Journal of Network and Computer Applications 125 (2019) 130–145

Contents lists available at ScienceDirect

Journal of Network and Computer Applications

journal homepage: www.elsevier.com/locate/jnca

Routing as a service (RaaS): An open framework for customizing routing services

Chao Bu[a], Xingwei Wang[b,*], Hui Cheng[c], Min Huang[d], Keqin Li[e]

[a] *School of Computer Science and Engineering, Tianjin University of Technology, Tianjin, 300384, China*
[b] *School of Computer Science and Engineering, Northeastern University, Shenyang, 110819, China*
[c] *School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, L3 5UX, UK*
[d] *College of Information Science and Engineering, Northeastern University, Shenyang, 110819, China*
[e] *Department of Computer Science, State University of New York, New Paltz, NY, 12561, USA*

A R T I C L E I N F O

Keywords:
Software defined networking
Network function virtualization
Routing as a service
Routing service product line
Dynamic software product line

A B S T R A C T

With the emergence of various types of network applications, the user communication requirements for them are becoming more and more diversified and personalized. In order to accommodate the user frequently changing demands for different network applications, the Internet Service Provider (ISP) traditionally purchases and operates new dedicated network equipment, which always incurs high capital expense (CAPEX) and operating expense (OPEX) from the economic viewpoint and also burdens network management. Inspired by the ideas of Software Defined Networking (SDN) and Network Function Virtualization (NFV), we consider dealing with the above challenge by reusing virtualized network functions and selecting appropriate ones to compose the customized routing services on the routing paths for different applications. In this paper, based on SDN and NFV, we propose Routing as a Service (RaaS) as an open framework to customize the specific routing services for applications. Then, we present Routing Service Product Line (RSPL) by introducing Dynamic Software Product Line (DSPL) into the proposed RaaS, so as to rapidly customize a large number of routing services with different characteristics. In addition, according to the proposed framework, we also carry out a case study to customizing routing services with benefits of both the user and the ISP considered. Simulation results show that the proposed RaaS is feasible and efficient.

1. Introduction

With the rapid development of Internet technologies and continuously expanding network size, many new types of network applications are emerging. Meanwhile, the user communication requirements for different network applications become more and more diversified and personalized, bringing great challenges to the traditional routing configuration which is still mainly in manual or command-default mode (Jiang et al., 2016) and difficult to satisfy the user frequently changing demands. In this paper, we propose dynamically providing services on the packets routing paths to achieve adaptive routing configuration. This requires the specialized network functions (i.e., middle-boxes) to implement distinctive packet processing operations beyond simple packet forwarding (Lima and Carvalho, 2011), such as improving security (e.g., firewall, DPI), enhancing performance (e.g., traffic shaping, caching proxy) and ensuring reliability (e.g., failure recovery, resource reservation). Traditionally, most of the professional network functions are usually performed by intermediary devices (Carpenter, 2002), that is, they are implemented based on special hardware (Cheng et al., 2015). In order to satisfy the user frequently changing communication requirements, the Internet Service Provider (ISP) has to keep purchasing and operating new network equipment, which incurs high capital expense (CAPEX) and operating expense (OPEX) (Wu et al., 2015). Therefore, the ISP needs a sustainable method which can satisfy the user high communication requirements with low investment and good time-to-value.

As the newly emerge network paradigms, Software Defined Networking (SDN) (Diego et al., 2015) and Network Function Virtualization (NFV) (Mijumbi et al., 2015) have potentials to deal with the above challenges. SDN decouples the control logic from data plane with the network intelligence being highly concentrated in the control plane. The logically centralized control plane made it easy to adjust network

* Corresponding author.
E-mail address: wangxw@mail.neu.edu.cn (X. Wang).

https://doi.org/10.1016/j.jnca.2018.10.010
Received 4 June 2018; Received in revised form 3 October 2018; Accepted 24 October 2018
Available online 28 October 2018

behaviors, balance network load, manage network functions and configure network services based on the global network view. Thus, SDN can provide the sound basis to support the innovation of routing service configuration mode. NFV decouples network functions from dedicate network equipment on which they run, that is, the hardware based network functions can be virtualized as the plain software based instances, which enables the instantiated network functions to run on commodity hardware. Taking the advantages of SDN and NFV, we consider reusing various virtualized network functions and selecting appropriate ones to compose the customized routing services on the routing paths for different network applications, so as to satisfy the user distinctive communication requirements. In this paper, based on SDN and NFV, we propose Routing as a Service (RaaS) as an open framework to customize routing service.

The routing service customization should be specific and personalized for each user. However, it is very hard for ISP to customize each routing service for each user independently and individually when facing a very large number of users, especially when the real-time demands of users are changing frequently and the new demands of users are emerging rapidly. The reason is that the ISP cannot afford the high service provision cost incurred by high consumption of computation and memory resources and that the ISP lacks the rapid production and deployment capability for the new unexpected services. In addition, users are not willing to bear the high cost of each routing service which is customized from scratch. Therefore, we introduce Dynamic Software Product Line (DSPL) (Bencomo et al., 2012) into the routing service composition, which can rapidly and automatically produce a very large number of different customized routing services for a very large number of users in real-time, meanwhile satisfy each user's personalized communication requirements.

DSPL is a software reuse paradigm to guide the developing software products from a common set of core assets (Hallsteinsen et al., 2013) rather than one by one from scratch (Lee et al., 2002). It composed of two processes: domain engineering and application engineering (Kamoun et al., 2016). In the domain engineering process, DSPL exploits commonality and variability among a family of software products so as to develop the reusable and dynamically reconfigurable core assets. The application engineering is responsible for automatically producing customized software products derived from domain engineering. That is, in the application engineering process, DSPL dynamically selects and reuses appropriate core assets according to the applications' diversified and personalized demands to rapidly customize different software products in the real-time. Thus, DSPL can massively produce software products of a certain domain. In this paper, we consider routing services as software products, and diverse network functions as available components that can be assembled into products. With the technologies of DSPL (e.g., commonality and variability modelling, feature modelling, orthogonal variability modelling), we develop the Routing Service Product Lines (RSPLs), so as to achieve massively customizing diversified routing services for different communication requirements in a rapid and automatic way in the real-time.

In summary, the major contributions of this paper are as follows:

- An open framework, RaaS, is proposed for customizing routing services based on the integration of SDN and NFV. RaaS provides a flexible and effective way to achieve routing service customization by assembling appropriate network functions and further optimize the user experience.
- RSPL is devised by using the idea of DSPL, so that the mass-customization for routing services can be achieved rapidly and automatically in the real-time. The commonality and variability model, the two-layered feature model and the orthogonal variability model are respectively established for RSPL.
- A case study is presented according to the proposed RaaS framework to illustrate how to maximize the ISP economic benefit with the optimization of user service experience considered.

The rest of this paper is organized as follows. In Section 2, we review the related work and compare our work with them. In Section 3, we present the system framework of the proposed RaaS. In Section 4, we describe the details of the established RSPL, and present the two-layered feature model and the orthogonal variability model. In Section 5, we carry out a case study to describe how to optimize the benefits for both the user and the ISP. In Section 6, we present simulation experiments and results. Finally, Section 7 concludes the paper.

2. Related work

There have been a lot of research on routing based on the idea of SDN. In (Yu et al., 2015), an adaptive routing for video streaming (ARVS) with QoS support over SDN was developed. It treats the base layer packets and enhancement layer packets as two levels of QoS flows respectively to reduce packet loss rate and enhance coverage under various network loads. In (Li et al., 2017), by applying the SDN controller to enable the central control of the entire network, a joint optimization model is proposed to consider high bandwidth utilization for provider and low delay for users. In (Chen et al., 2017), a system approach to provide deterministic delay guarantee for dynamic service chaining in SDN is presented. It achieved QoS-guaranteed service chaining by systematically evaluating the delay performance and designing the service traversal path. In (Kosugiyama et al., 2017), based on SDN controller and SDN switches, a method was devised to aggregate flows and minimize the number of flows in a network while all flows satisfy their allowable delay as QoS or SLA. In (Lin et al., 2016), by leveraging SDN's new system architecture, a multi-tenancy management framework was proposed. It enables the jointly optimized design of QoS-aware virtualization and routing by tenant isolation and prioritization as well as flow allocation, fulfilling QoS requirements of tenants' applications. In (Bentaleb et al., 2017), an SDN-enabled streaming architecture called SDNDASH was proposed. It aims to address HTTP adaptive streaming issues including video instability, quality of experience unfairness and network underutilization. The above research configure routing and manage flows based on the logically centralized controller in the SDN paradigm, which makes it easy and flexible to allocate resources and provide services under the global view. However, they cannot select and compose diverse appropriate network functions into new routing services with different characteristics. In addition, these research mainly focus on improving QoS without taking the user personalized requirements into account. In contrast, the proposed RaaS combines the centralized control idea of SDN with the network function virtualization of NFV to provide a programmable and extensible framework with diverse reusable network functions. According to the user personalized requirements, the routing services can be customized with appropriate functions.

Taking the advantages of the integration of SDN and NFV, we have proposed an adaptive routing service customization mechanism in our recent work (Bu et al., 2017). In this work, with the analysis of the relationships among the network operator, the ISP and the user, we presented the market-driven matching schemes and workflows between multiple application requests and multiple candidate services. It aims to achieve the optimal benefit equilibrium between the ISP and the user when customizing and providing routing services. In this paper, the proposed RaaS also takes the advantages of SDN and NFV, however, it aims to achieve automatically and quickly customizing a very large number of routing services in the real-time for a very large number of users with different communication requirements. It promotes routing service mass customization by introducing the software engineering idea (i.e. DSPL) into the processes of network function selection and routing service composition.

There are also some research on providing services with the idea of function composition based on NFV and SDN. In (Bueno et al., 2013), by using the tangible capabilities of SDN together with NFV, an open platform named OpenSCaaS for service chain as a service is presented. It provides an open, novel, and extensible approach for service-providers to enforce service-chaining policy without modifying current SDN standards or mandating any implementation constraints on middle-boxes. In (Wang et al., 2015), a combinatorial optimization model was developed to describe resources and components of dynamic function composition, and further a Markov approximation based distributed algorithm was proposed. In (Cheng et al., 2015), a service chain instantiation framework based on NFV and SDN was proposed. It combines network functions in cooperative and optimal way. In (Gharbaoui et al., 2017), an orchestrator is presented to achieve the adaptive provisioning of high-available and QoS-assured service chain paths in SDN/NFV infrastructures. In (Moyano et al., 2017), a novel network management model is proposed based on NFV and SDN, it allows residential users to define and control access network resources and the dynamic provision of traffic differentiation to fulfill QoS requirements. In (Grigoriou et al., 2017), based on the elasticity of SDN/NFV technologies, an approach for QoE management through cooperation and information exchange among network elements which are involved in the service delivery chain is proposed. Based on the integration of NVF and SDN, the above research can readily develop diversified services by composing various functions to deal with flows on demands. However, they have not explored the idea of mass-customization for routing services when facing large-scale users with different requirements. They also do not consider benefits of both the user and the ISP. In contrast, the proposed RaaS combines DSPL with network function selection and routing service composition, thus achieves the routing service mass-customization. Furthermore, RaaS takes benefits of both the user and the ISP into account when customizing routing services.

There are some research on customizing services by leveraging DSPL. In (Lee et al., 2012), a novel service-oriented product line (SOPL) was proposed, which combines feature-oriented analysis with a self-management QoS framework. In (Gamez et al., 2015), a software product line based approach for stateful service selection problem with transaction and QoS support was proposed. It chooses the best services by matching every task of the workflow to satisfy the user preferences and constraints. In (Kotonya et al., 2009), a consumer-centered approach was proposed. It integrates product line engineering with service-orientation by adapting feature-oriented product-line engineering to service-oriented development. In (Gomaa and Hashimoto, 2011), the dynamic software adaptation for SOPL was proposed, which uses a dynamic feature model for a family of service-oriented architectures. In (Nascimento et al., 2014), a self-adaptive solution was proposed, which leverages the ideas from Software Product Line Engineering to support fault-tolerant composition services. The above research leverage the feature modelling method of DSPL to improve the service customization. However, they mainly focused on the user requirements to establish the feature model without considering other participants (e.g., the ISP). In addition, they do not consider the routing problems when customizing services. In contrast, we have proposed a novel two-layer feature model by leveraging DSPL for routing service customization, and have taken both the user service experience and the ISP profit expectation into account.

There are some research on constructing models to evaluate the user service experience or assess the ISP economic benefit, so as to improve the service quality or optimize the profit. In (Wagle et al., 2015), an evaluation model is proposed, it considers both dynamic and static attributes during the selection of provider. It also verifies the service quality delivered for each service in the SLA commitment. In (Hsu and Lo, 2014), a QoS to Quality of Experience (QoE) mapping and adjustment model is proposed, it translates the network QoS parameters into the user's QoE. Thus, the ISP can use the proposed model to precisely measure and predict its customers' QoE. In (Msakni and Youssef, 2013), a classification of several objective models is presented to assess QoE based on QoS parameters. It also discusses some other factors that must be considered in the mapping process between QoE and QoS. In (Floris et al., 2018), it considers the collaboration among ISPs and OTTs for joint QoE-aware service management in terms of technical and economic aspects. Three different models are proposed to maximize the profit so as to provide different QoE to customers. In (Antonio et al., 2011), a simple QoS-based dynamic pricing approach is proposed, which attempts to increase user's satisfaction level by maximizing the provided QoS level and applying dynamic pricing strategies according to the QoS. It serves as the model to allow service providers to maximize their profits. These research have devised models and approaches of evaluating the user service experience or assessing the ISP economic benefit, by which to improve the service provision. However, most of them mainly focus on the user or the ISP separately, they do not take both of the user experience and the ISP profit together into account when providing services. Moreover, they focus on the user experience on service quality without considering the user experience on service price. They are also short of analyzing the impact of optimizing the ISP profit on the user selectivity for services. In contrast, the proposed RaaS has constructed corresponding evaluation models to deal with the above problems. The user experience on service quality and service price are considered together in the model to evaluate the user service experience. In addition, the user service experience serves as a key factor to optimize the ISP economic benefit when composing and pricing services.

3. The framework of RaaS

The proposed RaaS is based on the idea of decoupling control plane from data plane in SDN and decoupling network functions from physical network equipment in NFV. The framework of RaaS is composed of three layers, i.e., data layer, control layer and application layer. According to the requirements of both the user and the ISP for different network applications, the three layers cooperate with each other to achieve routing service customization. The details of the system framework of RaaS is shown in Fig. 1.

The data layer is in charge of processing packets according to forwarding rules. It consists of a series of network domains which may be built under different environments with different networking modes. Each domain contains multiple switches which are standardized and NFV-enabled (Mijumbi et al., 2015). These switches not only contain simple forwarding functions (i.e., standard IP routing functions), but also can be enhanced and programmed with special-purpose packet forwarding functions (i.e., various types of virtualized network functions). Here, path computation does not decouple from routing service composition, the corresponding routing algorithms serve as candidate network functions to be selected to compose customized routing services. Such enhancement enables the appropriate network functions to be dispatched into the corresponding switches, and the customized routing services to be composed on the paths for different network applications.

Based on the integration of NVF and SDN, the control layer is composed of the routing service orchestration center and the routing service decision-making center. The routing service orchestration center is established to virtualize the underlying network resources to form the virtualized resource pool by the virtualization technology (Lorena et al., 2015). In the virtualized resource pool, the underlying switches and links are virtualized to be logical nodes and logical links respectively, and various types of virtualized network functions are modularly designed with the standardized interfaces. With the virtualized resource pool, the underlying network resources can be monitored, managed and allocated in a unified and global way by the routing service decision-making center. The routing service decision-making center is established to perform mass-customization for routing services. By using the DSPL method, we devise diversified RSPLs for different types of network applications according to their domain knowledge. The RSPL contains reconfigurable core assets and multiple variation points which correspond to multiple feasible selection and composition possibilities for routing services. The variability model, feature model and orthogonal variability model are established in RSPL, so as to support diversified routing service customization according to various requirements of both the user and the ISP. Based on the established RSPLs in the RSPLs set, the

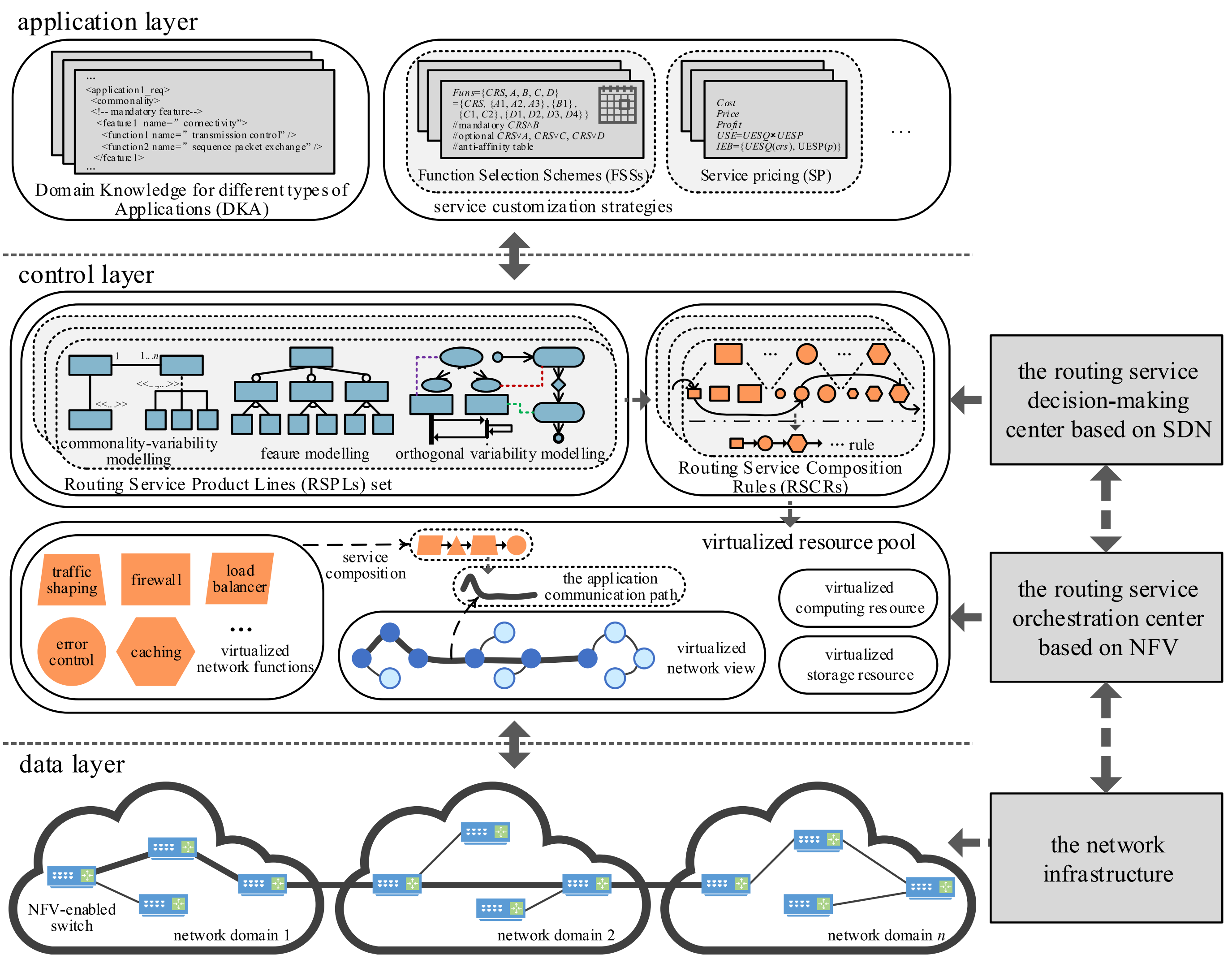

application layer
Domain Knowledge for different types of Applications (DKA)
Function Selection Schemes (FSSs)
Service pricing (SP)
service customization strategies
control layer
commonality-variability modelling
feaure modelling
orthogonal variability modelling
Routing Service Product Lines (RSPLs) set
Routing Service Composition Rules (RSCRs)
rule
the routing service decision-making center based on SDN
traffic shaping
firewall
load balancer
error control
caching
virtualized network functions
service composition
the application communication path
virtualized resource pool
virtualized computing resource
virtualized storage resource
virtualized network view
the routing service orchestration center based on NFV
data layer
NFV-enabled switch
network domain 1
network domain 2
network domain n
the network infrastructure

Fig. 1. The framework of the proposed RaaS.

appropriate network functions are selected to compose the customized routing services. This approach helps to achieve mass-customization for routing services. Each customized routing service optimizes the user service experience and maximizes the ISP profit expectation.

The application layer implements routing logic that supports the routing service decision-making. At this layer, routing control instances are provided to guide the control layer in providing the customized routing services. For example, three instances are shown in Fig. 1. Domain Knowledge for different types of Applications (DKA) is used to establish specific RSPLs. Since the requirements of both the user and the ISP are considered, the Function Selection Schemes (FSSs) and the Service Pricing (SP) strategy are devised to provide the feasible selection and pricing for routing service customization.

3.1. The notions of RaaS

We provide the following definitions and descriptions for the elements used in the RaaS, so as to abstract the routing service customization. And the elements can serve as the basis for routing service development.

Definition 1. An Application Request (AR) is defined as a six-tuple $\langle ar_{id}, R, S_s^{App_{id}}, S_d^{App_{id}}, ep, hp\rangle$. Here, ar_{id} is the application's unique identifier; $R = \{r_1, r_2, \dots, r_q | q \in \mathrm{N}_+\}$ is the set of the communication requirements of the user, and N_+ is the set of positive integers; $S_s^{App_{id}}$ and $S_d^{App_{id}}$ are the source switch and destination switch of ar_{id} respectively; ep and hp represent the user estimated price on the service and the user highest acceptable price to the required service respectively.

Definition 2. An ISP is defined as a five-tuple $\langle DKA, RSPL, FSS, C, sp\rangle$ which provides the routing service to the user. Here, $DKA = \{dka_1, dka_2, \dots, dka_p | p \in \mathrm{N}_+\}$ is the set of domain knowledge for different types of network applications and each element represents domain knowledge for one certain type of application; $RSPL = \{rspl_1, rspl_2, \dots, rspl_p | p \in \mathrm{N}_+\}$ is the set of RSPLs and each $rspl_i$ in $RSPL$ is the established RSPL corresponding to dka_i in DKA, $1 \le i \le p$; $FSS = \{fss_1, fss_2, \dots, fss_n | n \in \mathrm{N}_+\}$ is the set of function selection schemes and its element can be selected to compose the routing service by ISP; $C = \{c_1, c_2, \dots, c_n | n \in \mathrm{N}_+\}$ is the set of costs of implementing different schemes in FSS; sp is the service pricing strategy set by ISP.

For example, we suppose that $dka_i \in DKA$ denotes the domain knowledge of the video-chat type of applications. Its domain knowledge contains all possible communication characteristics required by this type of applications in the ISP's historical records, such as reliable interaction (i.e., connection oriented), private conversation (i.e., encryption and access control), video quality (i.e., traffic shaping, congestion control, bandwidth allocation), and so on. It can be described as Fig. 2.

Definition 3. A RSPL is defined as a five-tuple $\langle rspl_{id}, dka_{id}, CF, DF, NF\rangle$. Here, $rspl_{id}$ is the unique identifier of a RSPL; dka_{id} is the corresponding

```xml
<VideoChatApplication>
    <commonality>
        <feature name="connectivity" function="ConnectionFunction"
                service="ConnectionOriented">
            <protocol name="TransmissionControl">
                routingservice.mandatory.interaction.functionTC
            </protocol>
            <protocol name="SequentialPacketExchange">
                routingservice.mandatory.interaction.functionSPE
             </protocol>
             <protocol name="NetworkBasicIOSystem">
                 routingservice.mandatory.interaction.functionNBIOS
             </protocol>
        </feature>
    </commonality>
    <variability>
```

Fig. 2. The XML fragment example.

domain knowledge which is necessary to establish the RSPL, $dka_i \in DKA$; $CF = \{cf_1, cf_2, \ldots, cf_u | u \in \mathrm{N}_+\}$ and $DF = \{df_1, df_2, \ldots, df_v | v \in \mathrm{N}_+\}$ are common features and different features respectively defined according to dka_{id} for the RSPL; $NF = \{nf_1, nf_2, \ldots, nf_w | w \in \mathrm{N}_+\}$ is the set of selectable network functions to compose routing services based on the RSPL.

Definition 4. A Customized Routing Service (CRS) is defined as a seven-tuple $\langle crs_{id}, rscr, Pac, e1, e2, fs \rangle$. Here, crs_{id} is the CRS's unique identifier; $rscr$ is the corresponding Routing Service Composition Rule (RSCR) which contains the selected functions and their assembling forms to compose the CRS; $e1$ and $e2$ are the endpoints of the CRS; $Pac = \{pac_1, pac_2, \ldots, pac_e | e \in \mathrm{N}_+\}$ is the set of packets forwarded by the CRS; fs is the forwarding state of Pac (including transmission rate, size, loss rate, etc.), which has been dealt with by the CRS. When packets in Pac enter the CRS from one endpoint, they are dealt with then forwarded to the other endpoint under fs.

Definition 5. A Network Function (NF) is defined as a four-tuple $\langle nf_{id}, inp, op, CM \rangle$. Here, nf_{id} is the unique identifier of a network function, $nf_{id} \in NF$; inp and op are the input port and the output port of the function respectively; $CM = \{pullm, pushm, uncertainm\}$ is the set of connection modes of ports. *pullm* represents pull mode and supports pulling packets from another function; *pushm* represents push mode and supports pushing packets to another function; *uncertainm* represents uncertain mode and serves as the connector between *pullm* and *pushm*. *inp* is either *pullm* or *uncertainm*, and *op* is either *pushm* or *uncertainm*. A *pullm inp* of a function can only connect with an *uncertain op* of another function. A *pushm op* of a function can only connect with an *uncertain inp* of another function.

Definition 6. The operator $\bullet$ is defined to do the operation of selecting one element from a set. For example, $NF \bullet nf_i$ means selecting nf_i from NF.

Definition 7. The operator *put* is defined to do the operation of putting nf_i into crs_j by $put(nf_i, crs_k)$.

Definition 8. The operator *del* is defined to do the operation of deleting nf_i from crs_j by $del(nf_i, crs_k)$.

Definition 9. The operator *push* is defined to push packets to a function when packets are ready. If a *push* is operated by nf_i, nf_i is the source function of the *push*.

Definition 10. The operator *pull* is defined to pull packets from a function when packets are ready. If a *pull* is operated by nf_i, nf_i is the destination function of the *pull*.

Inference 1. The operation of replacing one function in a service with another function is achieved by doing one $\bullet$ operation, one *del* operation and one *put* operation.

Inference 2. The operation of composing a network service is achieved by doing $\bullet$ operation and *put* operation for x times, here, $x \in \mathrm{N}_+$.

Inference 3. The operation of forwarding packets by a routing service is achieved by doing *pull* operation for x times and *push* operation for y times by the selected functions in the service, here, $x, y \in \mathrm{N}_+$.

In addition, a CRS can be abstracted as shown in Fig. 3, which can also be used to describe the corresponding RSCR.

3.2. The workflow of RaaS

In the system of RaaS, the control layer cooperates with the application layer through the north-bound interface (Lopes et al., 2016). ISP periodically updates the routing control logic by resetting the DKA component, FSSs component and SP component according to the recent historical records (e.g., service provision information, user feedback information, etc.) which are collected and sent by the control layer. For example, new requested features for some type of applications appear, which should be added into the corresponding DKA; a CRS cannot adapt to new demands, whose functions selection scheme should be improved; and the SP should be changed because of some type of applications becoming popular. Meanwhile, the control layer should adjust the corresponding RSPL models according to the latest state of the components in the application layer, and return the newest service provision records to the application layer in time.

The control layer cooperates with the data layer through the south-

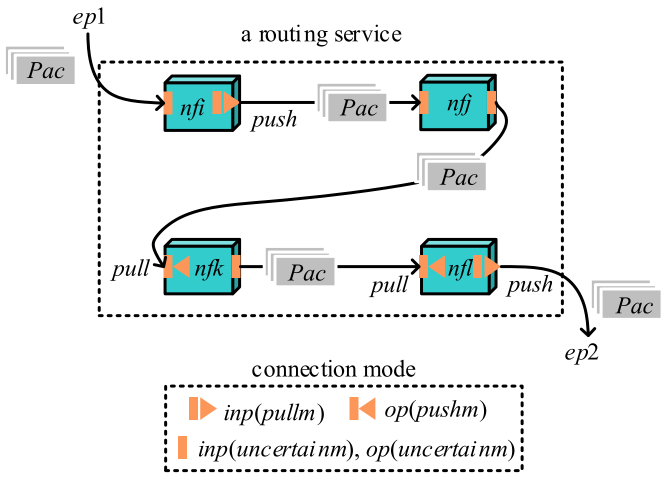


Fig. 3. The method to abstract a CRS. A routing service is treated as a directed acyclic graph here. The selected functions in the service can be treated as nodes in the graph and they deal with packets by *push* and *pull* operations based on the modes of their ports, which can be used to model the RSCR.

bound interface (Lopes et al., 2016). The control layer periodically measures network status with the method of Link Layer Discovery Protocol (LLDP) (Tarnaras et al., 2015), and waits for the application requests sent by the data layer through the packet_in message (Li et al., 2017)). Then, the control layer matches the application requests with the corresponding RSPLs. According to the FSSs and SP the RSCRs are created. At last, the rules are sent to the data layer through the packet_out message (Li et al., 2017) and the appropriate functions are deployed into the appropriate switches on the application communication paths. In addition, the detailed function deployed methods can refer to our previous work (Bu et al., 2018).

Algorithm 1

```
1.  for all new ARs do
2.    Match ARs with RSPLs;
3.      for each $ar_k$ do
4.        Obtain candidate NFs according to RSPL;
5.        Obtain feasible NF sets according to FSSs;
6.        Select the NF set (i.e., $NF$) according to SP;
7.          for each $nf_i \in NF$ do
8.            $NF \bullet nf_i$;
9.            $put(nf_i, crs_k)$;
10.         end for
11.       Provide $crs_k$ to the application;
12.     end for
13. end for
```

Algorithm 2

```
1.  for all existing ARs do
2.    if requirements in ARs are changing then
3.      for each $ar_k$ do/* $crs_k$ denotes its current and $NF_k$ denotes the functions in $crs_k$
    */
4.        Reselect the $NF$ according to RSPL, FSSs and SP;
5.        for each $nf_i \in (NF_k - NF \cap NF_k)$ do
6.          $del(nf_i, crs_k)$;
7.        end for
8.        for each $nf_i \in (NF - NF \cap NF_k)$ do
9.          $(NF - NF \cap NF_k) \bullet nf_i$;
10.          $put(nf_i, crs_k)$
11.        end for
12.        Provide $crs_k$ to the application;
13.     end for
14.   end if
15. end for
```

Algorithm 3

```
1.  for all existing ARs do
2.    if ARs are finishing then
3.      for each $ar_k$ do/* $crs_k$ denotes its current and $NF_k$ denotes the functions in $crs_k$
    */
5.        for each $nf_i \in NF_k$ do
6.          $del(nf_i, crs_k)$;
7.        end for
8.        Release $crs_k$;
9.      end for
10.   end if
11. end for
```

In this paper, the specific algorithms are devised for the main workflows running in the system, which are Algorithm 1 for new ARs coming, Algorithm 2 for existing ARs change, and Algorithm 3 for existing ARs accomplishment.

4. Routing Service Product Line

When a large number of users propose various personalized communication requirements for different network applications, ISP should be able to recognize differences among these requirements to achieve routing service customization in addition to the mass production of routing services. Thus, we present RSPL by using the methods of DSPL to devise the commonality and variability model, the feature model and the orthogonal variability model.

The relationships among these three models can be described as follows. (1) The commonality and variability model serves as the rules to establish the feature model by predefining the association relationships of commonality and variability among different routing services. (2) The feature model is established according to the corresponding DKA. It serves as the basis to create the orthogonal variability model in real-time by providing certain variation points and their corresponding variants. (3) According to the current requirements, new CRSs are composed and existing CRs are adjusted rapidly based on the certain created orthogonal variability models.

4.1. Commonality and variability modeling for RSPL

In order to achieve routing service customization, we predefine commonality and variability of the RSPL. The commonality among different routing services composed by a RSPL is modeled as common feature or mandatory feature of the RSPL. For example, all routing services which support the applications of VoIP type need the feature of interactivity. The variability among different routing services composed by a RSPL is modeled as different features which may be alternative or optional. The alternative features indicate that no more than one feature can be selected for routing services. For example, connectivity is an alternative feature because a routing service is either connection-oriented or connectionless. The optional features are selectable for routing services of a certain RSPL. For example, QoS feature is an optional feature because a routing service can select single or multiple kinds of QoS involved algorithms (e.g., packet scheduling, traffic shaping, buffer allocation, etc.) to satisfy different QoS requirements.

We define that each common feature corresponds to one mandatory point which must be selected when composing routing services. We define that each different feature (i.e., alternative feature or optional feature) corresponds to one variation point, and the selectable elements of each different feature corresponds to the selectable variants of each variation point. In this way, the variation points serve as the key to compose diverse routing services with the personalized requirements considered. Therefore, the commonality and variability model of the RSPL is illustrated in Fig. 4.

In this approach, common features and different features are defined for RSPL. It provides reusable basis for customizing routing services, and makes customization flexible and extendable. For example, according to the selectable variants of each variation point, we consider the corresponding selectable network functions as variants. Thus, it is easy to embed new network functions and improve existing functions according to the certain variation point. Moreover, maintenance cost can be reduced. For example, once one function is changed, the changes can be delivered to all RSPLs that use this function. It saves extra cost of independently tracing and changing each special related service. In addition, the management complexity of routing service can be simplified. For example, when the user requirements or the network status change, the routing service does not need to be entirely recomposed from scratch. Instead, it only adjusts a few appropriate network functions for the changed variation points in RSPL.

4.2. Feature modeling for RSPL

Feature modeling is the activity of identifying the externally visible characteristics of products in a product line and organizing them into a model, called feature model (Hinchey et al., 2012). Here, the visible characteristics of routing services can be described as their capabilities, such as the used domain technologies, protocols, algorithms and so on. According to the commonality and variability model, the different features are considered as the key to customize diversified and personalized routing services, thus the feature model is established mainly based on the different features mentioned above. In the feature model, we also should take the requirements of both the user and the ISP into account, and devise a separate layer in the model to guide function selection and

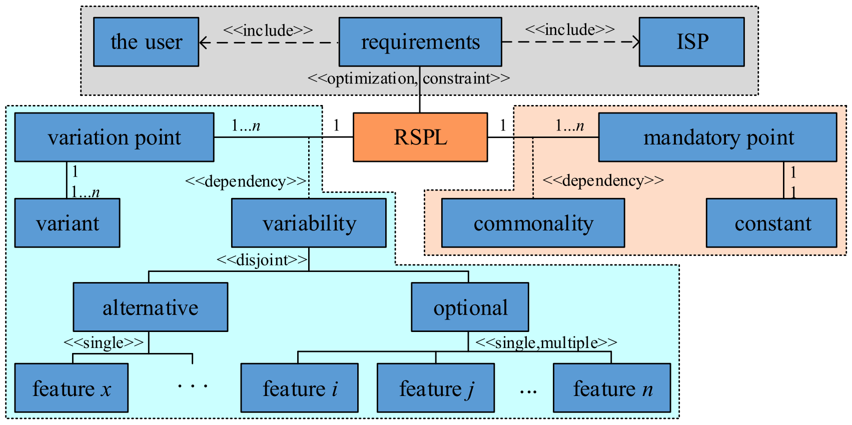


Fig. 4. The commonality and variability model.

composition. Therefore, a two-layer feature model which contains non-functional feature layer and function feature layer is proposed as shown in Fig. 5. Here, the static-composition refers to the elements which are mandatory for the routing services composed by a RSPL; the dynamic-composition refers to the elements which are dynamically selectable for the routing services composed by a RSPL according to the real-time requirements.

In the non-function feature layer, the non-functional features such as the user service experience and the ISP economic benefit are necessary to customize different routing services. Although they do not serve as the entity components of routing services, they serve as the optimization conditions or constraints when composing CRSs. We consider two factors which may influence the user service experience, which are the user experience on service quality and the user experience on service price. In this paper, we evaluate the user experience on service quality by the actual parameter values (e.g., delay, jitter, error rate, etc.) which can be provided by the routing service, and the user experience on service price by the relationship between the user maximum acceptable price and the actual service price. For the ISP economic benefit, the costs will be different with different FSSs which also lead to different parameter values. In addition, the pricing strategy should also be set reasonably to maximize the ISP profit without reducing the user acceptability for the routing service.

In the function feature layer, the functional features (e.g., transmission feature, QoS feature, etc.) form the possible capabilities of routing services, and the corresponding selectable functions form the entity components of routing services. The CRSs are considered as the special software products, and the network functions are considered as the optional software components which can be composed into the CRSs. According to the commonality and variability model, we treat the different features (i.e., QoS feature, security feature) as the variation points, and the functions under them as the selectable variants. In this approach, according to the non-functional features (i.e., optimization conditions, constraints) in the non-functional layer, the appropriate functions are selected for the corresponding functional features to compose the CRSs with different capabilities.

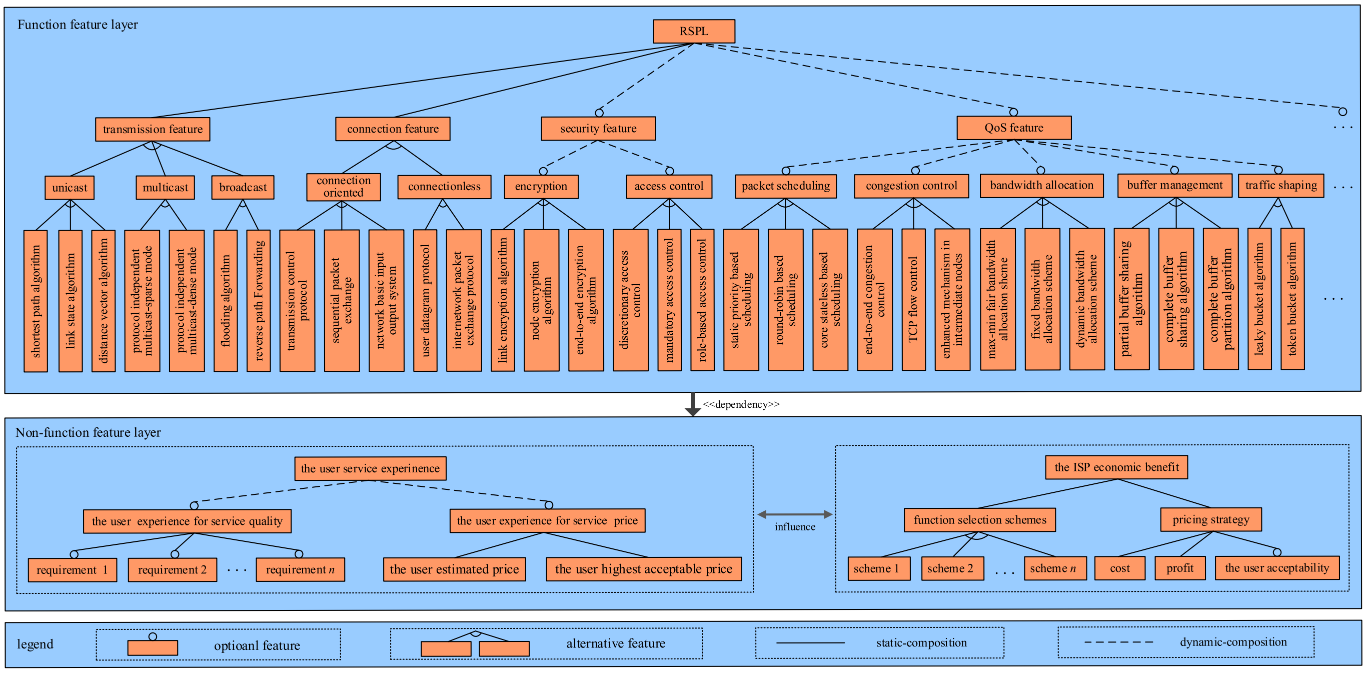

Function feature layer
RSPL
transmission feature
connection feature
security feature
QoS feature
. . .
unicast
multicast
broadcast
connection oriented
connectionless
encryption
access control
packet scheduling
congestion control
bandwidth allocation
buffer management
traffic shaping
. . .
shortest path algorithm
link state algorithm
distance vector algorithm
protocol independent multicast-sparse mode
protocol independent multicast-dense mode
flooding algorithm
reverse path Forwarding
transmission control protocol
sequential packet exchange
network basic input output system
user datagram protocol
internetwork packet exchange protocol
link encryption algorithm
node encryption algorithm
end-to-end encryption algorithm
discretionary access control
mandatory access control
role-based access control
static priority based scheduling
round-robin based scheduling
core stateless based scheduling
end-to-end congestion control
TCP flow control
enhanced mechanism in intermediate nodes
max-min fair bandwidth allocation shcme
fixed bandwidth allocation scheme
dynamic bandwidth allocation scheme
partial buffer sharing algorithm
complete buffer sharing algorithm
complete buffer partition algorithm
leaky bucket algorithm
token bucket algorithm
. . .
<<dependency>>
Non-function feature layer
the user service experience
the user experience for service quality
the user experience for service price
requirement 1
requirement 2
. . .
requirement n
the user estimated price
the user highest acceptable price
influence
the ISP economic benefit
function selection schemes
pricing strategy
scheme 1
scheme 2
. . .
scheme n
cost
profit
the user acceptability
legend
optioanl feature
alternative feature
static-composition
dynamic-composition

Fig. 5. The feature model.

4.3. *Orthogonal variability modeling for RSPL*

The orthogonal variability model is established to plan the complete workflow to analyze, design and run the routing service in real-time according to the current requirements based on the corresponding feature model. As suggested above, the variation point serves as the key to compose diverse routing services, its corresponding orthogonal variability model is also in charge of achieving the traceability of variability in different software phases when customizing routing services. For example, when the variants of some variation points have to be changed (e.g., be added, be replaced, be deleted, etc.) due to the changed requirements or network status, each variation point should keep consistency throughout the routing service lifecycle. Especially for the running services, they can be rapidly and accurately adjusted based on traceability due to variation point being basic element, so that it does not need to re-analyze, re-design and re-compose these services from scratch.

In this paper, we mainly take the QoS variation point as an example to illustrate the orthogonal variability model in detail in Fig. 6. As shown in Fig. 6, based on the QoS variation point (i.e., Fig. 6(A)), the variability is extended to other diagrams at different phases of composing a routing service, so as to associate variants with the selected function components. The involved diagrams in the orthogonal variability model include static view diagrams, such as class diagram (i.e., Fig. 6(B)) and use case diagram (i.e., Fig. 6(C)), and dynamic view diagrams, such as activity diagram (i.e., Fig. 6(D)) and sequence diagram (i.e., Fig. 6(E)). The static view diagram reflects the structural characteristics of QoS variation point in a routing service. For example, the class diagram shows the classes of a service and their relationships, and the use case diagram shows the system blueprint of a service. The dynamic view diagram reflects the behavioral characteristics of QoS variation point in a routing service. For example, the activity diagram shows the object interaction in a service, and the sequence diagram shows the event sequence among different objects.

In the orthogonal variability model, the variation point serves as the basic element of customizing routing services, and the overall process of composing a routing service in real-time according to a feature model are also illustrated. In the class diagram of Fig. 6(B), multiple network functions which can adjust parameters of QoS are classified and enumerated as the candidate ones to be selected on demand. For example, quite a few algorithms (e.g., partial buffer sharing, complete buffer sharing, complete buffer partition, etc.) can achieve buffer management. However, just one of them can be selected by the system as one of functional components in a specific routing service. In the use diagram of Fig. 6(C), participants and use cases are used to describe the QoS functional requirements of the system, so as to bulid the required service according to the requirement analysis. In the activity diagram of Fig. 6(D), it describes the behaviors when composing functions, and shows the order of activations executed by the involved classes. It also supports multiple thread description when multiple functions are simultaneously allocated. The interactive processes and the exchanged

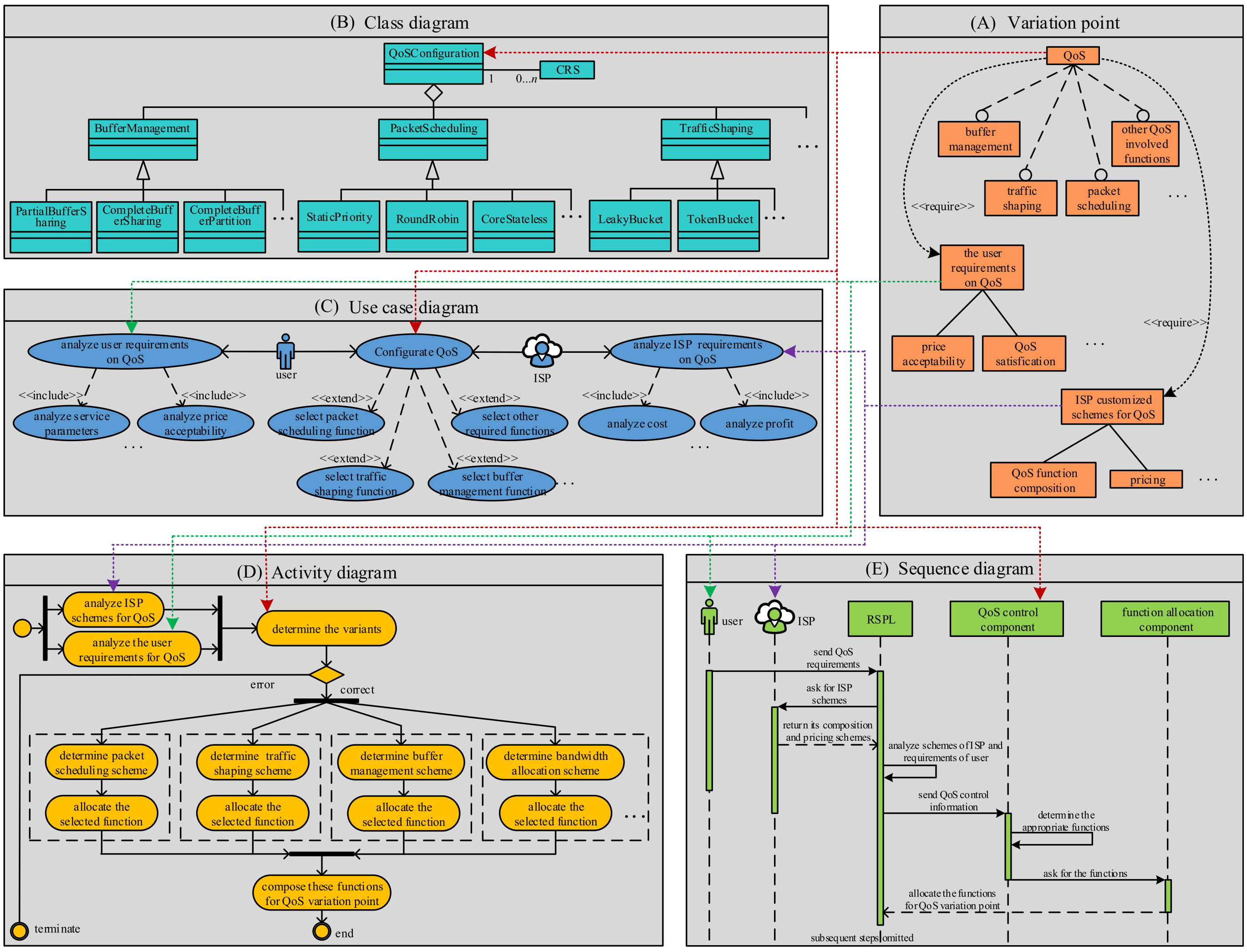

(B) Class diagram
QoSConfiguration
CRS
1
0...n
BufferManagement
PacketScheduling
TrafficShaping
PartialBufferSharing
CompleteBufferSharing
CompleteBufferPartition
StaticPriority
RoundRobin
CoreStateless
LeakyBucket
TokenBucket
(A) Variation point
QoS
buffer management
other QoS involved functions
traffic shaping
packet scheduling
<<require>>
the user requirements on QoS
price acceptability
QoS satisfication
ISP customized schemes for QoS
QoS function composition
pricing
(C) Use case diagram
analyze user requirements on QoS
user
Configurate QoS
ISP
analyze ISP requirements on QoS
<<include>>
<<extend>>
analyze service parameters
analyze price acceptability
select packet scheduling function
select other required functions
analyze cost
analyze profit
select traffic shaping function
select buffer management function
(D) Activity diagram
analyze ISP schemes for QoS
analyze the user requirements for QoS
determine the variants
error
correct
determine packet scheduling scheme
determine traffic shaping scheme
determine buffer management scheme
determine bandwidth allocation scheme
allocate the selected function
compose these functions for QoS variation point
terminate
end
(E) Sequence diagram
user
ISP
RSPL
QoS control component
function allocation component
send QoS requirements
ask for ISP schemes
return its composition and pricing schemes
analyze schemes of ISP and requirements of user
send QoS control information
determine the appropriate functions
ask for the functions
allocate the functions for QoS variation point
subsequent steps omitted

Fig. 6. The orthogonal variability model for QoS variation point.

messages among the user, ISP and RSPL system with respect to the time sequence are described in the sequence diagram of Fig. 6(E). It also emphasizes the sequential occurrence of events when running a service, so as to keep the involved components working orderly.

5. A case study

When providing routing services, ISP tends to maximize its own profit in two ways. One is to reduce the cost by choosing the suitable FSS and the other is to raise the price for the customized service by its pricing strategy. However, the FSS and pricing strategy also influence the User Service Experience (USE) which consists of the User Experience on Service Quality (UESQ) and the User Experience on Service Price (UESP) which further affect the user in selecting the customized routing service. For example, the user may always reject the service with poor quality and/or high price, and then ISP cannot get any profit. Therefore, it is necessary to maximize ISP profit whilst the user service experience is considered. A possible way is to formulate and exploit the relationship among the user, the ISP, and the CRSs.

In this section, we present a case study to illustrate how to practically take the requirements of both the user and the ISP in to account according to the instances (i.e., FSSs and SP) in the application layer of the proposed RaaS framework. In this paper, the FSSs is supposed to contain the combination constraints among network functions, for example, the anti-affinity between two functions nfi and nfj (i.e., $AAnf_{i,j} \neq 1$), which indicates that nfi and nfj cannot be implemented in a same service because of their conflicting resource demands or operating conditions. Meanwhile, the SP is set to maximize the ISP Economic Benefit (IBE). In this approach, according to the feature demands for the routing service, the set of feasible function combinations is denoted as the following *Funs*:

$$\begin{aligned}
Funs &= \{CRS, V_1, V_2, \dots, V_m\} && s.t.\ \ CRS \leftarrow V_1 \wedge V_2 \wedge \dots \wedge V_m \\
&= \{CRS, \{nf_1, nf_2, nf_{i-1}\}, && nf_1, nf_2, nf_{i-1} \in V_1 \\
&\quad \{nf_i, nf_j, nf_{j+1}, nf_{j+2}\}, \dots, && nf_i, nf_j, nf_{j+1}, nf_{j+2} \in V_2 \\
&\quad \{nf_l, nf_{l+1}, nf_k\}\} && nf_l, nf_{l+1}, nf_k \in V_m \\
&\Rightarrow \{\{crs_1, nf_1, nf_{j+1}, \dots, nf_{l+1}\} && AAnf_{1,j+1} \wedge \dots \wedge AAnf_{j+1,l+1} = 1 \\
&\quad \vee \{crs_2, nf_2, nf_{j+2}, \dots, nf_k\} \vee \dots && AAnf_{2,j+2} \wedge \dots \wedge AAnf_{j+2,k} = 1 \\
&\quad \vee \{crs_n, nf_{i-1}, nf_j, \dots, nf_k\}\} && AAnf_{i-1,j} \wedge \dots \wedge AAnf_{j,k} = 1 \\
&= \{fss_1, fss_2, \dots, fss_n\}. && fss_1, fss_2, \dots, fss_n \in FSS.
\end{aligned}$$

Here, $V_1, V_2, \dots, V_m$ are the corresponding variants of the involved features, and $nf_i, nf_j, \dots, nf_k$ are the optional functions of each variant. Thus, the set of n feasible function combinations $FSS = \{fss_1, fss_2, \dots, fss_n\}$ (i.e., Definition 2) can be obtained. All the schemes in *FSS* all can achieve the same routing service customization, however, they lead to different costs because of different service composition overheads (e.g., the required resources, the maintenance expenses, etc.). In addition, they also bring different service experiences to users. For example, the cost of fss_1 may be higher because it provides high-quality high-price service, on the contrary, the cost of fss_2 may be lower because it provides cost-effective service. The detailed descriptions are shown in Fig. 7.

5.1. The USE evaluation model

For multiple schemes in *FSS*, we propose to acquire the user selectivity on each of them by evaluating their influences on the USE (i.e., UESQ and UESP). For UESQ, assume that $r_1, \dots, r_c, \dots r_q$ are the service parameters to evaluate the schemes in *FSS*. In general, the user usually do not have professional knowledge to accurately express their quantitative requirements for the parameters, we use the intervals $[r_1^l, r_1^h]$, …, $[r_c^l, r_c^h]$, …, $[r_q^l, r_q^h]$ to represent the user's inaccurate requirements on $r_1, \dots, r_q$. There are two cases about the evaluation on parameter values. In one case, the higher the actual parameter value is, the better the user evaluation on this parameter (e.g., bandwidth) is, for example, if r_c belongs to this case, its ideal value r_c^+ and negative ideal value r_c^- can be defined as follows:

$$\begin{cases} r_c^+ = r_c^h \\ r_c^- = r_c^l \end{cases} \tag{1}$$

In the other case, the lower the actual parameter value is, the better the user evaluation on this parameter (e.g., delay) is, for example, if r_q belongs to this case, its ideal value r_q^+ and negative ideal value r_q^- can be defined as follows:

$$\begin{cases} r_q^+ = r_q^l \\ r_q^- = r_q^h \end{cases} \tag{2}$$

In this approach, the UESQ on fss_i ($fss_i \in FSS$) can be defined as follows:

$$d_i^+ = \sqrt{\sum_{c=1}^{q} \left(av_c - r_c^+\right)^2} \tag{3}$$

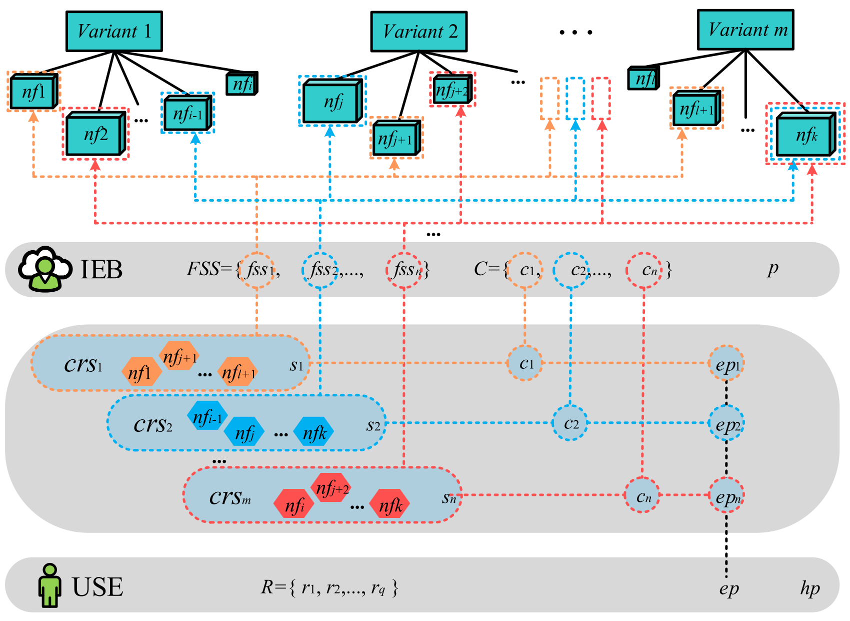


Fig. 7. The detailed description among the user, the ISP and the CRSs.

$$d_i^- = \sqrt{\sum_{c=1}^{q} (av_c - r_c^-)^2} \tag{4}$$

$$UESQ(fss_i) = \frac{d_i^-}{d_i^+ + d_i^-} \tag{5}$$

Here, av_c is the actual parameter value of r_c provided by the service composed by fss_i. d_i^+ is the Euclidean distance between the actual parameter values and the ideal parameter values, and d_i^- is the Euclidean distance between the actual parameter values and the negative ideal parameter values.

For UESP, we consider that it is affected by the actual price p set by ISP, the user estimated price ep, and the user highest acceptable price hp for the service. We define UESP as follows:

$$UESP = \begin{cases} 1, & p \le ep \\ 1 - \dfrac{p - ep}{hp - ep}, & ep < p < hp \\ \varepsilon, & p = hp \\ 0, & p > hp \end{cases} \tag{6}$$

Here, $0 < \varepsilon \ll 1$. Apparently, the higher the p set by ISP, the lower the $UESP$ is. When p set by ISP approaches hp, the $UESP$ is tends to ε. Once the $UESP$ is higher than hp, the $UESP$ is 0. In contrast, when p approaches ep, especially when p is lower than or equal to ep, the $UESP$ tends to or reach to its highest value 0.

In summary, we use UESQ and UESP together to determine USE, shown as follows:

$$USE = UESQ \times UESP \tag{7}$$

Here, we use the multiplication relationship, so as to indicate that USE depends on both of UESQ and UESP. Only when both of UESQ and UESP have high values, the user will have good service experience. If either of them is too low, the user will have bad service experience. Especially, if either of them tends to 0, the user will not select the corresponding service.

5.2. The IBE evaluation model

In order to maximize IBE, ISP should choose suitable FSS and price service reasonably. Assume that the cost of composing service crs_i by fss_i is c_i and p is the price set by ISP for the service composed by fss_i, $fss_i \in FSS$, $ci \in C$. Thus, the profit pro_i expected by ISP is defined as follows:

$$pro_i = (p - c_i) \times USE(crs_i, p) \tag{8}$$

Here, $USE(crs_i, p)$ is the USE for the service crs_i composed by fss_i with p according to Eq. (7), and it acts as the probability that the user accepts the service. According to Eq. (5) and Eq. (6), Eq. (8) can be extended as follows:

$$pro_i = (p - c_i) \times (UESQ(crs_i) \times UESP(p))$$
$$= (p - c_i) \times \begin{cases} UESQ(crs_i), & p \le ep \\ \begin{pmatrix} UESQ(crs_i) \\ \times \left(1 - \dfrac{p - ep}{hp - ep}\right) \end{pmatrix}, & ep < p < hp \\ UESQ(crs_i) \times \varepsilon, & p = hp \\ 0, & p > hp \end{cases} \tag{9}$$

Here, $0 < \varepsilon \ll 1$.

In fact, the user knows the actual value of ep, however, ISP does not know it. Assume that ISP knows the user hp through his historical transaction experiences. For ISP, the value of ep belongs to $[0, hp]$ and obeys a certain distribution function F. Therefore, ISP uses $F(ep)$ as the user estimated price, and makes its pricing strategy to maximize its profit according to $F(ep)$. Thus, replace ep in Eq. (9) with $F(ep)$ as follows:

$$pro_i' = (p - c_i) \times (UESQ(crs_i) \times UESP(p))$$
$$= (p - c_i) \times \begin{cases} UESQ(crs_i), & p \le F(ep) \\ \begin{pmatrix} UESQ(crs_i) \\ \times \left(1 - \dfrac{p - F(ep)}{hp - F(ep)}\right) \end{pmatrix}, & F(ep) < p < hp \\ UESQ(crs_i) \times \varepsilon, & p = hp \\ 0, & p > hp \end{cases} \tag{10}$$

In Eq. (10), c_i is known and $UESQ(crs_i)$ can be obtained by Eq. (7). When $p = hp$ and $p > hp$, pro_i approaches 0. Therefore, the maximum profit can be obtained under $p \le F(ep)$ or $F(ep) < p < hp$.

Under the condition of $p \le F(ep)$, ISP can maximize its profit $pro_i^{\max}$ with $p = F(ep)$:

$$pro_i^{\max} = (F(ep) - c_i) \times UESQ(crs_i) \tag{11}$$

Under the condition of $F(ep) < p < hp$, ISP can maximize its profit $pro_i^{\max}$ with $p = (hp + c_i)/2$ according to the first-order condition:

$$pro_i'^{\max} = \frac{(hp - c_i)^2}{4(hp - F(ep))^2} \times UESQ(crs_i) \tag{12}$$

Then, the maximum profit $pro_i^{*\max}$ can be obtained by comparing $pro_i^{\max}$ in Eq. (11) and $pro_i'^{\max}$ in Eq. (12):

$$pro_i^{*\max} = \begin{cases} (F(ep) - c_i) \times UESQ(crs_i), & F(ep) < \dfrac{hp + c_i}{2} \\ \dfrac{(hp - c_i)^2}{4(hp - F(ep))^2} \times UESQ(crs_i), & F(ep) \ge \dfrac{hp + c_i}{2} \end{cases} \tag{13}$$

Meanwhile, according to Eq. (11), Eq. (12) and Eq. (13), the pricing strategy $p^{\max}$ which maximizes ISP profit can be obtained as follows:

$$p^{\max} = \begin{cases} F(ep), & F(ep) < \dfrac{hp + c_i}{2} \\ \dfrac{hp + c_i}{2}, & F(ep) \ge \dfrac{hp + c_i}{2} \end{cases} \tag{14}$$

In fact, $p^{\max}$ is obtained for each FSS. And according to Eq. (8), the most suitable fss_i with IEB maximized can be chosen by comparing the profits of FSSs.

6. The simulation experiments

6.1. The simulation setup

To simulate the proposed RaaS, we use the Floodlight (Project Floodlight Open) as the control center of the framework for customizing routing services. It generates rules and distributes them to the switches on the application communication paths. The rule contains fields of matching and fields of instructions. The fields of matching are used to match requests for routing services. The fields of instructions contain action instructions (e.g., forwarding to controller, dropping, allocating functions, ordering actions) which are used to deal with packets, and function IDs which identify the available functions. Taking the OpenFlow as the example, the rule based on it in this paper is shown in Fig. 8.

We use OpenFlowClick (OpenFlowClick) to simulate the switch. OpenFlowClick is a kind of software based switch which is created based

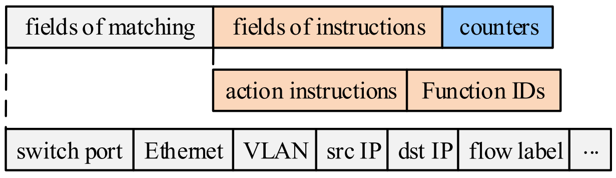
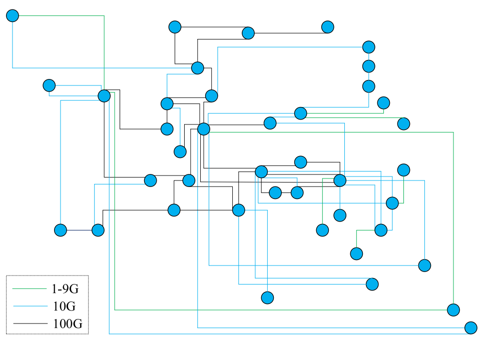


Fig. 8. The example of FTE.

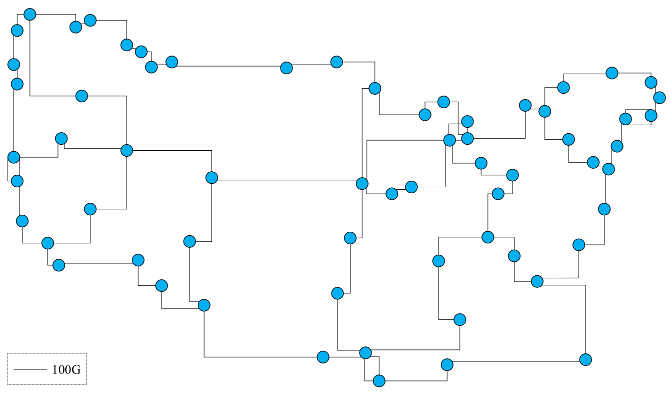
on Click modular router (Kohler et al., 2000) with OpenFlowClick element (Mundada et al., 2009) embedded into it. The Click modular router is extendable, programmable, and assembled by a series of packet processing modules (i.e., diverse virtualized network functions) called elements that can be flexibly selected and added. OpenFlowClick element allows a controller to install rules to guide packet processing through multiple selected network functions that serve as elements in Click modular router. It allows multiple Click modular routers being controlled by one single controller.

The simulation environment is established on Linux platform (Intel core i5 3.3 GHz, 16 GB DDR3 RAM). We adopt two typical and real network topologies which are GéANT (The Internet Topology Zoo) and INTERNET2 (INTERNET2) with different numbers of nodes and links and different bandwidth distribution to evaluate the proposed RaaS. The GéANT and INTERNET2 are shown in Fig. 9. They have different characteristics, such as the Total Number of Nodes (TNN), the Total Number of Links (TNL), the Number of Links with bandwidth Less than 10G (NLL), the Number of Links with bandwidth Between 10G and 100G (NLB), and the Number of Links with bandwidth More than 100G (NLM), which are shown in Table 1.

The resources of nodes such as CPU, storage and memory, follow a uniform distribution between 500 and 1000 units. The cost of each unit capacity is 1 per time unit. The requests arrive over time following the Poisson distribution, and each of the request has a life-cycle follows an exponential distribution with 1000 time units in average. The number of



(a) GéANT



(b) INTERNET2

Fig. 9. The network topologies.

Table 1
Characteristics of the two topologies.

	GéANT	INTERNET2
TNN (nodes)	41	64
TNL (links)	65	78
NLL (links < 10G)	8	0
NLB (10G < links < 100G)	30	0
NLM (links > 100G)	27	78

network functions required by each request follows a uniform distribution between 2 and 10, and the type of each network function is random. The node resources needed to support each network function follow the uniform distribution between 1 and 10 units. The simulation parameters and the corresponding distribution model are motivated by literature such as (Khan et al., 2016) and (Sahhaf et al., 2015), which study the network function provision problem. In this paper, we focus on the variation point of QoS to simulate the proposed RaaS and evaluate its performance. In particular, we mainly consider four parameters (e.g., delay, jitter, error rate and bandwidth) for QoS, and their demand intervals are based on the quality standard bounds (Network performance, 2011) and (Definitions of terms, 2008). For comparison purpose, we choose another current popular service composition approach based on service-chaining policy and compare it with RaaS under the integration of SDN and NFV environment. The chosen approach is OpenSCaaS (Bueno et al., 2013), and we simulate it for composing routing services.

6.2. The simulation results

We compare routing service setup time of the two approaches. The setup time is defined as the interval from the routing service request being received to the routing service being provided. Here, the setup time consists of two part: the service composition time and the service calculation time. The results are shown in Figs. 10 and 11.

When the number of candidate functions increases, the routing service setup time increases; when the number of flows supported by RaaS and OpenSCaaS increases, the routing service setup time increases too. The time overhead of RaaS always increases slower and less than that of OpenSCaaS, especially when the number of flows increases sharply. In more detail, the service calculation time of RaaS is less than 15% of the total service setup time even under the peak network load (i.e., Figs. 10(d) and 11(d)), while the service calculation time of OpenSCaaS is approximately 25% of the total service setup time at the same situation. Furthermore, the service composition of RaaS always takes less time than that of OpenSCaaS at the same situation. For example, the service composition of OpenSCaaS just takes only more 5 ms than that of RaaS under the lightest network load (i.e., Figs. 10(a) and 11(a)), while the former takes more 20 ms than the latter under the peak network load. The reason is as follows. Based on RSPL, RaaS have established the commonality and variability among different types of routing services, which can support quickly composing different services of a certain type. RaaS does not need to calculate all routing services entirely even for the new ones, it only needs to calculate the differences in the corresponding variation points without calculating each service from scratch every time. Moreover, according to the established feature model and the orthogonal variability model, RaaS can reuse appropriate core functions and automatically select candidate functions to customize routing services. In this approach, the routing services are produced as software products and diverse network functions are allocated as available components which can be assembled into products, which effectively improve the efficiency of routing service composition. However, OpenSCaaS does not consider rapidly reusing core functions of different types of routing services, and it does not assembles functions into services through the product line method. Thus, OpenSCaaS has to calculate and compose each routing service separately from scratch which spends much time, especially when the number of flows increases greatly.

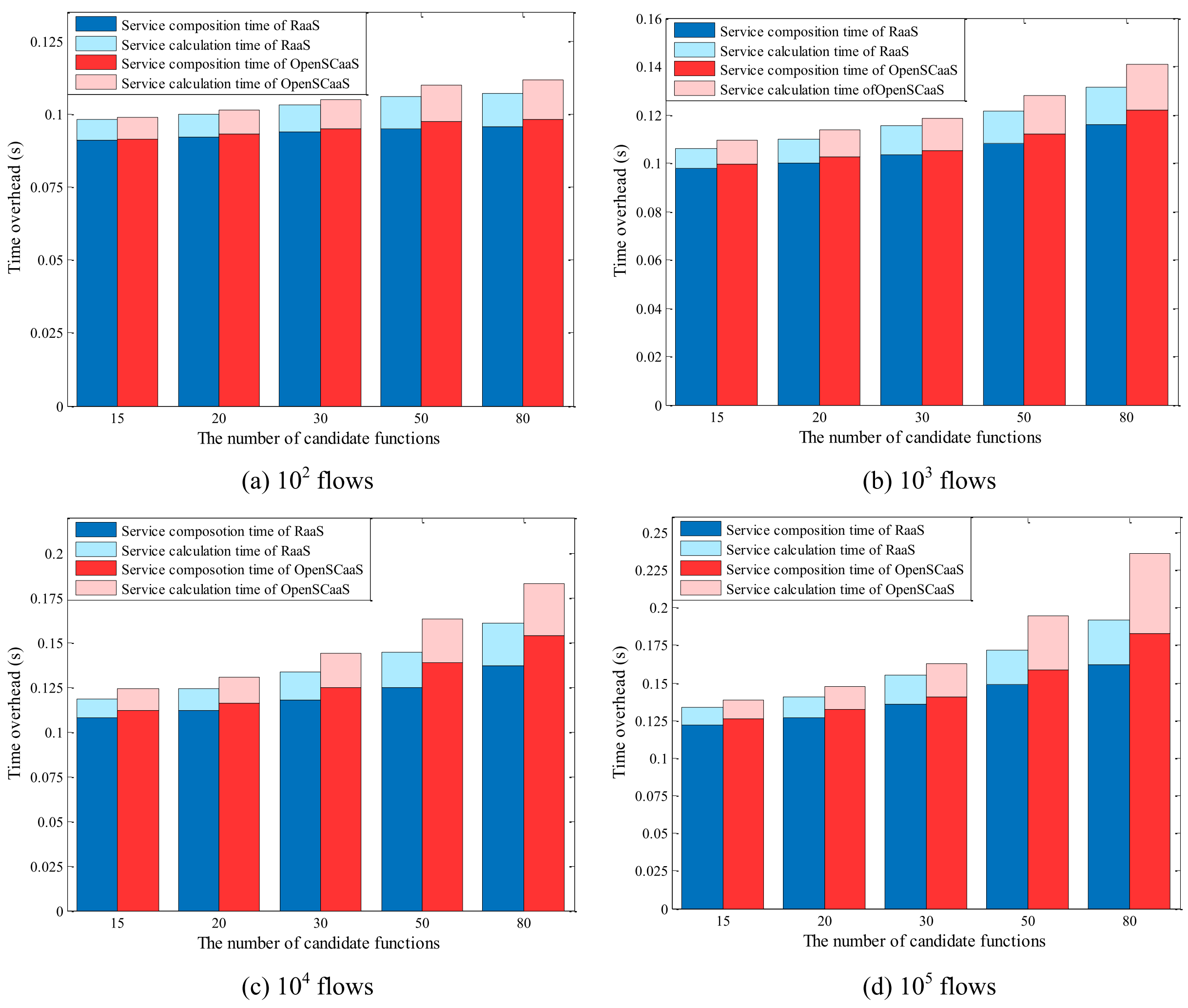


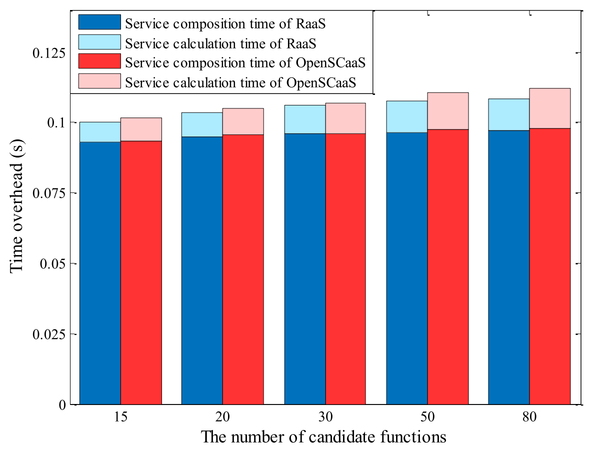
Fig. 10. Time overhead over GéANT.

We compare the user and the ISP satisfaction degree on the CRS composed by the two approaches. The user satisfaction degree is calculated by Eq. (7). ISP satisfaction degree is the ratio of its actually got profit to the service price. The results are shown in Figs. 12 and 13.

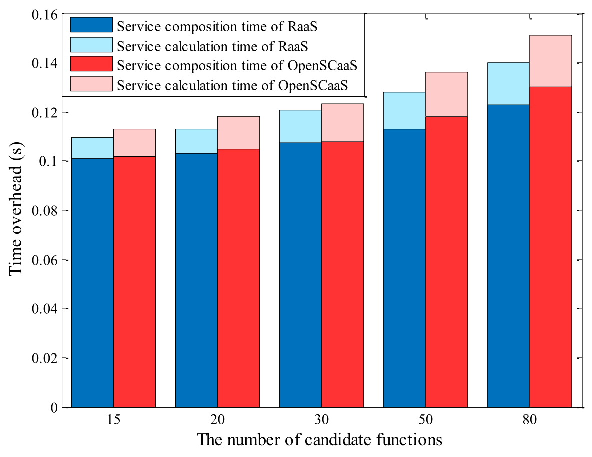
The average satisfaction degree on the services customized by RaaS is much higher than that by OpenSCaaS. In more detail, the user satisfaction degree and the ISP satisfaction degree of RaaS can reach to 96% and 95% respectively with the number of candidate function increasing, while they can only reach to 88% and 81% under OpenSCaaS at the same situation. Moreover, when the number of candidate function increases from 15 to 80, RaaS brings much higher increment in both satisfaction degrees (i.e., about 18% and 22% respectively) than OpenSCaaS (i.e., about 11% and 7% respectively). The reason is as follows. RaaS customizes routing services in a fine-grained way by leveraging the defined variation points of RSPL to select the classified and appropriate functions independently, which promotes satisfying different demands of users purposely and reducing composition costs. In RaaS, two evaluation models have been constructed. The user experience on service quality and service price are both considered to improve the user service experience, which is also used as a factor to optimize the ISP economic benefit when selecting network functions and pricing services. Thus, when the number of candidate function increases, RaaS can achieve more optimized function combinations, which improve the quality of the customized routing services and reduce the service provision costs. However, OpenSCaaS just considers satisfying user demands by establishing new service chains, which cannot distinguish the fine-grained differences among the user diversified demands. It focuses on the user experience on service quality without considering the impact of service price on the user satisfaction. Moreover, OpenSCaaS does not consider the issue of optimizing ISP benefits when providing services.

The user requirements on CRS may change when the CRS runs. Thus, we also compare the following two measures: CRS adjustment efficiency and CRS adjustment success ratio. The former is 1 minus the ratio of the CRS adjustment time to the CRS setup time, that is, the smaller the CRS adjustment time is, the higher the CRS adjustment efficiency is. The latter is the ratio of the number of the successfully adjusted CRSs to the total number of the CRSs of which requirements are changed. We randomly select 30% of the total CRSs and change their requirements, and the results are shown as Figs. 14 and 15.

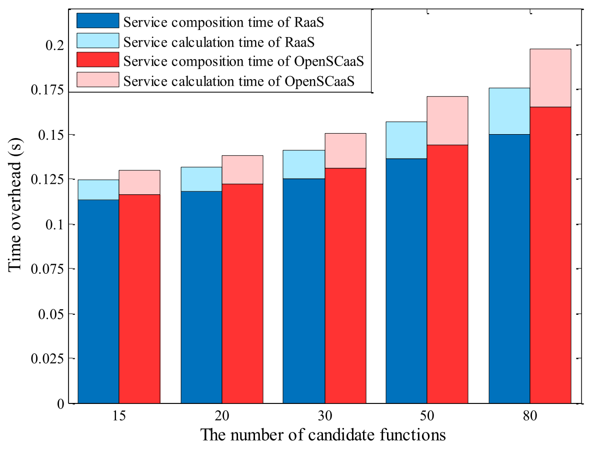
It can be seen that, CRS adjustment efficiencies and success ratios of the two approach decrease when the number of flows increases, and their values of RaaS are always higher than those of OpenSCaaS. In more detail, the CRS adjustment efficiency and the CRS adjustment success ratio of RaaS just decreases about 14% and 10% respectively when the number of flows increases by 1000 times, while their values decreases about 35% and 17% respectively under OpenSCaaS at the same situation. Moreover, under the peak network load, the CRS adjustment efficiency and the CRS adjustment success ratio of RaaS are about 30% and 8% higher those of OpenSCaaS. The reason is as follows. Although the number of flows which request routing services increases sharply, RaaS only needs to calculate the differences among the involved variation points according to the changed communication requirements based on



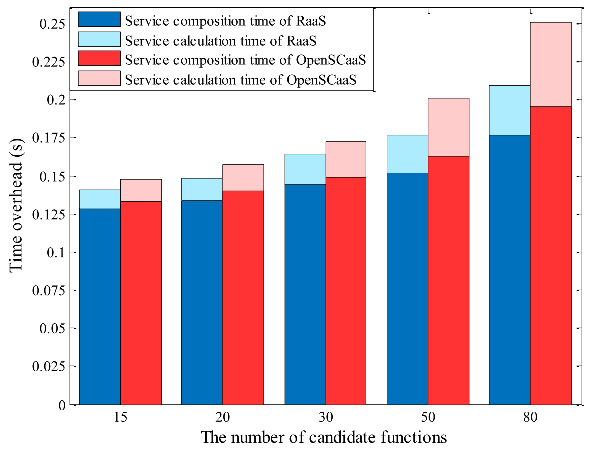
(a) 10^2 flows



(b) 10^3 flows

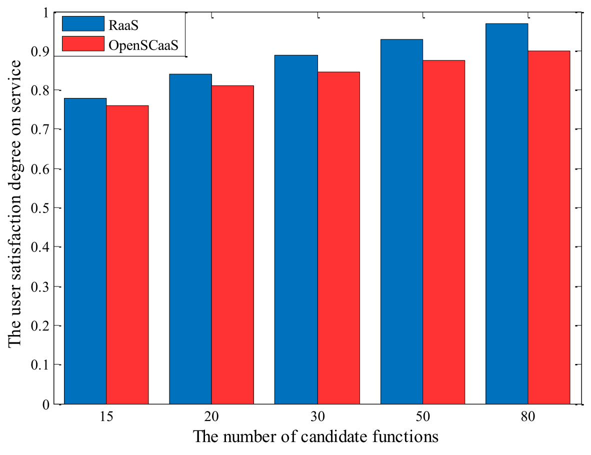


(c) 10^4 flows

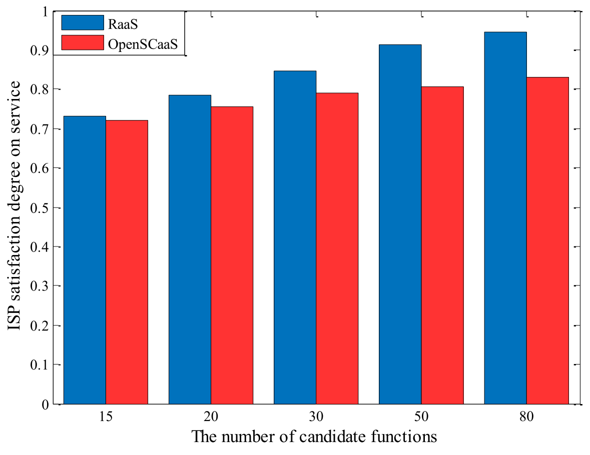


(d) 10^5 flows

Fig. 11. Time overhead over INTERNET2.



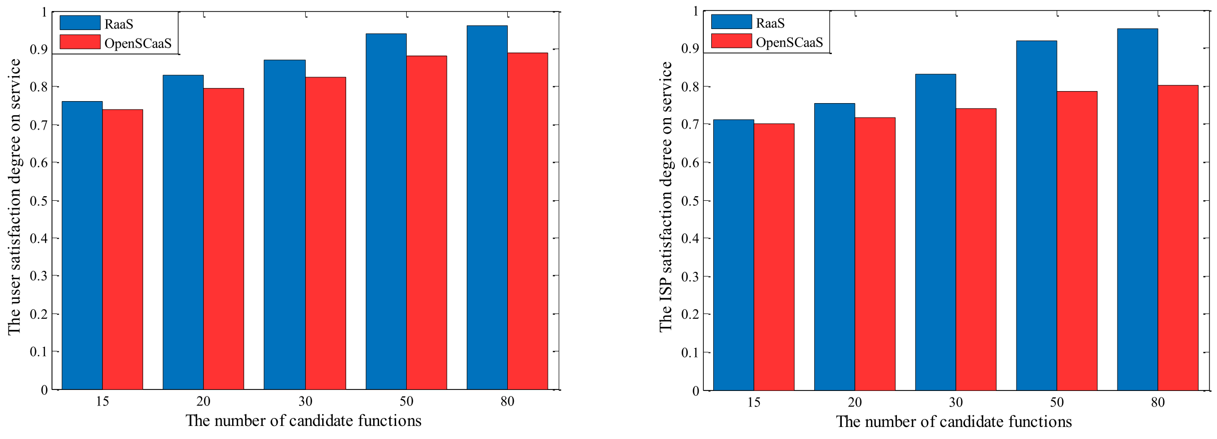
(a) The average user satisfaction degree over GéANT



(b) The average ISP satisfaction degree over GéANT

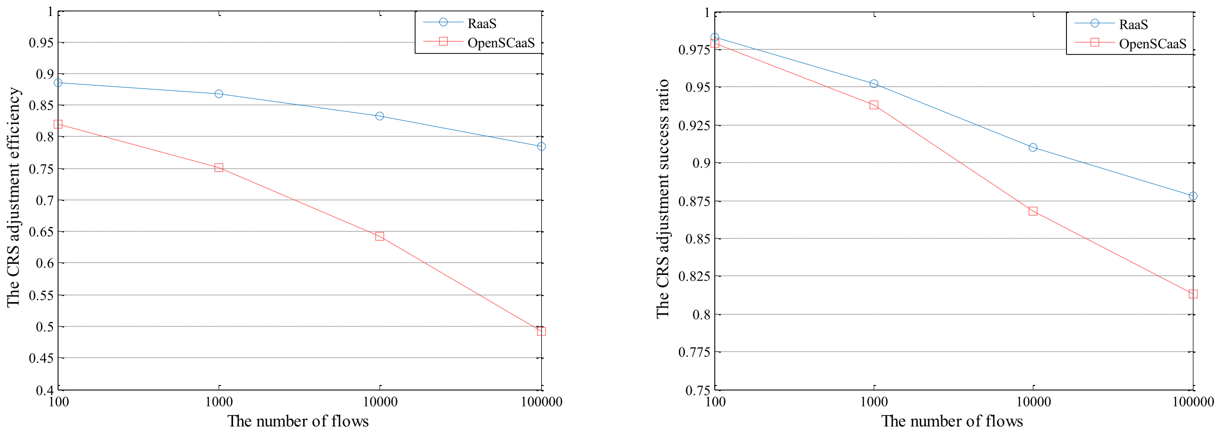
Fig. 12. The average satisfaction degrees for customized services over GéANT.

the RSPL. In the model establishing process (i.e., domain engineering process), the feasible function combinations for different demands have been considered, which can effectively reduce the impact of high network load on CRS adjustment efficiency. Furthermore, depending on the reusable, loosely coupling and composable characteristics of RSPL, RaaS only replaces the involved functions in the already customized routing services according to the specific variation points to adjust CRSs without establishing new CRSs from scratch. This decreases the CRS adjustment failure ratio under high network load. However, OpenSCaaS has to take much time to calculate the whole function configuration



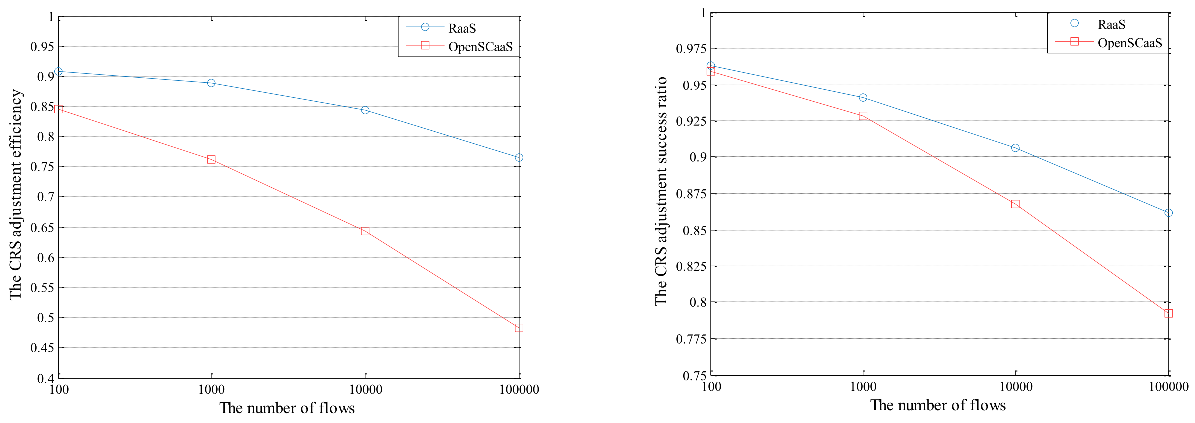
(a) The average user satisfaction degree over INTERNET2 (b) The average ISP satisfaction degree over INTERNET2

Fig. 13. The average satisfaction degrees for customized services over INTERNET2.



(a) The average CRS adjustment efficiency over GéANT (b) The average CRS adjustment success ratio over GéANT

Fig. 14. The average CRS adjustment efficiency over GéANT.



(a) The average CRS adjustment efficiency over INTERNET2 (b) The average CRS adjustment success ratio over INTERNET2

Fig. 15. The average CRS adjustment efficiency over INTERNET2.

according to the changed requirements and search for suitable functions from candidate functions one by one, which greatly reduce the CRS adjustment efficiency. In addition, OpenSCaaS also cannot reuse the running services, it has to re-compose new routing services to replace them from scratch once the running ones' demands being changed. This will significantly decrease the CRS adjustment success ratio especially when the available network resources are insufficient caused by the increasing network load.

7. Conclusion

In this paper, an open framework, that is, RaaS, for customizing routing service based on SDN and NFV, is proposed. We leverage the idea of DSPL to achieve routing service customization with requirements of both the user and the ISP considered. The two-layered feature model for RSPL is proposed and used to compose the personalized routing services with the diverse network functions. The orthogonal variability model for RSPL is proposed and used to achieve traceability and consistency of the variability in different phase of the development models. We formulate the formal definitions and descriptions to abstract routing service customization, and devise two evaluation models for USE and IEB to optimize benefits of both the user and the ISP.

In order to improve the practicability of our work, we plan to do prototype implementation of the proposed RaaS over the production networks to further verify its effectiveness and improve its performance. In addition, some more efficient network function selection schemes will be investigated in the near future.

Acknowledgments

This work is supported by the National Key R&D Program of China under Grant No. 2017YFB0801701, and the Major International (Regional) Joint Research Project of NSFC under Grant No. 71620107003.

References

Antonio, G.I., Juan, C.C., Antoni, B., Angélica, R., 2011. A QoS-based dynamic pricing approach for services provisioning in heterogeneous wireless access networks. Pervasive Mob. Comput. 7 (5), 569–583.
Bencomo, N., Hallsteinsen, S., Almeida, E.S., 2012. A view of the dynamic software product line landscape. Computer 45 (10), 36–41.
Bentaleb, A., Begen, A.C., Zimmermann, R., Harous, S., 2017. SDNHAS: an SDN-enabled architecture to optimize QoE in HTTP adaptive streaming. IEEE Trans. Multimed. 19 (10), 2136–2151.
Bu, C., Wang, X., Cheng, H., Huang, M., Li, K., Sajal, K.D., 2017. Enabling adaptive routing service customization via the integration of SDN and NFV. J. Netw. Comput. Appl. 93 (1), 123–136.
Bu, C., Wang, X., Huang, M., Li, K., 2018. SDNFV-based dynamic network function deployment: model and mechanism. IEEE Commun. Lett. 22 (1), 93–96.
Bueno, I., Aznar, J.I., Escalona, E., Ferrer, J., García-Espín, J.A., 2013. An OpenNaaS based SDN framework for dynamic QoS control. In: Proceedings of the IEEE SDN for Future Networks and Services, pp. 1–7.
Carpenter, B., 2002. Middleboxes: taxonomy and issues. RFC 3234.
Chen, Y., Wang, L., Lin, F., Lin, B., 2017. Deterministic quality of service guarantee for dynamic service chaining in software defined networking. IEEE Trans. Netw. Serv. Manag. 14 (4), 991–1002.
Cheng, G., Chen, H., Hu, H., Wang, Z., Lan, J., 2015. Enabling network function combination via service chain instantiation. Comput. Network. 92, 396–407.
Definitions of terms Related to Quality of Service, 2008. ITU-T E. 800.
Diego, K., Ramos, F.M.V., Paulo, V., Christian, E.R., Siamak, A., Steve, U., 2015. Software-defined networking: a comprehensive survey. Proc. IEEE 103 (1), 14–76.
Floris, A., Ahmad, A., Atzori, L., 2018. QoE-aware OTT-ISP collaboration in service management: architecture and approaches. ACM Trans. Multimed Comput. Commun. Appl 14 (2s), 36:1-36:24.
Gamez, N., Haddad, J.E., Fuentes, L., 2015. SPL-TQSSS: a software product line approach for stateful service selection. In: Proceedings of the International Conference on Web Services, pp. 73–80.
Gharbaoui, M., Fichera, S., Castoldi, P., Martini, B., 2017. Network orchestrator for QoS-enabled service function chaining in reliable NFV/SDN infrastructure. In: Proceedings of the 3rd Conference on Network Softwarization, pp. 1–5.
Gomaa, H., Hashimoto, K., 2011. Dynamic software adaptation for service-oriented product lines. In: Proceedings of the 15th International Software Product Line Conference, pp. 21–26.
Grigoriou, E., Barakabitze, A.A., Atzori, L., Sun, L., Pilloni, V., 2017. An SDN-approach for QoE management of multimedia services using resource allocation. In: Proceedings of the International Conference on Communications.
Hallsteinsen, S., Hinchey, M., Park, S., Schmid, K., 2013. Dynamic Software Product Lines. Systems and Software Variability Management, pp. 253–260.
Hinchey, M., Park, S., Schmid, Klaus, 2012. Building dynamic software product lines. Computer 45 (10), 211–220.
Hsu, W.H., Lo, C.H., 2014. QoS/QoE mapping and adjustment model in the cloud-based multimedia infrastructure. IEEE Syst. J. 8 (1), 247–255.
INTERNET2 Network Infrastructure Topology, URL≤http://www.internet2.
Jiang, Y., Lan, J., Wang, Z., Deng, Y., 2016. Embedding and reconfiguration algorithms for service aggregation in network virtualization. Int. J. Commun. Syst. 29 (1), 33–46.
Kamoun, A., Kacem, M.H., Kacem, A.H., 2016. Multiple software product lines for service oriented architecture. In: Proceedings of the 25th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, pp. 56–61.
Khan, M.M.A., Cheriton, D.R., Shahriar, N., Ahmed, R., Boutaba, R., 2016. Multi-path link embedding for survivability in virtual networks. IEEE Trans. Netw. Serv. Manag. 13 (2), 253–266.
Kohler, E., Morris, R., Chen, B., Jannotti, J., Kaashoek, M.F., 2000. The Click modular router. ACM Trans. Comput. Syst. 18 (3), 263–297.
Kosugiyama, T., Tanabe, K., Nakayama, H., Hayashi, T., Yamaoka, K., 2017. A flow aggregation method based on end-to-end delay in SDN. In: Proceedings of the International Conference on Communications, pp. 1–6.
Kotonya, G., Lee, J., Robinson, D., 2009. A consumer-centred approach for service-oriented product line development. In: Proceedings of the Joint Working IEEE/IFIP Conference on Software Architecture and European Conference on Software Architecture, pp. 211–220.
Lee, K., Kang, K.C., Lee, J., 2002. Concepts and guidelines of feature modeling for product line software engineering. In: Processings of the Proceedings of the 7th International Conference on Software Reuse: Methods, Techniques, and Tools, pp. 62–77.
Lee, J., Kotonya, G., Robinson, D., 2012. Engineering service-based dynamic software product lines. Computer 45 (10), 49–55.
Li, W., Qi, H., Li, K., Stojmenovic, I., Lan, J., 2017. Joint optimization of bandwidth for provider and delay for user in software defined data centers. IEEE Trans. Cloud Comput. 5 (2), 331–343.
Lima, S., Carvalho, P., 2011. Enabling self-adaptive QoE/QoS control. In: Proceedings of the 36th Conference on Local Computer Networks, pp. 239–242.
Lin, S.C., Wang, P., Luo, M., 2016. Jointly optimized QoS-aware virtualization and routing in software defined networks. Comput. Network. 96 (26), 69–78.
Lopes, F.A., Santos, M., Fidalgo, R., Fernandes, S., 2016. A software engineering perspective on SDN programmability. IEEE Commun. Surv. Tutor. 18 (2), 1255–1272.
Lorena, I., Ángel, L., Luis, J., 2015. Trends on virtualisation with software defined networking and network function virtualization. IET Netw. 4 (5), 255–263.
Mijumbi, R., Serrat, J., Gorricho, J., Bouten, N., Turck, F.D., Boutaba, R., 2015. Network function virtualization: state-of-the-art and research challenges. IEEE Commun. Surv. Tutor. 18 (1), 236–262.
Moyano, R.F., Fernández, D., Bellido, L., Merayo, N., Aguado, J.C., Miguel, I., 2017. NFV-based QoS provision for software defined optical access and residential networks. In: Proceedings of the 25th International Symposium on Quality of Service, pp. 1–5.
Msakni, H.G., Youssef, H., 2013. Is QoE estimation based on QoS parameters sufficient for video quality assessment?. In: Proceeding of the 9th International Wireless Communications and Mobile Computing Conference, pp. 538–544.
Mundada, Y., Sherwood, R., Feamster, N., 2009. An OpenFlow switch element for Click. In: Proceedings of the Symposium on Click Modular Router.
Nascimento, A.S., Rubira, C.M.F., Castor, F., 2014. ArCMAPE: a software product line infrastructure to support fault-tolerant composite services. In: Proceedings of the 15th International Symposium on High-assurance Systems Engineering, pp. 41–48.
Network performance Objectives for IP-based Services, 2011. ITU-T Y. 1541.
OpenFlowClick, URL http://archive.openflow.org/wk/index.php/OpenFlowClick.
Project Floodlight Open Source Software for Building Software-Defined Networks, URL ≤http://www.projectfloodlight.org/.
Sahhaf, S., Tavernier, W., Rost, M., Schmid, S., Colle, D., Pickavet, M., Demeester, P., 2015. Network service chaining with optimized network function embedding supporting service decompositions. Comput. Network. 93 (3), 492–505.
Tarnaras, G., Haleplidis, E., Denazis, S., 2015. SDN and ForCES based optimal network topology discovery. In: Proceedings of the IEEE Conference on Network Softwarization, pp. 1–6.
The Internet Topology Zoo, URL <Available: http://www.topology-zoo.org/.
Wagle, S.S., Guzek, M., Bouvry, P., Bisdorff, R., 2015. An evaluation model for selecting cloud services from commercially available cloud providers. In: Proceeding of the 7th International Conference on Cloud Computing Technology and Science, pp. 107–114.
Wang, P., Lan, J., Zhang, X., Hu, Y., Chen, S., 2015. Dynamic function composition for network service chain: model and optimization. Comput. Network. 92, 408–418.
Wu, J., Zhang, Z., Hong, Y., Wen, Y., 2015. Cloud radio access network (C-RAN): a primer. IEEE Netw. 29 (1), 35–41.
Yu, T., Wang, K., Hsu, Y., 2015. Adaptive routing for video streaming with QoS support over SDN networks. In: Proceedings of the International Conference on Information Networking, pp. 318–323.

Chao Bu received the B.S. degree in information security, the M.S., and Ph.D. degrees in software engineering from the Northeastern University, Shenyang, China, in 2010, 2012, and 2018 respectively. He is currently a Lecturer at the School of Computer Science and Engineering, Tianjin University of Technology. His research interests include service chain customization and future Internet, etc.

Xingwei Wang received the B.S., M.S., and Ph.D. degrees in computer science from the Northeastern University, Shenyang, China, in 1989, 1992, and 1998 respectively. He is currently a Professor at the School of Computer Science and Engineering, Northeastern University. His research interests include cloud computing and future Internet, etc.

Min Huang received the B.S. degree in automatic instrument, the M.S. degree in systems engineering, and the Ph.D. degree in control theory from the Northeastern University, Shenyang, China, in 1990, 1993, and 1999 respectively. She is currently a Professor at the College of Information Science and Engineering, Northeastern University. Her research interests include modeling and optimization for logistics and supply chain system, etc.

Hui Cheng received the B.S. and M.S. degrees in Computer Science from the Northeastern University, Shenyang, China, in 2001 and 2004 respectively, and the Ph.D. degree in Computer Science from the Hong Kong Polytechnic University, Hong Kong, China, in 2007. He is currently a Senior Lecturer at the School of Computing & Mathematical Sciences, Liverpool John Moores University, Liverpool, UK. His research interests include wireless mobile networks, optical networks, artificial intelligence, and dynamic optimization, etc.

Keqin Li received the B.S. degree in computer science from Tsinghua University, Beijing, China, in 1985, and the Ph.D. degree in computer science from the University of Houston, Texas, USA, in 1990. He is currently a SUNY distinguished professor of computer science in State University of New York at New Paltz. His research interests include parallel and distributed computing and computer networking, etc.