A UNIFIED CONTROL AND OPTIMIZATION FRAMEWORK FOR DYNAMICAL SERVICE CHAINING IN SOFTWARE-DEFINED NFV SYSTEM

YONG LI, FENG ZHENG, MIN CHEN, AND DEPENG JIN

ABSTRACT

In software-defined radio and cognitive radio enabled future 5G mobile networks, and SDN-NFV system will enable high flexibility in constructing service chaining, that is, steering flows through required service chains, thanks to the ability of dynamic service deployment and agile traffic routing. Given such flexibility, we need to carefully design the system to identify the optimal mechanism that maximizes both performance and resource utilization. To enable the SDN-NFV system to optimize the service chaining according to the user requirements and network environment, in this article, we propose a unified control and optimization framework that jointly controls and optimizes the resource allocation in networking and service provisioning. By modeling the network and services together and developing optimization techniques that jointly examine both, this framework benefits the overall system performance in various scenarios of traffic steering, VM selection, function assignment, and policy optimization. Furthermore, these achievable benefits are demonstrated by a typical example of simulation under a realistic scenario.

INTRODUCTION

With the rapid growth of wireless traffic demands, the current wireless technology will be overcome in the next 10 years. To struggle toward the high network capacity demands provoked by the explosion of current services, potential technologies of software-defined radio, cognitive radio, and networks for reconfigurable networks have been investigated for future 5G systems [1]. These involve the investigation of how to apply spectrally efficient technologies on brand-new network architecture. Current and future mobile network operators rely on network functions (NFs), or inline services in their networks, such as deep packet inspection (DPI), logging/metering/charging/advanced charging, firewall, proxy, intrusion detection and prevention (IDP), Network Address Translation (NAT), and so on, to manage users' mobile traffic, increase security, and improve performance. A service chain is defined as a sequence of NFs that should be traversed by given flows in a predefined order, which is required when the traffic needs to go through more than one NF. Moreover, if there are multiple service chains, the operator needs to configure the networking infrastructure to steer the traffic through the right service chain. Traditionally, NFs are deployed as dedicated middleboxes, which is referred as hardware-based NFs in the rest of the article. Hardware-based NF deployments are typically overprovisioned, or otherwise have to drop packets and disable certain functionalities in the face of heavy load. For example, an IDS may disable DPI capabilities under heavy load. Meanwhile, operators usually rely on error-prone and complex low-level manual configurations to coerce traffic through the desired set of NFs. Both NF deployment and policy enforcement require significant capital and operating investment.

To address the high cost issue, recently, there has been growing interest in replacing purpose-built hardware NF solutions with software-based NFs running on commodity devices, known as network functions virtualization (NFV) [2]. By decoupling the NFs from proprietary hardware appliances, NFV provides flexible provisioning of software-based network functionalities on top of an optimally shared physical infrastructure. On the other hand, software-defined networking (SDN) is being used to steer flows delicately through appropriate NFs to enforce policies and jointly manage network and NF loads [3]. It allows for new flexibility and centralized intelligence in traffic steering. Thus, integrating NFV and SDN enables unified control of NF deployment and traffic steering, which brings new opportunities to performance optimization.

Together, NFV and SDN can help achieve:
- Policy-aware NF deployment, which optimizes user utility and guarantees NF performance
- Accurate and efficient traffic steering to minimize the operating expense

However, to simultaneously achieve these goals, we need to jointly optimize and control the NFV orchestration and switching behavior. Cur-

Yong Li, Feng Zheng, and Depeng Jin are with Tsinghua University.

Min Chen is with Huazhong University of Science and Technology.

1536-1284/15/$25.00 © 2015 IEEE

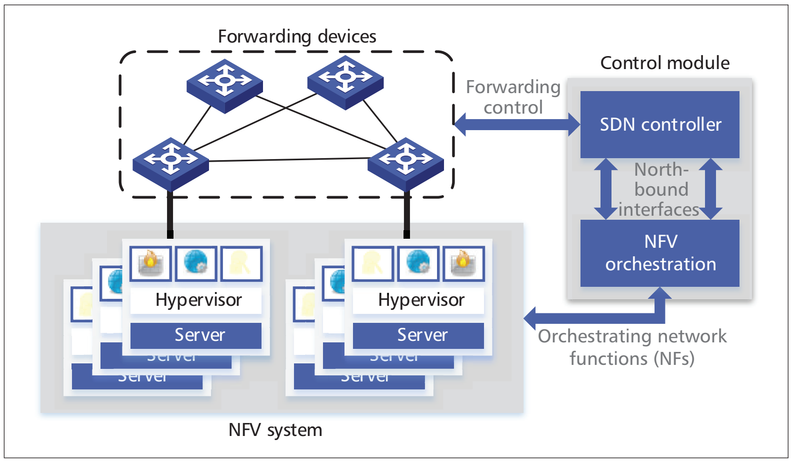

Figure 1. Software-defined NFV system overview.

rent research mainly focuses on the innovation of NFV platforms [4, 5] and traffic steering for non-virtualized inline services [6–8]. No existing mechanism in current SDN-NFV systems tackles the optimization and unified control of NFs and forwarding devices.

In this article, we establish a novel framework that jointly optimizes and controls the underlying devices to enable the SDN-NFV system to achieve aforementioned goals. This framework enables the joint optimization of NF provisioning and traffic steering by taking full advantage of the flexibility offered by the NFV system and enforces the optimization results with the unified control mechanism, which guarantees accuracy and efficiency of the NF services and traffic forwarding. The optimization framework models the network topologies and user requirements, which can be utilized extensively in problems associated with NF deployment and policy enforcement. In parallel, the unified control framework leverages standardized protocols and interfaces to conduct centralized control of traffic forwarding, NF assignment, and provisioning in order to enforce the optimization results and handle dynamic events (congestions, failures, etc.). Furthermore, this study opens up a new research direction for the design of SDN-NFV systems through a quantitative analysis based on a specific scenario.

We structure the article as follows. After giving the system overview in the next section, we introduce the unified control framework including the details of the newly designed interfaces. Then we introduce the optimization framework and several possible optimization objectives. After analyzing the capability of the proposed control and optimization framework, we introduce an example application and quantitatively assess the capability of the control and optimization framework by simulation targeting at realistic network environment and user requirements. Finally, we conclude the article.

System Overview

The proposed SDN-NFV system is illustrated in Fig. 1. It consists of a control module, forwarding devices, and an NFV system at the edge of the network. The rules of packet forwarding are determined by the SDN controller and implemented in the forwarding devices through forwarding tables. Efficient protocols (e.g., OpenFlow [3]) are used as standard interfaces in communications between the controller and forwarding devices. The NFV system leverages commodity servers to implement high bandwidth NFs at low cost. Hypervisors run on the servers to support the virtual machines (VMs) that implement the NFs as pieces of pure software. This system allows customizable data plane processing capabilities, such as firewalls, proxies, and IDSs, to be embedded within VMs, where an efficient virtualized platform working at line rate has been designed specifically for implementing inline services [4, 5].

The SDN controller and NFV orchestration system compose the control module. The NFV orchestration system is in charge of assigning and provisioning the virtualized NFs [9]. It is controlled by the SDN controller through standard interfaces. Both the NFV system and forwarding devices are monitored continuously by the control module. The forwarding devices export topology information (e.g., links between the devices, link capacities, and VM locations) and resource information (e.g., CPU and amount of TCAM) to the control module. In parallel, the NF states and packet processing capacities are collected by the NF orchestration system. To achieve optimal resource deployment and traffic routing, the control module takes as input the aforementioned information and the policy requirements, then runs the optimization required by operators and enforces the results with unified control of both the NFV system and the forwarding devices. On the other hand, to ensure high performance and availability, the control module also takes care of dynamic events such as congestion, load changes, and failures by reprovisioning a certain set of NFs.

For example, a group of operators demand that their traffic flows arrive at their destinations within the minimum hop counts. The control module takes their policy specifications and monitored topology and resource information as input, and jointly computes the optimal NF assignment and provisioning together with the routing paths of policies, thus achieving the minimum overall hop count. During data transmission, if severe congestion or a failure occurs at an NF, the NF instance is reprovisioned elsewhere (the placement is also optimized) by the control module, with its internal states migrated and the associated forwarding tables updated. The SDN-NFV system offers significant flexibility in both NF deployment and traffic steering. Moreover, some NFs can even be incorporated into this system. For example, once load balancing is virtualized, SDN can make it part of the path selection algorithm. That is, load balancing can be an augmentation to route computation, which itself is a control plane software process run on the control module.

The Unified Control Framework

A commodity server in the NFV system is able to support multiple functions simultaneously with multiple VMs running in it, which creates new opportunities to optimize the policy enforcement. On one hand, by applying specific functions to different traffic flows in one server, it is able to reduce the end-to-end cost and overall bandwidth consumption. On the other hand, by dynamically adjusting the functions running on each server, it is able to utilize the resources more efficiently over time. It is oblivious that achieving the above benefits requires flexible and unified control of NFs and forwarding devices, based on which the network operator can jointly control the deployment of the NFs and the forwarding devices. Meanwhile, to achieve fast recovery of possible congestion and failures of both NF software and hardware by dynamic provisioning of NFs, special care should be taken for the software-based NFs in both monitoring and control.

The proposed framework aims to conduct unified control of the forwarding devices and NFs for optimal policy enforcement and high performance and availability. As shown in Fig. 2, this framework includes a logically centralized control module, the forwarding devices, and an NFV system. The unified control system relies on measurement flows to collected information for the optimization framework and dynamic handler, and leverages the control flows to control underlying hardware and software. Now, we introduce the design details of these two flows.

Measurement Flow

Topology, resource, and NF state information is collected and reported by the NF and forwarding controllers as internal input, while the policies are generated from the operators' requirements as external input. The optimization framework and dynamic handler take this information as input to solve optimization problems and tackle dynamic events. Details of the measurement flows are illustrated as follows.

User Policy: The user policies are specified by the network operators without worrying about which NF to traverse and how the traffic is routed. The operator specifies the policy classes and the sequence of network service needed by each class. For example, three policies are required by the operators: IDS-proxy-firewall, IDS-proxy, and firewall-IDS, where IDS-proxy-firewall represents a policy requiring its traffic to traverse an IDS, a proxy, and a firewall sequentially. At the same time, the traffic volume traversing by each policy is also reported to the control module.

Topology Information: The SDN controller has information on the network topology including links between the forwarding devices and locations of the VMs. The network topology is typically reported by the forwarding devices, and the VM locations are provided by the NFV system. The link capacities are collected by the network management systems. All VMs are attached to the edge of the network, and one forwarding device can connect to several VMs, each of which implements one NF.

Resource Information: The metrics of link bandwidths and packet processing capacities are collected by the controller. As the VMs are highly homogeneous, the traffic processing rates varies with network functions instead of VMs. For example, a VM can process c_F traffic rate at peak when implementing a firewall, and c_I traffic rate while implementing an IDS.

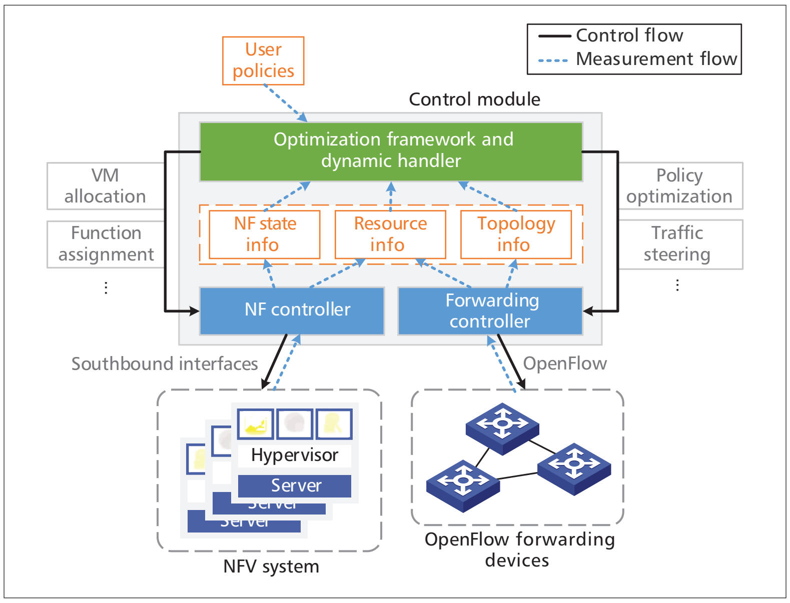

Figure 2. The unified control framework, in which the information is gathered by measurement flows, and control operations are implemented by the control flows.

NF State Information: When new NF instances are launched and traffic rerouted, the accuracy of the NF might be affected since the internal states are not migrated from the source NF [10]. The only way to avoid the trade-off between NF accuracy and performance is to allow the control module to quickly and safely move the internal state from the original instance to the new instance. Thus, the NF states are reported to the control module in the system. Updates of the states are triggered by certain events during packet processing.

Control Flow

The NFV system and forwarding devices are controlled by the control flows. We assume all forwarding devices support OpenFlow abstraction, and the forwarding controller uses the standard OpenFlow protocol to control them. For NFs, the NF controller encapsulates the complexities of distributed function and state control and, when requested, guarantees loss-free and order-preserving state operations. To achieve this, two types of southbound interfaces are designed, allowing the NF controller to closely control the behavior and performance of NFs to satisfy high-level objectives. Now, we illustrate the details of these two interfaces.

Function Provisioning Interfaces: The FPIs control whether a VM is launched and which NF it should implement. These interfaces are invoked when new NF instances are provisioned or old ones de-provisioned. Three operations are provided: *turnon*, *turnoff*, and *assign*. The NF manager interacts with the hypervisors to turn on or turn off a certain VM via the former two operations and assigns functions with the latter. For example, *assign*<*vmid, funcid*> assigns the function with function id *funcid* to the VM with the id *vmid*.

Function State Interfaces: When a congestion or failure occurs in an NF or an updated NF software is applied in the NFV system, new

The optimization framework works via control applications, that is, VM allocation, function assignment, policy optimization, and traffic steering, which convert the optimization results into lower-level operations that can be applied by the NF controller and forwarding controller. At the same time, the dynamic handler tackles congestion or failure in a similar fashion.

NF instances need to be launched, and traffic is rerouted. NFV allows us to launch a new instance in a matter of milliseconds[4], and SDN allows us to reroute traffic to that instance just as quickly. However, this simple rerouting of traffic can compromise NF accuracy due to the absence of internal NF state at the new instance. Function state interfaces (FSIs) are designed to move, copy or share, and combine NF state in a loss-free and order-preserving fashion to guarantee NF accuracy and overall performance when a certain set of NFs are re-provisioned elsewhere. To minimize NF modifications, state gathering and merging is delegated to NFs which perform these tasks within the context of their existing internal architecture. We provide three simple operations for NF states: *get*, *put*, and *delete*. Each group of states are associated with a certain set of flows; thus, each operation is an atomic one that only *put* or *delete* a delegated group of states within an NF. Complex operations such as migrating all internal states from one NF to another are translated into a set of atomic operations by a control application, and enforced by the NF manager and forwarding controller. An FSI also includes interfaces for observing and preventing state updates since there are times we need to know whether updates are ongoing (e.g., to decide when to move state) or to prevent an NF from updating its states (e.g., while state is being copied). Moreover, in case of NF failures, the NF manager needs to *get* the states if updates are observed to maintain an up-to-date copy of all NF states.

The optimization framework works via control applications, that is, VM allocation, function assignment, policy optimization, and traffic steering, which convert the optimization results into lower-level operations that can be applied by the NF controller and forwarding controller. At the same time, the dynamic handler tackles congestion or failure in a similar fashion.

Optimization Framework

An SDN-NFV system allows new flexibility in NF deployment and traffic routing. Our proposed optimization framework takes full advantage of this flexibility and jointly optimizes various variables in such a system. Assume the network with the topology of $G = <V,E>$, which comprises a set of nodes V (forwarding devices), and the nodes are interconnected by a set of links E. Each node may connect to several commodity servers running VMs. We regard the VMs as distinct entities and let $Q = \{q_1, q_2, \ldots, q_m\}$ denote the set of VMs. The function set is labeled as $F = \{f_1, f_2, \ldots, f_n\}$. Each VM $q \in Q$ can implement any desired function $f \in F$. Let P_k denote one of the K service sequences required by the operators, which is represented with a function sequence $<i_k, s_{k,1}, s_{k,2}, \ldots, s_{k,nk}, d_k>$, with $i_k, d_k \in V$ denoting the ingress and egress node of P_k, $s_{k,i} \in F$ representing the *ith* NF the traffic of policy class k should traverse. The traffic volume of P_k is labeled as T_k, also reported to the control module by the operators.

As shown in Fig. 2, the policies, topology information, and resource information are the inputs needed by the optimization framework. With this information, the SDN-NFV system can be optimized in several aspects:

- On-demand resource allocation, that is, we can launch only a subset of Q to minimize the energy cost and meanwhile guarantee low congestion and overload possibility.
- Policy-aware function assignment. As each VM can implement any delegated NF, NFs can be dynamically assigned to the VMs in an optimal way in order to achieve better performance, aware of the user policies.
- Smarter traffic engineering. Once the functions are assigned, traffic of particular flows should be steered to pass an appropriate sequence of NFs to enforce the policy specifications. Our proposed optimization framework is able to compute the optimal routing paths according to the policies.
- Policy optimization. Usually, the operators only require flows to bypass a certain set of NFs without regard to the order. Thus, the NF sequences can be optimized by adjusting the order of NFs for various objectives. Our proposed optimization framework covers most flexibility of the SDN-NFV system. Once an optimization objective is set, jointly optimizing the aforementioned variables can achieve the optimal performance the SDN-NFV system can offer. The main components of the optimization framework are summarized in Table 1.

Optimization Objectives

To meet various demands of operators, distinct objectives are of concern in the proposed optimization framework to optimize performance in different aspects. This framework can benefit operators by minimizing the cost of data transmission while benefiting Internet service providers (ISPs) by minimizing the capacity required by peak usage to cut capital expenses.

Hops: Services should be deployed to ensure that traffic is routed to the destination within the minimum hop counts. However, it is not possible to minimize the hop counts for every policy or operator. One possible solution is to minimize the overall hop count for all operators. Given a certain group of operators, the total hop count is the summation of the hop counts of all policies. For each service sequence (e.g., P_k) the length of its route path consists of three parts: the distance from the ingress node to the first NF, the total length of the paths between the NFs, and the distance from the last NF to the egress node. For each pair of NFs (f_a, f_b), the hop count between them can be denoted as $h(f_a,\ f_b)$. Then the overall hop count of K policies can be expressed as

$$\sum_{k=1}^{K}\left(h(i_k, s_{k,1}) + \sum_{i=1}^{n_k-1} h(s_{k,i}, s_{k,i+1}) + h(s_{k,n_k}, d_k)\right),$$

where the ingress and egress nodes can be regarded as a wire NF that simply sends packets from its input to its output.

Delay: The traffic of network service is highly sensitive to delay [11]. An example is when one data center (DC) contacts another in the process of responding to a user request because not all information is available in the first DC. Such interactive traffic is highly sensitive to delay, even small increase in response time (100 ms) degrades user experience [14]. For this type of traffic, the accumulated delay of the service chain through which the flow is required to pass must be minimized. There are two types of metrics to define minimum here. We may minimize the average delay for all operators or minimize the maximum delay to bound the worst user experience. The calculation of the delay of a policy is similar to that of the hop count. The hop count $h(f_a, f_b)$ between the NF pair (f_a, f_b) is replaced by the delay $d(f_a, f_b)$ between them.

NF Load Balance: An efficient virtualized platform working at line rate has been developed for high-performance network services [4, 5]. The investment in network services has been dramatically decreased by such platforms running on commodity hardware. However, as the number of NFs required in current networks becomes large, the capital cost is still an important issue for ISPs. Lower NF loads at peak can significantly save money for ISPs since cheaper devices can be applied. On the other hand, bounding the NF loads decreases the possibility of congestion and failure. In our framework, NF load balance can be achieved by minimizing the maximum NF load. The load on each NF is modeled in terms of the total volume of traffic across all service sequences of which it is a part.

Link Utilization: Network operators rely on high link bandwidth, which is an expensive resource, to improve the user experience and reliability. However, the network cycles through periods of peaks and troughs, making this investment not fully leveraged due to the lack of coordination among the services that use the network. Since network resources must be provisioned for peak usage to avoid congestion, the network is under-subscribed on average. With a global view of the network resources, our proposed optimization framework can jointly consider policy specifications and resource constraints. By carefully assigning NFs and routing the traffic flows, we are able to globally allocate traffic flows to balance the link utilization. Here, the network bandwidth can be allocated in a max-min fair manner to maximize the minimum utilization of links. For each link $e \in E$, its utilization is calculated by

$$\sum_{k=1}^{K} z(k)T_k/b(e),$$

where $z(k)$ denotes how many times policy k passes e and $b(e)$ represents the bandwidth of link e.

Energy Saving: With our proposed optimization framework, operators can reduce energy costs by minimizing the number of NFs needed and constructing active paths that exploit excess capacity. According to the traffic volumes and NF sequences of the policies, we can compute the lower bound of required NF number N_r and where they are implemented, and meanwhile ensure low possibility of congestion and failure. When new requirements are specified, we can accommodate the newly arrived flows with currently active devices if there is enough excess capacity. If not, the proposed framework can still be utilized to minimize the number of newly launched devices.

System Constraints

In order to meet the nature requirements of the system and ensure that the resource limits are not violated, serial system constraints should be considered, which include the constraints of NF implementation and system resources.

Objectives	Constraints	Capability
Hops: minimize $\sum_{k=1}^{K} h(P_k)$	NF implementation	Traffic steering
Delay: minimize $\sum_{k=1}^{K} d(P_k)$		VM selection
NF load balance: mininize $Max\{load(Q)\}$	System resources:	Function assignment
Link utilization: maximize $Min\left\{\sum_{k=1}^{K} z(k)T_k/b(e)\right\}$	TCAM, link/NF bandwidth	Policy optimization
Energy saving: minimize N_r		

Table 1. Summary of the optimization framework.

NF Implementation: Several basic constraints of the NFV system should be met. On one hand, current platforms specially developed for network services only support one NF in a VM. Thus, a VM should implement no more than one NF that is required by the NFV system. In parallel, all required functions should be implemented so that the policies can be successfully enforced. That is, for each $f \in F$, there should be at least one VM $q \in Q$ that implements f. These constraints are simple but fundamental to our problems.

System Resources: As stated before, we also need to consider the limited system resources, including computation and bandwidth resources, for example, the packet processing capacities of the VMs and the amount of TCAM available for installing forwarding rules in the forwarding devices. At the same time, it is also necessary to ensure that the link bandwidths are not violated.

Solution

For each objective introduced above, we can combine it with the constraints to form optimization problems for given systems and policies with the decision variables representing NF deployment and traffic routing. The objectives are mainly linear compositions of optimization variables, while the constraints of NF implementation and system resources are also linear constraints. Thus, the formulated optimization problems mentioned above fall into the category of linear programming problems, which can be solved by linear programming techniques, that is, using existing optimization toolkits, such as CPLEX and YALMIP.

On the other hand, since the SDN-NFV system is highly flexible in both service deployment and traffic steering, jointly optimizing several sets of variables, which can obtain better overall performance, becomes possible. However, joint optimization problems are usually nonlinear and thus NF-hard, making them inefficient to solve in real scenarios. To address this, heuristic solutions should be designed to solve these problems. For

In SDN-NFV systems, with the global view of the system and flexible control of the underlying devices enabled by our proposed unified control framework, it is possible to provision VMs dynamically through the southbound interfaces according to user requirements to save energy

example, we can decouple service deployment and traffic steering and solve them separately with heuristic approaches.

Capability of the Unified Control and Optimization Framework

Many technical problems can be tackled with the proposed unified control and optimization framework. The optimization framework models and solves various problems in terms of the optimization variables, while the unified control framework enforces the optimization results through southbound interfaces. In this section, we briefly analyze how we can model the problems and ultimately implement the solutions in the proposed optimization and control framework by considering traffic steering, VM selection, NF assignment, and policy optimization.

Traffic Steering

SDN offers new agility in traffic steering. Our control framework allows network operators to specify a logical routing policy and automatically translates this into forwarding rules. Prior to this, the routing paths are carefully selected by the optimization framework taking into account the physical topology, link capacities, and network resource constraints. Solid work has been done on traffic steering in hardware-based middlebox systems [6, 7]. However, in our proposed framework, traffic steering is jointly optimized with NF deployment, which can achieve better composition. To ensure that the transmission delay is bounded, for each policy, we steer traffic through a unique NF instance for each functionality instead of distributing the traffic over all instances. By defining binary variables denoting whether a VM is traversed by a policy, the traffic steering problem can be modeled as jointly solved by our proposed framework. The control of traffic steering is done by the forwarding controller via OpenFlow forwarding tables, which has already been thoroughly studied.

VM Selection

In traditional purpose-built systems, network functions are usually overprovisioned since they are provisioned for peak usage forecast by the network providers. In SDN-NFV systems, with the global view of the system and flexible control of the underlying devices enabled by our proposed unified control framework, it is possible to provision VMs dynamically through the southbound interfaces according to user requirements to save energy. The service sequences and corresponding traffic volumes are reported to the control module by the operators. Together with VM capabilities and other resource constraints, which are collected by the measurement flows, the optimization framework can select the minimum subset of Q to fulfill the demands. To model the VM selection problem in our proposed optimization framework, we first define binary indicator variables u_i denoting whether $q_i \in Q$ is turned on. The traffic volume bypassing each selected VM and relevent link utilizations are added as constraints. By solving the optimization problem, the proposed framework allows us to obtain the optimal selection of VMs. Once the VMs are selected, decisions are translated into lower-level operations and implemented using the FPIs, that is, selected VMs are launched by the *turnon* operation and the others disabled by *turnoff*.

Function Assignment

NFs can be dynamically allocated among the selected VMs by the control module, which is referred to as function assignment. For a set of function assignments, we can compute the optimal route paths of the policies. However, it is worth pointing out that function assignment has a significant impact on the optimal route paths. We need to assign proper NFs to the VMs based on policy requirements to achieve a maximum overall utility for all operators, taking into account the network resource constraints. The function assignment problem is incorporated in our proposed optimization framework by assigning $f \in F$ to $q \in Q$. The assignment of NFs can be modeled by binary variables $x_{i,j}$ representing whether f_i is assigned to q_j. The NF implementation constraints can be expressed with linear compositions of $x_{i,j}$. Other constraints are generated from the information offered by the measurement flows. The computed assignments can be exported to the NF controller as simple key-value pairs (e.g., <*funcid*, *vmid*>) and sequentially *assigned* to the VMs by the NF manager.

Policy Optimization

In some cases, policies specified by operators are not order-sensitive, that is, the operators only require particular flows to traverse a set of network functions for performance or security purposes without regard to their order. Chaining the NFs in an appropriate order with a global view of the system can bring extra benefits. In the policy optimization problem, the task of our proposed framework is to jointly consider the function assignment and traffic steering with the policies to seek the optimal arrangement of the NF sequences, that is, to find the optimal order of $<s_{k,1}, s_{k,2}, \ldots, s_{k,nk}>$ for each P_k. It is worth noting that there are several functionalities that are sensitive to their positions in the sequence of which we should take special care. For example, a firewall might be required to process the flow before/after all other NFs, and a proxy might be limited in a certain position since it may split a flow into several sessions, thus affecting the following processing. These limits should be considered in the problem. The policies are optimized within the control module, and no control of underlying devices is required.

Quantitative Analysis

In this section, we investigate the function assignment problem as a typical example to quantitatively analyze and assess the capability of the proposed control and optimization framework in designing an optimization solution.

Selected Scenario and Challenges

In the SDN-NFV system, NFs can be dynamically assigned among the VMs, and the assignment of NFs has significant impact on overall performance since the traffic flows are routed based on the assignment. By leveraging the proposed unified control and optimization framework, we can jointly optimize function assignment and traffic steering according to the network topology, user policies, and resource constraints. As shown in Fig. 2, the control module takes as input the user policies, resource information, and topology information, and models the joint optimization problem in terms of the required objective. The modeled problem is then solved by the optimization framework, which outputs a set of key-value pairs indicating the functionality each VM implements and a set of routing paths for the user policies. The key-value pairs are taken as parameters of the *assign* operation, and the routing paths are enforced by the forwarding controller.

However, some challenges need to be tackled in order to exploit our proposed framework in optimizing these large-scale practical systems. First, user policies vary with subscribers and application identities. Each user may specify multiple policies; the policies may conflict with each other, and so might the users. The assignment should coordinate these conflicts to achieve an overall optimization of all users. Second, network resources including bandwidth and computing resources should be taken into consideration given the various traffic volumes of the policies. These resource constraints should not be violated to ensure successful traffic routing. Third, users may want to add or alter processing policies dynamically in real time. There should be time-saving approaches to adapt to real-time changes.

Problem and Solution

Our proposed control and optimization framework can be utilized in the network with an NFV system attached to the edge. The commodity servers are connected to the edge forwarding devices instead of being located inline. For each VM, we assume that the VMs are homogeneous and that their packet processing capacities depend on which NF they implement. For each policy, we steer traffic through a unique NF instance for each functionality instead of distributing the traffic to several VMs; thus, we can delicately control the cost of that policy. The optimal solution to the function assignment problem is incorporated in our proposed control and optimization framework. Moreover, based on the model and principles of the optimization framework, a heuristic approach can further be designed for real-time lightweight uses (e.g., adapting to minor changes in policies) as a quicker reaction.

Optimal Solution: To achieve theoretical optimal performance, we need to jointly optimize function assignment and traffic steering since they are twisted problems. Under the above system settings, we model the function assignment problem by defining binary indicator variables $x_{i,j}$ representing whether f_i is assigned to q_j. The routing path, which is the output of the traffic steering problem, is also denoted by binary variables. The optimization objective can be to minimize the overall hop count or delay for all operators. Take hop count as an example. As shown above, the overall hop count can be expressed as

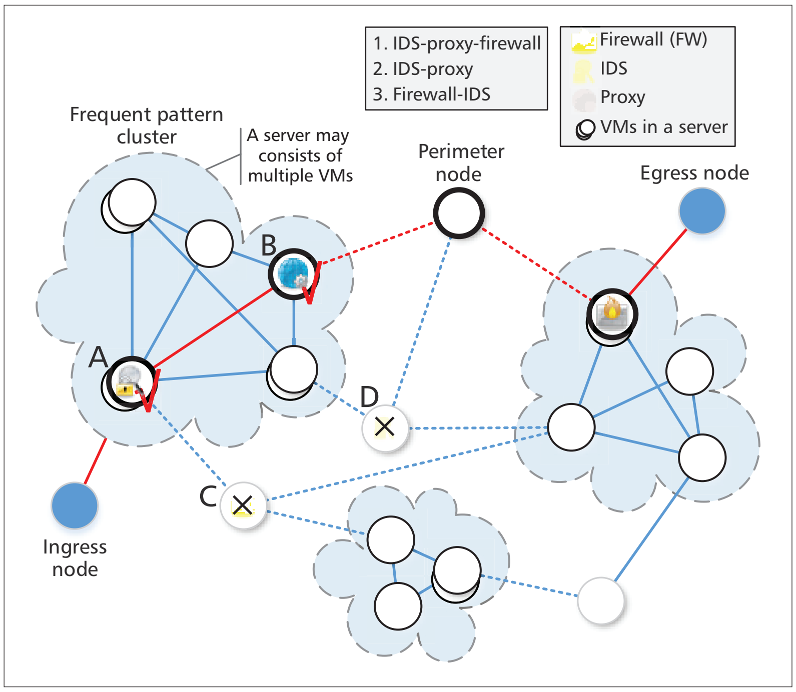

Figure 3. Illustration of the heuristic algorithm for function assignment stage, where frequent function patterns are allocated among neighbouring VMs, forming frequent clusters.

$$\sum_{k=1}^{K}\left(h(i_k, s_{k,1}) + \sum_{i=1}^{n_k-1} h(s_{k,i}, s_{k,i+1}) + h(s_{k,n_k}, d_k)\right).$$

Along with the constraints introduced earlier, we can solve the minimization problem to obtain the optimal function assignments. The problem formulation and solution can easily be extended and applied to other optimization objectives.

Heuristic Solution: Based on the model, objective, and constraints introduced in our proposed optimization framework, a heuristic solution can be designed as a time-saving enforcement of this framework. We decompose the joint optimization problem into two stages by which the optimization variables and constraints are untangled into two parts: the function assignment stage, where we use heuristic approaches to allocate the NFs under the constraints of user policies, the network topology, and resources, and the traffic steering stage, which is incorporated in our optimization framework, minimizing the hop count considering the resource constraints.

Since the traffic steering stage is already introduced above, we focus on the heuristic algorithm of the first stage. We again take the overall hop count as an example of optimization objective to express the heuristic solution. The basic idea of this algorithm is to allocate the most frequent service patterns to close VM neighbors, forming a number of frequent pattern clusters in the network topology. A service pattern is a pair of services that appear consecutively in the service chains regardless of the order. By clustering the most frequent patterns, the shortest paths will be chosen for the most frequent function patterns in the traffic routing stage to minimize the total hop count. We can also cluster the patterns with other metrics according to the optimization objective. The less used patterns are located at

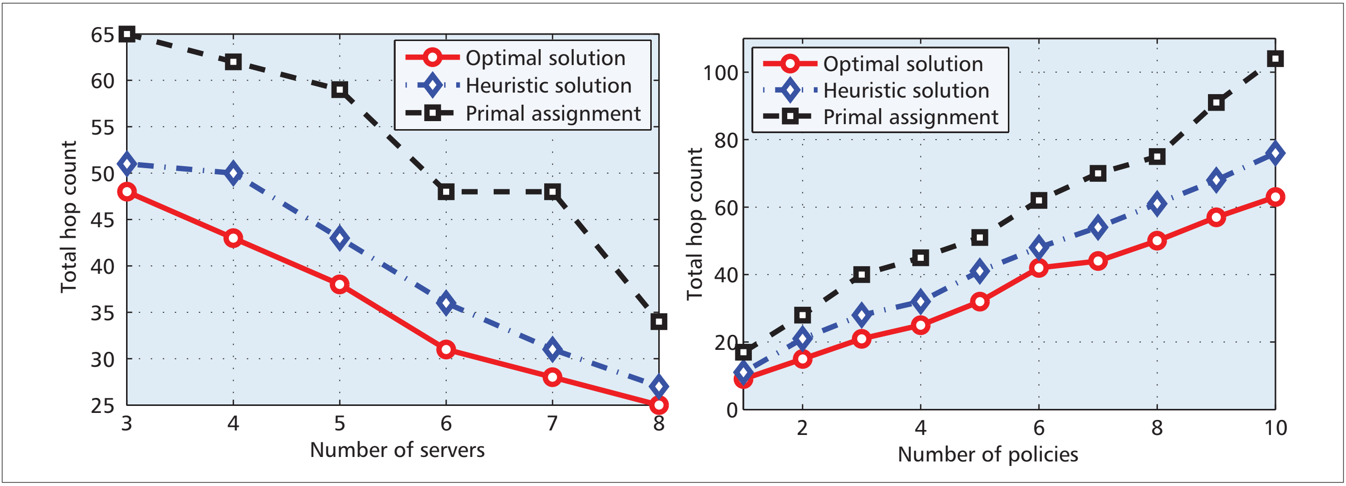

Figure 4. Hop counts v.s. server number and policy number in B4 topology, 2 VMs per server, $T_P = 0.4$: a) hop counts vs. number of servers, policies = 4; b) hop counts vs. number of policies, servers = 6.

the perimeter of these clusters. Prior to the pattern clustering, an appropriate number of VMs is allocated to each functionality based on the NF loads that are estimated according to the policies and their traffic volumes.

Assume that there are three user policies: IDS-proxy-firewall, IDS-proxy, and firewtall-IDS, as shown in Fig. 3, where each node denotes a server, and there might be multiple VMs in each server. There are three clusters of compactly distributed VMs. Some perimeter VMs that are relatively far from these clusters connect them. According to the three user policies, (IDS, proxy) is a frequent pattern in this example. The heuristic algorithm then intends to locate the functions in pattern (IDS, proxy) on nodes A and B, which are neighboring nodes in the same cluster, instead of on perimeter nodes C and D. We do the same thing to the other frequent patterns. The shortest path of policy IDS-proxy-firewall is shown based on the assignment.

Performance Evaluation

We evaluate the performance of the above proposed solution in both backbone and fat tree topologies [12]. For the backbone topologies, NFV servers could be attached to any of the nodes. For the fat tree topology, servers are only allowed to be connected to the edge forwarding devices. The link capacities are assumed to be equal for the backbone topologies, and for the fat tree topology the bandwidth of the links between the core and aggregation layer is two times that of the links between the aggregation layer and the edge. The placement of the servers is decided randomly for each simulation loop. We assume there are three NFs in F, and the NF-associated VM capacities are set based on practical experience to make them reasonable. The service sequences of user policies are also generated based on practical experience and some basic principles. Proper randomness is allowed to simulate the variety of user policies in realistic scenarios. The traffic volume of each policy obeys power-law probability distribution [13], and we adjust the amplitude T_P to control the overall traffic volume. The benchmark of our evaluations is the state-of-the-art algorithm called primal assignment [4]. It works as follows: first the VM numbers are allocated in the same fashion as the heuristic solution does prior to the assignment; then the NFs are randomly assigned to the VMs according to their numbers.

Figure 4 shows the overall hop counts and their variations under the topology of B4 [15], the inter-data center (DC) topology of Google. Each server consists of two VMs. Figure 4a shows the total hop count as a function of server numbers. The total hop count decreases with the increment of server number for both algorithms. The optimal solution and heuristic solution cut off the overall hop count by 34.4 and 27.3 percent on average. Figure 4b shows the total hop count as a function of policy numbers. Solutions based on our proposed framework outperform the primal assignment at all policy numbers. The benefit increases as the policy number increases. The optimal solution and heuristic solution cut the overall hop count by 40.2 and 25.5 percent on average. Figure 4 indicates that our proposed optimization framework enables effective policy-aware function assignments and traffic routing in backbones regardless of the scale of NFV system and the policy set. Moreover, the performance gain increases as the policy set gets larger. Figure 5 illustrates similar results under fat tree topology [12]. These results indicate that both solutions based on our optimization framework attain significant gain in overall cost of the operators in both WAN and DC networks, which further demonstrate the capability of our proposed control and optimization framework to jointly optimize various variables and achieve better overall performance.

Conclusions

In this work, we propose a unified control and optimization framework for dynamic service chaining for an SDN-NFV system that optimizes the system via jointly controlling the underlying devices with novel interfaces. The proposed framework enables joint modeling and optimization of crucial technical problems along with efficient enforcement in an SDN-NFV system.

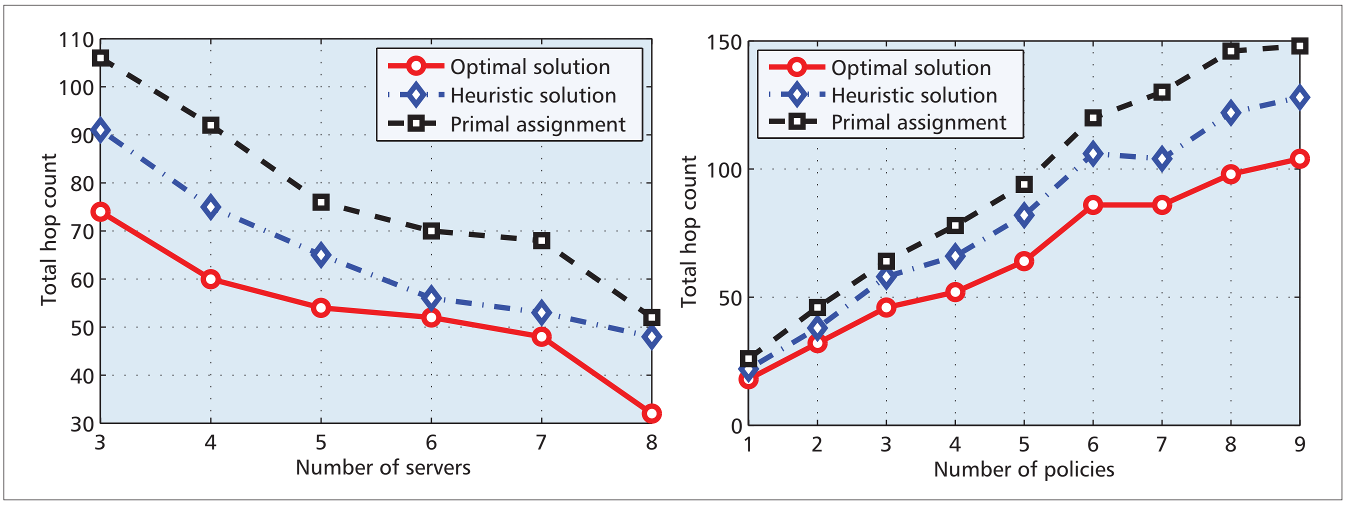

Figure 5. Hop counts vs. server number and policy number in fat tree topology where $k = 4$, 2 VMs per server, $T_P = 0.4$: a) hop counts vs. number of servers, policies = 4; b) hop counts vs. number of policies, servers = 6.

Simulations for the joint optimization problem of function assignment and traffic steering illustrate that our proposed framework cuts the overall user cost by at least 25.5 percent, which demonstrates the effectiveness of our proposed framework. This study thus provides a useful scheme for optimizing dynamic service chaining, and opens up a new research direction for the design of SDN-NFV systems.

Acknowledgment

This work is supported by the National Basic Research Program of China (973 Program) (No. 2013CB329001) and National Nature Science Foundation of China (No. 61301080, No. 61171065 and No. 61273214, No. 91338203, and No. 91338102).

References

[1] K. Zheng *et al.*, "10 Gb/s HetSNets with Millimeter-Wave Communications: Access and Networking — Challenges and Protocols," *IEEE Commun. Mag.*, vol. 53, no. 1, Jan. 2015, pp. 222–31.
[2] R. Guerzoni *et al.*, "Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges and Call for Action, Introductory White Paper," SDN and OpenFlow World Congress, 2012.
[3] N. McKeown *et al.*, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Comp. Commun. Rev.*, vol. 38, no. 2, 2008, pp. 69–74.
[4] J. Martins *et al.*, "Clickos and the Art of Network Function Virtualization," *Proc. USENIX NSDI '14*, pp. 459–73.
[5] J. Hwang, K. Ramakrishnan, and T. Wood, "Netvm: High Performance and Flexible Networking Using Virtualization on Commodity Platforms," *Proc. USENIX NSDI '14*.
[6] Z. A. Qazi *et al.*, "Simple-Fying Middlebox Policy Enforcement Using SDN," *Proc. ACM SIGCOMM '13*, pp. 27–38.
[7] Y. Zhang *et al.*, "Steering: A sOftware-Defined Networking for Inline Service Chaining," *Proc. IEEE ICNP '13*, pp. 1–10.
[8] S. K. Fayazbakhsh *et al.*, "Enforcing Network-Wide Policies in the Presence of Dynamic Middlebox Actions Using Flowtags," *Proc. USENIX NSDI '14*.
[9] A. Gember *et al.*, "Stratos: A Network-Aware Orchestration Layer for Virtual Middleboxes in Clouds," arXiv preprint arXiv:1305.0209, 2013.
[10] A. Gember-Jacobson *et al.*, "Opennf: Enabling Innovation in Network Function Control," *Proc. ACM SIGCOMM '14*, pp. 163–74.
[11] C.-Y. Hong *et al.*, "Achieving High Utilization with Software-Driven WAN," *Proc. ACM SIGCOMM '13*, pp. 15–26.
[12] C. E. Leiserson, "Fat-Trees: Universal Networks for Hardware-Efficient Supercomputing," *IEEE Trans. Comp.*, vol. 100, no. 10, 1985, pp. 892–901.
[13] M. E. Crovella and A. Bestavros, "Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes," *IEEE/ACM Trans. Networking*, vol. 5, no. 6, 1997, pp. 835–46.
[14] K. Zheng *et al.*, "Quality-of-Experience Assessment and Its Application to Video Services in LTE Networks, *IEEE Wireless Commun.*, vol. 22, no. 1, 2015, pp. 70–78.
[15] S. Jain *et al.*, "B4: Experience with a Globally-Deployed Software Defined WAN," *Proc. ACM SIGCOMM '13*, 2013, pp. 3–14.

Biographies

YONG LI [M'09] received his B.S. and Ph.D degrees from Huazhong University of Science and Technology (HUST)and Tsinghua University in 2007 and 2012, respectively. During 2012 and 2013, he was a visiting research associate with Telekom Innovation Laboratories and Hong Kong University of Science and Technology, respectively. During 2013 to 2014, he was a visiting scientist with the University of Miami, Florida. He is currently a faculty member in the Department of Electronic Engineering, Tsinghua University. His research interests are in the areas of mobile computing and social networks, urban computing and vehicular networks, and network science and future Internet. He has served as General Chair, Technical Program Committee (TPC) Chair, and TPC member for several international workshops and conferences. He is currently Associate Editor of the *Journal of Communications and Networking* and *EURASIP Journal of Wireless Communications and Networking*.

FENG ZHENG received B.E. and Master's degrees from the Department of Electronic Engineering, Tsinghua University in 2010 and 2013, respectively. His research interests include data center networks, cloud computing, and software-defined networks.

MIN CHEN [SM'09] is a professor in the School of Computer Science and Technology at HUST. He was an assistant professor in the School of Computer Science and Engineering at SNU from September 2009 to February 2012. He worked as a post-doctoral fellow in the Department of Electrical and Computer Engineering at the University of British Columbia (UBC) for three years. Before joining UBC, he was a postdoctoral fellow at SNU for one and a half years. His research focuses on the Internet of Things, machine-to machine communications, body area networks, body sensor networks, e-healthcare, mobile cloud computing, cloud-assisted mobile computing, ubiquitous networks and services, mobile agents, multimedia transmission over wireless networks, and more.

DEPENG JIN received B.S. and Ph.D. degrees from Tsinghua University in 1995 and 1999, respectively, both in electronics engineering. He is an associate professor at Tsinghua University and vice chair of the Department of Electronic Engineering. He was awarded the National Scientific and Technological Innovation Prize (Second Class) in 2002. His research fields include telecommunications, high-speed networks, ASIC design, and future Internet architecture.