Fairness-Aware VNF Sharing and Rate Coordination for High Efficient Service Scheduling

Bo Yi, *Member, IEEE*, Xingwei Wang, Min Huang, Sajal K. Das, *Fellow, IEEE*, and Keqin Li, *Fellow, IEEE*

Abstract—Network service provisioning becomes flexible and programmable with the help of Network Function Virfitualization (NFV), since NFV abstracts various service functions into software components called Virtual Network Function (VNF) and VNFs can be flexibly and quickly composed to form new services. It is commonly known that sharing the same VNF among different services can improve the resource utilization. However, we should be aware that such sharing also leads to serious resource preemption. In addition, VNF sharing aggravates the generation of the performance bottleneck, which then causes the rate mismatch problem between the upstream and downstream VNFs belonging to the same service chain. In this article, we propose a dynamic and flexible algorithm to jointly address the VNF sharing resource allocation and the rate coordination between the upstream and downstream VNFs. Specifically, 1) the VNFs are shared among different service chains with a fairness factor considered for the purpose of reducing the resource preemption probability and improving the resource utilization; 2) the backpressure indicator of each VNF is defined to judge its pressure condition, based on which we can dynamically adjust the processing rates between it and its downstream or upstream VNFs by maximizing the idle resource utilization. The experimental results indicate that the proposed algorithm outperforms the other methods in terms of the average delay, the flow completion time, the throughput and the backlog, etc. Meanwhile, the proposed algorithm achieves more stable performance than the other methods.

Index Terms—Network function virtualization, performance bottleneck, rate coordination, service chain, traffic scheduling

1 Introduction

Network functions or middleboxes (e.g., firewall, network address translator, etc.) that used to be coupled with purpose-built hardware are now implemented in software due to the emergence of Network Function Virtualization (NFV) [1]. These software based functions are called Virtual Network Functions (VNF) and run on top of Commercial-Off-The-Shelf (COTS) hardware [2]. The separation of VNF and COTS hardware introduces a flexible and programmable way for service provisioning by chaining different VNF instances in order. Thus, the traffic of service chains should be processed by all the required VNFs before being forwarded to the destination [3]. For example, as shown in Fig. 1, a simple case of NFV service chain in practice is illustrated, where two service chains are sent from the host to the server. In particular, the traffic of the first service chain needs to traverse the network functions of the intrusion detection system and firewall due to security reasons, while the second service chain needs to traverse the network function of load balancer due to the large amount of traffic. It is aware that the three network functions are actually VNFs and implemented in software in the scenario of NFV, so that they can be deployed and offloaded on demand. Namely, once the targets of service chains 1 and 2 are changed, we can flexibly and quickly deploy the demanded network functions and offload the original ones, so that both service quality and resource waste can be avoided efficiently.

Generally, one VNF instance is independently used by one service chain for the purpose of avoiding the resource conflict [4]. In this case, service providers do not need to worry about the interaction between different service chains when deploying them. However, such design would finally generate a lot of VNF instances that are several times the number of services. On one hand, some service chains may not be able to fully utilize the VNF capacities, which results in a great deal of resource waste [5]. For example, as shown in Fig. 2a, service chains *s1* and *s2* both require a VNF instance

- *Bo Yi and Xingwei Wang are with the College of Computer Science and Engineering, Northeastern University, Shenyang 110169, China. E-mail: yiboboscar@gmail.com, wangxw@mail.neu.edu.cn.*
- *Min Huang is with the College of Information Science and Engineering, Northeastern University, Shenyang 110819, China. E-mail: mhuang@mail.neu.edu.cn.*
- *Sajal K. Das is with the Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409 USA. E-mail: sdas@mst.edu.*
- *Keqin Li is with the Department of Computer Science, State University of New York, New York 12561 USA. E-mail: lik@newpaltz.edu.*

Manuscript received 9 February 2022; revised 6 August 2022; accepted 12 August 2022. Date of publication 17 August 2022; date of current version 26 August 2022.
This work was supported in part by the National Key R&D Program of China under Grant 2019YFB1802800, in par by the National Natural Science Foundation of China under Grants 62002055, 62032013 and 61872073, in part by China Postdoctoral Science Foundation under Grant 2020M680972, in part by the Postdoctoral Research Fund of Northeastern University of China under Grant 20200103, and in part by the Fundamental Research Funds for the Central Universities of China under Grant N2016012.
(Corresponding author: Xingwei Wang.)
Recommended for acceptance by A. C. Zhou.
Digital Object Identifier no. 10.1109/TPDS.2022.3199392

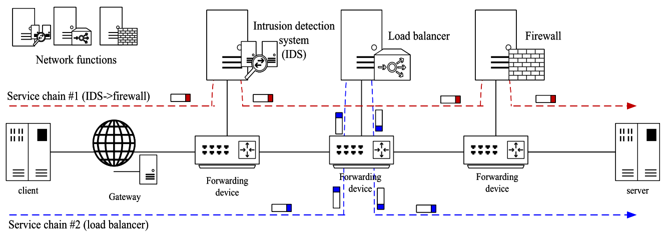


Fig. 1. Example of NFV service chain in practice.

(i.e., *VNF1*) which is not work-conserving, because their packets arrive at different times. Assuming that their packets arrive at completely different times as indicated in Fig. 2a, *s1* and *s2* can be served using one *VNF1* instance instead of two. On the other hand, the more VNFs placed on the same node, the more resource fragments will be required among them for security protection, which also leads to a lot of resource waste [7]. Taking these conditions into consideration, proper VNF sharing strategies among service chains should be carefully designed, especially in the resource-constrained scenario.

In addition, the environment used to host VNFs can be Virtual Machines (VM) or containers (e.g., Docker) [8]. Due to the limitation of the physical node capacity, VMs or containers hosting VNFs may have different processing rates. As explained, one service chain is actually composed of many VNFs. The mismatch of processing rates among VNFs belonging to the same service chain would cause serious performance degradation [7]. For example, as shown in Fig. 2b, the processing rate of the upstream VNF is $v1$ and that of the downstream VNF is $v2$, where $v1 \gg v2$. When more and more packets arrive at the downstream VNF from the upstream VNF, these packets will be dropped if the buffer of the downstream VNF overflows, which naturally results in meaningless processing work done by the upstream VNF. The CPU resource wasted in the upstream VNF could be better utilized to process packets of other flows. In this case, the downstream VNF becomes the bottleneck VNF which greatly degrades the service performance [9]. Hence, dynamic traffic scheduling for high efficient service provision is urgently needed, especially in the scenario where multiple VNFs compete for the same amount of resouces.

Existing research did not reveal the inner-relationship between the above challenges [10], [11], and they only focused on one aspect of them. However, the problems caused by VNF sharing and the bottleneck VNF are actually highly correlated [5]. Although the VNF sharing is enabled to improve resource utilization, the more services sharing the same VNF, the larger probability the performance bottleneck will be generated. Besides, VNF sharing means that more packets should be processed [12] on one VNF. Considering the fact that these packets may belong to different service chains, proper scheduling methods should be developed to handle them in case of overflow if the VNF is set to a finite buffer. For example, in Fig. 2c, the processing rates of the two VNFs are $v1$ and $v2$ respectively, and $v1 = v2$ in this case. It is noted that the downstream VNF is shared by three service chains (i.e., *s1*, *s2*, *s3*), while the upstream VNF is used only by one service chain (i.e., *s1*). Thus, the downstream VNF needs to process packets of three flows, which leads to the mismatch between the processing rates of the two VNFs in *s1* and this makes the downstream VNF a bottleneck.

Due to the inner-relationship between the challenges caused by the VNF sharing and the processing rate mismatch, they can be combined and addressed together for the purpose of improving not only resource utilization but also service performance. This combined problem can be described as: *Given a VNF instance, when there comes a flow, we should 1) decide how many processing resources should be allocated to this flow; 2) how to cooperate the rates between the upstream and downstream VNFs during its long run.*

Targeting on solving such an integration problem, we propose an efficient resource allocation and traffic scheduling algorithm to achieve high-performance services, which takes both the challenges caused by the VNF sharing and the processing rate mismatch into consideration. The main contributions of this work are as follows:

- First, we explore the inner-relationship between the VNF sharing and the processing rate mismatch during the service provisioning. Then, we combine the VNF sharing and the rate processing to derive a novel problem which is formulated as a mathematical model leveraging the concept of network calculus. In particular, the fairness of services is introduced to make a balance between the fairness and the performance indicator.
- Second, we propose a dynamic algorithm to address the VNF sharing and the rate mismatch problem. On one hand, the VNF sharing is addressed by a fair service capacity allocation strategy among services sharing this VNF, during which the fairness factor is considered. On the other hand, the processing rate mismatch is addressed by dynamically maximizing the idle resource as much as possible, and this process is triggered by the VNF backpressure indicator.
- Third, based on the formulated mathematical model and the proposed algorithm, three performance bounds (i.e., the throughput, backlog and delay) of the VNF constituted service chains are analyzed and provided to help better explain the benefits of the proposed algorithm.

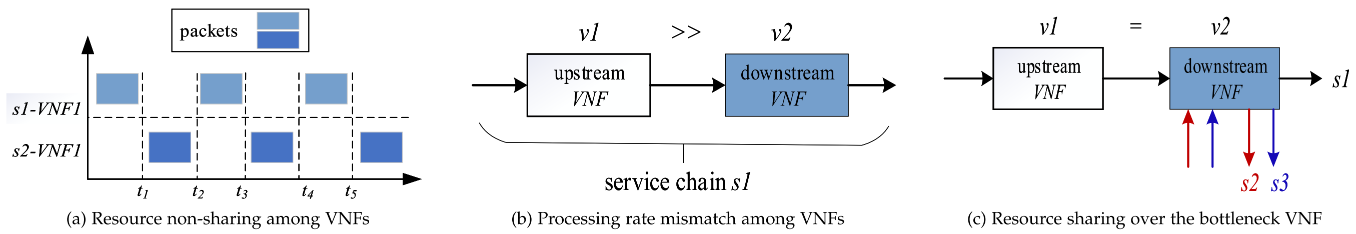

packets
s1-VNF1
s2-VNF1
t1
t2
t3
t4
t5
(a) Resource non-sharing among VNFs
v1
>>
v2
upstream VNF
downstream VNF
service chain s1
(b) Processing rate mismatch among VNFs
v1
=
v2
upstream VNF
downstream VNF
s1
s2
s3
(c) Resource sharing over the bottleneck VNF

Fig. 2. Challenge and motivation illustration.

The rest of this work are organized as follows. Section 2 overviews the related work. Section 3 formulates a mathematical model for the problem to be solved, while Section 4 proposes the algorithm to address this problem. Section 5 presents the performance evaluation and shows the experimental results. The conclusion is drawn in Section 6.

2 Related Work

The VNF sharing and processing rate mismatch are combined as one problem, that is, how to properly schedule packets of different service chains on the same VNF to achieve high resource utilization and harmonize the processing rates of different VNFs. In this regard, we focus on exploring and comparing the state-of-the-art researches studying this problem.

According to the results of literature survey, most of the existing studies only focused on solving one aspect, which meant that either the schedule issue caused by VNF sharing or the bottleneck issue caused by VNF processing rate mismatch was addressed. For example, Yi et al. [5] assumed the VNF processing rates to be the same and proposed a fair approach to address the resource allocation problem caused by sharing VNFs among service chains, while Kulkarni, et al. [7] mainly emphasized the network bottleneck resulted from the processing rate mismatch among VNFs and proposed a user space process scheduling abstraction to handle the function schedule and resource allocation. In addition, many other studies may present the problem in other ways. For instance, although Xuan et al. [13] proposed to address the VNF deployment and the service routing problem, it actually used the data center to offer some specific VNFs. Abstracting each data center as one node, the resource sharing existed among some VNFs, which naturally resulted in the traffic scheduling problem. In addition, Bringhenti et al. [14] focused on the time-sensitive flows and proposed a prototype framework to test the service performance on realistic use cases. Mijumbi et al. [15] and Alameddine et al. [16] proposed to address the VNF mapping problem. Mijumbi et al. [15] solved the VNF mapping and scheduling problem based on three greedy algorithms, while Alameddine et al. [16] considered the interplay between VNFs by using a column generation approach. Nevertheless, for both of them, the VNFs of different service chains were mapped on the same node, so that they actually addressed the issue caused by resource sharing.

There were also many studies proposed to address the scheduling problem from many other perspectives. For example, Liu et al. [17] broke the assumption that the network topology and resource requirement of the service chain did not change, and then the author studied how to dynamically schedule and adjust the resource allocation when the service chain changes. Nevertheless, it ultimately came down to a compromise problem between reducing the resource consumption and minimizing the scheduling overhead. Similarly, Gao et al. [18] tried to make a trade-off between the cost and delay when addressing the VNF placement and scheduling problem in clouds, during which more practical aspects such as an additional latency incurred by booting a VM and installing a VNF instance, were also considered. Besides, Masoud et al. [19] introduced new constraints for the network scheduling problem in NFV, which included the priority dependency between VNFs and the minimization of the VNF mapping blocking rate. Qu et al. [20] and M. Wang et al. [21] both focused on optimizing the delay of the NFV scheduling problem. The differences were that Qu et al. [20] intended to jointly reduce the delay from the perspectives of node queues and communication channels, while Wang et al. [21] intended to reduce the transmission delay only. Gu et al. [22] took the network utility maximization and the dynamic rate control into consideration when addressing the flow scheduling problem, while Pham et al. [23] formulated the flow scheduling problem as a matching game model for solving. It is noted that Gu et al. [22] assumed that the processing rates of VNFs were different, so that the dynamic rate control was required.

With the popularity of the machine learning technologies, many researches were proposed to achieve the intelligent VNF mapping and scheduling process. For example, Li et al. [24] used the Reinforcement Learning (RL) algorithm to address the VNF mapping and scheduling problem. The objective of this reference was to minimize the overall completion time of all services while satisfying their different end-to-end delay requirements. In particular, the RL model was used to learn the best scheduling policy by continuously interacting with the environment and determining the variable actions at each decision-making state. Gu et al. [25] combined the technologies of deep learning and RL (DRL) to address the VNF mapping and scheduling problem. The difference from [24] was that Gu et al. [25] intended to maximize the revenue. However, the training process of the intelligent algorithms usually suffered from a slow convergence challenge. In this regard, many optimization models and heuristic solutions were leveraged to guide and accelerate the training process by avoiding the blind action exploration. For instance, Zhang et al. [26], Huang et al. [27] and Kang et al. [28] tried to optimize such a process by leveraging the genetic and heuristic optimization algorithms respectively.

The VNF scheduling problem also occurs in many other application scenarios. For instance, Xuan et al. [13] addressed the resource sharing problem in optical data centers by building an optimal mathematical model, while Sun et al. [29] and Bhamare et al. [30] intended to address the resource scheduling problem in the cloud computing environment on the basis of the Markov chain model. This problem was also introduced into the Internet of Vehicles (IoV) environment, which was investigated and addressed by Li et al. [31]. However, the VNF mapping and scheduling were addressed separately using two Tabu search (TS)-based algorithms. Cao et al. [32] proposed to address the VNF embedding and scheduling problem in SDN/NFV-enabled networks, which was in fact the problem of service provisioning. Despite of this, the resource scheduling was not properly handled in the first round of the service provisioning and thus the re-scheduling process was required in the second round of the service provisioning. Likewise, Wang et al. [33] studied the service function chaining in SDN/NFV-enabled networks in three phases, that is, VNF composition, VNF forwarding graph embedding and VNF scheduling, which meant that the VNF scheduling should be under the constraints of the first two phases. Moreover, Ran et al. [34] specifically focused on the dynamic virtual measurement function scheduling in the software-oriented measurement environment due to the stochastic nature of measurement demands. Alameddine et al. [35] and Ren et al. [36] focused on scheduling VNFs for achieving ultra-reliable and ultra-low latency services in the integration fields of 5 G and edge computing, while Promwongsa et al. [37] proposed to address the VNF scheduling problem in the next 6 G networks. The difference was that Promwongsa et al. [37] focused more on meeting the stringent service deadline while the former two focused more on achieving low latency.

Based on the above analysis, it is aware that the resource sharing and the VNF processing rate mismatch focus on different aspects in the service chain provisioning process, and most of the state-of-the-art researches address either one of them solely. *However, we discover that the solutions of the VNF sharing and the processing rate mismatch have a great impact on the performance achieved by each other, since they are both associated with the resource. For example, the resource sharing result may alter the decision of the processing rate coordination, and vice versa. In order to achieve much better performance, we deeply analyze the inner-relationship between them and then combine them as one unique problem which is addressed in this work.*

3 Network Model and Problem Formulation

In this section, we first introduce the network model, based on which the problem is formulated. The major notations used in this paper are summarized in Table 1.

TABLE 1
Notations

Notation	Meaning
Model and objective	
$G(N, L)$	Physical network with a node set N and a link set L.
R_n, R_l	The capacities of physical node $n \in N$ and link $l \in L$.
$G'(V, E)$	Virtual network with a VNF set V and a virtual link set E.
x_n^v	The mapping relationship between the physical node and the virtual node.
y_l^e	The mapping relationship between physical links and virtual links.
Ψ, ψ	The set of service chains and one service belonging to this set.
$A^v(t), A^\psi(t)$	The cumulative amount of traffic data arriving at the nodes of v and ψ during $[0, t]$.
$D^v(t), D^\psi(t)$	The cumulative amount of traffic data leaving the nodes of v and ψ during $[0, t]$.
$\overline{z}_{i,v}^\psi$	The bool variable used to indicate whether the ith VNF of ψ is supplied by the VNF instance v.
$z_{i,e}^\psi$	The bool variable used to indicate whether the ith VNF and the $i+1$th VNF of ψ are connected by the link e.
$S^v(t)$	The service capacity of v during $[0, t]$.
F^ψ, f_i^ψ	The VNF set of the service request ψ and the ith VNF of ψ.
S_i^ψ	The service capacity of the ith VNF requested by ψ.
U^ψ, α^ψ	The utilify function and the fairness tuning parameter of ψ.
Algorithm	
$mean^\psi$	The mean value of processing rates of VNFs of ψ.
var^ψ	The variance of the processing rates of different VNFs belonging to ψ.
θ	The threshold used to judge whether a service chain is in the rate mismatch condition or not.
$\mathbb{M}$	It indicates a set, in which all the services are sharing the same VNF instance.
$\mathbb{L}$	It indicates a set, in which all the services are work-conserving, and $\mathbb{L} \subset \mathbb{M}$
$\mathbb{N}$	It indicates a set, in which all the services cannot fully utilize the allocated service capacities, and $\mathbb{N} \subset \mathbb{M}$
ρ, σ	The default traffic arriving rate and the burst parameters.
ϖ^v	The indicator of backpressure caused by v.
ε	The offset time between the arriving time of the first packet and the leaving time of the last packet.
$\mathfrak{T}^\psi, \mathfrak{B}^\psi, \mathfrak{D}^\psi$	The throughput, backlog and delay bound of ψ.

3.1 Network

The physical network is abstracted as the undirected graph $G(N, L)$, where N is the set of commercial-off-the-shelf physical nodes and L is the set of physical links. For any physical node $n \in N$, it is assumed to have both the switching and server capabilities. The available CPU resource of the node n is denoted by R_n and the available bandwidth resource of each link $l \in L$ is denoted by R_l.

The physical resources need to be virtualized in order to deal with various network dynamics, for example, increasing or decreasing the resources allocated to VNFs when network condition changes. Thus, we denote the virtual network (i.e., VNF forwarding graph) by $G'(V, E)$, where V is the set of VNF instances and E is the set of virtual links among these VNFs. In particular, the mapping relationship between the physical network and the virtual network is described by two binary variables x_n^v and y_l^e, as follows:

$$x_n^v, y_l^e \in \{0, 1\}, \forall v \in V, n \in N, e \in E, l \in L, \tag{1}$$

where the value of 1 indicates that v is mapped on n or e is mapped on l. Otherwise not.

3.2 VNF and Flow Model

Given any VNF instance $v \in V$, the cumulative amount of traffic data arriving at v during the time interval $[\tau, t]$ is denoted by $A^v(\tau, t)$, where $0 \leq \tau < t$. In particular, $A^v(\tau, t) = A^v(t)$ if $\tau = 0$. Note that, the cumulative function is non-negative and non-decreasing, thus, we have

$$A^v(\tau, t) = A^v(t) - A^v(\tau). \tag{2}$$

Likewise, we denote the cumulative amount of traffic data departing v by $D^v(t)$ and $D^v(\tau, t)$ respectively. Apparently, $D^v(t) \leq A^v(t)$ since the amount of departing traffic cannot exceed the amount of arriving traffic. Now, by denoting the service capacity of v during $[0, t]$ by $S^v(t)$, we can formulate the relationship between $A^v(t)$ and $D^v(t)$ by the following lemma:

Lemma 1: Given any work-conserving VNF v with the service capacity $S^v(t)$ fully utilized, the relationship between its traffic arriving function $A^v(t)$ and departing function $D^v(t)$ is as follows:

$$D^v(t) = \min_{\tau \in [0,t]} \{A^v(\tau) + S^v(\tau, t)\}. \tag{3}$$

Proof: Given the work-conserving VNF v with two time instances τ and $t(> \tau)$, we assume that τ and t are in the same busy period. Then, the entire service capacity of v will be consumed and it follows that

$$D^v(t) = D^v(\tau) + S^v(\tau, t), \tag{4}$$

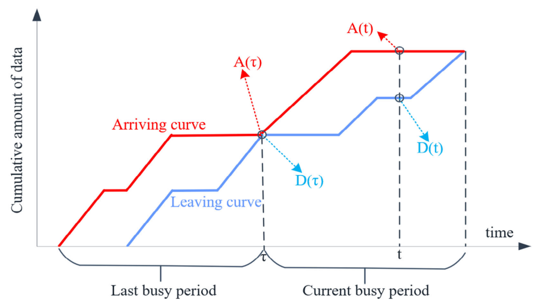


Fig. 3. A simple illustration diagram.

where $D^v(\tau)$ is the cumulative amount of traffic data leaving v at the time τ, and $S^v(\tau,t)$ is the amount of traffic data processed by v during $[\tau,t]$.

Since τ and t are assumed to be in the same busy period, we can let τ be the ending time of the last busy period (i.e., the beginning time of this busy period) before the time t. In this condition, we have

$$D^v(\tau)=A^v(\tau), \tag{5}$$

which means that the amount of traffic data arriving at v during $(0,\tau]$ is processed by v at the time τ, and this situation is illustrated in Fig. 3.

Now, substituting (5) into (4) and we have

$$D^v(t)=A^v(\tau)+S^v(\tau,t). \tag{6}$$

Since the value of τ is generally unknown, (6) is transformed into an estimation form as follows:

$$D^v(t)\geq \min_{\tau\in[0,t]}\{A^v(\tau)+S^v(\tau,t)\}. \tag{7}$$

However, as shown in Fig. 3, we can observe that τ may not always be the exact beginning time of the current busy period. Then, a more general expression of (5) is refined as follows:

$$D^v(\tau)\leq A^v(\tau), \tag{8}$$

which means that the amount of traffic data leaving v will not exceed the amount of traffic data arriving at v during the same time period $(0,\tau]$).

Next, substituting (8) into (4), we obtain another case that

$$D^v(t)\leq A^v(\tau)+S^v(\tau,t). \tag{9}$$

Likewise, an estimation form of (9) is obtained as follows:

$$D^v(t)\leq \min_{\tau\in[0,t]}\{A^v(\tau)+S^v(\tau,t)\}. \tag{10}$$

Now, jointly taking (7) and (10) into consideration, we have

$$D^v(t)= \min_{\tau\in[0,t]}\{A^v(\tau)+S^v(\tau,t)\}. \tag{11}$$

Then, lemma 1 is proved.

In addition, according to the format of (3), we can simplify it by using the min-plus convolution theory which replaces the addition operator with the minimum operator and replaces the multiplication operator with the addition operator, as follows:

$$D^v(t)\triangleq(A\otimes S)^v(t), \tag{12}$$

where the symbol of $\triangleq$ means that $D^v(t)$ can be defined as $(A\otimes S)^v(t)$.

3.3 Service Chain Model

The NFV service chain is usually composed of many sequential VNFs. Considering the fact that the traffic has to traverse these VNFs in strict order before reaching the destination, we can formulate the VNF-constituted service chain as a tandem system.

Given any service chain denoted by $\psi\in\Psi$, its VNF requirements can be described by $F^\psi=\{f_i^\psi|i\in[1,|F^\psi|]\}$, where f_i^ψ indicates the ith VNF demanded by ψ. The required VNFs are actually selected from the VNF forwarding graph, so that two boolean variables are defined as follows:

$$\overline{z}_{i,v}^\psi, \underline{z}_{i,e}^\psi\in\{0,1\},\forall v\in V, i\in[1,|F^\psi|], \tag{13}$$

where the value of 1 indicates that the ith VNF of ψ is actually supplied by the VNF instance v, and e is used to connect the ith VNF with the next one. Otherwise not.

According to the VNF traffic model, the cumulative arriving and departing traffics during the time interval $[0,t]$ for any f_i^ψ are denoted by $A_i^\psi(t)$ and $D_i^\psi(t)$ respectively. Note that, the service capacity of f_i^ψ is actually allocated by $v,\forall\overline{z}_{i,v}^\psi=1$. Then, we denote the service capacity of f_i^ψ by $S_i^\psi(t)$. Based on these notations, we define theorem 1 which formulates the concept of the overall service capacity of ψ.

Theorem 1. *Give any ψ with a VNF set requirement F^ψ, if the service capacity allocated to any VNF f_i^ψ is $S_i^\psi(t)$ and the routes between any two sequential VNFs are unobstructed, then, the overall service capacity of ψ is defined as follows:*

$$S^\psi(t)=S_1^\psi(t)\otimes S_2^\psi(t)\otimes\cdots\otimes S_{|F^\psi|}^\psi(t). \tag{14}$$

Proof. Let us consider the first two VNFs of ψ, that is, f_1^ψ and f_2^ψ. According to lemma 1 and (12), we have

$$D_1^\psi(t)=A_1^\psi(t)\otimes S_1^\psi(t). \tag{15}$$

Since the route between f_1^ψ and f_2^ψ is unobstructed, we have

$$A_2^\psi(t)=D_1^\psi(t). \tag{16}$$

Substituting (15) into (16), it follows that:

$$\begin{aligned}&A_2^\psi(t)=A_1^\psi(t)\otimes S_1^\psi(t)\\ \Longrightarrow\ &A_2^\psi(t)\otimes S_2^\psi(t)=A_1^\psi(t)\otimes S_1^\psi(t)\otimes S_2^\psi(t)\\ \Longrightarrow\ &D_2^\psi(t)=A_2^\psi(t)\otimes S_2^\psi(t)=A_1^\psi(t)\otimes S_1^\psi(t)\otimes S_2^\psi(t)\\ \Longrightarrow\ &D_2^\psi(t)=A_1^\psi(t)\otimes\left(S_1^\psi\otimes S_2^\psi\right)(t)\end{aligned} \tag{17}$$

Repeating the process of (17) based on lemma 1, the service chain can be modelled as follows:

$$D_{|F^\psi|}^\psi(t)=A_1^\psi(t)\otimes\left(S_1^\psi\otimes S_2^\psi\otimes\cdots\otimes S_{|F^\psi|}^\psi\right)(t). \tag{18}$$

Since $D_{|F^\psi|}^\psi(t)=A_1^\psi(t)\otimes S^\psi(t)$, theorem 1 is proved.

Now, according to theorem 1, the tandem service model of ψ can be easily expressed by (18). □

3.4 Problem Objective

For convenience, we simplify (18) as $D^{\psi}(t)=(A\otimes S)^{\psi}(t)$, which actually means the total output traffic of ψ during $[0,t]$. Thus, the mean throughput of ψ is $\frac{D^{\psi}(t)}{t}$. Since different services may have different priorities, the fairness factor should also be considered to prevent low-priority services from never being scheduled. In this way, the problem becomes how to make a compromise between the throughput and the fairness to services, which is perfectly consistent with the proportional fairness theory [38] that originally defines the relationship between the fairness and the traffic rate indicator as follows:

$$f^{\alpha}(\chi)=\begin{cases}(1-\alpha)^{-1}(\chi)^{1-\alpha}, & \text{if } \alpha\neq 1\\ \ln(\chi), & \text{if } \alpha=1\end{cases} \tag{19}$$

where α is the fairness parameter and χ is the traffic rate indicator.

Since our objective indicator is the throughput that is proportional to the traffic rate, we can replace χ with the throughput (i.e., $\frac{D^{\psi}(t)}{t}$) to form our utility function for any $\psi\in\Psi$, as follows:

$$\begin{aligned}U^{\psi}\left(\frac{D^{\psi}(t)}{t},\alpha^{\psi}\right)&=f^{\alpha^{\psi}}\left(\frac{D^{\psi}(t)}{t}\right)\\&=\begin{cases}(1-\alpha^{\psi})^{-1}\left(\frac{D^{\psi}(t)}{t}\right)^{1-\alpha^{\psi}}, & \text{if } \alpha^{\psi}\neq 1\\ \ln\left(\frac{D^{\psi}(t)}{t}\right), & \text{if } \alpha^{\psi}=1\end{cases}\end{aligned} \tag{20}$$

where $\alpha^{\psi}(\geq 0)$ is the corresponding fairness tuning parameter. A larger value of α^{ψ} means that we prefer more fairness than throughput.

In particular, when $\alpha^{\psi}=0$, $U^{\psi}(\frac{D^{\psi}(t)}{t},\alpha^{\psi})$ is degenerated to the mean throughput of ψ. Taking the above analysis into consideration, the problem objective is formulated as follows:

$$Maximize:\ \sum_{\psi\in\Psi}U^{\psi}\left(\frac{D^{\psi}(t)}{t},\alpha^{\psi}\right). \tag{21}$$

Nevertheless, many constraints should be followed for the purpose of achieving the above objective. First of all, the total traffic processed on the server n and the link l are limited by their resource capacities, that is:

$$\begin{cases}\sum_{v\in V}\sum_{\psi\in\Psi}\sum_{i\in[1,|F^{\psi}|]}x_n^v\times\overline{z}_{i,v}^{\psi}\times\frac{S_i^{\psi}(t)}{t}\leq R_n,\forall n\in N\\ \sum_{e\in E}\sum_{\psi\in\Psi}\sum_{i\in[1,|F^{\psi}|]}y_l^e\times\underline{z}_{i,e}^{\psi}\times\frac{D_i^{\psi}(t)}{t}\leq R_l,\forall l\in L\end{cases} \tag{22}$$

Besides, one VNF of ψ should only be placed on one physical server, while one virtual link of ψ can be mapped on multiple adjacent physical links, so that the mapping constraints are as follows:

$$\begin{aligned}&\begin{cases}\sum_{n\in N}x_n^v=1\\ \sum_{v\in V}\overline{z}_{i,v}^{\psi}=1\end{cases},\forall v\in V,i\in[1,|F^{\psi}|]\\&\begin{cases}\sum_{l\in L}y_l^e\geq 1\\ \sum_{v\in V}\underline{z}_{i,e}^{\psi}\geq 1\end{cases},\forall e\in E,i\in[1,|F^{\psi}|]\end{aligned} \tag{23}$$

Moreover, due to the dependency relationship between the adjacent VNFs of the same service chain, the input traffic of any ψ is always equal to or larger than the output traffic, as follows:

$$A_1^{\psi}(t)\geq D_1^{\psi}(t)\geq A_2^{\psi}(t)\geq\cdots\geq D_{|F^{\psi}|}^{\psi}(t)\geq 0 \tag{24}$$

4 Proposed Algorithm

In this work, we assume that the mapping relationship between the physical network and the VNF forwarding graph is already built and we focus on how to schedule the arriving packets of different service chains. Specifically, this problem can be refined into: *1) The service capacity of any VNF is limited, which may not be fairly allocated to the service chains sharing this VNF. 2) In addition, the service capacities of the upstream and downstream VNFs belonging to the same service chain may be different, which then causes the VNF processing rate mismatch issue and finally results in a waste of work.* Hence, the VNF sharing scheduling process is used to guarantee that all newly arrived services can be treated fairly in the beginning, while the rate coordination process is used to guarantee the service performance during the long run, both for the purpose of improving the overall resource utilization and throughput. In this way, the VNF sharing scheduling and the rate coordination will not affect each other. Instead, they will support each other to achieve the same goal.

4.1 VNF Share Scheduling

Given any VNF instance v, it is regarded as a work-conserving server and its processing rate is fixed by r^v. In this way, the service capacity it can offer during $[\tau,t]$ is as follows:

$$\begin{aligned}S^v(\tau,t)&=\sum_{\forall\psi\in\Psi}\sum_{i\in[1,|F^{\psi}|]}S_i^{\psi}(\tau,t)\times\overline{z}_{i,v}^{\psi}\\&=r^v(t-\tau),\forall v\in V.\end{aligned} \tag{25}$$

Note that, the service capacity of v should be discretized before it can assign CPU shares among different service chains. As explained, each service is attached with a fairness tuning parameter α^{ψ} to prevent it from never being scheduled. Furthermore, to better achieve the fairness to services, the service size (denoted by ω^{ψ}) should be considered as well. The relationship between the service size and the allocated service capacity can be defined as follows:

Definition 1: given any v shared by two services ψ and φ, if the traffic of ψ is continuously backlogged in v during $[\tau,t]$, we have

$$\frac{S^{\psi}(\tau,t)}{\omega^{\psi}}\geq\frac{S^{\varphi}(\tau,t)}{\omega^{\varphi}}, \tag{26}$$

because ψ is continuously backlogged and needs more service capacity to process the backlogged data for better fairness.

The larger the service size, the higher the fairness that would be demanded to prevent the large-size service from never being finished. Therefore, we associate the service size with the fairness tuning parameter by establishing a constraint that $\frac{\omega^{\psi}}{\omega^{\varphi}} = \frac{\alpha^{\psi}}{\alpha^{\varphi}}$. Substituting it into (26), we have

$$\begin{aligned}(26) &\Longrightarrow S^{\psi}(\tau,t) \geq \frac{\omega^{\psi} S^{\varphi}(\tau,t)}{\omega^{\varphi}} \\ &\Longrightarrow S^{\psi}(\tau,t) \geq \frac{\alpha^{\psi} S^{\varphi}(\tau,t)}{\alpha^{\varphi}} \\ &\Longrightarrow \sum_{\varphi\in\mathbb{M}} S^{\psi}(\tau,t) \geq \sum_{\varphi\in\mathbb{M}} \frac{\alpha^{\psi} S^{\varphi}(\tau,t)}{\alpha^{\varphi}} \\ &\Longrightarrow S^{\psi}(\tau,t) \geq \alpha^{\psi} \sum_{\varphi\in\mathbb{M}} \frac{S^{\varphi}(\tau,t)}{\alpha^{\varphi}} \\ &\Longrightarrow S^{\psi}(\tau,t) \geq \frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{M}} \alpha^{\varphi}} \times \sum_{\varphi\in\mathbb{M}} S^{\varphi}(\tau,t), \end{aligned} \tag{27}$$

where $\mathbb{M} \subseteq \Psi$ is the set of services sharing v.

Now, substituting (25) into (27), it follows that

$$(27) \Longrightarrow S^{\psi}(\tau,t) \geq \frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{M}} \alpha^{\varphi}} \times r^{v}(t-\tau), \forall \psi, \varphi \in \mathbb{M}. \tag{28}$$

In fact, a better allocation of service capacity can improve the throughput efficiently. Hence, the VNF sharing process is carried out based on the above service capacity allocation principle and the original objective is updated as follows:

$$\begin{aligned} &Maximize: \quad \sum_{\psi\in\Psi} U^{\psi}\left(\frac{D^{\psi}(t)}{t}, \alpha^{\psi}\right) \\ &s.t.: \quad (22), (23), (24), (25), (28). \end{aligned} \tag{29}$$

4.2 Rate Coordination

Generally, we cannot offer 100% fairness during the service capacity allocation process, so that the processing rate mismatch among VNFs will be inevitable and this will generate one or more bottleneck VNFs for service chains. For example, given any ψ with two VNFs f_{i-1}^{ψ} and f_i^{ψ}, if $S_{i-1}^{\psi}(t) \gg S_i^{\psi}(t)$, then f_i^{ψ} becomes the bottleneck and most of the work done by f_{i-1}^{ψ} may be wasted, because f_{i-1}^{ψ} directly forwards traffic data to f_i^{ψ}.

Therefore, coordinating the processing rates among VNFs of the same service chain is vitally important for improving the resource utilization and the network throughput. Now, for any VNF of ψ, let us say $f_i^{\psi}, \forall i \in [1, |F|^{\psi}]$, its service capacity has already been allocated, which can be expanded as follows:

$$S_i^{\psi}(\tau,t) = r_i^{\psi}(t-\tau), \tag{30}$$

where r_i^{ψ} is the rate of f_i^{ψ}, and $r_i^{\psi} \leq r^{v}$ for any ψ using v.

In order to evaluate the stability of ψ, the backpressure [41] indicator is introduced for the service chain, and the corresponding definition is given below.

Definition 2: Given any two adjacent VNF instances f_i^{ψ} and f_{i-1}^{ψ} of ψ, f_i^{ψ} will cause backpressure for its upstream VNF (i.e., f_{i-1}^{ψ}) if f_i^{ψ} is continuously backlogged. The backpressure is defined as follows:

$$\begin{aligned}\varpi_i^{\psi} &= \frac{A_i^{\psi}(\tau,t) - D_i^{\psi}(\tau,t)}{t-\tau} \\ &= \frac{D_{i-1}^{\psi}(\tau,t) - D_i^{\psi}(\tau,t)}{t-\tau}. \end{aligned} \tag{31}$$

where the larger the value of ϖ_i^{ψ} is, the more serious the rate mismatching between f_i^{ψ} and f_{i-1}^{ψ} is.

In this regard, a threshold value $\theta \in [0,1]$ is defined to judge whether the processing rates should be coordinated or not. However, in order to make it comparable and express the overall condition of ψ, ϖ_i^{ψ} is normalized as follows:

$$\varpi_i^{\psi} = \frac{\varpi_i^{\psi}}{\sqrt{\sum_{i\in(1,|F^{\psi}|]} \varpi_i^{\psi^2}}}. \tag{32}$$

Then, if $\varpi_i^{\psi} \leq \theta$, no operations are executed. Otherwise, the rate coordination should be executed in case of resource waste. According to the Cannikin Law theory [40], the low-rate VNF determines the overall performance of a service chain. Therefore, to avoid the resource waste as much as possible and improve the throughput, we need to either improve the low-rate and downstream VNF's service capacity or reduce the high-rate and upstream VNF's service capacity.

Now, assuming f_i^{ψ} is the VNF of ψ and it is the bottleneck VNF that needs to be endowed with extra available resources for packet processing. All the service chains sharing v are stored in the set $\mathbb{M}_v$. Dividing $\mathbb{M}_v$ into two sets $\mathbb{L}_v$ and $\mathbb{N}_v$, where $\mathbb{L}_v \cup \mathbb{N}_v = \mathbb{M}_v$. The elements in $\mathbb{N}_v$ are work-conserving, while those in $\mathbb{L}_v$ cannot fully utilize the allocated service capacities. Given two service chains $\psi \in \mathbb{N}_v, \varphi \in \mathbb{L}_v$, they share the same VNF v by f_i^{ψ} and f_j^{φ} with the allocated service capacities of $S_i^{\psi}(t)$ and $S_j^{\varphi}(t)$. Then, for any time period (τ, t), (27) is transformed as

$$S_i^{\psi}(\tau,t) \geq \frac{\alpha^{\psi} S_j^{\varphi}(\tau,t)}{\alpha^{\varphi}}. \tag{33}$$

Since f_j^{φ} cannot make the full use of its allocated service capacity, we can assign the idle resource of f_j^{φ} to f_i^{ψ} for achieving high resource utilization and throughput. In this way, it follows that

$$\begin{aligned}(33) &\Longrightarrow \sum_{\varphi\in\mathbb{L}_v} S_i^{\psi}(\tau,t) \geq \sum_{\varphi\in\mathbb{L}_v} \frac{\alpha^{\psi} S_j^{\varphi}(\tau,t)}{\alpha^{\varphi}} \\ &\Longrightarrow S_i^{\psi}(\tau,t) \geq \frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{L}_v} \alpha^{\varphi}} \sum_{\varphi\in\mathbb{L}_v} S_j^{\varphi}(\tau,t) \\ &\Longrightarrow S_i^{\psi}(\tau,t) \geq \frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{L}_v} \alpha^{\varphi}} \left(\sum_{\varphi\in\mathbb{M}_v} S_j^{\varphi}(\tau,t) - \sum_{\varphi\in\mathbb{N}_v} S_j^{\varphi}(\tau,t)\right) \\ &\Longrightarrow S_i^{\psi}(\tau,t) \geq \frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{L}_v} \alpha^{\varphi}} \left(r^{v}(t-\tau) - \sum_{\varphi\in\mathbb{N}_v} S_j^{\varphi}(\tau,t)\right) \end{aligned} \tag{34}$$

Since the elements of $\mathbb{N}_v$ and $\mathbb{L}_v$ are generally unknown, a generalized form of (34) is given as follows:

$$S_i^{\psi}(\tau,t) \geq \max_{\mathbb{L}_v \cup \mathbb{N}_v = \mathbb{M}_v} \left\{ \frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{L}_v} \alpha^{\varphi}} \left(r^{v}(t-\tau) - \sum_{\varphi\in\mathbb{N}_v} S_j^{\varphi}(\tau,t)\right) \right\}. \tag{35}$$

Nevertheless, we should be aware that all the service chain sharing v may be busy working, that is, $\mathbb{L}_v = \emptyset$. In this condition, we cannot make any coordination between f_i^{ψ} and f_j^{φ} in case of causing side effects. Despite of this, as explained, f_i^{ψ} is the bottleneck VNF and causes great backpressure for its upstream VNF (i.e., f_{i-1}^{ψ}). Thus, the rate coordination can be carried out between f_i^{ψ} and f_{i-1}^{ψ}.

According to Definition 2, f_i^{ψ} causes backpressure for f_{i-1}^{ψ}, which means that f_{i-1}^{ψ} is the one with higher processing rate and it is the upstream VNF. Hence, assuming that f_{i-1}^{ψ} has been mapped on the VNF u with other service chains (denoted by $f_{j'}^{\varphi'}$), f_{i-1}^{ψ} should contribute part of its service capacity out for the purpose of achieving both higher resource utilization and lower resource waste. Now, denoting the same service set notations as $\mathbb{M}_u, \mathbb{N}_u$ and $\mathbb{L}_u$, the service capacity decreasing of f_{i-1}^{ψ} can be expressed as follows:

$$\begin{aligned} S_{i-1}^{\psi}(\tau,t) &\le \frac{\alpha^{\psi}}{\sum_{\varphi'\in\mathbb{L}_u}\alpha^{\varphi'}}\sum_{\varphi'\in\mathbb{L}_u} S_{j'}^{\varphi'}(\tau,t) \\ &\le \frac{\alpha^{\psi}}{\sum_{\varphi'\in\mathbb{L}_u}\alpha^{\varphi'}}\left(r^u(t-\tau) - \sum_{\varphi'\in\mathbb{N}_u} S_{j'}^{\varphi'}(\tau,t)\right). \end{aligned} \tag{36}$$

Likewise, the elements of $\mathbb{N}_u$ and $\mathbb{L}_u$ are generally unknown and a generalized form of (36) is given as follows:

$$\begin{aligned} &S_{i-1}^{\psi}(\tau,t) \\ &\le \min_{\mathbb{L}_u\cup\mathbb{N}_u=\mathbb{M}_u}\left\{\frac{\alpha^{\psi}}{\sum_{\varphi'\in\mathbb{L}_u}\alpha^{\varphi'}}\left(r^u(t-\tau) - \sum_{\varphi'\in\mathbb{N}_u} S_{j'}^{\varphi'}(\tau,t)\right)\right\}. \end{aligned} \tag{37}$$

Since the downstream VNF of f_{i-1}^{ψ} is f_i^{ψ}, we cannot affect the original performance of ψ when carrying out the resource redistribution process. In this way, the service capacity of f_{i-1}^{ψ} should not be smaller than that of f_i^{ψ} after the resource redistribution, that is,

$$S_{i-1}^{\psi}(\tau,t) \ge S_i^{\psi}(\tau,t). \tag{38}$$

Substituting (35) into (38), the constraint is expanded as follows:

$$\begin{aligned} &S_{i-1}^{\psi}(\tau,t) \\ &\ge \max_{\mathbb{L}_v\cup\mathbb{N}_v=\mathbb{M}_v}\left\{\frac{\alpha^{\psi}}{\sum_{\varphi\in\mathbb{L}_v}\alpha^{\varphi}}\left(r^v(t-\tau) - \sum_{\varphi\in\mathbb{N}_v} S_j^{\varphi}(\tau,t)\right)\right\}. \end{aligned} \tag{39}$$

Based on the above rate coordination principle, the objective becomes

$$\begin{aligned} &Maximize: \quad \sum_{\psi\in\Psi} U^{\psi}\left(\frac{D^{\psi}(t)}{t}, \alpha^{\psi}\right) \\ &s.t.: \quad (22), (23), (24), (25), (28), (\mathbf{35}), (\mathbf{37}), (\mathbf{39}). \end{aligned} \tag{40}$$

Following the VNF sharing scheduling and rate coordination principle, the objective can be achieved. The overall corresponding pseudo-code is shown in Algorithm 1, where the time complexity of 1) lines 2-5, is $O(|\mathbb{M}_v| \times |F^{\psi}|)$; 2) lines 7-13, is $O(|\Psi|)$; 3) lines 15-19, is $O(|\mathbb{M}_v| \times |F^{\psi}|)$; 4) lines 20-32, is $O(|\Psi| + |\mathbb{M}_u|)$. The above four parts are executed serially in this pseudo-code, so that the overall time complexity is expressed as $O(|\mathbb{M}_v| \times |F^{\psi}|) + O(|\Psi|) + O(|\mathbb{M}_v| \times |F^{\psi}|) + O(|\Psi| + |\mathbb{M}_u|) \approx O(2 \times |\mathbb{M}_v| \times |F^{\psi}| + 2 \times |\Psi| + |\mathbb{M}_u|)$. Since $\mathbb{M}_v, \mathbb{M}_u \subseteq \Psi$, the worst case time complexity can be expressed as $O(2 \times |\Psi| \times |F^{\psi}| + 2 \times |\Psi| + |\Psi|) \approx O(|\Psi| \times (2 \times |F^{\psi}| + 3))$.

Algorithm 1. Fairness-Aware VNF Sharing and Rate Coordination Algorithm

```
Input: The VNF instances $v, u$, the service chain set $\Psi$, the
       fairness parameter $\alpha$, the backpressure threshold $\theta$
Output: Service capacity allocation solution
$S \leftarrow \emptyset$;
for $\psi \in \mathbb{M}_v$ do
  for $f_i^{\psi}$ mapped on $v$ do
    $S_i^{\psi} \leftarrow$ Initializing the service capacity for $f_i^{\psi}$ according to
    definition 1 and (28);
    $S \leftarrow S_i^{\psi}$;
$\mathbb{M}_v, \mathbb{L}_v, \mathbb{N}_v \leftarrow \emptyset$;
for $\psi \in \Psi$ do
  if $\psi$ uses the VNF instance $v$ then
    $\mathbb{M}_v \leftarrow \mathbb{M}_v \cup \psi$;
    if $\psi$ is work-conserving then
      $\mathbb{N}_v \leftarrow \mathbb{N}_v \cup \psi$;
    else
      $\mathbb{L}_v \leftarrow \mathbb{L}_v \cup \psi$;
if $\mathbb{L}_v \neq \emptyset$ then
  for $\psi$ sharing $v$ do
    $f_i^{\psi} \leftarrow$ Searching the VNF of $\psi$ mapped on $v$;
    Calculating the backpressure indicator $\varpi_i^{\psi}$ between $f_i^{\psi}$
    and $f_{i-1}^{\psi}$;
    if $\varpi_i^{\psi} > \theta$ and $f_i^{\psi} \in \mathbb{N}_v$ then
      $S_i^{\psi} \leftarrow$ Executing rate coordination by increasing the
      service capacities of $f_i^{\psi}$ from the elements in $\mathbb{L}_v$ accord-
      ing to (35);
      Updating $S \leftarrow S_i^{\psi}$;
else if $\mathbb{L}_v = \emptyset$ then
  $f_{i-1}^{\psi} \leftarrow$ Locating the upstream VNF of $f_i^{\psi}$;
  $u \leftarrow$ Finding the VNF that $f_{i-1}^{\psi}$ is mapped;
  $\mathbb{M}_u, \mathbb{L}_u, \mathbb{N}_u \leftarrow \emptyset$;
  for $\psi \in \Psi$ do
    if $\psi$ uses the VNF instance $u$ then
      $\mathbb{M}_u \leftarrow \mathbb{M}_u \cup \psi$;
      if $\psi$ is work-conserving then
        $\mathbb{N}_u \leftarrow \mathbb{N}_u \cup \psi$;
      else
        $\mathbb{L}_u \leftarrow \mathbb{L}_u \cup \psi$;
  if $\mathbb{N}_u \neq \emptyset$ && $f_{i-1}^{\psi} \in \mathbb{L}_u$ then
    $S_{i-1}^{\psi} \leftarrow$ Executing rate coordination by decreasing the
    service capacity of $f_{i-1}^{\psi}$ and distributing it to the elements
    in $\mathbb{N}_u$ according to (37) and (39);
    Updating $S \leftarrow S_{i-1}^{\psi}$;
return $S$;
```

4.3 Performance Bound Analysis

Given any service chain $\psi \in \Psi$, its cumulative amount of traffic during the time interval $[\tau, t]$ has been formulated as $A^{\psi}(\tau,t)$ which is limited by a widely affine envelope function [42] as follows:

$$A^{\psi}(\tau,t) \le \rho(t-\tau) + \sigma, \tag{41}$$

where $t > \tau \geq 0$. $\rho > 0$ is the traffic arriving rate and $\sigma \geq 0$ is the burst parameter for ψ.

Based on the above setting, the performance bounds of the throughput, the backlog and the delay are given in the following three theorems.

Theorem 2. *Given the service chain ψ with the throughput denoted by $\mathfrak{T}^{\psi}$, its performance bound can be expressed as follows:*

$$\mathfrak{T}^{\psi}(t) \leq \min\Big\{\rho + \frac{\sigma}{t}\Big\} \tag{42}$$

Proof. The cumulative amount of traffic departing ψ is denoted by $D^{\psi}(t)$, where $\frac{D^{\psi}(t)}{t}$ actually evaluates the average throughput during the time interval $[0, t]$. Then, it follows that

$$\mathfrak{T}^{\psi}(t) = \frac{D^{\psi}(t)}{t}. \tag{43}$$

Substituting (24) and (41) into (43), a generalized form for the throughput bound is presented as follows:

$$\begin{aligned}\Longrightarrow \mathfrak{T}^{\psi}(t) &\leq \frac{A^{\psi}(t)}{t}\\ &= \frac{\rho \times t + \sigma}{t}\\ &= \rho + \frac{\sigma}{t}\\ \Longrightarrow \mathfrak{T}^{\psi}(t) &\leq \min\Big\{\rho + \frac{\sigma}{t}\Big\}.\end{aligned} \tag{44}$$

Then, theorem 2 is proved. □

Theorem 3. *Given the service chain ψ with the backlog denoted by $\mathfrak{B}^{\psi}$, its performance bound can be expressed as follows:*

$$\mathfrak{B}^{\psi} \leq \max_{\tau\in[0,t]}\big\{(\rho - r^{\psi})(t-\tau) + \sigma\big\}. \tag{45}$$

Proof. The backlog actually means the total amount of traffic blocking in the queue and the traffic in processing, so that we have

$$\begin{aligned}\Longrightarrow \mathfrak{B}^{\psi}(t) &= A^{\psi}(t) - D^{\psi}(t)\\ &= A^{\psi}(t) - \min_{\tau\in[0,t]}\big\{A^{\psi}(\tau) + S^{\psi}(\tau, t)\big\}\\ \Longrightarrow \mathfrak{B}^{\psi}(t) &\leq \max_{\tau\in[0,t]}\big\{A^{\psi}(\tau, t) - S^{\psi}(\tau, t)\big\}\\ &= \max_{\tau\in[0,t]}\big\{\rho(t-\tau) + \sigma - r^{\psi}(t-\tau)\big\}\\ &= \max_{\tau\in[0,t]}\big\{(\rho - r^{\psi})(t-\tau) + \sigma\big\}\end{aligned} \tag{46}$$

Then, theorem 3 is proved. □

Theorem 4. *Given the service chain ψ with the delay denoted by $\mathfrak{D}^{\psi}$, its performance bound can be expressed as follows:*

$$\begin{aligned}&\mathfrak{D}^{\psi}(t)\\ &= \min\Big\{\varepsilon \geq 0 : \max_{\tau\in[0,t]}\big\{(\rho - r^{\psi})(t-\tau) + \sigma - r^{\psi}\varepsilon\big\} \leq 0\Big\}.\end{aligned} \tag{47}$$

Proof. Given a certain amount of traffic belonging to ψ, denoting the offset time between the arriving time of the first packet and the leaving time of the last packet by ε, the delay bound of ψ is formulated as follows:

$$\mathfrak{D}^{\psi}(t) = \min\big\{\varepsilon \geq 0 : A^{\psi}(t) - D^{\psi}(t+\varepsilon) \leq 0\big\}. \tag{48}$$

In particular, $A^{\psi}(t) - D^{\psi}(t+\varepsilon)$ is extended as follows:

$$\begin{aligned}&A^{\psi}(t) - D^{\psi}(t+\varepsilon)\\ &= A^{\psi}(t) - \min_{\tau\in[0,t]}\big\{A^{\psi}(\tau) + S^{\psi}(\tau, t+\varepsilon)\big\}\\ &\leq \max_{\tau\in[0,t]}\big\{A^{\psi}(\tau, t) - S^{\psi}(\tau, t+\varepsilon)\big\}\\ &= \max_{\tau\in[0,t]}\big\{\rho(t-\tau) + \sigma - r^{\psi}(t+\varepsilon-\tau)\big\}\\ &= \max_{\tau\in[0,t]}\big\{(\rho - r^{\psi})(t-\tau) + \sigma - r^{\psi}\varepsilon\big\}\end{aligned} \tag{49}$$

Substituting (49) into (48), theorem 4 is proved. □

5 Performance Evaluation

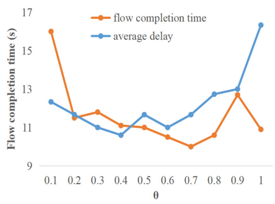
5.1 Setup

The proposed method is implemented using JAVA with a customized discrete event simulator and evaluated over the real-world topology of VltWavennet which has 83 nodes and 84 links. In particular, each node can be both server and switcher, that is, each node is able to host VNFs to provide services and forward packets to destinations. First, the number of VNFs in each SFC is determined randomly between [2,5] following a normalization distribution, because 2-5 common VNFs can usually satisfy the basic requirements of most services in practice. Second, the number of VNF types is set to 10, because 10 kinds of VNFs can cover the maximum number of the commonly used VNFs in practice. Third, the number of flows in the network ranges from 50 to 100 and each physical server is allowed to host [3,5] kinds of VNFs randomly. Therefore, the maximum number of VNF instances required is between [100, 500], where 100 = 50*2 and 500 = 100*5. Meanwhile, the number of VNFs offered is between [249, 415], where 249 = 83*3 and 415 = 83*5. Since $[249, 415] \subseteq [100, 500]$, this setting can well evaluate the performance of algorithms by simulating the environment with rich and poor resource. Finally, the bandwidth of each virtual link is set to 20 Mbps and the processing time of each VNF is randomly determined between [1,3]ms. The simulation time model follows a discrete distribution, since it is always possible to map a continuous time model (e.g., $CT(t1)$) to a discrete time model (e.g., $DT(t2)$) by using a time slot ζ, e.g., $DT(t2) = CT(\zeta \times t1)$.

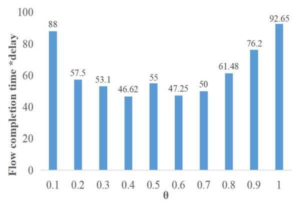
Besides, the simulation is experimented for 100 times and the average results are presented with the hardware of Intel (R) Core(TM) i7-6700 CPU 3.4 GHz and 8 G memory.

5.2 Benchmarks and Metrics

In order to present a comprehensive comparison, three state-of-the-art methods are used for comparison. The first one tries to study the VNF scheduling problem by using the priorities of different flows [33]. Briefly speaking, for this **P**riority based **Alg**orithm (P-Alg), it allocates more service capacities to the services with higher priorities. The second one tries to coordinate the VNF processing rates by jointly taking the VNF load and backpressure into consideration [7],



(a) flow completion time/average delay vs. θ



(b) flow completion time*average delay vs. θ

Fig. 4. Flow completion time*delay against different value of θ.

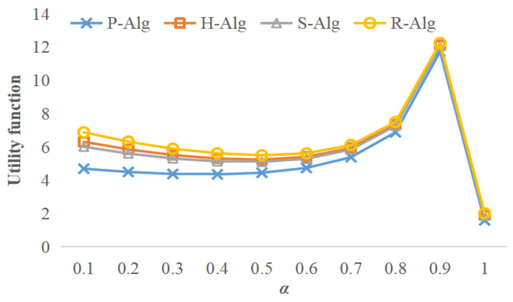


Fig. 5. Utility function.

and it is a **H**euristic based **Alg**orithm (H-Alg). The third one tries to coordinate the service rates by splitting the big traffic into different small ones [39] and thus it is referred to as the **S**plitting based **Alg**orithm (S-Alg). As for the proposed algorithm, it is referred to as the **R**ate-coordination **Alg**orithm (R-Alg). The four algorithms are tested and evaluated against multiple metrics including the utility, the average delay, the flow completion time, the packet loss rate, the throughput and the backlog.

5.3 Experiment Results

Before evaluating the performance, the threshold variable θ is determined by using the flow completion time and the average delay, since the two indicators are strongly affected by the value of θ in practice. The corresponding results are shown in Fig. 4, where Fig. 4a shows the results of the flow completion time and delay against θ respectively, while Fig. 4b shows the product of the flow completion time and delay against θ.

In Fig. 4a, it is easy to notice that with the increase of the value of θ, the flow completion time shows a decreasing trend while the average delay shows an increasing trend. This is reasonable because the bigger the value of θ is, the higher the frequency of executing the rate coordination is, which means: 1) less time would be required to finish the flow completion process; 2) more time would be required for the end-to-end delay. Due to such a conflict, we then determine the value of θ by calculating $\min\{flow\_completion\_time * average\_delay\}$, since we need to minimize both the flow completion time and the average delay as much as possible. The corresponding results are shown in Fig. 4b, where the minimum value of $flow\_completion\_time * average\_delay$ is 46.62 when θ is set to 0.4. Hence, we can conclude that setting θ to 0.4 could make a better balance between the flow completion time and the average delay, which are both important metrics. In this consideration, the value of θ is set to 0.4 for the following results.

Besides, the experimental results include the utility, the average delay, the flow completion time, the throughput and the backlog. Then, we present and analyze them as follows:

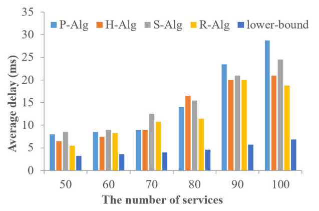
5.3.1 *Utility*

As explained, the objective is to maximize the utility function shown in (21). Then, we first calculate the utility achieved by each algorithm against different values of the fairness tuning parameter α. The corresponding results are shown in Fig. 5 and several phenomena can be observed. First of all, the proposed method (i.e., R-Alg) has the maximum utility, which means that it can achieve the best performance according to the definition. Second, the gap between the utilities achieved by different methods is becoming smaller and smaller with the increase of α. Third, the value of utility decreases slowly at the beginning (e.g., α is less than or equal to 0.4) and then increases quickly (i.e., α is larger than 0.5). Fourth, the utilities achieved by four methods all reduce to the lowest value when α becomes 1. For the reasons behind these phenomena, on one hand, the proposed method R-Alg introduces a rate coordination process which avoids the network congestion greatly and is not considered in the other three benchmarks. According to (20), the value of utility is proportional to the throughput, so that R-Alg achieves higher utility by improving the throughput. On the other hand, the utility function is calculated in two different ways on the condition that α equals to 1 or not, which is the main reason leading to the fourth phenomenon.

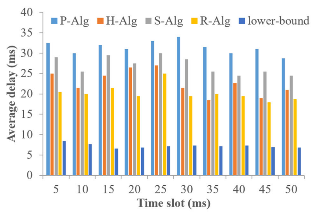
5.3.2 *Average Delay*

The results of the average delay are calculated and shown in Fig. 6, where Fig. 6a presents the average delay against the number of services and Fig. 6b presents the average delay against the time slot (i.e., the time slot of sending services). First of all, by observing Fig. 6a, we can discover that the average delay increases with the increase of the number of SFCs. It is commonly known that the number of SFC is proportional to the network load, so that the more SFCs are, the higher the network load is, which results in high delay accordingly. In addition, the average delay increases slowly when the number of SFC does not exceed about 60 and then increases quickly after 60. That is because the following arriving SFCs will demand far beyond the processing capacity of the network, so that congestion occurs frequently. Specifically, the increase in the average delay of P-Alg is the biggest, since it may cause a long waiting time for most flows after adjusting their priorities. On the contrary, the delay increasing of H-Alg, S-Alg and R-Alg is relatively slower, because they all take the network congestion into consideration.

Despite of this, the proposed R-Alg method still achieves a smaller average delay than P-Alg, H-Alg and S-Alg. This

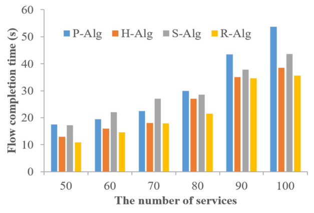


(a) Average delay vs. number of services

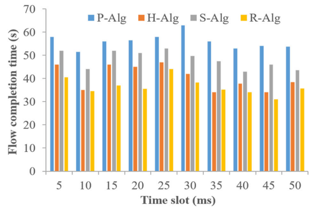


(b) Average delay vs. time slot

Fig. 6. Average delay.

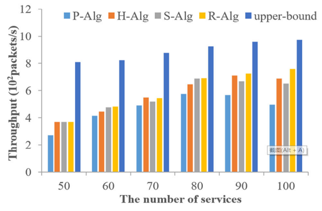


(a) Flow completion time vs. number of services

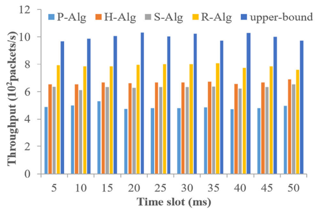


(b) Flow completion time vs. time slot

Fig. 7. Flow completion time.



(a) Average throughput vs. number of services



(b) Average throughput vs. time slot

Fig. 8. Average throughput.

can be reflected by the data that: 1) the average delay of P-Alg is about 1.49 times, 2) the average delay of H-Alg is about 1.15 times, 3) the average delay of S-Alg is about 1.42 times that of R-Alg. Via these data, it is easy to note that the delay of H-Alg is close to that of R-Alg, because the scheduling strategy of H-Alg has similar effect to that of the rate coordination in R-Alg. As for S-Alg, it suffers from a drawback that this method may fail to split flows, which results in high delay directly. As for the lower-bound delay, it is calculated according to the performance bound equation in Section 4.3. Apparently, it achieves the smallest and the most stable delay than the other methods.

Second, fixing the number of services to 100 and changing the time slots from 5 ms to 50 ms, the corresponding average results are shown in Fig. 6b and a similar relation among the delays achieved by different methods can be discovered, that is, lower-bound $<$ R-Alg $<$ H-Alg $<$ S-Alg $<$ P-Alg.

5.3.3 Flow Completion Time

The flow completion time is calculated by the gap between the flow starting time and the flow ending time, and it is used to evaluate the efficiency of methods, because the smaller the completion time is, the higher the efficiency of methods is. It is aware that the flow completion time has a close relationship with the average delay, that is, long delay generally has a positive probability to result in a long flow completion time when the network environment is fixed. Hence, we can observe similar trends between Figs. 7 and 6. For example, in Fig. 7a, the flow completion time of 1) R-Alg is the smallest; 2) H-Alg is the second smallest; 3) S-Alg is the third smallest; 4)P-Alg is the highest. This is the same conclusion shown in Fig. 6. The corresponding reasons have already been explained in the above section. Specifically, let us focus on the performance of R-Alg and H-Alg. When the number of SFC is between 60 and 70, the average delay of R-Alg is slightly higher than that of H-Alg in Fig. 6, while the flow completion time of R-Alg is slightly lower than that of H-Alg in Fig. 7. Such a contradiction is reasonable. Although the average delay can be regarded as a positive factor of the flow completion time, there are also many other factors that can affect the overall flow completion time, such as the time spent on the flow monitoring, service provisioning and even the rate coordination. Different flows demand uncertain time to handle these factors, so that the trend of flow completion time may differ slightly from the result of average delay.

As for the results in Fig. 7b, they are achieved under the same setting, that is, the number of services is fixed to 100. The flow completion time varies against the time slots for all methods. Despite of this, a similar phenomenon can be found that R-Alg has the smallest flow completion time, because the flow completion time in this setting is proportional to the average delay.

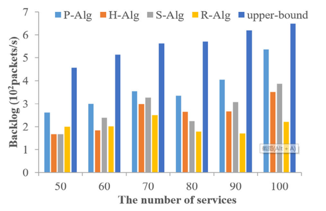
5.3.4 Average Throughput

The results of average throughput are presented in Figs. 8a and 8b against the number of services and the time slot respectively. Specifically, let us look at the results in Fig. 8a from an overall perspective, that is, the throughput increases until the stable condition is reached. With the increase of the service number, the network load also increases. Then, before reaching the upper limit of network processing capacity, the throughput will increase continuously. However, once this upper limit is reached, it becomes very hard to further improve the throughput. In this condition, the performance degradation may even happen. For example, the throughput of P-Alg decreases when the number of SFC exceeds 80, while that of H-Alg decreases when the number of SFC exceeds 90. This is the side effect caused by the priority adjustment, because improper priority adjustment may not only achieve unexpected performance, but also deteriorate the network congestion. On the contrary, R-Alg does not show such phenomenon, which indicates that R-Alg has good scalability and can better adapt to different workloads compared to the other three methods.

Besides, it is observed that the proposed R-Alg achieves the highest throughput due to the reason that it dynamically coordinates the packet processing rates of a flow along its routing path. Then, the congestion situation can be avoided greatly after the rate coordination process. One extreme situation can be observed is that the throughput of H-Alg becomes the largest when the number of service is 70. In this case, the throughput of H-Alg is about 0.8% higher than that of R-Alg. Nevertheless, the throughput of R-Alg is higher than that of H-Alg at the other cases, i.e., the number of SFC is {50, 60, 80, 90, 100}. However, as for the results in Fig. 8b, we can see that the throughput achieved by R-Alg is higher than that of P-Alg, H-Alg and S-Alg in terms of all the time slots, which means that R-Alg can also adapt to different service frequencies. Meanwhile, the throughput of R-Alg is the closest to the upper-bound throughput, which also reflects the advantages of R-Alg to a certain extent.

5.3.5 Average Backlog

In fact, the backlog and throughput are closely related to each other. The backlog indicates the number of packets accumulated in nodes and the corresponding results are


(a) Average backlog vs. number of services

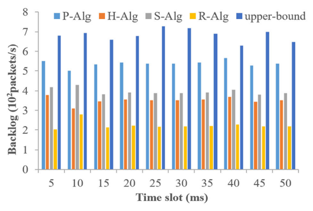

(b) Average backlog vs. time slot

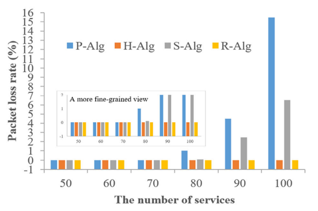
Fig. 9. Average backlog.

shown in Fig. 9, where Fig. 9a is the backlog against the number of services and Fig. 9b is the backlog against the time slots. In particular, the larger the backlog is, the more packets that are waiting to be processed are, which then results in long delay and low throughput. For example, taking the 100 services as an example, as shown in Fig. 8a, the throughput of 1) R-Alg is the highest; 2) H-Alg is the second highest; 3) S-Alg is the third highest; 4) P-Alg is the smallest. However, the backlog results of the four methods are just on the opposite as shown in Fig. 9a at the case of 100 services. Besides, we can also see that the backlog does not grow exactly with the increase of the number of services. This is reasonable, since the backlog is determined by not only the number of services, but also the factors of packet priority, available resource, etc.

Moreover, as explained, the decrease of backlog would lead to the increase of throughput. This can also be discovered by comparing the results in Figs. 8a and 9a. For example, when the number of services is less or equal to 80, the throughput increases quickly, while the backlog increases slowly. However, when the number of services exceeds 80, the throughput increases slowly (or even decreases), while the backlog increases quickly. For the results in Fig. 9b, we can see that the backlog of P-Alg is the closest to the upper-bound, which means P-Alg is more easier to generate the congestion point. On the contrary, the backlog of R-Alg is the farthest from the upper-bound, so that it can maximize the resource utilization.

5.3.6 Packet Loss Rate

The results of packet loss rate are presented in Fig. 10. Apparently, in Fig. 10a, the packet loss rates of H-Alg and R-Alg are both equal to 0 for the number of services ranging from 50 to 100. Namely, H-Alg and R-Alg achieve similar packet loss rate, which can be reflected by the results shown in Figs. 6, 7, 8, and 9. However, for P-Alg and S-Alg, their packet loss rates approach to 0 at the beginning (e.g., the number of services does not exceed 70) and then grow high when the number of services exceeds 70. In particular, the packet loss rate of P-Alg is the highest and it grows rapidly


(a) Packet loss rate vs. number of services

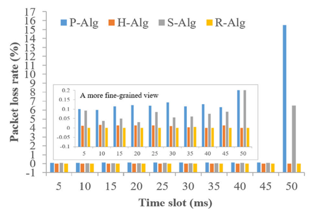

(b) Packet loss rate vs. time slot

Fig. 10. Packet loss rate.

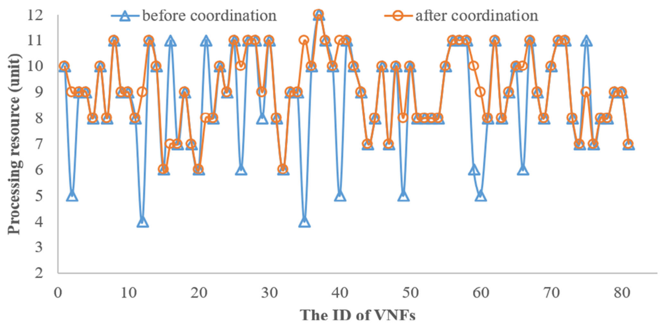


Fig. 11. Processing resource before and after coordination.

compared to the other methods. Hence, by observing the results in Fig. 10a, we can conclude that the congestion thresholds for P-Alg and S-Alg are 70 and 80 respectively. For P-Alg, the priority adjustment does not essentially address the network congestion issue, and it achieves the highest packet loss rate. For S-Alg, it is hard to find suitable VNFs for flow splitting especially when the number of services is large, and this condition also increases the packet loss rate.

In addition, similar phenomena can be found in Fig. 10b. From the fine-grained view, we can clearly see that R-Alg has the smallest packet loss rate. In this condition, R-Alg performs better than H-Alg, because R-Alg can adapt to both different service frequencies and network loads.

5.3.7 Rate Coordination

The VNF processing rate coordination is the core part of the proposed R-Alg. As explained, the VNFs are used to process the traffic of different service chains. However, for all VNFs deployed in the network, they have different processing rates due to the different amount of resources allocated to them. Such an unbalanced rate (or resource) distribution has a great influence on the network performance. Hence, we try to show the VNF rate coordination process by evaluating the processing resource allocated to VNFs. The corresponding results are shown in Fig. 11. In particular, two cases are evaluated, that is, the resource distribution of VNFs before executing the rate coordination process and the resource distribution of VNFs after executing the rate coordination process (i.e., R-Alg). Specifically, the blue line indicates the VNF resource distribution before executing R-Alg and the red line indicates the VNF resource distribution after executing R-Alg. Obviously, we can see that the processing resources allocated to VNFs before using R-Alg are distributed in a large range, that is, the gap between the maximum and minimum processing resource before using R-Alg is 8, while that after using R-Alg is 6. This phenomenon means that the rates of all VNFs become relatively close after using R-Algorithm In this way, a more balanced network environment can be reached, so that more packets can be scheduled without waiting too long.

5.3.8 Performance Stability Analysis

We next evaluate the performance stability of all algorithms from the perspectives of the delay, throughput, backlog and rate coordination. The corresponding statistical results are summarized in Table 2, including the minimum, maximum,

TABLE 2
Performance Stability Results

	Min value	Max value	Mean value	Variance
Delay				
P-Alg	8	28.75	15.29	77.81
H-Alg	6.5	21	13.42	42.54
S-Alg	8.5	24.5	15.17	42.16
R-Alg	**5.5**	**18.75**	**12.45**	**33.18**
Throughput				
P-Alg	2.72	5.75	4.69	1.28
H-Alg	3.69	7.11	5.68	1.93
S-Alg	3.69	6.69	5.63	1.64
R-Alg	**3.71**	**7.59**	**5.95**	**2.35**
Backlog				
P-Alg	2.61	5.37	3.65	0.94
H-Alg	1.67	3.51	2.55	0.48
S-Alg	1.67	3.86	2.75	0.63
R-Alg	**1.7**	**2.5**	**2.03**	**0.083**
Processing resource of VNFs of R-Alg				
before coordination	4	12	8.71	3.75
after coordination	6	12	9.12	2.1

mean and variance value of different indicators. The smaller the value of the variance is, the better the stability is.

First, for the statistical results of the delay, they are shown in rows 3-6 of Table 2. The min, max and mean delay of R-Alg are smaller than that of the other methods, that is, R-Alg (5.5, 18.75, 12.45) $<$ H-Alg(6.5, 21, 13.42) $<$ S-Alg(8.5, 24.5, 15.17) $<$ P-Alg(8, 28.75, 15.29). Note that the min delay of S-Alg is higher than that of P-Alg. Nevertheless, the delay variances of different algorithms are sorted in another order, that is, R-Alg(33.18) $<$ S-Alg(42.16) $<$ H-Alg(42.54) $<$ P-Alg(77.81). Considering the case that for the variance, the smaller the better, we can summarize that R-Alg can adapt to different workloads more easily than the other methods.

Second, for the statistical results about the throughput, they are shown in rows 8-11 of Table 2. As explained, R-Alg achieves the highest average throughput, which can be proved by the results in Table 2. Apart from this, the minimum, maximum and variance value of throughput achieved by R-Alg are also the largest. For the values of the minimum and maximum throughput, the larger the better. However, that is on the opposite of variance. Nevertheless, the large value of variance is caused by the large gap between the minimum and maximum throughput (i.e., 3.88), while the gap of the other three methods are 3.03, 3.42 and 3 respectively.

Third, for the statistical results about the backlog, they are shown in rows 13-16 of Table 2. We can see that P-Alg has the largest value of minimum, maximum, mean and variance value, which means that the throughput of P-Alg is the smallest. This phenomenon can be observed in Fig. 9, where the correctness of the proposed method can be guaranteed to a certain extent. In addition, the backlog performance of H-Alg and S-Alg are similar, that is, 1) the gap between their minimum backlog values is 0; 2) the gap between their maximum backlog values is 0.35; 3) the gap between their mean backlog values is 0.2; 4) the gap between their backlog variances is 0.25. Hence, we can regard that H-Alg and S-Alg have similar performance in terms of backlog. Lastly, the backlog achieved by R-Alg is between [1.7, 2.5]. Via comparison, the mean backlog of R-Alg is also smaller than that of the other three methods. The smaller the backlog is, the more packets are forwarded, and then the throughput is naturally higher. From this conclusion, the throughput achieved by R-Alg should be the highest, which is exactly consistent with the case shown in Fig. 9. One more condition that should be aware is that the backlog variance of R-Alg is far smaller than that of the other methods, which means that R-Alg can easily adapt to different workloads based on the rate coordination.

Fourth, the statistical results of the rate coordination are shown in rows 18-19 of Table 2, from which we can discover 1) the VNF processing resource distribution range becomes smaller after executing R-Alg, that is, changing from [4, 12] to [6, 12]; 2) the mean value of VNF processing resource becomes larger; 3) the variance of VNF processing resource becomes smaller. From these phenomena, we can conclude that the VNF processing rate coordination process of R-Alg can help improve the resource utilization (i.e., make full use of the idle resource) and achieve a more balanced network environment.

6 Conclusion

In this article, we address the VNF sharing scheduling and the processing rate mismatch problem among VNFs for the purpose of avoiding waste work and improving resource utilization. In particular, the VNF constituted services are formulated as a tandem system, for which different performance bounds are constructed. In order to address the VNF sharing scheduling and the rate mismatch problem, we intend to maximize the idle resource utilization via the proper rate coordination. Namely, the previously allocated resource that has not been fully utilized, will be redistributed to handle the busy traffic of other services, thus to balance the rates of different VNFs. Via experiments, we present that the proposed method can achieve better performance than the benchmarks. Future work includes exploring the rate mismatch problem in specific network interface card and large scale networks.

References

[1] L. Linguaglossa et al., "Survey of performance acceleration techniques for network function virtualization," *Proc. IEEE*, vol. 107, no. 4, pp. 746–764, Apr. 2019.

[2] B. Yi, X. Wang, K. Li, S. K. Das, and M. Huang, "A comprehensive survey of network function virtualization," *Comput. Netw.*, vol. 133, pp. 212–262, 2018.

[3] H. Hawilo, M. Jammal, and A. Shami, "Network function virtualization-aware orchestrator for service function chaining placement in the cloud," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 3, pp. 643–655, Mar. 2019.

[4] R. Riggio, A. Bradai, D. Harutyunyan, T. Rasheed, and T. Ahmed, "Scheduling wireless virtual networks functions," *IEEE Trans. Netw. Service Manag.*, vol. 13, no. 2, pp. 240–252, Jun. 2016.

[5] B. Yi, X. Wang, and M. Huang, "A generalized VNF sharing approach for service scheduling," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 73–76, Jan. 2018.

[6] D. Li, P. Hong, J. Pei, and W. Wang, "Availability aware VNF deployment in datacenter through shared redundancy," in *Proc. IEEE Conf. Standards Commun. Netw.*, 2018, pp. 1–6.

[7] S. G. Kulkarni et al., "NFVnice: Dynamic backpressure and scheduling for NFV service chains," in *Proc. IEEE Conf. ACM Special Int. Group Data Commun.*, 2017, pp. 71–84.

[8] X. Zhang, Y. Wang, G. Geng, and J. Yu, "Delay-optimized multicast tree packing in software-defined networks," *IEEE Trans. Services Comput.*, to be published, doi: 10.1109/TSC.2021.3106264.

[9] P. Naik, D. K. Shaw, and M. Vutukuru, "NFVPerf: Online performance monitoring and bottleneck detection for NFV," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw.*, 2016, pp. 154–160.

[10] B. Yi et al., "A comprehensive survey of network function virtualization," *Comput. Netw.*, vol. 133, pp. 212–262, 2018.

[11] L. Qu, C. Assi, and K. Shaban, "Delay-aware scheduling and resource optimization with network function virtualization," *IEEE Trans. Commun.*, vol. 64, no. 9, pp. 3746–3758, Sep. 2016.

[12] F. Malandrino et al., "Reducing service deployment cost through VNF sharing," *IEEE/ACM Trans. Netw.*, vol. 27, no. 6, pp. 2363–2376, Dec. 2019.

[13] H. Xuan, Y. Wang, S. Guan, and Z. Xu, "A New optimization model and algorithm for a network scheduling problem in inter-datacenters elastic optical networks," in *Proc. IEEE 13th Int. Conf. Comput. Intell. Secur.*, 2017, pp. 30–34.

[14] D. Bringhenti and F. Valenza, "A twofold model for VNF embedding and time-sensitive network flow scheduling," *IEEE Access*, vol. 10, pp. 44 384–44 399, 2022.

[15] R. Mijumbi, J. Serrat, J. -L. Gorricho, N. Bouten, F. De Turck and, and S. Davy, "Design and evaluation of algorithms for mapping and scheduling of virtual network functions," in *Proc. IEEE 1st Conf. Netw. Softwarization*, 2015, pp. 1–9.

[16] H. A. Alameddine, S. Sebbah, and C. Assi, "On the interplay between network function mapping and scheduling in VNF-Based networks: A. column generation approach," *IEEE Trans. Netw. Service Manag.*, vol. 14, no. 4, pp. 860–874, Dec. 2017.

[17] Y. Liu, H. Zhang, H. Guan, and Y. Wang, "A Method for adaptive resource adjustment of dynamic service function chain," *IEEE Access*, vol. 6, pp. 69 988–70 004, 2018.

[18] T. Gao et al., "Cost-efficient VNF placement and scheduling in public cloud networks," *IEEE Trans. Commun.*, vol. 68, no. 8, pp. 4946–4959, Aug. 2020.

[19] M. Masoud, S. Lee, and S. Belkasim, "Dynamic allocation of service function chains under priority dependency constraint, in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell.*, 2016, pp. 684–688.

[20] L. Qu, C. Assi, and K. Shaban, "Network function virtualization scheduling with transmission delay optimization," in *Proc. IEEE/IFIP Netw. Operations Manage. Symp.*, 2016, pp. 638–644.

[21] M. Wang, J. Liu, W. Chen, and A. Ephremides, "Joint queue-aware and channel-aware delay optimal scheduling of arbitrarily bursty traffic over multi-state time-varying channels," *IEEE Trans. Commun.*, vol. 67, no. 1, pp. 503–517, Jan. 2019.

[22] L. Gu et al., "Fairness-aware dynamic rate control and flow scheduling for network utility maximization in network service chain," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1059–1071, May 2019.

[23] C. Pham, N. H. Tran, and C. S. Hong, "Virtual network function scheduling: A matching game approach," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 69–72, Jan. 2018.

[24] J. Li, W. Shi, N. Zhang, and X. Shen, "Delay-aware VNF scheduling: A reinforcement learning approach with variable action set," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 1, pp. 304–318, Mar. 2021.

[25] L. Gu, D. Zeng, W. Li, S. Guo, A. Y. Zomaya, and H. Jin, "Intelligent VNF orchestration and flow scheduling via model-assisted deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, 2020, vol. 38, no. 2, pp. 279–291, Feb. 2020.

[26] Y. Zhang, F. He, T. Sato, and E. Oki, "Network service scheduling with resource sharing and preemption," *IEEE Trans. Netw. Service Manag.*, vol. 17, no. 2, pp. 764–778, Jun. 2020.

[27] X. Huang et al., "Online VNF chaining and predictive scheduling: Optimality and trade-offs," *IEEE/ACM Trans. Netw.*, vol. 29, no. 4, pp. 1867–1880, Aug. 2021.

[28] R. Kang, F. He, and E. Oki, "Robust virtual network function allocation in service function chains with uncertain availability schedule," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 3, pp. 2987–3005, Sep. 2021.

[29] H. Sun, C. Liu, H. Chen, and R. Xu, "An improved estimation of distribution algorithm for cloud computing resource scheduling," in *Proc. IEEE Tenth Int. Conf. Adv. Comput. Intell.*, 2018, pp. 484–489.

[30] D. Bhamare, M. Samaka, A. Erbad, R. Jain, L. Gupta, and H. A. Chan, "Multi-objective scheduling of micro-services for optimal service function chains," in *Proc. IEEE Int. Conf. Commun.*, 2017, pp. 1–6.

[31] J. Li, W. Shi, H. Wu, S. Zhang, and X. Shen, "Cost-aware dynamic SFC mapping and scheduling in SDN/NFV-Enabled networks for internet of vehicles," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 5824–5838, Apr. 2022.

[32] H. Cao, H. Zhu, and L. Yang, "Dynamic embedding and scheduling of service function chains for future SDN/NFV-enabled networks," *IEEE Access*, vol. 7, pp. 39 721–39 730, 2019.

[33] L. Wang, Z. Lu, X. Wen, R. Knopp, and R. Gupta, "Joint optimization of service function chaining and resource allocation in network function virtualization," *IEEE Access*, vol. 4, pp. 8084–8094, 2016.

[34] Y. Ran, X. Wu, P. Li, and Y. Luo, "Dynamic virtual measurement function scheduling in software-oriented measurement environment," in *Proc IEEE Int. Conf. Commun.*, 2017, pp. 1–6.

[35] H. A. Alameddine, L. Qu, and C. Assi, "Scheduling service function chains for ultra-low latency network services," in *Proc. IEEE 13th Int. Conf. Netw. Serv. Manage.*, 2017, pp. 1–9.

[36] B. Ren, S. Gu, D. Guo, G. Tang, and X. Lin, "Joint optimization of VNF placement and flow scheduling in mobile core network," *IEEE Trans. Cloud Comput.*, pp. 2022, In Press.

[37] N. Promwongsa, A. Ebrahimzadeh, R. H. Glitho, and N. Crespi, "Joint VNF placement and scheduling for latency-sensitive services," *IEEE Trans. Netw. Sci. Eng.*, vol. 9, no. 4, pp. 2432–2449, Jul./Aug. 2022.

[38] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Netw.*, vol. 8, no. 5, pp. 556–567, Oct. 2000.

[39] G. Lin et al., "Fairness-aware dynamic rate control and flow scheduling for network utility maximization in network service chain," *IEEE J. Sel. Areas Commun.*, vol. 37, 5, pp. 1059–1070, May 2019.

[40] W. Budiarso et al., "Influence of bucket shape and kinetic energy on breastshot waterwheel performance," in *Proc. IEEE 4th Int. Conf. Sci. Technol.*, 2018, pp. 1–6.

[41] A. Kabou, N. Nouali-Taboudjemat, S. Djahel, S. Yahiaoui, and O. Nouali, "Lifetime-aware backpressure: A new delay-enhanced backpressure-based routing protocol," *IEEE Syst. J.*, vol. 13, no. 1, pp. 42–52, Mar. 2019.

[42] D. Gunji, T. Imura, and H. Fujimoto, "Envelope model of load voltage on series-series compensated wireless power transfer via magnetic resonance coupling," in *Proc. IEEE PELS Workshop Emerg. Technol.: Wireless Power*, 2015, pp. 1–6.

Bo Yi (Member, IEEE) is currently a lecturer of computer science and engineering with the Northeastern University of China. He has authored and coauthored more than 20 journal and conference articles on *IEEE Transactions on Cloud Computing*, *IEEE Communications Letter*, *IEEE Access*, *Computer Networks*, etc. He is currently the reviewer of *IEEE Communications Survey & Tutorial*, *Communications Letter*, *Computer Networks*, *Journal of Network and Computer Applications*, etc. His research interests include service computing, routing, virtualization, cloud computing in SDN, NFV, DetNet, etc.

Xingwei Wang received the BS, MS, and PhD degrees in computer science from the Northeastern University, Shenyang, China, in 1989, 1992, and 1998, respectively. He is currently a professor with the College of Computer Science and Engineering, Northeastern University, Shenyang, China. His research interests include cloud computing and future Internet, etc. He has published more than 100 journal articles, books and book chapters, and refereed conference papers. He has received several best paper awards.

Min Huang received the BS degree in automatic instrument, the MS degree in systems engineering, and the PhD degree in control theory from the Northeastern University, Shenyang, China, in 1990, 1993, and 1999, respectively. She is currently a professor with the College of Information Science and Engineering, Northeastern University, Shenyang, China. Her research interests include modeling and optimization for logistics and supply chain system, etc. She has published more than 100 journal articles, books, and refereed conference papers.

Sajal K. Das (Fellow, IEEE) is the chair of Computer Science Department and Daniel St. Clair Endowed chair with the Missouri University of Science and Technology, Rolla. During 2008–2011, he served the US National Science Foundation as a program director with the Division of Computer Networks and Systems. His current research interests include wireless and sensor networks, mobile and pervasive computing, Big Data, cyber-physical systems, smart healthcare, distributed and cloud computing, security and privacy, biological and social networks, applied graph theory and game theory. He published more than 600 research articles in high quality journals and refereed conference proceedings. He holds 5 US patents, coauthored 51 book chapters and four books titled Smart Environments: Technology, Protocols, and Applications (2005), Handbook on Securing Cyber-Physical Critical Infrastructure: Foundations and Challenges (2012), Mobile Agents in Distributed Computing and Networking (2012), and Principles of Cyber-Physical Systems (2016). His h-index is 70 with more than 20,000 citations according to Google Scholar. He received 10 best paper awards in prestigious conferences such as ACM MobiCom'99, IEEE PerCom'06 and IEEE SmrtGridComm'12. He serves as the founding editor-in-chief of the *Pervasive and Mobile Computing journal*, and as associate editor of *IEEE Transactions on Mobile Computing*, *ACM Transactions on Sensor Networks*, and several others.

Keqin Li (Fellow, IEEE) is a SUNY distinguished professor of computer science with the State University of New York. He is also a national distinguished professor with Hunan University, China. His current research interests include cloud computing, fog computing and mobile edge computing, energy-efficient computing and communication, embedded systems and cyber-physical systems, heterogeneous computing systems, Big Data computing, high-performance computing, CPU-GPU hybrid and cooperative computing, computer architectures and systems, computer networking, machine learning, intelligent and soft computing. He has authored or coauthored more than 840 journal articles, book chapters, and refereed conference papers, and has received several best paper awards. He holds more than 70 patents announced or authorized by the Chinese National Intellectual Property Administration. He is among the world's top 5 most influential scientists in parallel and distributed computing in terms of both single-year impact and career-long impact based on a composite indicator of Scopus citation database. He has chaired many international conferences. He is currently an associate editor of the *ACM Computing Surveys* and the *CCF Transactions on High Performance Computing*. He has served on the editorial boards of the *IEEE Transactions on Parallel and Distributed Systems*, the *IEEE Transactions on Computers*, the *IEEE Transactions on Cloud Computing*, the *IEEE Transactions on Services Computing*, and the *IEEE Transactions on Sustainable Computing*.