Magoula, L., Barmpounakis, S., Stavrakakis, I. & Alonistioti, N. (2021). A genetic algorithm approach for service function chain placement in 5G and beyond, virtualized edge networks. *Computer Networks, 195*, 108157.

DOI: 10.1016/j.comnet.2021.108157

A Genetic Algorithm approach for Service Function Chain Placement in 5G and Beyond, Virtualized Edge Networks

Lina Magoula[1], Sokratis Barmpounakis[1], Ioannis Stavrakakis[1] and Nancy Alonistioti[1]

[1] *Department of Informatics and Telecommunications, University of Athens, Greece*

Abstract

Network Function Virtualization (NFV) is already considered as a structural enabler of today's networking technology and particularly the 5th Generation of Broadband and Cellular Networks (5G). NFV provides the means to flexibly and dynamically manage and allocate resources, without being restricted to the hardware limitations of the network/cloud infrastructure. Resource orchestration for specific 5G vertical industries and use case families, such as Industry 4.0 and Industrial Internet of Things (IIoT), often introduce very strict requirements in terms of network performance. In such a dynamic environment, the challenge is to efficiently place directed graphs of Virtual Network Functions (VNFs), named as SFCs (Service Function Chains), to the underlying network topology and to dynamically allocate the required resources. To this end, this work presents a novel framework, which makes use of a delay and location aware Genetic Algorithm (GA)-based approach, in order to perform optimized sequential SFC placement. Evaluation results clearly demonstrate the effectiveness of the proposed framework in terms of producing solutions that approximate well the global optimal, as well as achieving low execution time due to the employed GA-based approach and the incorporation of an early stopping criterion. The performance benefits of the proposed framework are evaluated in the context of an extensive set of simulation-based scenarios, under diverse network configurations and scales.

Keywords: resource orchestration; genetic algorithm; network function virtualization

1. Introduction

The concepts of Virtualized Network Functions (VNFs) and Software-Defined Networks (SDNs) have created a breakthrough in the area of networking, especially in the development of 5G, which encompasses a deluge of novel and challenging use cases, applications and services. Each service is represented by a Service Function Chain (SFC), defined as an ordered or partially ordered set of VNFs with a set of constraints that must be applied to packets, frames, and/or flows selected as a result of classification [1]. The above-mentioned use cases, applications and services rely on all network segments, namely the access, transport and core, in order to satisfy all their requirements in an End-to-End (E2E) manner. Particularly for the transport segment of the network, Network Function Virtualization brings considerable gains, enabling the network administrator to manage computing, network and storage resources in an elastic and context-aware manner, in order to adapt to the ever-changing requirements introduced by the underlying services and applications.

Some of the most challenging use cases relate to delay-sensitive applications for remote end-users that suffer from long-distance network propagation and transmission delays. This remains a crucial challenge for resource management in cloud computing. Although 5G introduced numerous disruptive Use Cases (UCs), such as Connected and Automated Mobility (CAM), smart cities, e-health, energy, etc., one of the most challenging vertical domains is considered Industry 4.0 and its encompassed IoT (Internet of Things) concept, namely Industrial Internet of Things (IIoT). IIoT and Industry 4.0 UCs primarily pose ultra-strict requirements related to E2E service reliability and latency (even below 1ms in some cases). Very recently, 3GPP introduced four UCs in the New Radio (NR) IIoT specifications in the latest Rel.16 [2], which are identified as key drivers for NR evolution for IIoT, namely Augmented reality and Virtual reality, Factory automation, Transportation and Electrical Power Distribution. Additionally, 3GPP introduces the Time Sensitive Networking (TSN) aspects of IIoT UCs, which further pose additional requirements related to ultra-high communication service reliability

and availability, as well as clock synchronization.

In an attempt to address such challenges, the architecture of Multi-Access Edge Computing (MEC) has been lately introduced to bring several parts of the required network and computing functionality to the edge of the network, closer to the end-users and IoT devices. Those devices and User Equipments (UEs) rely to a great extent on the availability and proximity of the network and computing resources in order to meet the Quality of Service (QoS) prescribed by the various Service Level Agreements (SLAs).

Applying NFV to MEC not only drastically shortens servicing delay for end-users but also enables service providers to benefit from lower expenditures and higher efficiency. From the standardization perspective, the ETSI Industry Specification Group for VNF (ISG VNF) has already introduced 3 successive releases with complementary features and capabilities ([3–5]). Besides ETSI, 3GPP also considers how VNF and MEC architectures impact existing standards at the architecture and system levels [6]. The combination of NFV and MEC enablers promises outstanding gains in the performance of numerous challenging services, giving at the same time rise to new challenges. For the latter, the geo-distributed nature of the MEC infrastructure, as well as the complex task of efficiently assigning the available distributed computing and communication resources of distributed servers via a centralized management entity is taken into account.

Crucial challenges are thus emerged in the context of managing the network, storage and computing resources of the network, both at the edge of the network (with limited resources but closer to the edge users and devices) and the core of the network (where an abundance of resources are typically available, though over larger distances and network hops). Advanced Management & Orchestration (MANO) operations and algorithms are required, capable of exploiting the resources' availability and constraints in an intelligent and most importantly flexible manner, in conjunction with the real-time requirements from the underlying network services (SFCs). As a result, VNFs should be efficiently placed to the underlying network infrastructure, satisfying VNF's resource demands (CPU, memory, etc.). In addition, in terms of traffic steering (routing), the path to transmit traffic load through the VNFs of a SFC should be determined towards minimizing the E2E delay of each service. The combination of both aforementioned problems is known as SFC placement and it is NP-hard.

To this end, we propose in this work a novel SFC placement framework, which operates on top of the well-established Network Function Virtualization Infrastructure (NFVI) and aims at minimizing the E2E delay of all requested latency-critical services (such as IIoT). The proposed framework builds upon a Genetic Algorithm (GA), which has been extended with a number of context-awareness enhancements towards further delay minimization.

The innovations and contributions of this work are the following. The proposed SFC placement framework takes into account four delay-related metrics (queuing, processing, transmission and propagation delay), while also identifying and selecting the optimal transmission and propagation delay-aware paths for each candidate SFC placement solution. In addition, our GA-based model integrates innovative features, such as location-awareness associated with the candidate solutions (hosts), towards further minimizing the end-to-end delay of the SFC paths and, thus, augmenting the GA's capabilities. Furthermore, a GA solutions' filtering optimization method is introduced and applied, towards further reducing the GA algorithm's execution time, in cases of very sparse network resource availability, which can cause unnecessary or even non-operational time overheads. Finally, a dynamic early stopping criterion for the GA algorithm is introduced in order to achieve faster convergence and termination than the static one typically employed. All aforementioned innovations have been evaluated and results are derived showing their effectiveness.

The rest of the paper is organized as follow: Section II presents an overview of the existing work on SFC placement in dynamic networks. Section III provides the details of the system model, problem formulation and the proposed GA-based framework. Section IV presents the results of the evaluation of the proposed framework, while Section V concludes this paper and discusses the findings, next steps, as well as relevant open research questions.

2. Related work

In this section we focus on related work on SFC placement for IIoT. Then, we discuss related VNF placement works.

In [7], the authors propose a novel VNF placement solution for IIoT networks towards system optimization in terms of latency minimization and resource utilization maximization by formulating the problem as Maximum Satisfiability (MaxSMT). Experimental results appear to be promising even in large scale scenarios, where the substrate network consists of a significant number of data centers. In [8], Ruiz *et al.* provide a GA-based solution, towards addressing the VNF placement problem in

a Wavelength Division Multiplexing (WDM) metro network equipped with MEC resources, putting emphasis on node failure protection. The authors also determine the set of backup resources (VNFs and links) to protect the system against node failure. Simulation results show that among the different schemes of protection, the one that uses shared VNF and network resources between backup instances achieves the best results in terms of both service blocking ratio and number of active resources. The authors in [9] present a preliminary model that combines NFV with node and link mapping approaches from Virtual Network Embedding (VNE). The proposed model is based on different requirements, such as locations, low latency, redundancy and security functions. The authors propose a greedy proactive heuristic solution for path-independent embedding of the composed SFCs. Their model is supported by a remote asset management use case, inspired by the principle of smart factory in Industry 4.0 and based on IIoT concepts.

Apart from Industry 4.0 specific studies, several works also investigate the problem of SFC and VNF placement at the Edge (and Cloud) of the network ([10–14]). In [10] the authors propose a formulation of the VNF placement problem, tailored to URLLC, as an optimization problem of two main objectives, namely the minimization of access latency and the maximization of service availability. A scalarization approach is used in order to transform the two aforementioned conflicting objectives to a single one. A GA-based solution is proposed that according to experimental results appears to be effective. In [11] a spatio-temporal model of mobile service usage over a geographical area is proposed, along with an enhanced version of the predictive VNF placement strategy, "Advanced Predictive Placement Algorithm" (APPA). The proposed spatio-temporal model defines the behavior of mobile users in terms of mobility patterns and service consumption. In [12] an heuristic algorithm is proposed for the SFC embedding problem in 5G networks with MEC, aiming at minimizing service interruption, while maximizing user satisfaction, considering user mobility in the network. In [13] the problem of reliability-aware SFC provisioning problem in MEC is examined, aiming at maximizing the number of admitted requests, while satisfying individual's request's reliability requirement. An ILP solution is proposed for small scale problem size, while a randomized algorithm along with an heuristic approach is proposed for large scale problem sizes. Last but not least, authors in [14] propose a fast and efficient heuristic algorithm called MaxZ, targeting the problem of orchestration in 5G networks that requires making decisions about VNF placement, CPU assignement and traffic routing. The authors also present a queueing-based model accounting the main features of 5G networks. Experimental results prove the effectiveness of the proposed methodology.

Besides the edge/cloud-specific solutions, there is a plethora of scientific papers formulating the SFC placement problem as an Integer Linear Program (ILP), Mixed Integer Linear Program (MILP) or Mixed Integer Quadratic Program (MIQP). The aforementioned models consider a variety of Key Performance Indicators (KPIs) and impose resource constraints in order to meet the targeted performance [15–20]. In particular, in [15] Sun *et al.* examine an offline version of the problem formulated as an ILP model. The objective of the ILP model is the minimization of the total SFC deployment cost. An online version of the solution is proposed, in which a time-efficient heuristic algorithm is extended so as to predict future VNF demands and reduce the setup delay of the SFCs. Simulation results show that the online algorithms could efficiently reduce the blocking probability of SFC requests and effectively increase the service provider's profit. In [16] Davit Harutyunyan *et al.* compare three variants of an ILP-based algorithm that aim to minimize E2E latency of requested services, service provisioning cost, and Virtual Service Function (VSF) migration frequency, respectively, while addressing the scalability issue of the ILP solutions proposing an heuristic algorithm. In [17] Richard Cziva *et al.* employ Optimal Stopping Theory principles to determine when to re-evaluate their placement solution, considering the changes in latency of a real-world scenario and migrating the respective VNFs if necessary. In [18] Luizelli *et al.* incorporate a Variable Neighborhood Search (VNS), targeting to minimize the number of VNF instances mapped on the Physical Nodes (PNs) and the operational costs respectively, while meeting network flow requirements and constraints. The authors provide evidence that their algorithm is able to effeciently find high quality solutions even in scenarios scaling to hundreds of VNFs. Alleg *et al.* in [19] formulate the VNF placement problem as MIQP targeting to minimize resource consumption and provide specific latency while considering the relationship between the resources allocated to a VNF instance and the expected latency. The study in [20] formulates the problem as a MILP one suitable for Internet Service Provider (ISP) operations and adopt a math-heuristic resolution method.

Since the computational complexity of ILP, MILP and MIQP increases immensely with the size of a mobile network, to become practically intractable, several papers propose a variety of alternative approaches, such as novel complexity-aware heuristics, genetic algorithms and decision tree learning [21–23].

In addition, many recent research works address the SFC placement problem using Deep Reinforcement Learning (DRL) models due to their efficiency and applicability in many complex environments [24–27]. In particular, the authors in [24] formulate the problem as a Binary Integer Programming (BIP) problem and determine the optimal solution from a prohibitively large solution space by using DRL techniques. In [25] the authors propose a parallel deployment scheme based on DRL, satisfying demands with minimum resources. The authors design an overall SFC placement scheme for all demands, and deploy all SFCs simultaneously. They evaluate the proposed algorithms using extensive simulations and experiments. Last but not least, both [26, 27] apply DRL techniques to derive feasible solutions while improving significantly the exploration of the action space.

To the best of our knowledge most of the existing efforts take into account only the processing and transmission delay metrics. There are several works considering four delay-related metrics (queuing, processing, transmission and propagation delay) aiming to provide a delay-aware SFC placement solution using commercial solvers, heuristic and meta-heuristic approaches [28–30]. However our work proposes a GA-based approach (known for its ability to deal with complex problems and parallelism [31]) considering all four aforementioned delay metrics, towards providing (sub)optimal solution to the delay-aware SFC placement optimization problem avoiding stagnating in local optima. In our problem formulation, we also consider selecting the optimal transmission and propagation delay-aware path for each SFC in order to minimize the total path's link delay. Furthermore, our GA-based model integrates further optimizations, such as location-aware augmented capabilities, towards further minimizing the end-to-end SFC paths' aggregated delay, as well as GA solutions' filtering, towards minimizing the GA algorithm's execution time (in cases of very sparse network resource availability, which can cause unnecessary or even non-operational time overheads). Finally, a dynamic early stopping criterion is imposed to GA model so as to enhance the static one, used in the classic version of GA, towards achieving faster convergence.

3. The proposed framework

In this section, the SFC placement problem is formulated as a Mixed Integer Linear Programming (MILP) optimization problem and an innovative GA meta-heuristic approach, tailored to our problem formulation, is proposed.

3.1. System Model

First, the physical and virtual network topology and SFC model considered in our problem formulation are introduced, along with related notations.

3.1.1. Network Model

The network physical infrastructure is modeled as an undirected graph $G_p(P, E)$ of Physical Nodes (PNs), where P is the set of PNs and E the set of physical links, assumed to be bidirectional. Each PN hosts a subset of Virtual Machines (VMs), from the set of total VMs V, based on its available resource capacity. It is assumed than the computational resources associated with a PN or a VM are defined in terms of the quadruple (number of CPU cores, total CPU speed, Memory size, Disk size). Accordingly, the resource capacity of a PN or a VM is denoted by $\{C^*(t)\}_{t\in T}$, where $t \in T = \{$ *number of CPU cores*, *total CPU speed*, *Memory size*, *Disk size*$\}$ and * refers to a PN p, $p \in P$, or a VM v, $v \in V$. Let $I(v, p)$ denote a binary variable indicating whether PN $p \in P$, hosts the VM $v \in V$. Each physical link $(p1, p2) \in E$, connecting neighbour PNs $p1$ and $p2$, is characterized by a bandwidth capacity $b^{(p_1,p_2)}$ and a propagation delay $D_{pr}^{(p_1,p_2)}$, which depends on transmission medium and the length of the physical link. Let $D_t^{(p_1,p_2)}$ denote the transmission delay, defining the amount of time required to transmit a packet into the physical link $(p1, p2) \in E$ and it is inversely proportional to bandwidth capacity of the physical link $b^{(p_1,p_2)}$. Table 1 summarizes the network model notations.

3.1.2. SFC Model

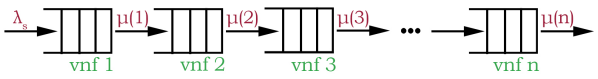


Figure 1: VNFs sequentially connected through virtual links, known as SFC

A service could be represented as a directed graph of VNFs, which is known as SFC, sequentially connected through virtual links (Figure 1). Let S denote the set of all SFCs ready to be placed in the underlying network infrastructure and let F denote the associated VNFs. Each SFC is modeled as a graph $G_s(N_s, E_s)$, where N_s is the set of VNFs of SFC $s \in S$ and E_s is the set of directed edges virtually connecting the VNFs in N_s. Each VNF instance $f \in N_s$ has a certain amount of requested computing resources in $T = \{$ *number of CPU cores*, *total*

Parameter	Description
G_p	Graph of Physical Infrastructure
P	Set of PNs in G_p
E	Set of bidirectional physical links interconnecting the PNs
V	Set of VMs
T	Set of all resource types
$\{C^*(t)\}_{t\in T}$	Resource capacity of PN $p \in P$ or VM $v \in V$ for resource type $t \in T$
$I(v, p)$	Binary variable indicating whether PN $p \in P$, hosts the VM $v \in V$
$b^{(p_1,p_2)}$	Bandwidth capacity of physical link between PNs $p_1, p_2 \in P$
$D_{pr}^{(p_1,p_2)}$	Propagation delay of physical link between PNs $p_1, p_2 \in P$
$D_t^{(p_1,p_2)}$	Transmission delay of physical link between PNs $p_1, p_2 \in P$

Table 1: Network Model Notation

$$min(\sum_{f\in N_s}\sum_{v\in V}((D_q^f + D_{pc}^f) \times I(f, v)) + \sum_{(f_1,f_2)\in E_s}\sum_{(p_1,p_2)\in E}((D_t^{(p_1,p_2)} + D_{pr}^{(p_1,p_2)}) \times I((p_1, p_2); (f_1, f_2)))) \quad (1)$$

where:

$$D_q^f = \frac{\lambda_s(f)}{\mu(f) \cdot (\mu(f) - \lambda_s(f))} \quad (2)$$

$$D_{pc}^f = \frac{1}{\mu(f)} \quad (3)$$

$$D_t^{(p_1,p_2)} = \frac{L}{b^{(p_1,p_2)}} \quad (4)$$

$$D_{pr}^{(p_1,p_2)} = \beta \times l^{(p_1,p_2)} \quad (5)$$

CPU speed, *Memory size*, *Disk size*} notated as $\{R_f^s(t)\}_{t\in T}$. Each virtual link connecting two VNFs $f_1, f_2 \in N_s$ is characterized by a specific requested bandwidth $b_s^{(f_1,f_2)}$. Let $I(f, v)$ denote a binary variable indicating whether VM $v \in V$, hosts the VNF $f \in F$. Let $I(p_1, p_2; f_1, f_2)$ denote a binary variable indicating whether the physical link $(p_1, p_2) \in E$ is part of the physical path between the PNs hosting VNFs f_1 and f_2.

The packet arrival process to each VNF of SFC s, $s \in S$, is assumed to be Poisson ([32]) with rate $\lambda_s(f)$, with an average packet size of L, fixed for all VNFs in N_s. According to the aforementioned assumption and in order to estimate the queuing and processing delay of each VNF of a SFC s, each VNF $f \in N_s$ is modeled as an $M/M/1$ queue, assuming its service time has an exponential distribution. Each VNF f, $f \in F$, is characterized by a service rate $\mu(f)$, in packets per second, proportional to the allocated resource capacity. Rate $\mu(f)$ highly affects the mean processing delay D_{pc}^f (service time) of the VNF $f \in F$, which is measured per packet, since the latter is inversely proportional to the service rate. Let D_q^f denote the time each packet spends in the $M/M/1$ queue of VNF $f \in N_s$ (queuing delay). Let D_{thr}^s denote the maximum E2E delay (threshold) imposed by the service implemented through SFC s, $s \in S$; the latter is defined as the sum of all aforementioned delay components (processing, queuing, transmission, propagation). Table 2 summarizes the SFC model notation.

3.2. Problem Formulation

In the current work only the sequential SFC placement is examined, in which case the SFCs are placed on the underlying network topology one-by-one, not returning again to modify an earlier SFC placement. Clearly, such a placement approach may result in (sub)optimal solutions due to the sequential placement constraint. The SFC sequential placement problem is formulated as an MILP optimization problem. The goal of the optimization model is to provide a (sub)optimal placement solution towards minimizing the E2E delay of the SFC. Each

Parameter	Description
S	Set of SFCs
F	Set of VNFs modeled as $M/M/1$ queues
$\mu(f)$	Processing rate of VNF $f \in F$
$\lambda_s(f)$	Packet arrival rate at VNF $f \in N_s$ of SFC $s \in S$
L	Average Packet Length
G_s	Directed Graph of VNFs of SFC $s \in S$
N_s	Set of VNFs of SFC $s \in S$
E_s	Set of of virtual links of SFC $s \in S$
R_f^s	Requested computing resources of VNF $f \in N_s$ of SFC $s \in S$
$b_s^{(f_1,f_2)}$	Requested bandwidth capacity of virtual link between VNFs $f_1, f_2 \in N_s$ of SFC $s \in S$
$I(f,v)$	Binary variable indicating whether VM $v \in V$, hosts the VNF $f \in F$
$I(p_1,p_2;f_1,f_2)$	Binary variable indicating whether the physical link $(p_1,p_2) \in E$ is part of the physical path between the PNs hosting VNFs f_1 and f_2
D_{pc}^f	Processing delay of VNF $f \in N_s$
D_q^f	Queuing delay of VNF $f \in N_s$
D_{thr}^s	E2E delay threshold imposed to SFC $s \in S$

Table 2: SFC Model Notation

$$\sum_{f \in F} \sum_{v \in V} (R_f^s(t) \times I(f,v)) \leq C^v(t), \forall s \in S, \forall t \in T \tag{6}$$

$$\sum_{(f_1,f_2) \in E_s} \sum_{(p_1,p_2) \in E} (b_s^{(f_1,f_2)} \times I((p_1,p_2);(f_1,f_2))) \leq b^{(p_1,p_2)}, \forall s \in S \tag{7}$$

$$\sum_{f \in F} \sum_{v \in V} (I(f,v)) = 1 \tag{8}$$

$$\sum_{f \in N_s} \sum_{v \in V} ((D_q^f + D_{pc}^f) \times I(f,v)) + \sum_{(f_1,f_2) \in E_s} \sum_{(p_1,p_2) \in E} ((D_t^{(p_1,p_2)} + D_{pr}^{(p_1,p_2)}) \times I((p_1,p_2);(f_1,f_2))) \leq D_{thr}^s \tag{9}$$

VNF can be placed on a VM that satisfies the required resource demands. Each virtual link between a pair of VNFs f_1, f_2, denoted as $(f_1,f_2) \in E_s$, can be mapped to a physical path that satisfies its bandwidth requirements $b_s^{(f_1,f_2)}$. A constraint is imposed on the estimated total E2E delay (through the adopted models) of each service to ensure that is upper bounded by the service delay threshold D_{thr}^s provided by the Network Service Descriptor (NSD), to guarantee conformance to the SLA. For each candidate SFC placement solution, if a pair of VNFs is hosted by different PNs and there are multiple paths between the latter, a weight w is assigned to each physical link across each candidate path that is inversely proportional to link delay (incl. transmission and propagation delay) as shown in function (10).

$$w = \frac{1}{\alpha \times D_t^{(p_1,p_2)} + (1-\alpha) \times D_{pr}^{(p_1,p_2)}} \tag{10}$$

where $(p_1,p_2) \in E$ and $\alpha \in (0,1)$ is a factor modulating the influence to w of the propagation and transmission delay, this influence being equal if $\alpha = 0.5$. In case the transmission delay $D_t^{(p_1,p_2)}$ is negligible, its contribution to w could be ignored completely by setting α equal to 0 and as a result the weight is inversely proportional to the propagation delay $D_{pr}^{(p_1,p_2)}$. In a similar manner, in case $\alpha = 1$, the weight is inversely proportional to the

transmission delay. The path with the maximum sum of the respective weights is chosen and is referred to as delay-aware path. For each physical link $(p1, p2) \in E$ belonging to the selected delay-aware path, constraint (7) is checked. A delay-aware, instead of a random, path selection for each SFC contributes further to the E2E delay minimization by decreasing the total link delay (transmission, propagation delay) of the selected physical path.

The objective function of our optimization problem for each SFC $s \in S$ is the minimization of the service E2E delay and it is formulated and given in (1). $D_{pr}^{(p_1,p_2)}$ is predefined and fixed, since it depends on the transmission medium (β) and the link length $l^{(p_1,p_2)}$.

The first part of the equation (1) expresses the sum of queuing and processing delay for each VNF f, $f \in N_s$, hosted by VM $v \in V$. The second part of the equation expresses the sum of transmission and propagation delays of each physical link $(p_1, p_2) \in E$ that is included in the physical delay-aware path between the PNs p_1, p_2 hosting the VNFs f_1, f_2. The objective function is subject to constraints (6-9).

Constraint (6) ensures that the resources (for all resource types $t \in T$) requested by all VNFs mapped to VM $v \in V$ will not exceed the VM's corresponding resource capacity. Constraint (7) ensures that the bandwidth required by all the virtual links $(f_1, f_2) \in E_s$ that include physical link $(p_1, p_2) \in E$, will not exceed that physical link's capacity. Constraint (8) ensures that each VNF $f \in F$ will be assigned to one and only one VM and thereby all VNF's requested resources will be satisfied by that VM. Finally, constraint (9) guarantees that the estimated total E2E delay of each SFC will not exceed the service delay threshold D_{thr}^{s} imposed by the NSD.

3.3. The proposed GA-based SFC Placement

Since the SFC placement problem is NP-hard, the complexity of the MILP solution would be prohibitively high under any realistic scenario. The latter applies also to the relatively easier sequential SFC placement problem. Consequently, heuristic approaches would be very much needed to be able to yield solutions to realistic use cases. In this section a GA-based meta-heuristic approach tailored to our problem formulation is proposed. A meta-heuristic-based approach is selected instead of an heuristic-based one, since the latter has typically a narrower search scope and it is more prone to stagnation in local optima.

As it will be described in the last part of this section, our GA-based model integrates two additional features for further optimization potential. The first one prescribes a location-aware host selection towards further reducing the E2E SFC paths' aggregated delay. The second one carries out some filtering of the GA-based solutions towards further reducing the GA algorithm's execution time.

3.3.1. Overview of Genetic Algorithms

Genetic Algorithms provide a feasible solution to strategically perform a global search by means of many local searches, ensuring generating high-quality solutions relying on biologically inspired operators, such as selection, crossover and mutation. A GA in each generation, constructs a population of *chromosomes*, which is a set of candidate solutions to the optimization problem. The target of the GA is to provide higher quality solutions over the generations using as criterion the fitness value of each chromosome, which represents the target metric of the optimization problem.

In each generation, the chromosomes of the current population are sorted in an increasing order based on their fitness value and the best-scored ones are selected as parents for crossover operation. The crossover is accomplished by randomly exchanging the genes of the two parent-chromosomes based on a predefined probability and create next-generation chromosomes. After the crossover, each one of the genes of the next-generation chromosomes will be altered based on a predefined probability in order to ensure diversity on the set of candidate solutions. The aforementioned operation is known as mutation. After the crossover and mutation operations are completed, the next-generation chromosomes are once again evaluated based on problem constraints for another round of selection and reproduction (generation). The loop of chromosome generation is terminated when either the maximum number of generations has been reached or the best-scored chromosome has not changed for a certain number of generations.

3.3.2. Encoding Scheme

In the proposed GA-based optimization model the (sub)optimal placement solution is generated for each SFC of the set S sequentially. A placement solution is represented - and fully identified - by a chromosome (candidate SFC placement solution). A chromosome is represented by a vector consisting of as many genes as the number of VNFs $|N_s|$ of the SFC $s \in S$ that are ready to be placed on the underlying network topology. The position of a gene j ($j \in [1, |N_s|]$) within a chromosome depicts the respective VNF, i, of SFC s, $s \in S$, represented by this chromosome. The value of the j^{th} gene of a chromosome depicts the VM that hosts the j^{th} VNF of SFC s represented by this chromosome; the range of a gene's value is $[1, |V|]$. At each generation of the GA

GA Term	Description in the context of the SFC placement optimization problem
generations	Total number of iterations, searching for the optimal SFC placement solution
fitness function	The objective function (Eq.1) of the SFC placement problem
chromosome	A candidate SFC placement solution
gene	A candidate VNF placement solution
population	Total number of candidate SFC placement solutions in each iteration
elites	The best-scored (based on the objective function Eq.1) SFC placement solutions in each iteration

Table 3: GA Terminology

the set of all chromosomes is referred to as the population of chromosomes (or candidate placement solutions) and its size is fixed and equal to *population size*. Table 3 summarizes GA terminology that will be used in the next sections, in the context of our SFC placement optimization problem.

3.3.3. GA Operations

In the proposed GA-based solution four operations (selection, crossover, mutation, mutagenesis) are applied sequentially in order to generate new chromosomes (offsprings) so as to produce the population of the next generation. The quality of each chromosome is characterized by a fitness value indicating how 'fit' a candidate solution is with respect to the problem objective. The fitness function (*FF*) considered in the current work is equal to the E2E delay of a SFC, the minimization of which is the target of our sequential placement problem. As a result, the fitness function of a SFC s, $s \in S$, is defined in Equation (11).

In each generation, parent chromosomes are selected for crossover in order to create offsprings for the next generations. Each offspring inherits its genes from the two selected parents. This operation is crucial to GA's convergence since fit parents tend to produce better and fitter solutions. The method considered for parent selection in the proposed GA-based solution is the Roulette Wheel ([33]). In a roulette wheel selection, a circular wheel is divided in n pies, where n is equal to the size of the population. The size of each pie assigned to each chromosome is based on its fitness value fv_i. This could be achieved by dividing the fitness value of each chromosome by the total fitness value of all chromosomes of the population, thereby normalizing them to 1. Since the objective of the optimization problem is the minimization of the fitness function, the selection probability pb_i assigned to each chromosome i is complementary to the respective size of pie and is defined as follows:

$$pb_i = 1 - \frac{fv_i}{\sum_{j=1}^{n} fv_j} \tag{12}$$

Equation (12) states that chromosomes with a lower fitness value will be less likely to be eliminated over the generations.

In order to select the first parent, all chromosomes are sorted in an increasing order based on their fitness value. A uniform random number $r \in (0, 1]$ is generated and based on that chromosome i is selected as first parent if it satisfies the condition $pb_i \geq r \geq pb_{i+1}$. This procedure is repeated for the second parent.

After selecting the two parents, a crossover operation will be applied in order to combine genetic information of the two selected parents to generate two new offsprings. A uniform crossover method is selected under this paper work. For each gene of the first offspring, a number $u \in [0, 1]$ is uniformly generated in order to decide from which parent it will inherit the gene based on the following condition.

$$\begin{cases} g[i] \leftarrow pr_1[i], & \text{if } u \geq cr\_rate \\ g[i] \leftarrow pr_2[i], & \text{otherwise} \end{cases} \tag{13}$$

In condition (13), $g[i]$ represents the i^{th} position of the offspring chromosome, the $pr_j[i]$ where $j \in \{1, 2\}$ is the i^{th} position of the parent chromosome j and the $cr\_rate \in [0, 1]$ is named as *crossover rate* and it denotes the selected threshold. Each gene of the second offspring is inherited by the opposite parent chosen for the respective gene of first offspring. *Figure* 2 illustrates the uniform crossover, where typically, each gene of the first offspring is chosen from either parent with *cr_rate* probability while each gene of the second offspring is chosen from the opposite parent selected for the respective gene of the first offspring.

After generating all new offsprings and before adding them to the new generation, another valuable operator

$$FF = \sum_{f \in N_s} \sum_{v \in V} ((D_q^f + D_{pc}^f) \times I(f, v)) + \sum_{(f_1, f_2) \in E_s} \sum_{(p_1, p_2) \in E} ((D_t^{(p_1, p_2)} + D_{pr}^{(p_1, p_2)}) \times I((p_1, p_2); (f_1, f_2))) \quad (11)$$

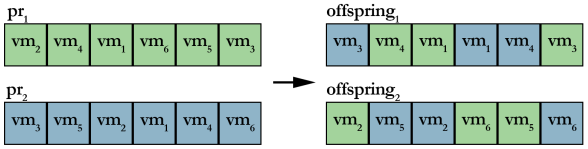


Figure 2: Uniform Crossover Operation

to GA's convergence is applied to each offspring, named mutation. Mutation is the part of the GA which is related to the exploration of the search space. Mutation is applied to randomly tweak genes of a chromosome in order to get a new candidate solution (mutated chromosome). The operation is used to maintain and introduce diversity in the genetic population by altering each gene of a chromosome with a probability *mu_rate* called *mutation rate*. Mutation is usually applied with a low probability, since higher values can alter a significant amount of genes of a chromosome and as a result can potentially lead the GA to a more random search. Under mutation, for each gene of each chromosome of the population a number $u \in [0, 1]$ is uniformly generated and compared against the mutation rate, in order to decide if the gene will be mutated or not. In case mutation is selected for a gene, its value will be altered by a randomly selected VM v, $v \in V$, as shown below.

$$\begin{cases} g_m[i] \leftarrow v, & \text{if } u \leq mu\_rate \\ g_m[i] \leftarrow g[i], & \text{otherwise} \end{cases} \quad (14)$$

In condition (14), $g_m[i]$ represents the i^{th} position of the chromosome which is candidate for mutation.

After mutation, a mutagenesis operation is applied as a survivor selection policy as described in the paper work of *Hedar et al.* in [34]. Mutagenesis provides the two worst-scored chromosomes with a second chance to randomly inherit one gene from one of the two highest-scored chromosomes (as parents) in the population. In essence mutagenesis is a crossover operation limited to one gene. This operation is crucial as it provides the possibility to produce fitter offsprings, while at the same time maintaining diversity in the population.

3.3.4. Optimization Methods on the proposed GA-based SFC Placement

As mentioned in the beginning of this section, the proposed GA-based model integrates two further optimization methods, with the first one being a location-aware host selection and the second one a GA's solution filtering, towards further reducing the E2E SFC paths' aggregated delay and GA algorithm's execution time respectively.

As described in Section 3.2, for each candidate SFC placement solution, a delay-aware path, instead of a random, is selected in order to further contribute to the E2E delay minimization. In many cases, due to the stochastic nature of the GA, there is a possibility of selecting hosts for the VNFs of a SFC that might be distant, in terms of network hops. In order to handle the latter, a location-aware optimization method is proposed, applying limits in terms of network hops (location-aware) between the selected hosts of a SFC. The location-aware optimization method is applied to the set of candidate virtual hosts V. More specifically, assuming that the proposed GA framework will be deployed in a MEC infrastructure, a random VM $v \in V$, that satisfies constraints ((6)-(9)), is selected as a host of the first VNF of a SFC. In case the proposed GA framework will be deployed in a more complex network topology, including multiple interconnected infrastructures, then the selection of a VM $v \in V$ as a host of the first VNF of a SFC will be filtered based on vertical's geolocation. After the host selection for the first VNF of a SFC, the initial set of VMs V (candidate hosts) is filtered having as criterion a maximum predefined number of hops $nhops$ from the first selected host. As a result, a new subset of physically connected neighbour hosts is produced, in order to reduce path's propagation delay and to further contribute to the minimization of the E2E delay. In case that hosts satisfying constraints ((6)-(9)) are not found within the range of the maximum number of hops set, the latter number of hops $nhops$ is increased by one and the aforementioned process is repeated.

The second optimization method is applied in case of an overloaded network topology. More precisely, when the ratio between the total available and total initial resource capacity is less than or equal to 10% -due to the already reserved resources from the placed VNFs- all overloaded VMs are excluded from the initial set of candidate hosts in order to avoid unnecessary or even non-operational time overheads.

3.3.5. The proposed Algorithm

For each SFC, an initial population of candidate VNF placement solutions (chromosomes) satisfying the im-

posed constraints (6)-(9), is randomly generated (*Alg.1, ln.5*). For each chromosome of the population a delay-aware path is selected and a location-aware method is applied aiming at further minimizing the E2E delay of a SFC as described in Section 3.2 (*Alg.1, ln.10-12*). The fitness value is calculated for each chromosome of the population (Equation 11) and is used to sort the population in an increasing order (*Alg.1, ln.12-14*). The two fittest chromosomes (namely the ones with the lowest fitness values), which are known as *elites*, are instantly included in the new population for the next generation so as to ensure that they will not be altered by any of the GA operations (*Alg.1, ln.15*).

Additionally, for each chromosome of the current population a selection operation, described in Section 3.3.3, is applied in order to choose two parents for crossover. For each pair of parents two new offsprings are produced after the application of the crossover operation described in Section 3.3.3 and all constraints are checked to ensure they are satisfied. If not, the crossover operation is repeated for the same pair of parents until producing offsprings satisfying the constraints. To avoid infinite loops, a termination criterion is applied after a predefined number of loops l and the pair of parents are added to the new population as offsprings (*Alg.1, ln.16-18 & Alg.2, ln.1-9*).

After the crossover operation, the set of all new offsprings are mutated in order to ensure population diversity and a mutagenesis operation (*Alg.4*) is applied to the two worst-fit chromosomes as described in Section 3.3.3. To avoid infinite loops, a termination criterion is applied to both operations after a predefined number of loops l. If the maximum number of loops has been reached during the mutation operation, the chromosome is added to the new population unaltered (no mutation) (*Alg.3, ln.1-9*). If the maximum number of loops has been reached during the mutagenesis operation, the two worst-fit chromosomes are added to the new population unaltered (no mutagenesis). Otherwise, the two updated worst-fit chromosomes are added to new population (*Alg.4, ln.5-8*). To enhance potentially slow convergence under overloaded conditions, a filtering method is applied that excludes all overloaded VMs from the initial set.

The aforementioned process is repeated for the next generation until convergence to the (sub)optimal delay-aware SFC placement solution. The loop of generations is terminated when either the maximum number of generations has been reached or the best-scored chromosome has not changed for a certain number of generations, denoted by *initial threshold*; the latter is typically chosen to be a small fraction of the total number of generations (15).

$$initial\ threshold = \frac{N_g}{k} \tag{15}$$

where N_g denotes the total number of generations and $k \in [1, N_g]$ is a fixed and predefined variable. In most of the GA-based scientific papers the abovementioned initial threshold (15) is chosen. An obvious observation is that the GA framework is decelerated under higher values of generations leading to higher initial thresholds (see (15)). In order to avoid such potential deceleration, an adaptive threshold *thr* is introduced, whose value is decreased by a step size *st* in each generation that does not change the fitness value for the best-scored chromosome (*Alg.5, ln.1-3*). If *thr* decreased to zero then the current best-scored chromosome is selected as final solution (*Alg.5, ln.4-6*). The step size is defined as:

$$st = initial\ threshold \times pc \tag{16}$$

where *pc* is a percentage value, which controls the rate at which *thr* is updated (reduced) at every iteration. Higher values for *pc* result in a premature convergence and as a consequence lower values are mostly recommended. If GA produces a new best-scored chromosome, then the threshold is re-initialized to its initial value (*initial threshold*) and the process is repeated (*Alg.5, ln.8*). Algorithm (1) summarizes the abovementioned procedure implementing the delay-aware sequential SFC placement.

4. Evaluation

In this section, the performance of the proposed GA-based framework for SFC placement is presented, analysed and evaluated thoroughly. An extended set of diverse experiments have been conducted, in order to examine the quality of solutions produced by the proposed algorithm, under different network set ups and algorithm configurations. It should be noted that the experiment configurations were selected to simulate a range of realistic environments, such as Industry 4.0, in terms of the network topology, number of physical hosts, as well as resource requirements and availability. The proposed GA framework has been developed using Python 3.0 on Jupyter Framework. The experiments were conducted on an Intel(R) Core(TM) i7-6800K CPU of 3.4 GHz speed with 8 GB of RAM. As a first step, we evaluate and fine-tune the proposed framework in terms of the GA algorithm population size that will be selected for the GA configuration. Table 4 includes all GA's configuration parameters. In this round of evaluation, 3 SFCs

Algorithm 1: Delay-aware SFC placement

Input: Virtual, Physical Network topology; SFC; population size; no. of generations, elites, k,l, pc
Output: Delay-aware SFC placement
create network topology graph $G_p(P, E)$
create SFC topology graph $G_s(N_s, E_s)$
randomly initialize population P with size equal to *population size*;
initial threshold = $\frac{no.of generations}{k}$
thr = *initial threshold*
st = *initial threshold* $\times$ *pc*
while *generation* $\leq$ *no.of generations* **do**
    foreach *chromosome chr in population P* **do**
        find delay-aware path (10) with location-aware host selection enabled
        calculate fitness value (11)
    end
    sort population P in an increasing order
    select *elites* chromosomes and add to new population P_{new}
    foreach *chromosome chr in population P* **do**
        /*Parent Selection Operation*/
        select two parents using **Roulette Wheel method**
        /*Crossover Operation*/
        $P_{new} \leftarrow$ **Crossover(l, parents)** [Alg. 2]
    end
    foreach *chromosome chr in population P* **do**
        /*Mutation Operation*/
        $P_{new} \leftarrow$ **Mutation(l, chr)** [Alg. 3]
    end
    Mutagenesis(P_{new}, l) [Alg. 4]
    Early Stopping(P_{new}, thr, st, l) [Alg. 5]
end

Algorithm 2: Crossover

while *no. crossover attempts* < *l* **do**
    apply *uniform* crossover to selected parents
    check constraints (6) - (9) for the two new offsprings
    if *offspring solutions are valid* **then**
        add offsprings to P_{new}
        break
    no. crossover attempts+=1
if *no. crossover attempts* = *l* **then**
    add parents to P_{new}
return P_{new}

Algorithm 3: Mutation

while *no. mutation attempts* < *l* **do**
    apply mutation to *chr*
    check constraints (6) - (9) for the mutated *chr*
    if *mutated chr is valid* **then**
        add mutated *chr* to P_{new}
        break
    no. mutation attempts+=1
if *no. mutation attempts* = *l* **then**
    add *chr* to P_{new}
return P_{new}

Algorithm 4: Mutagenesis

sort population P_{new} in increasing order
while *no. mutagenesis attempts* < *l* **do**
    apply mutagenesis to two worst-fit chromosomes
    check constraints (6) - (9)
    if *two worst-fit chromosomes are valid* **then**
        update the two worst-fit chromosomes
        add to P_{new}
        break
    no. mutagenesis attempts+=1

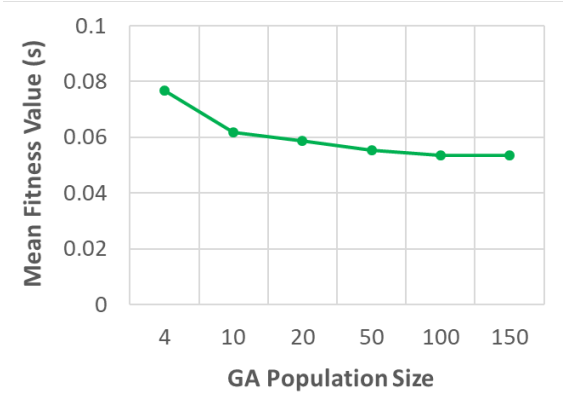
Algorithm 5: Early Stopping

sort population P_{new} in increasing order
if *best score chromosome same with previous generation* **then**
    decrease thr by st
    if *thr is equal to zero* **then**
        select fitness value of best-score chromosome as final solution
        break
else
    initialize thr to *initial threshold*

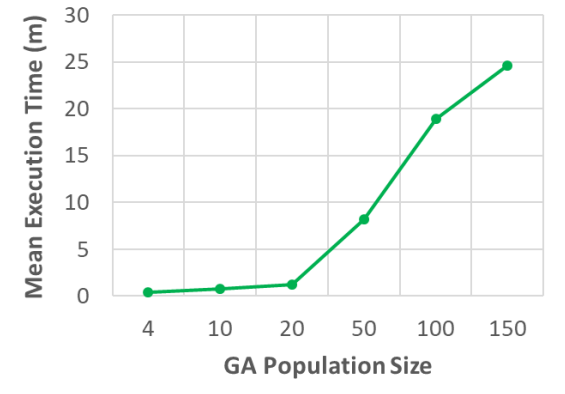
GA Parameter	Configuration
population size	50
generations	200
initial threshold	5
elites	2
crossover rate	0.5
mutation rate	0.2
mutagenesis rate	0.2

Table 4: GA Configuration

were deployed, each one comprising 5 VNFs. The network topology comprised 20 physical hosts, each one equipped with 2 VMs, 35 physical links and 8 network L2/3 switches. For each round of results, the Mean Fitness Value (MFV) and Mean Execution Time (MET) metrics are derived, over 15 different experiment runs per experiment section.



(a)



(b)

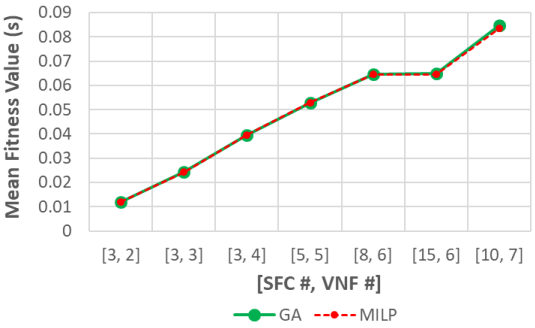
Figure 3: **(a)** Mean Fitness Value and **(b)** Execution Time, for different GA population sizes

As it is illustrated in Figures 3a and 3b, both the quality of the fitness value increases (MFV decreases) and the execution time length increases with the GA population size. Consequently, there is a clear trade off between a lower end-to-end delay and a fast GA execution time.

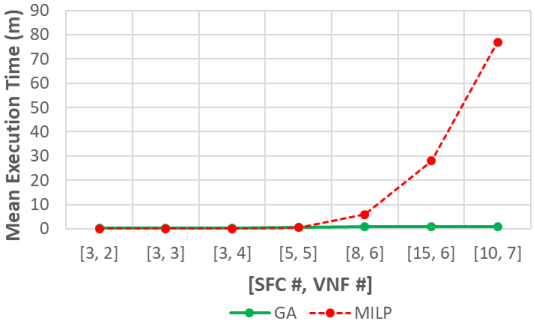
At this point, it is important to note, that this trade-off can be dynamically exploited according to the specific use case that the algorithm is applied to. In the presence of very dynamic environments and network services, a lower population size should be chosen in order to obtain an SFC placement solution fast and avoid service interruptions. For example, this is the case for population sizes up to 20 chromosomes, as indicated by Figure 3b. On the other hand, in less dynamic environments where a higher GA execution time is not an issue, a higher quality fitness value solution is possible and can be sought. Such a less dynamic environment (with the majority of devices and network services being pre-fixed) would be an Industry 4.0 scenario with pre-installed machinery and equipment. For the rest of the evaluation section and since less dynamic environments are examined, the population size is configured to 50 (instead of 20) in order achieve a solution of higher quality, taking into consideration that GA's execution time is not of critical importance (see Figure 3).

Next, the quality of the solution produced by the proposed algorithm is compared against the optimal, as provided by the MILP model (Section 3.2) under a brute force search approach. This comparison considers an increasing number of required SFCs and VNFs per SFC (Figures 4a and 4b). For almost all scales considered, the solution of the proposed algorithm coincides with the optimal one (MILP) (Figure 4a); Even in the last case (highest SFC and VNF scale equal to [10,7]), GA's solution is almost identical to the optimal (produced under the MILP using brute force search approach). GA's configuration (number of generations, population size etc.) along with the stochastic nature of algorithm, which is the reason not contained by local optima, justify the quality of GA's solution (identical to global optima). As shown in figure 4b, the proposed GA model presents low time complexity, for all scales considered, which is clearly not the case for the MILP model using brute force. This low complexity (observed to increase approximately linearly with the scale) indicates that the proposed algorithm can be applicable to much larger environments.

As a next step, the proposed scheme is juxtaposed with the MILP model under increasing scale of the topology, in terms of number of physical hosts (Figures 5a and 5b). Both present an identical performance in almost all topology scales selected, in terms of MFV. The proposed algorithm produces a solution almost equal to the

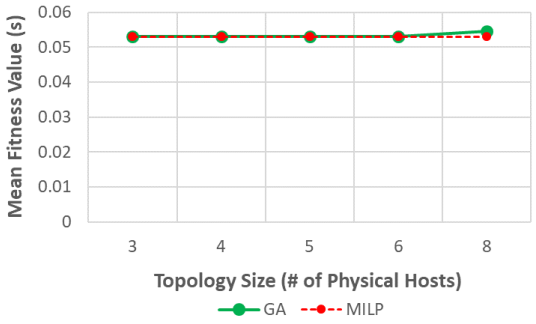


(a)

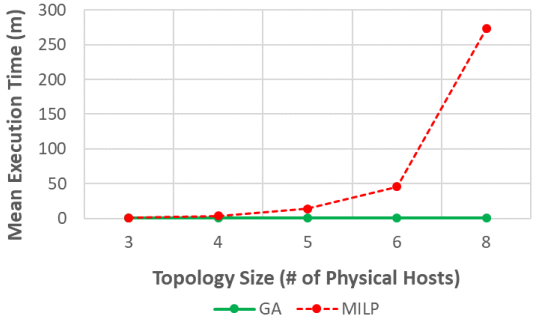


(b)

Figure 4: GA and MILP comparison in terms of **(a)** Mean Fitness Value and **(b)** Mean Execution Time, for different numbers of deployed SFCs and VNFs per SFC, over 15 experiment runs.



(a)



(b)

Figure 5: GA and MILP comparison in terms of **(a)** Mean Fitness Value and **(b)** Mean Execution Time, for different topology sizes in terms of number of available Physical Hosts.

optimal solution (produced under the MILP model) even when the search space increases (higher topology scales). Under the proposed algorithm, the observed MET is comparable to that under the smaller search space (lower topology scales), which is clearly not the case under the MILP approach where the observed MET seems to increase exponentially with the scale.

In the final part of our evaluation, the proposed scheme is evaluated when the introduced location-aware optimization method is applied. This method aims at prioritizing the "closer" (in terms of network hops) physical host choices when deploying a SFC and as a result avoiding unnecessary candidate solutions that may lead to higher METs; this method is likely to decrease MFV and it may or may not decrease MET.

Figures (6a - 6b, 7a - 7b) illustrate the effectiveness of the location-aware method of the proposed scheme. The solutions developed with and without (location-agnostic) the application of the location-aware method are compared for different *Utilization Ratios*. The latter can be shaped either by keeping the resource demands of each SFC fixed and increasing their number, or keeping the number of SFCs fixed and increasing their resource demands. It is expected that the higher the *Utilization Ratio*, the more complicated it becomes for the algorithm (i.e., higher MET) to locate the appropriate resources. The *Utilization Ratio* for each resource type $t \in T$ is defined as follows:

$$\frac{|S| \times |N| \times r(t)}{\sum_{v \in V} C^v(t)} \tag{17}$$

where $|S|$ is the number of SFCs, $|N|$ is the number of VNFs per SFC and $r(t)$ denotes the resource request for each resource type $t \in T$, applied to each VNF $f \in F$.

Assuming that all SFCs have identical resource requirements ($r(t)$) - as well as fixed number of VNFs - and given a fixed *Utilization Ratio*, the number of SFCs is equal to the maximum value produced by equation (17) for all resource types. As a result, when the resource demands of each SFC remain fixed while the number of SFCs increases, then r(t) of each involved VNF remains fixed while $|S|$ and $|N|$ increase in the Utilization Ratio. On the other hand, when the number of SFCs (and their VNFs respectively) remains fixed while the resource demands per SFC increase, then $|S|$ and $|N|$ remain constant while r(t) of each VNF increases in the Utilization Ratio.

For the first of the aforementioned cases (increasing number of SFCs and fixed resource demands) results are shown in Figures 6a and 6b. here is a clear decrease (of about 30%) in the MFV generated by the location-aware scheme (Figure 6a), in comparison with the location-agnostic scheme, as expected. However, MET under the location-aware scheme is gradually increasing with the number of SFCs, particularly for Utilization Ratios above 80% (Figure 6b). The reason why such behavior is observed relies on the fact that as the number of SFCs increases and the number of non-overloaded neighbour hosts - in terms of resource capacity - decreases, location-aware scheme requires higher execution time in order to find neighbor hosts that satisfy constraints (6) - (9).

For the second of the aforementioned cases (increasing resource demands and fixed number of SFCs) results are shown in Figures 7a and 7b. Although both the MFV (Figure 7a) and the MET (Figure 7b) metrics are improved, the most significant gains (ranging between 50-66%) are observed for the MET (Figure 7b). In terms of the MFV, the -almost negligible- improvement of 0,1% suggests that in non-dynamic network environments with fixed number of SFCs and varying VNF demands, neighbor (as opposed to far away) hosts have a higher chance to be selected under both schemes. The latter is associated with the fact that the per SFC resource requests, which are $\frac{1}{|S|}$ of the total network resource capacity, have a higher probability to be satisfied by one or more neighbor hosts, even for higher utilization ratios.

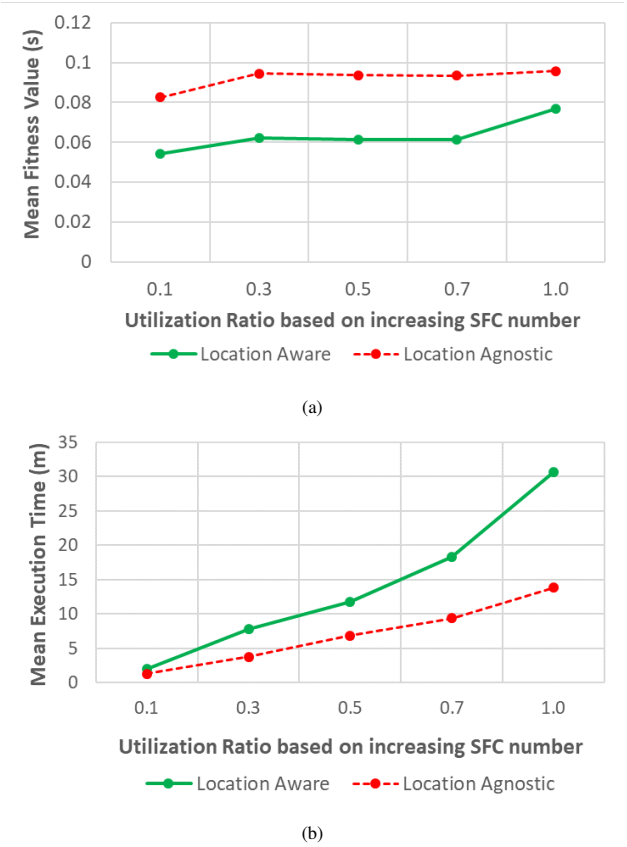


Figure 6: Location-Aware vs Location-Agnostic comparison for increasing Utilization Ratio, in terms of number of SFCs to be deployed

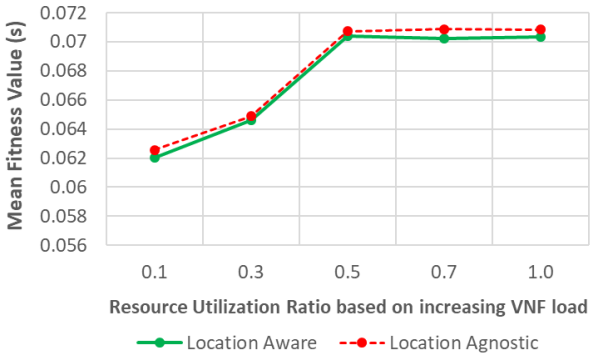
5. Discussion and Conclusions

This work presented an efficient, GA-based SFC placement scheme for 5G-enabled and beyond environments, such as Industry 4.0, aiming at minimizing the end-to-end delay of ultra-low delay industrial network operations, via also exploiting location aware information. A comprehensive problem formulation as a MILP optimization problem is presented and a low-complexity Genetic Algorithm - based solution to the SFC placement problem is derived, incorporating an early stopping criterion, as well as the proposed location-awareness and solution filtering optimizations. All aforementioned innovations have been evaluated and results are derived showing their effectiveness. The effectiveness of the proposed scheme is supported via an extensive set of simulation-based evaluation, under diverse network configurations and scales.

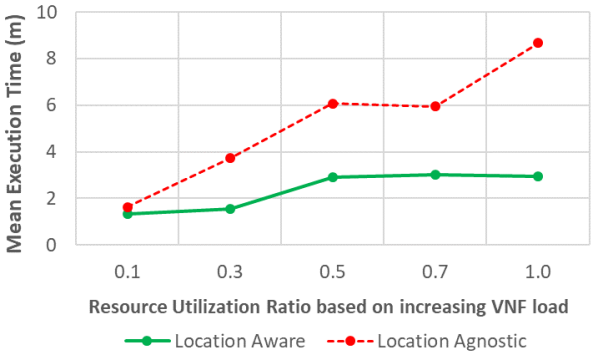
It is interesting to note that the proposed GA-based approach is shown to present a practically useful trade-off (controlled via the selected population size in the GA execution) between the closeness of the derived solution to the optimal (that yields the lowest possible delay) and the time needed to obtain this solution; as a result, by selecting accordingly the population size, the proposed approach could be applied to a wide range of use cases, ranging from Industry 4.0 scenarios (relatively static environment demanding ultra-low delays) to highly dynamic environments under moderate delay constraints requiring a quick solution. Future research directions include the consideration of more complex types of SFC topologies, as well as effective solution of a scheme that simultaneously (not sequentially) places SFCs in the underlying network topology. Furthermore, ways will be sought to reduce the likelihood of the premature convergence and, thus, enhance the effectiveness of the GA-based approach.

6. Acknowledgments

This research has been partially funded by EC H2020 5GPPP 5Growth project (Grant number: 856709).



(a)



(b)

Figure 7: Location-Aware vs Location-Agnostic comparison for increasing Utilization Ratio, in terms of VNF requested load

References

[1] P. Quinn, T. Nadeau, Problem Statement for Service Function Chaining, RFC 7498, 2015. URL: https://rfc-editor.org/rfc/rfc7498.txt. doi:10.17487/RFC7498.

[2] 3GPP TR 38.825, Study on NR Industrial Internet of Things (IoT), V16.0.0, 2019.

[3] NFV Release 2 Description, v1.11.0 (2020-07), ETSI, 2020.

[4] NFV Release 3 Description, v0.6.0 (2020-07), ETSI, 2020.

[5] NFV Release 4 Description, v0.2.0 (2020-07), ETSI, 2020.

[6] ETSI NFV and 3GPP 5G, [Online], 2019.

[7] G. Marchetto, R. Sisto, J. Yusupov, A. Ksentinit, Formally verified latency-aware vnf placement in industrial internet of things, in: 2018 14th IEEE International Workshop on Factory Communication Systems (WFCS), 2018, pp. 1–9.

[8] L. Ruiz Pérez, R. Durán Barroso, I. de Miguel, N. Merayo, J. Aguado, P. Fernández, R. Lorenzo, E. Abril, Design of VNF-Mapping with Node Protection in WDM Metro Networks, 2019, pp. 285–298.

[9] W. Mandarawi, J. Rottmeier, M. Rezaeighale, H. de Meer, Policy-based composition and embedding of extended virtual networks and sfcs for iiot, Algorithms 13 (2020) 240. URL: http://dx.doi.org/10.3390/a13090240. doi:10.3390/a13090240.

[10] L. Yala, P. A. Frangoudis, A. Ksentini, Latency and availability driven vnf placement in a mec-nfv environment, in: 2018 IEEE Global Communications Conference (GLOBECOM), 2018, pp. 1–7. doi:10.1109/GLOCOM.2018.8647858.

[11] A. Laghrissi, T. Taleb, M. Bagaa, H. Flinck, Towards edge slicing: Vnf placement algorithms for a dynamic realistic edge cloud environment, in: GLOBECOM 2017 - 2017 IEEE Global Communications Conference, 2017, pp. 1–6. doi:10.1109/GLOCOM.2017.8254653.

[12] Y. T. Chen, W. Liao, Mobility-aware service function chaining in 5g wireless networks with mobile edge computing, in: ICC 2019 - 2019 IEEE International Conference on Communications (ICC), 2019, pp. 1–6. doi:10.1109/ICC.2019.8761306.

[13] S. Lin, W. Liang, J. Li, Reliability-aware service function chain provisioning in mobile edge-cloud networks, in: 2020 29th International Conference on Computer Communications and Networks (ICCCN), 2020, pp. 1–9. doi:10.1109/ICCCN49398.2020.9209732.

[14] S. Agarwal, F. Malandrino, C. F. Chiasserini, S. De, Vnf placement and resource allocation for the support of vertical services in 5g networks, IEEE/ACM Transactions on Networking 27 (2019) 433–446.

[15] Q. Sun, P. Lu, W. Lu, Z. Zhu, Forecast-assisted nfv service chain deployment based on affiliation-aware vnf placement, in: 2016 IEEE Global Communications Conference (GLOBECOM), 2016, pp. 1–6.

[16] D. Harutyunyan, N. Shahriar, R. Boutaba, R. Riggio, Latency-aware service function chain placement in 5g mobile networks, in: 2019 IEEE Conference on Network Softwarization (NetSoft), 2019, pp. 133–141.

[17] R. Cziva, C. Anagnostopoulos, D. P. Pezaros, Dynamic, latency-optimal vnf placement at the network edge, in: IEEE INFOCOM 2018 - IEEE Conference on Computer Communications, 2018, pp. 693–701.

[18] M. C. Luizelli, W. Luis, L. S. Buriol, L. P. Gaspary, A fix-and-optimize approach for efficient and large scale virtual network function placement and chaining, Computer Communications 102 (2017) 67 – 77.

[19] A. Alleg, T. Ahmed, M. Mosbah, R. Riggio, R. Boutaba, Delay-aware vnf placement and chaining based on a flexible resource allocation approach, in: 2017 13th International Conference on Network and Service Management (CNSM), 2017, pp. 1–7. doi:10.23919/CNSM.2017.8255993.

[20] B. Addis, D. Belabed, M. Bouet, S. Secci, Virtual network functions placement and routing optimization, in: 2015 IEEE 4th International Conference on Cloud Networking (CloudNet), 2015, pp. 171–177. doi:10.1109/CloudNet.2015.7335301.

[21] M. A. Khoshkholghi, J. Taheri, D. Bhamare, A. Kassler, Optimized service chain placement using genetic algorithm, in: 2019 IEEE Conference on Network Softwarization (NetSoft), 2019, pp. 472–479.

[22] D. M. Manias, M. Jammal, H. Hawilo, A. Shami, P. Heidari, A. Larabi, R. Brunner, Machine learning for performance-aware virtual network function placement, in: 2019 IEEE Global Communications Conference (GLOBECOM), 2019, pp. 1–6.

[23] J. Martinéz-Peréz, F. Malandrino, C. Chiasserini, C. Bernardos, OKpi: All-KPI network slicing through efficient resource allocation, in: IEEE INFOCOM 2020 - IEEE Conference on Computer Communications, 2020.

[24] J. Pei, P. Hong, M. Pan, J. Liu, J. Zhou, Optimal vnf placement via deep reinforcement learning in sdn/nfv-enabled networks, IEEE Journal on Selected Areas in Communications 38 (2020) 263–278.

[25] H. Chai, J. Zhang, Z. Wang, J. Shi, T. Huang, A parallel placement approach for service function chain using deep reinforcement learning, in: 2019 IEEE 5th International Conference on Computer and Communications (ICCC), 2019, pp. 2123–2128.

[26] H. R. Khezri, P. A. Moghadam, M. K. Farshbafan, V. Shah-Mansouri, H. Kebriaei, D. Niyato, Deep reinforcement learning for dynamic reliability aware nfv-based service provisioning (2019) 1–6.

[27] P. T. A. Quang, Y. Hadjadj-Aoul, A. Outtagarts, A deep reinforcement learning approach for vnf forwarding graph embedding, IEEE Transactions on Network and Service Management 16 (2019) 1318–1331.

[28] D. B. Oljira, K. Grinnemo, J. Taheri, A. Brunstrom, A model for qos-aware vnf placement and provisioning, in: 2017 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN), 2017, pp. 1–7. doi:10.1109/NFV-SDN.2017.8169829.

[29] L. Aris, K. George, I. Mohamed, L. Ioannis, Vnf placement optimization at the edge and cloud †, Future Internet 11 (2019). URL: https://www.mdpi.com/1999-5903/11/3/69. doi:10.3390/fi11030069.

[30] G. Wang, S. Zhou, S. Zhang, Z. Niu, X. Shen, Sfc-based service provisioning for reconfigurable space-air-ground integrated networks, IEEE Journal on Selected Areas in Communications 38 (2020) 1478–1489. doi:10.1109/JSAC.2020.2986851.

[31] X.-S. Yang, Chapter 5 - genetic algorithms, in: X.-S. Yang (Ed.), Nature-Inspired Optimization Algorithms, Elsevier, Oxford, 2014, pp. 77–87. doi:https://doi.org/10.1016/B978-0-12-416743-8.00005-1.

[32] J. Sun, F. Liu, M. Ahmed, Y. Li, Efficient virtual network function placement for poisson arrived traffic, in: ICC 2019 - 2019 IEEE International Conference on Communications (ICC), 2019, pp. 1–7. doi:10.1109/ICC.2019.8761609.

[33] B. Thomas, Evolutionary algorithms in theory and practice (1996) 120.

[34] A.-R. Hedar, B. Ong, M. Fukushima, Genetic algorithms with automatic accelerated termination (2007).