

Optimal Dynamic Cloud Network Control

Hao Feng^{*†}, Jaime Llorca[†], Antonia M. Tulino^{†‡}, Andreas F. Molisch^{*}

^{*}University of Southern California, Email: {haofeng, molisch}@usc.edu

[†]Nokia Bell Labs, Email: {jaime.llorca, a.tulino}@nokia-bell-labs.com

[‡]University of Naples Federico II, Italy. Email: {antoni MARIA.tulino}@unina.it

Abstract—Distributed cloud networking enables the deployment of a wide range of services in the form of interconnected software functions instantiated over general purpose hardware at multiple cloud locations distributed throughout the network. We consider the problem of optimal service delivery over a distributed cloud network, in which nodes are equipped with both communication and computation resources. We address the design of distributed online solutions that drive flow processing and routing decisions, along with the associated allocation of cloud and network resources. For a given set of services, each described by a chain of service functions, we characterize the cloud network capacity region and design a family of dynamic cloud network control (DCNC) algorithms that stabilize the underlying queuing system, while achieving arbitrarily close to minimum cost with a tradeoff in network delay. The proposed DCNC algorithms make local decisions based on the online minimization of linear and quadratic metrics obtained from an upper bound on the Lyapunov drift-plus-penalty of the cloud network queuing system. Minimizing a quadratic vs. a linear metric is shown to improve the cost-delay tradeoff at the expense of increased computational complexity. Our algorithms are further enhanced with a shortest transmission-plus-processing distance bias that improves delay performance without compromising throughput or overall cloud network cost. We provide throughput and cost optimality guarantees, convergence time analysis, and extensive simulations in representative cloud network scenarios.

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 general purpose servers at multiple cloud locations and interconnected via a programmable network fabric [3], [4]. In this evolved virtualized environment, *cloud 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.

Together with the evident opportunities of this attractive scenario, there come a number of technical challenges. Critical among them is deciding where to execute each network function among the various servers in the network. The ample opportunities for running network functions at multiple locations opens an interesting and challenging space for optimization. In addition, placement decisions must be coordinated with routing decisions that steer the network flows to 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 [5]. The authors formulate the problem as a generalization of facility location and generalized assignment, and provide algorithms with bi-criteria approximation guarantees. However, the model in [5] does not capture *service chaining*, where flows are required to go through a given sequence of service functions, nor flow routing optimization. Subsequently, the work in [5] introduced a flow based model that allows optimizing the distribution (function placement and flow routing) of services with arbitrary function relationships (*e.g.*, chaining) over capacitated cloud networks. Cloud services are described via a directed acyclic graph and the function placement and flow routing is determined by solving a minimum cost network flow problem with service chaining constraints.

These studies, however, focus on the design of centralized solutions that assume global knowledge of service demands and network conditions. With the increasing scale, heterogeneity, and dynamics inherent to both service demands and the underlying cloud network system, we argue that proactive centralized solutions must be complemented with distributed online algorithms that enable rapid adaptation to changes in network conditions and service demands, while providing global system objective guarantees.

In this work, we address the service distribution problem in a dynamic cloud network setting, where service demands are unknown and time-varying. We provide the first characterization of a cloud network’s capacity region and design throughput-optimal *dynamic cloud network control* (DCNC) algorithms that drive local transmission, processing, and resource allocation decisions with global performance guarantees. The proposed algorithms are based on applying the *Lyapunov drift-plus-penalty* (LDP) control methodology [7]–[9] to a cloud network queuing system that captures both the transmission and processing of service flows, consuming network and cloud resources. We first propose DCNC-L, a control algorithm based on the minimization of a linear metric extracted from an upper bound of a quadratic LDP function of the underlying queuing system. DCNC-L is a distributed joint flow scheduling and resource allocation algorithm that guarantees overall cloud network stability, while achieving arbitrarily close to the minimum average cost with a tradeoff in network delay. We then design DCNC-Q, an extension of DCNC-L that uses a quadratic metric derived from the same upper bound expression of the LDP function. DCNC-Q preserves the throughput optimality of DCNC-L, and can significantly improve the cost-delay tradeoff at the expense of increased computational complexity. Finally, we show that

Partial results have been presented in conference papers [1], [2]. This work has been supported by NSF grant # 1619129 and CCF grant # 1423140.

network delay can be further reduced by introducing a *shortest transmission-plus-processing distance (STPD) bias* into the optimization metric. The generalizations of DCNC-L and DCNC-Q obtained by introducing the shortest STPD bias are referred to as EDCNC-L and EDCNC-Q, respectively.

Our contributions can be summarized as follows:

- We introduce a queuing system for a general class of *multi-commodity-chain (MCC)* flow problems that include the distribution of network services over cloud networks. In our MCC queuing model, the queue backlog of a given commodity builds up, not only from receiving packets of the same commodity, but *also* from processing packets of the preceding commodity in the service chain.
- For a given set of services, we characterize the capacity region of a cloud network in terms of the set of exogenous input flow rates that can be processed by the required service functions and delivered to the required destinations, while maintaining the overall cloud network stability. Importantly, the cloud network capacity region depends on both the cloud network topology and the service structure.
- We design a family of throughput-optimal DCNC algorithms that *jointly schedule computation and communication resources* for flow processing and transmission without knowledge of service demands. The proposed algorithms allow pushing total resource cost arbitrarily close to minimum with a $[O(\epsilon), O(1/\epsilon)]$ cost-delay trade-off, and they converge to within $O(\epsilon)$ deviation from the optimal solution in time $O(1/\epsilon^2)$.
- Our DCNC algorithms make local decisions via the online minimization of linear and quadratic metrics extracted from an upper bound of the cloud network LDP function. Using a quadratic vs. a linear metric is shown to improve the cost-delay tradeoff at the expense of increased computational complexity. In addition, the use of a STPD bias yields enhanced algorithms that can further reduce average delay without compromising throughput or cost performance.

The rest of the paper is organized as follows. We review related work in Section II. Section III describes the system model and problem formulation. Section IV is devoted to the characterization of the cloud network capacity region. We present the proposed DCNC algorithms in Section V, and analyze their performance in Section VI. Numerical experiments are presented in Section VII, and possible extensions are discussed in Section VIII. Finally, we summarize the main conclusions in Section IX.

II. RELATED WORK

The problem of dynamically adjusting network resources in response to unknown changes in traffic demands has been extensively studied in previous literature in the context of stochastic network optimization. In particular, Lyapunov drift control theory is particularly suitable for studying the stability properties of queuing networks and similar stochastic systems. The first celebrated application of Lyapunov drift control in multi-hop networks is the backpressure (BP) routing algorithm

[10]. The BP algorithm achieves throughput-optimality without ever designing an explicit route or having knowledge of traffic arrival rates, hence being able to adapt time-varying network conditions. By further adding a penalty term (*e.g.*, related to network resource allocation cost) to the Lyapunov drift expression, [7]-[9] developed the Lyapunov drift-plus-penalty control methodology. LDP control preserves the throughput-optimality of the BP algorithm while also minimizing average network cost.

LDP control strategies have shown effective in optimizing traditional multi-hop communication networks (as opposed to computation networks). Different versions of LDP-based algorithms have been developed. Most of them are based on the minimization of a linear metric obtained from an upper bound expression of the queuing system LDP function [7]-[9]. Subsequently, the inclusion of a bias term, indicative of network distance, into this linear metric was shown to reduce network delay (especially in low congested scenarios) [12], [13]. Furthermore, [14] proposed a control algorithm for single-commodity multi-hop networks based on the minimization of a quadratic metric from the LDP upper bound, shown to improve delay performance in the scenarios explored in [14].

In contrast to these prior works, this paper extends the LDP methodology for the dynamic control of network service chains over distributed cloud networks. The proposed family of LDP-based algorithms are suitable for a general class of MCC flow problems that exhibit the following key features: (i) *Flow chaining*: a commodity, representing the flow of packets at a given stage of a service chain, can be processed into the next commodity in the service chain via the corresponding service function; (ii) *Flow scaling*: the flow size of a commodity can differ from the flow size of the next commodity in the service chain after service function processing; (iii) *Joint computation/communication scheduling*: different commodities share and compete for both processing and transmission resources, which need to be jointly scheduled. To the best of our knowledge, this is the first attempt to address the service chain control problem in a dynamic cloud network setting.

III. 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 form of, *e.g.*, virtual machines (VMs) over general purpose servers, while an edge represents a logical link (*e.g.*, IP link) between two cloud locations. We denote by $\delta^+(i) \in \mathcal{V}$ and $\delta^-(i) \in \mathcal{V}$ the set of outgoing and incoming neighbors of $i \in \mathcal{V}$ in \mathcal{G} , respectively. We remark that in our model, cloud network nodes may represent large datacenters at the core network level, smaller edge datacenters at the metro and/or aggregation networks, or even fog [15] or cloudlet [16] nodes at the access network.

We consider a time slotted system with slots normalized to integer units $t \in \{0, 1, 2, \dots\}$, and characterize the cloud network resource capacities and costs as follows:

- $\mathcal{K}_i = \{0, 1, \dots, K_i\}$: the set of processing resource allocation choices at node i
- $\mathcal{K}_{ij} = \{0, 1, \dots, K_{ij}\}$: the set of transmission resource allocation choices at link (i, j)
- $C_{i,k}$: the capacity, in *processing flow units* (e.g., operations per timeslot), resulting from the allocation of k processing resource units (e.g., CPUs) at node i
- $C_{ij,k}$: the capacity, in *transmission flow units* (e.g., packets per timeslot), resulting from the allocation of k transmission resource units (e.g., bandwidth blocks) at link (i, j)
- $w_{i,k}$: the cost of allocating k processing resource units at node i
- $w_{ij,k}$: the cost of allocating k transmission 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. Service Model

A network service $\phi \in \Phi$ is described by a chain of VNFs. We denote by $\mathcal{M}_\phi = \{1, 2, \dots, M_\phi\}$ the ordered set of VNFs of service ϕ . Hence, the pair (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 service $\phi \in \Phi$ implies the request for the packets originating at the source node s to go through the sequence of VNFs specified by \mathcal{M}_ϕ before exiting the network at the destination node d .

We adopt a *multi-commodity-chain* (MCC) flow model, in which a commodity identifies the packets at a given stage of a service chain for a particular destination. Specifically, we use the triplet (d, ϕ, m) to identify the packets that are output of the m -th function of service ϕ for destination d . The source commodity of service ϕ for destination d is denoted by $(d, \phi, 0)$ and the final commodity that are delivered to destination d by (d, ϕ, M_ϕ) , as illustrated in Fig. 1.

Each VNF has (possibly) different processing requirements. We denote by $r^{(\phi, m)}$ the processing-transmission flow ratio of VNF (ϕ, m) in processing flow units per transmission flow unit (e.g., operations per packet). We assume that VNFs are fully *parallelizable*, in the sense that if the total processing capacity allocated at node i , $C_{i,k}$, is used for VNF (ϕ, m) , then $C_{i,k}/r^{(\phi, m)}$ packets can be processed in one timeslot.

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) , in output flow units per input flow unit. That is, the size of the output flow of VNF (ϕ, m) is $\xi^{(\phi, m)}$ times as large as 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.²

We remark that our service model applies to a wide range of 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 [17], [18].

²We assume arbitrarily small packet granularity such that arbitrary positive scaling factors can be defined.

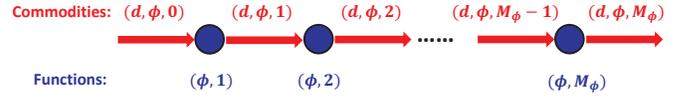


Fig. 1. A network service chain $\phi \in \Phi$. Service ϕ takes the source commodity $(d, \phi, 0)$ and delivers the final commodity (d, ϕ, M_ϕ) after going through the sequence of functions $\{(\phi, 1), \dots, (\phi, M_\phi)\}$. VNF (ϕ, m) takes commodity $(d, \phi, m-1)$ and generates commodity (d,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

C. Queuing Model

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 $\lambda_i^{(d, \phi, m)}$ its expected value. We assume that $a_i^{(d, \phi, m)}(t)$ is independently and identically distributed (i.i.d.) across timeslots, and that $a_i^{(d, \phi, m)}(t) = 0$ for $0 < m \leq M_\phi$, i.e., there are no exogenous arrivals of intermediate commodities in a service chain.³

At each timeslot t , every node buffers packets according to their commodities and makes transmission and processing **flow scheduling** decisions on its output interfaces. Cloud network queues build up from the transmission of packets from incoming neighbors and from the local processing of packets via network service functions. We define:

- $Q_i^{(d, \phi, m)}(t)$: the number of commodity (d, ϕ, m) packets in the queue of node i at the beginning of timeslot t
- $\mu_{ij}^{(d, \phi, m)}(t)$: the assigned flow rate at link (i, j) for commodity (d, ϕ, m) at time t
- $\mu_{i,pr}^{(d, \phi, m)}(t)$: the assigned flow rate from node i to its processing unit for commodity (d, ϕ, m) at time t
- $\mu_{pr,i}^{(d, \phi, m)}(t)$: the assigned flow rate from node i 's processing unit to node i for commodity (d, ϕ, m) at time t

The resulting *queuing dynamics* satisfies:

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

where $[x]^+$ denotes $\max\{x, 0\}$, and $Q_d^{(d, \phi, M_\phi)}(t) = 0, \forall d, \phi, t$. The inequality in (1) is due to the fact that the actual number of packets transmitted/processed is the minimum between the locally available packets and the assigned flow rate.

We assume that the processing resources of node i are co-located with node i and hence the packets of commodity $(d, \phi, m-1)$ processed during timeslot t are available at the queue of commodity (d, ϕ, m) at the beginning of timeslot $t+1$. We can then describe the *service chaining dynamics* at node i as follows:

$$\mu_{pr,i}^{(d, \phi, m)}(t) = \xi^{(\phi, m)} \mu_{i,pr}^{(d, \phi, m-1)}(t), \quad \forall d, \phi, m > 0. \quad (2)$$

The service chaining constraints in (2) state that, at time t , the rate of commodity (d, ϕ, m) arriving at node i from its processing unit is equal to the rate of commodity $(d, \phi, m-1)$ leaving node i to its processing unit, scaled by the scaling factor $\xi^{(\phi, m)}$. Thus, Eqs. (1) and (2) imply that the packets

³The setting in which $a_i^{(d, \phi, m)}(t) \neq 0$ for $0 < m \leq M_\phi$, while of little practical relevance, does not affect the mathematical analysis in this paper.

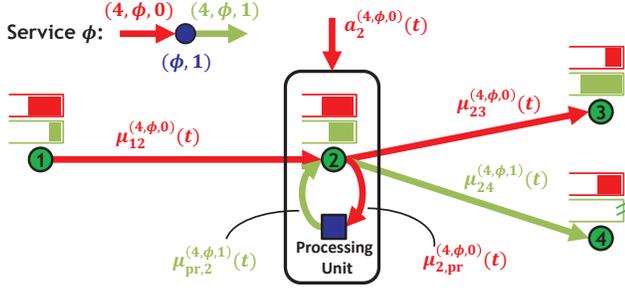


Fig. 2. Cloud network queuing model for the delivery of a single-function service for a client with source node 2 and destination node 4. Packets of both commodity $(4, \phi, 0)$ and commodity $(4, \phi, 1)$ can be forwarded across the network, where they are buffered in separate commodity queues. In addition, cloud network nodes can process packets of commodity $(4, \phi, 0)$ into packets of commodity $(4, \phi, 1)$, which can exit the network at node 4.

a commodity $(d, \phi, m - 1)$ processed during timeslot t are available at the queue of commodity (d, ϕ, m) at the beginning of timeslot $t + 1$.

As an example, the cloud network queuing system of an illustrative 4-node cloud network is shown in Fig. 2.

In addition to processing/transmission flow scheduling decisions, at each timeslot t , cloud network nodes can also make transmission and processing **resource allocation** decisions. We use the following binary variables to denote the resource allocation decisions at time t :

- $y_{i,k}(t) = 1$ if k processing resource units are allocated at node i at time t ; $y_{i,k}(t) = 0$, otherwise
- $y_{ij,k}(t) = 1$ if k transmission resource units are allocated at link (i, j) at time t ; $y_{ij,k}(t) = 0$, otherwise

D. Problem Formulation

The goal is to design a dynamic control policy, defined by a flow scheduling and resource allocation action vector $\{\mu(t), \mathbf{y}(t)\}$, that supports all average input rate matrices $\lambda \triangleq \{\lambda_i^{(d, \phi, m)}\}$ that are interior to the *cloud network capacity region* (as defined in Section IV), while minimizing the total average cloud network cost. Specifically, we require the cloud network to be *rate stable* (see Ref. [7]), i.e.,

$$\lim_{t \rightarrow \infty} \frac{Q_i^{(d, \phi, m)}(t)}{t} = 0 \quad \text{with prob. 1,} \quad \forall i, d, \phi, m. \quad (3)$$

The dynamic cloud network control problem can then be formulated as follows:

$$\min \limsup_{t \rightarrow \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \{h(\tau)\} \quad (4a)$$

$$\text{s.t. The cloud network is rate stable,} \quad (4b)$$

$$\mu_{pr,i}^{(d, \phi, m)}(\tau) = \xi^{(\phi, m)} \mu_{i,pr}^{(d, \phi, m-1)}(\tau), \quad \forall i, d, \phi, m, \tau, \quad (4c)$$

$$\sum_{(d, \phi, m)} \mu_{i,pr}^{(d, \phi, m)}(\tau) r^{(\phi, m+1)} \leq \sum_{k \in \mathcal{K}_i} C_{i,k} y_{i,k}(\tau), \quad \forall i, \tau, \quad (4d)$$

$$\sum_{(d, \phi, m)} \mu_{ij}^{(d, \phi, m)}(\tau) \leq \sum_{k \in \mathcal{K}_{ij}} C_{ij,k} y_{ij,k}(\tau), \quad \forall (i, j), \tau, \quad (4e)$$

$$\mu_{i,pr}^{(d, \phi, m)}(\tau), \mu_{pr,i}^{(d, \phi, m)}(\tau), \mu_{ij}^{(d, \phi, m)}(\tau) \in \mathbb{R}^+, \quad \forall i, (i, j), d, \phi, m, \tau, \quad (4f)$$

$$y_{i,k}(\tau), y_{ij,k}(\tau) \in \{0, 1\}, \quad \forall i, (i, j), d, \phi, m, \tau, \quad (4g)$$

where $h(\tau) \triangleq \sum_{i \in \mathcal{V}} h_i(\tau)$, with

$$h_i(\tau) = \sum_{k \in \mathcal{K}_i} w_{i,k} y_{i,k}(\tau) + e_i \sum_{(d, \phi, m)} \mu_{i,pr}^{(d, \phi, m)}(\tau) r^{(\phi, m+1)} + \sum_{j \in \delta^+(i)} \left[\sum_{k \in \mathcal{K}_{ij}} w_{ij,k} y_{ij,k}(\tau) + e_{ij} \sum_{(d, \phi, m)} \mu_{ij}^{(d, \phi, m)}(\tau) \right], \quad (5)$$

denotes the cloud network operational cost at time τ .

In (4), Eqs. (4c), (4d), and (4e) describe instantaneous service chaining, processing capacity, and transmission capacity constraints, respectively.

Remark 1. As in Eqs. (4c), (4d), (5), throughout this paper, it shall be useful to establish relationships between consecutive commodities and/or functions in a service chain. For ease of notation, unless specified, we shall assume that any expression containing a reference to $m - 1$ will only be applicable for $m > 0$ and any expression with a reference to $m + 1$ will only be applicable for $m < M_\phi$.

In the following section, we characterize the cloud network capacity region in terms of the average input rates that can be stabilized by any control algorithm that satisfies constraints (4b)-(4g), as well as the minimum average cost required for cloud network stability.

IV. CLOUD NETWORK CAPACITY REGION

The cloud network capacity region $\Lambda(\mathcal{G}, \Phi)$ is defined as the closure of all average input rates λ that can be stabilized by a cloud network control algorithm, whose decisions conform to the cloud network and service structure $\{\mathcal{G}, \Phi\}$.

Theorem 1. The cloud network capacity region $\Lambda(\mathcal{G}, \Phi)$ consists of all average input rates λ for which, for all i, j, k, d, ϕ, m , there exist MCC flow variables $f_{ij}^{(d, \phi, m)}$, $f_{pr,i}^{(d, \phi, m)}$, $f_{i,pr}^{(d, \phi, m)}$, together with probability values $\alpha_{ij,k}$, $\alpha_{i,k}$, $\beta_{ij,k}^{(d, \phi, m)}$, $\beta_{i,k}^{(d, \phi, m)}$ such that

$$\sum_{j \in \delta^-(i)} f_{ji}^{(d, \phi, m)} + f_{pr,i}^{(d, \phi, m)} + \lambda_i^{(d, \phi, m)} \leq \sum_{j \in \delta^+(i)} f_{ij}^{(d, \phi, m)} + f_{i,pr}^{(d, \phi, m)}, \quad (6a)$$

$$f_{pr,i}^{(d, \phi, m)} = \xi^{(\phi, m)} f_{i,pr}^{(d, \phi, m-1)}, \quad (6b)$$

$$f_{i,pr}^{(d, \phi, m)} \leq \frac{1}{r^{(\phi, m+1)}} \sum_{k \in \mathcal{K}_i} \alpha_{i,k} \beta_{i,k}^{(d, \phi, m)} C_{i,k}, \quad (6c)$$

$$f_{ij}^{(d, \phi, m)} \leq \sum_{k \in \mathcal{K}_{ij}} \alpha_{ij,k} \beta_{ij,k}^{(d, \phi, m)} C_{ij,k}, \quad (6d)$$

$$f_{i,pr}^{(d, \phi, M_\phi)} = 0, \quad f_{pr,i}^{(d, \phi, 0)} = 0, \quad f_{dj}^{(d, \phi, M_\phi)} = 0, \quad (6e)$$

$$f_{i,pr}^{(d, \phi, m)} \geq 0, \quad f_{ij}^{(d, \phi, m)} \geq 0, \quad (6f)$$

$$\sum_{k \in \mathcal{K}_{ij}} \alpha_{ij,k} \leq 1, \quad \sum_{k \in \mathcal{K}_i} \alpha_{i,k} \leq 1, \quad (6g)$$

$$\sum_{(d, \phi, m)} \beta_{ij,k}^{(d, \phi, m)} \leq 1, \quad \sum_{(d, \phi, m)} \beta_{i,k}^{(d, \phi, m)} \leq 1. \quad (6h)$$

Furthermore, the minimum average cloud network cost required for network stability is given by

$$\bar{h}^* = \min_{\{\alpha_{ij,k}, \alpha_{i,k}, \beta_{ij,k}^{(d, \phi, m)}, \beta_{i,k}^{(d, \phi, m)}\}} \bar{h}, \quad (7)$$

where

$$\begin{aligned} \bar{h} = & \sum_i \sum_{k \in \mathcal{K}_i} \alpha_{i,k} \left(w_{i,k} + e_i C_{i,k} \sum_{(d,\phi,m)} \beta_{i,k}^{(d,\phi,m)} \right) \\ & + \sum_{(i,j)} \sum_{k \in \mathcal{K}_{ij}} \alpha_{ij,k} \left(w_{ij,k} + e_{ij} C_{ij,k} \sum_{(d,\phi,m)} \beta_{ij,k}^{(d,\phi,m)} \right). \end{aligned} \quad (8)$$

Proof. The proof of Theorem 1 is given by Appendix A. \square

In Theorem 1, (6a) and (6b) describe generalized computation/communication flow conservation constraints and service chaining constraints, essential for cloud network stability, while (6c) and (6d) describe processing and transmission capacity constraints. The probability values $\alpha_{i,k}$, $\alpha_{ij,k}$, $\beta_{i,k}^{(d,\phi,m)}$, $\beta_{ij,k}^{(d,\phi,m)}$ define a *stationary randomized policy* as follows:

- $\alpha_{i,k}$: the probability that k processing resource units are allocated at node i ;
- $\alpha_{ij,k}$: the probability that k transmission resource units are allocated at link (i, j) ;
- $\beta_{i,k}^{(d,\phi,m)}$: the probability that node i processes commodity (d, ϕ, m) , conditioned on the allocation of k processing resource units at node i ;
- $\beta_{ij,k}^{(d,\phi,m)}$: the probability that link (i, j) transmits commodity (d, ϕ, m) , conditioned on the allocation of k transmission resource units at link (i, j) .

Hence, Theorem 1 demonstrates that, for any input rate $\lambda \in \Lambda(\mathcal{G}, \Phi)$, there exists a stationary randomized policy that uses fixed probabilities to make transmission and processing decisions at each timeslot, which can support the given λ , while minimizing overall average cloud network cost. However, the difficulty in directly solving for the parameters that characterize such a stationary randomized policy and the requirement on the knowledge of λ , motivates the design of online dynamic cloud network control solutions with matching performance guarantees.

V. DYNAMIC CLOUD NETWORK CONTROL ALGORITHMS

In this section, we describe distributed DCNC strategies that account for both processing and transmission flow scheduling and resource allocation decisions. We first propose DCNC-L, an algorithm based on minimizing a *linear metric* obtained from an upper bound of the quadratic LDP function, where only linear complexity is required for making local decisions at each timeslot. We then propose DCNC-Q, derived from the minimization of a *quadratic metric* obtained from the LDP bound. DCNC-Q allows simultaneously scheduling multiple commodities on a given transmission or processing interface at each timeslot, leading to a more balanced system evolution that can improve the cost-delay tradeoff at the expense of quadratic computational complexity. Finally, enhanced versions of the aforementioned algorithms, referred to as EDCNC-L and EDCNC-Q, are constructed by adding a shortest transmission-plus-processing distance (STPD) bias extension that is shown to further reduce network delay in low congested scenarios.

A. Cloud Network Lyapunov drift-plus-penalty

Let $\mathbf{Q}(t)$ represent the vector of queue backlog values of all the commodities at all the cloud network nodes. The cloud network *Lyapunov drift* is defined as

$$\Delta(\mathbf{Q}(t)) \triangleq \frac{1}{2} \mathbb{E} \left\{ \|\mathbf{Q}(t+1)\|^2 - \|\mathbf{Q}(t)\|^2 \mid \mathbf{Q}(t) \right\}, \quad (9)$$

where $\|\cdot\|$ indicates Euclidean norm, and the expectation is taken over the ensemble of all the exogenous source commodity arrival realizations.

The one-step Lyapunov drift-plus-penalty (LPD) is then defined as

$$\Delta(\mathbf{Q}(t)) + V \mathbb{E} \{ h(t) \mid \mathbf{Q}(t) \}, \quad (10)$$

where V is a non-negative control parameter that determines the degree to which resource cost minimization is emphasized.

After squaring both sides of (1) and following standard LDP manipulations (see Ref. [9]), the LDP can upper bound as

$$\begin{aligned} \Delta(\mathbf{Q}(t)) + V \mathbb{E} \{ h(t) \mid \mathbf{Q}(t) \} \leq & V \mathbb{E} \{ h(t) \mid \mathbf{Q}(t) \} + \\ & \mathbb{E} \{ \Gamma(t) + Z(t) \mid \mathbf{Q}(t) \} + \sum_i \sum_{(d,\phi,m)} \lambda_i^{(d,\phi,m)} Q_i^{d,\phi,m}(t), \end{aligned} \quad (11)$$

where

$$\begin{aligned} \Gamma(t) \triangleq & \frac{1}{2} \sum_i \sum_{(d,\phi,m)} \left\{ \left[\sum_{j \in \delta^+(i)} \mu_{ij}^{(d,\phi,m)}(t) + \mu_{i,\text{pr}}^{(d,\phi,m)}(t) \right]^2 \right. \\ & \left. + \left[\sum_{j \in \delta^-(i)} \mu_{ji}^{(d,\phi,m)}(t) + \mu_{\text{pr},i}^{(d,\phi,m)}(t) + a_i^{(d,\phi,m)}(t) \right]^2 \right\}, \\ Z(t) \triangleq & \sum_i \sum_{(d,\phi,m)} Q_i^{(d,\phi,m)}(t) \left[\sum_{j \in \delta^-(i)} \mu_{ji}^{(d,\phi,m)}(t) \right. \\ & \left. + \mu_{\text{pr},i}^{(d,\phi,m)}(t) - \sum_{j \in \delta^+(i)} \mu_{ij}^{(d,\phi,m)}(t) - \mu_{i,\text{pr}}^{(d,\phi,m)}(t) \right]. \end{aligned}$$

Our DCNC algorithms extract different metrics from the right hand side of (11), whose minimization leads to a family of throughput-optimal flow scheduling and resource allocation policies with different cost-delay tradeoff performance.

B. Linear Dynamic Cloud Network Control (DCNC-L)

DCNC-L is designed to minimize, at each timeslot, the linear metric $Z(t) + Vh(t)$ obtained from the right hand side of (11), equivalently expressed as

$$\min_{i \in \mathcal{V}} \sum \left[V h_i(t) - \sum_{(d,\phi,m)} \left(\sum_{j \in \delta^+(i)} Z_{ij,\text{tr}}^{(d,\phi,m)}(t) + Z_{i,\text{pr}}^{(d,\phi,m)}(t) \right) \right] \quad (12a)$$

$$\text{s.t.} \quad (4d) - (4g), \quad (12b)$$

where,

$$\begin{aligned} Z_{ij,\text{tr}}^{(d,\phi,m)}(t) & \triangleq \mu_{ij}^{(d,\phi,m)}(t) \left[Q_i^{(d,\phi,m)}(t) - Q_j^{(d,\phi,m)}(t) \right], \\ Z_{i,\text{pr}}^{(d,\phi,m)}(t) & \triangleq \mu_{i,\text{pr}}^{(d,\phi,m)}(t) \left[Q_i^{(d,\phi,m)}(t) - \xi^{(\phi,m+1)} Q_i^{(d,\phi,m+1)}(t) \right]. \end{aligned}$$

The goal of minimizing (12a) at each timeslot is to greedily push the cloud network queues towards a lightly congested state, while minimizing cloud network resource usage regulated by the control parameter V . Observe that (12a) is a linear metric with respect to $\mu_{i,\text{pr}}^{(d,\phi,m)}(t)$ and $\mu_{ij}^{(d,\phi,m)}(t)$, and hence (12) can be decomposed into the implementation of *Max-Weight-Matching* [19] at each node, leading to the following distributed flow scheduling and resource allocation policy:

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

- 1) Compute the *processing utility weight* of each processable commodity, (d, ϕ, m) , $m < M_\phi$:

$$W_i^{(d,\phi,m)}(t) = \left[\frac{Q_i^{(d,\phi,m)}(t) - \xi^{(\phi,m+1)} Q_i^{(d,\phi,m+1)}(t)}{r^{(\phi,m+1)}} - V e_i \right]^+,$$

and set $W_i^{(d,\phi,M_\phi)}(t) = 0, \forall d, \phi$. $W_i^{(d,\phi,m)}(t)$ is indicative of the potential benefit of processing commodity (d, ϕ, m) into commodity $(d, \phi, m+1)$ at time t , in terms of the difference between local congestion reduction and processing cost per unit flow.

- 2) Compute the max-weight commodity:

$$(d, \phi, m)^* = \arg \max_{(d,\phi,m)} \left\{ W_i^{(d,\phi,m)}(t) \right\}.$$

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

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

- 4) Make the following resource allocation and flow assignment decisions:

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

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

- 1) Compute the *transmission utility weight* of each commodity (d, ϕ, m) :

$$W_{ij}^{(d,\phi,m)}(t) = \left[Q_i^{(d,\phi,m)}(t) - Q_j^{(d,\phi,m)}(t) - V e_{ij} \right]^+.$$

- 2) Compute the max-weight commodity:

$$(d, \phi, m)^* = \arg \max_{(d,\phi,m)} \left\{ W_{ij}^{(d,\phi,m)}(t) \right\}.$$

- 3) If $W_{ij}^{(d,\phi,m)^*}(t) = 0$, set, $k^* = 0$. Otherwise,

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

- 4) Make the following resource allocation and flow assignment decisions:

$$\begin{aligned} y_{ij,k^*}(t) &= 1, \\ y_{ij,k}(t) &= 0, \quad \forall k \neq k^*, \\ \mu_{ij}^{(d,\phi,m)^*}(t) &= C_{ij,k^*}, \\ \mu_{ij}^{(d,\phi,m)}(t) &= 0 \quad \forall (d, \phi, m) \neq (d, \phi, m)^*. \end{aligned}$$

Implementing the above algorithm imposes low complexity on each node. Let J denote the total number of commodities. We have $J \leq N \sum_\phi (M_\phi + 1)$. Then, the total complexity associated with the processing and transmission decisions of node i at each timeslot is $O(J + K_i + \sum_{j \in \delta^+(i)} K_{ij})$, which is linear with respect to the number of commodities and the number of resource allocation choices.

Remark 2. Recall that, while assigned flow values can be larger than the corresponding queue lengths, a practical algorithm will only send those packets available for transmission/processing. However, as in [7]-[14], in our analysis, we assume a policy that meets assigned flow values with null packets (e.g., filled with idle bits) when necessary. Null packets consume resources, but do not build up in the network.

C. Quadratic Dynamic Cloud Network Control (DCNC-Q)

DCNC-Q is designed to minimize, at each timeslot, the metric formed by the sum of the quadratic terms $(\mu_{ij}^{(d,\phi,m)}(t))^2$, $(\mu_{i,\text{pr}}^{(d,\phi,m)}(t))^2$, and $(\mu_{\text{pr},i}^{(d,\phi,m)}(t))^2$, extracted from $\Gamma(t)$, and $Z(t) + Vh(t)$, on the right hand side of (11), equivalently expressed as

$$\begin{aligned} \min \quad & \sum_{i \in \mathcal{V}} \left\{ \sum_{(d,\phi,m)} \sum_{j \in \delta^+(i)} \left[\left(\mu_{ij}^{(d,\phi,m)}(t) \right)^2 - Z_{ij,\text{tr}}^{(d,\phi,m)}(t) \right] \right. \\ & + \sum_{(d,\phi,m)} \left[\frac{1 + (\xi^{(\phi,m+1)})^2}{2} \left(\mu_{i,\text{pr}}^{(d,\phi,m)}(t) \right)^2 - Z_{i,\text{pr}}^{(d,\phi,m)}(t) \right] \\ & \left. + V h_i(t) \right\} \end{aligned} \quad (13a)$$

$$\text{s.t.} \quad (4d) - (4g). \quad (13b)$$

The purpose of (13) is also to reduce the congestion level while minimizing resource cost. However, by introducing the quadratic terms $(\mu_{i,\text{pr}}^{(d,\phi,m)}(t))^2$ and $(\mu_{ij}^{(d,\phi,m)}(t))^2$, minimizing (13a) results in a “smoother” and more “balanced” flow and resource allocation solution, which has the potential of improving the cost-delay tradeoff, with respect to the max-weight solution of DCNC-L that allocates either zero or full capacity to a single commodity at each timeslot. Note that (13) can also be decomposed into subproblems at each cloud network node. Using the *KKT conditions* [20], the solution to each subproblem admits a simple waterfilling-type interpretation. We first describe the resulting local flow scheduling and resource allocation policy and then provide its graphical interpretation.

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

- 1) Compute the processing utility weight of each commodity. Sort the resulting set of weights in non-increasing order and form the list $\{W_i^{(c)}(t)\}$, where c identifies the c -th commodity in the sorted list.
- 2) For each resource allocation choice $k \in \mathcal{K}_i$:
 - 2.1) Compute the following *waterfilling rate threshold*:

$$G_{i,k}(t) \triangleq \left[\frac{\sum_{s=1}^{p_k} \frac{(r^{(s)})^2}{1+(\xi^{(s)})^2} W_i^{(s)}(t) - C_{i,k}}{\sum_{s=1}^{p_k} \frac{(r^{(s)})^2}{1+(\xi^{(s)})^2}} \right]^+,$$

where p_k is the smallest commodity index that satisfies $H_i^{(p_k)}(t) > C_{i,k}$, with $p_k = J$ if $C_{i,k} \geq H_i^{(J)}(t)$; and

$$H_i^{(c)}(t) \triangleq \sum_{s=1}^c \left[W_i^{(s)}(t) - W_i^{(c+1)}(t) \right] \frac{(r^{(s)})^2}{1+(\xi^{(s)})^2},$$

with $r^{(s)}$ and $\xi^{(s)}$ denoting the processing-transmission flow ratio and the scaling factor of the function that processes commodity s , respectively.

2.2) Compute the *candidate* processing flow rate for each commodity, $1 \leq c \leq J$:

$$\check{\mu}_{i,\text{pr}}^{(c)}(k, t) = \frac{r^{(c)}}{1+(\xi^{(c)})^2} \left[W_i^{(c)}(t) - G_{i,k}(t) \right]^+.$$

2.3) Compute the following optimization metric:

$$\Psi_i(k, t) \triangleq \sum_{c=1}^J \left[\frac{1+(\xi^{(c)})^2}{2} \left(\check{\mu}_{i,\text{pr}}^{(c)}(k, t) \right)^2 - \check{\mu}_{i,\text{pr}}^{(c)}(k, t) r^{(c)} W_i^{(c)}(t) \right] + V w_{i,k}.$$

3) Compute the processing resource allocation choice:

$$k^* = \arg \min_{k \in \mathcal{K}_i} \{ \Psi_i(k, t) \}.$$

4) Make the following resource allocation and flow assignment decisions:

$$\begin{aligned} y_{i,k^*}(t) &= 1, \\ y_{i,k}(t) &= 0, \quad \text{for } k \neq k^*, \\ \mu_{i,\text{pr}}^{(c)}(t) &= \check{\mu}_{i,\text{pr}}^{(c)}(k^*, t). \end{aligned}$$

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

- 1) Compute the transmission utility weight of each commodity. Sort the resulting set of weights in non-increasing order and form the list $\{W_{ij}^{(c)}(t)\}$, where c identifies the c -th commodity in the sorted list.
- 2) For each resource allocation choice $k \in \mathcal{K}_i$:
 - 2.1) Compute the following *waterfilling rate threshold*:

$$G_{ij,k}(t) \triangleq \frac{1}{p_k} \left[\sum_{s=1}^{p_k} W_{ij}^{(s)}(t) - 2C_{ij,k} \right]^+.$$

where p_k is the smallest commodity index that satisfies $H_{ij}^{(p_k)}(t) > C_{ij,k}$, with $p_k = J$ if $C_{ij,k} \geq H_{ij}^{(J)}(t)$; and

$$H_{ij}^{(c)}(t) \triangleq \frac{1}{2} \sum_{s=1}^c \left[W_{ij}^{(s)}(t) - W_{ij}^{(c+1)}(t) \right].$$

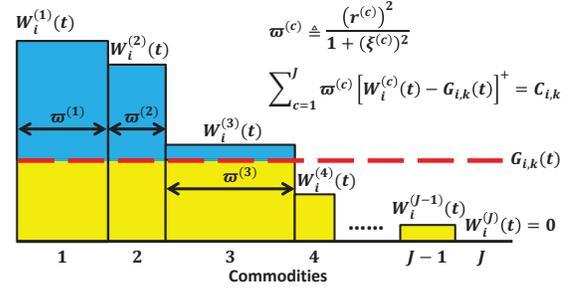


Fig. 3. Waterfilling interpretation of the local processing decisions of DCNC-Q at time t .

2.2) Compute the candidate transmission flow rate for each commodity, $1 \leq c \leq J$:

$$\check{\mu}_{ij}^{(c)}(k, t) = \frac{1}{2} \left[W_{ij}^{(c)}(t) - G_{ij,k}(t) \right]^+.$$

2.3) Compute the following optimization metric:

$$\Psi_{ij}(k, t) \triangleq V w_{ij,k} + \sum_{c=1}^J \left[\left(\check{\mu}_{ij}^{(c)}(k, t) \right)^2 - \check{\mu}_{ij}^{(c)}(k, t) W_{ij}^{(c)}(t) \right].$$

3) Compute the processing resource allocation choice:

$$k^* = \arg \min_{k \in \mathcal{K}_{ij}} \{ \Psi_{ij}(k, t) \}.$$

4) Make the following resource allocation and flow assignment decisions:

$$\begin{aligned} y_{ij,k^*}(t) &= 1, \\ y_{ij,k}(t) &= 0, \quad \forall k \neq k^*, \\ \mu_{ij}^{(c)}(t) &= \check{\mu}_{ij}^{(c)}(k^*, t). \end{aligned}$$

The total complexity is $O(J[\log_2 J + K_i + \sum_{j \in \delta^+(i)} K_{ij}])$, which is quadratic respective to the number of commodities and the number of resource allocation choices.

As stated earlier, DCNC-Q admits a waterfilling-type interpretation, illustrated in Fig. 3. We focus on the local processing decisions. Define a two-dimensional vessel for each commodity. The height of vessel c is given by the processing utility weight of commodity c , $W_i^{(c)}(t)$, and its width by $\frac{(r^{(c)})^2}{1+(\xi^{(c)})^2}$. For each resource allocation choice $k \in \mathcal{K}_i$, pour mercury on each vessel up to height $G_{i,k}(t)$ given in step 2.1 (indicated with yellow in the figure). If available, fill the remaining of each vessel with water (blue in the figure). The candidate assigned flow rate of each commodity is given by the amount of water on each vessel (step 2.2), while total amount of water is equal to the available capacity $C_{i,k}$. Finally, step 3 is the result of choosing the resource allocation choice k^* that minimizes (13a) with the corresponding assigned flow rate values. The local transmission decisions follow a similar interpretation that is omitted here for brevity.

D. Dynamic Cloud Network Control with Shortest Transmission-plus-Processing Distance Bias

DCNC algorithms determine packet routes and processing locations according to the evolution of the cloud network commodity queues. However, queue backlogs have to build up before yielding efficient processing and routing configurations,

which can result in degraded delay performance, especially in low congested scenarios.

In order to reduce average cloud network delay, we extend the approach used in [12], [13] for traditional communication networks, which consists of incorporating a bias term into the metrics that drive scheduling decisions. In a cloud network setting, this bias is designed to capture the delay penalty incurred by each forwarding and processing operation.

Let $\hat{Q}_i^{(d,\phi,m)}(t)$ denote the *biased* backlog of commodity (d, ϕ, m) at node i :

$$\hat{Q}_i^{(d,\phi,m)}(t) \triangleq Q_i^{(d,\phi,m)}(t) + \eta Y_i^{(d,\phi,m)}, \quad (14)$$

where $Y_i^{(d,\phi,m)}$ denotes the *shortest transmission-plus-processing distance bias* (STPD), and η is a control parameter used to balance the effect of the bias and the queue backlog. The bias term in (14) is defined as

$$Y_i^{(d,\phi,m)} \triangleq \begin{cases} 1, & \text{if } m < M_\phi, \\ H_{i,d}, & \text{if } m = M_\phi, \end{cases} \quad \forall i, d, \phi, \quad (15)$$

where $H_{i,j}$ denotes the shortest distance (in number of hops) from node i to node j . We note that $Y_i^{(d,\phi,m)} = 1$ for all processable commodities because, throughout this paper, we have assumed that every function is available at all cloud network nodes. In Sec. VIII-A, we discuss a straight-forward generalization of our model, in which each service function is available at a subset of cloud network nodes, in which case, $Y_i^{(d,\phi,m)}$ for each processable commodity is defined as the shortest distance to the closest node that can process commodity (d, ϕ, m) .

The enhanced EDCNC-L and EDCNC-Q algorithms work just like their DCNC-L and DCNC-Q counterparts, but using $\hat{Q}_i^{(d,\phi,m)}(t)$ in place of $Q_i^{(d,\phi,m)}(t)$ to make local processing and transmission scheduling decisions.

VI. PERFORMANCE ANALYSIS

In this section, we analyze the performance of the proposed DCNC algorithms. To facilitate the analysis, we define the following parameters:

- A_{\max} : the constant that bounds the aggregate input rate at all the cloud network nodes; specifically, $\max_{i \in \mathcal{V}} \mathbb{E}\{[\sum_{(d,\phi,m)} a_i^{(d,\phi,m)}(t)]^4\} \leq (A_{\max})^4$.
- C_{pr}^{\max} : the maximum processing capacity among all cloud network nodes; *i.e.*, $C_{\text{pr}}^{\max} \triangleq \max_{i \in \mathcal{V}} \{C_{i,K_i}\}$.
- C_{tr}^{\max} : the maximum transmission capacity among all cloud network links; *i.e.*, $C_{\text{tr}}^{\max} \triangleq \max_{(i,j) \in \mathcal{E}} \{C_{ij,K_{ij}}\}$.
- ξ_{\max} : the maximum flow scaling factor among all service functions; *i.e.*, $\xi_{\max} \triangleq \max_{(\phi,m)} \{\xi^{(\phi,m)}\}$.
- r_{\min} : the minimum transmission-processing flow ratio among all service functions; *i.e.*, $r_{\min} \triangleq \min_{(\phi,m)} \{r^{(\phi,m)}\}$.
- δ_{\max} : the maximum degree among all cloud network nodes, *i.e.*, $\delta_{\max} \triangleq \max_{i \in \mathcal{V}} \{\delta^+(i) + \delta^-(i)\}$.

A. Average Cost and Network Stability

Theorem 2. *If the average input rate matrix $\lambda = (\lambda_i^{(d,\phi,m)})$ is interior to the cloud network capacity region $\Lambda(\mathcal{G}, \Phi)$,*

then the DCNC algorithms stabilize the cloud network, while achieving arbitrarily close to minimum average cost $\bar{h}^(\lambda)$ with probability 1 (w.p.1), i.e.,*

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} h(\tau) \leq \bar{h}^*(\lambda) + \frac{NB}{V}, \quad (\text{w.p.1}) \quad (16)$$

$$\limsup_{t \rightarrow \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} \sum_{\tau=0, (d,\phi,m), i} Q_i^{(d,\phi,m)}(\tau) \leq \frac{NB+V[\bar{h}^*(\lambda+\kappa\mathbf{1})-\bar{h}^*(\lambda)]}{\kappa}, \quad (\text{w.p.1}) \quad (17)$$

where

$$B = \begin{cases} B_0, & \text{under DCNC-L and DCNC-Q,} \\ B_1, & \text{under EDCNC-L and EDCNC-Q,} \end{cases} \quad (18)$$

with B_0 and B_1 being positive constants determined by the system parameters C_{pr}^{\max} , C_{tr}^{\max} , A_{\max} , ξ_{\max} , and r_{\min} ; and κ is a positive constant satisfying $(\lambda + \kappa\mathbf{1}) \in \Lambda$.

Proof. The proof of Theorem 2 is given in Appendix B. \square

Theorem 2 shows that the proposed DCNC algorithms achieve the average cost-delay tradeoff $[O(1/V), O(V)]$ with probability 1.⁴ Moreover, (17) holds for any λ interior to Λ , which demonstrates the throughput-optimality of the DCNC algorithms.

B. Convergence Time

The convergence time of a DCNC algorithm indicates how fast its running time average solution approaches the optimal solution.⁵ This criterion is particularly important for online scheduling in settings where the arrival process is non-homogeneous, *i.e.*, the average input rate λ is time varying. In this case, it is important to make sure that the time average solution evolves close enough to the optimal solution much before the average input rate undergoes significant changes.

We remark that studying the convergence time of a DCNC algorithm involves studying how fast the average cost approaches the optimal value, as well as how fast the flow conservation violation at each node approaches zero.⁶

Let $\tilde{\mu}_{i,\text{pr}}^{(d,\phi,m)}(t)$, $\tilde{\mu}_{\text{pr},i}^{(d,\phi,m)}(t)$, and $\tilde{\mu}_{ij}^{(d,\phi,m)}(t)$ denote the *actual* flow rates obtained from removing all *null* packets that may have been assigned when queues do not have enough packets to meet the corresponding assigned flow rates. Define, for all $i, (d, \phi, m), t$,

$$\begin{aligned} \Delta f_i^{(d,\phi,m)}(t) &\triangleq \sum_{j \in \delta^-(i)} \tilde{\mu}_{ji}^{(d,\phi,m)}(t) + \tilde{\mu}_{\text{pr},i}^{(d,\phi,m)}(t) - a_i^{(d,\phi,m)}(t) \\ &\quad - \sum_{j \in \delta^+(i)} \tilde{\mu}_{ij}^{(d,\phi,m)}(t) - \tilde{\mu}_{i,\text{pr}}^{(d,\phi,m)}(t). \end{aligned} \quad (19)$$

⁴By setting $\epsilon = 1/V$, where ϵ denotes the deviation from the optimal solution (see Theorem 3), the cost-delay tradeoff is written as $[O(\epsilon), O(1/\epsilon)]$.

⁵We assume that the local decisions performed by the DCNC algorithms at each timeslot can be accomplished within a reserved computation time within each timeslot, and therefore their different computational complexities are not taking into account for convergence time analysis.

⁶Note that the convergence of the flow conservation violation at each node to zero is equivalent to *strong stability* (see (17)), if λ interior to $\Lambda(\mathcal{G}, \Phi)$.

Then, the queuing dynamics is then given by

$$Q_i^{(d,\phi,m)}(t+1) = Q_i^{(d,\phi,m)}(t) + \Delta f_i^{(d,\phi,m)}(t). \quad (20)$$

The convergence time performance of the proposed DCNC algorithms is summarized by the following theorem.

Theorem 3. *If the average input rate matrix $\lambda = (\lambda_i^{(d,\phi,m)})$ is interior to the cloud network capacity region $\Lambda(\mathcal{G}, \Phi)$, then, for all $\epsilon > 0$, whenever $t \geq 1/\epsilon^2$, the mean time average cost and mean time average actual flow rate achieved by the DCNC algorithms during the first t timeslots satisfy:*

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \{h(\tau)\} \leq \bar{h}^*(\lambda) + O(\epsilon), \quad (21)$$

$$\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \left\{ \Delta f_i^{(d,\phi,m)}(t) \right\} \leq O(\epsilon), \quad \forall i, (d, \phi, m). \quad (22)$$

Proof. The proof is of Theorem 3 given in Appendix C. \square

Theorem 3 establishes that, under the DCNC algorithms, both the average cost and the average flow conservation at each node exhibit $O(1/\epsilon^2)$ convergence time to $O(\epsilon)$ deviations from the minimum average cost, and zero, respectively.

VII. NUMERICAL RESULTS

In this section, we evaluate the performance of the proposed DCNC algorithms via numerical simulations in a number of illustrative settings. We assume a cloud network based on the continental US Abilene topology shown in Fig. 4. The 14 cloud network links exhibit homogeneous transmission capacities and costs, while the 7 cloud network nodes only differ in their processing resource set-up costs. Specifically, the following two resource settings are considered:

1) *ON/OFF resource levels:* each node and link can either allocate zero capacity, or the maximum available capacity; *i.e.*, $K_i = K_{ij} = 1, \forall i \in \mathcal{V}, (i, j) \in \mathcal{E}$. To simplify notation, we define $K \triangleq K_i + 1 = K_{ij} + 1, \forall i \in \mathcal{V}, (i, j) \in \mathcal{E}$. The processing resource costs and capacities are

- $e_i = 1, \forall i \in \mathcal{V}; w_{i,0} = 0, \forall i \in \mathcal{V}; w_{i,1} = 440, \forall i \in \mathcal{V} \setminus \{5, 6\}; w_{5,1} = w_{6,1} = 110.$
- $C_{i,0} = 0, C_{i,1} = 440, \forall i \in \mathcal{V}.$ ⁷

The transmission resource costs and capacities are

- $e_{ij} = 1, w_{ij,0} = 0, w_{ij,1} = 440, \forall (i, j) \in \mathcal{E}.$
- $C_{ij,0} = 0, C_{ij,1} = 440, \forall (i, j) \in \mathcal{E}.$

2) *Multiple resource levels:* the available capacity at each node and link is split into 10 resource units; *i.e.*, $K = 11, \forall i \in \mathcal{V}, (i, j) \in \mathcal{E}$. The processing resource costs and capacities are

- $e_i = 1, \forall i \in \mathcal{V};$
 $[w_{i,0}, w_{i,1}, \dots, w_{i,10}, w_{i,11}] = [0, 11, \dots, 99, 110],$ for $i = 5, 6;$
 $[w_{i,0}, w_{i,1}, \dots, w_{i,10}, w_{i,11}] = [0, 44, \dots, 396, 440], \forall i \in \mathcal{V} \setminus \{5, 6\};$
- $[C_{i,0}, C_{i,1}, \dots, C_{i,10}, C_{i,11}] = [0, 44, \dots, 396, 440], \forall i.$

The transmission resource costs and capacities are

⁷The maximum capacity is set to 440 in order to guarantee that there is no congestion at any part of the network for the service setting considered in the following.

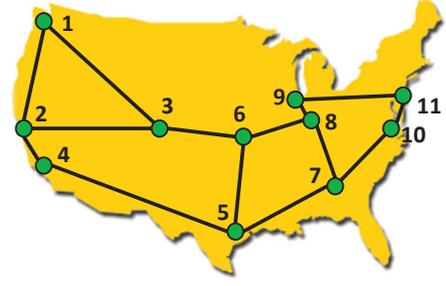


Fig. 4. Abilene US Continental Network. Nodes are indexed as: 1) Seattle, 2) Sunnyvale, 3) Denver, 4) Los Angeles, 5) Houston, 6) Kansas City, 7) Atlanta, 8) Indianapolis, 9) Chicago, 10) Washington, 11) New York.

- $e_{ij} = 1, \forall (i, j) \in \mathcal{E};$
 $[w_{ij,0}, w_{ij,1}, \dots, w_{ij,10}, w_{ij,11}] = [0, 44, \dots, 396, 440],$
 $\forall (i, j) \in \mathcal{E}.$
- $[C_{ij,0}, C_{ij,1}, \dots, C_{ij,10}, C_{ij,11}] = [0, 44, \dots, 396, 440],$
 $\forall (i, j) \in \mathcal{E}.$

Note that, for both ON/OFF and multi-level resource settings, the processing resource set-up costs at node 5 and 6 are 4 times cheaper than at the other cloud network nodes.

We consider 2 service chains, each composed of 2 virtual network functions: VNF (1,1) (Service 1, Function 1) with flow scaling factor $\xi^{(1,1)} = 1$; VNF (1,2) with $\xi^{(1,2)} = 3$ (expansion function); VNF (2,1) with $\xi^{(2,1)} = 0.25$ (compression function); and VNF (2,2) with $\xi^{(2,2)} = 1$. All functions have processing-transmission flow ratio $r^{(\phi,m)} = 1$, and can be implemented at all cloud network nodes. Finally, we assume 110 clients per service, corresponding to all the source-destination pairs in the Abilene network.

A. Cost-Delay Tradeoff

Figs. 5(a)-5(c) show the tradeoff between the time average cost and the time average end-to-end delay (represented by the total time average occupancy or queue backlog), under the different DCNC algorithms. The input rate of all source commodities is set to 1 and the cost/delay values are obtained after simulating each algorithm for 10^6 timeslots. Each tradeoff curve is obtained by varying the control parameter V between 0 and 1000 for each algorithm. Small values of V favor low delay at the expense of high cost, while large values of V lead to points in the tradeoff curves with lower cost and higher delay.

It is important to note that since the two resource settings considered, *i.e.*, ON/OFF ($K = 2$) vs. multi-level ($K = 11$), are characterized by the same maximum capacity and the same constant ratios $C_{i,k}/w_{i,k}$ and $C_{ij,k}/w_{ij,k}$, the performance of the linear DCNC algorithms (DCNC-L and EDCNC-L) does not change under the two resource settings. On the other hand, the quadratic algorithms (DCNC-Q and EDCNC-Q) can exploit the finer resource granularity of the multi-level resource setting to improve the cost-delay tradeoff. We also note that for the enhanced versions of the algorithms that use the STPD bias (EDCNC-L and EDCNC-Q), we choose the bias coefficient η among the values of multiples of 10 that leads to the best performance for each algorithm.⁸

⁸Simulation results for different values of η can be found in [2].

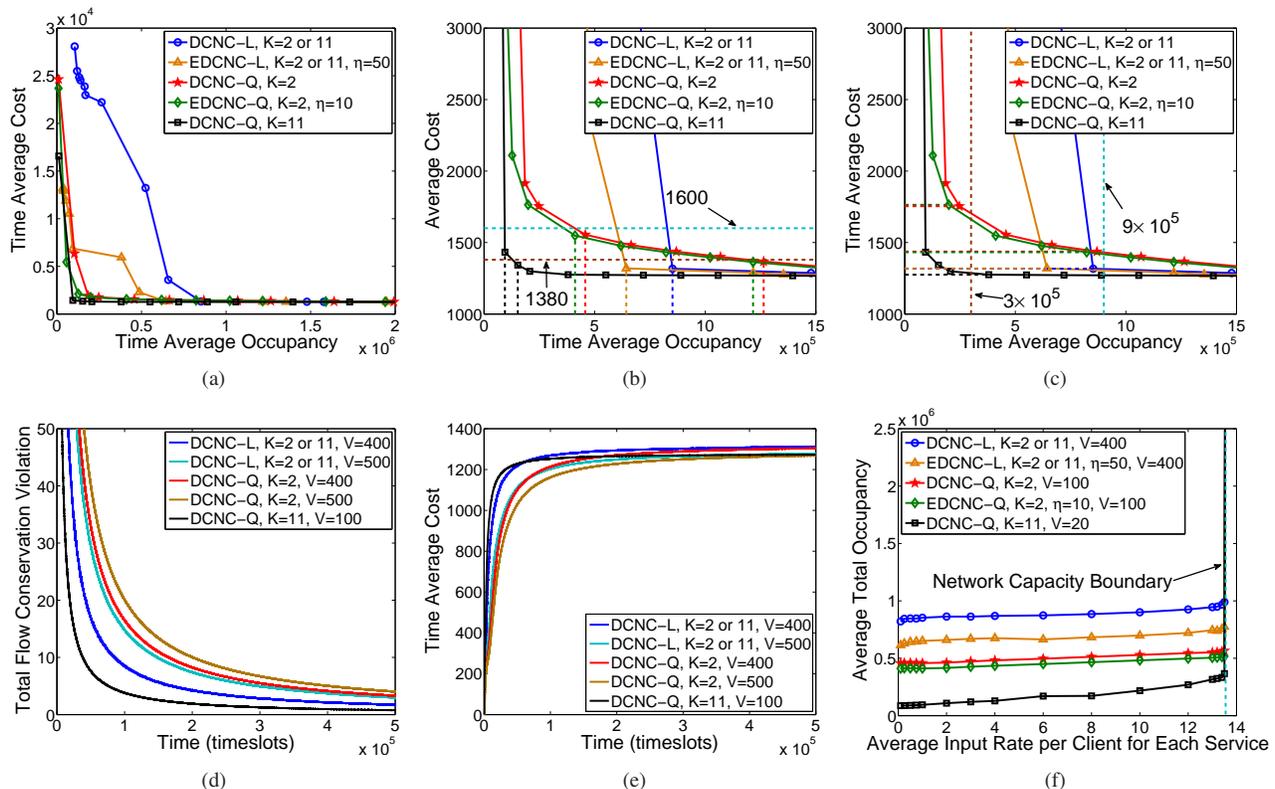


Fig. 5. Performance of DCNC algorithms. a) Time Average Occupancy v.s. Time Average Cost: a general view; b) Time average Occupancy v.s. Time Average Cost: given a target average cost c) Time average Occupancy v.s. Time Average Cost: given a target average occupancy; d) Total flow conservation violation evolution over time: effect of the V value; e) Time average cost evolution over time: effect of the V value; f) Time Average Occupancies with varying service input rate: throughput optimality

Fig. 5(a) shows how the average cost under all DCNC algorithms reduces at the expense of network delay, and converges to the same minimum value. While all the tradeoff curves follow the same $[O(1/V), O(V)]$ relationship established in Theorem 2, the specific trade-off ratios can be significantly different. The general trends observed in Fig. 5(a) are as follows. DCNC-L exhibits the worst cost-delay tradeoff. Recall that DCNC-L assigns either zero or full capacity to a single commodity in each timeslot, and hence the finer resource granularity of $K = 11$ does not improve its performance. However, adding the SDTP bias results in a substantial performance improvement, as shown by the EDCNC-L curve. Now let's focus on the quadratic algorithms. DCNC-Q with $K = 2$ further improves the cost delay-tradeoff, at the expense of increased computational complexity. In this case, adding the SDTP bias provides a much smaller improvement (see EDCNC-Q curve), showing the advantage of the more “balanced” scheduling decisions of DCNC-Q. Finally, DCNC-Q with $K = 11$ exhibits the best cost-delay tradeoff, illustrating the ability of DCNC-Q to exploit the finer resource granularity to make “smoother” resource allocation decisions. In this setting, adding the SDTP bias does not provide further improvement and it is not shown in the figure.

While Fig. 5(a) illustrates the general trends in improvements obtained using the quadratic metric and the SDTP bias, there are regimes in which the lower complexity DCNC-L and EDCNC-L algorithms can outperform their quadratic counterparts. We illustrate these regimes in Figs. 5(b) and 5(c), by zooming into the lower left of Fig. 5(a). As shown in Fig.

5(b), for the case of $K = 2$, the cost-delay curves of DCNC-L and EDCNC-L cross with the curves of DCNC-Q and EDCNC-Q. For example, for a target cost of 1380, DCNC-L and EDCNC-L result in lower average occupancies (8.52×10^5 and 6.43×10^5) than DCNC-Q (1.26×10^6) and EDCNC-Q (1.21×10^6). On the other hand, if we increase the target cost to 1600, DCNC-Q and EDCNC-Q achieve lower occupancy values (4.58×10^5 and 4.11×10^5) than DCNC-L (8.52×10^5) and EDCNC-L (6.43×10^5). Hence, depending on the cost budget, there may be a regime in which the simpler DCNC-L and EDCNC-L algorithms become a better choice. However, this regime does not exist for $K = 11$, where the average occupancies under DCNC-Q (1342 and 1433 respectively for the two target costs) are much lower than (E)DCNC-L.

In Fig. 5(c), we compare cost values for given target occupancies. With $K = 2$ and a target average occupancy of 9×10^5 , the average costs achieved by DCNC-L (1317) and EDCNC-L (1319) are lower than those achieved by DCNC-Q (1437) and EDCNC-Q (1432). In contrast, if we reduce the target occupancy to 3×10^5 , DCNC-Q and EDCNC-Q (achieving average costs 1754 and 1764) outperform DCNC-L and EDCNC-L (with cost values 2.64×10^4 and 6879 beyond the scope of Fig. 5(c)). With $K = 11$, DCNC-Q achieves average costs of 1286 and 1271 for the two target occupancies, outperforming all other algorithms.

B. Convergence Time

In Figs. 5(d) and 5(e), we show the time evolution of the total flow conservation violation (obtained by summing

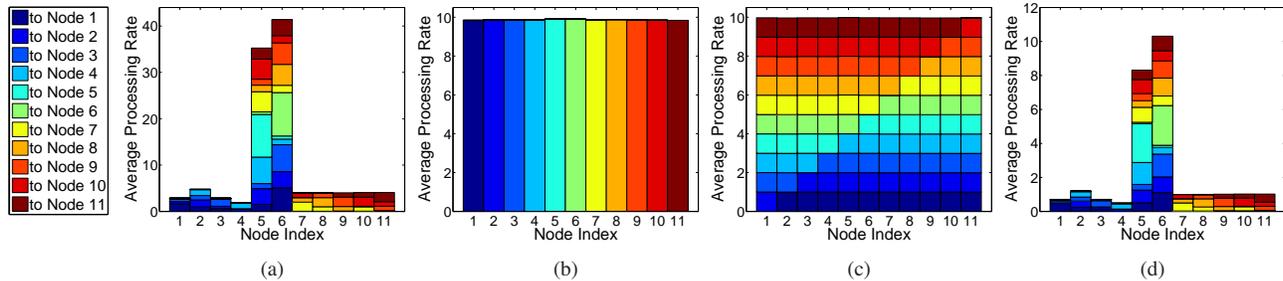


Fig. 6. Average Processing Flow Rate Distribution. a) Service 1, Function 1; b) Service 1, Function 2; c) Service 2, Function 1; d) Service 2, Function 2.

over all nodes and commodities, the absolute value of the flow conservation violation) and the total time average cost, respectively. The average input rate of each source commodity is again set to 1. As expected, observe how decreasing the value of V speeds up the convergence of all DCNC algorithms. However, note from Fig. 5(e) that the converged time average cost is higher with a smaller value of V , consistent with the tradeoff established in Theorem 2. Note that the slower convergence of DCNC-Q with respect to DCNC-L with the same value of V does not necessarily imply a disadvantage of DCNC-Q. In fact, due to its more balanced scheduling decisions, DCNC-Q can be designed with a smaller V than DCNC-L, in order to enhance convergence speed while achieving no worse cost/delay performance. This effect is obvious in the case of $K = 11$. As shown in Fig. 5(d) and Fig. 5(e), with $K = 11$, DCNC-Q with $V = 100$ achieves faster convergence than DCNC-L with $V = 400$, while their converged average cost values are similar.

C. Capacity Region

Fig. 5(f) illustrates the throughput performance of the DCNC algorithms by showing the time average occupancy as a function to the input rate (kept the same for all source commodities). The simulation time is 10^6 timeslots and the values of V used for each algorithm are chosen according to Fig. 5(b) in order to guarantee that the average cost is lower than the target value 1600. As the average input rate increases to 13.5, the average occupancy under all the DCNC algorithms exhibits a sharp raise, illustrating the boundary of the cloud network capacity region (see (17) and let $\kappa \rightarrow 0$).

Observe, once more, the improved delay performance achieved via the use of the STPD bias and the quadratic metric in the proposed control algorithms.

D. Processing Distribution

Fig. 6 shows the optimal average processing rate distribution across the cloud network nodes for each service function under the ON/OFF resource setting ($K = 2$). We obtain this solution, for example, by running DCNC-L with $V = 1000$ for 10^6 timeslots. The processing rate of function (ϕ, m) refers to the processing rate of its input commodity $(d, \phi, m - 1)$.

Observe how the implementation of VNF (1,1) mostly concentrates at node 5 and 6, which are the cheapest processing locations. However, note that part of VNF (1,1) for destinations in the west coast (nodes 1 through 4) takes place at

the west coast nodes, illustrating the fact that while processing is cheaper at nodes 5 and 6, shorter routes can compensate the extra processing cost at the more expensive nodes. A similar effect can be observed for destinations in the east coast, where part of VNF(1,1) takes place at east coast nodes.

Fig. 6(b) shows the average processing rate distribution for VNF (1,2). Note that VNF (1,2) is an expansion function. This results in the processing of commodity $(d, 1, 1)$ completely concentrating at the destination nodes, in order to minimize the impact of the extra cost incurred by the transmission of the larger-size commodity $(d, 1, 2)$ resulted from the execution of VNF (1,2).

For Service 2, note that VNF (2,1) is a compression function. As expected, the implementation of VNF (2,1) takes place at the source nodes, in order to reduce the transmission cost of Service 2 by compressing commodity $(d, 2, 0)$ into the smaller-size commodity $(d, 2, 1)$ even before commodity $(d, 2, 0)$ flows into the network. As a result, as shown in Fig. 6(c), for all $1 \leq d \leq 11$, commodity $(d, 2, 0)$ is processed at all the nodes except node d , and the average processing rate of commodity $(d, 2, 0)$ at each node $i \neq d$ is equal to 1, which is the average input rate per client.

Fig. 6(d) shows the average processing rate distribution for VNF (2,2), which exhibits a similar distribution to VNF (1,1), except for having different rate values, due to the compression effect of VNF (2,1).

VIII. EXTENSIONS

In this section, we discuss interesting extensions of the DCNC algorithms presented in this paper that can be easily captured via simple modifications to our model.

A. Location-Dependent Service Functions

For ease of notation and presentation, throughout this paper, we have implicitly assumed that every cloud network node can implement all network functions. In practice, each cloud network node may only host a subset of functions $\widetilde{\mathcal{M}}_{\phi,i} \subseteq \mathcal{M}_{\phi}, \forall \phi \in \Phi$. In this case, the local processing decisions at each node would be made by considering only those commodities that can be processed by the locally available functions. In addition, the STPD bias $Y_i^{(d,\phi,m)}$ would need to be updated as, for all i, d, ϕ ,

$$Y_i^{(d,\phi,m)} \triangleq \begin{cases} \min_{j:j \in \mathcal{V}, (m+1) \in \widetilde{\mathcal{M}}_{\phi,j}} \{H_{i,j} + 1\}, & \text{if } m < \widetilde{M}_{\phi}, \\ H_{i,d}, & \text{if } m = \widetilde{M}_{\phi}. \end{cases}$$

B. Propagation Delay

In this work, we have assumed that network delay is dominated by queuing delay, and ignored propagation delay. However, in large-scale cloud networks, where communication links can have large distances, the propagation of data across two neighbor nodes may incur non-negligible delays. In addition, while much smaller, the propagation delay incurred when forwarding packets for processing in a large data center may also be non-negligible. In order to capture propagation delays, let D_i^{pg} and D_{ij}^{pg} denote the propagation delay (in timeslots) for reaching the processing unit at node i and for reaching neighbor j from node i , respectively. We then have the following queuing dynamics and service chaining constraints:

$$Q_i^{(d,\phi,m)}(t+1) \leq \left[Q_i^{(d,\phi,m)}(t) - \sum_{j \in \delta^+(i)} \mu_{ij}^{(d,\phi,m)}(t) - \mu_{i,\text{pr}}^{(d,\phi,m)}(t) \right]^+ + \sum_{j \in \delta^-(i)} \mu_{ji}^{(d,\phi,m)}(t - D_{ji}^{\text{pg}}) + \mu_{\text{pr},i}^{(d,\phi,m)}(t) + a_i^{(d,\phi,m)}(t), \quad (23)$$

$$\mu_{\text{pr},i}^{(d,\phi,m)}(t) = \xi^{(\phi,m)} \mu_{i,\text{pr}}^{(d,\phi,m-1)}(t - D_i^{\text{pg}}). \quad (24)$$

Moreover, due to propagation delay, queue backlog observations become outdated. Specifically, the queue backlog of commodity (d, ϕ, m) at node $j \in \delta(i)$ observed by node i at time t is $Q_j^{(d,\phi,m)}(t - D_{ji}^{\text{pg}})$.

Furthermore, for EDCNC-L and EDCNC-Q, the STPD bias $Y_i^{(d,\phi,m)}$, for all i, d, ϕ , would be updated as

$$Y_i^{(d,\phi,m)} \triangleq \begin{cases} \min_{j \in \mathcal{V}} \{ \tilde{H}_{i,j} + D_i^{\text{pg}} \}, & \text{if } m < M_\phi. \\ \tilde{H}_{i,d}, & \text{if } m = M_\phi, \end{cases}$$

where $\tilde{H}_{i,j}$ is the length of the shortest path from node i to node j , with link $(u, v) \in \mathcal{E}$ having length D_{uv}^{pg} .

With (23), (24), and the outdated backlog state observations, the proposed DCNC algorithms can still be applied and be proven to retain the established throughput, average cost, and convergence performance guarantees, while suffering from increased average delay.

C. Service Tree Structure

While most of today's network services can be described via a chain of network functions, next generation digital services may contain functions with multiple inputs. Such services can be described via a *service tree*, as shown in Fig. 7.

In order to capture these type of services, we let $\mathcal{I}(\phi, m)$ denote the set of commodities that act as input to function (ϕ, m) , generating commodity (d, ϕ, m) . The service chaining constraints are then updated as

$$\mu_{\text{pr},i}^{(d,\phi,m)}(t) = \xi^{(\phi,n)} \mu_{i,\text{pr}}^{(d,\phi,n)}(t), \quad \forall t, i, d, \phi, m, n \in \mathcal{I}(\phi, m).$$

where $\xi^{(\phi,n)}, \forall n \in \mathcal{I}(\phi, m)$ denotes the flow size ratio between the output commodity (d, ϕ, m) and each of its input commodities $n \in \mathcal{I}(\phi, m)$. In addition, the processing capacity constraints are updated as

$$\sum_{(d,\phi,n)} \mu_{i,\text{pr}}^{(d,\phi,n)}(t) r^{(\phi,n)} \leq \sum_{k \in \mathcal{K}_i} C_{i,k} y_{i,k}(t), \quad \forall t, i,$$

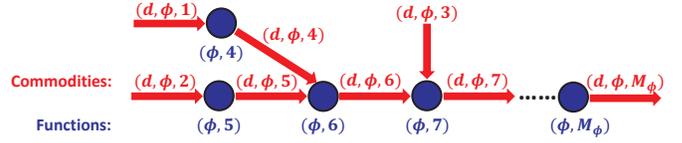


Fig. 7. A network service tree $\phi \in \Phi$. VNF (ϕ, m) takes input commodities (d, ϕ, n) , $n \in \mathcal{I}(\phi, m)$, and generates commodity (d, ϕ, m) .

where $r^{(\phi,n)}$ now denotes the computation requirement of processing a unit flow of commodity (d, ϕ, n) .

Using the above updated constraints in the LDP bound minimizations performed by the DCNC algorithms, we can provide analogous throughput, cost, and convergence time guarantees for the dynamic control of service trees in cloud networks.

IX. CONCLUSIONS

We addressed the problem of dynamic control of network service chains in distributed cloud networks, in which demands are unknown and time varying. For a given set of services, we characterized the cloud network capacity region and designed online dynamic control algorithms that jointly schedule flow processing and transmission decisions, along with the corresponding allocation of network and cloud resources. The proposed algorithms stabilize the underlying cloud network queuing system, as long as the average input rates are within the cloud network capacity region. The achieved average cloud network costs can be pushed arbitrarily close to minimum with probability 1, while trading off average network delay. Our algorithms converge to within $O(\epsilon)$ of the optimal solutions in time $O(1/\epsilon^2)$. DCNC-L makes local transmission and processing decisions with linear complexity with respect to the number of commodities and resource allocation choices. In comparison, DCNC-Q makes local decisions by minimizing a quadratic metric obtained from an upper bound expression of the LDP function, and we show via simulations that the cost-delay tradeoff can be significantly improved. Furthermore, both DCNC-L and DCNC-Q are enhanced by introducing a STPD bias into the scheduling decisions, yielding the EDCNC-L and EDCNC-Q algorithms, which exhibit further improved delay performance.

REFERENCES

- [1] H. Feng, J. Llorca, A. M. Tulino, and A. F. Molisch, "Dynamic Network Service Optimization in Distributed Cloud Networks," *IEEE INFOCOM SWFAN Workshop*, April 2016.
- [2] H. Feng, J. Llorca, A. M. Tulino, and A. F. Molisch, "Optimal Dynamic Cloud Network Control," *IEEE ICC*, 2016.
- [3] Bell Labs Strategic White Paper, "The Programmable Cloud Network - A Primer on SDN and NFV," June 2013.
- [4] Marcus Weldon, "The Future X Network," *CRC Press*, October 2015.
- [5] L. Lewin-Eytan, J. Naor, R. Cohen, and D. Raz, "Near Optimal Placement of Virtual Network Functions," *IEEE INFOCOM*, 2015.
- [6] M. Barcelo, J. Llorca, A. M. Tulino, and N. Raman, "The Cloud Service Distribution Problem in Distributed Cloud Networks," *IEEE ICC*, 2015.
- [7] L. Georgiadis, M. J. Neely, and L. Tassiulas, "Resource allocation and cross-layer control in wireless networks," *Now Publishers Inc.*, 2006.
- [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, "Stochastic network optimization with application to communication and queuing systems," *Synthesis Lectures on Communication Networks*, Morgan & Claypool, vol. 3, pp. 1–211, 2010.

- [10] L. Tassiulas, and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *IEEE Transactions on Automatic Control*, vol. 37, no. 12, pp. 1936-1948, Dec., 1992.
- [11] M. J. Neely, "A simple convergence time analysis of drift-plus-penalty for stochastic optimization and convex programs," *arXiv preprint arXiv:1412.0791*, 2014.
- [12] M. J. Neely, "Optimal Backpressure Routing for Wireless Networks with Multi-Receiver Diversity", *Ad Hoc Networks*, vol. 7, pp. 862-881, 2009.
- [13] 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.
- [14] S. Supittayapornpong and M. J. Neely, "Quality of information maximization for wireless networks via a fully separable quadratic policy," *IEEE Transactions on Information Theory*, vol. 52, pp. 2915-2934, July, 2006.
- [15] M. Chiang and T. Zhang, "Fog and IoT: An Overview of Research Opportunities", *IEEE Internet of Things Journal*, vol. 3, no. 6, pp. 854-864, Dec. 2016.
- [16] M. Satyanarayanan, P. Bahl, R. Caceres and N. Davies, "The Case for VM-Based Cloudlets in Mobile Computing", *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14-23, Oct.-Dec. 2009.
- [17] 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.
- [18] M. Barcelo, A. Correa, J. Llorca, A. M. Tulino, J. L. Vicario and A. Morell, "IoT-Cloud Service Optimization in Next Generation Smart Environments," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 4077-4090, 2016.
- [19] J. Kleinberg, and E. Tardos, "Algorithm design," *Pearson Education India*, 2006.
- [20] S. Boyd, and L. Vandenberghe, "Convex optimization," *Cambridge university press*, 2004.
- [21] M. J. Neely, "Queue Stability and Probability 1 Convergence via Lyapunov Optimization," *arXiv preprint arXiv:1008.3519*, 2010.
- [22] W. Rudin, "Principles of mathematical analysis," *New York: McGraw-hill*, vol. 3, 1964.
- [23] D. P. Bertsekas, "Convex optimization theory," *Belmont: Athena Scientific*, 2009.

APPENDIX A PROOF OF THEOREM 1

We prove Theorem 1 by separately proving necessary and sufficient conditions.

A. Proof of Necessity

We prove that constraints (6a)-(6h) are required for cloud network stability and that \bar{h}^* given in (7) is the minimum achievable cost by any stabilizing policy.

Consider an input rate matrix $\lambda \in \Lambda(\mathcal{G}, \Phi)$. Then, there exists a stabilizing policy that supports λ . We define the following quantities for this stabilizing policy:

- $X_i^{(d,\phi,m)}(t)$: the number of packets of commodity (d, ϕ, m) exogenously arriving at node i , that got delivered within the first t timeslots
- $F_{i,\text{pr}}^{(d,\phi,m)}(t)$ and $F_{i,\text{pr}}^{(d,\phi,m)}(t)$: the number of packets of commodity (d, ϕ, m) input to and output from the processing unit of node i , that got delivered within the first t timeslots, respectively;
- $F_{ij}^{(d,\phi,m)}(t)$: the number of packets of commodity (d, ϕ, m) transmitted through link (i, j) , that got delivered within the first t timeslots

where we say that a packet of commodity (d, ϕ, m) got delivered within the first t timeslots, if it got processed by functions $\{(\phi, m+1), \dots, (\phi, M_\phi)\}$ and the resulting packet

of the final commodity (d, ϕ, M_ϕ) exited the network at destination d within the first t timeslots.

The above quantities satisfy the following conservation law:

$$\sum_{j \in \delta^-(i)} F_{ji}^{(d,\phi,m)}(t) + F_{\text{pr},i}^{(d,\phi,m)}(t) + X_i^{(d,\phi,m)}(t) = \sum_{j \in \delta^+(i)} F_{ij}^{(d,\phi,m)}(t) + F_{i,\text{pr}}^{(d,\phi,m)}(t), \quad (25)$$

for all nodes and commodities, except for the final commodities at their respective destinations.

Furthermore, we define:

- $\alpha_{i,k}(t)$: the number of timeslots within the first t timeslots, in which k processing resource units were allocated at node i
- $\beta_{i,k}^{(d,\phi,m)}(t)$: the number of packets of commodity (d, ϕ, m) processed by node i during the $\alpha_{i,k}(t)$ timeslots in which k processing resource units were allocated
- $\alpha_{ij,k}(t)$: the number of timeslots within the first t timeslots, in which k transmission resource units were allocated at link (i, j)
- $\beta_{ij,k}^{(d,\phi,m)}(t)$: the number of packets of commodity (d, ϕ, m) transmitted over link (i, j) during the $\alpha_{ij,k}(t)$ timeslots in which k transmission resource units were allocated

It then follows that

$$\frac{F_{i,\text{pr}}^{(d,\phi,m)}(t)}{t} \leq \frac{\alpha_{i,k}(t)}{t} \frac{\beta_{i,k}^{(d,\phi,m)}(t) r^{(\phi,m+1)}}{\alpha_{i,k}(t) C_{i,k}} \frac{C_{i,k}}{r^{(\phi,m+1)}}, \quad \forall i, d, \phi, m < M_\phi, \quad (26)$$

$$\frac{F_{ij}^{(d,\phi,m)}(t)}{t} \leq \frac{\alpha_{ij,k}(t)}{t} \frac{\beta_{ij,k}^{(d,\phi,m)}(t)}{\alpha_{ij,k}(t) C_{ij,k}} C_{ij,k}, \quad \forall (i, j), d, \phi, m, \quad (27)$$

where we define $0/0 = 1$ in case of zero denominator terms.

Note that, for all t , we have

$$0 \leq \frac{\alpha_{i,k}(t)}{t} \leq 1, \quad 0 \leq \frac{\beta_{i,k}^{(d,\phi,m)}(t) r^{(\phi,m+1)}}{\alpha_{i,k}(t) C_{i,k}} \leq 1, \quad (28)$$

$$0 \leq \frac{\alpha_{ij,k}(t)}{t} \leq 1, \quad 0 \leq \frac{\beta_{ij,k}^{(d,\phi,m)}(t)}{\alpha_{ij,k}(t) C_{ij,k}} \leq 1. \quad (29)$$

In addition, let \bar{h} represent the liminf of the average cost achieved by this policy:

$$\bar{h} \triangleq \liminf_{t \rightarrow \infty} \frac{1}{t} \sum_{\tau=0}^{t-1} h(\tau). \quad (30)$$

Then, due to Boltzono-Weierstrass theorem [22] on a compact set, there exists an infinite subsequence $\{t_u\} \subseteq \{t\}$ such that

$$\lim_{t_u \rightarrow \infty} \frac{1}{t_u} \sum_{\tau=0}^{t_u-1} h(\tau) = \bar{h}, \quad (31)$$

the left hand of (26) and (27) converge to $f_{i,\text{pr}}^{(d,\phi,m)}$ and $f_{ij}^{(d,\phi,m)}$:

$$\lim_{t_u \rightarrow \infty} \frac{F_{i,\text{pr}}^{(d,\phi,m)}(t_u)}{t_u} = f_{i,\text{pr}}^{(d,\phi,m)}, \quad \lim_{t_u \rightarrow \infty} \frac{F_{ij}^{(d,\phi,m)}(t_u)}{t_u} = f_{ij}^{(d,\phi,m)}, \quad (32)$$

and the terms in (28) and (29) converge to $\alpha_{i,k}$, $\beta_{i,k}$, $\alpha_{ij,k}$, and $\beta_{ij,k}$:

$$\lim_{t_u \rightarrow \infty} \frac{\alpha_{i,k}(t_u)}{t_u} = \alpha_{i,k}, \quad \lim_{t_u \rightarrow \infty} \frac{\beta_{i,k}^{(d,\phi,m)}(t_u) r^{(\phi,m+1)}}{\alpha_{i,k}(t_u) C_{i,k}} = \beta_{i,k}, \quad (33)$$

$$\lim_{t_u \rightarrow \infty} \frac{\alpha_{ij,k}(t_u)}{t_u} = \alpha_{ij,k}, \quad \lim_{t_u \rightarrow \infty} \frac{\beta_{ij,k}^{(d,\phi,m)}(t_u)}{\alpha_{ij,k}(t_u) C_{ij,k}} = \beta_{ij,k}. \quad (34)$$

from which (6g) and (6h) follow.

Plugging (32), (33), and (34) respectively back into (26) and (27), letting $t_u \rightarrow \infty$ yields

$$f_{i,\text{pr}}^{(d,\phi,m)} \leq \frac{1}{r^{(\phi,m+1)}} \alpha_{i,k} \beta_{i,k}^{(d,\phi,m)} C_{i,k}, \quad (35)$$

$$f_{ij}^{(d,\phi,m)} \leq \alpha_{ij,k} \beta_{ij,k}^{(d,\phi,m)} C_{ij,k}, \quad (36)$$

from which (6c) and (6d) follow.

Furthermore, due to cloud network stability, we have

$$\lim_{t \rightarrow \infty} \frac{\sum_{\tau=0}^t a_i^{(d,\phi,m)}(t)}{t} = \lim_{t \rightarrow \infty} \frac{X_i^{(d,\phi,m)}(t)}{t} = \lambda_i^{(d,\phi,m)}, \quad \text{w.p.1, } \forall i, d, \phi, m, \quad (37)$$

and

$$\begin{aligned} \lim_{t_u \rightarrow \infty} \frac{F_{\text{pr},i}^{(d,\phi,m)}(t_u)}{t_u} &= \lim_{t_u \rightarrow \infty} \frac{\xi^{(\phi,m)} F_{i,\text{pr}}^{(d,\phi,m-1)}(t_u)}{t_u} \\ &= \xi^{(\phi,m)} f_{i,\text{pr}}^{(d,\phi,m-1)} \\ &\triangleq f_{\text{pr},i}^{(d,\phi,m)}, \quad \text{w.p.1, } \forall i, d, \phi, m, \end{aligned} \quad (38)$$

from which (6b) follows.

Evaluating (25) in $\{t_u\}$, dividing by t_u , sending t_u to ∞ , and using (32), (37), and (38), Eq. (6a) follows.

Finally, from (5), and using the quantities defined at the beginning of this section, we have

$$\begin{aligned} &\frac{1}{t_u} \sum_{\tau=0}^{t_u-1} h(\tau) \\ &= \sum_i \sum_{k \in \mathcal{K}_i} \left[\frac{\alpha_{i,k}(t_u) w_{i,k}}{t_u} + \sum_{(d,\phi,m)} \frac{r^{(\phi,m+1)} \beta_{i,k}^{(d,\phi,m)}(t_u) e_i}{t_u} \right] \\ &+ \sum_{(i,j)} \sum_{k \in \mathcal{K}_{ij}} \left[\frac{\alpha_{ij,k}(t_u) w_{ij,k}}{t_u} + \sum_{(d,\phi,m)} \frac{\beta_{ij,k}^{(d,\phi,m)}(t_u) e_{ij}}{t_u} \right] \\ &= \sum_i \sum_{k \in \mathcal{K}_i} \left[\frac{\alpha_{i,k}(t_u) w_{i,k}}{t_u} + \frac{\alpha_{i,k}(t_u)}{t_u} \sum_{(d,\phi,m)} \frac{r^{(\phi,m+1)} \beta_{i,k}^{(d,\phi,m)}(t_u) C_{i,k} e_i}{\alpha_{i,k}(t_u) C_{i,k}} \right] \\ &+ \sum_{(i,j)} \sum_{k \in \mathcal{K}_{ij}} \left[\frac{\alpha_{ij,k}(t_u) w_{ij,k}}{t_u} + \frac{\alpha_{ij,k}(t_u)}{t_u} \sum_{(d,\phi,m)} \frac{\beta_{ij,k}^{(d,\phi,m)}(t_u) C_{ij,k} e_{ij}}{\alpha_{ij,k}(t_u) C_{ij,k}} \right]. \end{aligned} \quad (39)$$

Letting $t_u \rightarrow \infty$, we obtain (8). Finally, (7) follows from taking the minimum over all stabilizing policies.

B. Proof of Sufficiency

Given an input rate matrix $\lambda \triangleq \{\lambda_i^{(d,\phi,m)}\}$, if there exists a constant $\kappa > 0$ such that input rate $\{\lambda_i^{(d,\phi,m)} + \kappa\}$, together

with probability values $\alpha_{ij,k}$, $\alpha_{i,k}$, $\beta_{ij,k}^{(d,\phi,m)}$, $\beta_{i,k}^{(d,\phi,m)}$, and flow variables $f_{ij}^{(d,\phi,m)}$, $f_{i,\text{pr}}^{(d,\phi,m)}$, satisfy (6a)-(6h), we can construct a stationary randomized policy that uses these probabilities to make scheduling decisions, which yields the mean rates:

$$\mathbb{E} \left\{ \mu_{i,\text{pr}}^{(d,\phi,m)}(t) \right\} = f_{i,\text{pr}}^{(d,\phi,m)}, \quad \mathbb{E} \left\{ \mu_{ij}^{(d,\phi,m)}(t) \right\} = f_{ij}^{(d,\phi,m)}. \quad (40)$$

Plugging (40) and $\{\lambda_i^{(d,\phi,m)} + \kappa\}$ in (6a), we have

$$\begin{aligned} &\mathbb{E} \left\{ \sum_{j \in \delta^+(i)} \mu_{ij}^{(d,\phi,m)}(t) + \mu_{i,\text{pr}}^{(d,\phi,m)}(t) - \sum_{j \in \delta^-(i)} \mu_{ji}^{(d,\phi,m)}(t) \right. \\ &\quad \left. - \xi^{(d,\phi,m+1)} \mu_{i,\text{pr}}^{(d,\phi,m)}(t) - a_i^{(d,\phi,m)}(t) \right\} \geq \kappa. \end{aligned} \quad (41)$$

By applying standard Lyapunov drift manipulations [7], we upper bound the Lyapunov drift $\Delta(\mathbf{Q}(t))$ (see Sec. V-A) as

$$\begin{aligned} \Delta(\mathbf{Q}(t)) &\leq NB_0 + \sum_{(d,\phi,m),i} Q_i^{(d,\phi,m)}(t) \mathbb{E} \left\{ \sum_{j \in \delta^-(i)} \mu_{ji}^{(d,\phi,m)}(t) \right. \\ &\quad \left. + \mu_{\text{pr},i}^{(d,\phi,m)}(t) - \sum_{j \in \delta^+(i)} \mu_{ij}^{(d,\phi,m)}(t) + \mu_{\text{pr},i}^{(d,\phi,m)}(t) + a_i^{(d,\phi,m)}(t) \right\} \\ &\leq NB_0 - \kappa \sum_{(d,\phi,m),i} Q_i^{(d,\phi,m)}(t), \end{aligned} \quad (42)$$

where B_0 is a constant that depends on the system parameters. With some additional manipulations, it follows from (42) that the cloud network is strongly stable, *i.e.*, the total mean average backlog is upper bounded. Therefore, $\{\lambda_i^{(d,\phi,m)}\}$ is interior to $\Lambda(\mathcal{G}, \Phi)$ (due to the existence of κ).

APPENDIX B PROOF OF THEOREM 2

We prove Theorem 2 for each DCNC algorithm by manipulating the linear term $Z(t)$ and the quadratic term $\Gamma(t)$ in the LDP upper bound expression given in (11).

A. DCNC-L

We upper bound $\Gamma(t)$ in (11) as follows:

$$\begin{aligned} \Gamma(t) &\leq \frac{1}{2} N \left[(\delta_{\max} C_{\text{tr}}^{\max} + C_{\text{pr}}^{\max} / r_{\min})^2 + \right. \\ &\quad \left. (\delta_{\max} C_{\text{tr}}^{\max} + \xi_{\max} C_{\text{pr}}^{\max} / r_{\min} + A_{\max})^2 \right] \triangleq NB_0. \end{aligned} \quad (43)$$

Plugging (43) into (11) yields

$$\begin{aligned} &\Delta(\mathbf{Q}(t)) + V \mathbb{E} \{ h(t) | \mathbf{Q}(t) \} \leq NB_0 \\ &+ \mathbb{E} \{ V h(t) + Z(t) | \mathbf{Q}(t) \} + \sum_{(d,\phi,m),i} \lambda_i^{(d,\phi,m)} Q_i^{(d,\phi,m)}(t). \end{aligned} \quad (44)$$

Since $\lambda \triangleq \{\lambda_i^{(d,\phi,m)}\}$ is interior to $\Lambda(\mathcal{G}, \Phi)$, there exists a positive number κ such that $\lambda + \kappa \mathbf{1} \in \Lambda$. According to (12), DCNC-L minimizes $Vh(t) + Z(t)$ among all policies subject to (4d)-(4g). We use $*$ to identify the stationary randomized policy that supports $\lambda + \kappa \mathbf{1}$ and achieves average cost $\bar{h}^*(\lambda + \kappa \mathbf{1})$, characterized by Theorem 1. The LDP function under DCNC-L can be further upper bounded as

$$\begin{aligned} &\Delta(\mathbf{Q}(t)) + V \mathbb{E} \{ h(t) | \mathbf{Q}(t) \} \leq NB_0 \\ &+ \mathbb{E} \{ V h^* + Z^*(t) | \mathbf{Q}(t) \} + \sum_{(d,\phi,m),i} \lambda_i^{(d,\phi,m)} Q_i^{(d,\phi,m)}(t) \end{aligned}$$

$$\begin{aligned}
&= NB_0 + V\bar{h}^*(\boldsymbol{\lambda} + \kappa\mathbf{1}) \\
&\quad + \sum_{(d,\phi,m),i} Q_i^{(d,\phi,m)}(t) \left[\sum_{j \in \delta^-(i)} f_{ji}^{*(d,\phi,m)} + f_{pr,i}^{*(d,\phi,m)} \right. \\
&\quad \quad \left. - \sum_{j \in \delta^+(i)} f_{ij}^{*(d,\phi,m)} + f_{pr,i}^{*(d,\phi,m)} + \lambda_i^{(d,\phi,m)} \right] \\
&\leq NB_0 + V\bar{h}^*(\boldsymbol{\lambda} + \kappa\mathbf{1}) - \kappa \sum_{(d,\phi,m),i} Q_i^{(d,\phi,m)}(t). \quad (45)
\end{aligned}$$

where the last inequality holds true due to (4b).

B. DCNC-Q

We extract the quadratic terms $(\mu_{ij}^{(d,\phi,m)}(t))^2$ and $(\mu_{i,pr}^{(d,\phi,m)}(t))^2$ by decomposing $\Gamma(t)$ as follows:

$$\Gamma(t) = \Gamma_{tr}(t) + \Gamma_{pr}(t) + \Gamma'(t), \quad (46)$$

$$\begin{aligned}
&\text{where} \\
\Gamma_{tr}(t) &\triangleq \sum_{(i,j)} \sum_{(d,\phi,m)} \left(\mu_{ij}^{(d,\phi,m)}(t) \right)^2 \\
\Gamma_{pr}(t) &\triangleq \frac{1}{2} \sum_{(d,\phi,m),i} \left[\left(\mu_{i,pr}^{(d,\phi,m)}(t) \right)^2 + \left(\mu_{pr,i}^{(d,\phi,m)}(t) \right)^2 \right]; \\
\Gamma'(t) &\triangleq \sum_{(d,\phi,m),i} \left\{ \mu_{i,pr}^{(d,\phi,m)}(t) \sum_{j \in \delta(i)} \mu_{ij}^{(d,\phi,m)}(t) + \right. \\
&\quad \sum_{j,v:j,v \in \delta(i), v \neq j} \mu_{ij}^{(d,\phi,m)}(t) \mu_{iv}^{(d,\phi,m)}(t) + \mu_{pr,i}^{(d,\phi,m)}(t) \sum_{j \in \delta(i)} \mu_{ji}^{(d,\phi,m)}(t) \\
&\quad \left. + \sum_{j,v:j,v \in \delta(i), v \neq j} \mu_{ji}^{(d,\phi,m)}(t) \mu_{vi}^{(d,\phi,m)}(t) + \frac{1}{2} \left(a_i^{(d,\phi,m)} \right)^2 \right. \\
&\quad \left. + a_i^{(d,\phi,m)}(t) \left[\sum_{j \in \delta(i)} \mu_{ji}^{(d,\phi,m)}(t) + \mu_{pr,i}^{(d,\phi,m)}(t) \right] \right\}.
\end{aligned}$$

According to (13), DCNC-Q minimizes the metric $\Gamma_{tr}(t) + \Gamma_{pr}(t) + Z(t) + Vh(t)$ among all policies subject to (4d)-(4g). Hence, the LDP function under DCNC-Q can be further upper bounded as

$$\begin{aligned}
&\Delta(\mathbf{Q}(t)) + V\mathbb{E}\{h(t)|\mathbf{Q}(t)\} \\
&\leq \mathbb{E}\{\Gamma'(t) + \Gamma_{tr}^* + \Gamma_{pr}^* | \mathbf{Q}(t)\} + V\bar{h}^*(\boldsymbol{\lambda} + \kappa\mathbf{1}) \\
&\quad + \mathbb{E}\{Z^*(t) | \mathbf{Q}(t)\} + \sum_{(d,\phi,m),i} \lambda_i^{(d,\phi,m)} Q_i^{(d,\phi,m)}(t). \quad (47)
\end{aligned}$$

On the other hand, note that

$$\begin{aligned}
&\Gamma'(t) + \Gamma_{tr}^* + \Gamma_{pr}^* \\
&\leq N \left[(1 + \xi_{\max}) C_{pr}^{\max} \delta_{\max} C_{tr}^{\max} / r_{\min} + (\delta_{\max} - 1) \delta_{\max} (C_{tr}^{\max})^2 \right. \\
&\quad \left. + A_{\max} (\delta_{\max} C_{tr}^{\max} + \xi_{\max} C_{pr}^{\max} / r_{\min}) + \frac{1}{2} (A_{\max})^2 \right] \\
&\quad + N \delta_{\max} (C_{tr}^{\max})^2 + \frac{1}{2(r_{\min})^2} N (C_{pr}^{\max})^2 \left[1 + (\xi_{\max})^2 \right] \\
&= \frac{1}{2} N \left[(\delta_{\max} C_{tr}^{\max} + C_{pr}^{\max} / r_{\min})^2 \right. \\
&\quad \left. + (\delta_{\max} C_{tr}^{\max} + \xi_{\max} C_{pr}^{\max} / r_{\min} + A_{\max})^2 \right] = NB_0. \quad (48)
\end{aligned}$$

Plugging (48) into (47) yield

$$\begin{aligned}
&\Delta(\mathbf{Q}(t)) + V\mathbb{E}\{h(t)|\mathbf{Q}(t)\} \\
&\leq NB_0 + \bar{h}^*(\boldsymbol{\lambda} + \kappa\mathbf{1}) - \kappa \sum_{(d,\phi,m),i} Q_i^{(d,\phi,m)}(t). \quad (49)
\end{aligned}$$

C. EDCNC-L and EDCNC-Q

Using (14) in (1), and following standard LDP manipulations (see Ref. [9]), the LDP function can be upper bounded as follows:

$$\begin{aligned}
&\Delta(\mathbf{Q}(t)) + V\mathbb{E}\{h(t)|\mathbf{Q}(t)\} \leq -\eta\Upsilon(t) \\
&\quad + NB_0 + \bar{h}^*(\boldsymbol{\lambda} + \kappa\mathbf{1}) - \kappa \sum_{(d,\phi,m),i} \hat{Q}_i^{(d,\phi,m)}(t), \quad (50)
\end{aligned}$$

where

$$\begin{aligned}
\Upsilon(t) &\triangleq \sum_{(d,\phi,m),i} Y_i^{(d,\phi,m)} \mathbb{E} \left\{ \sum_{j \in \delta^-(i)} \mu_{ji}^{(d,\phi,m)}(t) + a_i^{(d,\phi,m)}(t) \right. \\
&\quad \left. + \mu_{pr,i}^{(d,\phi,m)}(t) - \sum_{j \in \delta^+(i)} \mu_{ij}^{(d,\phi,m)}(t) - \mu_{i,pr}^{(d,\phi,m)}(t) \middle| \mathbf{Q}(t) \right\}. \quad (51)
\end{aligned}$$

Denote $Y_{\max} \triangleq \max_{i,(d,\phi,m)} Y_i^{(d,\phi,m)}$, which satisfies $Y_{\max} \leq \max_{i,j} \{H_{i,j}\} \leq N - 1$. Then, following (51), we lower bound $\Upsilon(t)$ as

$$\Upsilon(t) \geq -N [\delta_{\max} C_{tr}^{\max} + C_{pr}^{\max} / r_{\min}] \triangleq -NB_{\Upsilon}. \quad (52)$$

Plugging (52) into (50) and using $\hat{Q}_i^{(d,\phi,m)}(t) \geq Q_i^{(d,\phi,m)}(t)$ yields

$$\begin{aligned}
&\Delta(\mathbf{Q}(t)) + V\mathbb{E}\{h(t)|\mathbf{Q}(t)\} \\
&\leq NB_1 + \bar{h}^*(\boldsymbol{\lambda} + \kappa\mathbf{1}) - \kappa \sum_{(d,\phi,m),i} Q_i^{(d,\phi,m)}(t), \quad (53)
\end{aligned}$$

where $B_1 \triangleq B_0 + \eta B_{\Upsilon}$.

D. Network Stability and Average Cost Convergence with Probability 1

We can use the theoretical result in [21] for the proof of network stability and average cost convergence with probability 1. Note that the following bounding conditions are satisfied in the cloud network system:

- The second moment $\mathbb{E}\{(h(t))^2\}$ is upper bounded by $(\sum_{ij} w_{ij, K_{ij}} + \sum_i w_{i, K_i})^2$ and therefore satisfies

$$\sum_{\tau=0}^{\infty} \mathbb{E}\{(h(\tau))^2\} / \tau < \infty. \quad (54)$$

- $\mathbb{E}\{h(t)|\mathbf{Q}(t)\}$ is lower bounded as

$$\mathbb{E}\{h(t)|\mathbf{Q}(t)\} \geq 0. \quad (55)$$

- For all $i, (d, \phi, m)$, and t , the conditional fourth moment of backlog dynamics satisfies

$$\begin{aligned}
&\mathbb{E} \left\{ \left[Q_i^{(d,\phi,m)}(t+1) - Q_i^{(d,\phi,m)}(t) \right]^4 \middle| \mathbf{Q}(t) \right\} \\
&\leq (\delta_{\max} C_{tr}^{\max} + \xi_{\max} C_{pr}^{\max} / r_{\min} + A_{\max})^4 < \infty. \quad (56)
\end{aligned}$$

With (54)-(56), based on the derivations in [21], Eq. (45), (49), and (53) lead to network stability (17) and average cost (16) convergence with probability 1 under DCNC-L, DCNC-Q, EDCNC-L(Q), respectively.

APPENDIX C PROOF OF THEOREM 3

Let's first prove Eq. (21). To this end denote $\bar{h}(t) \triangleq \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E}\{h(\tau)\}$. Then, under the DCNC policy and after some algebraic manipulations similar to the ones used for (45), we upper bound the LDP function as follows:

$$\Delta(\mathbf{Q}(t)) + V\mathbb{E}\{h(t)|\mathbf{Q}(t)\} \leq NB + V\bar{h}^*(\boldsymbol{\lambda}), \quad (57)$$

where $\bar{h}^*(\boldsymbol{\lambda})$ is the minimum average cost given $\boldsymbol{\lambda}$. Taking the expectation over $\mathbf{Q}(t)$ on both sides of (57) and summing over $\tau = 0, \dots, t-1$ further yields

$$\frac{1}{2t} \left[\mathbb{E} \left\{ \|\mathbf{Q}(t)\|^2 \right\} - \mathbb{E} \left\{ \|\mathbf{Q}(0)\|^2 \right\} \right] \leq NB + V \left[\bar{h}^*(\boldsymbol{\lambda}) - \bar{h}(t) \right]. \quad (58)$$

Then it follows that, by setting $V = 1/\epsilon$ and for all $t \geq 1$,

$$\begin{aligned} \bar{h}(t) - \bar{h}^*(\boldsymbol{\lambda}) &\leq \frac{NB}{V} + \frac{1}{2Vt} \mathbb{E} \left\{ \|\mathbf{Q}(0)\|^2 \right\} \\ &\leq \left[NB + \frac{1}{2} \mathbb{E} \left\{ \|\mathbf{Q}(0)\|^2 \right\} \right] \epsilon, \end{aligned} \quad (59)$$

which proves (21).

In order to prove (22), we first introduce the following quantities for an arbitrary policy:

- $\mathbf{y}(t)$: the vector of elements $y_i(t)$ and $y_{ij}(t)$;
- $\tilde{\boldsymbol{\mu}}(t)$: the vector of actual flow rate elements $\tilde{\mu}_{i,\text{pr}}^{(d,\phi,m)}(t)$, $\tilde{\mu}_{\text{pr},i}^{(d,\phi,m)}(t)$, and $\tilde{\mu}_{ij}^{(d,\phi,m)}(t)$;
- $\tilde{\mathbf{x}}(t)$: the vector $[\mathbf{y}(t); \tilde{\boldsymbol{\mu}}(t)]$;
- $\Delta \mathbf{f}(t)$: the vector of elements $\Delta f_i^{(d,\phi,m)}(t)$ as in (20).

Summing both sides of (20) over $\tau = 0, 1, \dots, t-1$ and then dividing them by t , for all $i, d, \phi, m, t \geq 1$, yield,

$$\overline{\Delta \mathbf{f}}(t) \triangleq \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \{ \Delta \mathbf{f}(\tau) \} = \frac{1}{t} \mathbb{E} \{ \mathbf{Q}(t) - \mathbf{Q}(0) \}. \quad (60)$$

Lemma C.1. *If $\boldsymbol{\lambda}$ is interior to $\Lambda(\mathcal{G}, \Phi)$, there exists a constant vector $\boldsymbol{\rho}$ such that*

$$\bar{h}^*(\boldsymbol{\lambda}) - \bar{h}(t) \leq \boldsymbol{\rho}^\dagger \overline{\Delta \mathbf{f}}(t). \quad (61)$$

Proof. The proof is given in Appendix D \square

Plugging (60) into (61) yields

$$\begin{aligned} \bar{h}^*(\boldsymbol{\lambda}) - \bar{h}(t) &\leq \frac{1}{t} \boldsymbol{\rho}^\dagger \mathbb{E} \{ \mathbf{Q}(t) - \mathbf{Q}(0) \} \\ &\leq \frac{1}{t} \|\boldsymbol{\rho}\| \cdot (\|\mathbb{E}\{\mathbf{Q}(t)\}\| + \|\mathbb{E}\{\mathbf{Q}(0)\}\|), \end{aligned} \quad (62)$$

Under the DCNC policy, by further plugging (62) into the right hand side of (58), we have

$$\begin{aligned} &\frac{1}{2t} \left(\mathbb{E} \left\{ \|\mathbf{Q}(t)\|^2 \right\} - \mathbb{E} \left\{ \|\mathbf{Q}(0)\|^2 \right\} \right) \\ &\leq NB + V \frac{\|\boldsymbol{\rho}\|}{t} (\|\mathbb{E}\{\mathbf{Q}(t)\}\| + \|\mathbb{E}\{\mathbf{Q}(0)\}\|). \end{aligned} \quad (63)$$

By using the fact $\mathbb{E}\{\|\mathbf{Q}(t)\|^2\} \geq \|\mathbb{E}\{\mathbf{Q}(t)\}\|^2$ in (63), it follows that

$$\begin{aligned} &\|\mathbb{E}\{\mathbf{Q}(t)\}\|^2 - 2V \|\boldsymbol{\rho}\| \cdot \|\mathbb{E}\{\mathbf{Q}(t)\}\| - 2NBt \\ &- \mathbb{E}\{\|\mathbf{Q}(0)\|^2\} - 2V \|\boldsymbol{\rho}\| \cdot \|\mathbb{E}\{\mathbf{Q}(0)\}\| \leq 0. \end{aligned} \quad (64)$$

The largest value of $\|\mathbb{E}\{\mathbf{Q}(t)\}\|$ that satisfies (64) is given by

$$\begin{aligned} &\|\mathbb{E}\{\mathbf{Q}(t)\}\| \leq V \|\boldsymbol{\rho}\| + \\ &\sqrt{V^2 \|\boldsymbol{\rho}\|^2 + 2NBt + \mathbb{E}\{\|\mathbf{Q}(0)\|^2\} + 2V \|\boldsymbol{\rho}\| \cdot \|\mathbb{E}\{\mathbf{Q}(0)\}\|} \\ &\leq V \|\boldsymbol{\rho}\| + \sqrt{(V \|\boldsymbol{\rho}\| + \mathbb{E}\{\|\mathbf{Q}(0)\|\})^2 + 2NBt + \text{var}\{\|\mathbf{Q}(0)\|\}}. \end{aligned} \quad (65)$$

Finally, by setting $V = 1/\epsilon$ and $t = 1/\epsilon^2$, we plug (65) back into the right hand side of (60) and obtain

$$\begin{aligned} &\frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \left\{ \Delta f_i^{(d,\phi,m)}(\tau) \right\} \leq \frac{1}{t} (\|\mathbb{E}\{\mathbf{Q}(t)\}\| + \|\mathbb{E}\{\mathbf{Q}(0)\}\|) \\ &\leq \frac{V \|\boldsymbol{\rho}\|}{t} + \frac{\|\mathbb{E}\{\mathbf{Q}(0)\}\|}{t} \\ &\quad + \frac{1}{t} \sqrt{(V \|\boldsymbol{\rho}\| + \mathbb{E}\{\|\mathbf{Q}(0)\|\})^2 + 2NBt + \text{var}\{\|\mathbf{Q}(0)\|\}} \\ &\leq \left(2\mathbb{E}\{\|\mathbf{Q}(0)\|\} + \sqrt{\text{var}\{\|\mathbf{Q}(0)\|\}} \right) \epsilon^2 + \left(\sqrt{2NB} + 2 \|\boldsymbol{\rho}\| \right) \epsilon, \end{aligned} \quad (66)$$

which proves (22).

APPENDIX D PROOF OF LEMMA C.1

Given an arbitrary policy, define

- $\boldsymbol{\mu}(t)$: the vector of assigned flow rate elements $\mu_{i,\text{pr}}^{(d,\phi,m)}(t)$, $\mu_{\text{pr},i}^{(d,\phi,m)}(t)$, and $\mu_{ij}^{(d,\phi,m)}(t)$;
- $\mathbf{x}(t)$: the vector $[\mathbf{y}(t); \mathbf{x}(t)]$.

With a little bit abuse of notation, denote $h(\mathbf{x}(t)) \triangleq h(t)$; $\Delta \mathbf{f}(\tilde{\mathbf{x}}(t)) \triangleq \Delta \mathbf{f}(t)$. In addition, let \mathcal{X} represent the set of all possible vectors $\mathbf{x}(t)$ that satisfy the constraints (4c)-(4g). Note that $\tilde{\mathbf{x}}(t)$ also belongs to \mathcal{X} . Furthermore, let $\overline{\mathcal{X}}$ represent the convex hull of \mathcal{X} . Then, for all vectors $\mathbf{x} \in \overline{\mathcal{X}}$, the following conditions are satisfied:

- 1) $h(\mathbf{x}) - \bar{h}^*(\boldsymbol{\lambda})$ and $\Delta \mathbf{f}(\mathbf{x})$ are convex for all $\mathbf{x} \in \overline{\mathcal{X}}$;
- 2) $h(\mathbf{x}) - \bar{h}^*(\boldsymbol{\lambda}) \geq 0$ for all $\mathbf{x} \in \overline{\mathcal{X}}$ with $\Delta \mathbf{f}(\mathbf{x}) \leq 0$;
- 3) $\exists \tilde{\mathbf{x}} \in \overline{\mathcal{X}}$ with $\Delta \mathbf{f}(\tilde{\mathbf{x}}) \prec 0$, given $\boldsymbol{\lambda}$ interior to $\Lambda(\mathcal{G}, \Phi)$.

Item 2) above results immediately from Theorem 1, where any $\mathbf{x} \in \overline{\mathcal{X}}$ with $\Delta \mathbf{f}(\mathbf{x}) \leq 0$ can be treated as the $\mathbb{E}\{\mathbf{x}(t)\}$ under a stabilizing stationary randomized policy. Hence, according to Farkas' Lemma [23], there exists a constant vector $\boldsymbol{\rho} \succeq 0$ such that

$$h(\mathbf{x}) - \bar{h}^*(\boldsymbol{\lambda}) + \boldsymbol{\rho}^\dagger \Delta \mathbf{f}(\mathbf{x}) \geq 0, \quad \forall \mathbf{x} \in \overline{\mathcal{X}}. \quad (67)$$

Evaluating (67) in $\tilde{\mathbf{x}}(\tau)$ with $\tau = 0, \dots, t-1$, we have

$$\frac{1}{t} \sum_{\tau=0}^{t-1} h(\tilde{\mathbf{x}}(\tau)) - \bar{h}^*(\boldsymbol{\lambda}) + \frac{1}{t} \boldsymbol{\rho}^\dagger \sum_{\tau=0}^{t-1} \Delta \mathbf{f}(\tilde{\mathbf{x}}(\tau)) \geq 0, \quad (68)$$

from which it follows that

$$\begin{aligned} \boldsymbol{\rho}^\dagger \overline{\Delta \mathbf{f}}(t) &= \frac{1}{t} \boldsymbol{\rho}^\dagger \sum_{\tau=0}^{t-1} \mathbb{E} \{ \Delta \mathbf{f}(\tilde{\mathbf{x}}(\tau)) \} \geq \bar{h}^*(\boldsymbol{\lambda}) - \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \{ h(\tilde{\mathbf{x}}(\tau)) \} \\ &\stackrel{(a)}{\geq} \bar{h}^*(\boldsymbol{\lambda}) - \frac{1}{t} \sum_{\tau=0}^{t-1} \mathbb{E} \{ h(\mathbf{x}(\tau)) \} = \bar{h}^*(\boldsymbol{\lambda}) - \bar{h}(t), \end{aligned} \quad (69)$$

where the inequality (a) is due to $h(\tilde{\mathbf{x}}(t)) \leq h(\mathbf{x}(t))$ that results from the fact $\tilde{\boldsymbol{\mu}}(t) \preceq \boldsymbol{\mu}(t)$.