Energy-Efficient Provisioning for Service Function Chains to Support Delay-Sensitive Applications in Network Function Virtualization

Gang Sun, Run Zhou, Jian Sun, Hongfang Yu, Athanasios V. Vasilakos

Abstract—**The efficient deployment of virtual network functions (VNFs) for network service provisioning is key for achieving network function virtualization (NFV); however, most existing studies address only offline or one-off deployments of service function chains (SFCs) while neglecting the dynamic (i.e., online) deployment and expansion requirements. In particular, many methods of energy/resource cost reduction are achieved by merging VNFs. However, the energy waste and device wear for large-scale collections of servers (e.g., cloud networks and data centers) caused by sporadic request updating are ignored. To solve these problems, we propose an energy-aware routing and adaptive delayed shutdown (EAR-ADS) algorithm for dynamic SFC deployment, which includes the following features. 1) Energy-aware routing (EAR): By considering a practical deployment environment, a flexible solution is developed based on reusing open servers and selecting paths with the aims of balancing energy and resources and minimizing the total cost. 2) Adaptive delayed shutdown (ADS): The delayed shutdown time of the servers can be flexibly adjusted in accordance with the usage of each device in each time slot, thus eliminating the no-load wait time of the servers and frequent on/off switching. Therefore, EAR-ADS can achieve dual energy savings by both decreasing the number of open servers and reducing the idle/switching energy consumption of these servers. Simulation results show that EAR-ADS not only minimizes the cost of energy and resources but also achieves an excellent success rate and stability. Moreover, EAR-ADS is efficient compared with an improved Markov algorithm (SAMA), reducing the average deployment time by more than a factor of 40.**

Index Terms—**Energy efficient; Service function chain; Network function virtualization; Dynamic deployment**

I. INTRODUCTION

Recently, with the development of cloud computing and software-defined networking (SDN) technologies [1], network function virtualization (NFV) has become a popular research topic [2-7]. The emerging NFV paradigm aims to decouple the function from the physical network equipment. Instead, network functions are provided by related software running on the standard virtualized platform, i.e., industry-standard servers [8-12]. The NFV technique improves the performance of software-based solutions and has potential for commercial deployment in future networks [13-15]. Virtual network functions (VNFs) are the software implementations of network functions on industry-standard physical servers. VNFs can be linked together to form a service function chain (SFC) [16, 17] and provide necessary network services for traffic flow. The mapping of an SFC request requires finding several physical servers and physical paths that satisfy various constraints to the host and connect the VNFs. Previously, telecommunication service providers (TSPs) used middleware based on proprietary hardware or software devices to allocate network services. Although this approach has valuable advantages regarding function provisioning, it involves substantial capital expenditures (CAPEX) and operating expenditures (OPEX). In contrast, the application of NFV can effectively reduce costs and meet the changing needs of users in accordance with a given service level agreement (SLA).

A. *Motivation*

With the rapid development and popularization of information technology, the numbers of devices and pieces of communication equipment in data centers, such as those supporting cloud networks, have also increased rapidly. The energy consumed by information and communication technology (ICT) infrastructures currently accounts for more than 4% of the worldwide energy consumption, and this figure is expected to double in the next few years [18]. For large data centers, the number of servers can be as high as 200,000 [19]. Obviously, these servers have enormous energy requirements. Unfortunately, much of this energy is wasted by idle systems. As discussed in [20], operating with all physical nodes turned on can increase the carbon footprint and negatively affect the electricity cost of the system. Recently, studies have shown that an idle server consumes approximately 70% of the power consumed by the same server running at a full load [21]. Therefore, controlling the number of active nodes in the system is important for VNF placement. However, the controller should not frequently turn on/off arbitrary nodes, as turning nodes off and then bringing them back to normal operation can lead to wear, as shown in [22-24], which is detrimental to the lifetime of physical devices.

B. *Scope of this paper*

Currently, due to increasing market demand and energy/resource shortages, network operators (NOs) are eager to find ways to reduce the expenditure of time, resources and energy while improving their economic profit by scheduling new network services. Consequently, there has been an increase in research on energy conservation and resource utilization in academia and industry [25-31]. Unfortunately, there is no outstanding research result or effective solution for this joint problem.

Gang Sun, Run Zhou and Jian Sun are with Key Lab of Optical Fiber Sensing and Communications (Ministry of Education), University of Electronic Science and Technology of China, Chengdu, China (e-mail: gangsun@uestc.edu.cn; 2872994057@qq.com; sj@uestc.edu.cn).

Hongfang Yu is with Key Lab of Optical Fiber Sensing and Communications (Ministry of Education), University of Electronic Science and Technology of China, Chengdu, Chian; and is also with Peng Cheng Laboratory, Shenzhen, China (e-mail: yuhf@uestc.edu.cn).

Athanasios V. Vasilakos is with Department of Computer Science and Technology, Fuzhou University, China; and is also with Department of Computer Science, Electrical and Space Engineering, Lulea University of Technology, Sweden (e-mail: athanasios.vasilakos@ltu.se)

Additionally, 5G plays a very important role and has many uses in Internet of Things (IoT). The authors of [32-35] refer to the 5G services used in industrial networks and emphasize the importance of low delay. In [36-38], the authors focus on delay-sensitive IoT applications with the aim of improving their use. These studies show the importance of a low-latency network and the need for a delay-sensitive algorithm. However, while many studies and papers attempt to find a solution in the physical layer, few do so in the NFV area. Although some papers are concerned with the orchestration of VNF, they concentrate on resource or energy consumption and ignore the complexity and practicability of the algorithms. Thus, their algorithms require considerable time to converge, which is not suitable for delay-sensitive applications in 5G and IoT. In other words, it is urgently necessary to find a new algorithm that can orchestrate VNFs and allow the network to not only save energy and resources but also achieve realization in a short time.

Motivated by these facts, the scope of this paper is to design an EAR-ADS algorithm (including energy-aware routing and an adaptive delayed shutdown mechanism) for deploying the SFC to jointly optimize the cost, which consists of energy consumption, resource consumption, and wear and tear of physical equipment. More importantly, the algorithm proposed in this work is efficient and flexible, so it is particularly applicable to areas with high delay requirements, such as 5G and IoT.

C. *Contributions*

Given the aforementioned problems and information, the main contributions of this study are as follows:

- We study the online SFC deployment problem considering real service request scenarios and present the model for this problem.
- We propose an energy-aware routing algorithm to minimize server energy consumption while considering bandwidth consumption.
- We design an adaptive delayed shutdown mechanism for minimizing the wasted energy consumption generated by machine switching and no-load running.
- We conduct extensive simulations to verify the superiority of our proposed approaches.

The remainder of the paper is organized as follows. We review related work in Section II. In Section III, relevant network models and formulations are defined. The EAR-ADS is described in detail in Section IV. Simulation results are shown in Section V. We provide a discussion in Section VI and conclude the paper in Section VII.

II. Related Work

Interest in NFV was sparked by a 2012 white paper [39] by telecommunication operators that introduced VNFs running on commodity hardware. Several NFV management systems have since been designed. SIMPLE [40] implements an SDN-based policy enforcement layer for efficient middlebox-specific "traffic steering" in data centers. Marouen Mechtri et al. [41] proposed an SFC orchestration and evaluation framework for verifying and comparing deployment algorithms. Bhamare et al. [42] solved VNF placement issues by optimizing the intercloud business flow and response time. Huawei Huang et al. [43] determined a method of effectively dictating the deployment of an SFC based on request requirements, which was an enormous research challenge. Although some of the aforementioned studies addressed energy-related optimization with the consideration of delay bounds (e.g., the objective in [44, 45] is to minimize the number of latency-constrained CPUs), these solutions do not address consumption itself. Additionally, few studies have addressed the power-based VNF placement proposals of our case study. In [46], the authors aimed to minimize power consumption by allowing user delay requirements to be met using a genetic algorithm approach, while [47, 48] proposed heuristic algorithms that enabled power reduction improvements. The authors of [49] researched the problem of energy consumption in multidomain networks and did not take equipment wear into account. In contrast, this paper aims to decrease energy, time, resource consumption and equipment wear within a single-domain network. Moreover, [50-51] addressed the migration problem for active micro-servers that are needed in low-traffic conditions by turning off or suspending micro-servers to achieve energy savings. The authors of [52] tried to optimally place and schedule network flows in virtual network functions by formulating an ILP model while also reducing the total network power consumption by using the EACons algorithm. However, this paper did not consider machine wear, and it addressed only the offline version of the problem.

In the existing literature [47-52], there is almost no systematic research on the nonworking energy cost of servers caused by continuous updates and alterations to online SFCs. Thus, there is currently no effective way to solve this problem. In [53-55], the joint use of servers and physical links during SFC deployment was studied to make better use of network resources. The authors first calculated an appropriate path length and server selection using elastic adjustments in accordance with the network conditions and the requirements of each SFC. Then, an appropriate routing path length was calculated. It was determined whether extra server resources were to be used or resources were to be reused, and the necessary resources were provided by existing servers for each VNF in the SFC. Unfortunately, that research focused only on resource optimization and did not consider processing methods for idle servers or the energy savings problem regarding the arrival and revocation of online SFCs in a dynamic deployment environment. For example, in [53], the authors proposed a DMRT-SL algorithm to efficiently map the service requests in edge computing. It showed outstanding performance in terms of time delay but without considering the energy consumption. The authors of [54] studied the relation between the path length and the VM reuse factor. It proposed a DC-LaS algorithm to find a solution whose path length and reuse factor approximately met these expectations. However, it did not consider energy consumption, time efficiency or machine wear.

In [56], the authors showed that the joint problem of minimizing energy cost and flow consumption is an NP problem, designed SAMA to minimize energy and traffic costs, and discussed server operating costs. Although that study noted issues of energy waste and equipment wear due to no-load server operation and frequent server switching, it offered no real solution to these problems. More importantly, SAMA solved this problem with iterations based on the Markov approximation, and it uses sampling on state space to accelerate the convergence speed. However, in a complex network that has many nodes and many VNFs needing to be deployed, the state space will be astronomical, which will still lead to a very long convergence time. Additionally, if we add a node or a VNF to the map, the state space will grow exponentially. All of these factors make the convergence time of this algorithm too long and its efficiency too low, thus greatly increasing the time cost of deployment.

Therefore, in this paper, EAR-ADS is proposed to address these issues. An EAR strategy is established to ascertain link-tolerant hops to bypass as many network resources as possible using open nodes, which effectively reduces the number of open servers, the consumption of underlying bandwidth resources, and the time delay between service functions and data traffic. Additionally, regarding the energy loss due to SFC updates on the servers, we carefully design a set of ADS mechanisms that can be flexibly adjusted in accordance with the usage of each server, thus eliminating a large amount of energy waste caused by no-load waiting. The proposed method can also avoid machine wear caused by frequent server switching, thereby prolonging the server service life and reducing the replacement rate and purchase cost of equipment to a certain extent.

Table 1. Comparison of the research content and critical contributions of previous works

Algorithms	Machine wear	Resources	Time	Energy
EACons [52]	×	√	√	√
DMRT-SL [53]	×	√	√	×
DC-LaS [54]	×	√	×	×
SAMA [56]	√	√	×	√
Our approach	√	√	√	√

Table 1 shows the important aspects and critical technical contributions of previous NFV works. It also displays the difference between the research contents of this paper and others. (In this table, " √ " means that this algorithm takes this element into account, while "×" means that it does not.)

III. Problem Statement and Modeling

A. Problem statement

As shown in Figure 1, the NFV architecture is composed of three key elements: underlying hardware resources, the virtualization network function pool and NFV orchestration. These elements are defined in [4, 5] and are summarized as follows. Underlying hardware resources include computing hardware, storage and networks (composed of nodes and edges) that provide processing, storage and connectivity to VNFs. An NF is a functional block within a network infrastructure that has well-defined external interfaces and well-defined functional behavior [12, 13]. Therefore, a VNF is an implementation of an NF that is deployed for virtual resources such as a VM, which runs on underlying hardware. After instantiation, a VNF can be deployed on underlying hardware and realize specific network functions, such as DHCP, NAT and firewalls. We call the combination of VNFs the virtual network function pool. NFV orchestration provides the functionality required for the provisioning of VNFs and related operations, such as the configuration of VNFs and the infrastructure on which these functions run [15, 17, 57]. After orchestration, specific VNFs can be composed of an SFC and offer network services to users. All these key elements combine to form the NFV architecture [41]. Together, they offer the network service we use in an application layer as traditional SFC does but without requiring expensive network equipment.

With the rapid development of networks and various related devices, the annual energy consumption of network devices has become quite high, especially for cloud networks and data centers. However, from a statistical perspective, much of the energy consumption of this equipment is due to no-load operation and frequent switching, which is in direct opposition to the global aim of energy conservation and emissions reduction [20, 21, 27]. Such conditions also increase the switching times and greatly shorten the life of the physical device. OPNET can be used to find a VNF placement that minimizes the total cost incurred by the system, which is represented in the objective function as the sum of the operational and traffic costs. However, the problem above cannot be solved in polynomial time because it is NP-hard [42].

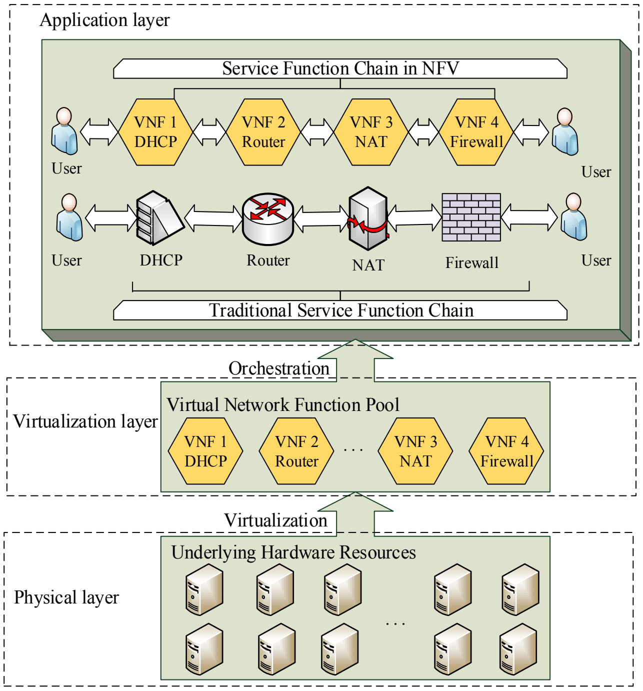


Fig. 1. Network function virtualization architecture.

Therefore, this paper proposes EAR-ADS, which addresses two main challenges: 1) how to balance energy and resource consumption, which requires minimizing energy consumption subject to existing resource constraints, and 2) how to realize adaptive shutdowns to avoid the long-term no-load operation and frequent on/off switching of machines, which involves minimizing unnecessary energy consumption and time delays. EAR addresses mainly the first challenge through resource virtualization and VNF orchestration; ADS is designed to solve the second challenge, which works on the physical layer.

B. Network model

(1) Substrate network: A substrate network consists of the underlying nodes that are directly connected via physical edges between the nodes. Each physical node has a set of service functions with resource attributes, and every physical edge has a corresponding bandwidth capacity. We represent the underlying physical network model as an undirected weighted graph $N_G = (V_G, E_G, R_G)$. Here, $V_G = \{v_1, v_2, \dots, v_{|V_G|}\}$ and $E_G = \{e_1, e_2, \dots, e_{|E_G|}\}$ represent the sets of nodes and edges, respectively, in the underlying network. $|V_G|$ and $|E_G|$ denote the numbers of nodes and edges, respectively, in the network. Every v_i or e_i offers certain node or edge resources. R_G represents the total resources provided by the physical network, including the node resources R_V (e.g., storage and computing) and the edge resources R_E. Our routing algorithm used in this paper adapts to undirected graphs and directed strongly connected graphs. In a real network, a weakly connected graph has no meaning in use, so this algorithm fits both directed and undirected graphs.

(2) SFC requests: An SFC request typically consists of one source, one destination, and multiple virtual nodes interconnected by virtual links. Every virtual node or virtual link has a specific resource demand. $SR = (S_F, L_F, R_F, s, t)$ represents a service request (SR) corresponding to an SFC, consisting of a series of VNFs connected by virtual links. Each VNF represents a certain network function (e.g., NAT, switch, firewall, IDS, WAN optimization, DHCP server or proxy). $S_F = \{f_1, f_2, \dots, f_{|S_F|}\}$ represents the series of VNFs corresponding to the SR, with a certain connection order between the VNFs, and *s* and *t* denote the source point and destination point, respectively, of the SR. We define $L_F = \{l_1, l_2, \dots, l_{|L_F|}\}$ as the set of virtual links between pairs of adjacent VNFs and $|L_F|$ as the number of links in the SR. In addition, different VNFs may represent different functions with different resource demands. R_F denotes the total resource requirement (including the node resource requirement R_S and the link resource requirement R_L) of the SR. A set of many SRs may simultaneously exist in the system.

(3) SFC mapping: The process of mapping SFCs to a physical network is called SFC mapping. SFC mapping can be successful only when the underlying network has sufficient node and edge resources. In this paper, we use $p_s = (V_G^p, E_G^p)$ to represent the SFC deployment plan. V_G^p represents the set of all underlying nodes involved in the deployment plan, which consists of the deployment node set and the forwarding node set. E_G^p is the set of all deployment edges involved in the deployment plan. To clearly explain our research problem, we assume the following:

- VNFs of different types for different SRs can be deployed on the same node when the resources are sufficient.
- A node can simultaneously serve as a deployment point and a forwarding point.
- A virtual link can be deployed on many coterminous edges of a physical network, which means that forwarding nodes exist in the network.

C. Energy-aware routing and adaptive delayed shutdown

Our goal is to minimize the total cost of dynamic SFC deployment. Compared with other algorithms, our algorithm demonstrates an outstanding effect in terms of energy savings and resource usage. Next, we will introduce EAR-ADS in detail. For convenience, the important notations used in describing the algorithm are listed in Table 2 for reference.

Table 2 Key notations used in this paper

Notation	Definition
$e_{on}^{v_i}, e_{off}^{v_i}$	Energy consumptions of server node v_i per unit time when being turned on and off
$m_{on}^{v_i}, m_{off}^{v_i}$	Numbers of times that v_i is turned on and off
$t_{on}^{v_i}, t_{off}^{v_i}$	Startup/shutdown delay for v_i per unit time
κ_{v_i}	Total number of times that v_i is switched on or off
$\mathfrak{I}_{V_i}$	Wear caused by a single instance of switching v_i on or off
$t_{empty}^{v_i}$	Total no-load time of v_i
$e_{empty}^{v_i}$	No-load energy consumption of v_i per unit time
U_{v_i}	Set of VNFs running on v_i in the current time slot
VNF_j	j^{th} VNF running on v_i
$t_{VNFs}^{v_i}$	Total time for which v_i serves as a deployment node
$t_{forward}^{v_i}$	Total time for which v_i serves as a forwarding node
$t_{work}^{v_i}$	Total working time of v_i
E_{work}^{vi}	Total working energy consumption of v_i
$e_{VNF\_j}^{v_i}$	Energy consumption for the j^{th} VNF deployed on v_i per unit time
$e_{base}^{v_i}$	Base energy consumption of v_i
$t_{VNF\_j}^{v_i}$	Working time for the j^{th} VNF deployed on v_i
$r_{f_i}^{SR_j}$	Node resource demand of VNF f_i of the j^{th} SR
$r_{v_i}^{total}$	Total resources of v_i
$r_{v_i}^{use}$	Used resources of v_i
$r_{v_i}^{remain}$	Remaining resources of v_i
$r_{l_i}^{SR_j}$	Bandwidth resource demand of virtual link l_i of the j^{th} SR
$\psi_{v_i}^{f_i}$	A binary variable indicating whether f_i is deployed on v_i
$r_{e_i}^{total}$	Total resources of e_i
$r_{e_i}^{use}$	Used resources of e_i
$r_{e_i}^{remain}$	Remaining resources of e_i
$\psi_{e_i}^{l_i}$	A binary variable indicating whether l_i is mapped to e_i

(1) Energy-aware routing

For online deployment problems, the arrival times, source and destination locations and VNF types all vary over time, and the usage of the underlying topological resources is also constantly changing. These characteristics pose enormous challenges for SFC deployment, especially with regard to energy conservation, as deployment involves choosing when to reuse an open server, when to choose the shortest path, when to shut down a device and when to maintain a no-load state. Because of the complexity of this problem, no outstanding research results or effective solutions have been reported to date. In this paper, the problem of energy conservation in SFC deployment is studied, and corresponding solutions are presented. At present, because of the wide usage of cloud networks and big data, the energy conservation problem is significant, with extensive prospects for application.

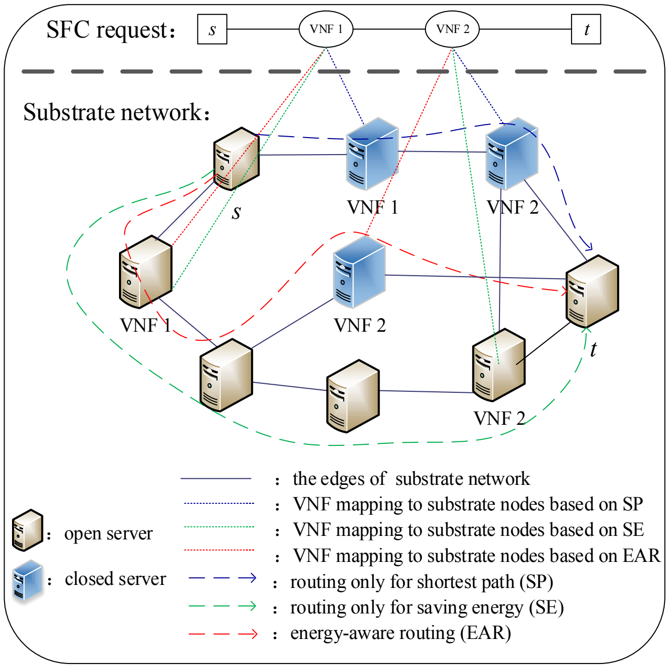

SFC request：
s
VNF 1
VNF 2
t
Substrate network：
s
VNF 1
VNF 2
VNF 1
VNF 2
t
VNF 2
：the edges of substrate network
：VNF mapping to substrate nodes based on SP
：VNF mapping to substrate nodes based on SE
：VNF mapping to substrate nodes based on EAR
：routing only for shortest path (SP)
：routing only for saving energy (SE)
：energy-aware routing (EAR)
：open server
：closed server

Fig. 2. Routes formed based on three strategies.

Figure 2 shows the routes formed based on three different strategies:

- The shortest path algorithm. This method, which is commonly used by researchers, finds the deployment path with the fewest hops; however because it does not consider the energy cost, it may result in the need to restart a large number of shut down servers, even when there are many open network nodes with sufficient resources. Consequently, this strategy results in considerable energy waste and equipment wear. As indicated by the blue path in Figure 2, the shortest possible path has a path length of 3, and the number of reused servers is 0.
- Solely energy-savings-based route selection. Although this method makes full use of the open servers, it can result in more efficient routes being bypassed because it will reuse open servers to the greatest possible extent, resulting in high overhead in terms of bandwidth resources. This strategy is highly undesirable when the number of SRs is large and the underlying resources are very limited. As shown by the green path in Figure 2, the path with the maximum energy savings has a path length of 5, and the number of reused servers is 4.
- The EAR policy. This algorithm considers both energy and bandwidth costs, and more importantly, it includes a tunable parameter κ that can be adjusted by the operator in real time in accordance with the current number of SRs and topological resource usage. Emphasizing the flexibility of the tradeoff between energy and bandwidth resources results in a deployment strategy that is agile and applicable and reduces the cost to the greatest extent. As shown by the red path in Figure 2, the EAR path length is 4, and the number of reused servers is 2. If there are multiple paths with the same number of hops, the algorithm will preferentially select an open node (e.g., VNF1); however, when reusing an open server will require a long detour, with a number of hops exceeding the value of parameter κ, that open server will be abandoned, and a new node will be selected to avoid overuse of the underlying resources (e.g., VNF2).

(2) Adaptive delayed shutdown mechanism

In the interval between the departure of the last VNF and the arrival of the next VNF, the state of the server is a decisive factor in determining the wasted energy consumption. There is little research on this topic at present, and it has considered only two possible states:

- Keep open: The server will be kept open even if the underlying node is in the no-load state. This will cause considerable no-load energy waste, especially in large-scale networks such as cloud networks and data centers.
- Shut down immediately: Once an SFC finishes running in the network, the corresponding node will be shut down immediately. This will result in the frequent on/off switching of servers, which will not only cause switching energy consumption and delay but also increase the wear on the machines, thus decreasing their lifetimes.

To solve these problems, this paper proposes an ADS mechanism: 1) For each open server, an initial delayed shutdown time $c_{delay}^{v_i} = \zeta$ is established ($c_{delay}^{v_i}$ denotes the delayed shutdown time of v_i in the current time slot, while ζ represents the initial value of the delayed shutdown time). 2) If the node is in the working state, then $c_{delay}^{v_i} = \zeta$ is maintained. 3) Otherwise, in each consecutive service time slot in which the node is in the no-load state, the delayed shutdown time is reduced to $c_{delay}^{v_i} = c_{delay}^{v_i} - 1$, and the server is shut down when $c_{delay}^{v_i} = 0$. 4) If the node accepts a new VNF or acts as a forwarding node during its no-load waiting time, then $c_{delay}^{v_i}$ is restored to its initial value $c_{delay}^{v_i} = \zeta$. Notably, for the purpose of decentralization, ζ can be adjusted for each node separately. This means that the initial value for every node can be determined in accordance with its use frequency, making this energy-saving system flexible, intelligent and effective.

Thus, it is evident that the algorithm proposed in this paper can not only avoid the machine wear and energy waste caused by long-term no-load operation and frequent on/off switching but also minimize the overall cost.

D. Cost definitions

We aim to minimize the overall cost to the network service provider throughout the entire process of deploying all SRs in N_G, considering the following costs.

(1) Energy consumption for switching servers on and off, $\boldsymbol{E_{switch}}$: Intuitively, frequent startup and shutdown of servers can result in a large amount of power consumption. When turning a server on, each of the server's components (e.g., CPU, memory, fans, and disks) needs to be started, and the power consumption may be several times that during normal operation; a similar mechanism applies for shutting down a server and its components. We use $e_{on}^{v_i}$ and $e_{off}^{v_i}$ to represent the energy consumption of v_i per unit time when turned on and off, respectively. $m_{on}^{v_i}$ and $m_{off}^{v_i}$ denote the numbers of times that v_i is turned on and off, respectively. The overall switching energy cost for the whole process in N_G is

$$E_{switch} = \sum_{v_i \in V_G} (m_{on}^{v_i} e_{on}^{v_i} + m_{off}^{v_i} e_{off}^{v_i}) \qquad (1)$$

(2) Startup/shutdown delay, T_{switch}: There is a period of time between pressing the startup/shutdown button and reaching the real operating/shutdown state; we call this time the startup/shutdown delay. We use $t_{on}^{v_i}$ and $t_{off}^{v_i}$ to represent the individual one-time startup and shutdown delays, respectively, for node v_i. The overall switching time delay for the whole process in N_G is

$$T_{switch} = \sum_{v_i \in V_G} (m_{on}^{v_i} t_{on}^{v_i} + m_{off}^{v_i} t_{off}^{v_i}) \qquad (2)$$

(3) Machine wear due to switching, $\Im$: It is obvious that frequently switching the machines on and off will cause great harm to the servers, shorten their service lifetimes and increase the equipment cost. Here, we use κ_{v_i} ($\kappa_{v_i} = m_{on}^{v_i} + m_{off}^{v_i}$) to represent the total number of times that server v_i is switched on or off and $\Im_{v_i}$ to denote the wear caused by a single instance of switching v_i on or off. The overall machine wear due to switching in the whole process in N_G is

$$\Im = \sum_{v_i \in V_G} \kappa_{v_i} \Im_{v_i},\ \kappa_{v_i} = m_{on}^{v_i} + m_{off}^{v_i} \qquad (3)$$

(4) No-load wait time, T_{empty}: Between the completion of the last VNF and the arrival of the next VNF, a physical node may be in a no-load state but remain active; we call the duration of this state the no-load wait time. $t_{empty}^{v_i}$ denotes the total no-load wait time of v_i. The total no-load wait time for the whole process in N_G is

$$T_{empty} = \sum_{v_i \in V_G} t_{empty}^{v_i} \qquad (4)$$

(5) No-load energy consumption, E_{empty}: Similarly, an open server generates a no-load energy consumption during its idle wait time. $e_{empty}^{v_i}$ represents the no-load energy consumption of v_i per unit time. The no-load energy consumption for the whole process in N_G is

$$E_{empty} = \sum_{v_i \in V_G} e_{empty}^{v_i} t_{empty}^{v_i} \qquad (5)$$

(6) Working time consumption, T_{work} : As mentioned before, any node v_i may have one or more VNFs deployed on it or may serve only as a forwarding node. First, we define U_{v_i} as the set of all VNFs running on v_i at a given time and $VNF\_j \in U_{v_i}$ as the j^{th} VNF running on v_i. Together, $t_{VNFs}^{v_i}$ (the total time for which v_i serves as a deployment node) and $t_{forward}^{v_i}$ (the total time for which v_i serves as a forwarding node) compose the total working time $t_{work}^{v_i}$ of v_i for all SRs. The working time consumption for all SRs in N_G is

$$T_{work} = \sum_{v_i \in V_G} t_{work}^{v_i} \qquad (5)$$

$$t_{VNF\_j}^{v_i} \le t_{work}^{v_i},\ t_{forward}^{v_i} \le t_{work}^{v_i}$$

Note that when U_{v_i}=null, v_i is used only as a forwarding node during that time.

(7) Working energy consumption, E_{work}: Similarly, when node v_i is used for the deployment/forwarding of SFCs, we call the corresponding energy cost the working energy consumption. E_{work}^{vi} denotes the total energy consumption of v_i in the working state, and $e_{VNF\_j}^{v_i}$ denotes the energy consumption for the j^{th} VNF deployed on $\mathrm{v_i}$ per unit time. Additionally, we use $t_{VNF\_j}^{v_i}$ to denote the working time for the j^{th} VNF on v_i. The working energy consumption for all SRs on v_i is

$$E_{work}^{v_i} = e_{base}^{v_i} t_{work}^{v_i} + \sum_{VNF\_j \in U_{v_i}} e_{VNF\_j}^{v_i} t_{VNF\_j}^{v_i} \qquad (6)$$

$$t_{VNF\_j}^{v_i} \le t_{work}^{v_i}$$

Note that if v_i serves only as a forwarding node for all SRs at all times, we have

$$E_{work}^{v_i} = e_{base}^{v_i} t_{work}^{v_i} \qquad (7)$$

Obviously, when the remaining resources are sufficient, running more VNFs on the same server will result in lower energy consumption because every open server has some base energy consumption. Thus, the working energy consumption for all SRs in N_G is

$$\begin{aligned} E_{work} &= \sum_{v_i \in V_G} E_{work}^{v_i} \\ &= \sum_{v_i \in V_G} \Big(e_{base}^{v_i} t_{work}^{v_i} + \sum_{VNF\_j \in U_{v_i}} e_{VNF\_j}^{v_i} t_{VNF\_j}^{v_i}\Big) \end{aligned} \qquad (8)$$

$$t_{VNF\_j}^{v_i} \le t_{work}^{v_i}$$

(8) Node resource consumption, $R_{V_G}^{use}$: When a VNF is deployed on the underlying nodes of the network, it will occupy some of the underlying node resources (e.g., computing resources and storage resources). $r_{f_i}^{SR_j}$ denotes the node resource demand of VNF f_i of the j^{th} SR, and $\psi_{v_i}^{f_i}$ is a binary variable indicating whether f_i is deployed on node v_i. $r_{v_i}^{total}$ represents the total resources of v_i, and $r_{v_i}^{use}$ represents the total node resource consumption of all VNFs deployed on v_i. After a VNF is successfully placed on v_i, the available node resources $r_{v_i}^{remain}$ will decrease accordingly:

$$r_{v_i}^{use} = \sum_{SR_j \in SRs} \sum_{f_i \in S_F} r_{f_i}^{SR_j} \psi_{v_i}^{f_i}$$

$$\begin{aligned} r_{v_i}^{remain} &= r_{v_i}^{total} - r_{v_i}^{use} \\ &= r_{v_i}^{total} - \sum_{SR_j \in SRs} \sum_{f_i \in S_F} r_{f_i}^{SR_j} \psi_{v_i}^{f_i} \end{aligned} \qquad (9)$$

$$\psi_{v_i}^{f_i} = \begin{cases} 1\text{ , if VNF } f_i \text{ is deployed on node } v_i \\ 0\text{ , otherwise} \end{cases}$$

Note that a VNF is deployed only when a node is open. Therefore, the overall node resource consumption for all SRs in N_G is

$$R_{V_G}^{use} = \sum_{v_i\in V_G}\sum_{SR_j\in SRs}\sum_{f_i\in S_F} r_{f_i}^{SR_j}\psi_{v_i}^{f_i} \tag{10}$$

(9) Bandwidth consumption, $R_{E_G}^{use}$: Similarly, mapping a virtual link to a physical edge will occupy network bandwidth resources. In the case of dynamic traffic requests and time-varying physical network conditions, the deployment paths often include one or more forwarding nodes. Unlike a deployment node for a VNF, a forwarding node does not consume resources, but it does consume energy. $r_{l_i}^{SR_j}$ denotes the bandwidth resource demand of virtual link l_i of the j^{th} SR, and $\psi_{e_i}^{l_i}$ is a binary variable indicating whether l_i is mapped to edge e_i. $r_{e_i}^{total}$ represents the total resources of e_i, and $r_{e_i}^{use}$ denotes the total bandwidth consumption on e_i. After a link is successfully mapped to e_i, the available bandwidth resources $r_{e_i}^{remain}$ will decrease accordingly:

$$\begin{aligned} r_{e_i}^{use} &= \sum_{SR_j\in SRs}\sum_{l_i\in L_F} r_{l_i}^{SR_j}\psi_{e_i}^{l_i} \\ r_{e_i}^{remain} &= r_{e_i}^{total} - r_{e_i}^{use} \\ &= r_{e_i}^{total} - \sum_{SR_j\in SRs}\sum_{l_i\in L_F} r_{l_i}^{SR_j}\psi_{e_i}^{l_i} \end{aligned} \tag{11}$$

$$\psi_{e_i}^{l_i} = \begin{cases} 1\text{ , if link } l_i \text{ is mapped on edge } e_i \\ 0\text{ , otherwise} \end{cases}$$

The overall node resource consumption for all SRs in N_G is

$$R_{E_G}^{use} = \sum_{e_i\in E_G}\sum_{SR_j\in SRs}\sum_{l_i\in L_F} r_{l_i}^{SR_j}\psi_{e_i}^{l_i} \tag{12}$$

In summary, the overall network resource consumption for all SRs in N_G is

$$R_{N_G}^{use} = R_{V_G}^{use} + R_{E_G}^{use} \tag{13}$$

Because no-load energy consumption and switching energy consumption are not necessary types of consumption for an SFC, we refer to them as wasted energy:

$$\begin{aligned} E_{waste} &= E_{switch} + E_{empty} \\ &= \sum_{v_i\in V_G}(m_{on}^{v_i}e_{on}^{v_i} + m_{off}^{v_i}e_{off}^{v_i}) + \sum_{v_i\in V_G} e_{empty}^{v_i}t_{empty}^{v_i} \\ &= \sum_{v_i\in V_G}(m_{on}^{v_i}e_{on}^{v_i} + m_{off}^{v_i}e_{off}^{v_i} + e_{empty}^{v_i}t_{empty}^{v_i}) \end{aligned} \tag{14}$$

Similarly, the wasted node delay is calculated as follows:

$$\begin{aligned} T_{waste} &= T_{switch} + T_{empty} \\ &= \sum_{v_i\in V_G}(m_{on}^{v_i}t_{on}^{v_i} + m_{off}^{v_i}t_{off}^{v_i}) + \sum_{v_i\in V_G} t_{empty}^{v_i} \\ &= \sum_{v_i\in V_G}(m_{on}^{v_i}t_{on}^{v_i} + m_{off}^{v_i}t_{off}^{v_i} + t_{empty}^{v_i}) \end{aligned} \tag{15}$$

$$k_{v_i} = m_{on}^{v_i} + m_{off}^{v_i}$$

The total energy consumption is as follows:

$$E = E_{waste} + E_{work} = E_{switch} + E_{empty} + E_{work} \tag{16}$$

The total time delay is as follows:

$$T = T_{waste} + T_{work} = T_{switch} + T_{empty} + T_{work} \tag{17}$$

E. Optimization Objectives and Constraints

(1) Optimization objectives:

- EAR aims to balance working energy and resource consumption while minimizing their total cost. Thus, the optimization objective for EAR is

$$\min C_1(E_{work}, R_{N_G}^{use})$$

where

$$\begin{aligned} & C_1(E_{work}, R_{N_G}^{use}) \\ & = \gamma E_{work} + (1\text{-}\gamma)(R_{V_G}^{use} + R_{E_G}^{use}) \end{aligned} \tag{18}$$

Because node resource consumption is fixed for a certain number of SFCs, the optimization objective can be revised as follows:

$$\min C_1(E_{work}, R_{E_G}^{use})$$

where

$$\begin{aligned} & C_1(E_{work}, R_{E_G}^{use}) \\ & = \gamma E_{work} + (1-\gamma)\ R_{E_G}^{use} \\ & = \gamma \sum_{v_i\in V_G} E_{work}^{v_i} + (1-\gamma)\sum_{e_i\in E_G}\sum_{SR_j\in SRs}\sum_{l_i\in L_F} r_{l_i}^{SR_j}\times\psi_{e_i}^{l_i} \\ & = \gamma\left[\sum_{v_i\in V_G}(e_{base}^{v_i}t_{work}^{v_i} + \sum_{VNF\_j\in U_{v_i}} e_{VNF\_j}^{v_i}t_{VNF\_j}^{v_i})\right] \\ & +(1-\gamma)\sum_{e_i\in E_G}\sum_{SR_j\in SRs}\sum_{l_i\in L_F} r_{l_i}^{SR_j}\times\psi_{e_i}^{l_i} \end{aligned} \tag{19}$$

where γ is the parameter that controls the balance between energy and resource consumption. For example, when too many requests or bandwidth resources are lacking, we may decrease γ to place more emphasis on resource conservation; conversely, we may increase γ to focus on energy conservation.

- The ADS mechanism aims to minimize wasted energy, including no-load and switching energy consumption, throughout the whole process. As mentioned before, switching servers on and off will lead to three kinds of consumption, denoted by E_{switch}, T_{switch}, and $\Im$, but they are all increased due to the same operation; therefore, we use E_{switch} to represent all these costs in the optimization objective for convenience. Similarly, we use E_{empty} to represent both the no-load energy consumption and the no-load wait time in the optimization objective. Thus, the optimization objective for the ADS mechanism can be expressed as

$$\min C_2(E_{switch}, E_{empty})$$

where

$$C_2(E_{switch}, E_{empty}) = E_{waste} = E_{switch} + E_{empty}$$
$$= \sum_{v_i \in V_G} (m_{on}^{v_i} e_{on}^{v_i} + m_{off}^{v_i} e_{off}^{v_i}) + \sum_{v_i \in V_G} e_{empty}^{v_i} t_{empty}^{v_i} \tag{20}$$

Thus, the overall optimization objective proposed in this paper is

$$\min C\ (C_1, C_2)$$

where

$$\begin{aligned} & C\ (C_1, C_2) \\ & = \gamma E_{work} + (1-\gamma)\ R_{E_G}^{use} + E_{switch} + E_{empty} \\ & = \gamma E_{work} + E_{switch} + E_{empty} + (1-\gamma)\ R_{E_G}^{use} \\ & = \sum_{v_i \in V_G} \left[\gamma E_{work}^{v_i} + (m_{on}^{v_i} e_{on}^{v_i} + m_{off}^{v_i} e_{off}^{v_i}) + e_{empty}^{v_i} t_{empty}^{v_i} \right] \\ & + (1-\gamma) \sum_{e_i \in E_G} \sum_{SR_j \in SRs} \sum_{l_i \in L_F} r_{l_i}^{SR_j} \times \psi_{e_i}^{l_i} \end{aligned} \tag{21}$$

As shown in the last equation above, the total optimization objective has two components: one is the total energy consumption for all SRs, and the other is the total bandwidth consumption for all SRs. This paper aims to minimize the total cost of both and offer a minimal consumption plan to service providers based on the resources remaining in the network and the number of SFCs.

(2) Constraints:

To successfully deploy an SFC in a physical network, the relevant resource constraints must be satisfied in every time slot. Therefore, all SRs should satisfy

$$\sum_{SR_j \in SRs} \sum_{f_i \in S_F} r_{f_i}^{SR_j} \times \psi_{v_i}^{f_i} \le r_{v_i}^{total} \tag{22}$$

$$\sum_{v_i \in V_G} \sum_{SR_j \in SRs} \sum_{f_i \in S_F} r_{f_i}^{SR_j} \psi_{v_i}^{f_i} \le R_{V_G}^{total} \tag{23}$$

(22) and (23) are constraints on the node resources. (22) ensures that the resources required for the VNFs deployed on v_i will not exceed the resources of this node, and (23) ensures that the resource consumption of all SRs deployed in the network will not exceed the total resources supplied by all nodes.

$$\sum_{SR_j \in SRs} \sum_{l_i \in L_F} r_{l_i}^{SR_j} \times \psi_{e_i}^{l_i} \le r_{e_i}^{total} \tag{24}$$

$$\sum_{e_i \in E_G} \sum_{SR_j \in SRs} \sum_{l_i \in L_F} r_{l_i}^{SR_j} \psi_{e_i}^{l_i} \le R_{EG}^{total} \tag{25}$$

(24) and (25) are constraints on the bandwidth resources. (24) ensures that the resource usage on each link does not exceed its capacity, and (25) ensures that the total bandwidth consumption is no greater than the total bandwidth supplied.

$$\sum_{v_i \in V_G} \sum_{SR_j \in SRs} \sum_{f_i \in S_F} r_{f_i}^{SR_j} \psi_{v_i}^{f_i} + \sum_{e_i \in E_G} \sum_{SR_j \in SRs} \sum_{l_i \in L_F} r_{l_i}^{SR_j} \psi_{e_i}^{l_i} \le R_G \tag{26}$$

(26) ensures that the total resource demands of all SFCs do not exceed the total resources supplied.

$$\sum_{v_i \in V_G} \psi_{v_i}^{f_i} = 1, \forall f \in S_F, \forall SR \in SRs \tag{27}$$

(27) ensures that a VNF can be deployed on only one physical node.

$$\sum_{f_i \in S_F} \psi_{v_i}^{f_i} \le 1, \forall SR \in SRs, \forall v \in V_G \tag{28}$$

(28) ensures that at most one VNF from a given SR can be deployed on a single physical node v_i.

IV. ALGORITHM DESIGN

EAR is an optimization method that balances the energy and bandwidth resource costs for online deployment, as shown in Algorithm 1. We wish to reduce the number of servers switched on as much as possible under a given hop constraint. For this purpose, the current node usage states in the network need to be detected before SFC deployment. If a node has already been turned on, it is added to the open set, V_G^{on}; otherwise, it is added to the off set, V_G^{off} (lines 4-10 of Algorithm 1). In addition, different subprocedures are called for VNFs in different positions (lines 11-17 of Algorithm 1). Finally, the complete SFC deployment scheme is obtained.

***Algorithm* 1:** Energy-aware routing for dynamic SFC deployment (EAR)

```
Input: (1) Physical network $N_G = (V_G, E_G, R_G)$;
       (2) Service requests $SR = (S_F, L_F, R_F, s, t)$;
Output: SR deployment scheme $p_s$.
1: Initialize network topology $N_G$
2: Set $p_s$ to null
3: for all arriving SRs, do
4:     Check the current physical network state
5:     for all $v_i$, $v_i \in V_G$, do
6:         if $v_i$ is on, then
7:             Add $v_i$ to $V_G^{on}$
8:         else add $v_i$ to $V_G^{off}$
9:         end if
10:    end for
11:    for all VNFs $f_i$ of the SR, $f_i \in S_F$, do
12:        if $f_i$ is the first or an intermediate VNF of the SR, then
13:            Call Procedure 1
14:        else $f_i$ is the last VNF of the SR, then
15:            Call Procedure 2
16:        end if
17:    end for
18: end for
```

Procedure 1 updates deployment scheme p_s. The VNFs are deployed one by one for each SFC. Procedure 1 (DM-F) is the deployment method for the intermediate VNFs. We set the link hop constraint parameter κ to balance the energy consumption and bandwidth. To save energy and make full use of the underlying network resources, we first look for the most appropriate deployment node in V_G^{on} for each intermediate VNF (lines 3-8 of Procedure 1) by using the breadth-first-search (BFS) algorithm. d_{min} is the shortest distance between previous deployment location v_j and current deployment location v_i. If a path that reuses an open server satisfies $d_{min} > \kappa$, we will abandon that open server and select a new node to avoid the excessive waste of bandwidth resources (lines 9-11 of Procedure 1). Thus, we determine the shortest path and the most suitable deployment point v_i, and finally, we deploy f_i on v_i and add the corresponding path to p_s. In the worst case, if there is no such node v_i in V_G^{off}, then we terminate the attempt to deploy this SR and clear the current p_s (lines 12-14 of Procedure 1).

***Procedure* 1:** Deployment for an intermediate VNF (DM-F)

```
Input: (1) Physical network $N_G$;
       (2) Set of switched-on nodes $V_G^{on}$;
       (3) Set of switched-off nodes $V_G^{off}$;
       (4) Source (previous deployment point $v_j$);
Output: Updated $p_s$.
1: for all VNFs $f_i$ of SR, do
2: Source ← $v_j$, sink ← $v_i$, set $d_{min} \leftarrow \infty$
3:   if $V_G^{on} \neq \emptyset$
4: Find the $v_i$ at the fewest hops from $v_j$ in $V_G^{on}$ subject to the
         resource constraints and obtain $d_{min}$; then
5:       if $d_{min} \leq \kappa$, then
6:           Deploy $f_i$ on $v_i$ and modify $p_s$
7:           Continue to deploy the next $f_i$
8:       end if
9: else find the $v_i$ at the fewest hops from $v_j$ in $V_G^{off}$ subject to
       the resource constraints and obtain $d_{min}$; then
10:        Repeat steps 5-8
11:    end if
12:    if failed to find a $v_i$ and a shortest path in $V_G$, then
13:        Clear $p_s$ and exit SR deployment
14:    end if
15: end for
```

Unlike in DM-F, for the last VNF, we must simultaneously consider the previous VNF deployment location and the destination of the SR. Therefore, we present Procedure 2 (DL-F), which aims to find a suitable node v_i on the shortest route between v_j and t on which to deploy the last VNF (lines 3-10 of Procedure 2).

***Procedure* 2:** Deployment for the last VNF (DL-F)

```
Input: (1) Physical network $N_G$;
       (2) Set of switched-on nodes $V_G^{on}$;
       (3) Set of switched-off nodes $V_G^{off}$;
       (4) Source (previous deployment point $v_j$);
       (5) Destination $t$ of the SR;
Output: Updated $p_s$.
1: Source ← $v_j$, sink ← $t$, set $d_{min} \leftarrow \infty$
2: if $V_G^{on} \neq \emptyset$
3: Find the $v_i$ that corresponds to the fewest hops for both $v_j \rightarrow$
       $v_i$ and $v_i \rightarrow t$ in $V_G^{on}$ subject to the resource constraints;
       then
4:     Place $f_i$ on $v_i$ and modify $p_s$ ; return $p_s$
5:     if failed to find a $v_i$ in $V_G^{on}$, then
6: Find the $v_i$ that corresponds to the fewest hops for both $v_j \rightarrow$
           $v_i$ and $v_i \rightarrow t$ in $V_G^{off}$ under resource constraints
7:         Repeat Step 4
8:     end if
9: else return to Step 6
10:  end if
11:  if failed to find a $v_i$ in $V_G^{off}$, then
12:      Clear $p_s$ and exit SR deployment
13:  end if
```

This paper mainly studies the problem of the dynamic deployment of SFCs; therefore, the SFCs are assumed to arrive and be completed dynamically, which will lead to considerable energy wastage. Considering that each server is used differently, we present the ADS algorithm for each node. The ADS algorithm aims to minimize the server's no-load energy consumption and switching energy consumption between the dynamic arrival and completion of SFCs. $c_{delay}^{v_i}$ is the current delayed shutdown time of v_i, and ζ is the initial value of the delayed shutdown time. For example, when a VNF is deployed on a new node, we initialize $c_{delay}^{v_i} = \zeta$. In each time slot, if a node is in the working state (meaning that the node is used either as a deployment point or a forwarding point), we maintain $c_{delay}^{v_i} = \zeta$. If v_i is in the no-load state but is chosen for reuse as a deployment point or a forwarding point for a newly arrived VNF, the delayed shutdown time of v_i will be reset to $c_{delay}^{v_i} = \zeta$ (lines 3-12 of Algorithm 2). If node v_i is in the no-load state (serving neither as a deployment point nor as a forwarding point) but $c_{delay}^{v_i} \neq 0$, then the delayed shutdown time will be decremented to $c_{delay}^{v_i} - 1$; if the node is empty and $c_{delay}^{v_i} = 0$, v_i will be shut down (lines 14-17 of Algorithm 2).

***Algorithm* 2:** Adaptive delay shutdown (ADS)

```
Input: (1) All $p_s$ of the SRs;
       (2) $V_G$;
       (3) Initial delayed shutdown time $\zeta$;
Output: Modified delayed shutdown time $c_{delay}^{v_i}$ for $V_G$.
1: for all $p_s$
2:   for all $v_i$
3: if $v_i$ is a deployment node or forwarding node, then
4:       if $v_i$ is shut down
5:           Turn on $v_i$ and set $c_{delay}^{v_i} = \zeta$
6:       else $v_i$ is open
7:           if $v_i$ is in the no-load state
8:               Reset $c_{delay}^{v_i} = \zeta$
9:           else $v_i$ is in the working state, then
10:              Maintain $c_{delay}^{v_i} = \zeta$
11:          end if
12:      end if
13:      else
14:       $v_i$ is not a deployment node or forwarding node
15:         if $v_i$ is in the no-load state
```

```
            $c_{delay}^{v_i} = c_{delay}^{v_i} - 1$
            **when** $c_{delay}^{v_i} = 0$, turn off the node
          **else if** $v_i$ is shut down
            Leave it shut down
          **end if**
        **end if**
      **end for**
    **end for**
```

Now, we provide a brief complexity analysis for the matching EAR-ADS algorithm. As discussed in Section IV, every time we find the next node to deploy the VNF, we use the BFS algorithm to find the shortest path from the last node to all other nodes and choose the best path based on EAR-ADS. The complexity of BFS is $O(M+E)$, where M is the number of nodes in the graph and E is the number of edges. For one SR, we need to execute BFS C times for one SR with C VNFs needing to be deployed on average. If N SRs need to be deployed in all SRs, the complexity of EAR-ADS is $O(CN(M+E))$.

V. Simulation Results and Analysis

We conducted extensive simulations to evaluate the performance of the proposed algorithm. In this section, we first introduce the simulation environment and then describe several performance parameters used in our simulations. Finally, we describe our main simulation results.

A. Simulation environment

In our simulations, we first compared the performance of EAR-ADS with that of SAMA [56] implemented in Java 8 and then verified the influence of different values of κ and ζ in our algorithm. To illustrate the superiority of EAR-ADS in various networks, the Waxman 2 model from the Georgia Tech Internetwork Topology Models (GT-ITM) [58] was adopted to generate a random topology as the underlying network. We used a machine with an Intel Core i3-4170 CPU and 8 GB of RAM to run the algorithm and assumed that the online arrival of the SRs followed a Poisson process [59]. As in [60], we set the parameters as follows: the services were deployed based on a random network topology with 50 nodes; each underlying node or edge has certain available resources; and SFCs with traffic loads of 1000-9000 and lengths of 3-12 are generated randomly. One SFC could consist of different kinds of VNFs, and every VNF or virtual link is assigned a random resource demand.

B. Simulation results and analysis

At the end of this section, we present the simulation-based analysis of the algorithm's behavior for different parameter values (κ= 1, 3, and 5 and ζ = 1, 3, and 5). To ensure fairness, we selected the middle value for each parameter (κ= 3 and ζ = 3) for the performance comparison with the previously reported algorithm.

Figure 3 compares the total energy costs of EAR-ADS and SAMA as the number of VNFs per SFC increases when the online loads (N) of SFCs are 1000 and 10000 units, respectively. We can see that the total energy cost of EAR-ADS is much lower (approximately 20%) than that of SAMA because EAR-ADS takes full advantage of the resources provided by the open servers. Subject to the link resource and hop constraints, we will preferentially select a server that has already been started up. In addition, an ADS scheme is designed for each node, which eliminates energy waste and machine wear caused by long-term no-load operation and frequent on/off switching. As shown in the figures, the proposed dual energy-saving mechanism for the green deployment of SFCs has obvious benefits.

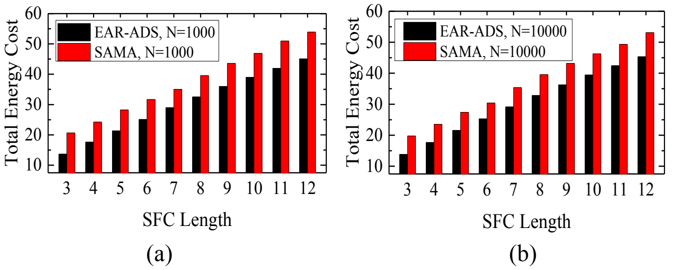


Fig. 3. Total energy cost per SFC for various SFC lengths. (a) Total energy cost *vs*. SFC lengths when the traffic load is 1000 units; (b) Total energy cost *vs*. SFC lengths when the traffic load is 10000 units.

Figure 4 compares the ratio of the total energy cost to the wasted energy for EAR-ADS and SAMA with an increasing load for SFC lengths of 3 and 12. Figure 4 (b) shows that EAR-ADS not only reduces the total energy cost but also minimizes the proportion of wasted energy (close to 0) by virtue of our ADS mechanism. When the SFC length is fixed, the deployment energy cost of EAR-ADS does not increase with an increasing load but instead remains relatively stable. This is because EAR-ADS uses operating servers as much as possible to minimize energy waste and dynamically adjusts its deployment decisions based on the real-time network situation. These results show that EAR-ADS is suitable for use in large-scale networks with high volumes of information requests and confirm the excellent practicality and application prospects of EAR-ADS compared with SAMA, which is highly unstable. The simulations illustrate that the EAR-ADS behaves well in high-traffic-load situations.

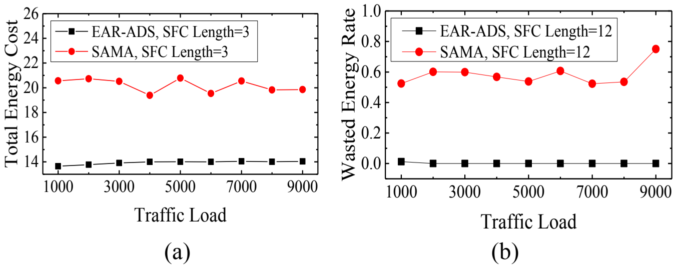


Fig. 4. Total energy costs per SFC under various traffic loads. (a) Total energy cost *vs*. traffic load; (b) Wasted energy rate *vs*. traffic load.

Figures 5 (a), (b), and (c) compare the bandwidth resource cost, blocking rates and average deployment times of EAR-ADS and SAMA for a representative intermediate traffic load (N=5000). It is evident that EAR-ADS is superior to SAMA in every respect. From Figure 5 (a), we can see that EAR-ADS has better behavior than SAMA in saving bandwidth for various SFC lengths. Although the bandwidth cost increases as the SFC length increases, it increases more slowly in EAR-ADS than in SAMA because EAR-ADS always finds the better route in all graphs, while SAMA finds it only in a sampling node set. Thus, SAMA has a strong possibility of ignoring the best route and obtaining a worse solution. Figure 5 (b) compares the blocking rates for the deployment of SFCs of several lengths. When the SFC length is short, the blocking rates of the two algorithms are not very different, but the gap becomes increasingly obvious as the SFC length increases, especially for SFC lengths greater than 6. As the SFC length grows to 12, the bandwidth consumption difference is almost 35%, and the total resource consumption of EAR-ADS is also 25% lower than that of SAMA; this explains why EAR-ADS has a blocking rate of only 40%, while that of SAMA is over 90%. These results show that EAR-ADS exhibits very stable performance and is suitable for use in high-frequency service situations. SAMA is a heuristic Markov algorithm; although it reduces the deployment time by reducing the convergence space, it does not fundamentally deviate from the Markov principle, and its repeated iterations and adjustments incur a high time cost.

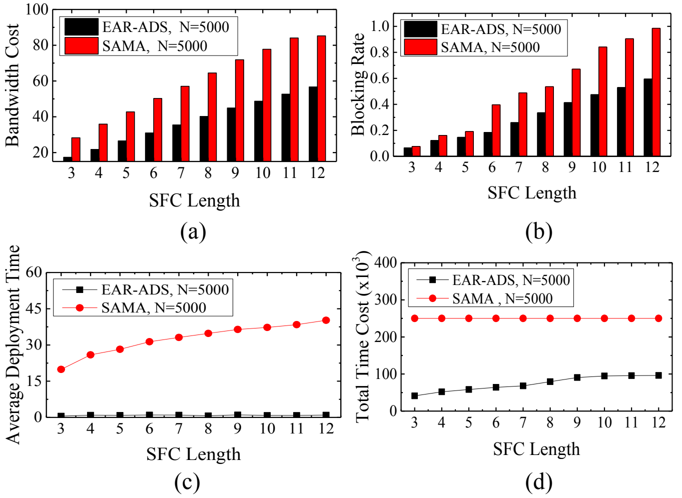


Fig. 5. Other costs and performance metrics for SFCs of various lengths when N=5000. (a) Bandwidth cost *vs*. SFC length; (b) Blocking rate *vs*. SFC length; (c) Average deployment time *vs*. SFC length; (d) Total time *vs*. SFC length.

Figure 5 (c) shows the average deployment time for SFCs of various lengths. The average deployment time for SFCs in EAR-ADS is approximately 0.8 ms. However, SAMA needs more than 35 ms and even gets longer with increasing SFC length. The deployment time of EAR-ADS is more than 40 times shorter than that of SAMA mainly because SAMA always finds the optimal solution in every iteration. However, using sampling on nodes to accelerate, SAMA ignores some nodes where VNFs can be deployed and ignores some better solutions, which may lead to higher consumption. This finding also verifies the conclusion that the EAR-ADS algorithm has lower complexity and is more efficient, which makes it very suitable in a delay-sensitive environment. Figure 5 (d) shows the comparison of the total time delay between EAR-ADS and SAMA when the traffic load is fixed to an intermediate value (N=5000). The total time means the total running time which leads to energy consumption of all servers, regardless of whether the state is working or no-load running. From Figure 5 (d), we can see that the total time of EAR-ADS is approximately 3-5 times lower than that of SAMA. Although the total time of EAR-ADS grows when SFC becomes longer, it always remains at a low level, while the total time of SAMA represents the upper limit of this value. Because of the delay-shutdown strategy used by EAR-ADS, the servers do not need to engage in no-load work for a long time, while SAMA let all servers keep working even in the no-load state.

Similarly, we select a representative intermediate SFC length of 6 to compare the costs and other performance metrics of EAR-ADS and SAMA for SFCs under various traffic loads in Figure 6. Figure 6 (a) shows the superiority of EAR-ADS regarding resource utilization. The average bandwidth cost of SAMA is 50, while that of EAR-ADS is only 31. There is a difference of approximately 20 units between the two algorithms in terms of bandwidth cost, and the performance of EAR-ADS shows almost no fluctuations as traffic load increases. This means that our algorithm can save a great deal of energy but does not sacrifice considerable resources to do so. In addition, Figure 6 (b) shows blocking rates under various traffic loads. When traffic is 1000 or 2000, the blocking rate of the two algorithms is close to the same; however, when the traffic load grows past 3000, the blocking rate of SAMA grows quickly. When the traffic load is equal to 9000 and the SFC length is 6, the blocking rate of SAMA is as high as 60%, while that of our algorithm is still lower than 20%. Figure 6 (c) shows the average deployment time under various traffic loads. As we discuss in Section II, EAR-ADS has lower complexity than SAMA because it considers only all nodes and edges in the graph, but SAMA must search the full state space, which is very large. Therefore, we can see that the average deployment time of EAR-ADS does not exhibit a large fluctuation when the traffic load becomes larger, but the performance of SAMA is poor in relation to its deployment time growth because when the traffic loads are higher, the state space of SAMA increases; consequently, the algorithm needs more time to find the better solution, and the performance of the new route worsens. These findings further illustrate that the EAR-ADS algorithm can behave well in high-traffic-load and delay-sensitive situations. Figure 6 (d) shows that SAMA's total time is much greater than that of EAR-ADS when the SFC length is fixed to an intermediate value of 6. Although both algorithms' total time grows when the traffic load increases, that of EAR-ADS grows slower than that of SAMA. The reason for this finding is explained in our discussion in the last paragraph.

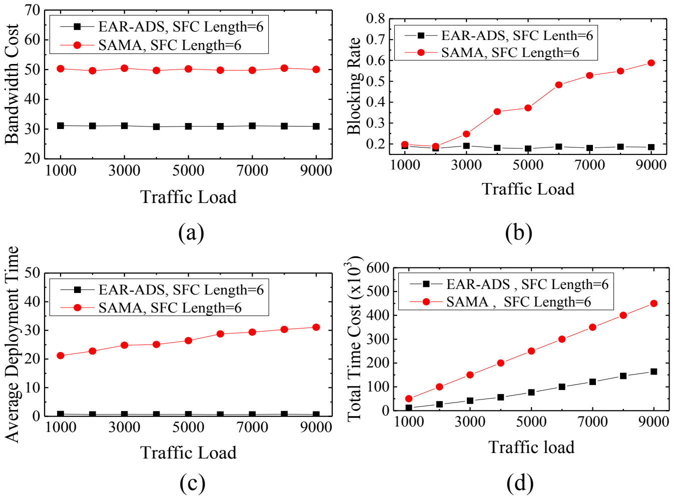
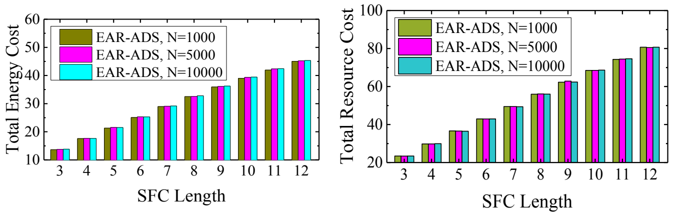


Fig. 6. Other costs and performance metrics under various traffic loads. (a) Bandwidth cost *vs*. traffic load; (b) Blocking rate *vs*. traffic load; (c) Average deployment time *vs*. traffic load; (d) Total time *vs*. traffic load.

Figure 7 shows the various costs and performance metrics for SFCs of various lengths for the proposed EAR-ADS algorithm under different traffic loads. For each SFC length, the results for total energy cost and total resource cost are almost the same for different values of *N*, thus showing that EAR-ADS achieves a low cost when addressing a large number of traffic requests.



(a) (b)

Fig. 7. Costs for SFCs of various lengths for the EAR-ADS algorithm under different traffic loads. (a) Energy cost *vs*. SFC length; (b) Resource cost *vs*. SFC length.

Figure 8 shows the wasted energy rates under various traffic loads for the EAR-ADS algorithm with different SFC lengths. As the traffic load increases, the waste rate decreases, which shows that the EAR-ADS algorithm not only effectively decreases the relative amount of energy wasted but also behaves even better as the SFC length increases.

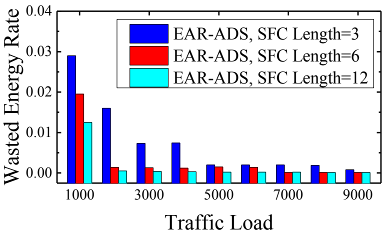
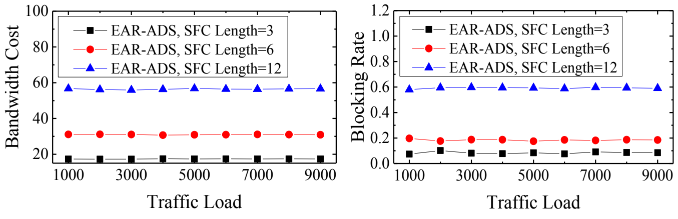


Fig. 8. Wasted energy rates under various traffic loads for the EAR-ADS algorithm with different SFC lengths.

Figure 9 shows that as the SFC length increases, the cost per SFC will also increase; this finding is intuitive because more VNFs will obviously require more node resources. However, in terms of the bandwidth cost and blocking rate, the behavior of EAR-ADS stabilizes as the traffic load increases. These findings further verify that EAR-ADS results in low costs and a low blocking rate when addressing a large number of traffic requests. Taken together, Figures 4, 6 and 7 show that EAR-ADS is stable under vast traffic loads.

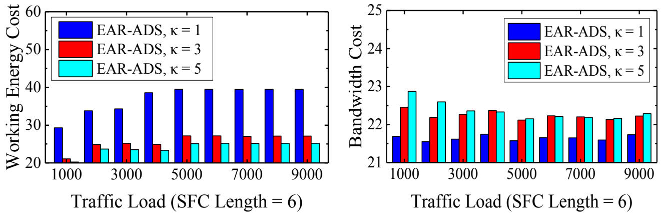


(a) (b)

Fig. 9. Costs and performance metrics under various traffic loads for the EAR-ADS algorithm with different SFC lengths. (a) Bandwidth cost *vs*. traffic load; (b) Blocking rate *vs*. traffic load.

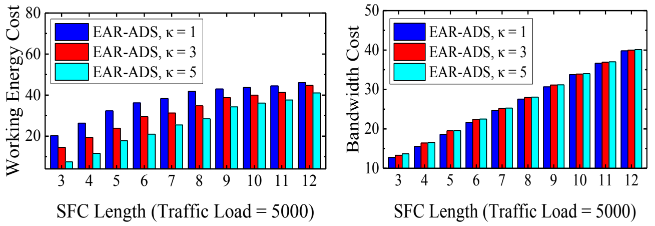
In Figures 10 and 11, we illustrate the changes in the working energy and bandwidth cost observed with different traffic loads and different SFC lengths. We tested 3 different κ values with ζ = 3.

Figure 10 shows that as we increase κ, the working energy cost markedly decreases for all 9 tested levels of traffic loads, while the bandwidth cost shows little change. This behavior occurs because as κ increases, we use more open servers to deploy new VNFs, whereas more machines will be newly started up when κ is lower. Thus, because the total base working energy cost will decrease as κ increases, the overall working energy cost will also decrease. However, because the algorithm will not choose a route that is too long, the bandwidth consumption will not increase considerably. Consequently, EAR-ADS can save a large amount of energy with little increase in the bandwidth load.



(a) (b)

Fig. 10. Working energy and bandwidth cost for SFCs under various traffic loads and with different values of κ. (a) Working energy cost *vs*. traffic load; (b) Bandwidth cost *vs*. traffic load.



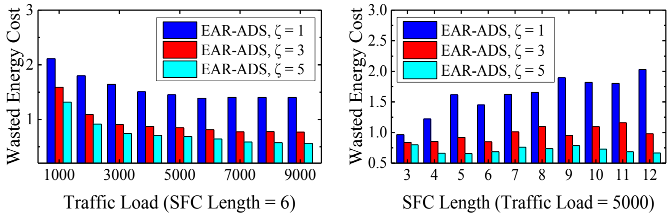
(a) (b)

Fig. 11. Working energy and bandwidth cost for SFCs of various lengths with different values of κ. (a) Working energy cost *vs*. SFC length; (b) Bandwidth cost *vs*. SFC length.

Figure 11 also shows the good behavior of EAR-ADS in terms of energy savings for SFCs of various lengths, while again, the bandwidth usage remains stable. The reasoning is the same as that for Figure 10; however, it is worth mentioning that because the graph is finite, as the SFC length increases, the distance between the two adjacent VNFs decreases. Thus, the improvement in energy cost achieved by the EAR-ADS algorithm becomes less conspicuous. However, in real-world situations, the topology will be very large; consequently, the algorithm will still greatly improve the energy cost.

Figure 12 shows how the energy wasted per SFC can be improved by choosing different ζ values. Here, we maintained a representative intermediate value of κ = 3 while varying ζ.

From Figure 12, we can see that at every level of traffic load and for every SFC length, as we increase the value of ζ, EAR-ADS greatly reduces the wasted energy cost because when ζ increases, the frequency with which all servers in the network are switched off decreases, and more servers will be reused as the EAR algorithm proceeds. This process leads to an enormous decrease in the switching energy cost. Meanwhile, because we do not allow the no-load running time to last too long, the total wasted energy cost is reduced. Consequently, the performance gap between different ζ values increases as the traffic load and the SFC length increase.



(a) (b)

Fig. 12. Wasted energy cost per SFC with different values of ζ. (a) Wasted energy cost *vs*. traffic load; (b) Wasted energy cost *vs*. SFC length.

VI. Discussion

This research focuses on energy savings in NFV practice and attempts to reduce resource consumption. In fact, the cost decrease in various aspects is a major challenge of NFV and has attracted much research in the academic community and industry. However, this is study only a small part of the NFV-based literature, and there are still many problems that need to be solved. In this section, we will discuss future NFV studies according to two elements: challenges and study directions.

A. Challenges faced in development

(1) Dynamic VNF deployment

Dynamic VNF deployment is a very important part of VNF provisioning, which may have a substantial impact. There will be many scenes that need to re-consider the VNFs we deployed and find new locations to place them to optimize resource consumption. For example, if some underlying machines fail or are under attack by an adversary, the network needs to detect them and quickly replace the virtual functions hosted by them. That will cause some new issues, for example, the services break off. Therefore, the replacement delay should be as short as possible to decrease the impact on QoS. Another issue is that new SFC routes may increase the bandwidth consumption, and the increase in latency will impact not only the services redeployed but also other services already deployed on these machines. Therefore, we also need to consider the load balance in VNF deployment. Therefore, dynamic VNF deployment is very important in real service operations, and it will face many challenges.

(2) Energy consumption reduction

Since energy bills have increased in recent decades, the reduction of energy consumption has become a hot topic in NFV-related research. Some studies, including this paper, have addressed it, but there is still not enough work on the topic. Currently, TSPs own data centers and offer services depending on them, which requires a large amount of energy every day. It is intuitive to close some of these centers that are unnecessary to run with no load. However, if the number of services increases abruptly when there is not enough underlying hardware available, it will take a great deal of time to launch these services; this will cost high service latency and seriously impact the QoS. Therefore, service volume detection and strategies based on it will be an effective method to realize energy consumption reduction. We also need to add this element into account when we design a dynamic VNF deployment plan. Thus, energy consumption reduction will be a big challenge but also a big opportunity to extend the existing research.

(3) Resources, function and service modeling

Network services, functions and resources need to be modeled and translated in order to deploy resource requirements, configurations, and management policies. This will greatly simplify the resource and service representations, which will significantly help academic research and industry optimization. Clear expression of the network and services will also enable network softwarization and realize configuring automation. However, the challenge is that it is very difficult to translate high-level policies into a lower-level configuration. Resource deployment algorithms and optimization mechanisms are difficult to transform into a simple model, which can lead the network to configure automatically. However, these tools the only way to realize network modeling in NFV practice, and there is a long way to go in future studies.

(4) Distributed network control

When network size, events and request volume increase, a single controller in a traditional network structure becomes unable to process all traffic and deploy all requirements. In fact, this will create a bottleneck in network operations. To solve this problem, we have to set more controllers to control different network areas and address service requirements in parallel. However, for the relativity of a network, which means that any plan in one area will influence the other areas, each controller has to have a global view of the whole network. Therefore, it is necessary to share the network conditions between controllers. Thus, all distributed systems will face the problem of consistency. For example, how does an underlying machine process the different configurations from different controllers? This may require some algorithms to guarantee consistency. Therefore, distributed network control is another challenge in NFV research.

(5) Security

For network operation, security should never be ignored. Security policies of each function, such as filtering traffic that meets specific rules or deep inspection of certain flows, should be considered when provisioning an SFC. The main challenge is how these functions offer different secure services to different demands. For example, for some flows that come from reliable sources, they may need to apply only firewall filtering rules; however, for traffic from suspicious sources, the flows should pass through all security functions. If we use only one strategy for all of the flows, it may be insecure or too sensitive. To do this, the interaction between different functions that are needed for providing network security should be examined. Another challenge in security is how to protect replacement of VNFs or rerouting traffic flow in case of an attack. It is difficult to find optimal places to deploy virtual security functions.

B. Potential future research directions

Indeed, NFV has the potential to grow significantly, which actually changes traditional proprietary network appliances and realizes non-proprietary and open-standard-based network and service deployments. This will eventually change the fabric of the telecommunications system. However, to obtain the benefits of an open-standard network, NFV needs to support intelligent programmability and network automation. There will be at least three main research directions of NFV in the future: network deployment strategy, network operation automation, and the integration of NFV with other technologies.

(1) Network deployment strategy

Network deployment is not a new problem only for NFV. However, after abstract network functions to blocks, the deployment strategy becomes a hot direction in NFV research. Many studies have focused on this problem; however, it is still insufficient. For different elements (e.g., bandwidth and computing resources, energy, and routing delays) and different business scenes (e.g., access point migration and IoT), the optimal strategy still needs to be sought to adapt to different demands, and network deployment will still be a main research direction in the future.

(2) Network operation automation

As we discussed before, SFC orchestration and deployment should be dynamic, which will lead to many VNF operations, such as installing and unloading on the underlying hardware. If we still operate and configure the network function manually, it will take considerable time and deeply impact the QoS. The high standard service of NFV requires the realization of network operation automation, which may require some new technologies in both algorithms and programming. As we discussed in the previous section, network modeling may help us reach this aim more quickly. Regardless, network operation automation must be the main direction of future research.

(3) Integration of NFV with other technologies

Another important research direction for NFV is the integration of NFV with other technologies. Recently, the integration of NFV with other technologies, such as SDN, IoT and 5G, has been a hot direction for both the academic research community and industry. However, different technologies have different rules and standards, and some of them are still underway. How to immigrate and utilize NFV in these technologies has great research significance. Furthermore, the dynamic and multidomain nature of future networks will face a large challenge in security and privacy protection. Nevertheless, the integration of NFV with other technologies will create huge economic benefits and offer a better solution to technology implementation and updating. Therefore, this will certainly be a new direction of future NFV research.

VII. Conclusion and Expectations

A. Summary of the data and performance

In this paper, we researched energy-efficient and flow-aware strategies for online SFC deployment. The key parameters of the proposed EAR-ADS method can be set and adjusted in real time in accordance with the network conditions and the characteristics of the SFC requests, allowing the number of open servers and the nonworking energy consumption to be simultaneously minimized. In particular, considering the traffic requirements and bandwidth consumption of SFCs, we propose an energy-aware routing strategy that seeks link-tolerant hops while using open nodes (i.e., physical servers) as much as possible. This algorithm will not negatively impact the QoS and SLAs expected by network operators. It can reduce the cost of the underlying bandwidth resources and shorten the time delay between SRs when offering services. To address the energy loss caused by SFC updates, we have carefully designed an adaptive delayed shutdown mechanism that can be flexibly adjusted in accordance with the service conditions of each server. This mechanism greatly reduces energy waste and server wear.

Simulation results show that with the EAR-ADS algorithm, the energy-saving effect is excellent without affecting other performance indices. The EAR-ADS algorithm greatly improves the performance regardless of how long the SFC length is or how large the traffic load is. This is because *i*) EAR-ADS allows nodes to run much less time than SAMA, which reduces the energy consumption by approximately 20% compared with SAMA. Against the current background of advocating for energy savings and emissions reduction, the method proposed in this paper has extensive application prospects and considerable research significance; *ii*) EAR-ADS greatly saves bandwidth resources. It reduces the bandwidth that SAMA consumes by approximately 35%, which means that a physical network with the same resource capacity can handle more requests simultaneously. Currently, the number of Internet requests has increased dramatically, and this technology is of great significance. *iii*) EAR-ADS keeps the blocking rate at a much lower level than SAMA. The blocking rate of EAR-ADS remains under 20% when the SFC length is 6 and is not higher than 60% as the SFC length grows to 12. Furthermore, it does not grow as the traffic load increases; *iv*) EAR-ADS greatly shortens the convergence time, which uses only 1/40 of the time that SAMA does. This property makes EAR-ADS very suitable for delaying sensitive applications in 5G or IoT environments. All these results show the superiority of our approach.

In addition to the numerical improvement, we will discuss more about the elements this paper takes into account and their meaning in real practice. EAR-ADS takes more elements into account than other algorithms, including machine wear, resources, time efficiency and energy consumption. Other algorithms do not take all these elements into account. This makes their solutions impractical for real applications. For example, machine wear is a very important aspect in practice because machine maintenance cost is a large element of operation costs. Frequent switching is the primary reason for machine wear. However, some algorithms (such as EACons) do not consider it. Moreover, energy consumption is the main point on which we focus in this paper. Many algorithms (such as DMRT-SL and DC-LaS) do not concern it, and in their deployment solutions, the machines always turn on and run even if there is no task on them. Obviously, these operations will cost considerable energy consumption, which means more energy waste and pollution. In fact, an algorithm that does not consider energy may not consider machine wear, either. Therefore, this scheme cannot be used in real operations. EAR-ADS takes energy into account and makes its deployment plan practical. Another important component is time efficiency, which means the average deployed time referred to in section V. In a real operation environment, after a service requirement is submitted, the time used to deploy the SFC and connect the VNFs is one of the most important metrics to measure the QoS. Most of the existing algorithms, such as SAMA, focus on energy and resource consumption and ignore the deployed time they use. Thus, these algorithms are impractical in use. Although we could hope that larger and faster machines will run this algorithm efficiently, when the number of requirements grows larger in peak flow, this may not be a smart idea. EAR-ADS can make a good deployment plan with high time efficiency, which proves its practicability in real applications.

B. Limitations of this study

Although this paper takes many elements into account, it still has some limitations that will affect its performance in use. EAR-ADS always finds the shortest paths to deploy SFCs as long as the paths have enough available bandwidth. Although routing through the shortest path can help to decrease bandwidth use, it also means the path the algorithm is close to the source point with high possibility. If a large amount of SFC requires a burst in a close range, this algorithm will deploy the SFCs in the same links, which will lead to network congestion. We will examine this point in future work and attempt to conduct some research on load balance for this problem.

C. Expectations

Recently, as 5G and IoT technologies grow quickly, the Internet of vehicles and mobile applications will also increase observably. Therefore, we should concentrate on the rapid changes in the access point. The quick movement of access points (source or target) will lead to an interruption of service or a waste of resources and energy. Therefore, seeking a new approach to solve these problems in a short time is imperative. Realizing the online migration of various VNFs, the seamless connection of services and resource redeployment to fit high-speed demand will remain major challenges in the future. Of course, in combination with various emerging technologies, the rapid, flexible and low-cost deployment of dynamic SFC with load balancing will be the subject of our future research work.

Acknowledgement

This research was partially supported by the National Key Research and Development Program of China (2019YFB1802802), PCL Future Greater-Bay Area Network Facilities for Large-scale Experiments and Applications (PCL2018KP001).

References

[1] S. Bera, S. Misra, A. Vasilakos. Software-defined networking for internet of things: A survey. IEEE Internet of Things Journal, 4(6), pp.1994-2008, 2017.

[2] B. Chatras, F.F. Ozog. Network Functions Virtualization: The Portability Challenge. IEEE Network, 30(4), pp. 4-8, 2016.

[3] L. Cui, F. Tso, S. Guo, et al. Enabling Heterogeneous Network Function Chaining. IEEE Transactions on Parallel and Distributed Systems, 30(4), pp. 842-854, 2019.

[4] R. Mijumbi, J. Serrat, J. Gorricho, et al. Network Function Virtualization: State-of-the-Art and Research Challenges. IEEE Communications Surveys & Tutorials, 18 (1), pp.236-262, 2016.

[5] G. Sun, Z. Chen, H. Yu, et al. Online Parallelized Service Function Chain Orchestration in Data Center Networks. IEEE Access, 7(1), pp.100147- 100161, 2019.

[6] A. Lamia, J. Intissar. SDN, slicing, and NFV paradigms for a smart home: A comprehensive survey. Transactions on Emerging Telecommunications Technologies, 30(8), 2019. DOI: 10.1002/ett.3744.

[7] K. Kaur, S. Garg, G. Kaddoum, et al. An energy-driven network function virtualization for multi-domain software defined networks. The First International Workshop on Intelligent Cloud Computing and Networking, pp.121-126, 2019.

[8] Z. Xu, W. Liang, M. Huang, et al. Efficient NFV-Enabled Multicasting in SDNs. IEEE Transactions on Communications, 67(3), pp. 2052-2070, 2019.

[9] S. Garg, K. Kaur, S. Ahmed, et al. MobQoS: mobility-aware and QoS-driven SDN framework for autonomous vehicles. IEEE Wireless Communications, 26(4), pp.12-20, 2019.

[10] G. Sun, D. Liao, D. Zhao, et al. Towards Provisioning Hybrid Virtual Networks in Federated Cloud Data Centers. Future Generation Computer Systems, 87, pp. 457-469, 2018.

[11] Y. Jia, C. Wu, Z. Li, et al. Online Scaling of NFV Service Chains Across Geo-Distributed Datacenters. IEEE/ACM Transactions on Networking, 26(2), pp. 699-710, 2018.

[12] C. Wang, X. Yuan, Y. Cui, et al. Toward secure outsourced middlebox services: Practices, challenges, and beyond. IEEE Network, 32(1), pp. 166-171, 2018.

[13] L. Linguaglossa, S. Lange, S. Pontarelli, et al. Survey of Performance Acceleration Techniques for Network Function Virtualization. Proceedings of the IEEE, 107(4), pp. 746-764, 2019.

[14] A. Rehman, R. Aguiar, J. Barraca. Network Functions Virtualization: The Long Road to Commercial Deployments. IEEE Access, 7, pp. 60439-60464, 2019.

[15] Z. Yan, P. Zhang, A. Vasilakos. A security and trust framework for virtualized networks and software defined networking. Security and communication networks, 9(16), pp.3059-3069, 2016.

[16] A. Medhat, T. Taleb, A. Elmangoush, et al. Service function chaining in next generation networks: state of the art and research challenges. IEEE Communications Magazine, 55(2), pp. 216-223, 2017.

[17] G. Sun, Z. Xu, H. Yu, et al. Toward SLAs Guaranteed Scalable VDC Provisioning in Cloud Data Centers. IEEE Access, 7(1), pp. 80219-80232, 2019.

[18] A. Barkat, M. Kechadi, G. Verticale, et al. Green Approach for Joint Management of Geo-Distributed Data Centers and Interconnection Networks. Journal of Network and Systems Management, 26(3), pp.723-754, 2018.

[19] P.X. Gao, A.R. Curtis, B. Wong, et al. It's not easy being green. in Proc. ACM SIGCOMM, pp. 211-222, 2012.

[20] M. Lin, A. Wierman, L.L. Andrew, et al. Dynamic right-sizing for power-proportional data centers. IEEE/ACM Transactions on Networking, 21(5), pp. 1378-1391,2013.

[21] A. Beloglazov, J. Abawajy, R. Buyya. Energy-aware Resource Allocation Heuristics for Efficient Management of Data Centers for Cloud Computing. Future Generation Computing Systems, 28(5), pp. 55-768, 2012.

[22] H. Qian, D. Medhi. Server operational cost optimization for cloud computing service providers over a time horizon. Usenix Conference on Hot Topics in Management of Internet, pp. 4-4, 2011.

[23] R. Shaw, E. Howley, E. Barrett. An energy efficient anti-correlated virtual machine placement algorithm using resource usage predictions. Simulation Modelling Practice and Theory, 93, pp. 322-342, 2019.

[24] N.H. Tran, D.H. Tran, S. Ren, et al. How geo-distributed data centers do demand response: A game theoretic approach. IEEE Transactions on Smart Grid, 7(2), pp. 937-947, 2016.

[25] Chiang, Y. Ju, Ouyang, et al. An Efficient Green Control Algorithm in Cloud Computing for Cost Optimization. IEEE Transactions on Cloud Computing, 3(2), pp. 145-155, 2015.

[26] D. Meisner, B.T. Gold, T.F. Wenisch. Power Nap: Eliminating server idle power. International Conference on Architectural Support for Programming Languages and Operating Systems, pp. 205-216, 2009.

[27] K. Kaur, S. Garg, G. Kaddoum, et al. A big data-enabled consolidated framework for energy efficient software defined data

centers in IoT setups. IEEE Transactions on Industrial Informatics, 2019. DOI 10.1109/TII.2019.2939573.

[28] B. Kaur, A. Kaur. An efficient approach for green cloud computing using genetic algorithm. International Conference on Next Generation Computing Technologies, pp. 10-15, 2015.

[29] S. Ren, M.A. Islam. Colocation demand response: Why do I turn off my servers. International Conference on Autonomic Computing, pp. 201-208, 2014.

[30] J. Wan, S. Tang, Z. Shu, et al. Software-defined industrial internet of things in the context of industry 4.0. IEEE Sensors Journal,16(20), pp.7373-7380, 2016.

[31] E. Tsiropoulou, P. Vamvakas, G. Katsinis, et al. Combined power and rate allocation in self-optimized multi-service two-tier femtocell networks. Computer Communications, 72, pp. 38-48, 2015.

[32] M. Khoshnevisan, V. Joseph, P. Gupta, et al. 5G industrial networks with CoMP for URLLC and time sensitive network architecture. IEEE Journal on Selected Areas in Communications, 37, pp. 947-959, 2019.

[33] J. García-Morales, M.C. Lucas-Estañ, J. Gozalvez. Latency-Sensitive 5G RAN Slicing for Industry 4.0. IEEE Access, 7, pp. 143139 - 143159, 2019.

[34] Y. Pointurier, N. Benzaoui, W. Lautenschlaeger, et al. End-to-End Time-Sensitive Optical Networking: Challenges and Solutions. Journal of Lightwave Technology, 37, pp. 1732-1741, 2019.

[35] S. Garg, K. Kaur, G. Kaddoum, et al. SDN based secure and privacy-preserving scheme for vehicular networks: a 5G perspective. IEEE Transactions on Vehicular Technology, 68(9), pp.8421-8434, 2019.

[36] D. Omer, R. Aburukba, T. Landolsi. Optimization Model for Time Sensitive IoT Requests. 2019 International Conference on Communications, Signal Processing, and their Applications (ICCSPA), pp. 1-4, 2019.

[37] I. Farris, A. Orsino, L. Militano, et al. Federations of connected things for delay-sensitive IoT services in 5G environments. 2017 IEEE International Conference on Communications (ICC), pp. 1-6, 2017.

[38] H. Mohseni, B. Eslamnour. Handover Management for Delay-sensitive IoT Services on Wireless Software-defined Network Platforms. 2019 3rd International Conference on Internet of Things and Applications (IoT), pp. 1-6, 2019.

[39] Network functions virtualization. White Paper, 2013. Available: https://portal.etsi.org/NFV/NFV_White_Paper2.pdf.

[40] S. Rajagopalan, D. Williams, H. Jamjoom, et al. Split/merge: System support for elastic execution in virtual middleboxes. The 10th Symposium on Networked Systems Design and Implementation, pp. 227-240, 2013.

[41] M. Mechtri, C. Ghribi, O. Soualah, et al. NFV Orchestration Framework Addressing SFC Challenges. IEEE Communications Magazine, 55(6), pp. 16-23, 2017.

[42] D. Bhamare, M. Samaka, A. Erbad, et al. Optimal virtual network function placement in multi-cloud service function chaining architecture. Computer Communications, 102, pp. 1-16, 2017.

[43] H. Huang, S. Guo, J. Wu, et al. Service chaining for hybrid network function. IEEE Transactions on Cloud Computing, pp. 1-13, 2017.

[44] B. Addis, D. Belabed, M. Bouet, et al. Virtual network functions placement and routing optimization. IEEE International Conference on Cloud Networking, pp. 171-177, 2015.

[45] H. Moens, F. De Turck. VNF-P: a model for efficient placement of virtualized network functions. International Conference on Network and Service Management, pp. 418-423, 2014.

[46] S. Kim, Y. Han, S. Park. An energy-aware service function chaining and reconfiguration algorithm in NFV. International Workshops on Foundations and Applications of Self-Systems, pp. 54-59, 2016.

[47] V. Eramo, A. Tosti, E. Miucci. Server resource dimensioning and routing of service function chain in NFV network architectures. Journal of Electrical and Computer Engineering, pp. 1-12, 2016.

[48] K. Hida, S.-I. Kuribayashi. Virtual routing function allocation method for minimizing total network power consumption. International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering, 10, pp. 997-1002, 2016.

[49] G. Sun, Y. Li, H. Yu, et al. Energy-efficient and traffic-aware service function chaining orchestration in multi-domain networks. Future Generation Computer Systems, 91, pp. 347-360, 2019.

[50] C. Pham, H.D. Tran, S. I. Moon, et al. A general and practical consolidation framework in Cloud NFV. IEEE International Conference on Information Networking, pp. 295-300, 2015.

[51] V. Eramo, M. Ammar, F. Lavacca. Migration energy aware reconfigurations of virtual network function instances in NFV architectures. IEEE Access, 5, pp. 4927-4938, 2017.

[52] N. Khoury, S. Ayoubi, C. Assi. Energy-aware placement and scheduling of network traffic lows with deadlines on virtual network functions. IEEE International Conference on Cloud Networking, pp. 89-94, 2016.

[53] G. Sun, Y. Li, Y. Li, et al. Low-latency orchestration for workflow-oriented service function chain in edge computing. Future Generation Computer Systems, 85, pp. 116-128, 2018.

[54] T. Kuo, B. Liou, K. Lin, et al. Deploying chains of virtual network functions: on the relation between link and server usage. IEEE/ACM Transactions on Networking, 26(4), pp.1562 -1576, 2018.

[55] G. Sun, G. Zhu, D. Liao, et al. Cost-efficient service function chain orchestration for low-latency applications in NFV networks. IEEE Systems Journal, 13(4): 3877-3888, 2019.

[56] C. Pham, N. Tran, S. Ren, et al. Traffic-aware and Energy-efficient VNF Placement for Service Chaining: Joint Sampling and Matching Approach. IEEE Transactions on Services Computing, 2018. DOI: 10.1109/TSC.2017.2671867.

[57] N. Saraiva, D. Lachos Perez, R. Rosa, et al. Network service orchestration: a survey. Computer Communications, 142, pp. 69-94, 2019.

[58] K. Calvert, E. Zegura. Gt-itm: Georgia tech internetwork topology models (Software). Georgia Tech. Available: http://www.cc.gatech.du/fac/Ellen.Zegura/gt-itm/gt-itm.tar.gz.

[59] G. Sun, D. Liao, D. Zhao, et al. Live Migration for Multiple Correlated Virtual Machines in Cloud-based Data Centers. IEEE Transactions on Services Computing, 11(2), pp. 279-291, 2018.

[60] J. Sun, G. Zhu, G. Sun, et al. A Reliability-Aware Approach for Resource Efficient Virtual Network Function Deployment. IEEE Access, 6, pp. 18238-18250, 2018.