Dynamic Auto Scaling Algorithm (DASA) for 5G Mobile Networks

Yi Ren*, Tuan Phung-Duc†, Jyh-Cheng Chen*, and Zheng-Wei Yu*
*Department of Computer Science, National Chiao Tung University, Hsinchu, Taiwan, R.O.C.
†Faculty of Engineering, Information and Systems, University of Tsukuba, Ibaraki, Japan
Emails:*{renyi, zwyu12260, jcc}@cs.nctu.edu.tw, †tuan@sk.tsukuba.ac.jp

***Abstract*—Network Function Virtualization (NFV) enables mobile operators to virtualize their network entities as Virtualized Network Functions (VNFs), offering fine-grained on-demand network capabilities. VNFs can be dynamically scale-in/out to meet the performance desire and other dynamic behaviors. However, designing the auto-scaling algorithm for desired characteristics with low operation cost and low latency, while considering the existing capacity of legacy network equipment, is not a trivial task. In this paper, we propose a VNF Dynamic Auto Scaling Algorithm (DASA) considering the tradeoff between performance and operation cost. We develop an analytical model to quantify the tradeoff and validate the analysis through extensive simulations. The results show that the DASA can significantly reduce operation cost given the latency upper-bound. Moreover, the models provide a quick way to evaluate the cost-performance tradeoff and system design without wide deployment, which can save cost and time.**

***Index Terms*—Auto Scaling Algorithm, Modeling and Analysis, Network Function Virtualization, 5G, Cloud Networks, Virtualized EPC**

I. Introduction

Cellular networks have been evolved to 4th generation (4G). Long Term Evolution-Advanced (LTE-A) has become a commonly used communication technology worldwide and is continuously expanding and evolving to 5th generation (5G). One of the most important technologies for 5G networks is to utilize Network Function Virtualization (NFV) to virtualize the network components in the core network which is called Evolved Packet Core (EPC). The virtualized EPC is commonly referred to as virtual EPC (vEPC) [1].

The emergence of NFV enables operators to manage their network equipment in a fine-grained and efficient way [2]. Indeed, legacy network infrastructure equipment suffers from the nature of user experience where data traffic usually have peaks during a day while having relative low utilization in the rest of time (e.g., in the midnight). To guarantee the Quality of user Experience (QoE), operators usually leave spare capacities for tackling the peak traffic while deploying network equipment. Accordingly, the network equipment are under low utilization during the non-busy period. NFV enables operators virtualize hardware resources and makes special-purpose network equipment toward software solutions, i.e., Virtualized Network Function (VNF) instances. A VNF can run on several Virtual Machines (VMs) which can scale-out/in to adjust the VNF's computing and networking capabilities, saving on both energy and resources.

Given the fact that *auto-scaling* VNF instance can decrease operation cost while meeting the demand for VNF service, it is critical to design good strategies to allocate VNF instance adaptively to fulfill the demands of service requirements. However, it is not a trivial task. Specifically, the operation cost is reduced by decreasing the number of power-on VNF instances. On the other hand, resource under-provisioning may cause Service Level Agreements (SLAs) violations. Therefore, the goal of a desirable strategy is to reduce operation cost while also maintaining acceptable levels of performance. Thus, a *cost-performance tradeoff* is formed: The VNF performance is improved by *scaling-out* the number of VNF instances while the operation cost is reduced by *scaling-in* the number of VNF instances.

In this paper, we study the cost-performance tradeoff while considering both the VM setup time and legacy equipment capacity. We propose Dynamic Auto Scaling Algorithm (DASA) to solve the problem. To the best of our knowledge, this has not been discussed in any previous literature. In the proposed DASA, we consider legacy 4G network equipment as a block and powered on all the time, while virtualized resources are divided into k VNF instances. The VNF instances are scaled in and out depending on the number of jobs in the system. A critical issue is how to specify a suitable k for the tradeoff. We propose detailed analytical models to answer this question. The cost-performance tradeoff is quantified as *operation cost metric* and *performance metric* of which closed-form solutions are derived and validated against extensive discrete-event simulations. Moreover, we develop a recursive method that reduces the complexity of the computational procedure from $O(k^3 \times K^3)$ to $O(k \times K)$, where K is the capacity of the system. The models enable wide applicability in various scenarios, and therefore, have important theoretical significance.

The rest of this paper is organized as follows. Section II reviews the related work. Section III briefly introduces some background material on mobile networks and NFV architecture. Section IV presents the proposed algorithm for VNF auto-scaling applications. Section V addresses the analytical models, followed by numerical results illustrated in Section VI. Section VII offers conclusions.

II. Related Work

VM (VNF instance[1]) auto-scaling mechanisms have been intensively studied [3]–[8]. However, *existing methods either ignore VM setup time or only consider virtualized resource itself while overlooking legacy (fixed) resources*. This is not practical in typical cellular networks. Although a scale-out request can be sent right way, a VNF instance cannot be available immediately. The lag time could be as long as 10 min or more to start an instance in Microsoft Azure and the lag time could be various from time to time [9]. It could happen that the instance is too late to serve the VNF if the lag time is not taken into consideration. The capacity of legacy network equipment is also an issue worth careful consideration. For example, a network operator deployed legacy network equipment wants to increase network capacities by using NFV technique. The desired solution should consider the capacities of both legacy network equipment and VNFs. Consider VNF only case that a VNF scaling-out from 1 VNF instance to 2 VNF instances increases 100% capacity. Whereas, its capacity only grows less than 1% if legacy network equipment (say 100 VNF instance capability) is counted. Current cloud auto-scaling schemes usually ignore the non-constant issue.

Perhaps the closest models to ours were studied in [4]–[8] that both the capacities of fixed legacy network equipment and dynamic auto-scaling cloud servers are considered. The authors in [4], [5] consider setup time without defections [4] and with defections [5]. Our recent work [7] relaxes the assumption in [4], [5] that after a setup time, all the cloud servers in the block are active concurrently. We further consider a more realistic model that each server has an independent setup time. However, in [4], [5], [7], all the cloud servers were assumed as a whole block, which is not practical where each cloud server should be allowed to scale-out/in dynamically. Considering all cloud servers as a whole block was relaxed to sub-blocks in [6], [8]. However, either setup time is ignored [6], or fixed legacy network capacity is not considered [8].

III. Background

Mobile Core Network (CN) is one of the most important parts in mobile networks. The main target of NFV is to virtualize the functions in the CN. The most recent CN is the Evolved Packet Core (EPC) introduced in Long Term Evolution (LTE). Here, we use an example to explain EPC and virtualized EPC (vEPC) when NFV is deployed. Fig. 1 shows a simplified example of NFV enabled LTE architecture consisted of Radio Access Network (RAN), EPC, and external Packet Data Network (PDN). In particular, the EPC is composed of legacy EPC and vEPC. In the following, we brief introduce them respectively.

A. Legacy EPC

EPC is the CN of the LTE system. Here, we only show basic network functions, such as Serving Gateway (S-GW), PDN Gateway (P-GW), Mobility Management Entity (MME), and Policy and Charging Rules Function (PCRF) in the EPC.

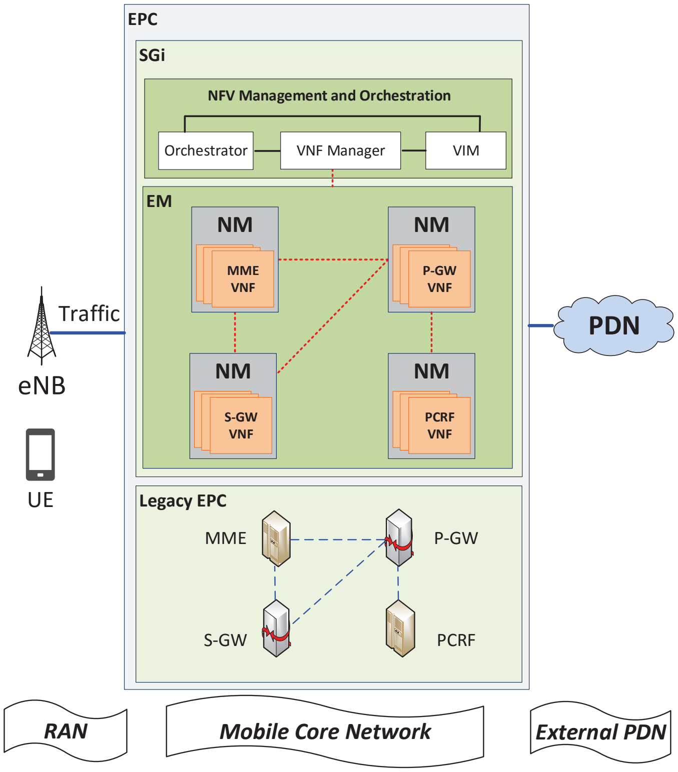

Fig. 1: A simplified example of NFV enabled LTE architecture.

B. vEPC

To virtualize the above network functions, 3GPP introduces NFV management functions and solutions for vEPC based on ETSI NFV specification [10], as shown in Fig. 1. The network functions (e.g., MME, PCRF) are denoted as Network Elements (NE), which are virtualized as VNF instances. Network Manager (NM) provides end-user functions for network management of NEs. Element Manager (EM) is responsible for the management of a set of NMs. NFV management and orchestration controls VNF instance scaling procedure, which are detailed as follows.

- VNF scale-in/out: VNF scale-out adds additional VMs to support a VNF instance, adding more virtualized hardware resources (i.e., compute, network, and storage capability) into the VNF instance. In contrast, VNF scale-in removes existing VMs from a VNF instance.
- VNF scale-up/down: VNF scale-up allocates more hardware resources into a VM for supporting a VNF instance (e.g., replace a One-core with Dual-core CPU). Whereas, VNF scale-down releases hardware resources from a VNF instance.

IV. Proposed VNF Instance Auto-Scaling Algorithm

The goal of VNF instance auto-scaling algorithm is to reduce operation cost while providing acceptable levels of performance. Here, the performance is evaluated by average response time per user request. More power-on VNF instances reduce the possibility of SLAs violations. However, this may

[1] In this paper, VM and VNF instance are used interchangeably.

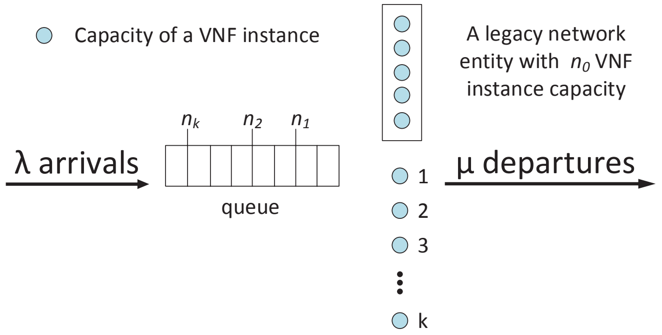
Capacity of a VNF instance
A legacy network entity with n_0 VNF instance capacity
n_k
n_2
n_1
λ arrivals
queue
μ departures
1
2
3
k

Fig. 2: A simplified queueing model for our system.

incur redundant power-on VNF instances, leading to more operation cost. We refer the tradeoff as cost-performance tradeoff. To balance the tradeoff, we first propose a VNF instance Dynamic Auto-Scaling Algorithm (DASA), and then present the optimal DASA.

A. System Model and DASA: Dynamic Auto-Scaling Algorithm

Consider a 5G EPC comprised of both legacy network entities (e.g., MME, PCRF) and VNFs. A VNF, consisting of k VNF instances, is used to add more capacities to its corresponding legacy network entity (see Fig. 2 as an example). We assume that the capacity of the legacy network entity equals to n_0 VNF instance capacities. That is, the total capacity of the network entity is $k + n_0 = N$. We assume that $n_1 = n_0 + 1$ and $n_i = n_{i-1} + 1$ ($i = 1, 2, \cdots k$). It should be noted that $n_k = N$. User request arrives with rate λ. A VNF instance accepts one job at a time with service rate μ. There is a limited First-Come-First-Served (FCFS) queue for those requests that have to wait for processing. The legacy network equipment is always on while its VNF instances will be added (or removed) according to the number of waiting user requests in the buffer. It is worth to mention that the VNF instances need some setup time to be available so as to process waiting requests.

Two thresholds, 'up' and 'down', or U_i and D_i, denote the control of the VNF instances $i = 1, 2, \cdots, k$.

- *U_i, power up the i-th VNF instances:* If the i-th VNF instance is turned off and the number of requests in the system increases from $U_i - 1$ to U_i, then the VNF instance is powered up after a setup time to support the system. During the setup time, a VNF instance cannot serve user requests, but consumes power (or money for renting cloud services). Here, we specify $U_i = n_i$.
- *D_i, power down the i-th VNF instances:* If the i-th VNF instance is operative, and the number of requests in the system drops from $D_i + 1$ to D_i, then the VNF instance is powered down instantaneously. Here, we choose $D_i = n_{i-1}$.

The system performance is evaluated by two metrics: the average response time in the queue per request, W_q, and the average number of VNF instance consuming power, S. The closed-form solutions of W_q and S are given as (1) and (2) in Section V. Thus, the system performance P has the form

$$P = w_1 W_q + w_2 S,$$

TABLE I: List of Notations

Notation	Explanation
N	The number of servers in server center
K	The number of maximum jobs can be accommodated in the system
k	The number of VNF instances
P	System performance
W	Average response time per job
W_q	Average response time in the queue per job
S	Average VM cost
w_1	Weight factor for W_q
w_2	Weight factor for S
n_0	The number of permanently operative servers
U_i	The up threshold to control the reserve sub-blocks
D_i	The down threshold to control the reserve sub-blocks
m_i	The i-th reserve sub-block ($i = 1, 2, \cdots k$).
λ	Job arrival rate
μ	Service rate for each server
α	Setup rate for each virtual server

where coefficients w_1 and w_2 denote the weight factors for W_q and S, respectively. Increasing w_1 (or w_2) emphasizes more on W_q (or S). Here, we do not specify either w_1 or w_2 due to the fact that such a value should be determined by a mobile operator and must take management policies into consideration. Next, we provide an example of specifying k based on different weights accordingly to a operator's management policies.

V. Analytical Model

In this section, we propose the analytical model for DASA. The goal of the analytical model is to analyze both the operation cost and the system performance for DASA. Given the analytical model, one can quickly obtain the operation cost and system performance for DASA, without real deployment, saving on cost and time.

We model the system as a queueing model with N servers and a capacity of K, i.e., the maximum of K jobs can be accommodated in the system. Job arrivals follow Poisson distribution with rate λ. A VNF instance (server) accepts one job at a time, and its service rate follows the exponential distribution with rate μ. There is a limited FCFS queue for those jobs that have to wait for processing.

In this system, a server is turned off immediately if it has no job to do. Upon arrival of a job, an OFF server is turned on if any and the job is placed in the buffer. However, a server needs some setup time to be active so as to serve waiting jobs. We assume that the setup time follows an exponential distribution with mean $1/\alpha$. Let j denotes the number of customers in the system and i denotes the number of active servers. The number of reserves (server) in setup process is $\min(j - n_i, N - n_i)$. Here, $n_i = n_{i-1} + m_i$, where $m_i = 1$ for all i (block size is one). Therefore, in this model a server in reserve blocks is in either BUSY or OFF or SETUP. We assume that waiting jobs are served according to an FCFS manner. We call this model an M/M/N/K/Setup queue.

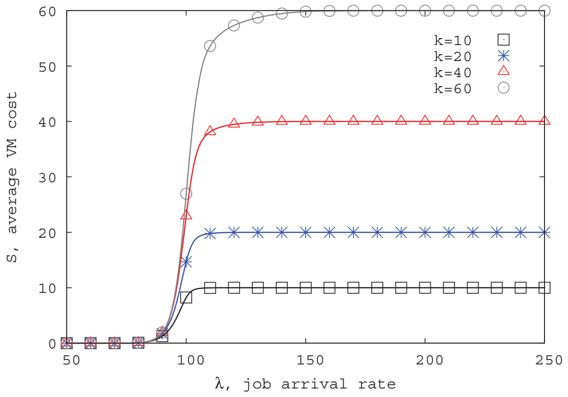

Fig. 3: Impacts of k on S ($n_0 = 100$).

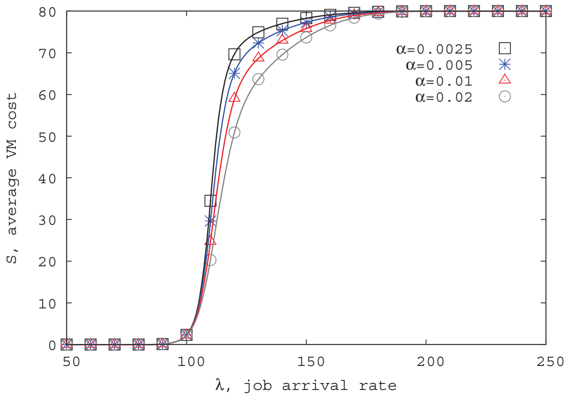

Fig. 4: Impacts of α on S ($k = 80$).

Let $\pi_{i,j}$ denote the probability that there are i busy VNF instances and j customers in the system. We found a simple and efficient recursive scheme to obtain $\pi_{i,j}$. Due to the page limitation, we only show the final derivation results as follows. Interested reader may refer to [11] for detailed mathematical analysis.

$$W = \frac{\sum_{i=0}^{n_0-1} \pi_{0,j} j + \sum_{i=0}^{k} \sum_{j=n_i}^{K} \pi_{i,j} j}{\lambda(1 - \sum_{i=0}^{k} \pi_{i,K})},$$

where the numerator and denominator are the mean number of customers in the system and the arrival rate of accepted customers to the system, respectively.

We obtain

$$W_q = W - \frac{1}{\mu}. \tag{1}$$

The mean number of VNF instances is given by

$$S = \sum_{(i,j)\in\mathcal{S}} \pi_{i,j}(n_i - n_0) + \sum_{i=0}^{k} \sum_{j=n_i}^{K} \pi_{i,j} \min(j - n_i, N - n_i), \tag{2}$$

where the first term is the number of VNF instances that are already active while the second term is the mean number of VNF instances in setup mode. It is worth to mention that the complexity of the computational procedure is of order $O(k \times K)$ instead of $O(k^3 \times K^3)$ if we directly solve the system of balance equations by a general method.

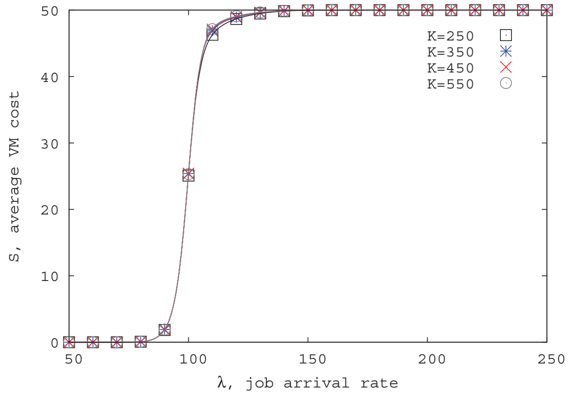

Fig. 5: Impacts of K on S ($k = 50$).

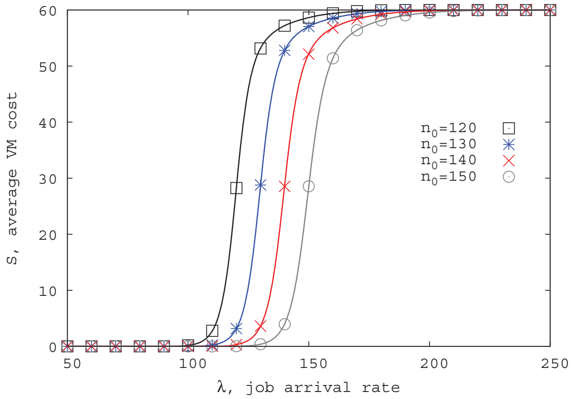

Fig. 6: Impacts of n_0 on S ($k = 60$).

VI. Simulation and Numerical Results

The analytical results in Section V are validated by extensive simulations by using ns-2, version 2.35 [12]. Here, we used real measurement results for parameter configuration[2]: λ by Facebook data center traffic [13], μ by the base service rate of a Amazon EC2 VM [14], and α by the average VM startup time [15]. If not further specified, the following parameters are set as the default values for performance comparison: $n_0 = 110$, $\mu = 1$, $\alpha = 0.005$, $K = 250$, $\lambda = 50 \sim 250$ (see Table 1 for details).

Figs. 3-9 illustrate both the simulation and analytical results in terms of the performance metrics: average VM cost S and average response time in a queue per job W_q, respectively. In the figures, the *lines* denote analytical results, and the *points* represent simulation results. In the following sections, we show the impacts of λ, k, K, n_0, α on the performance metrics S and W_q, respectively.

A. Impacts of arrival rate λ

Figs. 3-6 show the impacts of λ on S. Generally, one can see that S is 0 at the beginning, then grows sharply, later raises smoothly and reach at a bound as λ increases. The reasons are as follows. When $\lambda \ll n_0\mu$, the incoming jobs are handled by the legacy equipment. No VMs are turned on. Later, VMs are turning on as λ approaches to $(n_0 + k)\mu$. Accordingly,

[2] Due to the simulation time limitation, λ and μ are scaled down accordingly with the same ratio λ/μ.

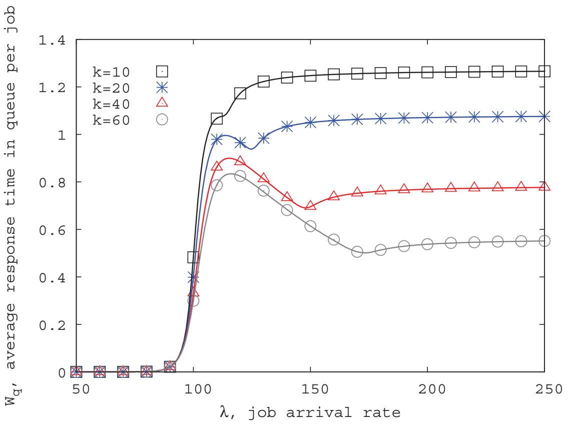

Fig. 7: Impacts of k on W_q ($n_0 = 100$).

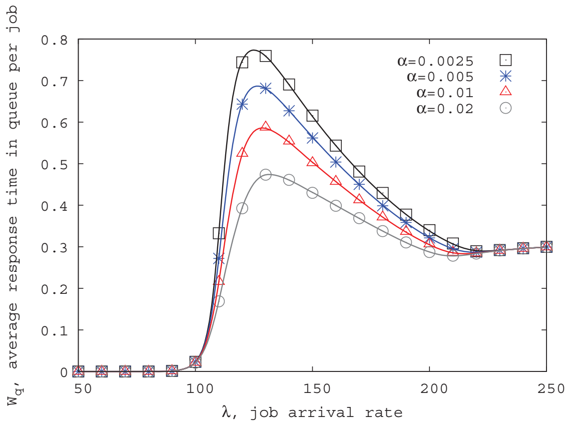

Fig. 8: Impacts of α on W_q ($k = 80$).

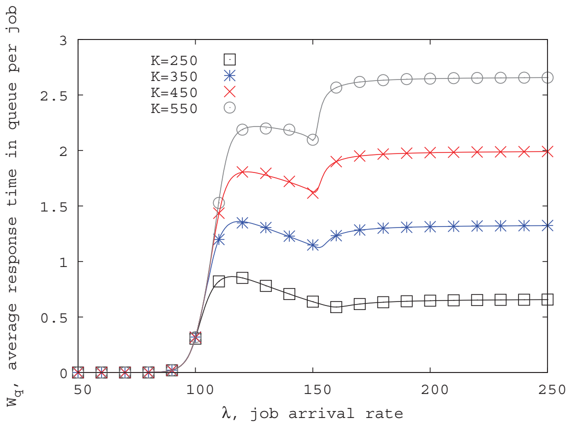

Fig. 9: Impacts of K on W_q ($k = 50$).

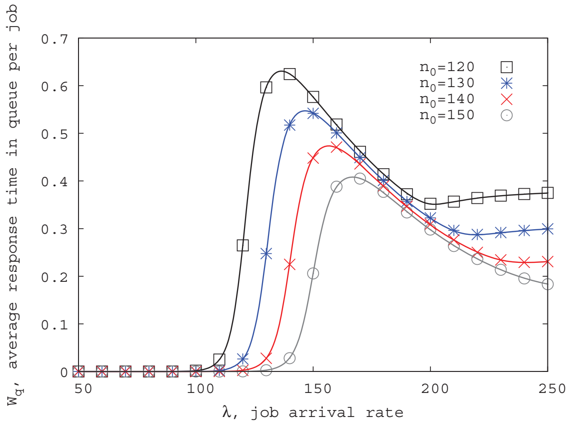

Fig. 10: Impacts of n_0 on W_q ($k = 60$).

the server cost S increases as λ grows. Then S stops growing when $\lambda > (n_0 + k)\mu$. Because all the k VMs are turned on so that S is bounded as k VM costs.

Figs. 7-10 illustrate the impacts of λ on W_q. Interestingly, the trend of the curves can generally be divided into three phases[3]: ascent phase, descent phase, and saturation phase. In the first phase, W_q grows sharply due to the setup time of VMs. Specifically, when $\lambda \ll n_0\mu$, W_q is almost 0 as all jobs are handled by legacy equipment. As λ approaches to $n_0\mu$ and then larger than $n_0\mu$, VMs start to be turned on. However, in this phase W_q still raises due the setup time of VMs. The reason is that VMs just start to be turned on and do not reach their full capacities. In the second phase, we can see that W_q starts to descend because the VMs start serving jobs. In the third phase, however, W_q starts to ascend again and then saturate at a bound. The reason of ascent is that when $\lambda \geq (n_0 + k)\mu$, the system is not able to handle the coming jobs. Finally, the curves go to saturation because the capacity of the system is too full to handle the jobs and the value of W_q is limited by K.

B. Impacts of the number of VNF instances k

Fig. 3 shows the impacts of k on S. The impacts of k is shown as the ascend phase before the bound or the gap between the initial point and the bound. A larger k leads to a longer ascend phase (or bigger gap). Moreover, as λ grows, a larger k means that more VMs could be used to handle the growing job requests. So S increases accordingly. If a operator wants to bound VM budget S, the operator can specify a suitable k based on (2). We can also see that the gaps are the same in Figs. 4-6 due to the same k in these figures.

Fig. 7 illustrates the impacts of k on W_q. The impacts of k are shown as the length of the second phase as discussed in Sec. VI-A. The length of the second phase prolongs as k increases. Because a larger k gives the system more capability to handle the raising job requests. That is, it delays the time that the system capacity reaches its bound. If an operator wants to bound job request response time W_q, the operator can choose a suitable k based on (1).

C. Impacts of VM setup rate α

Recall that α is the setup rate of VMs. To change setup rate, one can adjust resources (e.g., CPU, memory) for VMs. Fig. 4 shows the impacts of α on S. The impacts of α are shown as the slope of the curves. A larger α means smaller slope, but α has no effects at the beginning and the end of the curves. The reasons are as follows. A larger α means smaller VM setup time. A smaller VM setup time helps VM faster to be turned on and to handle jobs so that the system is more efficient than the VM with large setup time.

[3] In Figs. 8 and 10, only two phases are displayed due to the range of λ. Given a larger λ, all the three phases will be shown.

Fig. 8 illustrates the impacts of α on W_q. Again, the impacts of α are shown as the slope of curves. In contrast, a larger α leads to smaller slopes. Also, α decides the maximum value of W_q. The reason is that smaller setup time enables VM to handle jobs faster.

D. Impacts of system capacity K

Fig. 5 and Fig. 9 depict the impacts of K on S and W_q, respectively. Based on our observation on Fig. 5, K has limited impacts on S. As we discussed in Sec. VI-A, S is mainly decided by k. As shown in Fig. 9, the impacts are significant on W_q. Different K makes huge gaps between the curves. The curves also form as three phases. A large K leads to a larger W_q. The reason is that it enables more jobs waiting in the queue rather than dropping them.

E. Impacts of legacy equipment capacity n_0

Fig. 6 and Fig. 10 illustrate the impacts of n_0 on S and W_q, respectively. We observe that the curves initiate at 0 then fix at 0 for a period and start to grow up as λ increases. n_0 decides the length of the period when the curves start to grow up. The reason is that the legacy equipment can handle jobs within its capacity. When λ exceeds the capacity of the legacy equipment, both S and W_q start to grow up.

VII. Conclusions

In this paper, we have proposed DASA for addressing the tradeoff between performance and operation cost. We developed analytical and simulation models to study the average job response time W_q and operation cost S. The NFV enabled EPC is modeled as queueing model while legacy network equipment are considered as reserved a block of servers; and VNF instances are powered on and off according to the number of job requests present. Our model fills the research gap by taking both VM setup time and the capacity of legacy equipment into consideration in NFV enable EPC scenarios. Our study provides mobile operators with guidelines to bound their operation cost and configure job request response delay. Based on our performance study, the operators can further design optimization strategies without wide deployment, saving on cost and time.

References

[1] ETSI GS NFV-INF 001 v.1.1.1, *Network Functions Virtualisation (NFV); Infrastructure Overview*, ETSI Std., Jan. 2015.

[2] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, “NFV: state of the art, challenges, and implementation in next generation mobile networks (vEPC),” *IEEE Network*, vol. 28, no. 6, pp. 18–26, Nov./Dec. 2014.

[3] G. Galante and L. C. E. d. Bona, “A survey on cloud computing elasticity,” in *Proc. IEEE Int’l Conference on Utility and Cloud Computing (UCC)*, Nov 2012, pp. 263–270.

[4] I. Mitrani, “Managing performance and power consumption in a server farm,” *Annals of Operations Research*, vol. 202, no. 1, pp. 121–134, 2013.

[5] ——, “Service center trade-offs between customer impatience and power consumption,” *Elsevier Performance Evaluation*, vol. 68, no. 11, pp. 1222 – 1231, Nov. 2011.

[6] ——, “Trading power consumption against performance by reserving blocks of servers,” *Springer Computer Performance Engineering*, vol. LNCS 7587, pp. 1–15, 2013.

[7] J. Hu and T. Phung-Duc, “Power consumption analysis for data centers with independent setup times and threshold controls,” in *Proc. Int’l Conf. Numerical Analysis And Applied Mathematics (ICNAAM’14).*

[8] T. Phung-Duc, “Multiserver queues with finite capacity and setup time,” in *Proc. 22nd. Int’l Conf. Analytical and Stochastic Modelling Techniques and Applications, ASMTA’15*, May 2015, pp. 173–187.

[9] Z. Hill, J. Li, M. Mao, A. Ruiz-Alvarez, and M. Humphrey, “Early observations on the performance of Windows Azure,” in *Proc. 19th ACM HPDC*, Jun. 2010, pp. 367–376.

[10] 3GPP TR 32.842 V13.1.0, “Telecommunication management; Study on network management of virtualized networks (Release 13),” Tech. Rep., Dec. 2015.

[11] “Design and analysis of Dynamic Auto Scaling Algorithm (DASA) for 5G mobile networks,” NCTU, Tech. Rep., 2016. [Online]. Available: https://arxiv.org/abs/1604.05803

[12] ”The network simulator - ns-2.” Available: http://www.isi.edu/nsnam/ns/.

[13] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren, “Inside the social network’s (datacenter) network,” in *Proc. ACM SIGCOMM*, 2015.

[14] M. Gilani, C. Inibhunu, and Q. H. Mahmoud, “Application and network performance of Amazon elastic compute cloud instances,” in *Proc. IEEE 4th Int’l Conf. Cloud Networking (CloudNet)*, 2015, pp. 315–318.

[15] M. Mao and M. Humphrey, “A performance study on the VM startup time in the cloud,” in *IEEE 5th Int’l Conf. Cloud Computing (CLOUD),*, 2012, pp. 423–430.