Control of Multi-Resource Infrastructures: Application to NFV and Computation Offloading

Yeongjin Kim[†], Hyang-Won Lee[‡] and Song Chong[♮]
[†]Samsung Electronics
[‡]Department of Software, Konkuk University
[♮]School of Electrical Engineering, KAIST
E-mail: [†]yj.kim@netsys.kaist.ac.kr, [‡]leehw@konkuk.ac.kr, [♮]songchong@kaist.edu

***Abstract*—Network function virtualization (NFV) and Computation offloading (CO) are state-of-the-art technologies for flexible utilization of networking and processing resources. These two technologies are closely related in that they enable multiple physical entities to process a function provided in a service, and the service (or end host) chooses which resources to use. In this paper, we propose a generalized dual-resource system, which unifies NFV service and CO service frameworks, and formulate a multi-path problem for choosing resources to use in NFV and CO services. The problem is reformulated as a variational inequality by using Lagrange dual theory and saddle point theory. Based on this formulation, we propose an extragradient-based algorithm that controls and splits the sending rate of a service. We prove that the algorithm converges to an optimal point where system cost minus service utility is minimized. Simulations under diverse scenarios demonstrate that our algorithm achieves high quality of service while reducing the system cost by jointly considering dual-resource coupling and service characteristics.**

I. Introduction

In traditional networks, most of the network functions, such as deep packet inspection (DPI), intrusion detection system (IDS), firewall and charging, are implemented exclusively on specific hardware. For instance, data accounting in LTE system is implemented in the P-GW, and thus, all the packets to be charged must pass through the P-GW. This paradigm possibly results in inefficient utilization of networking resources, as some nodes can be heavily congested while other nodes are idle, which degrades the quality of network service. Network function virtualization (NFV) addresses this issue by enabling network functions to be implemented virtually in several nodes in the network [1]. Through dynamic service chaining, a network service can be routed to any one of multiple candidate paths which have different characteristics of delay, throughput, and cost. Hence, NFV has several advantages in terms of resource efficiency, manageability, and scalability [2]. Network service providers in the U.S., AT&T and Verizon, have launched NFV-enabled core networks for multi-protocol label switching (MPLS), wide area network (WAN) optimization and secure connectivity in 2016 [3], [4].

Meanwhile, computation offloading (CO) enables end host with limited processing capacity to offload computation functions, such as transcoding and voice/image processing, to a more powerful machine in the edge or remote cloud, by using additional network bandwidths [5]. Consequently, multiple offloading options are available to the end host, such as processing the function locally or offloading the function to one of the edges or remote clouds. By utilizing processing resources in the cloud, the end host can reduce its computation delay/cost, or increase its computation throughput.

The aforementioned NFV service and CO service are closely related in the context as follows. There can be multiple physical entities that can process a given function, e.g., network function for NFV service and computation function for CO service, and hence, the service (or end host) can choose which resources to use in order to process the function it provides [6]. This problem inherently contains the traditional routing problem but poses additional complexity since processing resource (such as CPU capacity) as well as networking resource (such as link capacity) must be considered [7]. For example, it may be hard to fully utilize a node having plenty of processing resources, if paths to the node have limited link capacities. Unlike the traditional routing where a path is determined based solely on networking resources, with NFV and CO services, a path should be determined in the form of a series of processing and networking (i.e., dual) resources. Then, by choosing an efficient path, resource utilization or quality of services (QoS) can be improved. Moreover, we can expect additional system cost reduction or QoS enhancement if the service can utilize multiple candidate paths, simultaneously.

There have been various studies for dual-resource sharing. Several metrics are proposed for efficiency and fairness of multi-resource sharing [8]–[10], which are the generalizations of existing metrics for single-resource sharing. Shin *et al.* [11] show that conventional TCP and active queue management (AQM) schemes can significantly lose throughput and suffer unfairness under processing-constrained networks, and propose a new AQM scheme for a dual-resource environment. Li *et al.* [6] and Obadia *et al.* [12] propose virtual network function (VNF) placing and single-path routing algorithms in NFV-enabled networks. Kwak *et al.* [13] propose a computation offloading policy of end user in cloud computing environment by jointly considering dual-resources of a local device. Zhao *et al.* [14] propose a load balancing algorithm for computation offloading in data centers by jointly considering dual-resources of cloud servers.

Although there are a number of papers addressing issues on NFV and CO, separately, there has been little effort to investigate the aforementioned problems in a single framework. However, in a real environment, different service types, such as traditional routing service, NFV service and CO service, co-exist and share networking and processing[1] resources, and hence, these services should be jointly managed in a realistic sense. In this paper, we develop a unified framework for NFV and CO services in a dual-resource sharing environment where either networking or processing can be bottlenecked. Our framework formulates the generic problem that subsumes the traditional routing, NFV, and CO problems, and hence enables to solve any instance of aforementioned problems regardless of service type via an algorithm solving the generic problem.

Under virtualization, the entire system is viewed as a (virtual) resource, and a service tends to use many resources all over the system to improve the QoS. However, this can result in additional costs, such as delay, due to the use of remote resources and congestion due to the use of multiple paths. To alleviate this issue, our formulation introduces a cost function in addition to the service utility. Hence, our objective is to minimize the system cost (caused by networking and processing resources) minus quality of service, under given resource capacity constraints. We reformulate our problem into variational inequality by using Lagrange dual theory and saddle point theory. We propose an extragradient-based algorithm which controls the sending rate of services and splits it into multiple candidate paths by jointly taking into account dual-resource coupling and service characteristics. Our algorithm has the advantages in that it can be implemented in a distributed manner as well as it convergences to an optimal solution. We run simulations and evaluate the proposed algorithm under diverse scenarios, including network services for NFV and computation services for CO.

The contributions of this paper are summarized as follows:

- We model network function virtualization and computation offloading in a unified framework as a dual-resource sharing system.
- We formulate the multi-path problem for unified NFV and CO as a variational inequality.
- We develop an extragradient-based sending rate control and multi-path routing algorithm for minimizing the system cost minus the utility of service.
- We prove that the proposed algorithm converges to an optimal solution.
- We thoroughly evaluate our algorithm based on simple scenarios which provide insight on how NFV and CO services are routed depending on system parameters, and a real large-scale scenario to examine its applicability in practice.

In the rest of this paper, we begin with system model in Section II. We propose the algorithm in Section III. In Section IV, we prove convergence of the algorithm to the optimal solution. In Section V, we evaluate the algorithm by simulations. Finally, we conclude this paper in Section VI.

[1]Both network functions and computation functions may be virtualized in the same node.

II. System Model

We consider general topology consisting of networking resources (i.e., links) indexed by $l \in \mathcal{L}$ and processing resources (i.e., nodes) indexed by $k \in \mathcal{K}$. The capacities of resources l and k are denoted by C_l (in bits/sec) and C_k (in cycles/sec), respectively. Each processing resource supports a set of virtual functions, e.g., network functions for NFV service and computation functions for CO service. There are multiple co-existing services indexed by a set $\mathcal{W}$, where each service has a source-destination pair (s_w, d_w)[2] and virtual functions to be processed. Distinct from previous studies [6], [12], [15]–[17], the service can have the same source and destination in our model, i.e., $s_w = d_w$. This is the case for the CO service such that processed functions are consumed by the source of the service. When a service request w occurs, a resource manager, which has complete knowledge on the resource status, informs the set of candidate paths indexed by $\mathcal{P}_w$.[3] Each path $p \in \mathcal{P}_w$ is composed of networking resources $\mathcal{L}_p$ and processing resources $\mathcal{K}_p$. The networking resources in $\mathcal{L}_p$ connect the source s_w and destination d_w[4], and the processing resources in $\mathcal{K}_p$ support the virtual functions provided by service w[5]. We also denote $\mathcal{P} = \cup_{w\in\mathcal{W}}\mathcal{P}_w$ as the set of all candidate paths in our system. We assume that a service can utilize multiple candidate paths, simultaneously [16], [18]. The source of service w regulates sending rate $R_w \geq 0$ (in bits/sec) and splits the sending rate into multiple candidate paths where $x_p \geq 0$ (in bits/sec) is the rate allocation on path p. Then the sending rate of service w is the sum of allocated rates to its candidate paths as follows:

$$R_w = \sum_{p\in\mathcal{P}_w} x_p, \quad \forall w \in \mathcal{W}. \tag{1}$$

A processing resource k is utilized by multiple candidate paths of co-existing services and the virtual functions performed on the resource k may be different for each path p. For example, some paths use the resource k for video transcoding, whereas other paths use the same resource for DPI. To reflect this heterogeneity, we define $\rho_{k,p}$ (in cycles/bit) as the required CPU cycles in resource k for processing a single bit of task on path p, called the processing density. Consequently, the load (in cycles/sec) on processing resource k for path p is $\rho_{k,p}x_p$. Moreover, depending on the functions performed on resource k for path p, the output bit-rates after processing on resource k may be different. For example, if the function is for just checking, such as data accounting and virus scanning, the output bit-rate is the same as input bit-rate. On the

[2]The service with multiple source or destination also can be considered in our framework with minor extension.

[3]How to choose the candidate paths in the resource manager is out of our scope, but we discuss it in Section V.

[4]For the CO service w, it is possible to have $\mathcal{L}_p = \phi$ for the path $p \in \mathcal{P}_w$ using only local computing.

[5]For the traditional routing service w which does not require any virtual function, $\mathcal{K}_p = \phi$ for all path $p \in \mathcal{P}_w$.

other hand, if the function converts the input data, such as encryption, transcoding and packet aggregation, the output bit-rate may change. Note that the change of rate is seen only after processing, and thus seen at the link which is the successor of processing resource on the path. To take into account the change of bit rate after processing, we introduce the parameter $\sigma_{l,p}$ defined as bit conversion ratio seen at networking resource l if traffic on path p is nonzero. Therefore, the load (in bits/sec) on networking resource l for path p is $\sigma_{l,p}x_p$. For the given rate allocation x_p for all paths $p \in \mathcal{P}$, the total loads on networking resource l denoted by F_l (in bits/sec) and processing resource k denoted by F_k (in cycles/sec) can be written as follows:

$$F_l = \sum_{p \in \mathcal{P}_l} \sigma_{l,p} x_p \quad \text{and} \quad F_k = \sum_{p \in \mathcal{P}_k} \rho_{k,p} x_p, \tag{2}$$

where $\mathcal{P}_l$ (and $\mathcal{P}_k$) is the set of paths that the networking resource l (and processing resource k) belongs to. Note that although the expressions of F_l and F_k have similar form, the fundamental bases for the expressions are different as we aforementioned. Next, we define cost functions $D_l(F_l)$ on networking resource l and $D_k(F_k)$ on processing resource k, which can be delay, energy or monetary costs. We assume that $D_l(F_l)$ and $D_k(F_k)$ are convex, increasing in F_l and F_k, respectively, and they have the bounded second derivatives. Define $U_w(R_w)$ as the utility function of service w and assume that $U_w(R_w)$ is concave and increasing in sending rate R_w, and it has the bounded second derivative. The utility function represents the quality of service which depends on the sending rate R_w regardless of how the sending rate is split into candidate paths.

III. Cost-Utility Optimal Algorithm in Dual-resource Sharing

A. Problem Formulation

In this section, we formulate the sending rate control and multi-path routing problem under dual-resource sharing. Our objective is to minimize the total cost of networking and processing resources minus total utility of services subject to capacity constraints of dual-resources.

$$\textbf{(P1):} \min_{\boldsymbol{x}} \left(G(\boldsymbol{x}) = \sum_{l \in \mathcal{L}} D_l(F_l) + \sum_{k \in \mathcal{K}} D_k(F_k) - V \sum_{w \in \mathcal{W}} U_w(R_w) \right), \quad \text{subject to} \begin{cases} x_p \geq 0, & \forall p \in \mathcal{P}, \\ F_l \leq C_l, & \forall l \in \mathcal{L}, \\ F_k \leq C_k, & \forall k \in \mathcal{K}, \end{cases} \tag{3}$$

where $\boldsymbol{x} = (x_p, \forall p \in \mathcal{P})$ is called a primal variable. The constant V is a trade-off parameter between system cost minimization and service utility maximization. Note that the objective function considers only the aggregate rate at the source or resource, i.e., it is not "strictly" convex on $\boldsymbol{x}$, and consequently, there can be multiple optimal solutions achieving the same objective value [19]. In this type of multi-path problem, optimizing only the utility of sending rate can incur unpredictable system cost, such as delay. Hence, our objective of cost-utility minimization enables to find an efficient solution achieving high service utility and low system cost[6]. **(P1)** is a convex optimization problem where the objective is convex on $\boldsymbol{x}$ and the constraints are affine inequalities. We denote $\boldsymbol{x}^* = (x_1^*, \cdots, x_{|\mathcal{P}|}^*) \in \boldsymbol{X}^*$ as any rate allocation vector in the optimal primal solution set $\boldsymbol{X}^*$ for **(P1)**.

The Lagrangian dual problem, which is generally used in convex optimization to handle constraints [20, ch. 5], can be formulated as follows:

$$\textbf{(P2):} \quad \max_{\boldsymbol{\lambda}} \left(\min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda}) \right), \tag{4}$$

$$\text{subject to} \begin{cases} x_p \geq 0, & \forall p \in \mathcal{P}, \\ \lambda_l \geq 0, & \forall l \in \mathcal{L}, \\ \lambda_k \geq 0, & \forall k \in \mathcal{K}, \end{cases} \tag{5}$$

where $\boldsymbol{\lambda} = (\lambda_l, \lambda_k, \forall l \in \mathcal{L}, k \in \mathcal{K})$ is called dual variable, and $L(\boldsymbol{x}, \boldsymbol{\lambda})$ is a Lagrangian function defined as

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = G(\boldsymbol{x}) + \sum_{l \in \mathcal{L}} \lambda_l \left(F_l - C_l \right) + \sum_{k \in \mathcal{K}} \lambda_k \left(F_k - C_k \right). \tag{6}$$

Let $\boldsymbol{\lambda}^* \in \boldsymbol{\Lambda}^*$ be any vector in an optimal dual solution set $\boldsymbol{\Lambda}^*$ for **(P2)**. Then, it is known that minimizers of $L(\boldsymbol{x}, \boldsymbol{\lambda}^*)$ are also primal optimal, i.e., $\arg\min_{\boldsymbol{x} \geq 0} L(\boldsymbol{x}, \boldsymbol{\lambda}^*) \in \boldsymbol{X}^*$, called zero duality gap, when the primal problem is convex optimization [20, p. 226]. By using this technique, we can eliminate the capacity constraints in (3) which cause coupling issues among rate allocations. More precisely, when we develop an iterative projection algorithm, the projection of primal variable $\boldsymbol{x}$ onto the capacity constraints (3) (which is complicated and hard to decentralize) can be transformed into the projection of dual variable $\boldsymbol{\lambda}$ onto positive domain (5) (which can be implemented in a distributed manner[7]). Typically, to find the solution of dual optimization problems, a gradient projection algorithm is used [15]. However, the convergence of gradient projection algorithm is guaranteed only when the primal objective function in **(P1)** is strictly convex on $\boldsymbol{x}$ because this condition makes the objective function in **(P2)** be differentiable on $\boldsymbol{\lambda}$ [15], [19]. Alternatively, a sub-gradient method [17] can be used for non-strictly convex problems, but in our scenario, it is difficult to know the exact sub-gradient value at each iteration, i.e., solving $\min_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda})$ directly for given $\boldsymbol{\lambda}$ is not easy because we consider multiple candidate paths for each service. Therefore, we take an alternative approach based on saddle point theorem presented in the following.

Lemma 1 (Saddle point theorem)**.** [21, prop. 5.1.6]. *Finding the optimal primal and dual solutions* $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ *for* **(P2)** *is equivalent to a saddle point problem which finds* $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) \geq (\mathbf{0}, \mathbf{0})$ *satisfying*

$$\textbf{(P3):} \quad L(\boldsymbol{x}^*, \boldsymbol{\lambda}) \leq L(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) \leq L(\boldsymbol{x}, \boldsymbol{\lambda}^*), \ \forall \boldsymbol{x} \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}. \tag{7}$$

Proof. Please refer to our technical report [22]. □

By reformulating **(P2)** into **(P3)**, we can regard the dual

[6] We show the simulation results for the utility-maximal and cost-utility-minimal algorithms and demonstrate the effectiveness of our objective.

[7] We proposed a distributed projection algorithm in Section III.

variable $\boldsymbol{\lambda}$ being independent of primal variable $\boldsymbol{x}$.

Lemma 2 (Variational inequality problem)**.** *[23, ch. 3.5]. Finding saddle points $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ for* **(P3)** *is equivalent to a variational inequality problem which finds $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) \geq (\mathbf{0}, \mathbf{0})$ satisfying*

$$\textbf{(P4):}\ ((\boldsymbol{x}, \boldsymbol{\lambda}) - (\boldsymbol{x}^*, \boldsymbol{\lambda}^*))^\top f(\boldsymbol{x}^*, \boldsymbol{\lambda}^*) \geq 0,\ \forall \boldsymbol{x} \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}, \quad (8)$$

where $f : \{(\boldsymbol{x}, \boldsymbol{\lambda}) | \boldsymbol{x} \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}\} \mapsto \mathbb{R}^{|\mathcal{P}|+|\mathcal{L}|+|\mathcal{K}|}$ defined as

$$f(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{pmatrix} \nabla_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda}) \\ -\nabla_{\boldsymbol{\lambda}} L(\boldsymbol{x}, \boldsymbol{\lambda}) \end{pmatrix}. \quad (9)$$

Proof. Please refer to our technical report [22]. □

Lemmas 1 and 2 show that problem **(P4)** is equivalent to **(P1)**. From now on, we will concentrate on deriving an algorithm to find the solutions of **(P4)**.

B. Deriving Algorithm

In this section, we propose an extragradient-based [24, ch. 12] sending rate control of services and multi-path routing algorithm in dual-resource sharing. From the initial state $(\boldsymbol{x}^0, \boldsymbol{\lambda}^0) \geq (\mathbf{0}, \mathbf{0})$, the algorithm iterates the process as follows:

$$(\boldsymbol{x}^{t+1}, \boldsymbol{\lambda}^{t+1}) = \left[(\boldsymbol{x}^t, \boldsymbol{\lambda}^t) - \gamma f(\boldsymbol{x}^{t+1/2}, \boldsymbol{\lambda}^{t+1/2}) \right]^+, \quad (10)$$

$$\text{where} \quad (\boldsymbol{x}^{t+1/2}, \boldsymbol{\lambda}^{t+1/2}) = \left[(\boldsymbol{x}^t, \boldsymbol{\lambda}^t) - \gamma f(\boldsymbol{x}^t, \boldsymbol{\lambda}^t) \right]^+, \quad (11)$$

and γ is a constant step size for the iteration and $[\cdot]^+$ is an orthogonal projection onto $\{(\boldsymbol{x}, \boldsymbol{\lambda}) | \boldsymbol{x} \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}\}$. Unlike gradient-based approaches, (10) moves $(\boldsymbol{x}^t, \boldsymbol{\lambda}^t)$ to the gradient direction at $(\boldsymbol{x}^{t+1/2}, \boldsymbol{\lambda}^{t+1/2})$ not $(\boldsymbol{x}^t, \boldsymbol{\lambda}^t)$.

To implement the algorithm (10), the value of $(\boldsymbol{x}^{t+1/2}, \boldsymbol{\lambda}^{t+1/2})$ should be known to each part of the system, including sources of services and networking/processing resources (See (11)). To this end, we separate the computations (10) and (11) in one iteration into a series of two iterations, i.e., performs (11) in odd-numbered time slots and (10) in odd-numbered time slots, and hence, the algorithm operates in a distributed manner.

Extragradient-based sending rate control and multi-path routing algorithm.

Initial state: $\boldsymbol{x}^0 \geq \mathbf{0}, \boldsymbol{\lambda}^0 \geq \mathbf{0}$, step size: γ,

At each iteration $\tau = 1, 2, ...$

1: Substitute $(\boldsymbol{x}, \boldsymbol{\lambda})$ as follows:

$$(\boldsymbol{x}, \boldsymbol{\lambda}) = \begin{cases} (\boldsymbol{x}^{\tau-1}, \boldsymbol{\lambda}^{\tau-1}), & \text{if } \tau \text{ is odd,} \\ (\boldsymbol{x}^{\tau-2}, \boldsymbol{\lambda}^{\tau-2}), & \text{otherwise.} \end{cases} \quad (12)$$

2: Each networking resource $l \in \mathcal{L}$ and processing resource $k \in \mathcal{K}$ updates λ_l and λ_k, respectively, as follows:

$$\begin{aligned} \lambda_l^\tau &= \left[\lambda_l - \gamma \left(C_l - F_l^{\tau-1} \right) \right]^+, \\ \lambda_k^\tau &= \left[\lambda_k - \gamma \left(C_k - F_k^{\tau-1} \right) \right]^+, \end{aligned} \quad (13)$$

3: Each source of service w updates x_p for all $p \in \mathcal{P}_w$ as follows:

$$x_p^\tau = \left[x_p - \gamma \left(\sum_{l \in \mathcal{L}_p} \sigma_{l,p} (D_l'(F_l^{\tau-1}) + \lambda_l^{\tau-1}) + \sum_{k \in \mathcal{K}_p} \rho_{k,p} (D_k'(F_k^{\tau-1}) + \lambda_k^{\tau-1}) - V U_{w_p}'(R_{w_p}^{\tau-1}) \right) \right]^+, \quad (14)$$

In (13), to update a dual variable λ_l^τ or λ_k^τ which represents the congestion price, the corresponding networking or processing resource requires the load on the resource. Note that from the given λ_l (or λ_k) determined by (12), λ_l^τ (or λ_k^τ) becomes larger than λ_l (or λ_k) when the total load on resource l (or k) exceeds its capacity and vice versa. As seen in (14), if the congestion price increases, the load on the congested resource is likely to decrease. To update the sending rate of service w and split the sending rate into the candidate paths $\mathcal{P}_w$, the source requires the first derivatives of costs, and congestion prices on dual-resources along with the candidate paths. The source tries to decrease x_p^τ to reduce the system cost and congestions, and increase x_p^τ to improve the service utility. Because the cost functions $D_l(\cdot)$ and $D_k(\cdot)$ are convex and increasing, their derivatives $D_l'(\cdot)$ and $D_k'(\cdot)$ are positive and increasing. Hence, the rate decrement is dominated by the bottleneck resources which cause a huge amount of cost and congestion. Moreover, for the same amount of cost and congestion, the rate decrement is dominated by the resources with high bit conversion ratio $\sigma_{l,p}$ or processing density $\rho_{k,p}$. On the other hand, because the utility function $U_w(\cdot)$ is concave and increasing, its derivative $U_w'(\cdot)$ is positive and decreasing. Hence, when the sending rate of service w is low, the source actively increase the rate allocation for all the candidate paths in $\mathcal{P}_w$. Moreover, when our objective is biased towards the utility, i.e., large V, the source aggressively increases the rate allocation. Note that each source and resource need memory to store their own information at $\tau - 2$ (See (12)) which is the additional overhead of the extragradient algorithm compared to gradient algorithms.

IV. Theoretical Analysis

In this section, we prove that our algorithm converges to the optimal solution of **(P4)**, i.e., the optimal solution of **(P1)**. First, we introduce two properties of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ defined in (9), called pseudo-monotonicity and Lipschitz continuity.

Definition 1 (Pseudo-monotonicity)**.** *A function $g : \mathcal{Z} \mapsto \mathbb{R}^n$ is pseudo-monotone on a set $\mathcal{Z} \subset \mathbb{R}^n$ with respect to a set $\mathcal{Z}^* \subset \mathcal{Z}$, $\mathcal{Z}^* \neq \phi$ if for any $\boldsymbol{z}^* \in \mathcal{Z}^*$, the following property holds:*

$$(\boldsymbol{z} - \boldsymbol{z}^*)^\top g(\boldsymbol{z}) \geq 0, \quad \forall z \in \mathcal{Z}. \quad (15)$$

Lemma 3 (Pseudo-monotonicity of f)**.** *Our function f defined in* (9) *is pseudo-monotone on $\{(\boldsymbol{x}, \boldsymbol{\lambda}) | \boldsymbol{x} \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}\}$ with respect to $(\boldsymbol{X}^*, \boldsymbol{\Lambda}^*)$.*

Proof. Please refer to our technical report [22]. □

Lemma 4 (Lipschitz continuity of f)**.** *$f(\boldsymbol{x}, \boldsymbol{\lambda})$ is Lipschitz continuous on $(\boldsymbol{x}, \boldsymbol{\lambda}) \geq (\mathbf{0}, \mathbf{0})$ with some constant $A > 0$, i.e.,*

$$\|f(\boldsymbol{x}^1, \boldsymbol{\lambda}^1) - f(\boldsymbol{x}^2, \boldsymbol{\lambda}^2)\|_2 \leq A\|(\boldsymbol{x}^1, \boldsymbol{\lambda}^1) - (\boldsymbol{x}^2, \boldsymbol{\lambda}^2)\|_2, \quad \forall (\boldsymbol{x}^1, \boldsymbol{\lambda}^1), (\boldsymbol{x}^2, \boldsymbol{\lambda}^2) \geq (\mathbf{0}, \mathbf{0}), \tag{16}$$

Proof. Please refer to our technical report [22]. □

Next, we introduce a well-known projection theorem.

Lemma 5 (Projection theorem)**.** *$\mathcal{Z} \subset \mathbb{R}^n$ is a non-empty, closed, convex set and $[\boldsymbol{x}]^+$ is an orthogonal projection of vector $\boldsymbol{x} \in \mathbb{R}^n$ onto $\mathcal{Z}$, i.e., $[\boldsymbol{x}]^+ = \arg\min_{\boldsymbol{z} \in \mathcal{Z}} \|\boldsymbol{z} - \boldsymbol{x}\|_2$. Then we have*

$$\boldsymbol{y} = [\boldsymbol{x}]^+, \text{ if and only if } (\boldsymbol{z} - \boldsymbol{y})^\top (\boldsymbol{x} - \boldsymbol{y}) \leq 0, \ \forall \boldsymbol{z} \in \mathcal{Z}. \tag{17}$$

Because our domain $\{(\boldsymbol{x}, \boldsymbol{\lambda}) | \boldsymbol{x} \geq \mathbf{0}, \boldsymbol{\lambda} \geq \mathbf{0}\}$ is also non-empty, closed and convex, Lemma 5 holds. Next, we derive a property of iteration (10) using Lemma 3, 4 and 5.

Lemma 6. *The sequence $(\boldsymbol{x}^t, \boldsymbol{\lambda}^t)$ generated by iteration* (10) *and* (11) *has following relationship.*

$$\|(\boldsymbol{x}^{t+1}, \boldsymbol{\lambda}^{t+1}) - (\boldsymbol{x}^*, \boldsymbol{\lambda}^*)\|_2^2 \leq \|(\boldsymbol{x}^t, \boldsymbol{\lambda}^t) - (\boldsymbol{x}^*, \boldsymbol{\lambda}^*)\|_2^2 - (1 - \gamma^2 A^2)\|(\boldsymbol{x}^{t+1/2}, \boldsymbol{\lambda}^{t+1/2}) - (\boldsymbol{x}^t, \boldsymbol{\lambda}^t)\|_2^2. \tag{18}$$

Proof. Please refer to our technical report [22]. □

Theorem 1. *For the step size $\gamma \in (0, 1/A)$, the sequence $(\boldsymbol{x}^t, \boldsymbol{\lambda}^t)$ generated by* (10) *and* (11) *converges to a solution in $(\boldsymbol{X}^*, \boldsymbol{\Lambda}^*)$.*

Proof. Please refer to our technical report [22]. □

Theorem 1 shows that our iterative algorithm finds optimal sending rate control and multi-path routing for **(P1)**. Note that a general gradient algorithm does not guarantee convergence to the optimal solution because it requires strong monotonicity [23, prof. 5.4] of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ which is a stronger condition than the one in Lemma 3.

V. Evaluation

In this section, we present simulation results for our sending rate control and multi-path routing algorithm in dual-resource systems. We first simulate the proposed algorithm in simple scenarios in order to validate the generality of our model and the effect of our algorithm. We also examine our algorithm in a real large-scale scenario to examine its applicability in practice.

A. Simulation in simple scenarios

Topology description. We consider two simple scenarios where three nodes and four directed links are connected as in Fig. 1. In scenario 1, two services with different source/destination require virtual functions in triangular topology, e.g., NFV service scenario. Each service can use one-hop path using two resources and two-hop path using three resources as described in Table. I. In scenario 2, one service with the same source/destination requires virtual functions in serial topology, e.g., CO service scenario. The path using

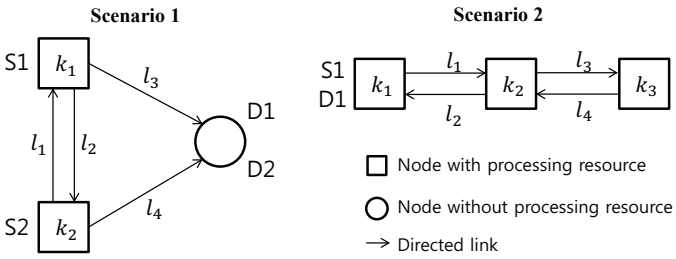
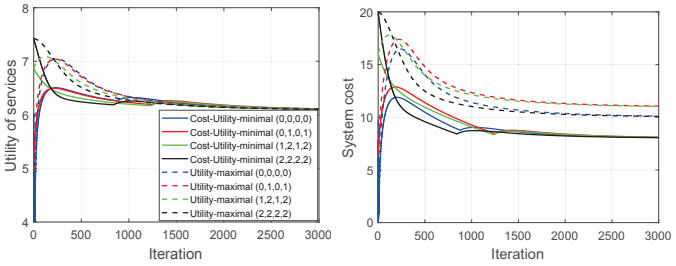


Fig. 1: Topologies of two simple scenarios.



(a) Utility over iterations under different initial rate allocations $\boldsymbol{x}^0$.

(b) System cost over iterations under different initial rate allocations $\boldsymbol{x}^0$.

Fig. 2: Comparison of our algorithm with utility-maximal algorithm in scenario 1 when $C_{k_1} = 2$ GHz, $V = 20$ and $\gamma = \frac{1}{200}$.

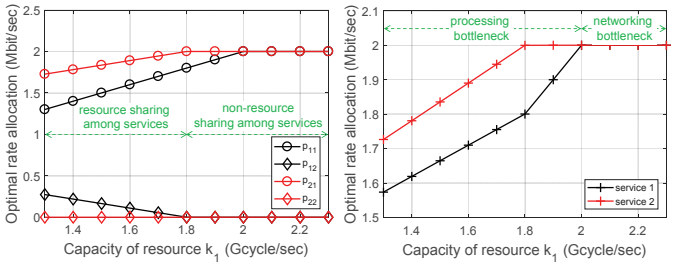
only resource k_1 models local computing, the path using resources k_2 models computation offloading to edge clouds, and the path using resource k_3 models computation offloading to remote clouds. In this scenario, the cloud node has the largest processing capacity, i.e., $C_{k_1} < C_{k_2} < C_{k_3}$, but requires the least cost for the same processing load, i.e., $D_{k_1}(F) > D_{k_2}(F) > D_{k_3}(F)$. The detailed descriptions of two simple scenarios are summarized in Table I.

Impact of cost-awareness in scenario 1. To observe the impact of cost-aware resource management, we compare our cost-utility-minimal algorithm with the utility-maximal algorithm derived assuming no cost term, i.e., $D_l(F) = D_k(F) = 0$ for all $l \in \mathcal{L}, k \in \mathcal{K}$. Fig. 2 shows the service utility and system cost of our algorithm and utility-maximal algorithm over iterations when $C_{k_1} = 2$ GHz, $V = 20$ and $\gamma = \frac{1}{200}$, i.e., completely symmetric topology. We run the algorithms under four different initial rate allocations $\boldsymbol{x}^0 \in \{(0,0,0,0), (0,1,0,1), (1,2,1,2), (2,2,2,2)\}$ Mbps. In Fig. 2(a), both algorithms converge to the maximum utility regardless of initial rate allocations. Note that the maximum utility can be achieved as long as the sending rate of each service reaches 2 Mbps, i.e., $R_1 = R_2 = 2$ Mbps. In Fig. 2(b), in addition to maximum utility, our algorithm also converges to the minimum system cost which can be achieved only when the two services do not share any resources, i.e., $x_{p11} = x_{p21} = 2$ Mbps and $x_{p12} = x_{p22} = 0$ Mbps. On the other hand, depending on the initial rate allocations, the utility-maximal algorithm converges to the different system costs which are not minimum. As mentioned above, there are several

[8]The processing densities of typical computation functions are within the range [0.5, 20] Kcycles/bit [13].

TABLE I: Detailed description of two simple scenarios.

	Scenario 1	**Scenario 2**
source/destination	different	same
number of services	$\|\mathcal{W}\| = 2$	$\|\mathcal{W}\| = 1$
candidate paths	$p_{11} : k_1 \to l_3$ $p_{12} : l_2 \to k_2 \to l_4$ $p_{21} : k_2 \to l_4$ $p_{22} : l_1 \to k_1 \to l_3$	$p_{11} : k_1$ $p_{12} : l_1 \to k_2 \to l_2$ $p_{13} : l_1 \to l_3 \to k_3 \to l_4 \to l_2$
resource capacity (Mbps or GHz)	$C_l = 2, \quad \forall l \in \mathcal{L}$ $C_{k_1} \in \{1.3, 1.4, \cdots, 2.3\}$, $C_{k_2} = 2$	$C_l = 2, \quad \forall l \in \mathcal{L}$ $C_{k_1} = 2$, $C_{k_2} = 20$, $C_{k_3} = 40$
resource cost	$D_l(F_l) = F_l, \quad \forall l \in \mathcal{L}$ $D_k(F_k) = F_k, \quad \forall k \in \mathcal{K}$	$D_l(F_l) = F_l, \quad \forall l \in \mathcal{L}$ $D_{k_1}(F_{k_1}) = F_{k_1}$, $D_{k_2}(F_{k_2}) = \frac{1}{2}F_{k_2}$, $D_{k_3}(F_{k_3}) = \frac{1}{3}F_{k_3}$
service utility	$U_w(R_w) = \log(0.1 + R_w) - \log(0.1)$	$U_w(R_w) = \log(0.1 + R_w) - \log(0.1)$
processing density[8] (Kcycles/bit)	$\rho_{k,p} = 1, \quad \forall k \in \mathcal{K}_p, p \in \mathcal{P}$	$\rho \in \{1, 2, \cdots, 20\}$ $\rho_{k,p} = \rho, \quad \forall k \in \mathcal{K}_p, p \in \mathcal{P}$
bit conversion ratio (bits/bit)	$\sigma_{l,p} = 1, \quad \forall l \in \mathcal{L}_p, p \in \mathcal{P}$	$\sigma \in \{0.05, 0.1, \cdots, 1\}$ $\sigma_{l_1,p_{12}} = 1, \sigma_{l_2,p_{12}} = \sigma$ $\sigma_{l_1,p_{13}} = 1, \sigma_{l_3,p_{13}} = 1, \sigma_{l_4,p_{13}} = \sigma, \sigma_{l_2,p_{13}} = \sigma$



(a) Optimal rate allocations for each path under different values of C_{k_1}.

(b) Optimal rate allocations for each service under different values of C_{k_1}.

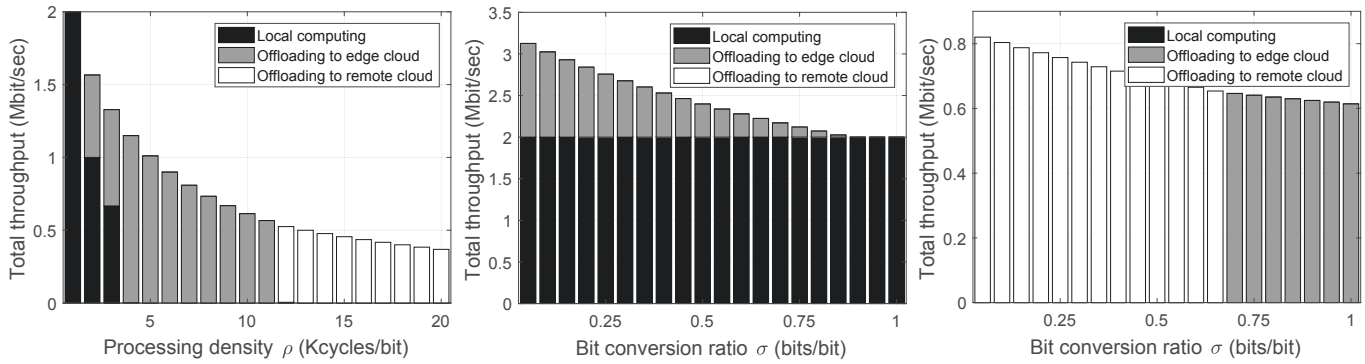
Fig. 3: Impact of resource availability for our algorithm in scenario 1 when $V = 20$.

utility-optimal points and these points may attain different system cost such as delay. Simulation results clearly show that with NFV or CO services, it is desirable to consider cost function as well as throughput performance, like our algorithm does.

Impact of resource availability in scenario 1. To observe the behavior of our algorithm under different resource availabilities, we change the capacity of processing resource k_1, i.e., C_{k_1}, from 1.3 GHz to 2.3 GHz. Fig. 3 shows optimal rate allocations of our algorithm for each path (Fig. 3(a)) and each service (Fig. 3(b)) under different values of C_{k_1}. In the region $C_{k_1} \in \{1.3, \cdots, 2.0\}$ GHz where the processing resource k_1 is a bottleneck, rate allocation shows different characteristics for $C_{k_1} \in \{1.3, \cdots, 1.8\}$ GHz and $C_{k_1} \in \{1.8, \cdots 2.0\}$ GHz. In the region $C_{k_1} \in \{1.3, \cdots, 1.8\}$ GHz, service 1 uses both candidate paths p_{11} and p_{12}, and processing resource k_2 and networking resource l_4 are shared by the two services. The reason is when each service uses only its one-hop path, i.e., p_{11} and p_{21}, there is a huge gap between sending rates of two services, which leads to low total utility. Note that because the two-hop path p_{12} incurs higher cost than the one-hop path p_{21} (if the same rate is assigned to both paths), the rate allocation for service 2, i.e., R_2, is higher than that for service 1, i.e., R_1 (See Fig. 3(b)). For the same reason, R_2 increases faster than R_1 as the capacity C_{k_1} increases. In the region $C_{k_1} \in \{1.8, \cdots, 2.0\}$ GHz, service 1 does not use the path p_{12} anymore and the two services do not share any resources. It is because the increment of total utility is no higher than the increment of system cost when the path p_{12} is activated. In other words, high utility can be achieved without using the expensive path p_{12}. In the region $C_{k_1} \in \{2.0, \cdots, 2.3\}$ GHz where the networking resource l_3 is a bottleneck, rate allocations are not changed because path p_{11} is fully utilized and (as mentioned above) using p_{12} only decreases the objective value.

Impact of service characteristics in scenario 2. To observe the behavior of different service parameters, we change the processing density ρ and bit conversion ratio σ for the required computation function[9]. Fig. 4 shows the total throughput of the service served by local computing, and computation offloading to edge and remote cloud. The total throughput decreases as ρ and σ increase because the service, with high processing density ρ and bit conversion ratio σ, requires more system cost to achieve the same amount of utility. In Fig. 4(a), under low processing densities, the service is mainly served by local computing because local processing resource is sufficient to handle computation functions with low processing density and thus achieve high throughput without resorting to resources at the edge or remote cloud which only require excessive networking costs. As the processing density increases, the service is mainly served by edge cloud (when $\rho \in \{4, \cdots, 11\}$ Kcycles/bit) and remote cloud (when $\rho \in \{12, \cdots, 20\}$ Kcycles/bit). In this case, in spite of additional networking costs, computation offloading to edge and remote cloud can achieve higher throughput and lower processing costs than local computing. Next, we separate the cases when the processing density is low ($\rho = 1$ Kcycles/bit) and high ($\rho = 10$ Kcycles/bit) and observe the total throughput for different bit conversion ratios σ in Fig. 4(b) and 4(c). For the low processing density in Fig. 4(b), local computing

[9]Please refer to Table I to see how ρ and σ affect $\rho_{k,p}$ and $\sigma_{l,p}$ in scenario 2.

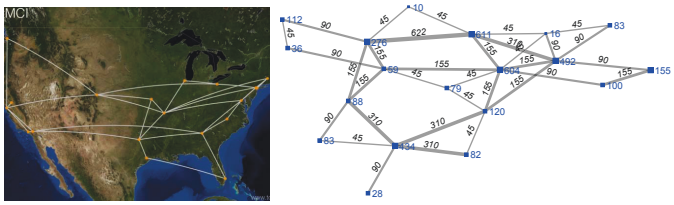


(a) Throughput of service under different processing density ρ when $\sigma = 1$.

(b) Throughput of service under different bit conversion ratio σ when $\rho = 1$.

(c) Throughput of service under different bit conversion ratio σ when $\rho = 10$.

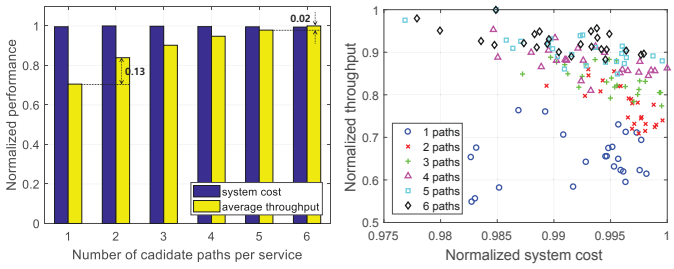
Fig. 4: Impact of service characteristics for our algorithm in scenario 2 when $V = 10$.



(a) Geographical topology.

(b) Topology represented by dual-resource graph.

Fig. 5: The real Internet topology in the U.S. (by MCI [25]).



(a) Normalized cost and throughput under different numbers of candidate paths P.

(b) Normalized cost and throughput under the different set of candidate paths $\mathcal{P}_w$.

is fully utilized regardless of σ because it is highly cost-efficient and independent of σ, i.e., local computing does not use any networking resources. However, due to the limited processing capacity of local computing, edge cloud is also utilized partially depending on its cost-efficiency which is a function of σ, i.e., the lower the bit conversion ratio, the more edge cloud is used. For the high processing density in Fig. 4(c), the computation offloading policy is rapidly changed between edge cloud and remote cloud within the range $\sigma \in \{0.65, 0.70\}$ bits/bit. In this case, both processing resources in edge and remote cloud are not limited, and thus the service is served by only the most cost-efficient path.

B. Simulation in a real large-scale scenario

Topology description. We consider a real dataset of the Internet topology in the U.S. in 2011 by MCI [25] as in Fig. 5(a). This dataset contains 19 nodes with their locations and 45 wired links with their bandwidths. Because the dataset does not specify node processing resources, we synthetically generate them as follows. For each node k, we calculate the total bandwidth of the links connected to the node, denoted by z_k. A node with many neighbors and high link bandwidths may be the hub of the Internet with high processing capacity. Thus the capacity of processing resource for node k is chosen by the Normal distribution $N(z_k/2, z_k/2)$ which is truncated by $[0, z_k]$. The resulting dual-resource graph is presented in Fig. 5(b) where the capacities of networking and processing resources are figured by the width of links and size of nodes, respectively. We define 9 virtual functions with different combinations of processing densities $\rho \in \{1, 4, 10\}$ Kcycles/bit and bit conversion ratios $\sigma \in \{1, 0.5, 0.1\}$ bits/bit, and each node supports a random set of virtual functions where the number of supported functions is proportional to the node's processing capacity. We consider 20 services where the source, destination and required function of each service are chosen randomly. As the cost functions of dual-resources, we use $D_l(F_l) = \frac{F_l}{C_l - F_l}$ and $D_k(F_k) = \frac{F_k}{C_k - F_k}$ which represent the average packet delay of M/M/1 queue [26, p. 434].

Impact of candidate path selection. For simplicity, we use the same number of candidate paths for all services, i.e., $|\mathcal{P}_w| = P$, for all $w \in \mathcal{W}$. We simulate our algorithm under the real large-scale topology by changing the value of $P \in \{1, 2, \cdots, 6\}$. For given P, the candidate paths are chosen randomly from 8-shortest paths for each service by the resource manager[10]. Fig. 6(a) shows the normalized system cost and average service throughput of our algorithm under different numbers of candidate paths. The average throughput of services increases with the number of candidate paths without incurring additional system costs. As the number of candidate paths grows, the sending rate (and thus utility) can be increased, however this will also increase the cost because more paths contribute to the cost. Hence, rate allocation on each candidate path should be carefully determined. This result

[10]We apply this randomness in order to avoid the situation where some resources are used by the most of candidate paths.

shows that our algorithm can exploit multiple paths in order to increase the service utility while keeping the system cost unchanged. However, the effect of increasing P diminishes as P increases, e.g., the throughput increment from $P = 1$ to $P = 2$ is 0.13, whereas it is only 0.02 from $P = 5$ to $P = 6$. Note that larger P requires higher computation overhead of our algorithm, and hence, the number of candidate paths should be carefully determined to balance between the improvement of system performance and computation overhead. Fig. 6(b) shows normalized system cost and average service throughput of our algorithm under different sets of candidate paths. Although the number of candidate paths is the same, the achievable service throughput and system cost highly depend on the selection of candidate paths. Moreover, the system performance with more candidate paths can be worse than that with fewer candidate paths. Thus, it is also important to find an appropriate set of candidate paths incurring low cost, e.g., a short path or less overlapped path with other paths, and supporting high service throughput, e.g., a path composed of high capacity networking and processing resources. We leave this issue for our future work.

Please refer to our technical report [22] to see more simulation results in the large-scale scenario.

VI. Conclusion

In this paper, we studied quality of service and cost optimization problem in a dual-resource system where dynamic service chaining for network function virtualization and offloading policy for computation offloading are modeled in a unified framework. Based on this problem, we developed an extragradient-based algorithm that iteratively decides the sending rate of service and multi-path routing. Our algorithm jointly considers dual-resource coupling and service-dependent properties to efficiently handle the resource and service. We proved that our algorithm converges to an optimal solution. Simulation results demonstrate the importance of cost-aware multi-path routing and candidate path selection. Our results in this paper give an insight into how different kinds of coexisting services that require multiple resource types can be viewed and managed in a unified framework.

VII. Acknowledgement

This work was supported by Institute for Information & communications Technology Promotion (IITP) grant funded by the Korea government (MSIT) (R-20161130-004520, Research on Adaptive Machine Learning Technology Development for Intelligent Autonomous Digital Companion). Hyang-Won Lee was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No.2015R1A1A1A05001477).

References

[1] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, "NFV: State of the art, challenges, and implementation in next generation mobile networks (vEPC)," *IEEE Network*, vol. 28, no. 6, pp. 18–26, 2014.

[2] E. Portal, "Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action," 2012.

[3] "AT&T, FlexWare." [Online]. Available: https://www.business.att.com/enterprise/Family/network-services/virtual-network-functions/

[4] "Verizon, virtual network services." [Online]. Available: http://www.verizon.com/about/news/verizons-virtual-network-services-creates-living-network/

[5] K. Kumar and Y. Lu, "Cloud computing for mobile users: Can offloading computations save energy?" *Computer*, vol. 43, no. 4, pp. 51–56, 2010.

[6] Y. Li, F. Zheng, M. Chen, and D. Jin, "A unified control and optimization framework for dynamical service chaining in software-defined NFV system," *IEEE Wireless Communications*, vol. 22, no. 6, pp. 15–23, 2015.

[7] P. Makris, D. N. Skoutas, and C. Skianis, "A survey on context-aware mobile and wireless networking: On networking and computing environments' integration," *IEEE Communications Surveys Tutorials*, vol. 15, no. 1, pp. 362–386, First 2013.

[8] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types." in *Proc. of USENIX NSDI*, Boston, MA, USA, Mar. 2011, pp. 24–37.

[9] T. Bonald and J. Roberts, "Multi-resource fairness: Objectives, algorithms and performance," in *Proc. of ACM SIGMETRICS*, 2015, pp. 31–42.

[10] C. Joe-Wong, S. Sen, T. Lan, and M. Chiang, "Multiresource allocation: Fairness-efficiency tradeoffs in a unifying framework," *IEEE/ACM Trans. on Networking*, vol. 21, no. 6, pp. 1785–1798, Dec. 2013.

[11] M. Shin, S. Chong, and I. Rhee, "Dual-resource TCP/AQM for processing-constrained networks," *IEEE/ACM Trans. on Networking*, vol. 16, no. 2, pp. 435–449, Apr. 2008.

[12] M. Obadia, J. L. Rougier, L. Iannone, V. Conan, and M. Brouet, "Revisiting nfv orchestration with routing games," in *Proc. of IEEE NFV-SDN*, Palo Alto, CA, USA, Nov 2016, pp. 107–113.

[13] J. Kwak, Y. Kim, J. Lee, and S. Chong, "DREAM: Dynamic resource and task allocation for energy minimization in mobile cloud systems," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 12, pp. 2510–2523, Dec. 2015.

[14] J. Zhao, H. Li, C. Wu, Z. Li, Z. Zhang, and F. Lau, "Dynamic pricing and profit maximization for the cloud with geo-distributed data centers," in *Proc. of IEEE INFOCOM*, Toronto, Canada, Apr. 2014, pp. 118–126.

[15] S. H. Low and D. E. Lapsley, "Optimization flow control. I. basic algorithm and convergence," *IEEE/ACM Trans. on Networking*, vol. 7, no. 6, pp. 861–874, Dec. 1999.

[16] W.-H. Wang, M. Palaniswami, and S. H. Low, "Optimal flow control and routing in multi-path networks," *Performance Evaluation*, vol. 52, no. 2–3, pp. 119 – 132, 2003.

[17] L. Chen, S. H. Low, M. Chiang, and J. C. Doyle, "Cross-layer congestion control, routing and scheduling design in ad hoc wireless networks," in *Proc. of IEEE INFOCOM*, Barcelona, Catalunya, Spain, Apr. 2006, pp. 1–13.

[18] M. Yu, Y. Yi, J. Rexford, and M. Chiang, "Rethinking virtual network embedding: substrate support for path splitting and migration," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 17–29, 2008.

[19] X. Lin and N. B. Shroff, "Utility maximization for communication networks with multipath routing," *IEEE Trans. on Automatic Control*, vol. 51, no. 5, pp. 766–781, May 2006.

[20] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[21] D. P. Bertsekas, *Nonlinear programming*. Athena scientific Belmont, 1999.

[22] Y. Kim, H.-W. Lee, and S. Chong, "Control of multi-resource infrastructures: Application to nfv and computation offloading," *Technical Report*. [Online]. Available: https://www.dropbox.com/s/3qkdjpxiqrlynh3/Unified_tech_report.pdf?dl=0

[23] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 23.

[24] F. Facchinei and J.-S. Pang, *Finite-dimensional variational inequalities and complementarity problems*. Springer Science & Business Media, 2007.

[25] "Internet topology dataset: The Internet Topology Zoo." [Online]. Available: http://www.topology-zoo.org/index.html

[26] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks*. Prentice-hall Englewood Cliffs, NJ, 1987, vol. 2.