An Online Placement Scheme for VNF Chains in Geo-Distributed Clouds

Ruiting Zhou[†][*]
[†]School of Cyber Science and Engineering, Wuhan University
[*]Department of Computer Science, University of Calgary, rzho@ucalgary.ca

***Abstract*—Network Function Virtualization (NFV) provides virtualized network services through service chains of virtual network functions (VNFs). VNFs typically execute on virtual machines in a cloud infrastructure, which consists of geo-distributed cloud data centers. Compared to traditional cloud services, key challenges in virtual network service provisioning lie in the optimal placement of VNF instances while considering inter-VNF traffic and end-to-end delay in a service chain. The challenge further escalates when a service chain requires online processing upon the its arrival. We propose an online algorithm to address the above challenges, while aim to maximize the aggregate chain valuation. We first study a one-time VNF chain placement problem. Leveraging techniques of exhaustive sampling and ST rounding, we propose an efficient one-time algorithm to determine the placement scheme of a given service chain. We then propose a primal-dual online placement scheme that employs the one-time algorithm as a building block to make decisions upon the arrival of each chain. Through both theoretical analysis and trace-driven simulations, we verify that the online placement algorithm is computationally efficient and achieves a good competitive ratio.**

I. Introduction

Traditional network functions, *e.g.*, intrusion detection systems (IDSs), firewall, proxies and network address translation (NAT) are typically implemented on dedicated hardware, which often require over-provisioning and are inflexible to update. More recently, Network Function Virtualization (NFV) emerges as an agile alternative. NFV advocates a new network architecture that employs IT virtualization to consolidate network functions onto virtualized, cloud-based platforms *e.g.,* virtual machines (VMs) on industry standard servers, to provide communication services [3]. NFV benefits network service providers by significantly reducing the deployment cost and management overhead, as well as improving service agility.

In an NFV environment, a virtual network function (VNF) takes on the responsibility of handling specific network functions that run on one or more virtual machines (VMs). A VNF by itself does not necessarily provide a usable product or service. An NFV service provider implements service chains to provide complex services to its customers. A service chain refers to the structure of a network service where a sequence of VNFs are linked, with a predefined direction of traffic flow [2]. A service chain can either be deployed within a single data center or distributed over multiple data centers. For example, consider a service chain "S-CSCF$\rightarrow$P-CSCF" which vitualizes the signal control panel in IP Multimedia Subsystem (IMS) [5]. S-CSCF registers user equipments (UEs) for IP multimedia services and P-CSCF contacts UEs and the IMS network. The service chain can be placed close to IMS UEs; or instances of S-CSCF are distributed close to UEs, with instances of P-CSCF reside in the full IMS network.

This work was supported in part by NSFC (61370108, 61672257, 61571335, 61628209) and by Hubei Science Foundation (2016CFA030, 2017AAA125).

This work focuses on the dynamic deployment of VNF chains over geo-distributed cloud data centers. We take the perspective of a network service provider, who deploys VNFs on VMs and assembles service chains upon requests on the fly. The deployment of a VNF chain involves not only the placement of VNFs, *i.e.,* allocating each VNF to a data center, but also routing inter-VNF traffic, *i.e.,* identifying paths with available bandwidth to send traffic between neighbor VNFs. Furthermore, as an important performance indicator, the end-to-end delay of a flow to pass through its VNF chain should be bounded. Even in the offline setting with full information, such a chain deployment problem translates into an NP-hard combinational optimization problem. The challenge further escalates when we target a practical online placement scheme that makes on-spot decisions upon the arrival of each chain.

II. Results Overview

We extend the existing literature on VNF provisioning, and propose an efficient online placement scheme such that: i) service chains with different values arrive stochastically; each chain specifies its required VNFs, the traffic demand between two consecutive VNFs and an upper bound of its end-to-end delay; ii) the algorithm is computationally efficient and executes in polynomial time; iii) the aggregate value of deployed chains is approximately maximized.

A. Problem Formulation

We formulate the offline optimization problem into an integer program (IP) with quadratic constraints that capture inter-VNF traffic cost and end-to-end-delay. While polynomial in size, the quadratic IP admits no direct application of the classic primal-dual schema for algorithm design. We leverage the recent *compact-exponential* optimization framework [6] to encode each valid placement scheme in a variable, and reformulate the original IP into a *compact-exponential Integer Linear Program (ILP)* as below, which contains only conventional packing-type constraints, but at the cost of involving an exponential number of variables.

$$\text{maximize} \sum_{i\in[I]}\sum_{l\in\zeta_i} b_{il}x_{il} \tag{1}$$

$$\text{subject to:} \sum_{\substack{i\in[I]:\\ t_i^-\le t\le t_i^+}}\sum_{l\in\zeta_i} f_{m,t}^{il}x_{il} \le C_m,\ \forall m\in\mathcal{M}, \forall t\in[T], \tag{1a}$$

$$\sum_{l\in\zeta_i} x_{il} \le 1,\ \forall i\in[I], \tag{1b}$$

$$x_{il}\in\{0,1\},\ \forall i\in[I], \forall l\in\zeta_i. \tag{1c}$$

In the above compact-exponential ILP, ζ_i is the set of feasible placement schemes for request i. A *feasible* scheme l is a vector that satisfies end-to-end delay constraints. Variable $x_{il}\in\{0,1\}$ indicates whether request i's scheme l is accepted (1) or not (0). Constraint (1a) is the resource capacity constraint. Constraint (1b) ensures that each service chain is placed according to at most one scheme.

Solving the compact exponential ILP directly is still infeasible in practice, when complete knowledge over the entire system lifespan is not available. We instead first relax the resource capacity constraints that impose inter-chain coupling, and focus on a one-time problem to determine the optimal placement of the current VNF chain. We then design an online algorithm framework that simultaneously works on the compact-exponential ILP and its dual LP, invoking the one-time algorithm as a subroutine, towards computing efficient placement schemes based on the value of dual variables.

B. One-time Algorithm Design

We reformulate the one-time VNF chain placement problem into an integer quadratic program (IQP), to minimize chain placement cost. We first consider a simplified scenario of a single type of computational resource. The IQP has an objective function of degree 2, and is proven NP-hard to solve. We apply an *exhaustive sampling* technique [1] based on a random-sampling process to reduce its degree from 2 to 1, at the cost of losing some accuracy. The degree-reduced problem becomes a general assignment problem with extra constraints. We solve this problem to optimal, and apply the *ST rounding* technique [4] to round the fractional solution to an integral solution. More specifically, we construct a bipartite graph based the fractional solution, and output the minimum-cost integer matching in this graph. We then further consider the general scenario, and propose a heuristic algorithm to provide good solutions with low computational complexity.

C. Online Placement Scheme Design

We proceed to consider resource capacity constraints, and design an online algorithm framework that utilizes the one-time algorithm to determine each chain's placement upon its arrival, without relying on future information. We apply the primal-dual technique to the compact-exponential ILP and its dual LP, and interpret dual variables as unit resource prices at different times. Upon receiving a chain request, given current resource prices, the one-time algorithm computes an α-approximate placement scheme with an estimated cost. We divide the estimated cost by α to obtain a lower bound of the optimal cost, and compare the chain value with it. If the value is higher, the chain is deployed and dual variables are updated; otherwise the chain is discarded.

More specifically, A_{online} in Algorithm 1 is our online algorithm framework, which calls A_{sub} for each chain to determine its placement scheme. By default, all variables are set to zero. Line 1 initializes the value of λ, to prepare for the update of the dual variable $p_{m,t}$. Upon the arrival of a request i, we first compute the computation and communication costs when assigning VNF instances to different data centers in line 3. A_{sub} is executed for each chain to compute a placement scheme l^* and the corresponding cost $cost_i$. If request i can obtain a positive utility in some scheme, it is accepted, and the corresponding primal and dual variables are updated (lines 6-8). We then increase the usage of resources and raise resource prices accordingly (lines 9-12).

Algorithm 1 A Primal-dual Online Framework A_{online}

```
Input: $\{\Phi_i\}, \{C_m\}, \{a_{vk}\}, U, \alpha$
 1: Initialize $\lambda = 2(\alpha U + 1)$;
 2: Upon the arrival of the $i$th chain
 3: Compute $\{P_{vs}^i\}$ and $\{P_{v_1,v_2,s_1,s_2}^i\}$ based on $\{p_{m,t}\}$;
 4: $(cost_i, l^*) = A_{sub}(\Phi_i, \{P_{vs}^i\}, \{P_{v_1,v_2,s_1,s_2}^i\}, \{a_{vk}\})$;
 5: if $b_i - \frac{cost_i}{\alpha} > 0$ then
 6:     $u_i = b_i - \frac{cost_i}{\alpha}; x_i = 1; x_{il^*} = 1$;
 7:     Update $y_{vs}^i$ and $f_{m,t}^{il^*}$ according to option $l^*$.
 8:     Accept chain $i$ and allocate its service chain according
        to $y_{vs}^i$;
 9:     for $t \in [t_i^-, t_i^+]$ do
10:         $z_{m,t} = z_{m,t} + f_{m,t}^{il^*}, \forall m \in \mathcal{M}$;
11:         $p_{m,t} = \lambda^{\frac{z_{m,t}}{C_m}} - 1, \forall m \in \mathcal{M}$;
12:     end for
13: else
14:     Reject chain $i$.
15: end if
```

We conduct theoretical analysis on the competitive ratio and prove its upper bound. The effectiveness of our one-time and online algorithms are evaluated through trace-driven simulation studies.

References

[1] S. Arora, D. Karger, and M. Karpinski. Polynomial time approximation schemes for dense instances of NP-hard problems. In *Proc. of ACM STOC*, 1995.

[2] S. Gu, Z. Li, C. Wu, and C. Huang. An efficient auction mechanism for service chains in the NFV market. In *Proc. of IEEE INFOCOM*, 2016.

[3] J. Martins, M. Ahmed, C. Raiciu, V. Olteanu, M. Honda, R. Bifulco, and F. Huici. ClickOS and the art of network function virtualization. In *Proc. of USENIX USDI*, 2014.

[4] D. B. Shmoys and va Tardos. An approximation algorithm for the generalized assignment problem. *Mathematical Programming*, 62(1):461–474, 1993.

[5] Wikipedia. *IP Multimedia Subsystem*. https://en.wikipedia.org/wiki/IP_Multimedia_Subsystem.

[6] R. Zhou, Z. Li, C. Wu, and Z. Huang. An efficient cloud market mechanism for computing jobs with soft deadlines. *IEEE/ACM Transactions on Networking*, 25(2):793–805, 2017.