Optimal Dynamic Cloud Network Control

Hao Feng*, Jaime Llorca†, Antonia M. Tulino†, Andreas F. Molisch*
*University of Southern California, Email: {haofeng, molisch}@usc.edu
†Bell Labs, Alcatel-Lucent, Email: {jaime.llorca, a.tulino}@alcatel-lucent.com

***Abstract*—Distributed cloud networking enables the deployment of network services in the form of interconnected virtual network functions instantiated over general purpose hardware at multiple cloud locations distributed across the network. The network service distribution problem has been considered as a static global optimization problem to find the placement of virtual functions and the routing of network flows that meet a given set of demands with minimum cost. In this paper, we address the design of distributed online solutions that drive local routing, processing, and resource allocation decisions while providing global objective guarantees. We present a distributed joint transmission-processing flow scheduling and resource allocation algorithm that stabilizes the underlying cloud network queuing system, while achieving arbitrarily close to minimum average network cost, with a tradeoff in network delay. We further enhance our algorithm with a *shortest transmission-plus-processing distance bias* that improves the delay performance without compromising throughput nor overall cloud network cost. We provide simulation results that confirm our theoretical analysis, illustrate the effect of the shortest transmission-plus-processing distance bias, and demonstrate remarkably good convergence to the optimal cloud network configuration.**

Index Terms — distributed cloud networking, virtual network function, distributed optimization, dynamic control

I. Introduction

Distributed cloud networking builds on network functions virtualization (NFV) and software defined networking (SDN) to enable the deployment of network services in the form of elastic virtual network functions instantiated over commercial off the shelf (COTS) servers at multiple cloud locations and interconnected via a programmable network fabric [1]-[3]. In this evolved virtualized environment, network operators can host a variety of highly adaptable services over a common physical infrastructure, reducing both capital and operational expenses, while providing quality of service guarantees. While this approach is very attractive for network providers, it poses several technical challenges. Chief among them is how to efficiently assign network functions to the various servers in the network. These placement decisions must be coordinated with the routing of network flows through the appropriate network functions, and with resource allocation decisions that determine the amount of resources (*e.g.,* virtual machines) allocated to each function.

The problem of placing virtual network functions in distributed cloud networks was first addressed in [4]. The problem is formulated as a generalization of Generalized Assignment (GA) and Facility Location (FA), and a $(O(1), O(1))$ bi-criteria approximation with respect to both overall cost and capacity constraints is provided. Shortly after, [5] introduced the cloud service distribution problem (CSDP), where the goal is to find the placement of network functions and the routing of network flows that minimize the overall cloud network cost. The CSDP is formulated as a minimum cost network flow problem, in which flows consume both network and cloud resources as they go through the required virtual functions. The CSDP is shown to admit polynomial-time solutions under linear costs and fractional flows. However, both of these works have two main limitations:

- They consider a *static* scenario with a priori known demands. However, with the increasing heterogeneity and dynamics inherent to both service demands and the underlying cloud network, we argue that online algorithms that enable rapid adaptation to changes in network conditions and service demands are essential.
- They consider a centralized optimization. However, the complexity associated with the resulting global optimization problem and the need to have global knowledge of the service demands, limits the use of centralized algorithms, specially in large-scale distributed cloud networks and under time-varying demands.

Our previous work in [6] is the first to address the service distribution problem in a dynamic cloud network setting. The proposed dynamic cloud network control (DCNC) algorithm is based on augmenting the Lyapunov drift-plus-penalty control method [7]-[9], which had only been used in *traditional* (transmission) networks, to account for both transmission and processing flows, consuming network and cloud resources. However, several issues remain open for improvement: *i)* the analysis in [6] considers only expected time-averaged performance guarantees, and *ii)* the resulting network delay can be significant, especially in lightly loaded networks.

This paper extends the work in [6] with the following contributions: *i)* we provide the bounds for time average cost and time average occupancy (total queue backlog) with probability 1 (instead of in expected time average); *ii)* we design a key *shortest transmission-plus-processing distance bias* extension to the DCNC algorithm, which is shown to significantly reduce network delay without compromising throughput nor overall cloud network cost; and *iii)* we present simulation results that illustrate the effect of incorporating the shortest transmission-plus-processing distance bias into the DCNC algorithm, as well as its efficiency in reducing overall cloud network cost and delay for different parameter settings and network scenarios that include up to 110 clients.

The rest of the paper is organized as follows. Sec. II intro-

duces the model and problem formulation. Sec. III describes the proposed algorithm, including the shortest transmission-plus-processing distance bias extension. Simulation results are presented in Sec. IV. We summarize the main conclusions in Sec. V.

II. Model and Problem Formulation

A. Cloud network model

We consider a cloud network modeled as a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = N$ vertices and $|\mathcal{E}| = E$ edges representing the set of network nodes and links, respectively. In the context of a cloud network, a node represents a distributed cloud location, in which virtual network functions (VNFs) can be instantiated in the the form of virtual machines (VMs) over COTS servers, while an edge represents a logical link (*e.g.,* IP link) between two cloud locations. We denote by $\delta(i) \in \mathcal{V}$ the set of neighbor nodes of $i \in \mathcal{V}$ in $\mathcal{G}$.

Cloud and network resources are characterized by their processing and transmission capacity and cost, respectively. In particular, we define:

- $\mathcal{K}_i = \{0, 1, \cdots, K_i\}$: the set of possible processing resource units at node i
- $\mathcal{K}_{ij} = \{0, 1, \cdots, K_{ij}\}$: the set of possible transmission resource units at link (i, j)
- $C_{i,k}$: the capacity, in processing flow units, resulting from the allocation of k resource units (*e.g.,* VMs) at node i
- $C_{ij,k}$: the capacity, in transmission flow units, resulting from the allocation of k resource units (*e.g.,* 1G links) at link (i, j)
- $w_{i,k}$: the cost of setting up k resource units at node i
- $w_{ij,k}$: the cost of setting up k resource units at link (i, j)
- e_i: the cost per processing flow unit at node i
- e_{ij}: the cost per transmission flow unit at link (i, j)

B. Network service model

A network service $\phi \in \Phi$ is described by a chain of VNFs. We denote by $\mathcal{M}_\phi = \{1, 2, \cdots, M_\phi\}$ the ordered set of VNFs of service ϕ. Hence, the tuple (ϕ, m), with $\phi \in \Phi$ and $m \in \mathcal{M}_\phi$, identifies the m-th function of service ϕ. We refer to a client as a source-destination pair (s, d), with $s, d \in \mathcal{V}$. A client requesting network service $\phi \in \Phi$ implies the request for the network flows originating at source node s to go through the sequence of VNFs specified by $\mathcal{M}_\phi$ before exiting the network at destination node d.

Each VNF has (possibly) different processing requirements, which may also vary among cloud locations. We denote by $r^{(\phi,m)}$ the processing-transmission flow ratio of VNF (ϕ, m). That is, when one transmission flow unit goes through VNF (ϕ, m), it occupies $r^{(\phi,m)}$ processing resource units. In addition, our service model also captures the possibility of flow scaling. We denote by $\xi^{(\phi,m)} > 0$ the scaling factor of VNF (ϕ, m). That is, the size of the output flow of VNF (ϕ, m) is $\xi^{(\phi,m)}$ times larger than its input flow. We refer to a VNF with $\xi^{(\phi,m)} > 1$ as an expansion function, and to a VNF with $\xi^{(\phi,m)} < 1$ as a compression function. Moreover, a processing delay $D_i^{(\phi,m)}$ (in time units) is incurred in executing VNF (ϕ, m) at node i, as long as the processing flow satisfies the node capacity constraint.

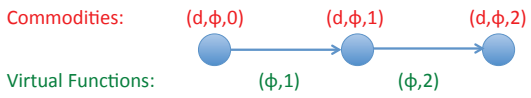


Fig. 1. A network service chain $\phi \in \Phi$ composed of $M_\phi = 2$ functions. Service ϕ takes source commodity $(d, \phi, 0)$ and delivers final commodity $(d, \phi, 2)$ after going through the sequence of functions $\{(\phi, 1), (\phi, 2)\}$. VNF (ϕ, m) takes commodity $(d, \phi, m-1)$ and generates commodity (d, ϕ, m).

We remark that our service model applies to a wide range of cloud services that go beyond NFV services, and that includes, for example, Internet of Things (IoT) services, expected to largely benefit from the proximity and elasticity of distributed cloud networks [10].

As in [6], we adopt a *multi-commodity-chain* flow model, in which a commodity represents a network flow at a given stage of a service chain. We use the triplet (d, ϕ, m) to identify a commodity flow that is output of the m-th function of service ϕ for client d. The source commodity of service ϕ for client d is identified by $(d, \phi, 0)$ and the final commodity delivered to d by (d, ϕ, M_ϕ), as illustrated in Fig. 1.

C. Queuing Model

We consider a time slotted system with slots normalized to integral units $t \in \{0, 1, 2, \cdots\}$. We denote by $a_i^{(d,\phi,m)}(t)$ the exogenous arrival rate of commodity (d, ϕ, m) at node i during timeslot t, and by $\lambda_i^{(d,\phi,m)}$ the expected value of $a_i^{(d,\phi,m)}(t)$, referred to as average input rate. We assume that $a_i^{(d,\phi,m)}(t)$ is independently and identically distributed (i.i.d.) across timeslots.

At each timeslot t, every node makes a transmission and processing decision on all of its output interfaces. We use $\mu_{ij}^{(d,\phi,m)}(t)$ to denote the assigned flow rate at link (i, j) for commodity (d, ϕ, m) at time t, $\mu_{i,pr}^{(d,\phi,m)}(t)$ to denote the assigned flow rate from node i to its processing unit for commodity (d, ϕ, m) at time t, and $\mu_{pr,i}^{(d,\phi,m)}(t)$ to denote the assigned flow rate from node i's processing unit to node i for commodity (d, ϕ, m) at time t.

During network evolution, internal network queues buffer packets according to their commodities. We define the *queue backlog* of commodity (d, ϕ, m) at node i, $Q_i^{(d,\phi,m)}(t)$, as the amount of commodity (d, ϕ, m) in the queue of node i at the beginning of timeslot t. The $Q_i^{(d,\phi,m)}(t)$ process evolves according to the following queuing dynamics:[1]

$$Q_i^{(d,\phi,m)}(t+1) \leq \left[Q_i^{(d,\phi,m)}(t) - \sum_{j \in \delta(i)} \mu_{ij}^{(d,\phi,m)}(t) - \mu_{i,pr}^{(d,\phi,m)}(t) \right]^+ + \sum_{j \in \delta(i)} \mu_{ji}^{(d,\phi,m)}(t) + \mu_{pr,i}^{(d,\phi,m)}(t) + a_i^{(d,\phi,m)}(t). \quad (1)$$

[1]Throughout the paper, we use $[x]^+$ to denote $\max\{x, 0\}$.

In addition, at each timeslot t, cloud network nodes can also make resource allocation decisions. We denote by $y_{ij,k}(t)$ the binary variable indicating the allocation of k transmission resource units at link (i,j) in timeslot t, and by $y_{i,k}(t)$ the binary variable indicating the allocation of k processing resource units at node i in timeslot t.

D. Problem Formulation

The goal is to design a control algorithm that, given exogenous arrival rates with average input rate matrix $\boldsymbol{\lambda} = (\lambda_i^{(d,\phi,m)})$, supports all service demands while minimizing the average cloud network cost. Specifically, we require the cloud network to be rate stable (see Ref. [7]), *i.e.,*

$$\lim_{t\to\infty} \frac{Q_i^{(d,\phi,m)}(t)}{t} = 0 \quad \text{with prob. } 1 \quad \forall i,(d,\phi,m). \tag{2}$$

The dynamic service distribution problem (DSDP) can then be formulated as follows:

$$\min \quad \limsup_{t\to\infty} \quad \frac{1}{t}\sum_{\tau=0}^{t-1} \mathbb{E}\left\{h(\tau)\right\} \tag{3a}$$

$$\text{s.t.} \quad \text{The cloud network is rate stable with input rate } \boldsymbol{\lambda} \tag{3b}$$

$$\mu_{pr,i}^{(d,\phi,m)}(\tau) = \xi^{(\phi,m)}\mu_{i,pr}^{(d,\phi,m-1)}(\tau - D_i^{(\phi,m)}) \quad \forall i,d,\phi,m>0,\tau \tag{3c}$$

$$\sum_{(d,\phi,m>0)} \mu_{i,pr}^{(d,\phi,m-1)}(\tau) r^{(\phi,m)} \le \mu_i(\tau) \le \sum_{k\in\mathcal{K}_i} C_{i,k}\, y_{i,k}(\tau) \quad \forall i,\tau \tag{3d}$$

$$\sum_{(d,\phi,m)} \mu_{ij}^{(d,\phi,m)}(\tau) \le \mu_{ij}(\tau) \le \sum_{k\in\mathcal{K}_{ij}} C_{ij,k}\, y_{ij,k}(\tau) \quad \forall (i,j),\tau \tag{3e}$$

$$\mu_{i,pr}^{(d,\phi,m)}(\tau), \mu_{pr,i}^{(d,\phi,m)}(\tau), \mu_{ij}^{(d,\phi,m)}(\tau) \in \mathbb{R}^+ \quad \forall i,(i,j),d,\phi,m,\tau \tag{3f}$$

$$y_{i,k}(\tau), y_{ij,k}(\tau) \in \{0,1\} \quad \forall i,(i,j),d,\phi,m,\tau \tag{3g}$$

where (3c) describes the instantaneous commodity-chain constraints, (3d) and (3e) are instantaneous transmission and processing capacity constrains, and the cost function $h(\tau)$ is given by

$$h(\tau) = \sum_{i\in\mathcal{V}}\left(e_i\,\mu_i(\tau) + \sum_{k\in\mathcal{K}_i} w_{i,k}\, y_{i,k}(\tau)\right) + \sum_{(i,j)\in\mathcal{E}}\left(e_{ij}\,\mu_{ij}(\tau) + \sum_{k\in\mathcal{K}_{ij}} w_{ij,k}\, y_{ij,k}(\tau)\right). \tag{4}$$

In the following section, we present a dynamic control algorithm that obtains arbitrarily close to optimal solutions to (3) in a fully distributed fashion.

III. Dynamic cloud network control

In this section, we first describe a distributed dynamic cloud network control (DCNC) strategy that extends the Lyapunov drift-plus-penalty algorithm to account for both transmission and processing flow scheduling and resource allocation decisions, first introduced in [6]. We then extend the analysis of [6] to show that DCNC provides arbitrarily close-to-optimal solutions with probability 1. Finally, we present E-DCNC, an enhanced dynamic cloud network control algorithm that introduces a shortest transmission-plus-processing distance bias to reduce network delay without compromising throughput or average cloud network cost.

A. DCNC algorithm

Local transmission decisions: At the beginning of each timeslot t, each node i observes the queue backlogs of all its neighbors and performs the following operations for each of its outgoing links (i,j), $j\in\delta(i)$:

1) For each commodity (d,ϕ,m), compute the *transmission utility* weight
$$W_{ij}^{(d,\phi,m)}(t) = \left[Q_i^{(d,\phi,m)}(t) - Q_j^{(d,\phi,m)}(t) - Ve_{ij}\right]^+$$
where V is a non-negative control parameter that determines the degree to which cost minimization is emphasized.
2) Compute the optimal commodity $(d,\phi,m)^*$ as:
$$(d,\phi,m)^* = \arg\max_{(d,\phi,m)}\left\{W_{ij}^{(d,\phi,m)}(t)\right\}$$
3) If $W_{ij}^{(d,\phi,m)^*}(t) = 0$, then, $k^* = 0$. Otherwise,
$$k^* = \arg\max_{k}\left\{C_{ij,k}W_{ij}^{(d,\phi,m)^*}(t) - Vw_{ij,k}\right\}$$
4) Take the following resource allocation and flow rate assignment decisions:
$$\begin{aligned} &y_{ij,k^*}(t) = 1\\ &y_{ij,k}(t) = 0 \quad \forall k\neq k^*\\ &\mu_{ij}^{(d,\phi,m)^*}(t) = C_{ij,k^*}\\ &\mu_{ij}^{(d,\phi,m)}(t) = 0 \quad \forall (d,\phi,m)\neq(d,\phi,m)^* \end{aligned}$$

Local processing decisions: At the beginning of each timeslot t, each node i observes its local queue backlogs and performs the following operations:

1) For each commodity (d,ϕ,m), compute the *processing utility* weight
$$W_i^{(d,\phi,m)}(t) = \frac{1}{r^{(\phi,m+1)}}\left[Q_i^{(d,\phi,m)}(t) - \xi^{(\phi,m+1)}Q_i^{(d,\phi,m+1)}(t) - Ve_i\right]^+$$
The *processing utility* weight $W_i^{(d,\phi,m)}(t)$ indicates the benefit of executing function $(\phi,m+1)$ to process commodity (d,ϕ,m) into commodity $(d,\phi,m+1)$ at time t, in terms of the local backlog reduction per processing unit cost.
2) Compute the optimal commodity $(d,\phi,m)^*$ as:
$$(d,\phi,m)^* = \arg\max_{(d,\phi,m)}\left\{W_i^{(d,\phi,m)}(t)\right\}$$

3) If $W_i^{(d,\phi,m)^*}(t) = 0$, then, $k^* = 0$. Otherwise,

$$k^* = \arg\max_k \left\{ C_{i,k} W_i^{(d,\phi,m)^*}(t) - V w_{i,k} \right\}$$

4) Take the following resource allocation and flow rate assignment decisions:

$$\begin{aligned} & y_{i,k^*}(t) = 1 \\ & y_{i,k}(t) = 0 \quad \forall k \neq k^* \\ & \mu_{i,pr}^{(d,\phi,m)^*}(t) = \frac{1}{r^{(\phi,m+1)^*}} C_{i,k^*} \\ & \mu_{i,pr}^{(d,\phi,m)}(t) = 0 \quad \forall (d,\phi,m) \neq (d,\phi,m)^* \end{aligned}$$

Observe from the above algorithm description that the finite processing delay $D_i^{(\phi,m)}$ is not involved in the implementation of DCNC. The reason is that omitting $D_i^{(\phi,m)}$ in the scheduling decisions of DCNC does not affect the throughput optimality or the average cost convergence. This is shown in the proof of Theorem 1, which is given in [12] due to space limitations.

B. Performance Analysis

In this section, we extend the analysis of [6] to show that the DCNC algorithm achieves the average cost-delay tradeoff $[O(1/V), O(V)]$ with probability 1 (w.p.1). To this end, letting Λ denote the *cloud network capacity region*, whose characterization is described in [6, Theorem 1], the following theorem follows:

Theorem 1. *If the average rate matrix $\boldsymbol{\lambda} = (\lambda_i^{(d,\phi,m)})$ is strictly interior to the capacity region Λ, and $\xi^{(\phi,m)}$, $r^{(\phi,m)}$, $D_i^{(\phi,m)}$, and $\sum_{(d,\phi,m)} \mathbb{E}[(a_i^{(d,\phi,m)}(t))^4]$ are bounded, then the DCNC algorithm stabilizes the cloud network, while achieving arbitrarily close to minimum average cost $h^*(\boldsymbol{\lambda})$ w.p.1, i.e.,*

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau=0}^{t} h(\tau) \leq \overline{h}^*(\boldsymbol{\lambda}) + \frac{NB}{V}, \quad (w.p.1) \tag{5}$$

$$\limsup_{t\to\infty} \frac{1}{t} \sum_{\tau,i,d,\phi,m} Q_i^{(d,\phi,m)}(\tau) \leq \frac{NB + V[\overline{h}^*(\boldsymbol{\lambda}+\epsilon\mathbf{1}) - \overline{h}^*(\boldsymbol{\lambda})]}{\epsilon} \quad (w.p.1) \tag{6}$$

where B is a constant depending on the system parameters $C_{ij,K_{ij}}$, C_{i,K_i}, $A_{\max}$, $\xi^{(\phi,m)}$, and $r^{(\phi,m)}$; ε is a positive constant satisfying $(\boldsymbol{\lambda} + \varepsilon) \in \Lambda$; and, with an slight abuse of notation, $\overline{h}^*(\boldsymbol{\lambda})$ denotes the minimum average cost of the DSDP formulated in (3)*.*

□

Note that the parameter V drives the average cost arbitrarily close to the minimum cost $\overline{h}^*(\boldsymbol{\lambda})$ with a corresponding linear increase in average network congestion (average delay).

Proof: The proof of Theorem 1 is given in [12] and ommitted here due to space limitations. ■

C. Delay Improvement via Enhanced Dynamic Cloud Network Control (E-DCNC)

The DCNC algorithm determines the routes and the service distribution according to the evolving backlog accumulation in the network. However, queue backlogs have to build up in the appropriate direction before yielding efficient routes and service distribution, which can result in degraded delay performance.

The delay performance of multi-hop queuing networks can be improved by introducing a bias term into the weight of the dynamic control algorithm [9][11], where the bias term represents the number of hops or the geometric distance to the destination. Control decisions are then made based on the joint observation of the backlog state and the bias term. In order to leverage this technical track to reduce end-to-end delay in cloud networks, the bias term needs to capture the effect of both transmission and processing delay. Accordingly, we propose E-DCNC, and enhanced DCNC algorithm designed to reduce cloud network delay. Specifically, for each queue backlog $Q_i^{(d,\phi,m)}(t)$, we define a modified queue backlog $\tilde{Q}_i^{(d,\phi,m)}(t)$:

$$\tilde{Q}_i^{(d,\phi,m)}(t) \triangleq Q_i^{(d,\phi,m)}(t) + \eta Y_i^{(d,\phi,m)}, \tag{7}$$

where $Y_i^{(d,\phi,m)}$ denotes the *shortest transmission-plus-processing distance bias* term, and η is a control parameter representing the degree to which we emphasize the bias with respect to the backlog. Furthermore, we define

$$Y_i^{(d,\phi,m)} \triangleq \begin{cases} \min\limits_{j\in\mathcal{V}} \left\{ H_{ij} + D_j^{(\phi,m)} \right\}, & 0 \leq m < M_\phi \\ H_{id}\,, & m = M_\phi \end{cases}, \tag{8}$$

where H_{ij} represents the number of hops from node i to node j along the shortest path.

E-DCNC is formed by using $\tilde{Q}_i^{(d,\phi,m)}(t)$ in place of $Q_i^{(d,\phi,m)}(t)$ in the DCNC algorithm, and modifying the condition for choosing $k^* = 0$ to be as follows (see step 3 of DCNC algorithm description):

- For local transmission decisions: $\tilde{W}_{ij}^{(d,\phi,m)^*}(t) = 0$ or $Q_i^{(d,\phi,m)}(t) = 0$;
- For local processing decisions: $\tilde{W}_i^{(d,\phi,m)^*}(t) = 0$ or $Q_i^{(d,\phi,m)}(t) = 0$.

The motivation of changing the condition for setting $k^* = 0$ is to avoid unnecessary resource consumption when $\tilde{W}_{ij}^{(d,\phi,m)^*}(t)$ or $\tilde{W}_i^{(d,\phi,m)^*}(t)$ are positive, but the queue of commodity $(d,\phi,m)^*$ is empty.

It can be shown that the throughput optimality and average resource cost efficiency of E-DCNC can also be guaranteed. Moreover, as shown by the simulation experiments presented in the following section, a significantly lower congestion level is achieved under E-DCNC, demonstrating enhanced delay performance, particularly when the network is lightly loaded, without compromising overall throughput nor cloud network cost. In fact, note that with the bias term defined in (8) and under light traffic load, for commodities that require further processing, $0 \leq m < M_\phi$, flows tend to be routed along the path with the smallest combined transmission-processing delay, while for final commodities, $m = M_\phi$, data flows follow the shortest path to their corresponding destination node d.

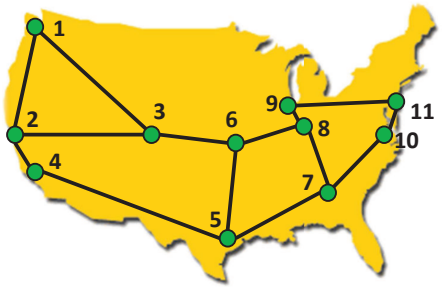


Fig. 2. Abilene US Continental Network. Nodes are indexed as: 1) Seattle, 2) Sunnyvale, 3) Denver, 4) Los Angeles, 5) Houston, 6) Kansas City, 7) Atlanta, 8) Indianapolis, 9) Chicago, 10) Washington, 11) New York.

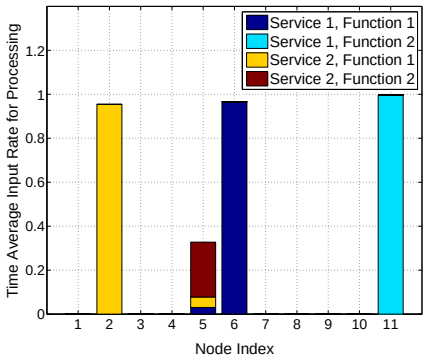


Fig. 3. Processing flow rate distribution over all cloud nodes in small service scale scenario.

IV. Simulation Experiments

In this section, we present numerical results obtained from simulating the DCNC and E-DCNC algorithms during 10^6 timeslots.

We assume a cloud network based on the Abilene US continental network, shown in Fig. 2. All 14 links exhibit homogeneous transmission capacities and costs, and all 11 nodes represent cloud locations with homogeneous processing capacity choices. In particular, two service scenarios on the Abilene network are simulated: a small scale scenario of 2 services with 2 source-destination pairs (or clients); a large scale scenario with 1 service and 110 source-destination pairs.

The common simulation settings for the two scenarios are as follows. All links have two capacity choices $C_{ij,0}$ and $C_{ij,1}$, with corresponding resource allocation costs $w_{ij,0} = 0$ and $w_{ij,1} = 1$, and load-dependent costs $e_{ij} = 1$. All nodes have two processing capacity choices $C_{i,0}$ and $C_{i,1}$, and resource allocation costs, $w_{i,0} = 0$ and $w_{i,1} = 1$. However, cloud nodes have different load-dependent processing costs. In particular, we assume Kansas City (node 5) and Houston (node 6) have cheaper processing costs, $e_5 = e_6 = 1$, while for any other node $e_i = 3$. Every node in the network has the ability to implement all the functions of all the services.

A. The Small Scale Service Scenario

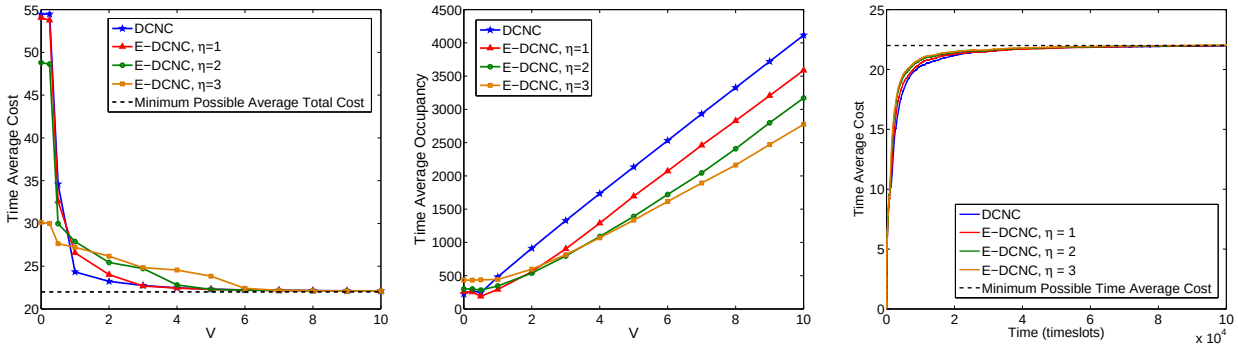
In this scenario, we have $C_{ij,0} = 0$, $C_{ij,1} = 1$, $C_{i,0} = 0$, $C_{i,1} = 1$, and consider 2 services, each composed of 2 virtual functions. All 4 functions have the same complexity, given by a processing-transport flow ratio of 1 and a processing delay of 10 timeslots. In terms of flow scaling, the first and second functions of Service 1 have a scaling factor of 1 and 3, respectively. That is, the second function of Service 1 is an expansion function. For Service 2, the first and second functions have a scaling factor of 0.25 and 1, respectively. That is, the first function of Service 1 is a compression function. We assume one source-destination pair for Service 1, with source in Seattle (node 1) and destination in New York (node 11), and another for Service 2, with source in Sunnyvale (node 2) and destination in Atlanta (node 7). Both source nodes receive exogenous arrivals with rate satisfying i.i.d. Poisson distribution across timeslots with mean value 1.

Fig. 3 shows the processing flow rate distribution for the four functions across the cloud network nodes under DCNC. Observe how VNF $(1, 1)$ (first function of service 1) is mostly implemented in Kansas City (node 6), which is the node with lowest processing cost along the shortest path from Seattle (Node 1) to New York (Node 11). Note, however, that the expansion function VNF $(1, 2)$ is largely implemented in New York (node 11), which is the final destination of Service 1, in order to minimize the transmission cost impact of the larger final commodity $(11, 1, 2)$ resulting from the execution of VNF $(1, 2)$. For Service 2, the first function, VNF $(2, 1)$, is a compression function with scaling factor 0.25. As expected, VNF $(2, 1)$ is mostly implemented at the source node (node 2), in order to reduce the transmission cost of Service 2 by compressing the source commodity $(7, 2, 0)$ into commodity $(7, 2, 1)$ before commodity $(7, 2, 0)$ even flows into the network. The processing of commodity $(7, 2, 1)$ by VNF $(2, 2)$ to generate commodity $(7, 2, 2)$ concentrates at node 5 because this node has the lowest processing cost among the nodes on the shortest path from Sunnyvale (node 2) to Atlanta (node 7). Note that the average flow rate of commodity $(7, 2, 1)$ is approximately 0.25 due to the effect of the compression function $(2, 1)$.

With the simulated average demand rate of 1 for both services, the minimum average cloud network cost can be computed by inspection. Specifically, the optimal flow paths for the two services are the two respective shortest paths. The optimal function placement are node 6 for VNF $(1, 1)$, node 11 for VNF $(1, 2)$, node 2 for VNF $(2, 1)$, and node 5 for VNF $(2, 2)$. The resulting average cost is 22.

Figs. 4a and 4b demonstrate the tradeoff between the time average cost and the time average occupancy (time average total queue backlog) as a function of the control parameter V under both DCNC and E-DCNC. According to Fig. 4a, the time average costs under both DCNC and E-DCNC converge to approximately 22, while the time average occupancies increase linearly with respect to V. These results clearly demonstrate the $[O(1/V), O(V)]$ cost-delay tradeoff as suggested by the performance bounds of Theorem 1.

On the other hand, comparing the performance of DCNC



(a) (b) (c)

Fig. 4. Network service distribution in small-scale service scenario. a) Time average cost v.s. control parameter V; b) Time average total occupancy vs. control parameter V; c) Time average cost evolution over time.

and E-DCNC with different η values, it can be seen from Fig. 4b that E-DCNC exhibits significantly lower average occupancy than DCNC, demonstrating a better delay performance. Note that the average occupancy further reduces as η is increased from 1 to 3. Note also, from Fig. 4a, that the larger the η value, the larger the control parameter V has to be to approximate the minimum average cloud network cost.

Fig. 4c exhibits the evolution of the average cloud network cost over time under DCNC and E-DCNC with different η values, when $V = 7$. Note that the converging speed of E-DCNC is no lower than the converging speed of DCNC.

The delay performances of DCNC and E-DCNC can also be compared by simulating the average occupancy evolving over different exogenous average input rate within the network capacity region. As shown in Fig. 5, note that, while maintaining the same average input rate for the two source commodities, $(11, 1, 1)$ and $(7, 2, 1)$, as this common average input rate increases up to approximately 1.34, the average total queue backlog under DCNC and E-DCNC with different η values exhibit sharp increases. This indicates that the average input rate has reached the boundary of the network capacity region.We can also see that the average occupancy under E-DCNC is lower than under DCNC and further decreases as the value of η increases, as long as the average input rate does not exceed the boundary of the network capacity region. This result is consistent with the result shown in Fig. 4b.

B. The Large Scale Service Scenario

In this scenario, we conduct simulations of DCNC and E-DCNC for a single service, but with 110 clients, *i.e.*, all the possible source-destination pairs in the Abilene network. The service has 1 virtual function with processing-transport flow ratio of 1, processing delay of 10 timeslots, and scaling factor of 1. The source node s of each client or source-destination pari (s, d) receives exogenous arrivals of commodity $(1, 1, d)$ according to an i.i.d. (across timeslots) Poisson distribution with mean rate 1. We consider two cases in terms of available capacities: Case 1: $C_{ij,0} = 0$, $C_{ij,1} = 110$, $C_{i,0} = 0$ and $C_{i,1} = 110$; and Case 2: $C_{ij,0} = 0$, $C_{ij,1} = 30$, $C_{i,0} = 0$ and $C_{i,1} = 30$.

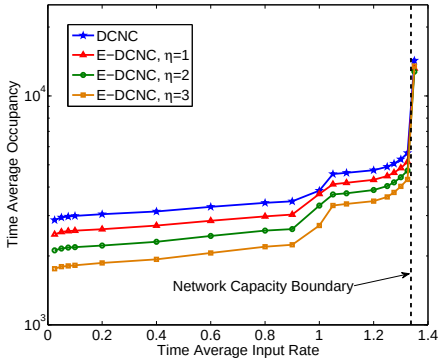


Fig. 5. Average occupancy in small scale service scenario under DCNC and E-DCNC with varying exogenous average input rate and $V = 10$.

Figs. 6 and 7 show the evolution of the average cost and the average occupancy, respectively, with respect to the control parameter V, under DCNC for both Case 1 and Case 2, and under E-DCNC for Case 1. Similarly to the small scale service scenario, the results demonstrate the $[O(1/V), O(V)]$ cost-delay tradeoff. In addition, Fig. 7 shows that the occupancy level in Case 1 is generally lower than that in Case 2, which is intuitive due to the fact that the resource unit capacity (both for transmission and processing) in Case 1 (=110) is larger than that in Case 2 (=30). Focusing on Case 1, the scenario with largest maximum capacity (=110), Fig. 7 shows how the shortest transmission-plus-processing distance bias allows E-DCNC to significantly reduce the average occupancy (and hence average delay) compared to DCNC. Also, note from Fig. 6 that the difference in convergence speed of the average cost with respect to V between DCNC and E-DCNC is almost negligible. Therefore, Figs. 6 and 7 indicate that the delay performance gain of E-DCNC is nearly "free", which is due to the fact that the network in Case 1 is lightly loaded.

Figs. 8(a) and 8(b) show the processing rate distribution among all the nodes under DCNC in Case 1 and Case 2, respectively. In both cases, the processing load in Kansas city (node 5) and Houston (node 6) is much higher than the processing load in the rest of the nodes. This is because the

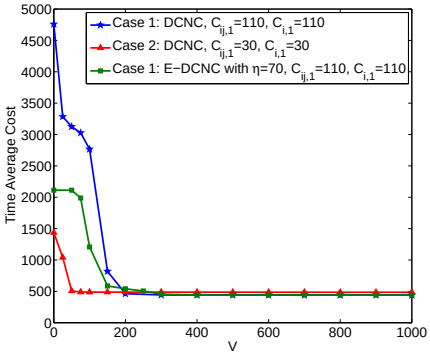


Fig. 6. Network service distribution in large-scale service scenario: Time average cost v.s. control parameter V.

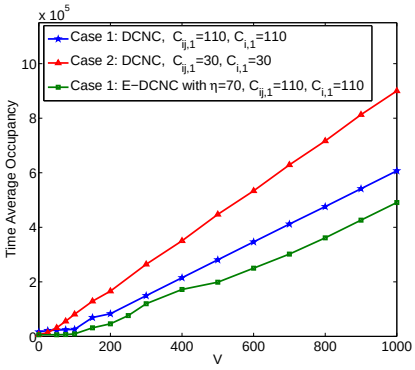


Fig. 7. Network service distribution in large-scale service scenario: Time average occupancy vs. control parameter V.

processing cost at node 5 and 6 ($e_5 = e_6 = 1$) is cheaper than the processing costs at the other nodes ($e_i = 3$). As a result, the majority of the processing flow has the tendency to concentrate at nodes 5 and 6. In addition, it is interesting to note that the processing flow rate in Case 1 concentrates even more on nodes 5 and 6 than in Case 2. This is because of the link capacity difference between the two cases. Specifically, in Case 1, the maximum capacity of each link is large enough such that, given the average exogenous input rates in this simulation setting, no link will be a bottleneck for the delivery of data to node 5 or 6, and therefore the processing tends to concentrate at nodes 5 and 6. In contrast, in Case 2, since the capacity of each link is only 30, the delivery of data suffers from the bottleneck effect imposed by certain links. As a result, the processing in Case 2 "spreads out" to other nodes and concentrates less at nodes 5 and 6.

V. Conclusion

We address the network service distribution problem in a dynamic cloud network setting, in which demands are unknown and time varying. We extend the *Lyapunov drift-plus-penalty* network control algorithm to account for both transmission and processing of network flows and the corresponding allocation of network and cloud resources. We present a distributed

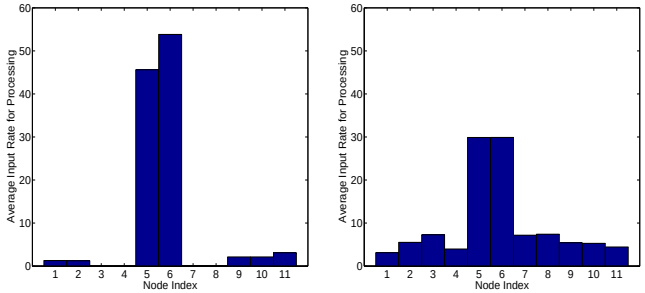


Fig. 8. Processing flow rate distribution over all cloud nodes in large-scale service scenario: a) $C_{ij,1} = 110,\ C_{i,1} = 110$; b) $C_{ij,1} = 30,\ C_{i,1} = 30$.

joint flow scheduling and resource allocation algorithm that stabilizes the underlying queuing system within this region, while achieving arbitrarily close to minimum average cloud network cost, with a tradeoff in network delay. We show that the time average cost and occupancy meet the stability constraints with probability 1. We then introduce an appropriate *shortest transmission-plus-processing distance bias* into the dynamic cloud network control (DCNC) algorithm. We show via simulations that the resulting enhanced E-DCNC algorithm significantly reduces network delay, specially in lightly loaded scenarios, without compromising overall cloud network cost. The results presented in this paper can serve as guidelines for the deployment and dynamic orchestration of next generation cloud network services.

Acknowledgements

Part of this work was supported financially by the NSF.

References

[1] Bell Labs Strategic White Paper, "Metro Network Traffic Growth: An Architecture Impact Study," December 2013.
[2] Marcus Weldon, "The Future X Network," *CRC Press*, October 2015.
[3] Alcatel-Lucent Strategic White Paper, "The Programmable Cloud Network - A Primer on SDN and NFV," June 2013.
[4] L. Lewin-Eytan, J. Naor, R. Cohen, D. Raz, "Near Optimal Placement of Virtual Network Functions," *IEEE INFOCOM*, 2015.
[5] M. Barcelo, J. Llorca, A. M. Tulino, N. Raman, "The Cloud Servide Distribution Problem in Distributed Cloud Networks" *IEEE ICC*, 2015.
[6] H. Feng, J. Llorca, A. M. Tulino, A. F. Molisch, "Dynamic service optimization in distributed cloud networks", *IEEE INFOCOM SWFAN Workshop*, April 2016.
[7] M. J. Neely, "Stochastic network optimization with application to communication and queueing systems", *Synthesis Lectures on Communication Networks*, Morgan & Claypool, vol. 3, pp. 1–211, 2010.
[8] M. J. Neely, "Energy optimal control for time-varying wireless networks", *IEEE Transactions on Information Theory*, vol. 52, pp. 2915–2934, July, 2006.
[9] M. J. Neely, "Optimal Backpressure Routing for Wireless Networks with Multi-Receiver Diversity", *2006 40th Annual Conference on Information Sciences and Systems*, vol. 3, pp. 18–25, March, 2006.
[10] S. Nastic, S. Sehic, D.-H. Le, H.-L. Truong, and S. Dustdar, "Provisioning software-defined IoT cloud systems," *Future Internet of Things and Cloud (FiCloud)*, pp. 288–295, 2014.
[11] M. J. Neely, E. Modiano and C. E. Rohrs, "Dynamic power allocation and routing for time-varying wireless networks", *Selected Areas in Communications, IEEE Journal on*, vol. 23, pp. 89-103, Jan., 2005.
[12] H. Feng, J. Llorca, A. M. Tulino, A. F. Molisch, "Optimal Dynamic service optimization in distributed cloud networks", *to be submitted*.