Live Migration for Service Function Chaining

Dongcheng Zhao[1], Gang Sun[1,2], Dan Liao[1,3], Rahat Iqbal[4] and Victor Chang[5]
[1]*Key Lab of Optical Fiber Sensing and Communications (Ministry of Education), UESTC, Chengdu, China*
[2]*Center for Cyber Security, UESTC, Chengdu, China*
[3]*Guangdong Institute of Electronic and Information Engineering, UESTC, Dongguan, China*
[4]*Coventry University, Coventry, U.K.*
[5]*Xi'an Jiaotong Liverpool University, Suzhou, China*

Keywords: Network Function Virtualization, Virtual Network Function, Service Function Chaining, Migration, Middle Boxes.

Abstract: Network Function Virtualization (NFV) has been proposed to solve these challenges of hardware middle boxes such as high Capital Expenditures (CAPEX) and Operational Expenditures (OPEX). NFV aims to move packet processing from hardware middle boxes to software middle boxes running on commodity hardware. In NVF, users or virtual machines (VMs) communicate through the service function chaining. Therefore, when VMs are migrated, the service function chaining also needs to be migrated. Most research on migration focus on the issue of VM migration, and at present there is little research on the migration problem of the service function chaining. Therefore, in this paper we focus on the service function chaining migration, we will introduce the serial migration strategy and the parallel migration strategy for multiple VMs into the migration problem of the service function chaining, and propose an improved serial migration strategy for the service function chaining that is based on the serial migration strategy. We then present the *m* mixed migration strategy for the service function chaining that is based on the improved serial migration strategy and the parallel migration strategy. We conduct detailed simulations to evaluate the performance of the *m* mixed migration strategy in terms of the migration time and the downtime. We also develop the M/M/C/C and the M/M/C queuing models to calculate performance indicators, such as the blocking rate of each migration request.

1 INTRODUCTION

In the traditional telecommunication networks, Network Functions (NFs) or middle boxes (such as firewalls, content filters, proxies, WAN optimizers, Intrusion Detection Systems (IDSs) and Intrusion Prevention Systems (IPSs).) are implemented as some physical proprietary devices and equipment. However, with users' requirements for more diverse and new services continue to increase, service providers must correspondingly purchase, store and operate new physical devices to satisfy users' requirements (Mijumbi, 2016). However, the purchase of new physical devices will generate high CAPEX and OPEX (Bari et al 2015; Wu et al 2015; Bondan et al 2014). These physical devices require specially trained personnel for deployment and maintenance.

To solve the above challenges, researchers have proposed the Network Function Virtualization (NFV) which aims to move packet processing from hardware middle boxes to software middle boxes running on commodity hardware. These network functions implementing in software middle boxes are called as Virtual Network Functions (VNFs). In NFV, the packet processing tasks are performed by software middle boxes rather than hardware middle boxes, and multiple VNFs are typically in a particular order connected to compose the service function chaining that provide different network services (Gember-Jacobson et al 2014; Mehraghdam et al 2014). For example, a communication or traffic between two virtual machines (VMs) needs pass through the service function chaining: VM→Firewall→IDS→Proxy→ VM, is commonly deployed between two users to enforce security policies of traffic filtering. Typically, the type and order of each VNF in the service function chaining is determined depending on the classification of traffic, service-level agreement (SLA), and operator's provisioning policies, and so on (Xia et al, 2015).

Zhao, D., Sun, G., Liao, D., Iqbal, R. and Chang, V.
Live Migration for Service Function Chaining.
DOI: 10.5220/0006364701490156
In *Proceedings of the 2nd International Conference on Internet of Things, Big Data and Security (IoTBDS 2017)*, pages 149-156
ISBN: 978-989-758-245-5

As an emerging technology, NFV has received extensive attention from industry, academia, and standardization bodies. At present, the NFV location also has become a hot research problem, and there exist some researches on the NFV location problem (Li et al 2015; Cohen et al 2015; Kim et al 2015). Since VMs communicate through the service function chaining, and therefore the position of the VM will affect the position of each VNF in the service function chaining. And because of VM migrations often occur, so when VMs are migrated, the service function chaining also needs to be migrated by following these VMs. In (Li et al 2015; Cohen et al 2015; Kim et al 2015), these proposed methods can solve the NFV location problem, however, these methods can't solve the migration problem of the service function chaining, and at present there is little research on the migration problem of the service function chaining. Researchers have proposed a variety of VM migration technologies (Tso et al 2013; Shrivastava et al 2011; Cerroni 2014; Callegati et al 2013) for live migration of VMs in data center. Although these migration strategies can solve VM migration problem, they can't be directly used for the problem of service function chaining migration.

Therefore, in this paper, we will introduce the serial migration strategy and the parallel migration strategy for multiple VMs that are proposed in (Cerroni 2014; Callegati et al 2013) into the migration problem of the service function chaining. And we study the improved serial migration strategy for the service function chaining and analyze its performance. Then we present the *m* mixed migration strategy for the service function chaining. This *m* mixed migration strategy aims to minimize the total migration time while satisfying the maximum downtime constraint that users agreed with the service providers in the SLAs. In order to evaluate the blocking ratio, we model the problem by using M/M/C/C queuing model. We use the M/M/C multi-server queuing model to evaluate the average waiting queue length and the average waiting time.

2 STRATEGIES FOR MIGRATION FOR SERVICE FUNCTION CHAINING

In this section, we discuss the migration of the service function chaining in NFV environment. In NFV, each VNF of the service function chaining is instantiated by a VM, namely, the VNFs hosted by VMs. Therefore, when we migrate the service function chaining, we can treat each VNF of the service function chaining as a VM for processing, and when we describe migration for VNF in the following, we use VM to instead of VNF. In this work, we use the pre-copy migration mechanism (Callegati and Cerroni, 2013) to migrate each VNF running on VMs, the VM memory is migrated iteratively through using the pre-copy migration mechanism. The VM memory contains the original memory and the dirty (i.e., generated or modified) memory. The original memory is the memory when starting the VM migration. The dirty memory is generated or modified during the iterative transmission process. The iterative migration process of VM_i (i =1, 2,…,M) is described in (Callegati and Cerroni, 2013). We can calculate the total migration time (denoted as $T_{i,mig}$) of the i-th VM or VNF as follows.

$$T_{i,mig} = \sum_{i=1}^{n_i+1} T_{i,j} = \frac{V_m}{R}\frac{1-r^{n_i+1}}{1-r} \quad (1)$$

$$n_i = \min\{\lceil \log_r (V_{th} / V_m) \rceil, n_{\max}\} \quad (2)$$

Equation (1) shows that the migration time is the sum of the time of each iteration. Equation (2) gives the actual number of iterations which denoted as n_i. Where $T_{i,j}$ is the total time consumed in the j-th iteration for transmitting the i-th VM's dirty memory. We assume that the amount of original memory of each VNF is V_m. We use V_{th} to denote the iteration threshold for stopping the iteration and n_{max} to present the maximum number of iterations. R is the transmission rate and $r = PD/R$ is the ratio of the dirtying rate to the transmission rate, where D and P are memory page dirtying rate and memory page size, respectively. Here the memory page dirtying rate is the rate at which the dirty (or modified) memory page is generated. The number of VNFs in the service function chaining is M.

In this paper, we focus on the migration strategy for the service function chaining. In (Cerroni 2014; Callegati et al 2013), the authors proposed a serial migration strategy for migrating multiple VMs. Similar to (Cerroni 2014; Callegati et al 2013), we can use a similar strategy to serially migrate each VNF in the service function chaining. In the serial migration strategy for the service function chaining, the M VMs instantiating VNFs are migrated one at a time, and the actual number of each VM's iterations is $n(s)$. Therefore, similar to (Cerroni 2014; Callegati et al 2013), the serial migration time (denoted as T_{mig}^{s}) and the downtime (denoted as T_{down}^{s}) can be calculated as in Equations (3) and (4), respectively.

$$T_{mig}^{s} = \sum_{i=1}^{M} T_{i,mig} = \frac{MV_m}{R}\frac{1-r^{n(s)+1}}{1-r} \quad (3)$$

$$T_{down}^{s} = \frac{V_m}{R} r^{n(s)} + (M-1)\frac{V_m}{R}\frac{1-r^{n(s)+1}}{1-r} + T_{res} \quad (4)$$

Where $n(s) = \min\{\lceil \log_r (V_{th}/V_m) \rceil, n_{\max}\}$ and $0 < r < 1$.

The authors in (Cerroni 2014; Callegati et al 2013) also proposed a strategy for the parallel migration of multiple VMs. Similarly, we can use a similar strategy to parallel migrate each VNF in the service function chaining. In the parallel migration strategy for the service function chaining, the M VMs instantiating VNFs are migrated simultaneously, and the actual number of each VM's iterations is $n(p)$. The migration time (denoted as T_{mig}^{p}) and the downtime (denoted as T_{down}^{p}) are formulated as follows.

$$T_{mig}^{p} = \sum_{i=1}^{M} T_{i,mig} = \frac{MV_m}{R}\frac{1-(Mr)^{n(p)+1}}{1-Mr} \quad (5)$$

$$T_{down}^{p} = \frac{MV_m}{R}(Mr)^{n(p)} + T_{res} \quad (6)$$

Where the actual number of each VM's iterations is $n(p) = \min\{\lceil \log_{Mr} (V_{th}/V_m) \rceil, n_{\max}\}$, $0 < Mr < 1$ is the ratio of the dirtying rate of memory page to the transmission rate.

2.1 Improved Serial Migration for Service Function Chaining

In the serial migration strategy for the service function chaining, when the first VM stops, the stopped or migrated VMs loose connection(s) with each other and hence, the service becomes unavailable. So when we have stopped one VM, the other correlated VMs will also be stopped. Therefore, these stopped VMs do not produce any new dirty memory during the migration process, and the VM memory only needs to be migrated one time, without multiple iterations transmission. Thus, there is room for shortening the migration time and the downtime of the service function chaining. Accordingly, we propose an improved serial migration strategy of the service function chaining.

In the improved serial migration strategy, we use the pre-copy scheme to migrate the first VM and use the post-copy scheme to migrate the rest of the VMs one by one. The post-copy strategy (Hines et al, 2009) includes three phases: (*i*) Stop the source VM and copy the CPU state to the destination VM; (*ii*) Restart the destination VM; and (*iii*) Copy the VM memory that will be used. In the post-copy strategy, when the VM is restarted, the VM memory is empty. If the VM tries to access a memory page that has not yet been copied, this memory page needs to be brought from the source VM. Accordingly, we get the total migration time of the *i*-th VM in the worst case:

$$T_{i,mig} = \frac{V_m}{R}, i = 2,...,M \quad . \quad (7)$$

Equation (7) gives the time for migrating all the memory of a VM. However, most of the time, many of the memory pages will not be used, hence we only need to copy the VM memory that will be used. Therefore we introduce a correction factor α into Equation (7), where α is the ratio of the actual migrated memory to all memory. So αV_m denotes the actual amount of memory that needs to be migrated. And since V_{th} is the threshold for stopping the iterations, so αV_m is larger than V_{th}, namely, $\alpha V_m > V_{th}$, thus $\alpha > V_{th}/V_m$. Therefore, we can get $V_{th}/V_m < \alpha \leqslant 1$.

$$T_{i,mig} = \frac{\alpha V_m}{R}, \frac{V_{th}}{V_m} < \alpha \leq 1, i = 2,...,M \quad (8)$$

In the improved serial migration strategy, the first VM's actual number of iterations is $n(s_1)$ and $n(s_1) = n(s)$, and the rest M-1 VMs are migrated by using the post-copy migration scheme, so they do not need iterative migration. According to the improved serial migration strategy for the service function chaining and Equations (1) and (8), we can get the migration time of the improved serial migration strategy (denoted as T_{mig}^{s1}) as follows.

$$T_{mig}^{s1} = \frac{V_m}{R}\frac{1-r^{n(s)+1}}{1-r} + (M-1)\frac{\alpha V_m}{R}, \frac{V_{th}}{V_m} < \alpha \leq 1 \quad (9)$$

When the first VM stopped, the other VMs will also be stopped and the service will be unavailable. Thus, we can get the downtime of the improved serial migration strategy (denoted as T_{down}^{s1}) as in Equation (10).

$$T_{down}^{s1} = \frac{V_m}{R} r^{n(s)} + (M-1)\frac{\alpha V_m}{R} + T_{res}, \frac{V_{th}}{V_m} < \alpha \leq 1 \quad (10)$$

2.2 Comparing Serial, Improved Serial and Parallel Migration

Similar to (Cerroni 2014; Callegati et al 2013), we assume that $n(s) = \log_r(V_{th}/V_m)$ and $n(p) = \log_{Mr}(V_{th}/V_m)$. We can compare our improved serial migration

strategy with the serial migration strategy as follows.

$$T_{mig}^{s} - T_{mig}^{s1} = \frac{(M-1)V_m}{R}\frac{(1-\alpha)+(r-r^{n(s)+1})}{1-r} > 0 \quad (11)$$

where $V_{th}/V_m < \alpha \le 1, 0 < r < 1$ .We can also get:

$$T_{down}^{s} - T_{down}^{s1} = \frac{(M-1)V_m}{R}\frac{(1-\alpha)+(r-r^{n(s)+1})}{1-r} > 0 \quad (12)$$

where $V_{th}/V_m < \alpha \le 1, 0 < r < 1$.

From Equations (11) and (12), we can see that the downtime and the migration time of our improved serial migration strategy are shorter than that of the serial migration strategy.

In the following, we compare the migration time and downtime of improved serial migration strategy with that of the parallel migration strategy.

$$\begin{aligned} & T_{mig}^{p} - T_{mig}^{s1} \\ &= \frac{1}{R}\frac{(Mr-r)(V_m-V_{th})}{(1-Mr)(1-r)} + \frac{(M-1)}{R}\left(\frac{V_m - MrV_{th}}{1-Mr} - \alpha V_m\right) \quad (13) \\ &> 0 \end{aligned}$$

where, $\frac{1}{R}\frac{(Mr-r)(V_m-V_{th})}{(1-Mr)(1-r)} > 0$, $\frac{V_{th}}{V_m} < \alpha \le 1$ and $0 < Mr < 1$ when α=1 , $\frac{(M-1)}{R}\left(\frac{V_m - MrV_{th}}{1-Mr} - \alpha V_m\right)$ has the minimum value.

Since $\frac{(M-1)}{R}\left(\frac{V_m - MrV_{th}}{1-Mr} - \alpha V_m\right) > \frac{(M-1)}{R}\frac{Mr(V_m-V_{th})}{1-Mr} > 0$, we have:

$$T_{down}^{p} - T_{down}^{s1} = \frac{(M-1)(V_{th}-\alpha V_m)}{R} < 0, when \frac{V_{th}}{V_m} < \alpha \le 1 \quad (14)$$

Equations (13) and (14) show that the migration time of our improved serial migration strategy is shorter than that of the parallel migration strategy; and when $V_{th}/V_m < a \leqslant 1$, the downtime of our improved serial migration strategy is longer than that of the parallel migration strategy.

2.3 *M* Mixed Migration of the Service Function Chaining

Since the migration time affects the performance of the whole IT infrastructure, the downtime will affect the quality of experience (QoE) of users. In order to guarantee the QoE, the service providers have to negotiate the maximum tolerable downtime of each migration request with the users. While satisfying the constraint on maximum tolerable downtime, we should reduce the migration time as much as possible, since it has a substantial positive impact on the performance of the whole IT infrastructure.

For this purpose, we propose a mixed migration strategy for the service function chaining that is based on the improved serial migration and parallel migration strategies. We first use the pre-copy migration scheme to migrate *m* VMs in parallel. When the *m* VMs have been stopped running, we stop the rest of the VMs, and then use the post-copy migration strategy to serially migrate the rest of the VMs. When all VMs are migrated completely, we restart all VMs simultaneously. We call such mixed migration strategy as the *m* mixed migration strategy in this work.

Therefore, we can get the migration time T_{mig}^{m} and the downtime T_{down}^{m} of the *m* mixed migration strategy as follows, where the actual number of iterations of each VM is $n(m)=\min\{\lceil \log_{mr}(V_{th}/V_m)\rceil, n_{\max}\}$, $1 \leqslant m \leqslant M$ and $0 < Mr < 1$. Note that, when m =1, the *m* mixed migration strategy turns into the improved serial migration strategy. Similarly when m =*M*, the *m* mixed migration strategy turns into the parallel migration strategy.

$$\begin{aligned} T_{mig}^{m} &= \frac{mV_m}{R}\frac{1-(mr)^{n(m)+1}}{1-mr} \\ &\quad + (M-m)\frac{\alpha V_m}{R}, \ \frac{V_{th}}{V_m} < \alpha \le 1 \end{aligned} \quad (15)$$

$$\begin{aligned} T_{down}^{m} &= \frac{mV_m}{R}(mr)^{n(m)} \\ &\quad + (M-m)\frac{\alpha V_m}{R} + T_{res}, \ \frac{V_{th}}{V_m} < \alpha \le 1 \end{aligned} \quad (16)$$

From Equations (15) and (16), we can see that the migration time increases with the growth of *m*, and the downtime decreases with the growth of *m*. Assuming $m_1 < m_2$ and $n(m) = \log_{mr}(V_{th}/V_m)$, we have:

$$\begin{aligned} T_{mig}^{m_1} - T_{mig}^{m_2} &= \frac{(m_2-m_1)((m_1+m_2-m_1m_2r)rV_{th}-V_m)}{R(1-m_1r)(1-m_2r)} \\ &\quad + \frac{\alpha(m_2-m_1)(1-m_1r)(1-m_2r)V_m}{R(1-m_1r)(1-m_2r)} \end{aligned} ,$$

Since $\frac{\alpha(m_2-m_1)(1-m_1r)(1-m_2r)V_m}{R(1-m_1r)(1-m_2r)} > 0$, when a=1, $T_{mig}^{m_1} - T_{mig}^{m_2}$ achieves the maximum value.

$$T_{mig}^{m_1} - T_{mig}^{m_2} \le \frac{(m_2-m_1)((m_1+m_2-m_1m_2r)(rV_{th}-rV_m)}{R(1-m_1r)(1-m_2r)} < 0$$

Since $0 < Mr < 1$, $0 < m_1r < 1$ and $0 < m_1m_2r < m_1$, hence:

$$m_1 + m_2 - m_1m_2r > m_1 + m_2 - m_1 > m_2 > 0 .$$

When m =1, the m mixed migration strategy turns into the improved serial migration strategy; whereas when m =M, the m mixed migration strategy turns into the parallel migration strategy. Therefore, when $1 \le m_1 < m_2 \le M$, we have:

$$T_{mig}^{s1} = T_{mig}^{1} \le T_{mig}^{m_1} < T_{mig}^{m_2} \le T_{mig}^{M} = T_{mig}^{p} . \quad (17)$$

Similarly,

$$T_{down}^{m_1} - T_{down}^{m_2} = \frac{(m_2 - m_1)(\alpha V_m - V_{th})}{R} > 0 .$$

Where α V_m is the VM memory requiring transmission in the post-copy migration process. So when $\alpha V_m > V_{th}$ and $1 \leqslant m_1 < m_2 \leqslant M$, we have:

$$T_{down}^{p} = T_{down}^{M} \le T_{down}^{m_2} < T_{down}^{m_1} \le T_{down}^{1} = T_{down}^{s1} . \quad (18)$$

From (17) and (18) we can see that the migration time increases with the growth of m, and the downtime decreases with the growth of m.

For a given maximum downtime $T_{down}^{\max}$ and $n(m)=\log_{mr}(V_{th} / V_m)$, we can obtain the inverse solution of m through Equation (16), and m computed as in Equation (19). Note that, because the downtime of the m mixed migration strategy decreases with the growth of m, so when m=1, the downtime of the m mixed migration strategy achieves the maximum value; when m=M, the downtime of the m mixed migration strategy achieves the minimum value. Thus, the maximum downtime $T_{down}^{\max}$ must fall into [T_{down}^{M}, T_{down}^{1}].

$$m = \left\lceil \frac{RT_{down}^{\max} - RT_{res} - \alpha M V_m}{V_{th} - \alpha V_m} \right\rceil, \ \frac{V_{th}}{V_m} < \alpha \le 1 \quad (19)$$

3 MODELING THE SERVICE FUNCTION CHAINING MIGRATION

In this paper, the migration request is the service function chaining. We assume that these VMs are migrated by uniformly using bandwidth resources of each link. Therefore, these migration requests can share the bandwidth resources of the underlying network. According to these assumptions, we can establish two kinds of scheduling models for the migration request by using the M/M/C/C queuing model and M/M/C queuing model.

3.1 Migration Blocking Rate

In the M/M/C/C queuing model, there are C service channels, and the system can accommodate up to C customers. We assume that the total bandwidth of the entire underlying network is B, and the requested bandwidth of each migration request is R. Thus, the backbone network can simultaneously provide services for C requests, C=$\lfloor B / R \rfloor$. In the M/M/C/C queuing model, the input stream is a Poisson process with arrival rate λ and service time is independent and identical exponential distribution. The requests arrive one by one, and each arrival request is independent of each other. The service time of each request is T, so the service rate of sub channels is $u = 1/T$. The service time T is equal to the average migration time of each request.

$$T = \sum_{m=1}^{M} \alpha_m T_{mig}^{m} \quad (20)$$

Where, α_m is the probability of using the m mixed strategy for migrating the service function chaining.

When the number of migration requests is C, there is not enough bandwidth for the next migration request. When the next migration request arrives, it will be blocked. Therefore, the scheduling model of these arrival requests can be modeled as the M/M/C/C queuing model. According to these above assumptions, we can calculate the blocking rate as follows.

$$B = p_c = \frac{(T\lambda)^c}{c!} \frac{1}{1 + \sum_{i=1}^{C} \frac{(T\lambda)^i}{i!}} \quad (21)$$

3.2 Migration Waiting Time

Since the network provider has reserved resources for the VNFs before migration, if the migration request is blocked, the reserved resources will be released. And the blocked migration request will be resubmitted to datacenters at a later time. However, the migration request may be blocked again and thus the SLAs may be violated, resulting in a penalty to the service provider. Thus, instead of simply rejecting a request, service providers may place blocked migration request in a queue and try to service it later. The service provider has to pay attention to the waiting time and ensure that the tolerable waiting time in SLAs is satisfied to avoid a penalty. Accordingly, we develop a model based on the M/M/C queuing model to quantify the waiting time in this subsection.

In the M/M/C queuing model, there are *C* service channels, and the queue length is unlimited. If there are n ($1 \leqslant n \leqslant C$) migration requests, then n service channels have to work for serving the n migration requests. When the number of the migration requests exceeds *C* (i.e., $n > C$), then (n-*C*) migration requests will be placed into the waiting queue. An example for the M/M/C queuing and waiting model is shown in Figure 1.

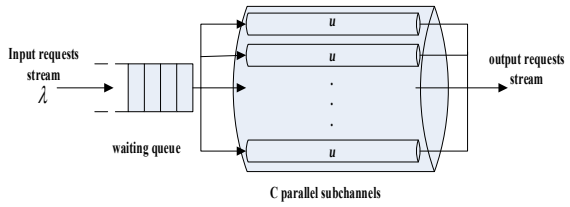

Figure 1: The M/M/C queuing model.

The other assumptions of the M/M/C queuing model are similar with that of M/M/C/C queuing model. According to these assumptions mentioned above, we have the average time of queuing and waiting as follows.

We have:

$$P_0 = \left[\frac{(T\lambda)^C}{C!} \frac{1}{1 - T\lambda / C} + \sum_{i=0}^{C-1} \frac{(T\lambda)^i}{i!} \right]^{-1},$$

$$P_n = \begin{cases} \frac{1}{n!}(T\lambda)^n P_0, & n < C \\ \frac{1}{C!C^{n-C}}(T\lambda)^n P_0, & n \geq C \end{cases}.$$

The average length of waiting queue is the number of requests that are waiting for service. Thus, the average length of waiting queue can be calculated as in Equation (22).

$$L_q = \sum_{n=C+1}^{\infty} (n - C) P_n = \frac{P_0}{C!}(T\lambda)^C \frac{T\lambda / C}{(1 - T\lambda / C)^2} \quad (22)$$

The average waiting time is the average time waiting for service before being serviced, and can be calculated as in Equation (23).

$$W_q = L_q / \lambda \quad (23)$$

4 NUMERICAL RESULTS

In this section, we numerically compare and analyze the performance of the serial migration strategy, the improved serial migration strategy, the parallel migration strategy and the *m* mixed migration strategy of the service function chaining. Similar to (Kim et al, 2015), we set the values of parameters as follows: α=1 ; *M*=5; *R*=1Gbps; *B*=10Gbps; V_m=1GB; *P*=4KB; *D*=2500pps; *r*=0.08; V_{th}=0.1GB; n_{max}=8; T_{res}=0.1s; T_{mig}^1 = 40.64s; T_{mig}^2 = 42.74s; T_{mig}^3 =46.82s; T_{mig}^4 = 53.55s; T_{mig}^5 = 64s.

Figure 2 (a), (b) show the migration time and downtime of the serial migration, the improved serial migration and the parallel migration strategies as a function of the ratio of the dirtying rate to the transmission rate. Fig.2(c) presents the migration time and the downtime of the three compare strategies, as a function of the number of VNFs in the service function chaining (i.e., *M*). From Fig.2, we can see that the numerical results and our theoretical analysis are consistent. The migration time of the serial migration strategy is shorter than the parallel migration strategy; and the downtime of the serial migration strategy is longer than the parallel migration strategy. When α = 1, the migration time and the downtime of the improved serial migration strategy are also shorter than the serial migration strategy. Whereas when α= 0.7, these advantages of the improved serial migration strategy are more notable. The migration time of the improved serial migration strategy is shorter than that of the parallel migration strategy, and the downtime of the improved serial migration is longer than that of the parallel migration. Therefore, different migration strategy may be applicable for different performance requirements. For example, if service providers consider minimizing downtime as the optimization objective, then they should use the parallel migration strategy; if service providers consider minimizing migration time as the optimization objective, then they should use the improved serial migration strategy. Based on the improved serial migration strategy and the parallel migration strategy, we propose the *m* mixed migration strategy, for further improving the performance.

Figure 3 shows the migration time of the *m* mixed migration strategies when *M* varies from 4 to 6. We can see that a larger *m* leads to a longer migration time. This is because that larger *m* means more VNFs need to be parallel migrated and the migration time of the parallel migration strategies is longer than that of the improved serial migration strategy. Thus, the migration time of the *m* mixed migration strategy increases with the growth of the value of *m*. Thus, service providers who want to reduce the migration time, should try to lower the ratio of dirtying rate to transmission rate or choose the *m* mixed migration strategy with smaller *m*.

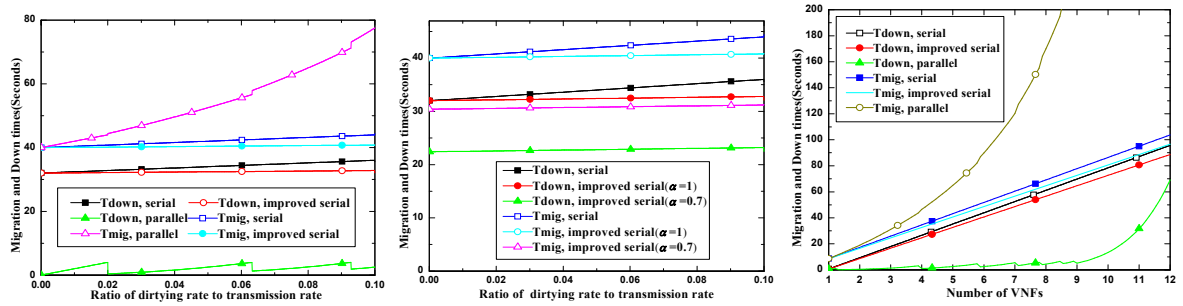

Figure 2: Comparisons on Migration time and downtime.

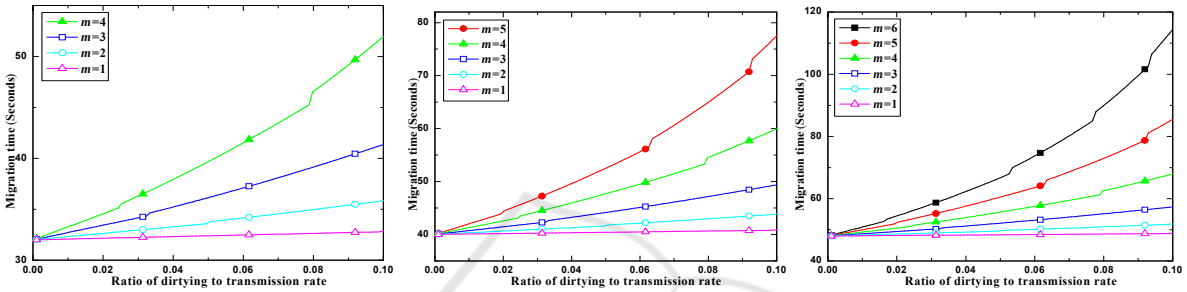

Figure 3: Migration time as a function of ratio of dirtying rate to transmission rate for different *m* mixed migration strategies.

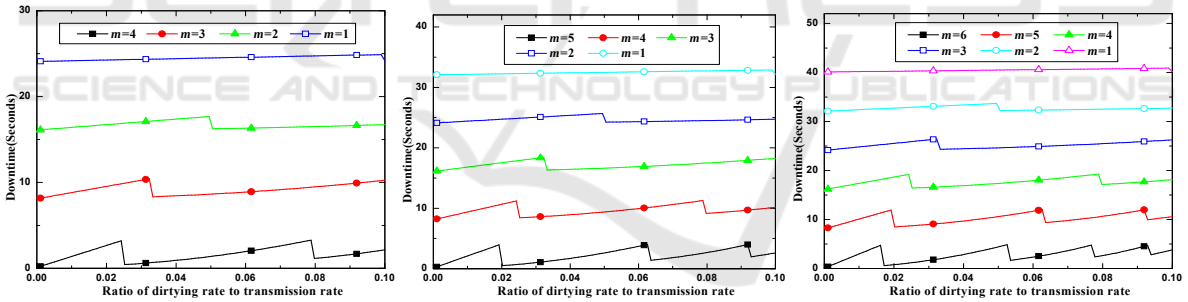

Figure 4: Downtime as a function of ratio of dirtying rate to transmission rate for different *m* mixed migration strategies.

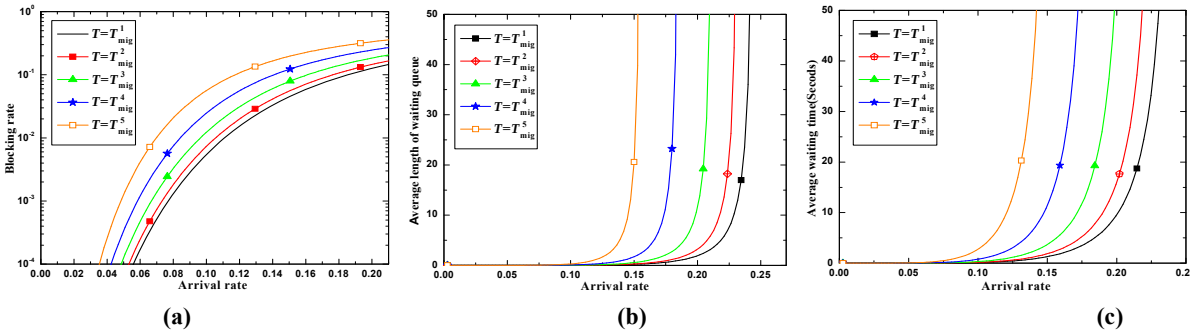

Figure 5: Blocking rate, average waiting queue length, average waiting time as functions of arrival rate for different *m* mixed migration strategies.

Figure 4 demonstrates the downtime of the *m* mixed migration strategies under different values of *M*. We can see that the downtime of *m* mixed migration strategy decreases with the increase in *m*. This is because larger *m* means more VMs need to be parallel migrated and the downtime of the parallel migration strategy is shorter than that of the improved serial migration strategy. Therefore, if service providers intend to reduce downtime, they should choose the *m* mixed migration strategy with larger *m*.

Figure 5 compares the blocking rate, the average length of waiting queue and the average waiting time of each migration request achieved by the *m* mixed migration strategy under various scenarios. We note that since longer average migration time leads to a longer service time, hence, the blocking rate, the average length of waiting queue and the average waiting time all increase with the growth in the value of *T*.

5 CONCLUSIONS

In this paper, we study how to live migrate the service function chain in network function virtualization environment. We present the efficient strategies for the problem of service function chaining migration. Through our proposed methods, the service providers can efficiently migrate the service function chaining while lowering the constraints of the migration time and downtime.

ACKNOWLEDGMENT

This work was partially supported by Natural Science Foundation of China (61571098), Guangdong Science and Technology Foundation (2013A040600001, 2013B090200004, 2014B090901007, 2015A040404001, 2013B040300001).

REFERENCES

Mijumbi, R., Serrat, J., Gorricho, J., Bouten, N., Turck, F., D., Boutaba, R., 2016. Network Function Virtualization: State-of-the-art and Research Challenges. *IEEE Communications Surveys & Tutorials*, 1(18), 236-262.

Bari, M. F., Chowdhury, S. R., Ahmed, R., Boutaba, R., 2015. On Orchestrating Virtual Network Functions. *International Conference on Network and Service Management (CNSM)*, 50-56.

Wu, J., Zhang, Z., Hong, Y., Wen, Y., 2015. Cloud radio access network (C-RAN): a primer. *IEEE Network*, 1(29), 35-41.

Bondan, L., Santos, C., R., P., D., Granville, L., Z., 2014. Management Requirements for ClickOS-based Network Function Virtualization. *International Conference on Network and Service Management (CNSM) and Workshop*, 447-450.

Gember-Jacobson, A., Viswanathan, R., Prakash, C., Grandl, R., Khalid, J., Das, S., Akella, A., 2014. OpenNF: enabling innovation in network function control. *ACM SIGCOMM*, 163-174.

Mehraghdam, S., Keller, M., Karl, H., 2014. Specifying and Placing Chains of Virtual Network Functions. *IEEE 3rd International Conference on Cloud Networking (CloudNet)*, 7-13.

Xia, M., Shirazipour, M., Zhang, Y., Green, H., Takacs, A., 2015. Optical Service Chaining for Network Function Virtualization. *IEEE Communications Magazine*, 4(53), 52-158.

Li, X., Qian, C., 2015. Low-Complexity Multi-Resource Packet Scheduling for Network Function Virtualization. *IEEE INFOCOM*, 1400-1408.

Cohen, R., Lewin-Eytan, L., Naor, J., Raz, D., 2015. Near Optimal Placement of Virtual Network Functions. *IEEE INFOCOM*, 1346-1354.

Kim, T., Kim, S., Lee, K., Park, S., 2015. A QoS Assured Network Service Chaining Algorithm in Network Function Virtualization Architecture. *IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid)*, 1221-1224.

Tso, F., P., Hamilton, G., Oikonomou, K., Pezaros, D., p., 2013. Implementing Scalable, Network-Aware Virtual Machine Migration for Cloud Datacenters. *IEEE Sixth International Conference on Cloud Computing*, 557-564.

Shrivastava, V., Zerfos, P., Lee, K., Jamjoom, H., Liu, Y., Banerjee, S., 2011. Application-aware virtual machine migration in datacenters. *IEEE INFOCOM*, 66-70.

Cerroni, W., 2014. Multiple Virtual Machine Live Migration in Federated Cloud Systems. *IEEE INFOCOM Workshop on Cross-Cloud Systems*, 25-30.

Callegati, F., Cerroni, W., 2013. Live Migration of Virtualized Edge Networks: Analytical Modeling and Performance Evaluation. *IEEE Workshop on Software Defined Networks for Future Networks and Services (SDN4FNS 2013)*, 1-6.

Hines, M., R., Deshpande, U., Gopalan, K., 2009. Post-copy live migration of virtual machines. *ACM Operating Systems Review*, 43(3), 14-26.