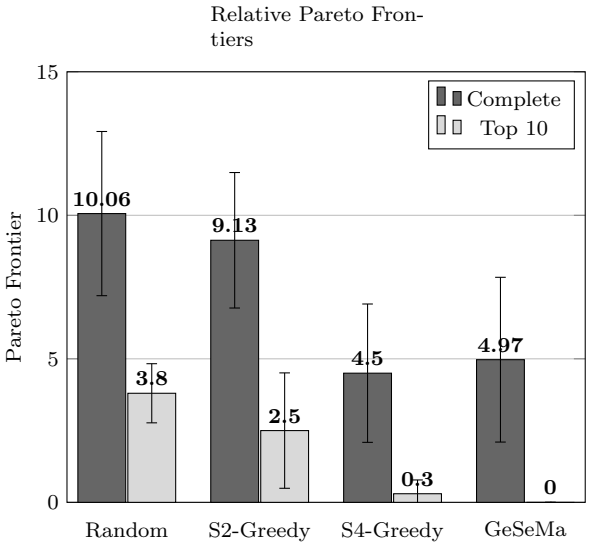

Fig. 11 Frontiers Comparison (11 VNFs)

dates, only GeSeMa presented all candidates in the relative Pareto frontier (*i.e.*, 0). This fact shows the efficiency and stability of the proposed solution for mapping the service topology even as the number of functions varies.

Finally, the last experiments compare the total execution time (in seconds) for the solutions to map topologies with 7 (Figure 12), 9 (Figure 13), and 11 (Figure 14) functions. Note that 10 seconds of the total execution time is re-

Table 3 Number of Candidates Recovered From Mapping Solutions

	Random	**S2-Greedy**	**S4-Greedy**	**GeSeMa**
7 Functions	291	241	225	353
9 Functions	338	299	312	422
11 Functions	274	242	200	373

served to execute the main loop of each of the solutions. The extra time spent by the various solutions to run internal operations, such as request processing, graph creation, object instantiation, and output recording, were very similar. The differences of the total execution times did not exceed 1.05 seconds - 7.6% - even in the worst scenario (mapping of 11 functions, which presented the largest difference between S2-Greedy and Random). In particular, GeSeMa presents slightly smaller execution times than Random (0.39, 0.29, and 0.21 seconds to map 7, 9, and 11 functions respectively), and slightly larger execution times than S2-Greedy (0.31, 0.55, and 0.84 seconds to map 7, 9, and 11 functions) and S4-Greedy (0.2, 0.39, and 0.59 for 7, 9, and 11 functions, respectively).

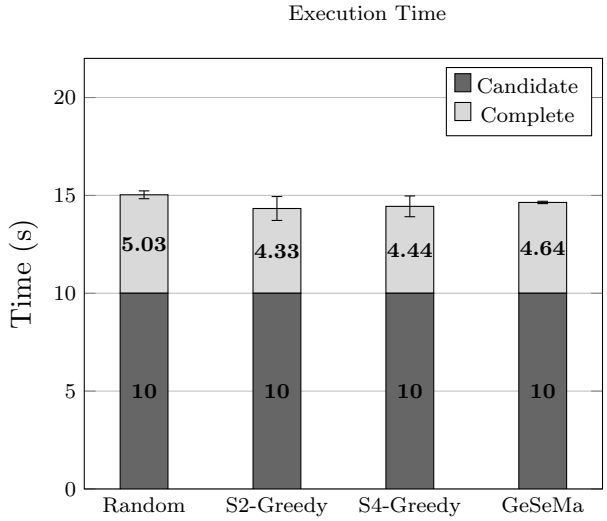

Fig. 12 Exec. Time Comparison (7 VNFs)

4.2 Comparison Between GeSeMa and GA+LCB

GA+LCB is a mapping solution based on a mono-objective genetic algorithm [16]. In addition to the traditional mapping process (mapping the main network functions of a network service), GA+LCB includes a backup map-

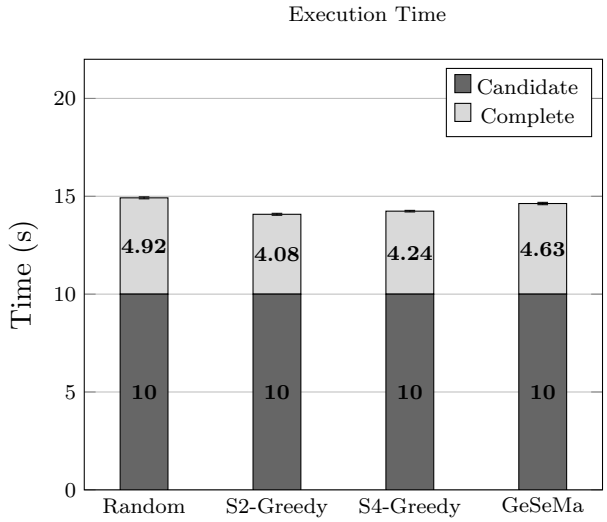

Fig. 13 Exec. Time Comparison (9 VNFs)

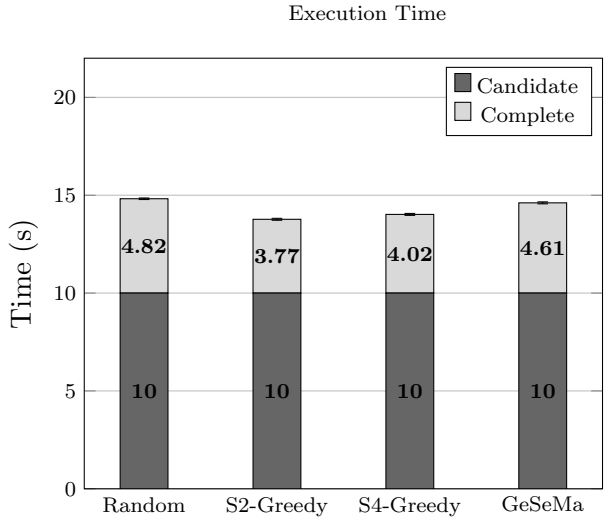

Fig. 14 Exec Time Comparison (11 VNFs)

ping mechanism that creates a backup schema for the requested network service. However, as GeSeMa does not create backups, for comparison purposes GA+LCB is executed to map the main functions, not the backups. The GA+LCB objective function was configured as a maximization of the modified domain importance (imp_k from [16]), which consists of the maximization of three metrics – link availability (da_k), bandwidth availability (dc_k), and the availability factor (A_k) – and the minimization of a single metric – inter-

domain delay (dd_k). The GA+LCB solution computes this evaluation setup as $E = w1{*}nor(da_k){+}w2{*}nor(dc_k){+}w3{*}nor(A_k){+}w4{*}(1{-}nor(dd_k))$, where nor indicates a normalization function and w_n the metric weight ($\sum_{n=1}^{4} w_n = 1$). GeSeMa, in turn, evaluates the candidates using the same metrics and objectives but evaluating the Pareto frontiers.

In this case study, GeSeMa and GA+LCB are employed to map a network service with 9 generic network functions. The network topology is the same consisting of 114 AWS global domains used in the experiments of Subsection 4.1. The mapping of network functions should not exceed the computational resource limits of the domains, and no more than two network functions should be mapped to each domain. Both solutions were configured to obey both maximum delay and minimum availability constraints. The values for metrics dc_k and A_k are defined randomly in the intervals $[100, 500]$ and $[0.95, 0.99]$, respectively; the value of da_k is 114 for all the domains (the network topology is a complete graph); and the value of dd_k is defined considering the geographical distance between pairs of domains $gd_{k,k+n}$ in the curve $gd_{k,k+n} * (1 - e^{nor(gd_{k,k+n})*-4}) * 0.05$. As required by GA+LCB, the initial domain and the final domain are specified in the request document.

The genetic parameters of both GA+LCB and GeSeMA were configure to be as similar as possible. GA+LCB includes a crossover of half of the population using a native algorithm. Thus, we configured GeSeMa with a crossover ratio of 0.5 using the SBX algorithm (SBX has similar behavior to the GA+LCB crossover algorithm). The mutation ratio is set to 0.05, GA+LCB uses a specific, simple mutation algorithm; GeSeMa uses a replacement mutation algorithm. GA+LCB executes a traditional roulette selector; GeSeMa employs a binary tournament selector. GA+LCB creates the initial population based on a k-shortest path algorithm; GeSeMa creates the initial population randomly. GA+LCB uses a self-designed mono-objective genetic algorithm with elitism features; GeSeMa adopts SPEA2. The population size of 50 was the same for both solutions, as well as the execution of 20000 generations. The experiments were performed in the same machine used for the first case study (Subsection 4.1). All the algorithms, scripts, datasets, and requests used to compare GA+LCB and GeSeMa are available at https://github.com/ViniGarcia/NFV-FLERAS/tree/GesemaExperiments. Experiments were repeated 30 times with a confidence level of 95%.

The first experiment compares the quality of the candidates returned by GeSeMa and GA+LCB. We use the mean of the relative Pareto frontiers for the comparison. Figure 15 shows the mean frontiers of candidates returned for two cases: "complete" (frontiers of all candidates from all executions are used to compute the mean value) and "top 10" (frontiers of top ten candidates of all executions are used to compute the mean value). The GA+LCB solution presented a better mean of the relative frontiers in the "complete" case. However, GeSeMa surpasses the GA+LCB results in the "top 10" experiment. This behavior occurs due to the number of candidates returned from GA+LCB at each execution: precisely one. Thus, GA+LCB returns a total of 30 candidates with the best E value achieved in each execution of the solution. GeSeMa, in

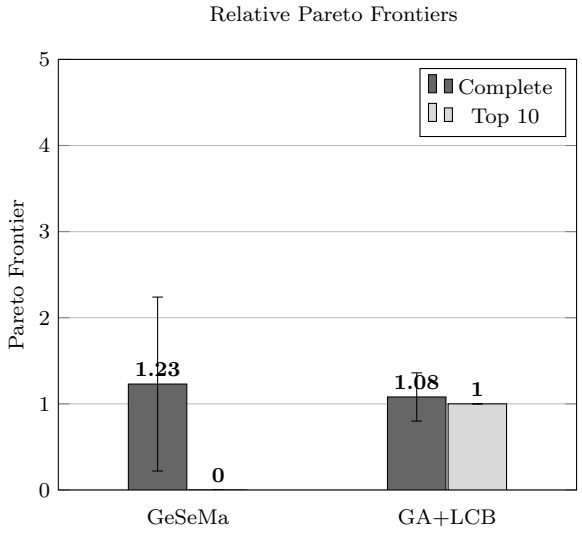

Fig. 15 Frontiers Comparison (Genetic)

turn, returns the entire Pareto frontier, which typically contains multiple candidates. In this experiment, GeSeMa provided approximately 49 candidates per execution, from a total of 1463 candidates evaluated in the "complete" case. Some of these candidates are not better fitted than the ones returned by the GA+LCB, but, as demonstrated by the "top 10" case, the best candidates of GeSeMa are more fitted than the best candidates of GA+LCB.

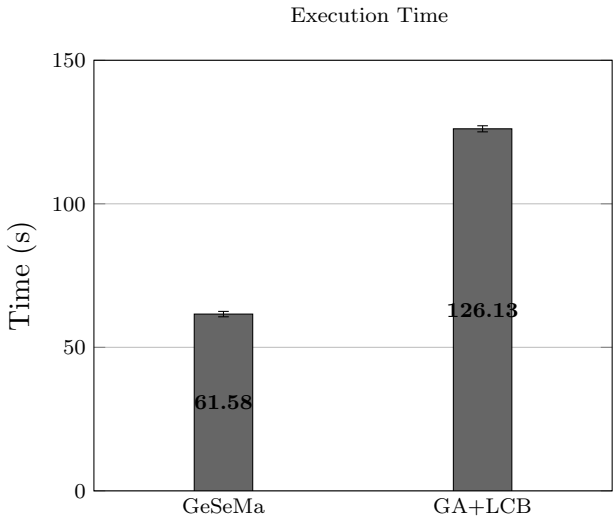

Fig. 16 Exec. Time Comparison (Genetic)

The second experiment compares the mean execution times of GA+LCB and GeSeMa to map the service in the AWS network topology. Figure 16 shows the results. GeSeMa presented a better mean execution time, being 104% faster than GA+LCB. These results can be explained as follows. First, GeSeMa employs a lightweight random initial population strategy, while GA+LCB uses a k-smallest path heuristic to create a possibly more fitted initial population. Thus, the GA+LCB strategy requires the execution of shortest path algorithms that take quite a lengthy amount of time to run in large network topologies. Second, the evaluation of multiple optimization metrics with a mono-objective genetic algorithm requires an aggregated index (in GA+LCB, called E). The creation of this index imposes an extra time to process the normalization and weighting required by each generation. Third, GA+LCB does not have any mechanism to avoid the evaluation of candidates which have been already discarded but reappear during the execution of the genetic algorithm. GeSeMa, in turn, uses a taboo list to ignore those candidates.

5 Conclusion

The deployment of virtualized network functions and services is one of the most important process of their lifecycle. Resource allocation (NFV-RA) is the core of the deployment process and consists of three tasks: composition; embedding; and scheduling. In this context, multi-domain mapping allows embedding a network service across a distributed environment consisting of multiple administrative domains. Current multi-domain mapping solutions do not enable the stakeholders to customize their evaluation setups. In this paper we presented Genetic Service Mapping (GeSeMa), an intelligent mapping solution that uses genetic metaheuristics to execute a customizable mapping of service topologies across multi-domain environments. We evaluated the feasibility and performance of GeSeMa compared with other alternatives. Results confirm that GeSeMa produces mappings of superior quality in comparison with both classic and state-of-the solutions, while presenting low execution times.

Providing a flexible, customizable evaluation setup to solve the NFV-RA problem allows stakeholders to get exactly what they need, also taking into account their restrictions. That is the main purpose of GeSeMa: to optimize service mapping according to network policies defined by those that are able to describe their specific scenarios, services, resource availability, constraints, and also the requirements of their end-users. This flexibility also brings further possibilities. A fine-grained customized evaluation setup can include aspects that are not purely technical and commercial. In this way, social and environmental objectives can also be included in the evaluation setup, such as favoring the service mapping to some regions due to humanitarian reasons or due to the use of renewable energy sources.

In future work, we aim to make GeSeMa dynamic and adaptive, continuously evolving the service mapping and suggesting migrations in real-time as

both the network topology and the evaluation setup change. In this way, the solution will be able to tackle the dynamism of devices in particular networks, such as 5G and IoT. Furthermore, we envision that an adaptive GeSeMa can work well in catastrophic scenarios and battlefield networks, where the available resources are unstable and can change at any time. Finally, taking advantage of the flexibility of the evaluation setup customization, we aim to release a service version of GeSeMa to be explored in the context of NFV marketplaces, such as FENDE [3] and T-NOVA [39].

References

1. Juliver Gil Herrera and Juan Felipe Botero. Resource allocation in nfv: A comprehensive survey. *Transactions on Network and Service Management*, 13(3):518–532, 2016.
2. Bo Yi, Xingwei Wang, Keqin Li, Sajal K. Das, and Min Huang. A comprehensive survey of network function virtualization. *Computer Networks*, 133:212–262, 2018.
3. Lucas Bondan, Muriel Figueredo Franco, Leonardo da Cruz Marcuzzo, Giovanni Venancio, Ricardo Santos, Ricardo Pfitscher, Eder Scheid, Burkhard Stiller, Filip De Turck, Elias Procopio Duarte, Alberto Egon Schaeffer-Filho, Carlos Raniery Paula dos Santos, and Lisandro Zambenedetti Granville. Fende: Marketplace-based distribution, execution, and life cycle management of vnfs. *Communications Magazine*, 57(1):13–19, 2019.
4. Vinicius Fulber-Garcia, Guilherme de Freitas Gaiardo, Leonardo da Cruz Marcuzzo, Raul Ceretta Nunes, and Carlos Raniery Paula dos Santos. Demons: A ddos mitigation nfv solution. In *International Conference on Advanced Information Networking and Applications*, pages 769–776. IEEE, 2018.
5. Rogerio C. Turchetti and Elias Procopio Duarte. Implementation of failure detector based on network function virtualization. In *International Conference on Dependable Systems and Networks Workshops*, pages 19–25. IEEE, 2015.
6. Giovanni Venâncio, Rogério C Turchetti, Edson T. Camargo, and Elias P. Duarte Jr. Vnf-consensus: A virtual network function for maintaining a consistent distributed software-defined network control plane. *International Journal of Network Management*, 31(3):e2124, 2021.
7. Paul Quinn and Tom Nadeau. Problem Statement for Service Function Chaining - RFC 7498. Technical report, Internet Engineering Task Force, 2015.
8. Juliver Gil-Herrera and Juan Felipe Botero. A scalable metaheuristic for service function chain composition. In *Latin-American Conference on Communications*, pages 1–6, Guatemala City, Guatemala, 2017. IEEE.
9. Vinicius Fulber-Garcia, Marcelo Caggiani Luizelli, Carlos Raniery Paula dos Santos, and Elias Procopio Duarte. Cusco: A customizable solution for nfv composition. In *International Conference on Advanced Information Networking and Applications*, pages 204–216, Caserta, Italy, 2020. Springer.
10. Yixiang Wang, Ping Lu, Wei Lu, and Zuqing Zhu. Cost-efficient virtual network function graph (vnfg) provisioning in multidomain elastic optical networks. *Journal of Lightwave Technology*, 35(13):2712–2723, 2017.
11. Guy Even, Moti Medina, and Boaz Patt-Shamir. On-line path computation and function placement in sdns. *Theory of Computing Systems*, 63(2):306–325, 2019.
12. Sameer G Kulkarni, Wei Zhang, Jinho Hwang, Shriram Rajagopalan, KK Ramakrishnan, Timothy Wood, Mayutan Arumaithurai, and Xiaoming Fu. Nfvnice: Dynamic backpressure and scheduling for nfv service chains. In *Special Interest Group on Data Communication*, pages 71–84, Los Angeles, USA, 2017. ACM.
13. Alexandre Huff, Giovanni Venancio, Vinicius Fulber-Garcia, and Elias P. Duarte. Building multi-domain service function chains based on multiple nfv orchestrators. In *Conference on Network Function Virtualization and Software Defined Networks*, pages 19–24. IEEE, 2020.

14. F. Schardong, I. Nunes, and A. Schaeffer-Filho. Nfv resource allocation: a systematic review and taxonomy of vnf forwarding graph embedding. *Computer Networks*, 185:107726, 2021.
15. Qiong Zhang, Xi Wang, Inwoong Kim, Paparao Palacharla, and Tadashi Ikeuchi. Vertex-centric computation of service function chains in multi-domain networks. In *NetSoft Conference and Workshops*, pages 211–218, Seoul, South Korea, 2016. IEEE.
16. Yimin Li, Lifang Gao, Siya Xu, Qinghai Ou, Xinyu Yuan, Feng Qi, Shaoyong Guo, and Xuesong Qiu. Cost-and-qos-based nfv service function chain mapping mechanism. In *Network Operations and Management Symposium*, pages 1–9, Budapest, Hungary, 2020. IEEE.
17. Panteleimon Rodis and Panagiotis Papadimitriou. Intelligent network service embedding using genetic algorithms. In *Symposium on Computers and Communications*, pages 1–7, Athens, Greece, 2021. IEEE.
18. Justine Sherry, Shaddi Hasan, Colin Scott, Arvind Krishnamurthy, Sylvia Ratnasamy, and Vyas Sekar. Making Middleboxes Someone Else's Problem: Network Processing As a Cloud Service. *Computer Communication Review*, 42(4):13–24, 2012.
19. NFVISG ETSI. Network functions virtualization, white paper. `$https://portal.etsi.org/NFV/NFV_White_Paper.pdf$`, 2012. Accessed in 01 Apr. 2021.
20. Giovanni Venâncio, Vinicius Fulber-Garcia, Leonardo da Cruz Marcuzzo, Thales Nicolai Tavares, Muriel Figueredo Franco, Lucas Bondan, Alberto Egon Schaeffer-Filho, Carlos Raniery Paula dos Santos, Lisandro Zambenedetti Granville, and Elias Procopio Duarte. Beyond vnfm: Filling the gaps of the etsi vnf manager to fully support vnf life cycle operations. *International Journal of Network Management*, n/a(n/a):e2068, n/a.
21. Vinicius Fulber-Garcia, Elias Procopio Duarte, Alexandre Huff, and Carlos Raniery Paula dos Santos. Network service topology: Formalization, taxonomy and the custom specification model. *Computer Networks*, 178:107337, 2020.
22. Alexandre Huff, Giovanni Venancio, Leonardo da C. Marcuzzo, Vinicius Fulber-Garcia, Carlos R. P. dos Santos, and Elias P. Duarte. A holistic approach to define service chains using click-on-osv on different nfv platforms. In *Global Communications Conference*, pages 1–6. IEEE, 2018.
23. Thales Nicolai Tavares, Leonardo da Cruz Marcuzzo, Vinicius Fulber-Garcia, Giovanni Venâncio de Souza, Muriel Figueredo Franco, Lucas Bondan, Filip De Turck, Lisandro Zambenedetti Granville, Elias Procópio Duarte Junior, Carlos Raniery Paula dos Santos, et al. Niep: Nfv infrastructure emulation platform. In *International Conference on Advanced Information Networking and Applications*, pages 173–180. IEEE, 2018.
24. Amina Boubendir, Emmanuel Bertin, and Noemie Simoni. Naas architecture through sdn-enabled nfv: Network openness towards web communication service providers. In *Network Operations and Management Symposium*, pages 722–726, Istanbul, Turkey, 2016. IEEE.
25. Oğuzhan Hasançebi and Fuat Erbatur. Evaluation of crossover techniques in genetic algorithm based optimum structural design. *Computers & Structures*, 78(1-3):435–448, 2000.
26. Bogdan Tomoyuki Nassu, Elias P. Duarte Jr, and Aurora T. Ramirez Pozo. A comparison of evolutionary algorithms for system-level diagnosis. In *Annual Conference on Genetic and Evolutionary Computation*, pages 2053–2060, 2005.
27. Elias P. Duarte Jr, Aurora T. R. Pozo, and Bogdan T. Nassu. Fault diagnosis of multiprocessor systems based on genetic and estimation of distribution algorithms: a performance evaluation. *International Journal on Artificial Intelligence Tools*, 19(01):1–18, 2010.
28. Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and Tamt Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, 6(2):182–197, 2002.
29. Eckart Zitzler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm. *TIK Report*, 103, 2001.
30. David Dietrich, Ahmed Abujoda, and Panagiotis Papadimitriou. Network service embedding across multiple providers with nestor. In *Networking Conference*, pages 1–9, Toulouse, France, 2015. IEEE.

31. Jordi Ferrer Riera, Josep Batallé, José Bonnet, Miguel Días, Michael McGrath, Giuseppe Petralia, Francesco Liberati, Alessandro Giuseppi, Antonio Pietrabissa, Alberto Ceselli, et al. Tenor: Steps towards an orchestration platform for multi-pop nfv deployment. In *NetSoft Conference and Workshops*, pages 243–250, Seoul, South Korea, 2016. IEEE.
32. Ahmed Abujoda and Panagiotis Papadimitriou. Distnse: Distributed network service embedding across multiple providers. In *International Conference on Communication Systems and Networks*, pages 1–8, Bangalore, India, 2016. IEEE.
33. Jiuyue Cao, Yan Zhang, Wei An, Xin Chen, Yanni Han, and Jiyan Sun. Vnf placement in hybrid nfv environment: Modeling and genetic algorithms. In *International Conference on Parallel and Distributed Systems*, pages 769–777, Wuhan, China, 2016. IEEE.
34. Francisco Carpio, Samia Dhahri, and Admela Jukan. Vnf placement with replication for load balancing in nfv networks. In *International Conference on Communications*, pages 1–6, Paris, France, 2017. IEEE.
35. Selma Khebbache, Makhlouf Hadji, and Djamal Zeghlache. A multi-objective non-dominated sorting genetic algorithm for vnf chains placement. In *Annual Consumer Communications & Networking Conference*, pages 1–4, Las Vegas, USA, 2018. IEEE.
36. Sanaz Tavakoli-Someh and Mohammad Hossein Rezvani. Utilization-aware virtual network function placement using nsga-ii evolutionary computing. In *Conference on Knowledge Based Engineering and Innovation*, pages 510–514, Tehran, Iran, 2019. IEEE.
37. Ningning Ma, Jiao Zhang, and Tao Huang. A model based on genetic algorithm for service chain resource allocation in nfv. In *International Conference on Computer and Communications*, pages 607–611, Chengdu, China, 2017. IEEE.
38. Wikileaks. Amazon atlas 2015. `https://wikileaks.org/amazon-atlas/`, 2018. Accessed in 01 Apr. 2021.
39. Georgios Xilouris, Eleni Trouva, Felicia Lobillo, João M. Soares, Jorge Carapinha, Michael J. McGrath, George Gardikis, Pietro Paglierani, Evangelos Pallis, Letterio Zuccaro, et al. T-nova: A marketplace for virtualized network functions. In *European Conference on Networks and Communications*, pages 1–5. IEEE, 2014.

Declarations

Ethical Approval

Not applicable.

Competing Interests

The authors state that this paper does not present any financial competing interests; this research is an initiative of the Network, Distributed Systems, and Security Laboratory from the Federal University of Paraná (UFPR), Brazil, representing its academic interests.

Authors' Contributions

Vinicius Fulber-Garcia and Elias Duarte Jr. wrote the main manuscript text; Vinicius Fulber-Garcia, Marcelo Luizelli, Carlos Paula dos Santos and Eduardo Spinosa worked on the programming, tests, and figures. All authors reviewed the manuscript.

Funding

This paper is not a result of funded research.

Availability of Data and Materials

All the programming files and experiment scripts are available at the following repository: https://github.com/ViniGarcia/NFV-FLERAS.

Vinicius Fulber-Garcia received his Ph.D. in Computer Science from the Informatics Department of the Federal University of Paraná (UFPR - Brazil) in 2022. He also holds a Computer Science degree (2016) from the Federal University of Santa Maria (UFSM - Brazil) and a Master's degree (2019) in Computer Science from the UFSM Post-Graduate Program in Computer Science. Currently, he is a Professor of Computer Science at the Federal University of Paraná (UFPR), Brazil.

Marcelo Caggiani Luizelli holds a Ph.D. in Computer Science from the Federal University of Rio Grande do Sul (UFRGS). During his Ph.D., Luizelli was a visiting researcher at Nokia Bell Labs and the Technion University, both in Israel. Currently, he is an associate faculty at the Federal University of Pampa (Unipampa) in Brazil. His research activities have been focusing on emerging paradigms such as SDN, NFV, and programmable data planes.

Carlos Raniery Paula dos Santos is Adjunct Professor of Computer Science at the Department of Applied Computing of the Federal University of Santa Maria (UFSM), Brazil. He holds Ph.D. (2013) and M.Sc. (2008) degrees in Computer Science, both received from the Federal University of Rio Grande do Sul (UFRGS), where he was also Postdoctoral Research Fellow from October 2013 to September 2014. From May 2010 to April 2011 he was a visiting researcher at the IBM T.J. Watson Research Center - Hawthorne, where he developed projects on IT Service Management and Security Management. He has worked on the organization of several international conferences, is on the TPC of many network and service management events, including IM, NOMS and CNSM, and is reviewer of important journals in the area, such as IJNM, TNSM, COMNET, and COMMAG. He also serves as Secretary of the IEEE Technical Committee on Network Operations and Management (CNOM) since 2017. His current research interests focus on design and management of Future Networks and Technologies, including aspects such as network virtualization, quality of service management, network programmability, and security management.

Eduardo J. Spinosa is a professor at the Department of Informatics of the Federal University of Paraná, Brazil. He holds a Ph.D. in Computer

Science from the University of São Paulo. His main research interests are machine learning, bio-inspired computing, artificial neural networks, deep learning, swarm intelligence, evolutionary computation, natural language processing, recommender systems, robotics, bioinformatics, novelty detection, data streams.

Elias Procopio Duarte Junior is a Full Professor at Federal University of Parana, Curitiba, Brazil, where he is the leader of the Computer Networks and Distributed Systems Lab (LaRSis). His research interests include Computer Networks and Distributed Systems, their Dependability, Management, and Algorithms. He has published more than 250 peer-reviewer papers and has supervised more than 130 students both on the graduate and undergraduate levels. Prof. Duarte is currently Associate Editor of the Computing (Springer) journal and IEEE Transactions on Dependable and Secure Computing, and has served as chair of 25 conferences and workshops in his fields of interest. He received a Ph.D. degree in Computer Science from Tokyo Institute of Technology, Japan, 1997, M.Sc. degree in Telecommunications from the Polytechnical University of Madrid, Spain, 1991, and both BSc and MSc degrees in Computer Science from Federal University of Minas Gerais, Brazil, 1987 and 1991, respectively. He chaired the Special Interest Group on Fault Tolerant Computing of the Brazilian Computing Society (2005-2007); the Graduate Program in Computer Science of UFPR (2006-2008); and the Brazilian National Laboratory on Computer Networks (2012-2016). He is a member of the Brazilian Computing Society and a Senior Member of the IEEE.