Coflow Scheduling in Input-Queued Switches: Optimal Delay Scaling and Algorithms

Qingkai Liang and Eytan Modiano
Laboratory for Information and Decision Systems
Massachusetts Institute of Technology, Cambridge, MA

***Abstract*—A coflow is a collection of parallel flows belonging to the same job. It has the all-or-nothing property: a coflow is not complete until the completion of all its constituent flows. In this paper, we focus on optimizing *coflow-level delay*, i.e., the time to complete all the flows in a coflow, in the context of an $N \times N$ input-queued switch. In particular, we develop a throughput-optimal scheduling policy that achieves the best scaling of coflow-level delay as $N \to \infty$. We first derive lower bounds on the coflow-level delay that can be achieved by any scheduling policy. It is observed that these lower bounds critically depend on the variability of flow sizes. Then we analyze the coflow-level performance of some existing coflow-agnostic scheduling policies and show that none of them achieves provably optimal performance with respect to coflow-level delay. Finally, we propose the Coflow-Aware Batching (CAB) policy which achieves the optimal scaling of coflow-level delay under some mild assumptions.**

I. Introduction

Modern cluster computing frameworks, such as MapReduce [1] and Spark [2], have been widely used in large-scale data processing and analytics. Despite the differences among these frameworks, they share a common feature: the computation is divided into multiple stages and a collection of parallel data flows need to be transferred between groups of machines in successive computation stages. Often the next computation stage cannot start until the completion of all of these flow transfers. For example, during the shuffle phase in MapReduce, any reducer node cannot start the next reduce phase until it receives intermediate results from all of the mapper nodes. As a result, the response time of the entire computing job critically depends on the completion time of these intermediate flows. In some applications, these intermediate flow transfers can account for more than 50% of job completion time [3].

The recently proposed *coflow* abstraction [4] represents such a collection of parallel data flows between two successive computation stages of a job, which exposes application-level requirements to the network. It builds upon the *all-or-nothing* property observed in many applications [6]: a coflow is not complete until the completion of all its constituent flows. As a result, one of the most important metrics in this context is *coflow-level delay* (also referred to as *coflow completion time* in some literature [5]–[7]), i.e., the time to complete all of the flows in a coflow. To improve the overall response time of a job, it is crucial to schedule flow transfers in a way that the coflow-level delay can be reduced. Unfortunately, researchers have largely overlooked such application-level requirements and there has been little work on coflow-level delay optimization.

In this paper, we study coflow-level delay in the context of an $N \times N$ input-queued switch with stochastic coflow arrivals. In each slot, a random number of coflows, each of them consisting of multiple parallel flows, arrive to the input-queued switch where each input/output port can process at most one packet per slot. Such an input-queued switch model is a simple yet practical abstraction for data centers with full bisection bandwidth, where N represents the number of servers. Due to the large scale of modern data centers, we are motivated to study the scaling of coflow-level delay as $N \to \infty$. In particular, we are interested in the optimal scaling of coflow-level delay, i.e., the scaling under an "optimal" scheduling policy[1]. As far as we know, this is the first paper to present coflow-level delay analysis in a large-scale stochastic system.

The contributions of this paper are summarized as follows.

- We derive lower bounds on the expected coflow-level delay that can be achieved by any scheduling policy in an $N \times N$ input-queued switch. These lower bounds critically depend on the variability of flow sizes. In particular, it is shown that if flow sizes are light-tailed, no scheduling policy can achieve an average coflow-level delay better than $O(\log N)$.
- We analyze the coflow-level performance of several coflow-agnostic scheduling policies, where coflow-level information is not leveraged. It is shown that none of these scheduling policies achieves a *provably optimal* scaling of coflow-level delay. For example, the expected coflow-level delay achieved by randomized scheduling is $O(N \log N)$ if coflow sizes are light-tailed, far above the $O(\log N)$ lower bound.
- We show that $O(\log N)$ is the optimal scaling of average coflow-level delay when flow sizes are light-tailed and coflow arrivals are Poisson. This optimal scaling is achievable with our Coflow-Aware Batching (CAB) policy.

The organization of this paper is as follows. We first review related work in Section II. The system model is introduced in

This work was supported by NSF Grants CNS-1116209, CNS-1617091, and by DARPA I2O and Raytheon BBN Technologies under Contract No. HROO 1 l-l 5-C-0097.

[1]An "optimal" policy should be throughput-optimal, i.e., stabilize the system whenever the load $\rho < 1$, and should achieve the minimum coflow-level delay among all throughput-optimal policies.

978-1-5090-5336-0/17/$31.00 ©2017 IEEE

Section III. In Section IV, we demonstrate fundamental lower bounds on the expected coflow-level delay that can be achieved by any scheduling policy. In Section V, we analyze the coflow-level performance of some coflow-agnostic scheduling policies. In Section VI, we propose the Coflow-Aware Batching (CAB) policy and show that it achieves the optimal coflow-level delay scaling under some conditions. Finally, simulation results and conclusions are given in Sections VII and VIII, respectively.

II. Related Work

We start with a brief literature review on coflow-level optimization and delay scaling in input-queued switches.

Coflow-level Optimization. The notion of coflows was first proposed by Chowdhy and Stoica [4] to convey job-specific requirements such as minimizing coflow-level delay or meeting some job completion deadline. Unfortunately, coflow-level optimization is often computationally intractable. For example, it was shown in [5] that minimizing the average coflow-level delay is NP-hard. As a result, many heuristic scheduling principles were developed to improve coflow-level delay. In [8], a decentralized coflow scheduling framework was proposed to give priority to coflows according to a variation of the FIFO principle, which performs well for light-tailed flow sizes. In [5], the Varys scheme improves the performance of [8] by leveraging more sophisticated heuristics such as "smallest-bottleneck-first" and "smallest-total-size-first", where global information about coflows is required. The D-CAS scheme in [10] exploits a similar "shortest-remaining-time-first" principle for coflow scheduling. The Aalo framework [6] generalizes the classic least-attained service (LAS) discipline [9] to coflow scheduling; such a scheme does not require prior knowledge about coflows. Zhong *et al.* [11] develop an approximation algorithm to minimize the average coflow-level delay in data centers. Additionally, Chen *et al.* [7] jointly consider coflow routing and scheduling in data centers. Despite these efforts towards coflow-level optimization, most prior works do not provide any analytical performance guarantee, and there is a lack of fundamental understanding of coflow-level scheduling, especially in the context of large-scale stochastic systems.

Optimal Delay Scaling in Input-Queued Switches. The optimal (*packet-level*) delay scaling in input-queued switches (i.e., the delay scaling under an optimal scheduling policy) has been an important area of research for more than a decade. The randomized scheduling policy [19] (based on Birkhoff-Von Neumann decomposition) achieves an average packet delay of $O(N)$. The well-known Max-Weight Matching (MWM) [20] policy and various approximate MWM algorithms [13], [21] are shown to have an average packet delay no greater than $O(N)$, although it is conjectured that this bound is not tight for a wide range of traffic patterns [21]. Recently, Maguluri *et al.* [14], [15] show that MWM can achieve the optimal $O(1)$ packet-level delay in the *heavy-traffic regime*. Neely *et al.* [18] propose a batching scheme that achieves an average packet delay of $O(\log N)$; this is the best known result for packet-level delay scaling as $N \to \infty$ *under general traffic conditions*.

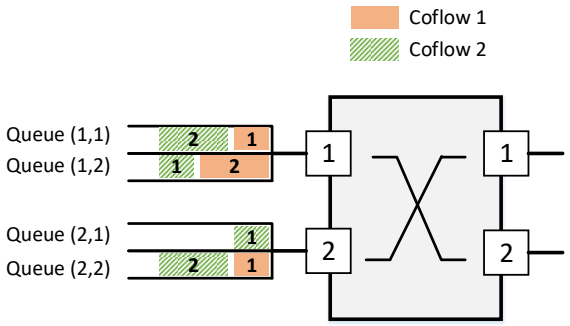

Fig. 1. A 2×2 input-queued switch with two coflow arrivals.

Zhong *et al.* [12] consider the joint scaling of queue length as $N \to \infty$ and $\rho \to 1$. They propose a policy that gives an upper bound of $O(1/(1-\rho)+N^2)$; this joint scaling is shown to be "optimal" in the *heavy-traffic regime* where $\rho = 1 - O(\frac{1}{N^2})$. However, to the best of our knowledge, the optimal scaling of packet-level delay under a general traffic condition is still an open problem in input-queued switches. By comparison, the optimal scaling of *coflow-level delay*, which is an upper bound for packet-level delay, has not been studied before. In this paper, we make the first attempt in deriving the optimal coflow-level delay scaling as $N \to \infty$.

III. System Model

A. Network Model

We consider an $N \times N$ input-queued switch with N input ports and N output ports. The system operates in slotted time, and the slot length is normalized to one unit of time. In each slot, each input can transfer at most one packet and each output can receive at most one packet (this is referred to as "crossbar constraints"). Such an input-queued switch model is simple yet very useful in modeling many practical networked systems. For example, data centers with full bisection bandwidth can be abstracted out as a giant input-queued switch interconnecting different machines. Note that each input port may have packets destined for different output ports, which can be represented as *Virtual Output Queues* (VOQ). There are a total of N^2 virtual output queues, indexed by (i,j) for $i,j \in [N]$, where $[N] \triangleq \{1, \cdots, N\}$ and queue (i,j) holds packets from input i to output j. Figure 1 shows a 2×2 input-queued switch with four virtual output queues.

The schedule of packet transmissions in slot t can be represented by an $N \times N$ matrix $\mathbf{S}(t) = (S_{ij}(t))$ where $S_{ij}(t) = 1$ if the connection between input i and output j is activated. A feasible schedule $\mathbf{S}(t)$ is one that satisfies the crossbar constraints, i.e., $\mathbf{S}(t)$ must be a binary matrix where there is at most one "1" in each row and each column.

B. Coflow Abstraction

A coflow is a collection of parallel data flows belonging to the same job. It has the all-or-nothing property: a coflow is not complete until the completion of all its constituent flows. Coflows are a useful abstraction for many communication patterns in data-intensive computing applications such

as MapReduce (see [4] for applications of coflows). Note that the traditional point-to-point communication is a special case of coflows (i.e., a coflow with a single flow).

Formally, we represent each coflow by a random traffic matrix $\mathbf{X} = (X_{ij})$ where X_{ij} is the number of packets in this coflow that need to be transmitted from input i to output j. Note that each coflow may contain many small flows from input i to output j, that are aggregated into a single batch X_{ij} for ease of exposition. In the following, X_{ij} will be referred to as the "**batch size**" or "**flow size**" from input i to output j. We assume that all the packets in a coflow are released simultaneously upon the arrival of this coflow. Figure 1 illustrates two coflow arrivals whose traffic matrices are $\mathbf{X}_1 = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}$ and $\mathbf{X}_2 = \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix}$. Let $\mathbb{E}[X_{ij}] = \beta_{ij}$ and assume that coflows arrive to the system with rate λ. Then the arrivals of packets to queue (i, j) is a *batch arrival process* with rate λ and mean batch size β_{ij}. Also define $\beta \triangleq \max\Big(\max_i \sum_j \beta_{ij}, \max_j \sum_i \beta_{ij}\Big)$. In our analysis, we also make the following assumptions:

(1) The arrival times and the batch sizes of different coflows are independent.
(2) X_{ij}'s are independent random variables.
(3) If the n-th moment of X_{ij} is finite, we assume $\mathbb{E}[(\sum_j X_{ij})^n] = O(1)$ for all $i \in [N]$ and $\mathbb{E}[(\sum_i X_{ij})^n] = O(1)$ for all $j \in [N]$ as $N \to \infty$.
(4) The sub-critical condition holds: $\rho = \lambda\beta < 1$.

In this paper, we focus on optimizing *coflow-level delay*, i.e., *the time between the arrival of a coflow until all the packets associated with this coflow are transmitted*. In particular, we are interested in the scaling of coflow-level delay in a large-scale system, as $N \to \infty$. Our objective is to find a scheduling policy that achieves the best dependence of coflow-level delay on N while stabilizing the system whenever $\rho < 1$.

IV. Lower Bounds on Coflow-level Delay

Before we investigate any specific scheduling policy, it is useful to study the fundamental scaling properties of coflow-level delay as $N \to \infty$. In this section, we develop lower bounds on coflow-level delay in input-queued switches. These lower bounds serve as the baselines when we evaluate the coflow-level performance of a scheduling policy. We first introduce the notion of *clearance time*.

Definition 1 (Clearance Time)**.** *The clearance time of a coflow* $\mathbf{X} = (X_{ij})$ *is*

$$\tau(\mathbf{X}) = \max\Big(\max_i \sum_j X_{ij}, \max_j \sum_i X_{ij}\Big). \tag{1}$$

Clearly, $\tau(\mathbf{X})$ is the maximum number of packets in a row or a column of $\mathbf{X}$. Since each input/output port can process at most one packet per slot, the minimum time to clear all the packets in $\mathbf{X}$ must be no smaller than $\tau(\mathbf{X})$. In fact, $\mathbf{X}$ can be cleared in exactly $\tau(\mathbf{X})$ slots by using the *optimal clearance algorithm* described in [16]. As a result, $\tau(\mathbf{X})$ is the minimum time needed to transmit all the packets in a coflow $\mathbf{X}$. In the rest of this section, we investigate the scaling of clearance time as $N \to \infty$ and its relationship to coflow-level delay.

Depending on the distributions of X_{ij}'s (the flow sizes), the scaling of clearance time exhibits different behaviors. First, we consider the general case where the flow size distribution is arbitrary (as long as $\mathbb{E}[X_{ij}] < \infty$ for all $i, j \in [N]$).

Lemma 1. *For a coflow* $\mathbf{X} = (X_{ij})$ *with* $\mathbb{E}[X_{ij}] < \infty$*, the expected clearance time is* $\mathbb{E}[\tau(\mathbf{X})] = O(N)$*. Moreover, there exist distributions of* X_{ij}*'s such that* $\mathbb{E}[\tau(\mathbf{X})] = \Omega(N^{\frac{1}{1+\epsilon}})$ *for any* $\epsilon > 0$*.*

Proof. The $O(N)$ upper bound is nearly trivial. It is clear that $\mathbb{E}[\tau(\mathbf{X})] \le \mathbb{E}[\sum_{i,j} X_{ij}] \le N\beta = O(N)$. To prove the lower bound, we find some distributions of X_{ij}'s such that $\mathbb{E}[\tau(\mathbf{X})] = \Omega(N^{\frac{1}{1+\epsilon}})$ for any $\epsilon > 0$. Consider the scenario where $\mathbf{X}$ is a diagonal matrix: $X_{ij} = 0$ with probability 1 for $i \neq j$ and X_{ii} has the power law for all $i \in [N]$, i.e., $\mathbb{P}[X_{ii} \ge k] = 1/k^{1+\epsilon},\ k = 1, 2, \cdots$, where $\epsilon > 0$. Note that $\mathbb{E}[X_{ii}] = \frac{1+\epsilon}{\epsilon}$ but it can be easily scaled or shifted to have an arbitrary expectation. Under such flow size distributions, it can be proved that $\mathbb{E}[\tau(\mathbf{X})] = \Omega(N^{\frac{1}{1+\epsilon}})$ as $N \to \infty$. The detailed proof can be found in the technical report [27]. □

If X_{ij} also has a finite variance, we can obtain a better scaling behavior of clearance time as $N \to \infty$. In this case, we assume $\mathrm{Var}(\sum_j X_{ij}) \le \sigma^2$ for all $i \in [N]$ and $\mathrm{Var}(\sum_i X_{ij}) \le \sigma^2$ for all $j \in [N]$, where σ^2 is a constant independent of N.

Lemma 2. *For a coflow* $\mathbf{X} = (X_{ij})$ *with* $Var(X_{ij}) < \infty$ *for all* $i, j \in [N]$*, the expected clearance time is* $\mathbb{E}[\tau(\mathbf{X})] = O(\sqrt{N})$*. Moreover, there exist distributions of* X_{ij}*'s such that* $\mathbb{E}[\tau(\mathbf{X})] = \Omega(N^{\frac{1}{2+\epsilon}})$ *for any* $\epsilon > 0$*.*

Proof. Devroye [22] shows that if $Y_1, \cdots, Y_n$ are (possibly dependent) random variables with finite means and finite variances, then $\mathbb{E}[\max_i Y_i] \le \max_i \mathbb{E}[Y_i] + \sqrt{n}\max_i \sqrt{\mathrm{Var}(Y_i)}$. If $\mathrm{Var}(X_{ij}) < \infty$ for all $i, j \in [N]$, it follows that

$$\begin{aligned}\mathbb{E}[\max_i \sum_j X_{ij}] &\le \max_i \mathbb{E}[\sum_j X_{ij}] + \sqrt{N}\max_i \sqrt{\mathrm{Var}(\sum_j X_{ij})} \\ &\le \beta + \sqrt{N}\sigma.\end{aligned}$$

Similarly, it can be shown that $\mathbb{E}[\max_j \sum_i X_{ij}] \le \beta + \sqrt{N}\sigma$. As a result, we have $\mathbb{E}[\tau(\mathbf{X})] \le \mathbb{E}[\max_i \sum_j X_{ij}] + \mathbb{E}[\max_j \sum_i X_{ij}] \le 2\beta + 2\sqrt{N}\sigma$. Consequently, $\mathbb{E}[\tau(\mathbf{X})] = O(\sqrt{N})$. The $\Omega(N^{\frac{1}{2+\epsilon}})$ lower bound can be proved in a similar way to Lemma 1 with the power being $2+\epsilon$ instead of $1+\epsilon$ such that the variance is finite. □

Furthermore, if X_{ij}'s have light-tailed[2] distributions, the scaling of clearance time is logarithmic.

Lemma 3. *For a coflow* $\mathbf{X} = (X_{ij})$ *with light-tailed* X_{ij}*'s, the expected clearance time is* $\mathbb{E}[\tau(\mathbf{X})] = O(\log N)$*. Moreover, there exist distributions of* X_{ij}*'s such that this bound is tight.*

[2]In this paper, a light-tailed distribution is the one with a finite Moment Generating Function in the neighborhood of 0. In other words, it has an exponentially decreasing tail.

TABLE I
LOWER BOUNDS ON COFLOW-LEVEL DELAY

Condition	Coflow-level Delay
$\mathbb{E}[X_{ij}] < \infty$	$\Omega(N^{\frac{1}{1+\epsilon}})$ for any $\epsilon > 0$
$\text{Var}(X_{ij}) < \infty$	$\Omega(N^{\frac{1}{2+\epsilon}})$ for any $\epsilon > 0$
X_{ij}'s are light-tailed	$\Omega(\log N)$
X_{ij}'s are deterministic	$\Omega(1)$

Proof. (Sketch) Since X_{ij}'s are light-tailed, we can use Chernoff bound to prove that $\sum_{i=1}^{N} X_{ij}$ or $\sum_{j=1}^{N} X_{ij}$ is stochastically dominated by some (shifted) exponential random variables. It is well-known that the expectation of the maximum of N independent exponential random variables is $O(\log N)$ as $N \to \infty$. Thus, the expected clearance time, as defined in (1), is also $O(\log N)$. The tightness of this bound can be proved by finding a distribution of X_{ij} that has an *exact* exponential decaying tail, such as the geometric distribution. The detailed proof can be found in the technical report [27]. □

Finally, if X_{ij}'s are deterministic, it is clear that $\mathbb{E}[\tau(\mathbf{X})] \leq \beta = \Theta(1)$. Since clearance time is the minimum time to transmit all the packets in a coflow, it is a natural lower bound on coflow-level delay (which is the time between the arrival of a coflow until all the packets associated with this coflow are transmitted). Consequently, the above results essentially impose fundamental limits on the coflow-level delay that can be achieved by any scheduling policy.

Theorem 1. *The expected coflow-level delay achieved by any scheduling policy cannot be better than $O(g(N))$, where*
(1) $g(N) = N^{\frac{1}{1+\epsilon}}$ for any $\epsilon > 0$ if $\mathbb{E}[X_{ij}] < \infty$;
(2) $g(N) = N^{\frac{1}{2+\epsilon}}$ for any $\epsilon > 0$ if Var$(X_{ij}) < \infty$;
(3) $g(N) = \log N$ if X_{ij}'s have light-tailed distributions;
(4) $g(N) = 1$ if X_{ij}'s are deterministic.

The scaling properties of expected coflow-level delay are summarized in Table I. It can be observed that the lower bound on coflow-level delay critically depends on the *variability* of X_{ij}'s: the less X_{ij}'s vary, the smaller lower bound on coflow-level delay we can obtain.

In the rest of this paper, we mainly focus on the case where X_{ij}'s have light-tailed distributions unless otherwise stated. The heavy-tailed case is left for future work.

V. COFLOW-AGNOSTIC SCHEDULING

To gain further insights into the design of coflow-level scheduling policies, we study the performance of some *coflow-agnostic* scheduling policies where coflow-level information (e.g., which packets/flows belong to the same coflow) is not leveraged. In particular, we study the coflow-level performance of two simple scheduling policies: randomized scheduling and periodic scheduling.

Randomized Scheduling. Let $M_1, \cdots, M_{N!}$ be the $N!$ perfect matchings (permutation matrices) associated with the $N \times N$ switch. With the Birkhoff-Von Neumann decomposition, we can find probabilities $\{p_1, p_2, \cdots, p_{N!}\}$ such that the matrix $(\lambda\beta_{ij}) \leq \sum_{k=1}^{N!} p_k M_k$ (where $\beta_{ij} = \mathbb{E}[X_{ij}]$). Such a decomposition is always feasible since $(\lambda\beta_{ij})$ is sub-stochastic by Assumption (4) in Section III-B. In each slot, the randomized policy uses matching M_k as the schedule, with probability p_k. Under uniform traffic, a simple way to implement the randomized policy is to connect the N input ports with a random permutation of the N output ports. Such a policy is guaranteed to stabilize the network as long as $\rho < 1$ although λ and (β_{ij}) need to be known in advance. The detailed description of this policy can be found in [19] and it can be easily shown that the randomized policy achieves $O(N)$ average *packet-level* delay [18].

Periodic Scheduling. This policy is similar to randomized scheduling except that the scheduling decisions are deterministic. Specifically, for some (sufficiently long) period T, we use matching M_k for exactly $p_k T$ times every T slots. Under uniform traffic, a simple way to implement periodic scheduling is to connect each input port i to output port $[(i+t) \bmod N]+1$ in slot t. This policy also achieves $O(N)$ average packet-level delay whenever $\rho < 1$ [18].

Now we analyze the coflow-level delay achieved by the above two policies. In contrast to the simple analysis of packet-level delay, it is non-trivial to analyze the coflow-level delay achieved by these policies, due to the correlation between packets (e.g., packets belonging to the same coflow arrive simultaneously). For ease of exposition, we assume that traffic is uniform such that $\mathbb{E}[X_{ij}] = \frac{\beta}{N}$ and Var$(X_{ij}) = \frac{\sigma^2}{N}$ for all $i, j \in [N]$. We also assume that coflow arrivals are Poisson. The analysis can be easily extended to the general case.

Theorem 2. *Suppose X_{ij}'s have light-tailed distributions. The expected coflow-level delay achieved by the randomized or periodic scheduling policy is $O(N \log N)$ whenever $\rho < 1$.*

Proof. (Sketch) For a certain coflow $\mathbf{X}$, denote by W_i the time to transfer all its packets arriving to input i. Under this notion, the expected coflow-level delay is $\mathbb{E}[\max_i W_i]$. First, it can be shown that $\mathbb{E}[W_i] = O(N)$ and that W_i has a light-tailed distribution by classical queuing theory. Ideally, if we can further prove that W_i's are also *independent* random variables, then the expected coflow-level delay $\mathbb{E}[\max_i W_i]$ is $O(N \log N)$ as $N \to \infty$ by following a similar line of argument in Lemma 3. Unfortunately, due to the simultaneous arrival of all the packets/flows in the same coflow and the potential interleaving introduced by the scheduling policy, W_i's are *correlated*. To remedy this situation, we instead prove that W_i's are *associated random variables* [25]. An important property of associated random variables is that $\mathbb{E}[\max_i W_i] \leq \mathbb{E}[\max_i W_i']$ where W_i''s are *independent* random variables identically distributed as W_i's. As a result, we can conclude that the expected coflow-level delay $\mathbb{E}[\max_i W_i] \leq \mathbb{E}[\max_i W_i'] = O(N \log N)$. The detailed proof is presented in the technical report [27]. □

Remark. Comparing with the $O(N)$ packet-level delay, we can observe a coflow-level delay "dilation" factor of $O(\log N)$. Intuitively, the delay dilation is due to the additional "assembly

delay": packets processed earlier must wait for packets (in the same coflow) that are processed later.

Finally, it is worth mentioning that the randomized or periodic scheduling policy is the simplest throughput-optimal policy in input-queued switches, but it sheds light on the non-triviality of coflow-level analysis (e.g., the correlation between packets) and the potential weakness of coflow-agnostic algorithms (as can be seen from the coflow-level delay dilation). The coflow-level analysis of more sophisticated policies, such as MaxWeight Matching (MWM) scheduling, are very challenging and left for future work. In fact, even the *packet-level* delay of MWM is still an open problem [14], [15].

In conclusion, there has been no throughput-optimal scheduling policy that achieves the provably optimal scaling with respect to coflow-level delay. In the next section, we propose a new coflow-aware scheduling policy that achieves the optimal scaling of coflow-level delay while maintaining throughput optimality and requiring no traffic statistics.

VI. Coflow-Aware Scheduling

In this section, we develop a coflow-aware scheduling policy that achieves $O(\log N)$ expected coflow-level delay whenever $\rho < 1$ under the assumption that arrivals of coflows are Poisson and flow sizes X_{ij}'s are light-tailed. The policy is called the Coflow-Aware Batching (CAB) scheme. Note that in Section IV, we showed that if X_{ij}'s are light-tailed, no scheduling policy can achieve an expected coflow-level delay better than $O(\log N)$. As a result, the CAB policy attains the lower bound, which implies that $O(\log N)$ is optimal scaling of coflow-level delay (under Poisson arrivals and light-tailed flow sizes).

A. Coflow-Aware Batching (CAB) Policy

The basic idea of the CAB policy is to group timeslots into frames of size T slots and clear coflows in batches, where one batch of coflows correspond to the collection of coflows arriving in the same frame. Coflows that are not cleared during a frame are handled separately in future frames. By properly setting the frame size T, the CAB policy can achieve the desirable $O(\log N)$ average coflow-level delay. Note that Neely *et al.* [18] proposed a similar batching scheme to reduce *packet-level* delay. By comparison, our CAB policy explicitly leverages coflow-level information (e.g., which packets belong to the same coflow) to reduce *coflow-level* delay. More importantly, as mentioned in Section V, coflow-level delay analysis is fundamentally different from packet-level analysis. The detailed description of the CAB policy is as follows.

Coflow-Aware Batching (CAB) Scheduling Policy

Setup.

- Timeslots are grouped into frames of size T slots.

Notations.

- Denote by $\mathbf{L}(k) = \big(L_{ij}(k)\big)$ the aggregate traffic matrix of all the coflows arriving in the k-th frame, where $L_{ij}(k)$ is the total number of packets from input i to output j that arrive during the k-th frame.

Procedures.

(1) In the $(k+1)$-th frame, we try to clear the coflows that arrived in the k-th frame, i.e., $\mathbf{L}(k)$. Let $\mathbf{B}(k+1)$ be the traffic matrix we choose to clear in the $(k+1)$-th frame (which may be less than $\mathbf{L}(k)$), and denote by τ the clearance time of $\mathbf{L}(k)$. If $\tau \leq T-1$, then $\mathbf{L}(k)$ can be cleared within the first $T-1$ slots in the $(k+1)$-th frame. In this case, we just set $\mathbf{B}(k+1) = \mathbf{L}(k)$. Note that we only use the first $T-1$ slots in a frame while the remaining slot is reserved for clearing "overflow" coflows as discussed below. If $\tau > T-1$, then *overflow* occurs and only a subset of $\mathbf{L}(k)$ can be cleared in the $(k+1)$-th frame. In this case, we sequentially add coflows to $\mathbf{B}(k+1)$ in order of their arrival in the k-th frame until $\mathbf{B}(k+1)$ becomes maximal, i.e., adding any other coflow will make the clearance time of $\mathbf{B}(k+1)$ exceed $T-1$. If a coflow is selected to $\mathbf{B}(k+1)$, it is referred to as a ***conforming coflow*** otherwise it is called a ***non-conforming coflow***.

(2) All the conforming coflows that arrive in the k-th frame are scheduled during the $(k+1)$-th frame by clearing $\mathbf{B}(k+1)$ in minimum time using an optimal clearance algorithm (e.g., see [16]).

(3) All the non-conforming coflows are put into a separate FIFO queue. In the last slot of each frame, this FIFO queue gets served by the switch. Note that non-conforming coflows are served one at a time, and the service time (measured in the number of *frames*) of each non-conforming coflow is its clearance time.

In words, the first $T-1$ slots in a frame are used to serve conforming coflows arriving in the previous frame and the remaining slot is reserved to serve non-conforming coflows in a FIFO manner. Note that conforming coflows (that arrive in the same frame) are cleared together in a batch while non-conforming coflows are served one at a time in the separate FIFO queue. Under the CAB policy, either all the packets in a coflow are conforming or none of them are conforming.

B. Performance of the CAB policy

The following theorem shows that the CAB policy achieves $O(\log N)$ expected coflow-level delay whenever $\rho < 1$ (under Poisson coflow arrivals and light-tailed flow sizes).

Theorem 3 (Average Coflow-level Delay)**.** *Suppose coflows arrive according to a Poisson process and flow sizes are light-tailed. By selecting a proper frame size $T = O(\log N)$, the CAB policy achieves $O(\log N)$ expected coflow-level delay if $\rho < 1$.*

The choice of T will be specified later in Section VI-C. In fact, the CAB policy not only guarantees that the *average* coflow-level delay is $O(\log N)$ but also ensures that the $O(\log N)$ delay is achievable for an arbitrary coflow with high probability.

Corollary 1 (Tail Coflow-level Delay)**.** *By selecting a proper frame size $T = O(\log N)$, the CAB policy achieves $O(\log N)$*

delay for an arbitrary coflow with probability $1 - O(\frac{1}{N^2})$ *whenever* $\rho < 1$.

In the following, we present a proof sketch for the above results while the detailed proof is given in the technical report [27]. The proof itself suggests the choice of T.

C. Proof Sktech to Theorem 3 and Corollary 1

For simplicity, we assume that traffic is uniform such that $\mathbb{E}[X_{ij}] = \frac{\beta}{N}$ and $\text{Var}(X_{ij}) = \frac{\sigma^2}{N}$ for all $i, j \in [N]$. The analysis can be easily extended to non-uniform traffic.

We discuss the expected coflow-level delay experienced by a conforming and a non-conforming coflow, respectively.

- Conforming coflows are cleared within 2 frames: the frame where they arrive plus the frame where they are cleared. As a result, the coflow-level delay experienced by a conforming coflow is at most $2T$ time slots, i.e.,

$$\text{Delay(conforming)} \leq 2T.$$

- A non-conforming coflow first waits for at most T slots (the frame where it arrives) and then waits in the separate FIFO queue. As a result, the coflow-level delay experienced by a non-conforming coflow is

$$\text{Delay(non-conforming)} \leq T + \text{Delay(FIFO queue)}.$$

Let η be the long-term fraction of non-conforming coflows. Then the average coflow-level delay of an *arbitrary coflow* is

$$\mathbb{E}[W] \leq (1-\eta)2T + \eta[T + \text{Delay(FIFO queue)}]. \quad (2)$$

In the following, we choose $T = O(\log N)$ (the specific value of T will be made clear later). Under such a choice of T, we prove that η is miniscule and Delay(FIFO queue) is not very large. Thus, it can be concluded that $\mathbb{E}[W] = O(\log N)$.

Step 1: Determine the value of T.

We first show that the overflow probability decreases exponentially with the frame size T.

Lemma 4. *Let P_o be the overflow probability in an arbitrary frame. If $\rho < 1$, there exists some constant $\gamma > 0$ such that*

$$P_o \leq 2N \exp(-\gamma T). \quad (3)$$

Proof. (Sketch) Note that an overflow occurs in the k-th frame if the clearance time of $\mathbf{L}(k)$ is greater than $T-1$ slots, i.e., if any of the following $2N$ inequalities is violated.

$$\begin{aligned} \sum_j L_{ij}(k) < T, \ i = 1, \cdots, N, \\ \sum_i L_{ij}(k) < T, \ j = 1, \cdots, N, \end{aligned} \quad (4)$$

where $L_{ij}(k)$ is the number of packets that arrive to queue (i,j) during the k-th frame. Clearly, $\sum_j L_{ij}(k)$ (or $\sum_i L_{ij}(k)$) is the total number of packets that arrive to input i (or destined for output j) during the k-th frame, which corresponds to the number of packet arrivals during T time slots in a *Poisson process with batch arrivals* where the arrival rate is λ and the mean batch size is β.

Let $Y(T)$ be the total number of packet arrivals during T time slots in the above *batch Poisson process*. It is clear that $Y(T) = \sum_{n=1}^{N(T)} B_n$. Here, $N(T)$ is the number of coflow arrivals during the T time slots, and has a Poisson distribution with rate λT; B_n is the number of packets brought by the n-th coflow to a certain input or output port, which is identically distributed as $\sum_j X_{ij}$ or $\sum_i X_{ij}$ and $\mathbb{E}[B_n] = \beta$. By Wald's equality, we have $\mathbb{E}[Y(T)] = \mathbb{E}[N(T)]\mathbb{E}[B_1] = \rho T < T$ if $\rho < 1$. By the Chernoff bound, we can prove that there exists some constant $\gamma > 0$ such that

$$\mathbb{P}[Y(T) \geq T] \leq \exp(-\gamma T). \quad (5)$$

Applying the union bound, we can conclude that the overflow probability (i.e., at least one of the $2N$ inequalities in (4) is violated) is bounded by $P_o \leq 2N \exp(-\gamma T)$, which completes the proof. □

Remark 1. Lemma 4 implies that if we want to keep the overflow probability below δ, we can choose $T \geq \frac{\log(2N/\delta)}{\gamma}$, where $\gamma > 0$ is some constant independent of N. Since T is an integer, we can choose

$$T = \lceil \frac{\log(2N/\delta)}{\gamma} \rceil. \quad (6)$$

The value of δ will be specified later such that $T = O(\log N)$ and the average coflow-level delay is also $O(\log N)$. The constant γ can also be found in a systematic way but omitted here for brevity (see the technical report [27] for details).

Remark 2. Lemma 4 holds only if $\rho < 1$. If $\rho \geq 1$, $\mathbb{E}[Y(T)] = \rho T \geq T$ and the Chernoff bound (5) does not hold.

Step 2: Determine the value of η.

Next, we derive an upper bound for the long-term fraction of non-conforming coflows, i.e.,

$$\eta = \lim_{K \to \infty} \frac{\sum_{k=1}^{K} A_{non}(k)}{\sum_{k=1}^{K} A(k)},$$

where $A(k)$ is the total number of coflows that arrive during the k-th frame and $A_{non}(k)$ is the number of non-conforming coflows in the k-th frame. We begin by identifying a stochastic bound for the number of coflow arrivals in an **overflow** frame.

Lemma 5. *Suppose $N(T)$ is a Poisson random variable with rate λT. Given that an overflow occurs in a frame, the number of coflows arrivals in this frame is stochastically dominated by $N(T) + T/\beta$ when T is sufficiently large.*

Proof. (Sketch) Let $N(T)$ be the total number of coflow arrivals in an arbitrary frame of T time slots. Clearly, $N(T)$ is a Poisson random variable with rate λT. Denote by $\mathbf{X}^{(n)} = (X_{ij}^{(n)})$ the traffic matrix of the n-th coflow, and let $\mathbf{L}$ be the aggregate traffic matrix of these coflows, i.e., $\mathbf{L} = \sum_{n=1}^{N(T)} \mathbf{X}^{(n)}$. It is clear that an overflow occurs if the clearance time of $\mathbf{L}$ is greater than $T-1$, i.e., $\tau(\mathbf{L}) \geq T$. We first find a lower bound on the overflow probability when T is sufficiently large.

Claim 1. *There exists some $T_0 > 0$ such that for any $T > T_0$ the overflow probability*

$$\mathbb{P}\Big[\tau(\mathbf{L}) \geq T\Big] \geq \mathbb{P}\Big[N(T) \geq T/\beta\Big].$$

This claim can be proved by applying Central Limit Theorem.

Next we evaluate the probability that there are at least m coflow arrivals in an overflow frame.

Claim 2. *Given that an overflow occurs in a frame, the probability that there are at least m coflow arrivals in this frame is upper bounded by $\mathbb{P}[N(T) \geq m | N(T) \geq T/\beta]$ when T is sufficiently large, i.e.,*

$$\mathbb{P}\Big[N(T) \geq m \Big| \tau(\mathbf{L}) \geq T\Big] \leq \mathbb{P}\Big[N(T) \geq m \Big| N(T) \geq T/\beta\Big].$$

This claim can be easily proved by using Claim 1 and elementary probability calculations.

Claim 3. *If $N(T)$ is a Poisson random variable, then $\mathbb{P}[N(T) \geq m + r | N(T) \geq r] \leq \mathbb{P}[N(T) \geq m]$.*

This claim was proved in Appendix B of [18].

The above claims imply that when T is sufficiently large

$$\begin{aligned}\mathbb{P}\Big[N(T) \geq m \Big| \tau(\mathbf{L}) \geq T\Big] &\leq \mathbb{P}\Big[N(T) \geq m \Big| N(T) \geq \frac{T}{\beta}\Big] \\ &\leq \mathbb{P}\Big[N(T) \geq m - \frac{T}{\beta}\Big] \\ &= \mathbb{P}\Big[N(T) + \frac{T}{\beta} \geq m\Big],\end{aligned}$$

where the first inequality is due to Claim 2 and the second inequality is due to Claim 3. The above inequalities imply that given an overflow occurs in a frame, the number of coflow arrivals in this frame is stochastically dominated by $N(T)+T/\beta$ when T is sufficiently large. This completes the proof of Lemma 5. □

With Lemma 5, we can find an upper bound for the long-term fraction of non-conforming coflows.

Lemma 6. *If the overflow probability is δ, the long-term fraction of non-conforming coflows among all coflows is upper bounded by $\eta \leq \frac{2\delta}{\rho}$ when the frame size T is sufficiently large.*

Proof. According to Lemma 5, the expected number of non-conforming coflows in an overflow frame is at most $\lambda T + T/\beta$ (note that this is a loose bound since we treat all the coflows in an overflow frame as non-conforming coflows). As a result, the long-term fraction of non-conforming coflows is

$$\lim_{K\to\infty} \frac{\sum_{k=1}^{K} A_{non}(k)}{\sum_{k=1}^{K} A(k)} \leq \lim_{K\to\infty} \frac{\delta K(\lambda T + T/\beta)}{K\lambda T} = \frac{\delta(\lambda T + T/\beta)}{\lambda T}.$$

Note that the inequality holds because the number of overflow frames is δK as $K \to \infty$ and the average number of non-conforming coflows in each overflow frame is at most $\lambda T + T/\beta$. Noticing that $T \geq 1$ and $\rho = \lambda\beta < 1$, we have $\eta \leq \frac{\delta(\lambda T + T/\beta)}{\lambda T} = \delta(1 + \frac{T}{\rho T}) \leq \delta(\frac{1}{\rho} + \frac{1}{\rho}) = \frac{2\delta}{\rho}$, which completes the proof. □

Step 3: Determine delay in the separate FIFO queue.

The third step is to find an upper bound for the average delay experienced by non-conforming coflows in the separate FIFO queue. Note that Lemma 4 shows that whenever $\rho < 1$, the overflow probability δ can be made arbitrarily small by setting the frame size T as in (6). In particular, we can choose the frame size to be $T = \lceil \frac{\log(2N/\delta)}{\gamma} \rceil = O(\log N^3) = O(\log N)$ to achieve the overflow probability $\delta = O(\frac{1}{N^2})$. Under such a choice of T, we can prove the following lemma.

Lemma 7. *Under a proper choice of T, the FIFO queue holding non-conforming coflows is stable whenever $\rho < 1$ and the average delay experienced by non-conforming coflows in the FIFO queue is $O(NT^3)$.*

Proof. (Sketch) Note that non-conforming coflows are placed in a discrete-time FIFO queue and are served one at a time. At the end of each frame, a batch of non-conforming coflows arrives to this queue, with probability δ. Since non-conforming coflows arrive to the FIFO queue in batches, the arrivals of non-conforming coflows are dependent. To circumvent this dependence, we notice that the arrivals of different batches of non-conforming coflows are independent: in each frame, there is a batch arrival with probability δ, independent of any other frames. As a result, we overestimate the delay of any *individual* non-conforming coflow by the delay experienced by the *entire batch* of non-conforming coflows (i.e., the time between the arrival of the entire batch and the completion of all the non-conforming coflows in this batch) plus T slots (the size of the frame in which the non-conforming coflows arrive).

As an overestimate, all the coflows that arrive in an overflow frame are treated as non-conforming coflows. Let M be the total number of non-conforming coflows in an overflow frame, and denote by $\tilde{\mathbf{X}}^{(n)} = (\tilde{X}_{ij}^{(n)})$ the traffic of the n-th non-conforming coflow in this overflow frame. Let $\tilde{\mathbf{L}} = \sum_{n=1}^{M} \tilde{\mathbf{X}}^{(n)}$ be the aggregate traffic matrix associated with these non-conforming coflows. It follows that the service time for the entire batch of non-conforming coflows is $U = \tau(\tilde{\mathbf{L}})$ (measured in the number of frames since only one slot per frame is used to serve non-conforming coflows). Clearly, the service times for different batches of non-conforming coflows are independent. As a result, if the entire batch of non-conforming coflows is treated as a "customer", the FIFO queue is a discrete-time GI/GI/1 queue with Bernoulli arrivals (of rate δ per frame) and general service time U. The average waiting time (measured in the number of frames) in such a system can be exactly characterized [26]:

$$\text{Delay(FIFO queue)} = \frac{\delta\mathbb{E}[U^2] - \delta\mathbb{E}[U]}{2(1 - \delta\mathbb{E}[U])} + \mathbb{E}[U]. \quad (7)$$

It can be shown (see the technical report [27] for details) that

$$\mathbb{E}[U] = \Theta(NT^2), \quad \mathbb{E}[U^2] = \Theta(N^2T^4). \quad (8)$$

Taking (8) into (7), we have

$$\text{Delay(FIFO queue)} \leq O(T^4) + O(NT^2) = O(NT^2).$$

Note that the above delay is measured in the number of *frames*, which implies that the expected delay (measured in the number of *timeslots*) in the FIFO queue is $O(NT^3)$.

Finally, it is worth mentioning that the GI/GI/1 queue is stable whenever $\delta\mathbb{E}[U] < 1$. Since $\delta = O(\frac{1}{N^2})$ and $\mathbb{E}[U] = \Theta(NT^2) = \Theta(N\log^2 N)$, such an inequality can be made to hold true for any $N \geq 1$ under a suitably small δ. □

Remark. If $\rho \geq 1$, then Lemma 4 does not hold and the overflow probability δ cannot be made arbitrarily small regardless of the choice of T. In this case, the FIFO queue holding non-conforming coflows is unstable.

Step 4: Putting it all together.

Finally, we can evaluate the average coflow-level delay experience by both conforming and non-conforming coflows. By Lemma 6, the fraction of non-conforming coflows is at most $\eta \leq \frac{2\delta}{\rho}$. If $\rho < 1$, Lemma 4 shows that we can choose the frame size to be $T = \lceil \frac{\log(2N/\delta)}{\gamma} \rceil = O(\log N^3) = O(\log N)$ to achieve the overflow probability $\delta = O(\frac{1}{N^2})$. Under such a choice of T, Lemma 7 shows that Delay(FIFO queue) $= O(NT^3)$. Taking the values of T, η and Delay(FIFO queue) into (2), we can conclude that the average coflow-level delay under the CAB policy is

$$\begin{aligned}\mathbb{E}[W] &\leq (1-\frac{2\delta}{\rho})2T + \frac{2\delta}{\rho}[T + \text{Delay(FIFO queue)}] \\ &\leq 2T + \frac{2\delta}{\rho}\text{Delay(FIFO queue)} \\ &= O(T) + O(\delta NT^3) = O(T) = O(\log N).\end{aligned} \tag{9}$$

This completes the proof to Theorem 3.

The above proof to Theorem 3 shows that the coflow-level delay experienced by any conforming coflow is no greater than $2T = O(\log N)$ and the fraction of conforming coflows is more than $1 - \frac{2\delta}{\rho} = 1 - O(\frac{1}{N^2})$. As a result, the CAB policy also ensures that the $O(\log N)$ delay is achievable for an arbitrary coflow with high probability $1 - O(\frac{1}{N^2})$, which completes the proof to Corollary 1.

D. Discussions

Computational Complexity. The computational complexity of the CAB policy can be analyzed in a similar way to the original batching policy [18]. The computational complexity is $O(N^{1.5}\log N)$ per slot.

Choosing Parameters. In the CAB policy, the frame size is set to be $T = \lceil \frac{\log(2N/\delta)}{\gamma} \rceil$. As a result, we need to choose the parameters δ and γ. In the technical report [27], we present a rigorous way in obtaining the value of δ and γ.

VII. Simulation Results

In this section, we numerically evaluate the coflow-level performance achieved by the CAB policy.

A. Simulation Setup

In our simulations, coflows arrive to the system according to a Poisson process with rate $\lambda = 0.3$ (per slot). The flow sizes (X_{ij}'s) follow a geometric distribution with mean $\frac{\beta}{N}$ where $\beta = 2.5$ (measured in the number of packets). Hence, the offered load is $\rho = 0.75$. The simulation is run for a sufficiently long time (10^6 slots) such that the steady state is reached. The frame size T is set as in equation (6) where the parameters γ and δ can be obtained in a systematic way described in the technical report [27].

B. Scaling with $N \to \infty$

First, we evaluate the scaling of coflow-level delay as $N \to \infty$. The following schemes are compared:

(1) CAB policy.

(2) Randomized scheduling as described in Section V.

(3) Max-Weight Matching (MWM) scheduling: the schedule in slot t is the maximum weight matching of $\mathbf{Q}(t)$ where $\mathbf{Q}(t) = (Q_{ij}(t))$ is the queue length matrix in slot t.

Figure 2 shows the comparison of these schemes with respect to the average coflow-level delay, where the horizontal axis is on a logarithmic scale. As the theoretical bound suggests, the CAB policy achieves the logarithmic scaling as $N \to \infty$ (i.e., a straight line in the figure). By comparison, the average coflow-level delay achieved by the randomized scheme grows much faster with N. Moreover, it can be observed that the CAB policy outperforms the randomized scheme even for very small N (e.g., $N = 40$).

Another interesting observation is that the MWM policy has an exceptional coflow-level performance. It is observed that the MWM policy empirically achieves the optimal logarithmic coflow-level delay scaling as $N \to \infty$. The MWM policy also slightly outperforms the CAB policy by some constant factor. Unfortunately, the coflow-level delay analysis of the MWM policy is very challenging and left for future work.

C. Coflow-level Delay Dilation

Next, we compare the coflow-level delay with the packet-level delay under the randomized policy and the CAB policy. In particular, we are interested in the *coflow-level delay dilation factor* which is the ratio between the average coflow-level delay and the average packet-level delay. As is illustrated in Figure 3, the randomized policy has a coflow-level delay dilation factor of $O(\log N)$; this observation empirically validates the tightness of the $O(N\log N)$ bound shown in Theorem 2 (note that the average packet delay achieved by the randomized policy is exactly $\Theta(N)$). By comparison, the delay dilation factor for the CAB policy remains at a constant level as $N \to \infty$, which shows the benefits of "coflow-awareness".

D. Scaling with $\rho \to 1$

Finally, we numerically study the sensitivity of the coflow-level performance under different scheduling policies as the offered load $\rho \to 1$. This is shown in Figure 4. Clearly, the CAB policy is more sensitive to the offered load ρ than the

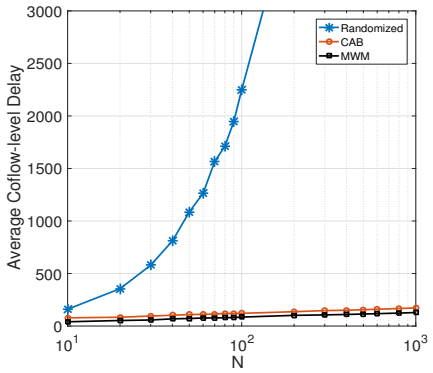

Fig. 2. Average coflow-level delay under different scheduling policies. Note that the horizontal axis is in the log scale.

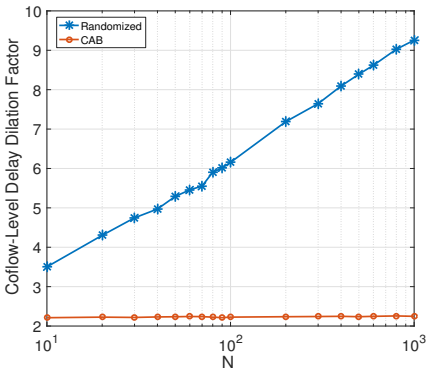

Fig. 3. Coflow-level delay dilation factor (the ratio between average coflow-level delay and packet-level delay) under different scheduling policies. Note that the horizontal axis is in the log scale.

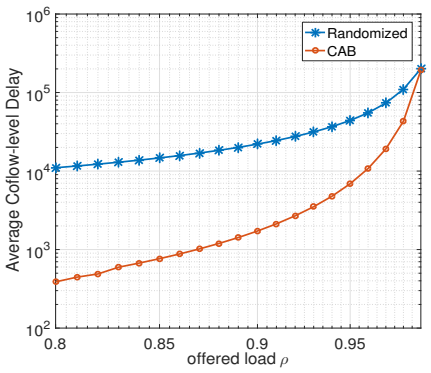

Fig. 4. Scaling of average coflow-level delay as $\rho \to 1$ ($N = 200$). Note that the vertical axis is in the log scale.

randomized policy. In the heavy-traffic regime, the randomized policy even outperforms the CAB policy. Indeed, it is shown in the technical report [27] that the average coflow-level delay achieved by the randomized policy grows as $O(\frac{1}{1-\rho})$ as $\rho \to 1$. By comparison, the CAB policy achieves $O(\frac{1}{(1-\rho)^2})$ average coflow-level delay as $\rho \to 1$. As a result, the price for the better scaling with N is the worse dependence on ρ.

VIII. Conclusion

In this paper, we investigate the optimal scaling of coflow-level delay in an $N \times N$ input-queued switch as $N \to \infty$. We develop lower bounds on the coflow-level delay that can be achieved by any scheduling policy. In particular, when flow sizes have light-tailed distributions, the lower bound $O(\log N)$ can be attained by the proposed Coflow-Aware Batching (CAB) policy. Thus, the optimal scaling of coflow-level delay is $O(\log N)$ under light-tailed flow sizes.

References

[1] J. Dean and S. Ghemawat. MapReduce: Simplified data processing on large clusters. in *OSDI*, 2004.

[2] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker, I. Stoica. Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing. in *USENIX NSDI*, 2012.

[3] M. Chowdhury, M. Zaharia, J. Ma, M. I. Jordan, and I. Stoica. Managing data transfers in computer clusters with orchestra. in *Proceedings of the ACM SIGCOMM*, pp. 98–109, 2011.

[4] M. Chowdhury and I. Stoica. Coflow: A Networking Abstraction for Cluster Applications. in *ACM Hotnets*, 2012.

[5] M. Chowdhury, Y. Zhong, and I. Stoica. Efficient Coflow Scheduling with Varys. in *ACM SIGCOMM*, 2014.

[6] M. Chowdhury and I. Stoica. Efficient Coflow Scheduling Without Prior Knowledge. in *ACM SIGCOMM*, 2015.

[7] Y. Zhao *et al.*. RAPIER: Integrating Routing and Scheduling for Coflow-aware Data Center Networks. in *IEEE INFOCOM*, 2015.

[8] F. R. Dogar, T. Karagiannis, H. Ballani, and A. Rowstron. Decentralized Task-Aware Scheduling forData Center Networks. in *ACM SIGCOMM*, 2014.

[9] I. A. Rai, G. Urvoy-Keller, and E. W. Biersack. Analysis of LAS scheduling for job size distributions with high variance. in *ACM SIGMETRICS Performance Evaluation Review*, vol. 31, no. 1, pp. 218–228, 2003.

[10] S. Luo *et. al*. Minimizing Average Coflow Completion Time with Decentralized Scheduling. in *IEEE ICC*, 2015.

[11] Z. Qiu *et al.*. Minimizing the total weighted completion time of coflows in datacenter networks. in *SPAA*, 2015.

[12] D. Shah, N. Walton, and Y. Zhong. Optimal Queue-Size Scaling in Switched Networks. in *ACM SIGMETRICS*, 2012.

[13] D. Shah and M. Kopikare. Delay bounds for the approximate Maximum Weight matching algorithm for input queued switches. in *IEEE Infocom*, 2002.

[14] S. T. Maguluri and and R. Srikant. Heavy-Traffic Behavior of the MaxWeight Algorithm in a Switch with Uniform Traffic. in *ACM SIGMETRICS Performance Evaluation Review*, vol. 43, no. 2, pp. 72-74, 2015.

[15] S. T. Maguluri, S. K. Burle and R. Srikant. Optimal Heavy-Traffic Queue Length Scaling in an Incompletely Saturated Switch. in *ACM SIGMETRICS*, 2016.

[16] T. Inukai. An Efficient SS/TDMA Time Slot Assignment Algorithm. in *Transactions on Communications*, vol. 27, no. 10, pp. 1449-1455, 1979.

[17] N. Papadatos. Maximum variance of order statistics. in *Ann. Inst. Statist. Math.*, vol. 47, no. 1, pp. 185-193, 1995.

[18] M. Neely, E. Modiano, and Y. S. Cheng. Logarithmic delay for $n \times n$ packet switches under the cross-bar constraint. in *IEEE/ACM Transactions on Networking*, vol. 15, no. 3, 2007.

[19] C-S Chang, W-J Chen, and H-Y Huang. Birkhoff-von Neumann input buffered crossbar switches. in *Proc. IEEE INFOCOM*, 2000.

[20] N. McKeown, V. Anantharam, and J. Walrand. Achieving 100% throughput in an input-queued switch. in *IEEE INFOCOM*, 1996.

[21] E. Leonardi, M. Mellia, F. Neri, and M. Ajmone Marsan. Bounds on average delays and queue size averages and variances in input-queued cell-based switches. in *IEEE INFOCOM*, 2001.

[22] L. P. Devroye. Inequalities for the completion times of stochastic Pert networks. in *Mathematics of Operations Research*, vol. 4, no. 4, pp. 441-447, 1979.

[23] R. Nelson and A. N. Tantawi. Approximate Analysis of Fork/Join Synchronization in Parallel Queues. in *IEEE Transactions on Computers*, vol. 37, no. 6, 1988.

[24] A. A. Borovkov. Stochastic Processes in Queueing Theory (English translation). Springer-Verlag, New York, 1976.

[25] J. D. Esary, F. Proschan, and D. W. Walkup. Association of Random Variables, with Applications. in *Ann. Math. Statist.*, Vol. 38, No. 5, pp. 1466-1474, 1967.

[26] Torben Meisling. Discrete-Time Queuing Theory. in *Operations Research*, Vol. 6, No. 1, pp. 96-105, 1957.

[27] Q. Liang and E. Modiano. Coflow Scheduling in Input-Queued Switches: Optimal Delay Scaling and Algorithms. *arXiv:1701.02419*, 2017.