A Novel Data Center Network Architecture with Zero in-Network Queuing

Tara Javidi, Chang-Heng Wang, and Tugcan Aktaş
Department of Electrical and Computer Engineering
University of California, San Diego
Email: {tjavidi, chw009, taktas}@ucsd.edu

***Abstract*—In-network queuing in the Internet-style networks enables the distributed operation and scalability across the network at the cost of excessive delay and tardy flow completion times. Data center networking, in contrast, are proposed to depart from this classical approach and avoid in-network queuing all together. In this new class of network solutions serving inter-data center traffic, a densely packed fairly local area network of stationary end hosts are often managed by a single entity, allowing for fine-grained management and scheduling of flows across the data center.**

The overall objective of this work is to develop a framework, from first principles, which relies on the unique attributes of data centers to propose a transformative novel networking architecture with increased level of efficiency and significantly smaller latency. By separating the control and data planes, the proposed hybrid architecture avoids in-network queuing and results in significantly lower delay. The critical technical challenge is to design end-end circuit switching mechanisms that account for monitoring as well as circuit reconfiguration delays. Furthermore, the design has to minimize the computational complexity of the scheduling algorithm as well as the cost of monitoring across the network. In this context, this paper underlines a family of recent technologies and networking advances as promising enablers and discusses the most significant set of challenges.

I. Introduction

As it has been recently observed in the networking literature, the data center networking must significantly depart from classical and Internet-inherited networking in order to allow fine-grain management and scheduling of the flows of data [1], [2]. This departure from classical networking has been motivated by the strict low-latency requirements of the applications and the feasibility of large-scale yet centralized management of the network. In other words, any given data center is managed, more or less, by a single entity whose objective is to organize densely packed servers (end nodes) into a highly optimized large-scale distributed and parallel computation infrastructure.

In this position paper, we propose a framework, from first principles, which relies on the above unique attributes of data centers as well as technological advances in various aspects of networking, to propose a potentially transformative novel architecture with increased level of efficiency and significantly smaller latency. Our proposed architecture shares important attributes with recent networking solutions such as pFabric [1] and Fastpass [2] that aim at (near-)zero in-network queuing. However, our framework sharply deviates from prior work on (near-)zero in-network queuing in that we propose a hybrid architecture consisting of physically distinct monitoring/control and data planes. The physical separation and decoupling of the monitoring/control plane from the data plane allows for the optimization of the attributes of each component of the network. To start, we envision a centralized controller that exercises (very) tight control over end-end flows in form of a dynamic (fine-grained) circuit switching in the data plane: which top of rack (ToR) switch can send packets and what paths packets take. The proposed dynamic fine-grained circuit switching matches the flow rates to the available network capacity at the time scales of the monitoring and control instead of matching the rates over longer time-scales as it is done with distributed congestion control.

This means that the optimized operation of the data plane depends on how tight the monitoring and control of the centralized scheduler is relative to the dynamics of the traffic demand across the network. Hence, the second component of the proposed framework is a dedicated (and physically distinct) network providing secure, reliable, and ultra-low latency channel from ToR and core switches to and from the centralized controller. In other words, we push the monitoring and control functionalities (which are critical for fine-grained dynamic circuit switching) away from data-plane into an entirely separate network which is optimized for ultra-low-latency operation of monitoring the traffic demands and control of switches.

The remainder of the paper is organized as follows. In Section II, we elaborate on the proposed architecture. We detail the network architecture as well as the temporal processes enabling the control and operation of the network. This rises to a generalization of the concept of cross-bar scheduling with non-negligible reconfiguration time. In Section III, we address the feasibility of the proposed architecture by discussing a variety of technology enablers: open-flow switches, optical-switches, and wireless radio access technologies. In Section **??**, we discuss some of the major challenges in scaling the proposed solutions and discuss areas of future work.

II. System Model: Separation of Monitoring/Control and Data Planes

A. Network Architecture

As shown in Fig. 1, we consider a set of N top of rack (ToR) switches, labeled by $\{1, 2, \ldots, N\}$, which are interconnected by a network. Each ToR switch can serve as a source

978-3-9018-8273-9/15/ ©2015 IFIP

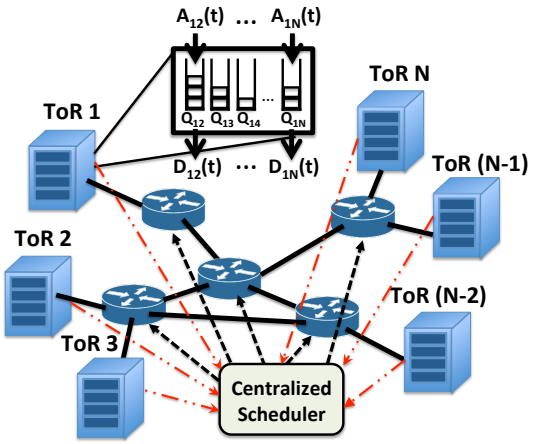

Fig. 1. Network Architecture: Reminiscent of SDN architecture, the proposed solution relies on a separation of monitoring, control, and data planes. The proposed separation, however, goes beyond logical separation and is achieved via physically separated networks.

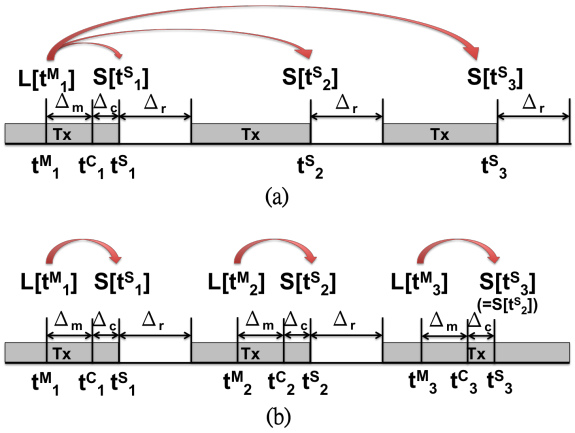

Fig. 2. Timing diagram for different scheduling strategies. (a) Quasi-static monitoring: Series of schedules determined in a single schedule computation, and some schedules could depend on out-dated queue information when being deployed. (b) Active monitoring: Each schedule is computed based on the most up-to-date edge queue information.

and a destination simultaneously. We aim for no queuing in the core network, hence all the queuing occurs in the edge of the network, i.e. within the ToR switches. Each ToR switch maintains $N-1$ edge queues (either physically or virtually), which are denoted by Q_{ij}, where $j \in \{1,2,\ldots,N\}\backslash\{i\}$. Packets going from the ToR switch i to j are enqueued in the edge queue Q_{ij} before transmission.

Let $\mathbf{S}(t) \in \{0,1\}^{N\times N}$ denote the end-to-end connectivity (also known as schedule) at t, which indicates the circuits established between the ToR switches. Accordingly, $S_{ij}(t)=1$ indicates that a circuit from ToR i to ToR j exists at time t, and $S_{ij}(t)=0$ means that there is no connection from ToR i to ToR j. Note that $S_{ii}(t)=0$ for all t and $i \in \{1,2,\ldots,N\}$. We also assume at any t each ToR can only transmit to at most one destination, and can only receive from at most one source, *i.e.*, $\sum_i S_{ij}(t) \le 1, \sum_j S_{ij}(t) \le 1$. Furthermore, $\mathbf{S}(t)$ should be such that for any pair of $S_{ij}(t), S_{i'j'}(t) > 0$, there exists non-blocking circuits available across the data center to transmit packets simultaneously from ToR switch i' to j'. Let $\mathfrak{S}$ be the set of all feasible schedules (respecting the bi-section bandwidth and parallel scheduling requirements).

Let $A_{ij}(t)$ and $D_{ij}(t)$ be the number of packets arrived at and departed from queue Q_{ij} at time t, respectively. Let $L_{ij}(t)$ be the number of packets in the edge queue Q_{ij} at the beginning of the time slot t. For ease of notation, we set $A_{ii}(t) = D_{ii}(t) = L_{ii}(t) = 0$ for all t and write $\mathbf{A}(t) = [A_{ij}(t)], \mathbf{D}(t) = [D_{ij}(t)], \mathbf{L}(t) = [L_{ij}(t)]$, where $\mathbf{A}(t), \mathbf{D}(t), \mathbf{L}(t) \in \mathbb{N}_+^{N\times N}$. It is clear that in order to achieve small delay the central controller's objective must be to ensure a small $\mathbf{L}(t)$ at all time. On the other hand, in practice the arrival processes $A_{ij}(t)$ is random and not fully predictable. In other words, central controller must design the departure process $\mathbf{D}(t)$ such that it tracks the arrival process $\mathbf{A}(t)$ as closely as possible. On the other hand, so long as Q_{ij} holds sufficiently large number of packets, $D_{ij}(t) \propto S_{ij}(t)$. This means that in order to ensure small delay, the central controller designs the schedule carefully to respond to the network traffic condition and the backlog state $\mathbf{L}(\cdot)$.

1) Stability and Capacity Region: An edge queue Q_{ij} is strongly stable if its queue length $L_{ij}(t)$ satisfies:

$$\limsup_{t\to\infty} \frac{1}{t}\sum_{\tau=1}^{t} \mathbb{E}\{\mathbf{L}_{ij}(\tau)\} < \infty$$

and we say the system of queues is stable if Q_{ij} is strongly stable for all $i,j \in \{1,2,\ldots,N\}, i \neq j$. A scheduling policy is said to stabilize the system if the system is stable under that scheduling policy. With this notion of stability, we define the capacity region $\mathcal{C}$ of the network as the set of all traffic rate matrix such that there exists a scheduling policy which stabilizes the system.

The capacity region is given by the interior of the convex hull of the feasible schedules $\mathcal{F}$ [3], that is

$$\mathcal{C} = \left\{ \sum_{\mathbf{S}\in\mathcal{F}} \alpha_{\mathbf{S}}\mathbf{S} : \sum_{\mathbf{S}\in\mathcal{F}} \alpha_{\mathbf{S}} < 1,\ \alpha_{\mathbf{S}} \ge 0,\ \forall \mathbf{S} \in \mathcal{F} \right\}$$

We call a random ergodic arrival process $\{\mathbf{A}(t)\}$ with long-term traffic rate matrix $\boldsymbol{\lambda} = [\lambda_{ij}] \in \mathbb{R}^{N\times N}$, where $\lambda_{ij} = \mathbb{E}\{\frac{1}{T}\sum_{t=1}^{T} A_{ij}(t)\}$, admissible if $\boldsymbol{\lambda} \in \mathcal{C}$. Furthermore, we define the traffic load as $\rho(\boldsymbol{\lambda}) = \max\{r : \boldsymbol{\lambda} \in r\bar{\mathcal{C}},\ 0 < r < 1\}$, where $\bar{\mathcal{C}}$ is the closure of $\mathcal{C}$.

B. Temporal Structure

One of the main attributes of our work is to identify three distinct time sequences associated with monitoring, computation, and schedule reconfiguration.

Definition 1. Let $\{t_k^M\}_{k=1}^{\infty}$ denote the time instances that the state of edge queues are uploaded to the centralized scheduler. Specifically, the information available at the sheduler is a subset of the edge queue lengths $\{\mathbf{L}(t_k^M)\}_{k=1}^{\infty}$.

Note that with a careful design of monitoring/control plane, as we will see in the next section, we can ensure high reliability in the estimated values and low noise across any network, so long as we allow Δ_m to be sufficiently large.

Definition 2. Let $\{t_k^C\}_{k=1}^{\infty}$ denote the time instances when a set of new schedules are computed. A scheduler could generate

one schedule or multiple schedules, which depends on the scheduling policy used.

Definition 3. Let $\{t_k^S\}_{k=1}^{\infty}$ denote the time instances when the schedule is reconfigured. The schedule between two schedule reconfiguration time instances remains the same, *i.e.*

$$\mathbf{S}(\tau) = \mathbf{S}(t_k^S), \quad \forall \tau \in [t_k^S, t_{k+1}^S - 1]$$

Each of the three processes is associated with a corresponding delay as described below.

Definition 4. Let Δ_m be the delay of the monitoring process. This means that the edge queue lengths at time t_k^M, $\mathbf{L}(t_k^M)$, is available at the scheduler after time $t_k^M + \Delta_m$. Therefore, at any time instance t, the edge queue lengths information available at the scheduler is the set $\{\mathbf{L}(t_k^M)\}_{k=1}^{n}$, where $n = \max\{k : t_k^M + \Delta_m < t\}$.

Definition 5. Let Δ_c be the delay of the computation process of the scheduler generating a set of new schedules. This means that the schedules computed at time t_k^C are available (could be used) after time $t_k^C + \Delta_c$.

Definition 6. Let Δ_r be the reconfiguration delay associated with establishing a new schedule across the network. During the period of schedule reconfiguration, no packet transmission could occur in the network. This means that $\forall i, j \in \{1, 2, \dots N\}$, $\forall k \in \mathbb{N}_+$, and $0 \leq \tau \leq \Delta_r$, we have $D_{ij}(t_k^S + \tau) = 0$.

The above abstraction allows us to map the practical challenges in form of a delay $\Delta_m + \Delta_c + \Delta_r$; for example, the beginning of a schedule at time t, $S(t)$, is actually being reconfigured at time $t - \Delta_r$. The computation of this schedule began at time $t - \Delta_r - \Delta_c$, and the computation is based on information of edge queues at time $t - \Delta_r - \Delta_c - \Delta_m$. More precisely, $\mathbf{S}(t)$ is only selected based on $\mathbf{L}(t - \Delta_m - \Delta_c)$ and will take effect at time $t + \Delta_r$. Fig. 2 illustrates this abstraction. In other words, the above timing parameters restrict the scheduling algorithms from using spontaneous edge queue information.Note that in this work, we are primarily interested in the case of $\Delta_c \approx 0$, while $\Delta_m, \Delta_r > 0$, since we imagine the computation power in a data center to be fairly ubiquitous.

The problem of low complexity dynamic circuit switching with non-negligible monitoring and reconfiguration delays, $\Delta_m, \Delta_r > 0$, constitutes a topic of extensive research, building on prior work on cross-bar switch scheduling [4]. Our proposed solution relies on the temporal structure discussed above and hence on considering the timing process associated with monitoring/control and scheduling individually. The benefit of this approach is that by considering the monitoring and scheduling processes separately, we can rely on dynamic monitoring even in face of limitations in rate of scheduling. Dynamic monitoring allows for on-line and adaptive selection of scheduling times (and hence scheduling rate). This deviates from the existing approaches as we will discuss next.

In [5], a closed loop scheduling policy based on the Birkhoff and von Neumann (BvN) theorem [6] is proposed. The BvN theorem states that any admissible doubly stochastic matrix can be decomposed as a convex combination of permutation matrices. The BvN scheduling policy [7] assumes the knowledge of the arrival statistics and relies on a BvN

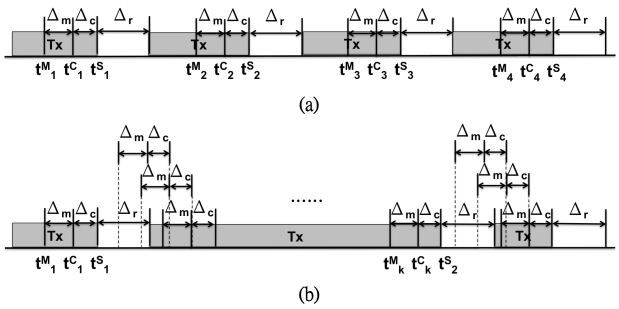

Fig. 3. Timing diagrams for: (a) Periodic scheduling: While each schedule is dependent on the most up-to-date edge queue information, the schedule update is done periodically. (b) Adaptive scheduling: The scheduler frequently monitors the queue information and computes schedule weights. The schedule reconfiguration time instances become aperiodic. Each schedule is computed based on the most up-to-date edge queue lengths.

decomposition of arrival rate matrix into a set of schedules. The preposed traffic matrix scheduling (TMS) policy [5] falls in the class of fixed batch scheduling policies proposed in [8] in the context of switching with non-negligible reconfiguration delay, $\Delta_r > 0$. The scheduling policies in [5] and [8] both involve "quasi-static monitoring": selecting series of schedules based on a single schedule computation process. When monitoring and computation times are identically coupled, generated schedules may depend on very out-dated information, as shown in Figure 2 (a). In [9], it is shown that it is always beneficial to employ "active monitoring": schedules must be selected based on frequent and up-to-date queue information, as shown in Figure 2 (b). In other words, decoupling the monitoring and scheduling process in order to allow for active monitoring results in significant improvements.

To account for the non-negligible reconfiguration delay, hence the loss in the duty cycle, much of the prior work either rely on the explicit traffic statistics or a conservative upper bound to restrict the rate of schedule reconfigurations. In other words, most prior work consider a periodic choice of scheduling times, *i.e.*$t_k^S = kT$, $k \in \mathbb{N}_+$, where T would be the mean schedule duration and is selected appropriately. In contrast, under the Adaptive MaxWeight (AMW) proposed in [9], the times to perform schedule reconfiguration are not limited to the multiples of T, which had to be chosen with some knowledge of Δ_r and traffic load. Instead the schedule reconfigurations can occur sooner or later depending on the current state of the system, which in turn is a result of prior effective switching rate. Figure 3 illustrates this two classes of the central control operation.

III. Technological Enablers and Promising First Steps

A. SDN and OpenFlow Protocol

The software defined networking (SDN) concept aims to allow network operators to manage network elements through abstraction of low-level functionalities such as switching and routing. This is now usually done by the separation of the control and data plane in the network: The physical switches (data plane) in the network perform forwarding functionalities according to their own flow tables; whereas the flow tables are manageable by a (remote) software-based controller (control plane) through a secure channel, as shown in Fig. 4. OpenFlow [10] is a standardized protocol that implements the SDN concept. Through OpenFlow protocol and its circuit switch

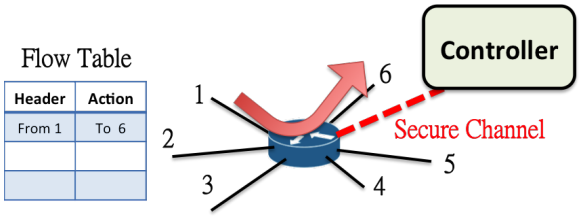

Fig. 4. Components of an OpenFlow-enabled switch

addendum [11], the switching functionality of a packet switch or either a circuit switch can be treated as flows managed by flow tables, as suggested in [12].

The control over the flows through OpenFlow protocol is a perfect technology enabler for the proposed architecture, since the centralized controller could then configure the schedule of the network by directly manipulating the flow tables of the network switches.

We envision the proposed architecture to operate three main functionalities:

(1) Collect edge queue information from ToR switches

(2) Compute the schedule based on collected edge queue information

(3) Configure circuit schedule by modifying flow tables of network switches

With the controller manipulating the network switches through OpenFlow protocol, the functionality (3) could be realized. In the following subsections, we further discuss on suitable candidates for the functionalities (1) and (2).

B. Adaptive MaxWeight Scheduling Algorithm

The scheduling algorithm is the key to the efficient use of the data bandwidth provided by the underlying network. The temporal structure of our proposed architecture allows for our scheduling instances to be adapted to reconfiguration delay Δ_r and traffic dynamics.

Considering the reconfiguration delay Δ_r (which can account for the circuit setup time required through OpenFlow protocol and/or physical restriction on the swtiticing rate of optical devices) gives rise to a new class of scheduling algorithm. Fo instance, we expect the Adaptive MaxWeight (AMW) [9] to be a candidate scheduling algorithm.

Fig. 5 shows the delay performance of the AMW compared with a periodic scheduling policy Periodic MaxWeight (PMW) [9] under zero monitoring delay. It is clear that the AMW achieves lower delay at any traffic load even if the period of the PMW is tuned to the optimum. The performance gain comes from the fact that the AMW continuously assess the effectiveness of the current schedule and reconfigure the schedule only if it is worth to do so. The result demonstrates the benefit of extensive monitoring and computing. These results, however, only hold under the assumption that monitoring delay Δ_m is negligible. In fact, Fig. 6 shows that the performance sees significant degradation as the monitoring delay increases above 50 μsec.

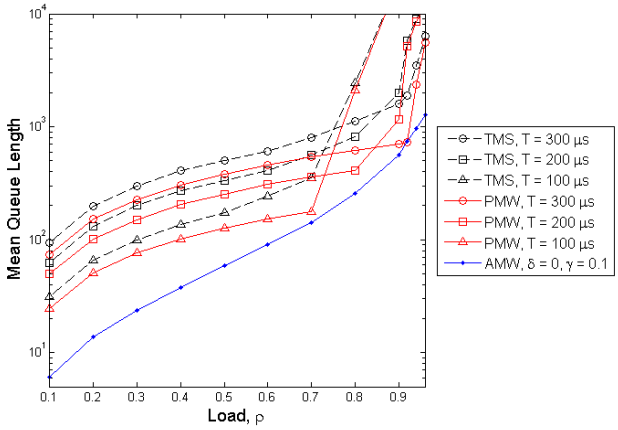

Fig. 5. Mean queue length versus traffic load ρ under a nonuniform traffic. The TMS policy reconfigures the schedule $q = 10$ times within qT time duration.

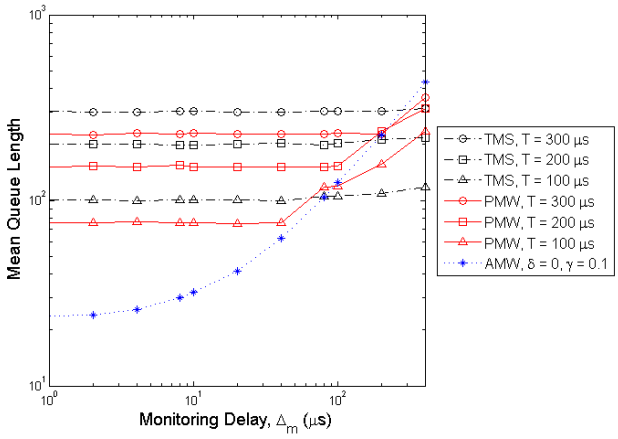

Fig. 6. Mean queue length versus monitoring delay Δ_m under nonuniform traffic. The traffic load is fixed as $\rho = 0.3$. The edge queue state is monitored/updated every microsecond, $t^M_{k+1} - t^M_k = 1\mu$s for all k.

C. Wireless Monitoring/Control Plane

The low-latency monitoring of the states of the edge queues is the crucial part of the control operations carried out by the centralized scheduler. Furthermore, it is important to complete this monitoring operation with very low delay in order to compute efficient schedules. In a companion paper [14], we have demonstrated that the monitoring/control plane can be realized by a simple single-hop wireless network based on mmWave multiple-input multiple-output (MIMO) radio access technology. As an initial design for the monitoring plane, in a companion paper [14], we consider some well-known enabling technologies. To start with, Orthogonal Frequency Division Multiple Access (OFDMA) [15] is investigated as the digital modulation scheme. OFDMA is a particularly efficient modulation scheme when the delay spread of the wireless channel is relatively small. Moreover, due to the orthogonal transmissions of the ToR switch signals, the receiver structure is fairly simple to implement for OFDMA. Finally, we can also make use of the static nature of the channels from the ToR switch to the centralized scheduler. Once the frequency

responses of these channels are accurately estimated, we can allocate frequency resources to ToR switches by using a spatially adaptive scheme that only depends on the locations of the ToR switches and scatterers in the environment. In addition to OFDMA, we propose the utilization of the MIMO and beamforming at the transmitting and receiving units in order to improve reliability of communication both by increasing the received power via pencil-beams and also by decreasing the intersymbol interference (ISI) thanks to the reduced number of multipaths.

IV. Next Steps and Future Research

With the technology enabler discussed above, we expect an fully integrated zero in-network queueing network architecture could be developed. The process would start from a proof-of-concept prototype and follow by the deployment on existing networks. Mininet [16] is a container-based emulation testbed that is suitable for experimenting with OpenFlow protocol. We expect to implement the controller described above and test the feasibility of the proposed architecture as the first proof-of-concept prototype. The full compatibility of Mininet and OpenFlow protocol allows smooth migration from emulation testbed to real deployment on OpenFlow-enabled switches. Therefore, along with the development of the wireless monitoring plane over ToR switches, the fully integrated network architecture could be further expected to be realized on existing OpenFlow-enabled networks.

Beyond the implementation of the proposed network architecture, there is an abundance of future research directions and challenges. In most data center applications, the flow completion time is usually a more desirable measure of performance over the per-packet delay. While, the per-packet delay performance is an important performance measure, in a data center, it is far more critical to further consider scheduling algorithms that aim to optimize the flow completion time. In particular, we expect to combine salient design feature of Adaptive MaxWeight algorithm with those of Shortest Remaining Processing Time (SPRT) to minimize flow completion time via prioritization of flows based on flow size and consider this as a future research direction.

Another area of future work aims to lower the complexity of Adaptive MaxWeight algorithm mentioned above. We believe that the concept of the adaptive policies is general enough to transform the plethora of low complexity scheduling algorithms in the literature (e.g. [17], [18]) into ones that address the reconfiguration delay. To be specific, the core idea of the adaptive policies is to obtain a measure of the effectiveness of the current schedule, and by continuously monitoring the edge queue information, the scheduler is able to determine the "right" time to reconfigure the circuits to the "right" schedule. Low complexity algorithms related to MaxWeight policies such as the randomized scheduling algorithm proposed by Tassiulas [18] could inherently take the schedule weight as the measure as in the Adaptive MaxWeight algorithm. The tradeoff between the complexity and the delay performance is another important question of future research interest.

As given in our companion paper [14], the monitoring delay achieved by the OFDMA mmWave MIMO radio access technology increases as the size of the data center becomes very large. This trend is summarized in Fig. 7. In particular, as the number of ToRs grow beyond 550 or so, our simple proposed OFDMA mmWave MIMO radio access solution fails to achieve lower latency. As a result, designing wireless

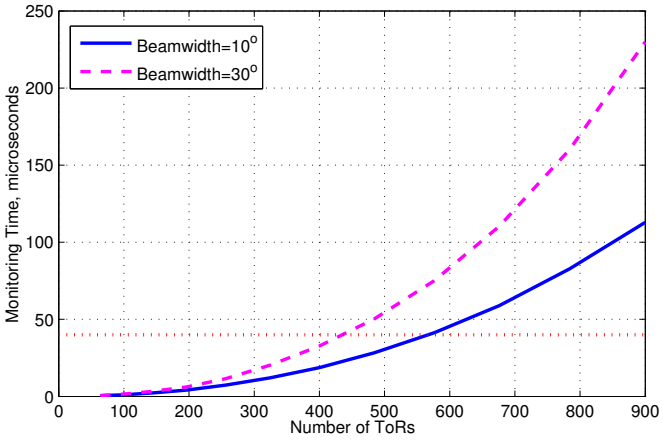

Fig. 7. Monitoring delay with respect to the number of ToRs

solutions with lower delay is an important area of ongoing research. For example, to address the inefficiency caused by the preallocation of the system resources and orthogonalization of access (subcarrier allocation in OFDMA) , Asynchronous Code Division Multiple Access (A-CDMA) modulation is likely to provide significant performance improvements. Another direction of future research involves developing communication strategies that are efficient in the finite block length regime.

Acknowledgment

This work has been partially supported by L-3 Communications and NSF Center for Integrated Access Networks (Grant EEC-0812072).

References

[1] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker, "pfabric: Minimal near-optimal datacenter transport," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, (New York, NY, USA), pp. 435–446, ACM, 2013.

[2] J. Perry, A. Ousterhout, H. Balakrishnan, D. Shah, and H. Fugal, "Fastpass: A centralized "zero-queue" datacenter network," in *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, (New York, NY, USA), pp. 307–318, ACM, 2014.

[3] L. Tassiulas and A. Ephremides, "Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks," *Automatic Control, IEEE Transactions on*, vol. 37, pp. 1936–1948, Dec 1992.

[4] N. McKeown, V. Anantharam, and J. Walrand, "Achieving 100input-queued switch," in *INFOCOM '96. Fifteenth Annual Joint Conference of the IEEE Computer Societies. Networking the Next Generation. Proceedings IEEE*, vol. 1, pp. 296–302 vol.1, Mar 1996.

[5] G. Porter, R. Strong, N. Farrington, A. Forencich, P. Chen-Sun, T. Rosing, Y. Fainman, G. Papen, and A. Vahdat, "Integrating microsecond circuit switching into the data center," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, (New York, NY, USA), pp. 447–458, ACM, 2013.

[6] G. Birkhoff, "Tres observaciones sobre el algebra lineal," *Univ. Nac. Tucumán Rev. Ser. A5*, no. 147-150, 1946.

[7] C.-S. Chang, W.-J. Chen, and H.-Y. Huang, "Birkhoff-von neumann input buffered crossbar switches," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 3, pp. 1614–1623 vol.3, Mar 2000.

[8] K. Ross and N. Bambos, "Adaptive batch scheduling for packet switching with delays," in *High-performance Packet Switching Architectures* (I. Elhanany and M. Hamdi, eds.), pp. 65–79, Springer London, 2007.

[9] C.-H. Wang, T. Javidi, and G. Porter, "End-to-end scheduling for all-optical data centers," in *INFOCOM, 2015 Proceedings IEEE*, April 2015.

[10] Open Network Foundation, *OpenFlow Switch Specification Version 1.4.0*, October 2013.

[11] S. Das, *Extensions to the OpenFlow Protocol in support of Circuit Switching*, June 2010.

[12] S. Das, G. Parulkar, and N. McKeown, "Unifying packet and circuit switched networks," in *GLOBECOM Workshops, 2009 IEEE*, pp. 1–6, Nov 2009.

[13] FCC Report, "Amendment of parts 2, 15 and 97 of the commissions rules to permit use of radio frequencies above 40 ghz for new radio applications," tech. rep., December 1995.

[14] T. Aktas, C.-H. Wang, and T. Javidi, "Wicod: Wireless control plane serving an all-optical data center," in *WiOpt, 2015 Proceedings IEEE*, May 2015.

[15] C. Y. Wong, R. Cheng, K. Lataief, and R. Murch, "Multiuser ofdm with adaptive subcarrier, bit, and power allocation," *Selected Areas in Communications, IEEE Journal on*, vol. 17, pp. 1747–1758, Oct 1999.

[16] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, Hotnets-IX, (New York, NY, USA), pp. 19:1–19:6, ACM, 2010.

[17] P. Giaccone, B. Prabhakar, and D. Shah, "Randomized scheduling algorithms for high-aggregate bandwidth switches," *Selected Areas in Communications, IEEE Journal on*, vol. 21, pp. 546–559, May 2003.

[18] L. Tassiulas, "Linear complexity algorithms for maximum throughput in radio networks and input queued switches," in *INFOCOM '98. Seventeenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, vol. 2, pp. 533–539 vol.2, Mar 1998.