Cluster Comput (2017) 20:2107–2117
DOI 10.1007/s10586-017-1004-3

VNF-EQ: dynamic placement of virtual network functions for energy efficiency and QoS guarantee in NFV

Sanghyeok Kim[1] · **Sungyoung Park**[1] · **Youngjae Kim**[1] · **Siri Kim**[2] · **Kwonyong Lee**[3]

Received: 11 March 2017 / Revised: 29 May 2017 / Accepted: 19 June 2017 / Published online: 1 July 2017

Abstract With the advances of network function virtualization and cloud computing technologies, a number of network services are implemented across data centers by creating a service chain using different virtual network functions (VNFs) running on virtual machines. Due to the complexity of network infrastructure, creating a service chain requires high operational cost especially in carrier-grade network service providers and supporting stringent QoS requirements from users is also a complicated task. There have been various research efforts to address these problems that only focus on one aspect of optimization goal either from users such as latency minimization and QoS based optimization, or from service providers such as resource optimization and cost minimization. However, meeting the requirements both from users and service providers efficiently is still a challenging issue. This paper proposes a VNF placement algorithm called VNF-EQ that allows users to meet their service latency requirements, while minimizing the energy consumption at the same time. The proposed algorithm is dynamic in a sense that the locations or the service chains of VNFs are reconfigured to minimize the energy consumption when the traffic passing through the chain falls below a pre-defined threshold. We use genetic algorithm to formulate this problem because it is a variation of the multi-constrained path selection problem known as NP-complete. The benchmarking results show that the proposed approach outperforms other heuristic algorithms by as much as 49% and reduces the energy consumptions by rearranging VNFs.

Keywords NFV · VNF placement · Service function chaining · Energy efficient · Reconfiguration

✉ Sungyoung Park
parksy@sogang.ac.kr

Sanghyeok Kim
sangh228@sogang.ac.kr

Youngjae Kim
youkim@sogang.ac.kr

Siri Kim
siri.kim1012@gmail.com

Kwonyong Lee
kwonyong82@gmail.com

[1] Department of Computer Science and Engineering, Sogang University, 35 Baekreom-ro, Mapo-gu, Seoul, Korea

[2] SK Telecom, 65 Eulji-ro, Jung-gu, Seoul, Korea

[3] SDS Tech. Lab, SK Telecom, 65 Eulji-ro, Jung-gu, Seoul, Korea

1 Introduction

Due to the recent growth of Internet of Things (IoT), the traffic on data networks has increased significantly. The user requirements associated with these data services are also rapidly changing, making it difficult for network service providers (NSPs) to efficiently support the users [1,2]. Currently, NSPs use dedicated middlebox equipments such as firewalls, NATs, and load-balancers in conjunction with their own solutions to provide various network services. Using those physical equipments is quite costly and is inflexible as network services are becoming increasingly complex.

With the advances of cloud computing and virtualization technologies, new networking paradigms such as software-defined networking (SDN) and network function virtulization (NFV) have emerged [3]. These emerging paradigms separate network functions from the underlying physical resources and allows NSPs to deploy various network services on top of the virtualized infrastructure. In this environment, NSPs usually launch virtual network functions (VNFs) such as firewalls and proxy servers on top of virtual machines

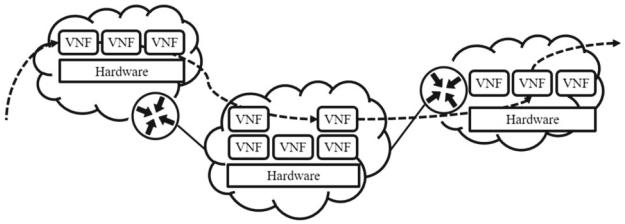


Fig. 1 Service function chaining: SFC

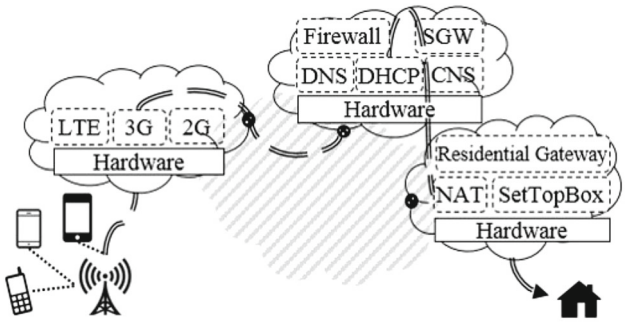


Fig. 2 NFV usecase

(VMs) inside data centers and connect various types of VNFs based on customer demands in order to provide network services as shown in Fig. 1. Providing these VNFs with a certain order or deploying the VNFs is known as service function chaining (SFC).

For example, in order to provide secure residential services from mobile users, a service chain: user → service gateway (SGW) → firewall (FW) → residential gateway (RGW) → NAT → terminal can be created as shown in Fig. 2. The type and order of each VNF is typically determined based on service level agreements (SLAs) between users and service providers, or provisioning policies from operators. Due to the complexity of network infrastructure, creating a service chain requires high operational cost especially in carrier-grade network service providers and meeting QoS requirements over the chain is also a challenging task. Moreover, one of the main difficulties in creating an SFC is that the importance and relevance of each optimization criterion depend on many factors, including network latency, resource utilization, operation cost, power usage etc., which contradicts each other from the viewpoints of both users and service providers.

According to a recent study [4], the power consumption of data centers is increasing rapidly and various research efforts to reduce the power in data centers are actively underway [5–7]. In an environment where VNFs are created on VMs that utilize the resources provided by the underlying physical machines (PMs), the decision on where the VNFs are located has a significant impact on the efficient use of resources and power consumption in data centers. Furthermore, as shown in Fig. 3, the fluctuations of network traffic can be severe, with

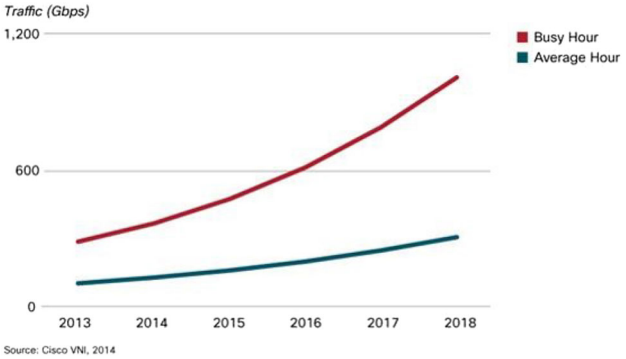


Fig. 3 Traffic difference between busy and average hour

huge traffic differences between busy and average hours, and this problem is predicted to become worse [8]. This is because a number of physical servers are used to process the traffic at busy times, while still guaranteeing the QoS demanded by the user. If the same number of physical servers were used even during the average times, the number of idle servers would increase, leading to a waste of energy. According to [9], idle physical servers waste more than 60% of the maximum power consumption. Therefore, it is also necessary to rearrange the VNFs in order to prevent the waste of energy during the periods of relatively low traffic.

A number of studies have been carried out recently regarding the SFC construction and efficient placement of VNFs [10]. Some research works [11,12] present early studies on how to place VNFs to satisfy the QoS requirements of users. However, the VNF types are limited, and detailed SFC algorithms for reducing energy consumption are not provided. VNF-OP [13] provides an optimazation algorithm as well as a heuristic algorithm to minimize the network operational expenditures (OPEX) without violating SLAs. This proposal includes energy cost in the overall OPEX and decides VNF locations dynamically by considering the strategy that minimizes OPEX and penalty for SLA violations. A proposal like [14] suggests a bi-criteria soluition that locates VNFs to minimize the overall network cost, while adhering to the allocation size of the servers. In this work, they assume that locating VNFs is performed in a practical environment where servers have limited resources for allocating functions. Enterprises that provide commercial services, such as Ericsson [15] and Huawei [16], also propose solutions regarding an efficient management of SFC from the perspectives of service providers. Their main focuses are either on developing a simple SFC scheme according to the subscribers policy, or on building a mechanism to reduce the overall operational cost. Detailed algorithms for such solutions are currently not publicized. There are few studies to address the energy and reconfiguration issues related to the SFC.

This paper proposes a VNF placement algorithm called VNF-EQ that allows users to meet their service latency

requirements, while minimizing energy consumption at the same time. In the proposed algorithm, a shortest path between a source and a destination of a service is first computed using the Dijkstra's algorithm based on the service latency requirements from users. Then, the VNFs are allocated to appropriate VMs along the path that can minimize the energy consumption. The proposed algorithm is dynamic in a sense that the locations of VNFs are rearranged or possibly the SFC path is reconfigured to minimize the energy consumption when the traffic passing through the service chain falls below a pre-defined threshold. In this case, the VNFs with low service traffic are merged into other VNFs with the same type as much as we can and the original VNFs are de-allocated to reduce the overall energy consumption. We use genetic algorithm (GA) to formulate this problem because it is a variation of the multi-constrained path selection problem known as NP-complete. We also implement two heuristic algorithms to compare and analyze the performance of the proposed approach: VNF-QF (VNF placement with latency-first approach) and VNF-EF (VNF placement with energy-first approach). Compared with the results of these heuristic algorithms, we are able to see that the proposed approach outperforms other heuristics by as much as 49% while still satisfying the QoS requirements. The benchmarking results also show that the reconfiguration algorithm can reduce the energy consumption per service when it is applied to all three algorithms.

This paper is organized as follows. In Chap. 2, we examine various SFC-related solutions. In Chap. 3, we present the proposed energy-aware VNF placement and reconfiguration algorithms to reduce energy consumptions while meeting the QoS requirements of users. Chapter 4 discusses the performance evaluation and analysis results of the proposed approach. Chapter 5 concludes this paper and discusses possible future works.

2 Related works

A comprehensive survey of current SFC architecture and recent developments in SFC is presented in [10]. This research also summarizes various SFC optimization approaches and their related research efforts such as network latency minimization, resource utilization optimization, cost minimization, power minimization, SLA-based optimization, and QoS-based optimization.

VNF-P [11] is a hybrid NFV network structure in which existing network equipments and VNFs are used together to meet network service demands. If network service demands increase rapidly, the network functions provided by existing hardware are initially used up to its maximum capacity. However, if the demands exceed the maximum capacity of hardware, or the QoS requirements cannot be guaranteed, software-based VNFs are instantiated and used to handle the demands instead. The main focus of this research is how and where to allocate VNFs in this hybrid NFV environment. Although such an algorithm can reduce the number of physical servers, no clear algorithm is suggested for creating an SFC that requires multiple VNFs. Moreover, the VNF types are only limited to firewalls and routers, making it difficult for the method to be applied in real situations. A flexible network structure to locate VNFs dynamically in fluctuating networks and service environments is also proposed in [12]. They suggest three network algorithms for load balancing, energy saving, and QoS guarantee. Instead of proposing how to create an SFC with different types of VNFs from source to destination, they mainly focus on where to place VNFs in order to meet the service requirements in a dynamic situation. Moreover, as the VNF being arranged is limited to a virtual router, it is difficult to apply them when the SFC requires multiple VNFs to provide a service.

VNF-OP [13] proposes a solution to determine the required number and palcement of VNFs that optimizes network operational cost and utilization, without violating SLAs. This approach includes energy cost and penalty for SLA violation as a part of network OPEX and dynamically rearranges VNFs to minimize the OPEX. This research is similar to our proposal in such a way that it proposes a solution from the viewpoints of both users and NSPs. However, the reconfiguration of SFC path is not considered in this work. Another research that proposes a bi-criteria solution is [14]. In this research, they propose near optimal approximation algorithms guaranteeing a NFV placement with theoretically proven performance based on the fact that servers have limited resources to run VNFs.

Several solutions for SFC construction from industry such as Ericsson and Huawei have been proposed. Ericsson dynamic service chaining [15] enables NSPs to manage and orchestrate the network services. With this solution, the traffic from each subscriber only traverses a certain path consisting of a set of service functions as defined by the policy for that particular subscriber. Huawei service based routing (SBR) [16] is a solution that provides value added service (VAS) by combining service enablers to create flexible and scalable mobile service zones. The SBR controls the policies and combines the service enablers dynamically based on user awareness, radio access awareness, service type awareness, and so on. Huawei predicts that, with SBR, it is possible to provide rapid service chaining and reduce the total cost of ownership (TCO) by up to 60%, under typical conditions. However, their solutions are targeted at meeting the service requirements from the perspectives of service providers (e.g., overall operational cost) and are mostly hidden, which prevents us from directly comparing their algorithms with ours.

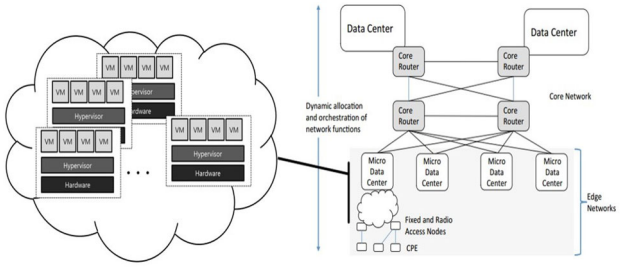


Fig. 4 System model

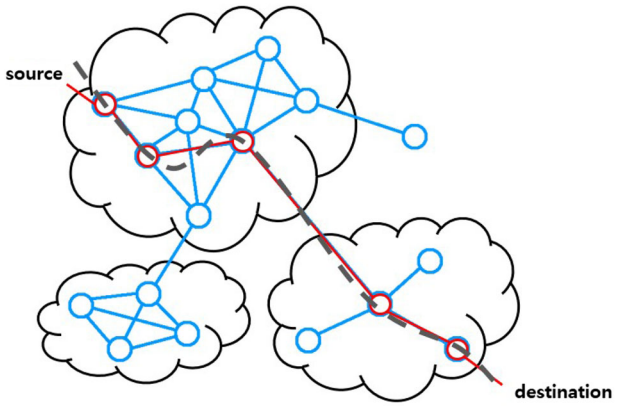


Fig. 5 Example graph for the problem

3 VNF-EQ: energy-aware VNF placement and reconfiguration algorithm

In this section, we describe energy-aware VNF placement and reconfiguration algorithms for reducing energy consumption while guaranteeing the service latency requirements by users.

3.1 System model

Figure 4 depicts a system model that is used to propose the algorithm presented in this paper. As shown in the figure, we assume that micro data centers are located on the edge network and are connected to the main data centers through core routers. Multiple cloud servers can be located at micro data centers and main data centers. Each server is virtualized by a hypervisor so that multiple VMs can be created. We also assume that each VM can be added, deleted, and migrated as necessary. The network providers can place multiple VNFs on appropriate VMs and create an SFC path to provide various network services.

3.2 Problem definition

Assume that several data centers with a collection of physical servers are connected with a graph $G = (V, E)$ as shown in Fig. 5. In the graph, V refers to the data centers with a collection of physical servers on which the VNFs can be placed, while E refers to the virtual link between the data centers. If the number of VNFs that are used to create an SFC path is K when providing the service S_X to user X, the SFC path is expressed as $Path_{SX} = VNF_1 \rightarrow VNF_2 \rightarrow ... \rightarrow VNF_K$ and the energy consumption along the path is E_{SX}, which is the sum of energy consumption of each VNF. Then, if the total number of SFC path requests is N, the total energy consumption E^{total} can be expressed by the sum of energy consumption of each SFC path as shown in Eq. (1). Therefore, if the target service latency along the path is Lat_{SX}^{target}, the objective of the proposed algorithm is to minimize the overall energy consumption E^{total}, while satisfying $Lat_{SX}^{target} \geq Lat_{SX}^{estimate}$, where $Lat_{SX}^{estimate}$ is an estimated service latency along the path that includes transmission times and VNF processing times.

Meanwhile, in order to prevent the waste of energy during average hours (used originally to guarantee the QoS at busier times), the locations or the service chains of VNFs are reconfigured when the service traffic falls below a predefined threshold value. In this case, the VNFs with low service traffic are consolidated with other VNFs with the same service type and the original VNFs are deallocated to minimize E^{total}, thereby reducing the number of idle servers. A server is assumed to be active when it has at least one VNF with active service traffic, while a server is idle when there is no active service traffic running on VNFs.

$$E^{total} = \sum_{i=1}^{N} E_{SX_i} \tag{1}$$

3.3 Energy-aware VNF placement algorithm

The algorithm proposed in this paper consists of two parts. First, the source requesting the service and the destination of network flow are set as the source and destination of an SFC, respectively. The Dijkstra's algorithm is then used to find the shortest path between the two points. If the path of the network flow is determined, the VNFs should be placed on the physical servers across the path. The placement should be made to satisfy the QoS requirements (service latency), while also minimizing the overall energy consumption. Second, if the service traffic on the SFC path is below a pre-defined threshold value, then the locations of VNFs are rearranged or the SFC path are possibly reconfigured to minimize the overall energy consumption. In what follows, the detailed explanation on the first step is given below and the algorithm is illustrated in Algorithm 1.

3.3.1 Shortest path from the source to the destination

In order to decide a shortest path between a source and a destination, two constraints such as the service type for service S_X and the latency limitation Lat_{SX}^{target} are used to calculate the path. Since the VNFs are placed on the physical servers along this path and the service latency includes both transmission time and VNF processing time, the estimated latency $Lat_{SX}^{estimate}$ is calculated based on the assumption that a VNF with average processing time determined from the target VNFs is placed on the path, which reduces the possibility of latency violation. Moreover, we also avoid a physical server running other VNFs and thus possibly violating their target latencies by placing the requested VNF.

3.3.2 VNF placement using GA

In order to minimize the overall energy consumption while guaranteeing the service latency requested by the users, we use a genetic algorithm (GA). The GA solves a problem through two main steps after the population construction. The first step is the selection of two parents that will produce the first offspring, and the second step is to decide how many offspring will be created through crossover and mutation by the selected parents.

Population construction A physical server is randomly selected K times from the designated path. Then, the $Lat_{SX}^{estimate}$ along the path is calculated. The service latency Lat_{SX} for the service S_X can be expressed by Eq. (2), where PT_{SX} refers to the total time for processing service S_X while TT_{SX} refers to the sum of times for transmitting the result from one VNF to the next VNF along the path. PT_{SX} and TT_{SX} are defined in Eq. (3):

$$Lat_{SX} = PT_{SX} + TT_{SX} \tag{2}$$

$$PT_{SX} = \sum_{i=1}^{K} PT_{VNF_i} \quad TT_{SX} = \sum_{i=1}^{K-1} TT_{VNF_i} \tag{3}$$

PT_{SX} is related to the number of physical CPUs (pCPUs) in the server on which VNFs are placed. If the total number of virtual CPUs (vCPUs) allocated for a VNF is less than or equal to the number of pCPUs, PT_{VNF_i} (processing time for virtual function) is assumed to be $PT_{VNF_i}^{ideal}$ for simplicity, where $PT_{VNF_i}^{ideal}$ is an ideal processing time when one pCPU is mapped to one vCPU. As a consequence, if $pCPU_i \geq vCPU_i$, $PT_{VNF_i}^{estimate} = PT_{VNF_i}^{ideal}$. On the contrary, if $pCPU_i < vCPU_i$, the vCPUs should be scheduled to use pCPUs. In this case, $PT_{VNF_i}^{estimate}$ can be expressed as Eq. (4) assuming that the scheduling algorithm is round robin.

$$PT_{VNF_i}^{estimate} = \frac{vCPU_i}{pCPU_i} \times PT_{VNF_i}^{ideal} \tag{4}$$

Selection This is a process of selecting solutions that will become the candidates to be transferred from one generation to the next. The energy consumption E_i of a physical server i depends on its status. If the i^{th} physical server is powered off, its energy consumption is 0. When the server is idle, $E_i = E_i^{idle}$, where E_i^{idle} represents the idle system power. Meanwhile, according to [13,17], the energy consumption of a physical server increases linearly depending on the CPU utilization of pCPU in the server. Therefore, to simply the problem, we assume that the energy consumption E_i is only proportional to the CPU utilization as defined in Eq. (5). When $vCPU_i^{used} \geq pCPU_i^{total}$ (i.e., the CPU utilization is 100% or more), the energy consumption E_i is E_i^{max}. If the number of servers running VNFs along the SFC is K and the total number of services is N, the energy used for the service E_{SX} and the overall energy consumption E^{total} can be expressed as Eq. (6), respectively.

Based on the notations given in Eqs. (5) and (6), a fitness function f_{SX} is proposed using the roulette wheel selection algorithm as defined in Eq. (7), where E_{worst}^{total} and E_{best}^{total} represent the maximum and minimum energy values generated from the candidates. This fitness function is designed such that the fitness value of the best solution becomes k times that of the worst solution and the low energy consumption produces better fitness value. If a selected candidate does not meet the latency requirement, the energy consumption of the cadidate is doubled (i.e., $E_{SX} = E_{SX} \times 2$) to reduce the possibility to be selected in the subsequent iterations. The value of k is set to 3 since $k = 3$ or 4 is generally used in the roulett wheel selection.

$$E_i = E_i^{idle} + \frac{vCPU_i^{used}}{pCPU_i^{total}} \times \left(E_i^{max} - E_i^{idle}\right) \tag{5}$$

$$E_{SX} = \sum_{i=1}^{K} E_i \qquad E^{total} = \sum_{i=1}^{N} E_{SX_i} \tag{6}$$

$$f_{SX} = \left(E_{worst}^{total} - E^{total}\right) + \frac{\left(E_{worst}^{total} - E_{best}^{total}\right)}{(k-1)} \tag{7}$$

Crossover and mutation This is a process for creating a new solution by crossing over the parent solutions selected by the fitness function f_{SX}, and by mutating the created offspring. One-point crossover is used as the crossover method, with the crossover rate set as 0.015 (1.5 %).

3.4 VNF reconfiguration algorithm

As reported in [8], the network traffic is bursty and fluctuates widely over time. The traffic differences between busy and average hours are also huge. NSPs gradually increase

Algorithm 1 VNF Placement Algorithm

K : number of servers running VNFs for S_X
N : total number of service requests
Lat_{SX}^{target}, $Lat_{SX}^{estimate}$: target/estimated latency for S_X
E_i : energy consumption of each server in the SFC path
E_i^{idle}, E_i^{max} : idle/max energy of each server
E_{SX} : total energy consumption of S_X
E^{total} : overall total energy consumption
f_{SX} : fitness function used in GA
E_{worst}^{total}, E_{best}^{total} : max/min energy among candidates
k : control value for selection pressure in GA (k ← 3)
$bcnt$: number of times S_X is blocked

procedure VNF PLACEMENT
  while S_X exists in the input or waiting queue **do**
    S_X ← service request
    Source ← terminal
    Destination ← user
    Find a shortest path from **Source** to **Destination**
    (Latency should be smaller than target latency)
    if a path exists **then**
      Use GA to find a place for each VNF
      Generate population and get $Lat_{SX}^{estimate}$
      $E_i \leftarrow E_i^{idle} + \frac{vCPU_i^{used}}{pCPU_i^{total}} \times \left(E_i^{max} - E_i^{idle}\right)$
      $E_{SX} \leftarrow \sum_{i=1}^{K} E_i$, $E^{total} \leftarrow \sum_{i=1}^{N} E_{SX_i}$
      $f_{SX} \leftarrow \left(E_{worst}^{total} - E^{total}\right) + \frac{\left(E_{worst}^{total} - E_{best}^{total}\right)}{(k-1)}$
      Apply one point crossover and mutation
      Place VNFs if $Lat_{SX}^{target} \geq Lat_{SX}^{estimate}$
      if placement is possible **then**
        Allocate VNFs on the specified path
      else if placement is not possible **then**
        Increase priority of S_X by 2^{bcnt}
        if $bcnt \leq$ MAXBLOCK **then**
          push S_X to waiting queue
        end if
      end if
    else if a path does not exist **then**
      Remove the service request S_X
    end if
  end while
end procedure

or optionally over-provision resources to handle spikes or to meet the QoS requirements from the users during the busy hours. However, when traffic volumes change at average hours, this over-provisioning results in a waste of resources which may attribute to increasing the overall operational cost. In the proposed VNF placement algorithm given in the previous subsection, the main focus is to avoid latency violation of each service, while minimizing the overall energy consumption. For this, we assume that each VNF runs the maximum traffic every hour. Although, this arrangement avoids the latency violation of each service, it may waste CPU resources and consume unnecessary energy. Therefore, this paper also proposes a VNF reconfiguration algorithm such that a VNF with low service traffic is merged with another VNF with the same type. On the other hand, when the service traffic in the shared VNF exceeds a pre-defined threshold value, the VNF is splited and a new service path is established. The detailed explanation on the algorithm is given below with a pseudo-code presented in Algorithm 2.

VNF sharing The goal of VNF reconfiguration is to reduce the number of used VNFs and idle servers, thereby minimizing the waste of energy. For this purpose, this algorithm presents a solution to share one VNF by several service paths. If the amount of service traffic using a VNF is less than α (α is a configurable parameter set by administrators), the reconfiguration step is triggered and the service traffic is relocated from the original VNF to the other destination VNFs with the same type. In this case, the sum of the original traffic volume on the selected VNF and the traffic volume relocating for sharing should not exceed a pre-defined value β (β is also a configurable parameter set by administrators). This is to ensure the stability of the service by limiting the amount of traffic. When a VNF is relocated, the energy is not considered and a VNF with the smallest latency on the paths that are reachable to the detination is selected. This reduces the amount of traffic handled by the VNF and prevents the possibility of latency violation that may occur by sharing.

VNF spliting When multiple services share a VNF, the total sum of the traffic processed by the shared VNF may exceed β as the traffic fluctuates especially in busy hours. This does not guarantee the stability of the service. To solve this problem, we suggest a VNF spliting solution in the proposed reconfiguration algorithm. If the sum of the amount of traffic handled by the VNF is larger than β, the recently shared service traffic is removed from the VNF and a new service path is created using the placement algorithm presented in Subsect. 3.3.

4 Performance evaluation

In order to show the effectiveness of VNF-EQ, we implement a custom made simulator using C that can handle different parameters and algorithms proposed in Sects. 3.3 and 3.4. We evaluate the performance based on the following four aspects: (1) comparing total energy consumption, (2) comparing the blocking rate of service requests, (3) comparing the energy consumption per service, (4) comparing the amount of energy saved by using reconfiguration. Moreover, we implement two heuristic algorithms to compare their performance with that of VNF-EQ: VNF-QF (QoS(latency)-first VNF palcement) and VNF-EF (energy-first VNF placement). The VNF-QF places VNFs to minimize the latencies of all service paths, while the VNF-EF places VNFs to minimize the overall energy consumption. For this, the VNF-QF tries to distribute VNFs almost evenly to every physical server and the VNF-EF consolidates VNFs as much as we can to save energy. All three algorithms used in the experiments guarantee the latency requirement of a new service and the latency requirements of existing services in deploying VNFs.

Algorithm 2 Reconfiguration Algorithm

α : lower bound of service traffic for VNF sharing
β : upper bound of service traffic for VNF splitting
$T^{SX}_{VNF_i}$: traffic amount of service S_X on a VNF
$T^{total}_{VNF_i}$: total traffic amount running on a VNF

while $T^{SX}_{VNF_i} \geq \alpha$ **and** $T^{total}_{VNF_i} \leq \beta$ **do**
  Monitor traffic on every service path
end while
Call VNF RECONFIGURATION

procedure VNF RECONFIGURATION
  if $T^{SX}_{VNF_i} < \alpha$ **then**
    (VNF Sharing)
    For all $VNF_k \in$ VNFs of service S_X
    Search VNF_j satisfying $T^{total}_{VNF_j} + T^{SX}_{VNF_k} \leq \beta$
    Allocate $T^{SX}_{VNF_k}$ on VNF_j
    Deallocate VNF_k
  else if $T^{total}_{VNF_i} > \beta$ **then**
    (VNF Splitting)
    For all $VNF_k \in$ VNFs of service S_X
    Deallocate $T^{SX}_{VNF_k}$ on VNF_k
    Create a new service S_N
    call VNF PLACEMENT using a new S_N
  end if
end procedure

Table 1 Server attributes for the simulation

Server attributes [18]		
Num of cores	E^{max} / E^{idle}	pCPU : vCPU
12	243W / 126W	1 : 2
VNF attributes [19–21]		
VNF type	$CPU_{required}$	Processing capacity
Proxy	2	1 Gbps
Firewall	4	2 Gbps
IDS	4	600 Mbps

4.1 Experimental environment

For the simulation, we assume that there are 7 data centers and each center is equipped with 16 physical servers (i.e., total 112 physical servers). We also assume that 7 data centers are inter-connected with random topology and the physical servers within the data centers are connected with a fat tree topology. The bandwidths between data centers and physical servers are all assumed to be 1 Gbps. Each physical server is assumed to have 12 cores and the ratio of pCPU to vCPU is 1 to 2 (i.e., 24 virtual CPUs per server). Therefore, the maximum number of VMs per PM is 24. The peak and idle power consumptions are 243W and 126W, respectively, as shown in Table 1.

We have also generated various service requests with length 3 (e.g., Proxy → Firewall → IDS) using the VNF attributes provided in Table 1. We have used publicly available data sheets from hardware manufacturers and some of the parameters such as PT^{ideal}_{VF} are calculated from the real hardware data [18–21]. Meanwhile, the target service for the performance evaluation is video-streaming service, which has the property of being a long-lived flow with packet sizes from 64 to 1500 bytes [22,23]. Traffic was generated using the Weibull distribution because the inter-arrival time of the traffic follows the distribution.

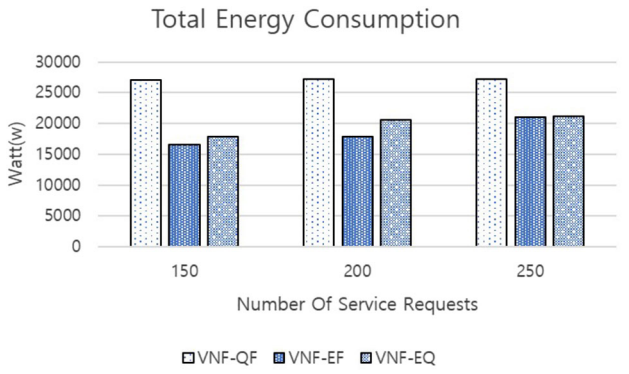


Fig. 6 Comparison of total energy consumption

4.2 Performance comparison

4.2.1 Comparison of VNF placement

In order to analyze the placement performance of the proposed algorithm (VNF-EQ), we measure total energy consumption, service blocking rate, and energy consumption per service of VNF-EQ, VNF-QF, and VNF-EF algorithms, respectively. The performance results are based on the average values after repeating the experiment ten times, with three different traffic types such as 150, 200, and 250 service requests per second.

Figure 6 compares the total energy consumptions of the VNF-EQ, VNF-QF, and VNF-EF algorithms, with three different traffic types. As shown in Fig. 6, the total energy consumption of VNF-EQ is less than that of VNF-QF, while it is a little bit bigger than that of VNF-EF for all three traffic types. This can be explained by the results shown in Fig. 7. For example, the service blocking rates in VNF-EF is much higher than that of VNF-EQ, which means that the number of successful service requests in VNF-EF is small compared to that of VNF-EQ. Since the total energy consumption that only accumulates the energy consumptions of successful service requests is typically affected by the service blocking rate, the total energy consumption alone should not be used to measure how much energy is reduced.

Figure 8 shows the comparison of energy consumption per service when three algorithms use only the VNF placement algorithm. This experiment is to see how well the GA-based

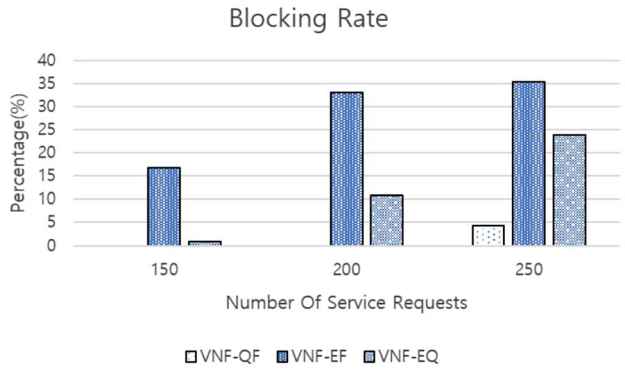


Fig. 7 Comparison of service blocking rate

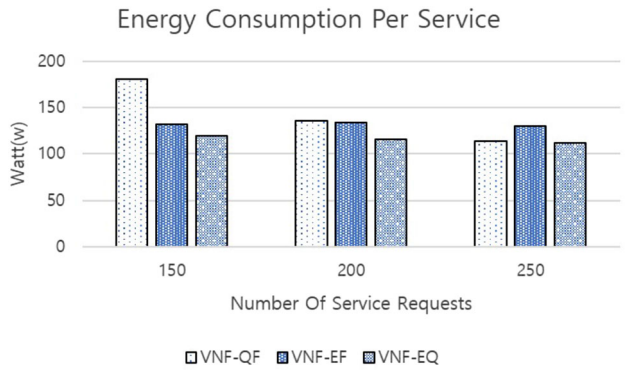


Fig. 8 Comparison of energy consumption per service

placement algorithm proposed in this paper performs in order to reduce the energy without violating the service latency requirement. As shown in Fig. 8, the energy consumptions per service of VNF-EQ in all three traffic types are smaller than those of VNF-QF and VNF-EF. Moreover, the performance gap between VNF-EQ and VNF-EF is widening as we increase the amount of traffic. For example, the performance difference between VNF-EQ and VNF-EF is only 10% in 150 service requests per second, while the difference goes up to 16% in 250 service requests per second. It is more likely that the performance of VNF-EF is a little limited since it explores only a smaller solution space than the GA-based solution.

4.2.2 Comparison of VNF reconfiguration

To investigate the effectiveness of reconfiguration, we first measure the energy consumptions per service for VNF-EQ, VNF-EF, and VNF-QF with and without reconfiguration during 400 simulation times. The energy consumption per service is an appropriate indicator of energy saving since the value for total energy consumption does not show the effect of service blocking. It is worthy to note that if the blocking rate is getting higher, the total energy consumption is decreased. For this experiment, the service request traffic is

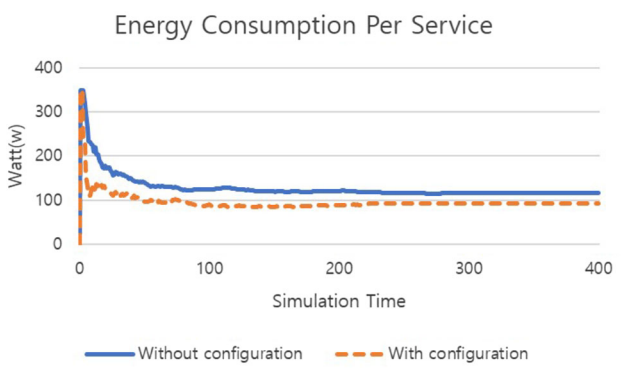


Fig. 9 Energy consumption with/without reconfiguration (VNF-EQ)

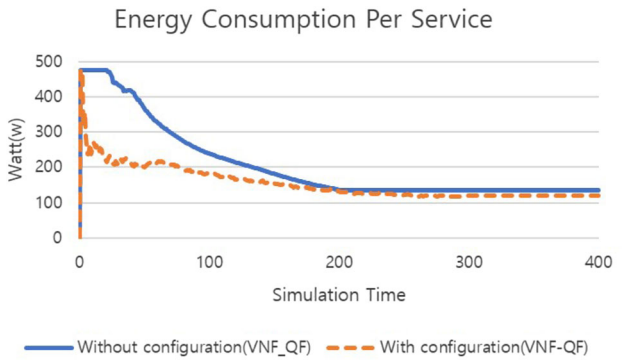


Fig. 10 Energy consumption with/without reconfiguration (VNF-QF)

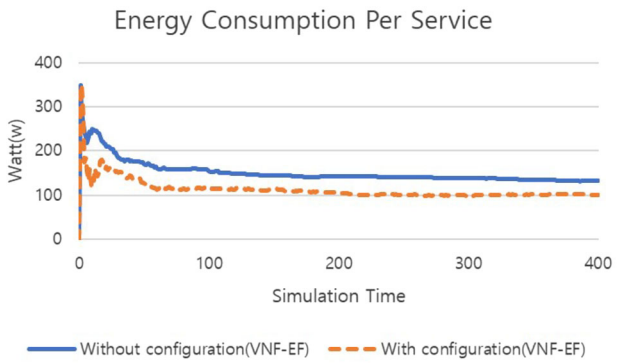


Fig. 11 Energy consumption with/without reconfiguration (VNF-EF)

set to 200 requests per second and two configurable threshold values such as α and β are set to 20 and 80% of the maximum service traffic, respectively.

Figures 9 through 11 present the performance results of three algorithms when the reconfiguration is applied to each algorithm. As can be seen in Fig. 9, the performance of VNF-EQ is improved by about 21% when the reconfiguration algorithm is applied. For example, the total energy consumption of VNF-EQ with reconfiguration at 400 simulation time is only 79% of the VNF-EQ performance when the reconfiguration is not used. When a VNF is initially placed and

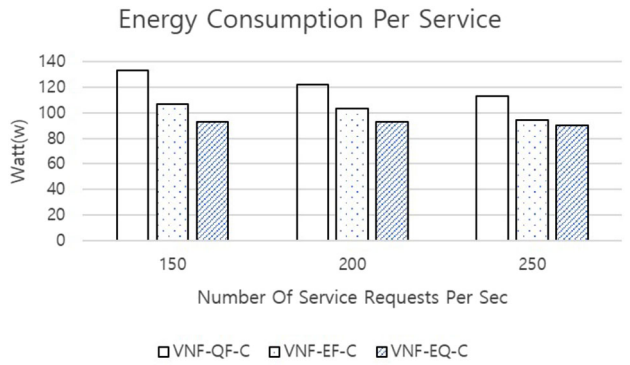


Fig. 12 Comparison of energy consumption per service with reconfiguration

heavily used, it is more likely that each VNF uses one VM alone. In the proposed algorithm, however, when the reconfiguration is triggered, VNFs can be merged by VNF sharing and the original VNF is deallocated, which reduces the overall energy consumption. The reconfiguration also affects the performance of VNF-QF and VNF-EF as shown in Figs. 10 and 11. As expected, both VNF-QF and VNF-EF with reconfiguration also outperform corresponding algorithms without reconfiguration, and the performance difference in VNF-EF is even higher than that of VNF-EQ. For example, the maximum performance improvement in VNF-EF between the two cases (with reconfiguration vs without reconfiguration) is about 24%. This is because the two reconfiguration strategies such as VNF sharing and VNF splitting play an important role in increasing the efficiency of energy use by continuous monitoring of each traffic.

Figure 12 illustrates the performance comparison using energy consumption per service when all of the three algorithms use the reconfiguration strategies. As shown in Fig. 12, the VNF-EQ with reconfiguration (VNF-EQ-C) outperforms other two algorithms. The performance improvement in low service rate (150 service requests per second) is much bigger than those in higher service rates (200 and 250 service requests per second). For example, the performance difference between VNF-EQ with reconfiguration (VNF-EQ-C) and VNF-QF with reconfiguration (VNF-QF-C) in 150 requests per second is about 42%, while the difference in 250 requests per second is only 25%. However, if we compare VNF-EQ-C with other two algorithms without reconfiguration (shown in Fig. 8), the VNF-EQ-C outperforms two algorithms by about 49% for all three traffic types.

5 Conclusion and future work

In this paper, we have proposed an energy-aware VNF placement and reconfiguration algorithm, while still guaranteeing the service latency target requested by the user in a cloud server environment. The proposed approach is based on the GA and we have also developed two heuristic approaches to anaylze and compare the performance of the proposed algorithm. The experimental results show that the proposed algorithm reduces energy consumption by as much as 19% in average when only placement strategy is applied and the performance is improved up to 49% when the reconfiguration is combined with the placement algorithm.

As the proposed algorithm does not consider a change in network bandwidth during the search for the SFC path, the VNFs should be rearranged to guarantee the QoS when the bandwidth is restricted. We are also planning to consider more complex optimization factors such as resource utilization and operation cost, etc. Further, as the GA is used for the rearrangement of VNFs (VNF splitting), it is possible that reconfiguration decision takes a little bit longer than we expect, which is not appropriate if the reconfiguration has to be done quickly. We are currently investigating those issues.

Acknowledgements This research was supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the ITRC(Information Technology Research Center) support program (IITP-2017-2016-0-00465) supervised by the IITP(Institute for Information & communications Technology Promotion).

References

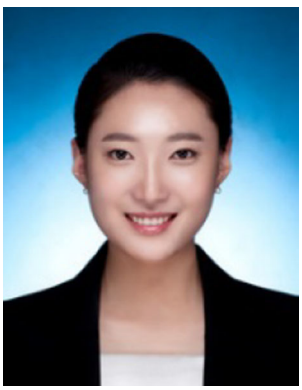
1. John, W., Pentikousis, K., Agapiou, G., Jacob, E., Kind, M., Manzalini, A., Risso, F., Staessens, D., Steinert, R., Meirosu, C.: Research directions in network service chaining. In: Future Networks and Services (SDN4FNS), IEEE SDN, pp. 1–7 (2013)
2. Masutani, H., Nakajima, Y., Kinoshita, T., Hibi, T., Takahashi, H., Obana, K., Shimano, K., Fukui, M.: Requirements and design of flexible NFV network infrastructure node leveraging SDN/OpenFlow. In: Optical Network Design and Modeling, pp. 258–263 (2014)
3. Network Function Virtualization.: European Telecommunications Standards Institute (ETSI). http://www.etsi.org/technologiesclusters/technologies/nfv
4. Zhang, Y., Ansari, N.: Green data centers: Handbook of green information and communication systems (2012)
5. Anagnostopoulou, V., Biswas, S., Savage, A., Bianchini, R., Yang, T., Chong, F.T.: Energy conservation in datacenters through cluster memory management and barely-alive memory servers. In: 2009 Workshop on Energy Efficient Design (2009)
6. Chen, H., Kesavan, M., Schwan, K., Gavrilovska, A., Kumar, P., Joshi, Y.: Spatially-aware optimization of energy consumption in consolidated data center systems. In: ASME 2011 Pacific Rim Technical Conference and Exhibition on Packaging and Integration of Electronic and Photonic Systems. American Society of Mechanical Engineers, pp. 461–470 (2011)
7. Ghosh, S., Redekopp, M., Annavaram, M.: Knightshift: shifting the i/o burden in datacenters to management processor for energy efficiency. In: Computer Architecture, Springer, NewYork pp. 183–197 (2012)
8. Cisco Vni-report. https://dunstewart.wordpress.com/2014/06/10/top-5-data-trends-from-cisco-vni-report/
9. Meisner, D., Gold, B.T., Wenisch, T.F.: PowerNap: eliminating server idle power. In: ACM Sigplan Notices, pp. 205–216 (2009)

10. Bhamare, D., Jain, R., Samaka, M., Erbad, A.: A suevey on service function chaining. J. Netw. Comput. Appl. **75**, 138–155 (2016)
11. Moens, H., De Turck, F.: VNF-P: A model for efficient placement of virtualized network functions. In: Network and Service Management (CNSM). In: 2014 10th International Conference on, pp. 418–423 (2014)
12. Clayman, S., Maini, E., Galis, A., Manzalini, A., Mazzocca, N.: The dynamic placement of virtual network functions. In: Network Operations and Management Symposium (NOMS), pp 1–9 (2014)
13. Bari, M.F., Chowdhury, S.R., Ahmed, R., Boutaba, R.: On Orchestrating Virtual Network Functions. In: International Federation for Information Processing (2015)
14. Cohen, R., Lewin-Eytan, L., Naor, J., Raz, D.: Near Optimal Placement of Virtual Network Functions. In: Computer Communications (INFOCOMM), 2015 IEEE Conference on pp. 1346–1354 (2015)
15. Bala, T.: Dynamic service chaining with SDN. In: Cloud Evolution Blog, Ericsson (2014)
16. Huawei white paper, Enabling Agile Service Chaining with Service Based Routing. http://www.huawei.com/ilink/en/download/HW_308622
17. Chen, Q., Grosso, P., van der Veldt, K., De Laat, C., Hofman, R., Bal, H.: Profiling energy consumption of VMs for green cloud computing. In: Dependable, Autonomic and Secure Computing (DASC), pp. 768–775 (2011)
18. Dell PowerEdge R620 Server data sheet. http://www.dell.com/downloads/global/products/pedge/en/Dell-PowerEdge-R620-750W-E5-2620-40-Family-Data-Sheet.pdf
19. Cisco ASA 5525-X IPS. http://www.cisco.com/c/en/us/support/security/asa-5525-x-adaptive-security-appliance/model.html
20. Cisco ASAv30 Firewall. http://www.cisco.com/c/en/us/products/collateral/security/adaptive-security-virtual-appliance-asav/datasheet-c78-733399.html
21. Cisco S680 Proxy. http://www.cisco.com/c/en/us/support/security/web-security-appliance-s680/model.html
22. Martins, J., Ahmed, M., Raiciu, C., Olteanu, V., Honda, M., Bifulco, R., Huici, F.: ClickOS and the Art of Network Function Virtualization. In: Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation. USENIX association, pp. 459–473 (2014)
23. Rao, A., Legout, A., Lim, Y., Towsley, D., Barakat, C., Dabbous, W.: Network chracteristics of video streaming traffic. In: CONEXT (2011)

Sanghyeok Kim is a Master candidate of Computer Science and Engineering Department at Sogang University, Korea. He received his B.S. degree in computer science and engineering from Sogang University, Korea in 2017. His research interests include cloud computing, distributed file and storage.

Sungyoung Park is a professor in the Department of Computer Science and Engineering at Sogang University, Seoul, Korea. He received his B.S. degree in computer science from Sogang University, and both the M.S. and Ph.D. degrees in computer science from Syracuse University. From 1987 to 1992, he worked for LG Electronics, Korea, as a research engineer. From 1998 to 1999, he was a research scientist at Telcordia Technologies (formerly Bellcore), where he developed network management software for optical switches. His research interests include cloud computing and systems, virtualization technologies, autonomic computing, and embedded system software.

Youngjae Kim received the B.S. degree in computer science from the Sogang University, Korea in 2001, the M.S. degree from KAIST, South Korea in 2003, and the Ph.D. degree in computer science and engineering from the Pennsylvania State University in 2009. He joined Sogang University as a faculty member in the department of computer science and engineering in 2016. Before joining Sogang, he was a faculty member in the department of software and computer engineering at Ajou University, South Korea in 2015-2016. Prior to joining Ajou University, he was a research staff member in the Oak Ridge National Laboratory in 2009–2015. His research interests include distributed file and storage, parallel I/O, operating systems, emerging storage technologies, and performance evaluation.



Siri Kim is an engineer in SK Telecom, Seoul, Korea. She received her B.S. and M.S. degrees in computer science and engineering from Sogang University, Seoul, Korea. Her research interests include cloud computing, software-defined network, and open source cloud programming.

Kwonyong Lee is an engineer in New Computing Lab, Corporate R&D Center, SK Telecom, Seoul, Korea. He received his B.S., M.S., and Ph.D. degrees in computer science and engineering from Sogang University, Seoul, Korea. His research interests include virtualization, cloud computing, software-defined infrastructure, all-flash scale-out storage, and AI computing.