Dynamic Resource Orchestration of Service Function Chaining in Network Function Virtualizations

Bangchao Yu[1,2,3(✉)], Wei Zheng[1,2,3], Xiangming Wen[1,2,3], Zhaoming Lu[1,2,3], Luhan Wang[1,2,3], and Lu Ma[1,2,3]

[1] Beijing Laboratory of Advanced Information Networks, Beijing, China
yubc0321@gmail.com, zhengweius@163.com, lzy_0372@163.com, wluhan@bupt.edu.cn
[2] Beijing Advanced Innovation Center for Future Internet Technology, Beijing, China
[3] Beijing Key Laboratory of Network System Architecture and Convergence, Beijing University of Posts and Telecommunications, Beijing, China

Abstract. Network Functions Virtualization is a new network architecture framework and is revolutionizing the way networking service that how to design and deploy. NFV promotes virtualizing network functions and improves the flexibility to resource orchestration for request service function chains. However, how to find the most suitable resource in NFV-based network resource is a challenge. This paper presents a comprehensive state of the NFV resource orchestration by introducing a dynamic resource orchestration architecture that can configure dynamic resources. With consideration of load balance, energy cost and resource consumption, the resource orchestration is formulated as a multi-objective optimal problem. Finally, a multi-objective simulated annealing algorithm is used to obtain the optimal resource strategy to deploy network functions. Simulation results show that the solution for dynamic resource orchestration can achieve approximate optimal solution in acceptable time and reduce 8% energy consumption with a 0.89 Jain's fairness index.

Keywords: NFV · Service function chains · Performance optimization Resource orchestration

1 Introduction

Network Function Virtualization (NFV) is revolutionizing the way networking service are designed and deployed, and is a promising technique to meet the requirements of explosive data traffic in fifth generation (5G) mobile communications [1]. Compared to traditional network where dedicated hardware is required for each function (manually installed into the network), the NFV virtualizes the network functions (NFs) (e.g., local cache, firewalls, load balancers, databases), which can be hosted on commodity hardware (i.e., industry standard servers, storage and switches). NFV transforms the way that how network operators architect their infrastructure by leveraging the full-blown virtualization technology to separate software instance from hardware

K. Long et al. (Eds.): 5GWN 2017, LNICST 211, pp. 132–145, 2018.
https://doi.org/10.1007/978-3-319-72823-0_14

platform and by decoupling functionally from location for faster networking service [2]. These functions decoupled from the underlying hardware are knows as Virtualized Network Functions (VNFs).

Under the paradigm of NFV, the networks provide considerably more flexibility, leads to efficient and scalable resource usage and reduce costs. Despite the emergence of NFV, deploying and orchestrating VNFs still requires more research and development. To facilitate the service management in NFV environments, service function chaining technique is proposed. A SFC is describes the various NFs and how they interact to provide a complete network service [3]. In order to offer various service, the operator needs to configure the networking infrastructure to direct the right traffic through the right service chain. Since the deployment and management of NFs providing the services usually require a significant capital investment, the operators are seeking ways to optimize the resource utilization of theses NFs. The key to exploiting the potential of the NFV based networks is the issue of resource orchestration in SFC.

Resource Orchestration (RO) is the set of operations that cloud providers and application owners undertake for selecting, deploying, monitoring, and dynamically controlling the configuration of hardware and software resources as a system of quality of service-assured components that can be seamlessly delivered to end users [4]. Orchestration can significantly reduce the cost and the time required for provisioning new services with respect to today's standards by allowing the joint allocation of different types of physical or virtual resource.

However, there are still many issues that need to be tackled in the resource orchestration of SFC. In [5], the authors study the network slicing based on 5G and future mobile network. One is to achieve dynamic variation of service resource requirements arise from a number of factors, including resource capacity demand, failures, end-user access patterns, and variations in resource prices. The other is configuring dynamic resources. The impetus behind NFV environment if the ever-increasing demand to manage growth and increase computing flexibility by dynamic scaling up or down resources based on demand [2]. Moreover, dynamic configuration of resources is a complex issue because of lack of visibility and control across heterogeneous services at different layers.

Recently, there have been many efforts on how to configure dynamic resources from both academia and industry. In [6], the authors study the problem of dynamic and programing resource orchestration in Mobile Cloud Computing (MCC) environment, and propose a multi-objective particle swarm optimization algorithm by considering resource choice for complicated service where multiple tasks are composed together. Another work in [7] studies NFV resource allocation problem, and proposed a coordinated approach Jointly Optimize the Resource Allocation in NFV (JoraNFV) to jointly optimization three phases in NFV resource allocation. While initial research results are promising, more than that, in many case designing effective dynamic resource orchestration techniques that cope with large-scale NFV environment remains a deeply challenging problem.

In this paper, we focus on the configuring dynamic resources orchestration in NFV environment. More specifically, we search for the optimal resource strategies and dynamically configure resource to meet requirements and constraints. To solve this problem, we propose a dynamic resource orchestration framework that contains a software defined architecture along with the multi-objectives optimization algorithm aiming to obtain the optimal resource configurations.

The rest of this paper is organized as follows: Some related works are presented in Sect. 2. After that, we present the dynamic resource orchestration architecture in Sect. 3. The algorithm to solve configuring dynamic resource problem is introduced in Sect. 4. Section 5, we report performance evaluation results. Section 6 concludes this paper.

2 Related Works

Over the last few years, some works have already been done to study resource orchestration in NFV environment.

Recent studies show that, in an enterprise network, the number of middle-boxes is comparable with the number of routers and switches needed to maintain the operation of the network [8]. Sherry et al. [9] presents a comprehensive state of the art of NFV-RA by introducing a novel classification of the main approaches that pose solutions to solve the resource allocation of demanded network services in NFV-based network infrastructure. He proposed three different stages of NFV-RA. In addition, NFV facilitates installation and deployment of VNFs on general purpose server [10, 11], thus allowing dynamic migration of VNFs form one server to another, that is, to any place of the network [12].

In [13], a two-layer home-to-home cloud framework which allows intra-home resources to be shared among consumer electronic devices and inter-home resources to be shared among neighboring houses is proposed to find the most suitable resources in the home cloud. But the optimal orchestration strategy in [13] fails to scale or down resources based on demand dynamically.

VNFs can be deployed and reassigned to share different physical and virtual resource of the infrastructure, so as to guarantee scalability and performance requirements. Tao et al. [14] work on the resource service composition in industry grid system, and exploit the standard particle swarm algorithm to find the solution. In [15], the authors propose two heuristic algorithms to optimize middle-box placement problem. One is greedy algorithm, and the other is simulated annealing algorithm. Their simulation results show that the Simulated Annealing algorithm outperforms the Greedy algorithm. However, the resource orchestration techniques mentioned above don't effectively support dynamic resource configuration. For example, service or dataflow can't be dynamically and automatically partitioned or migrated arbitrarily from one cloud service to another if demand cycles increase.

Therefore, the dynamic resource orchestration in the NFV environment should take the CPUs and energy consumption into consideration. In this work, a dynamic NFV-based resource orchestration architecture and a multi-object simulated annealing is proposed to find the appropriate solution for service to meet requirement and constraints.

3 System Architecture

3.1 NFV-Based Resource Orchestration Architecture

In this section we present the proposed integrated NFV Resource Orchestration and Management Architecture in order to deploy Virtual Network Functions (VNFs) on top

of an integrated cloud and network platform (i.e., NFV infrastructure (NFVI)). The NFVI is composed of heterogeneous multi-domain and multi-layer transport networks with heterogeneous transport and control technologies interconnecting distributed Databases Centers, providing computing, storage, and network resources. Today, cloud computing systems follow a service-driven, layered software architecture model, with Software as a service, Platform as a service, Infrastructure as a service. On top of this physical infrastructure, we deploy an NFVI virtualization layer responsible for virtualizing the computing, storage, and network resources of the NFVI by means of a virtualized infrastructure manager.

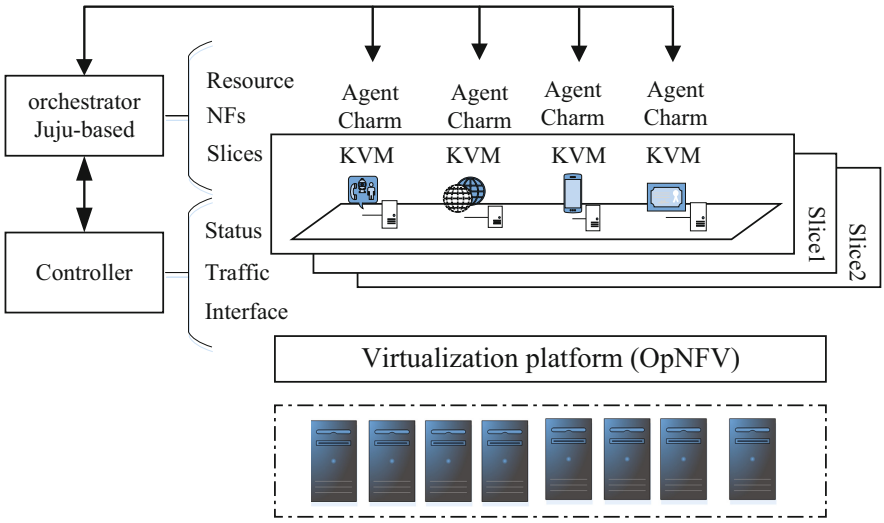

Fig. 1. NFV-based Resource Orchestration System

As shown in Fig. 1, the NFV based Resource Orchestration System is composed of Resource and Management Controller, VNFs (for example, database appliance, load balancer, auto scaling, virtualized HSS, virtualized MME, virtualized S-PGW) and the physical infrastructure. The Resource and Management Controller as well as Orchestration Controller is a centered controller for resource orchestration by interacting with distributed flexible Agents. The communication and monitoring functions for the resources are encapsulated in the Agents.

3.2 Resource Orchestration Controller

The Resource Management and Orchestration Controller is composed of three functional modules: monitor function, decision function and execution function, as show in Fig. 2(a). They are responsible for the set of operations that cloud providers and application owners undertake for selecting, deploying, monitoring, and dynamically controlling the configuration of hardware and software. The abstract resource and management orchestration operations in the lifecycle of an application or service are resource selection, deployment, monitoring and control, as shown in Fig. 2(b).

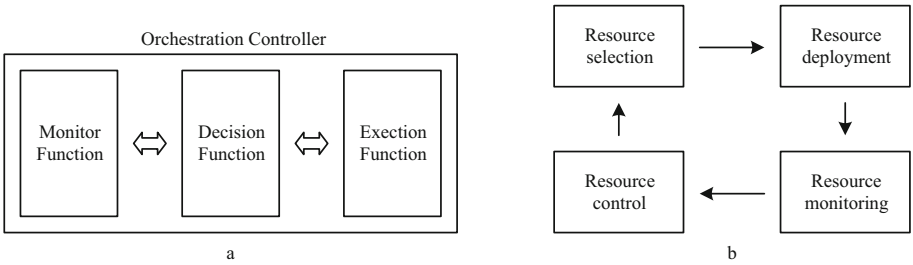

Fig. 2. Orchestration controller and operations

Monitor Function collects and aggregates the status and performance metrics of consumer electronic devices. By this function, the Orchestration Controller obtains the resource parameters such as CPU, storage and communication capability, or the Cache memory.

Decision Function receives the register message of connection, or the control message from Agent, and records the resource parameters. It also gathers the perceiving and monitoring information, and transmits the resource orchestration policy from orchestration controller to the selected VFs.

Execution Function is calculation module in the Resource Orchestration Controller, which takes use of the monitoring information and obtains the resource orchestration policy from the optimization problem. The service function can be abstracted to application with specific functions and parameters. Then according to resource requirements, this function executes the intelligent algorithm and provides the optimal resource orchestration strategy for a virtual function.

3.3 Interactive Procedure

The could computing resources may not stick to only one service or platform, so taking advantage of virtualization resource has become a key issue. Based on this, our goal is to achieve the automatic management of Virtual functions (VFs) and service orchestration, meeting the quality of service (QoS)-assured components that can be seamlessly delivered to end users. In Fig. 3, for collecting the resource status, Orchestration Controller subscribes the information for each Agent. When the user's behavior or the timer is triggered, Agent sends the information to the monitor function box in Orchestration Controller. When a service's request arrives at the Orchestration Controller, it finishes the virtualization resource orchestration (in the Decision Function box) and transmits the strategies to the Execution Function box.

When Agent receives the strategy, it performs the service management or deployment and executes the service. After that, it collects the result of its previous services, updates the results, deletes the records of itself and transmits the results to the

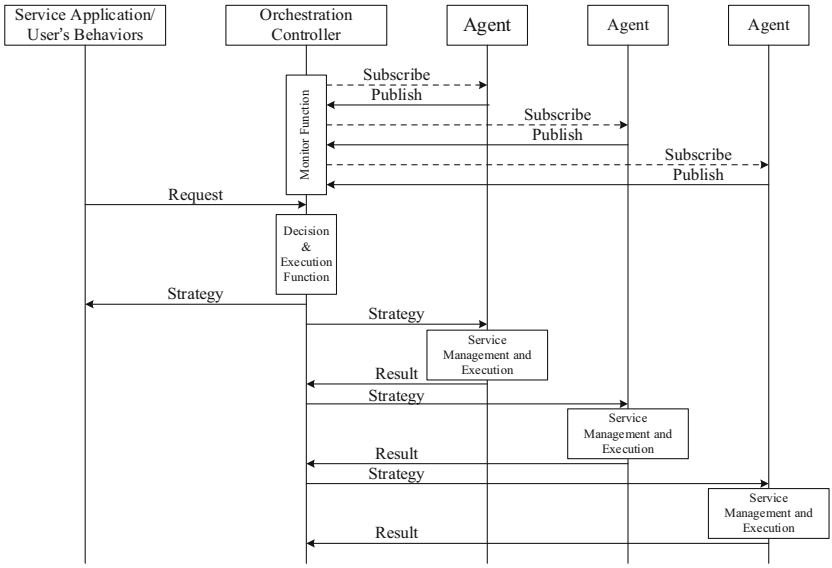

Fig. 3. Interactive procedure for resource orchestration

Orchestration Controller. The interaction message of Agents and Orchestration Controller are implemented by using Representational State Transfer (REST) Application Programming Interfaces (APIs) that is easy for extending.

4 Resource Orchestration Algorithm

4.1 Problem Definition

Figure 4 depicts the basic resource orchestration in NFV resource management. The left part is the service chaining which consist of function units (VFs) hosted on NFV nodes. By isolation, multiple VFs can run on the same physical could server, e.g. three units on Infrastructure-based resource area A and four units on Infrastructure-based resource area B. It should be noted than the CPU, storage, and network resources in A and B is different. On the right part of Fig. 4, there is mapping part from physical resources to virtual resources. The NFV resource are abstracted to services with specific functions and parameters. The VFs in a complicated service chaining is modeled as dataflow graph and denoted by Directed Acyclic Graph, depicted in Fig. 3. Due to the difference existing in the hardware resources, the appropriate resource is chosen for VFs according to the global optimization goal during user's behaviors. As the metrics such as resource cost, resource availability and CPU load average are not relevant or not following the same trend or even conflicting with each other, improving one metric may result in the deterioration of another one.

The multi-objective resource orchestration is to find a solution set Resource Orchestration Path (ROP_{SET}) in which each solution ROP_d has extreme optional value of aggregated metric, i.e.

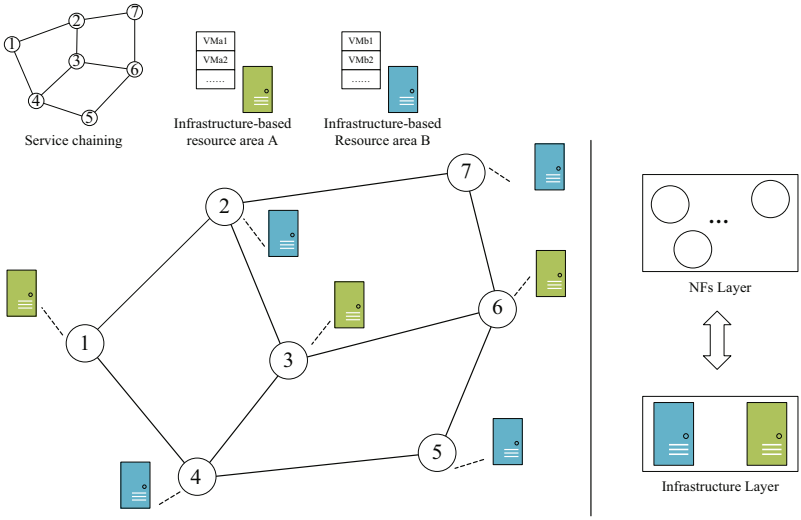

Fig. 4. Example of resource orchestration in NFV

$$\min F(ROP_d) = \alpha \cdot \min E(ROP_d) + \beta \cdot \min C(ROP_d) + \gamma \cdot \min L(ROP_d) \quad (1)$$

where $d = 1, \ldots, n$; $ROP_d \in ROP_{set}$, $\alpha + \beta + \gamma = 1$, E($ROP_d$), C($ROP_d$) and L($ROP_d$) are the energy consumption, resource charges and CPU load average of ROP_d respectively. F is the target vector function.

Energy consumption ($E_s^{r,i}$) means the total energy consumption of service i when VFs node is hosted on the resource r. It is determined by the amount of transmitted data. e_{trans}^r is the energy consumption of transmitting per unit (bit) data to resource r. V_{load}^i is the data volume of service i deploy to the NFV resource and $V_{interact}^i$ is interaction data volume between NFV resource and resource orchestration controller during the service i execution. e_{comput} is the energy consumption of computing per unit (bit) data and V_{comput}^i is the local data volume of the service i.

$$E_S^{r,i} = e_{trans}^r \cdot (V_{load}^i + V_{interact}^i) + e_{comput} \cdot V_{comput}^i \quad (2)$$

Resource charges ($C_S^{r,i}$) refers to the price pay for the specific NFV resource r where service i will be offload to. Let $c_{storage}^r$ be the storage charge of selected resource r, c_{comput}^r be the computing charge of selected resource r, and $c_{commun}^{r,c}$ be the communication cost from resource r to resource orchestrator controller when service i will be offloaded to. $V_{commun}^{r,c}$ is the data volume transmitted from resource r to resource orchestrator controller.

$$C_S^{r,i} = c_{storage}^r + c_{comput}^r + c_{commun}^{r,c} \cdot V_{commun}^{r,c} \quad (3)$$

CPU load average ($L_s^{r,i}$) represents the average system load over a period of time. An idle computer has a number of 0. Each process using or waiting for CPU increments the load number. For single-CPU systems that are CPU bound, one can think of load average as a percentage of system utilization during the respective time period. For system with multiple CPUs, one must divide the number by the number of processors in order to get a comparable percentage. l_A^i is the CPU load average of infrastructure-based resource area A when service i is hosted and n_A is the total number of service hosted on A.

$$L_S^{r,i} = \min[\sum_{i=0}^{n_A} l_A^i, \sum_{i=0}^{n_B} l_B^i, \cdots] \tag{4}$$

4.2 Resource Orchestration Algorithm

The resource orchestration is to select the optimal from all possible candidates considering multiple objectives (e.g. energy consumption, resource cost, resource availability), which is known as multi-objective optimization problem. We take use of the Multi-Objective Simulated Annealing (MOSA) algorithm to solve this problem, for it is a general random search framework that can get near-optimal solutions of combinational optimization problems. Besides, the advantage of MOSA is it can keep improving the solution effectively if time allows.

Algorithm: MOSA for the dynamic resource orchestration

Inputs: temperature T ; parameter λ; probability set p
Output: resource orchestration strategy ROP_d
// Initialization
$T(t) = \lambda T$, $t = t_0$, $p(\Delta F) = e^{-\Delta F/t}$, $IteNum = L \cdot |ROP_{set}|$, $i = 0$
while $i < IteNum$ **do**
  $ROP_e \leftarrow$ generate neighbor of ROP_d
  calculate $\Delta F = F(ROP_e) - F(ROP_d)$, $p(\Delta F)$
  if $\Delta F < 0$ **then**
    $ROP_d \leftarrow ROP_e$
    if $rand(0,1) < p(\Delta F)$ **then**
      $ROP_d \leftarrow ROP_e$ with probability p
    end
  end
  $i = i + 1$
  if $mod(i, L) == 0$ **then**
    $t \leftarrow$ update temperature t
  end
Return ROP_d

The MOSA algorithm for resource orchestration is stepped as follows:

5 Performance Evaluation

5.1 Simulation Scenarios

The implementation set-up consists of a test bed with seven VMs of desktop PCs performing as the NFV clouding computing resource. The VFs are all configured with 1 GB memory, CPU 3.2 GHz, gigabit Ethernet network interfaces, and Linux operating system, and are deployed with the Resource Controller for orchestration resources.

In the simulation scenarios, we generate specific service function chain that include different types services which hosted on NFV based resources. For each component service, the V^i_{load}, $V^i_{interact}$, V^i_{comput} and $V^{r,c}_{commun}$ are generated according to a uniform distribution. We generated serval resource providers to represent different NFV based resources with e^r_{trans}, e_{comput}, $c^r_{storage}$, c^r_{comput} and $c^{r,c}_{commun}$ follow a uniform distribution. The detailed setting is shown in Table 1. The resource parameters of service functions are generated while the service function is getting started. The available time T_{start} and T_{end} are set to 0 and 5 min, respectively. And we take an example of a service function chain that contains NFs and dataflow depicted in Fig. 5.

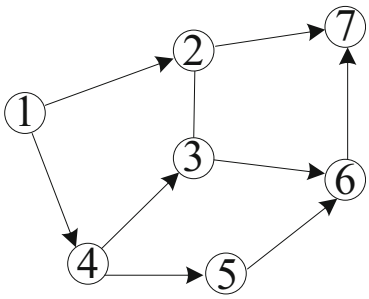

Fig. 5. Service dataflow in simulation scenarios

Table 1. Simulation parameters

NFV resource		Service		MOSA	
Parameters	Values	Parameters	Values	Parameters	Values
e^r_{trans}	[1, 5]	V^i_{load}	[10,80]	α	0.3
e_{comput}	3	$V^i_{interact}$	[10,50]	β	0.4
$c^r_{storage}$	[1, 10]	V^i_{comput}	[20,100]	λ	0.3
c^r_{comput}	[20,50]	$V^{r,c}_{commun}$	[5,30]	l^i_A	[0.1,0.7]
$c^{r,c}_{commun}$	[0,1]				

5.2 Simulation Results

In order to evaluate the performance of the proposed Resource Orchestration Architecture and the MOSA algorithm, we carried out the evaluation through a combination of testbed based experiments and numerical simulations. We use the testbed to evaluate the real service functions in small scale network with different physical infrastructures. In our evaluation, we compare the proposed algorithms against a baseline algorithm (Greedy Algorithm) and a basic resource orchestration (without any algorithm).

(1) *Energy evaluation*

To evaluate the performance of MOSA algorithm, we generate seven NFs that contains multiple instances hosted on different hardware resources in total. The physical hardware is different in computing resource, storage resources, etc. We started the resource orchestration with MOSA algorithm at 40 min when the virtual machine was deployed, for the installation and configuration of service functions.

As shown in the Fig. 6, we can observe that MOSA-based approach slightly outperforms GA-based, and both MOSA and GA-based approach outperform basic RO without any algorithm in the runtime of orchestration operations. There are mainly due to two reasons. Firstly, both MOSA and GA-based approach are taking the energy consumption into consideration before configuring resources. Second reason is the CPU load average of hardware or software resource are also considered in MOSA approach, which is related to the energy consumption. On average, the MOSA-based approach will reduce 8% energy consumption than basic RO and reduce 3% than GA-based algorithm.

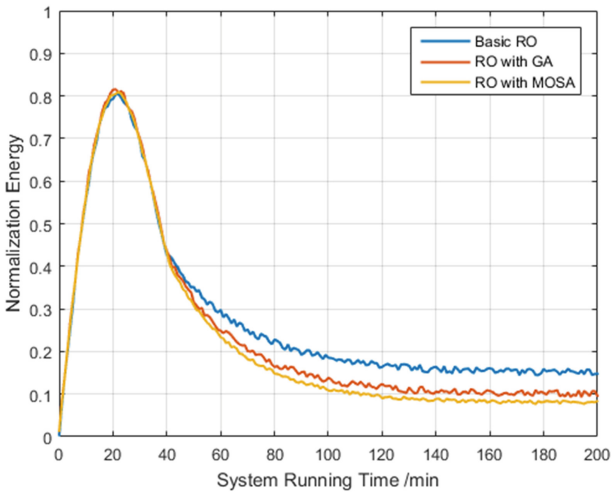

Fig. 6. Energy consumption of infrastructure

(2) *Load evaluation*

In the MOSA algorithm, we considered the CPU load average of the physical infrastructure or service functions. The Jain's Fairness Index indicates the fairness of different algorithm for resource invocation [16]. It is defined as follows:

$$f(x_1, x_2, x_3, \ldots, x_n) = \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n \sum_{i=1}^{n} x_i^2} \tag{5}$$

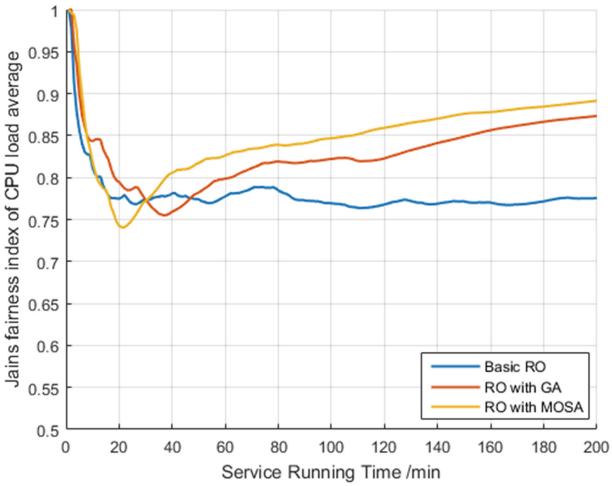

Fig. 7. Jain's fairness index of CPU load average

The Jain's Fairness Index was calculated for each of SFs and is shown in Fig. 7. It can be observed that the fairness is improved with the MOSA-based or GA-based approach compared to basic RO. The Jain's index can achieve about 0.89 with MOSA and 0.87 with GA, which is higher than basic RO with about 0.77. It shows that, considering CPU load average and configuring dynamic resources, MOSA approach perform consistently on different resources. The Jain's index of MOSA becomes better when the network scale becomes larger and larger, and more instances are deployed, which also make resources more fair.

(3) *Execution Time*

To evaluate the execution time, we use MOSA and GA-based approach to deploy service functions chain on different physical infrastructure. The request service chain contains 7 VNFs and the number of virtual nodes in network is set from 10 to 100 with an interval of 10. The simulation result is shown in Fig. 8. With the increase of the number of nodes, the execution time are increasing, but the MOSA and GA-based approach take less time than basic resource orchestration without any algorithm. The MOSA-based approach can hit optimal solutions of configure dynamic resource in acceptable time.

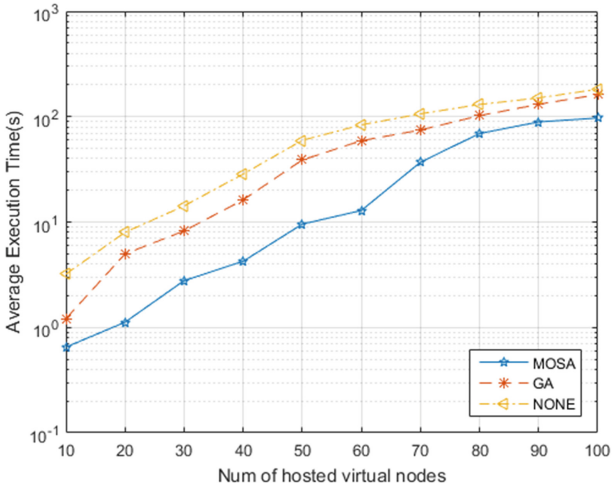

Fig. 8. Execution time performance of algorithms

(4) *Multi-object evaluation*

To facilitate understanding results, we use the three-dimensional figures to show the optimal solutions in resource orchestration. The result is shown in Fig. 9. It contains 30 instances of service functions for each algorithm. Pentagram represent the value of MOSA algorithm in the objective space. We can find that the MOSA can obtain more optimal results than GA algorithm and basic resource orchestration.

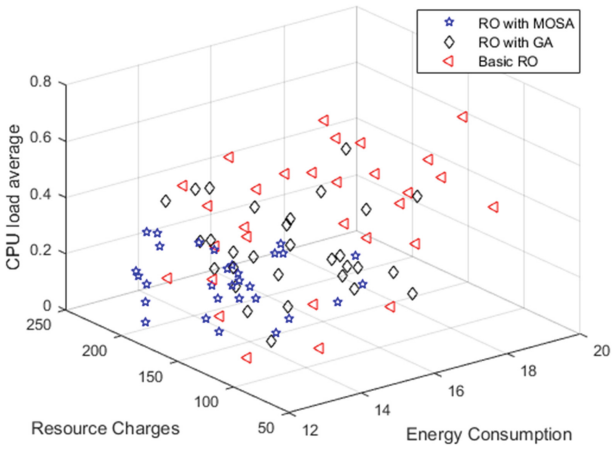

Fig. 9. Comparisons of resource orchestration results

6 Conclusion

In this paper, we target the problem of dynamic resource orchestration in NFV environment. Our work goes beyond existing approaches by considering resource choice for service function chaining, while others mainly focus on single influencing factors or static deployment problem. We thus designed the NFV-based Resource Orchestration Architecture to dynamic select the optimal resource candidates for VNFs. We defined the resource orchestration as an optimization problem with the multi-objective, i.e. energy consumption of infrastructure, resource charges and CPU load average. To obtain the optimal results for resource orchestration, we introduce the Multi-Objective Simulated Annealing in the core of resource orchestration controller. Simulation results show that the multiple solutions for dynamic resource orchestration can archive the optimal solutions in acceptable time. The proposed algorithm can be used in many scenarios, for example, it can be used to dynamically optimize service functions in NFV-based environment.

Acknowledgment. This work is supported by a grant from the National High Technology Research and Development Program of China (863 Program), No. 2014AA01A701, and Beijing Municipal Science and technology Commission research fund project No. Z16111000500000.

References

1. Zhang, H., et al.: Fronthauling for 5G LTE-U ultra dense cloud small cell networks. IEEE Wireless Commun. **23**(6), 48–53 (2016)
2. Chiosi, M., et al.: Network functions virtualisation: an introduction, benefits, enablers, challenges and call for action. In: SDN and OpenFlow World Congress (2012)
3. Halpern, J., Pignataro, C.: Service Function Chaining (SFC) Architecture. No. RFC 7665 (2015)
4. Ranjan, R., et al.: Cloud resource orchestration programming: overview, issues, and directions. IEEE Internet Comput. **19**(5), 46–56 (2015)
5. Zhang, H., et al.: Network slicing based 5G and future mobile networks: mobility, resource management, and challenges. IEEE Commun. Mag. **55**, 138–145 (2017)
6. Qi, Q., et al.: Dynamic resource orchestration for multi-task application in heterogeneous mobile cloud computing. In: 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS). IEEE (2016)
7. Wang, L., et al.: Joint optimization of service function chaining and resource allocation in network function virtualization. IEEE Access **4**, 8084–8094 (2016)
8. Fiorani, M., et al.: Challenges for 5G transport networks. In: 2014 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). IEEE (2014)
9. Sherry, J., Ratnasamy, S., At, J.S.: A survey of enterprise middlebox deployments (2012)
10. Herrera, J.G., Botero, J.-F.: Resource allocation in NFV: a comprehensive survey. IEEE Trans. Netw. Serv. Manage. **13**, 518–532 (2016)
11. Hirschman, B., et al.: High-performance evolved packet core signaling and bearer processing on general-purpose processors. IEEE Network **29**(3), 6–14 (2015)
12. Bronstein, Z., et al.: Uniform handling and abstraction of NFV hardware accelerators. IEEE Network **29**(3), 22–29 (2015)

13. Qi, Q., et al.: Resource orchestration for multi-task application in home-to-home cloud. IEEE Trans. Consum. Electron. **62**(2), 191–199 (2016)
14. Tao, F., et al.: Resource service composition and its optimal-selection based on particle swarm optimization in manufacturing grid system. IEEE Trans. Industr. Inf. **4**(4), 315–327 (2008)
15. Liu, J., et al.: Improve service chaining performance with optimized middlebox placement. IEEE Trans. Serv. Comput. **10**(4), 560–573 (2017)
16. Ma, W.-M., et al.: Utility-based fairness power control scheme in OFDMA femtocell networks. Dianzi Yu Xinxi Xuebao(J. Electron. Inf. Technol.) **34**(10), 2287–2292 (2012)