Information-agnostic Coflow Scheduling with Optimal Demotion Thresholds

Yuanxiang Gao[1], Hongfang Yu[1], Shouxi Luo[1] and Shui Yu[2]
[1]Key Laboratory of Optical Fiber Sensing and Communications, Ministry of Education
University of Electronic Science and Technology of China, Chengdu, P. R. China
[2] School of IT, Deakin University, Australia

***Abstract*—Previous coflow scheduling proposals improve the coflow completion time (CCT) over per-flow scheduling based on prior information of coflows, which makes them hard to apply in practice. State-of-art information-agnostic coflow scheduling solution Aalo adopts Discretized Coflow-aware Least-Attained-Service (D-CLAS) to gradually demote coflows from the highest priority class into several lower priority classes when their sent-bytes-count exceeds several predefined demotion thresholds.**

However, current design standards of these demotion thresholds are crude because they do not analyze the impacts of different demotion thresholds on the average coflow delay. In this paper, we model the D-CLAS system by an M/G/1 queue and formulate the average coflow delay as a function of the demotion thresholds. In addition, we prove the valley-like shape of the function and design the Down-hill searching (DHS) algorithm. The DHS algorithm locates a set of optimal demotion thresholds which minimizes the average coflow delay in the system. Real-data-center-trace driven simulations indicate that DHS improves average CCT up to 6.20× over Aalo.

I. Introduction

A coflow is a collection of semantic-related flows between two groups of machines in Data Center Networks (DCNs) [1]. Previous coflow-aware schedulers [2-6] achieve superior CCT performance over per-flow schedulers [7-9] based on prior knowledge of coflows (e.g. sizes of coflows before transmitting). These information-aware coflow schedulers are hard to implement because the prior knowledge of coflows is unknown in practice [10]. Thus, state-of-art information-agnostic coflow scheduler Aalo [10] alters Coflow-aware Least-Attained-Service (CLAS) which needs infinite number of priorities to Discretized-CLAS (D-CLAS) due to limited number of priorities, 2-8 in commodity switches [9]. In a D-CLAS system, a coflow is gradually demoted from the highest priority class to several lower priority classes when its sent-bytes-count exceeds several predefined thresholds. If a system uses P priorities, system designers need to predefine a demotion-thresholds-vector $[x_1, x_2, ..., x_{P-1}]$ based on which coflows are demoted gradually.

Existing design standards of the demotion-thresholds-vector focus on setting the spaces between each threshold such as uniformly-spaced scheme [10] and exponentially-spaced scheme [10]. In uniformly-spaced scheme, the spaces between each threshold are equally valued by a constant, for example $\frac{1}{P}$ of maximal coflow size. In exponentially-spaced scheme, the spaces between each threshold are exponentially increasing with a constant increasing rate, typically 10. Current design standards do not take into account the statistic characteristics of coflows (e.g. coflow size distribution, traffic intensity). Hence, they are crude for the goal of minimizing average CCT.

In order to analyze the impacts of the demotion-thresholds-vector on delay features of a D-CLAS system, we model a D-CLAS system through an M/G/1 priority queue and derive formulations of the average coflow queueing delays. Through re-transforming these formulations, we derive the *improvement ratio function* which quantifies the average queueing delay improvements of D-CLAS systems over a FIFO (First-In-First-Out) system under the impacts of different demotion-thresholds-vectors. Thus, we design the optimal demotion-thresholds-vector which results in minimal average coflow queueing delay by searching the minimum of the *improvement ratio function*.

However, the searching space for locating the minimum of the *improvement ratio function* is huge (about 10^6). To reduce the searching space, we prove that the valley-like shape of the *improvement ratio function* curves owns *three-stage property*, namely, *down-hill stage*, *minimal stage* and *uphill stage*. Through picking only the *down-hill stage* and *minimal stage* as our searching space, we design the *down-hill searching* (DHS) algorithm which searches the minimum of *improvement ratio function* within a narrow searching space (1.25% of total searching space) and the DHS algorithm is faster than general searching algorithms ($80\times$).

We evaluate DHS through extensive trace-driven simulations. Our simulation results demonstrate that DHS decreases the average CCT of Aalo by $1.08\times$ to $6.20\times$.

In summary, the contributions of this paper are twofold:

1. We model the D-CLAS system by an M/G/1 priority queue and formulate the average coflow queueing delays. Through re-transforming corresponding formulations, we derive the *improvement ratio function* which is an average delay function with respect to the demotion-thresholds-vector and related to statistic characteristics of coflows.

This work was supported in part by the 973 Program under Grant No. 2013CB329103, the 863 Program under Grant No. 2015AA015702 and 2015AA016102, the National Natural Science Foundation of China under Grant No. 61271171, 61271165, and 61571098, and the Ministry of Education – China Mobile Research Fund under Grant No. MCM20130131, China Postdoctoral Science Foundation (2015M570778)

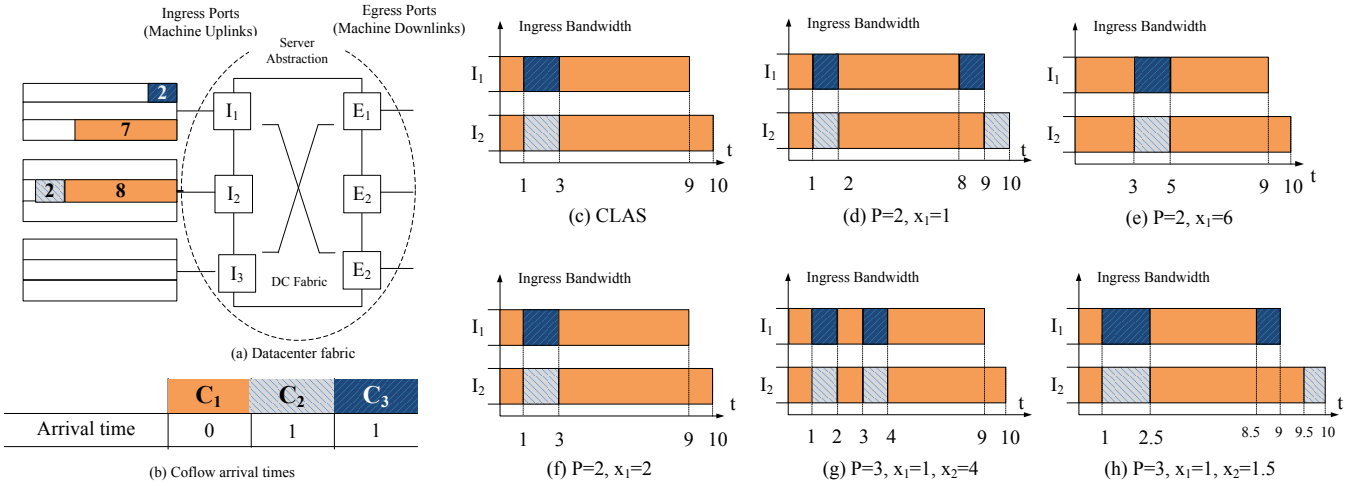


Fig. 1. Motivation Example. (a) Online coflow-aware scheduling over a 3×3 data center fabric; (b) Coflow arrival times; Assuming the service rate of each port is 1 data unit per second; Strict priority is adopted between coflows with different priorities and first-in-first-out (FIFO) order is used for coflows sharing a same priority; The average CCT for (c) CLAS (infinite number of priorities) is 4.67 time units; (d) D-CLAS with $P = 2$, $x_1 = 1$ is 8.67 time units; (e) $P = 2$, $x_1 = 6$ is 6 time units; (f) $P = 2$, $x_1 = 2$ is 4.67 time units; (g) $P = 3$, $x_1 = 1$ and $x_2 = 4$ is 5.33 time units; (h) $P = 3$, $x_1 = 1$ and $x_2 = 1.5$ is 8.83 time units.

Hence we search the minimum of the function to obtain an optimal demotion-thresholds-vector.

2. We prove the *three-stage property* through which we show the valley-like shape of the *improvement ratio function*. Furthermore, we utilize the *three-stage property* to design the *down-hill searching* (DHS) algorithm which searches the minimum of the *improvement ratio function* within a narrower searching space compared with total searching space (1.25%) and DHS is faster than general searching algorithms ($80\times$).

The rest of the paper is organized as follows. Section II illustrates the motivation of our work. In Section III, the M/G/1 queueing model is applied to derive the *improvement ratio function*. In Section IV, we prove the *three-stage property* of the *improvement ratio function* and the DHS algorithm is proposed. Section V evaluates the average CCT improvements of DHS over Aalo through trace-driven simulations. Section VI investigates related work and Section VII concludes the paper and sheds light on our future direction.

II. Motivation

In this section we explain why it is necessary to apply a proper demotion-thresholds-vector for D-CLAS scheduling.

Consider the example in Fig. 1 that compares D-CLAS schemes with different demotion-thresholds-vector settings. In Fig. 1(a) there are three coflows waiting for transmitting in DC fabric (which abstracts the entire data center as one non-blocking switch [8], [10]) with three ingress/egress ports. This example only considers coflows without egress contention, but it is enough to expose our key observations. Fig. 1(b) shows the arrival time of each coflow.

The first takeaway is that *given the number of priorities P, the demotion thresholds should be carefully designed for the goal minimizing average CCT.* This insight exposes by comparing Fig. 1(c)$\sim$(f). For CLAS with infinite number of priorities (continuous priorities) in the system as shown in Fig. 1(c), the coflows $< C_1, C_2, C_3 >$ are completed at time $< 10, 3, 3 >$ respectively. Thus, the CCTs of $< C_1, C_2, C_3 >$ are $< 10, 2, 2 >$ respectively and corresponding average CCT is $\frac{10+2+2}{3} = 4.67$. However, for D-CLAS scheduling with two priorities, coflows are demoted from higher priority to lower priority only when their already sent data exceeds a predefined demotion threshold x_1. In Fig. 1(d), a careless choice of the demotion threshold $x_1 = 1$ harms average CCT as $\frac{9+9+8}{3} = 8.67$. Similarly, as Fig. 1(e) shows, another irrationally chosen demotion threshold $x_1 = 6$ suffers average CCT as $\frac{10+4+4}{3} = 6$. Finally, a well-designed demotion threshold $x_1 = 2$ in Fig. 1(f) achieves average CCT as 4.67, same as CLAS case shown in Fig. 1(c). The average CCT improvements of the well-designed demotion threshold $x_1 = 2$ is $\frac{8.67}{4.67} = 1.86\times$ over the carelessly-designed demotion threshold $x_1 = 1$.

The second takeaway is that *simply increasing the number of priorities without designing the demotion thresholds properly does not necessarily result in smaller average CCT.* This observation is uncovered by comparing Fig. 1(g) with Fig. 1(d) and Fig. 1(h) with Fig. 1(f). In Fig. 1(g), three priorities are adopted. An additional demotion threshold $x_2 = 4$ does vastly reduce the average CCT as $\frac{10+3+3}{3} = 5.33$ relative to 8.67 given by Fig. 1(d) where $P = 2$ and x_1 is also 1. However, counter-intuitively, see Fig. 1(h), if we pick the x_2 carelessly as $x_2 = 1.5$, the average CCT is even deteriorated as $\frac{9.5+9+8}{3} = 8.83$, much worse than the average CCT 4.67 resulting from the well-designed demotion threshold $x_1 = 2$ with only two priorities as Fig. 1(f) shows. The average CCT improvements of the well-designed demotion threshold $x_1 = 2$ is $\frac{8.83}{4.67} = 1.89\times$ over the carelessly-designed demotion-thresholds-vector $[x_1, x_2] = [1, 1.5]$.

We summarize the two observations obtained from the motivating example as follows:

(1). A proper design of the demotion-thresholds-vector vastly decreases the average CCT compare to careless designs.

(2). The design of the demotion-thresholds-vector is closely related to the statistic characteristics of coflows such as coflow size distribution, coflow arrival time or traffic intensity as shown in Fig. 1(a) and Fig. 1(b).

Motivated by the two key observations exhibited above, we take into account the statistic characteristics of coflows for our design of an optimal demotion-thresholds-vector by the M/G/1 priority queueing model below.

III. Model and Analysis

In this section, we first abstract a D-CLAS system as an M/G/1 queue [12] in Section III-A. Then, we re-transform the average coflow queueing delay formulations to derive the *improvement ratio function* in Section III-B.

A. System Model

We start this part by introducing two abstractions of a D-CLAS system from a queueing theory perspective.

- **Server abstraction of the DCN:** Many previous flow scheduling work [8], [10] in DCN have abstracted the whole DCN as a DC fabric in Fig. 1(a). Following their efforts, we further abstract the whole DCN as a server where coflows arrive, get serviced then departure as shown by the dash circle in Fig. 1(a).
- **Customer abstraction of coflows:** D-CLAS systems ensure all flows belonging to a coflow share a consistent priority across the DCN and get serviced as a whole [10]. Thus, each coflow in Fig. 1(a) can be abstracted as a customer in the M/G/1 queueing model.

In addition, by the assumption that coflows arrive in a Poisson process, a D-CLAS scheduling system is modeled by an M/G/1 queueing system.

B. Demotion Thresholds Analysis

In order to derive the *improvement ratio function* for a D-CLAS system, we start this part by some definitions.

Definitions: We denote the arrival rate of coflows across the DCN as λ. The distribution of coflow sizes is a random variable X. The probability density function of X is denoted as $b(x)$. The cumulative distribution function of X is $B(x)$. Let $x_T = [x_1, x_2, ..., x_{P-1}]$ denotes the (P-1)-dimension demotion-thresholds-vector for the system with P priorities where $x_1 < x_2 < ... < x_{P-1}$. For convenience, we denote the minimal coflow size and maximal coflow size of X as x_0 and x_P respectively.

Design goal: Set a proper demotion-thresholds-vector x_T to minimize the average coflow queueing delay.

Improvement ratio function: The average coflow queueing delay W for a system adopting FIFO order is given by [12],

$$W = \frac{W_0}{1-\rho}, \tag{1}$$

where $W_0 = \frac{\lambda E[X^2]}{2}$ and $E[X^2]$ is the second moment of the coflow size distribution. The traffic load ρ in (1) is defined by the product of the coflow arrival rate and the mean coflow size, $\lambda E[X]$. We assume $\rho < 1$ hence $\lambda E[X] < 1$.

For the D-CLAS system with two priority classes, to expose the average queueing delay improvements of the prioritization system over the FIFO system. We purposely re-transform the average queueing delay W_T of the system with two priorities as,

$$W_T = W \frac{1-\alpha_1 \lambda E[X]}{1-\alpha_1 \lambda E[X_1]}, \tag{2}$$

where $\alpha_1 = \int_{x_0}^{x_1} b(x)dx$, denoting the percentage of coflows with sizes in $[x_0, x_1]$ and $E[X_1] = \int_{x_0}^{x_1} xb(x)dx/\alpha_1$, representing the average size of coflows with sizes in $[x_0, x_1]$. From (2), we define the *improvement ratio function* $R(x_1)$ as,

$$R(x_1) = \frac{1-\alpha_1 \lambda E[X]}{1-\alpha_1 \lambda E[X_1]}. \tag{3}$$

Due to the fact that $E[X] \geq E[X_1]$ and $\lambda E[X] < 1$, we have,

$$0 < R(x_1) \leq 1. \tag{4}$$

Hence, the *improvement ratio function* quantifies the average queueing delay improvements of the two-priority D-CLAS system over a FIFO system by a function of the demotion threshold x_1. Furthermore, the *improvement ratio function* is correlated with coflow size distribution X and arrival rate λ. Thus, under particular statistic characteristics of coflows, the minimum x_1^* of $R(x_1)$ is the optimal demotion threshold which maximizes the average queueing delay improvements of D-CLAS systems over a FIFO system.

Similarly, the coflow queueing delay W_M for systems with multiple priorities ($P \geq 2$) can be re-transformed as,

$$W_M = W \cdot R_G(x_T), \tag{5}$$

where the generalized *improvement ratio function* $R_G(x_T)$ is given by,

$$R_G(x_T) = \sum_{p=1}^{P} \frac{\alpha_i(1-\rho)}{(1-\lambda \sum_{i=1}^{p-1} \alpha_i E[X_i])(1-\lambda \sum_{i=1}^{p} \alpha_i E[X_i])}, \tag{6}$$

where $\alpha_p = \int_{x_{p-1}}^{x_p} b(x)dx$ which denotes the percentage of coflows with sizes in $[x_{p-1}, x_p]$ and $E[X_p] = \int_{x_{p-1}}^{x_p} xb(x)dx/\alpha_p$, denoting the average size of coflows with sizes in $[x_{p-1}, x_p]$.

When infinite number of priority classes (continuous priorities) are adopted by a system, the lower bound R_L of R_G is achieved by letting $P \to \infty$ as,

$$R_L = \int_{x_0}^{x_P} \frac{(1-\rho)b(x)}{[1-\lambda \int_0^x yb(y)dy]^2} dx. \tag{7}$$

Minimizing the $R_G(x_T)$ under the constraint $x_1 < x_2 < ... < x_{P-1}$ is a complex optimization problem. Solving this problem remains an open problem [9]. In order to transform this unsolvable problem to a tractable problem, we let the demotion-thresholds-vector x_T owns the property that

$x_{p+1} = \beta x_p$ for $p \geq 1$ where β is a parameter typically valued by an integer no large than 10. Thus, we transform the unsolvable problem to the tractable problem minimizing $R_G(x_1, \beta)$ without constraints. For given β, we focus on finding an optimal demotion-threshold x_1^* which minimizes $R_G(x_1, \beta)$. Then the optimal demotion-threshold-vector x_T^* is given by the assumption $x_{p+1}^* = \beta x_p^*$ for $p \geq 1$.

IV. Property and Algorithm

In this section, we prove the *three-stage property* of the *improvement ratio function* in Section IV-A. Exploiting the property, we design the DHS algorithm to fast locate the optimal demotion-thresholds-vector within a narrow searching space in Section IV-B.

A. Three-stage Property

Normally, in order to search the minimum x_1^* of the *improvement ratio function* $R(x_1)$ or $R_G(x_1, \beta)$, x_1 need to traverse the whole range of possible coflow sizes from x_0 to x_P due to the lack of knowledge about where the minimum located. The searching space from x_0 to x_P is huge for real coflow size distributions (about 10^6 MB). To reduce the searching space, we utilize the *three-stage property* which reveals the valley-like shape of the *improvement ratio function* $R(x_1)$.

Theorem 1 (*Three-Stage Property*): For a system with two priorities, given λ, the shape of $R(x_1)$ owns three-stage property. **(1). Down-hill stage:** $R(x_1)$ decreases monotonously from the point $(x_0, 1)$ to (x_1^*, R_m) where R_m is the minimal value of $R(x_1)$; **(2). Minimal stage:** $R(x_1)$ takes its minimum (x_1^*, R_m) when $x_1 = x_1^*$; **(3). Uphill stage:** $R(x_1)$ increases monotonously from (x_1^*, R_m) to $(x_2, 1)$.

The proof of Theorem 1 is postponed to Appendix.

B. Down-hill Searching Algorithm

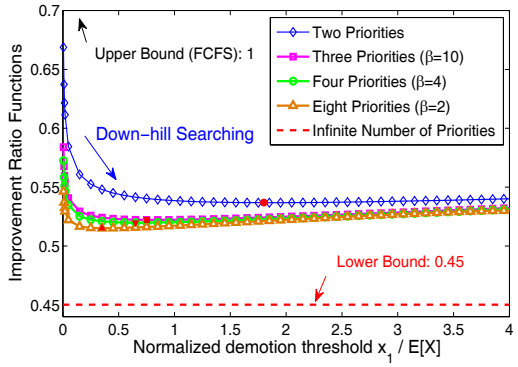
We utilize Theorem 1 to reduce the searching space by designing the *down-hill searching* (DHS) algorithm. Given β, the DHS (Algorithm 1) continuously estimates values of *improvements ratio function* $R_G(x_1, \beta)$ by increasing the demotion threshold x_1 (recorded by x_L, x_R in Algorithm 1) gradually from x_0 with a predefined step length α. At their down-hill stage, the values of $R_G(x_1, \beta)$ decrease continuously (the condition $R_G(x_L, \beta) > R_G(x_R, \beta)$ holds in Algorithm 1). Once the values of $R_G(x_1, \beta)$ begin to increase, the $R_G(x_1, \beta)$ enter their uphill stage and the DHS stops searching and sets x_1^* as the x_1 at last step (recorded by x_L in Algorithm 1).

Then $x_2^*, x_3^*, ..., x_{P-1}^*$ are given by the assumption $x_{p+1}^* = \beta x_p^*$ for $p \geq 1$. Through the DHS algorithm, the increasing x_1 stops in time before it traverses the whole range from x_0 to x_P.

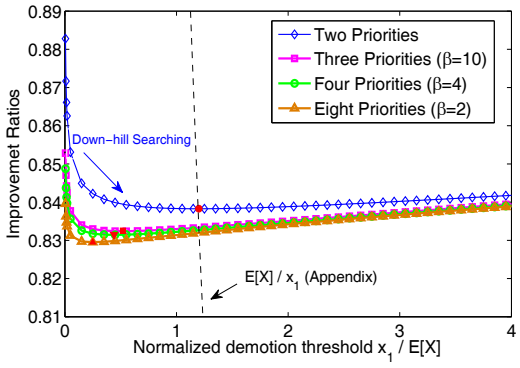
A case study of DHS would be shown in Section V-A. where the minimum x_1^* of $R(x_1)$ or $R_G(x_1, \beta)$ occur around $x_1 = E[X]$ (about 1.25×10^4 MB). Since the gap between minimal coflow size and maximal coflow size is about 10^6 MB, the searching space of the DHS algorithm is around $\frac{1.25 \times 10^4}{10^6}$ =1.25% of total searching space hence 80× faster than a general exhausting searching algorithm.

Algorithm 1 Down-hill Searching (DHS)

Inputs:
The number of priorities, P
The traffic load, ρ; The minimal coflow size, x_0
The CDF of the coflow size distribution, $B(x)$
The parameter β, The down-hill step length, α
Outputs:
An optimal demotion-thresholds vector, $x_1^*, x_2^*, ..., x_{P-1}^*$
$x_L \Leftarrow x_0$;$x_R \Leftarrow x_0 + \alpha$;
while $R_G(x_L, \beta) > R_G(x_R, \beta)$ **do**
  $x_L \Leftarrow x_L + \alpha$; $x_R \Leftarrow x_R + \alpha$;
end while
$x_1^* \Leftarrow x_L$;
if $P > 2$ **then**
  for $p = 2$ to $P - 1$ **do**
    $x_p^* = \beta x_{p-1}^*$;
  end for
end if



(a) $\rho = 0.5$

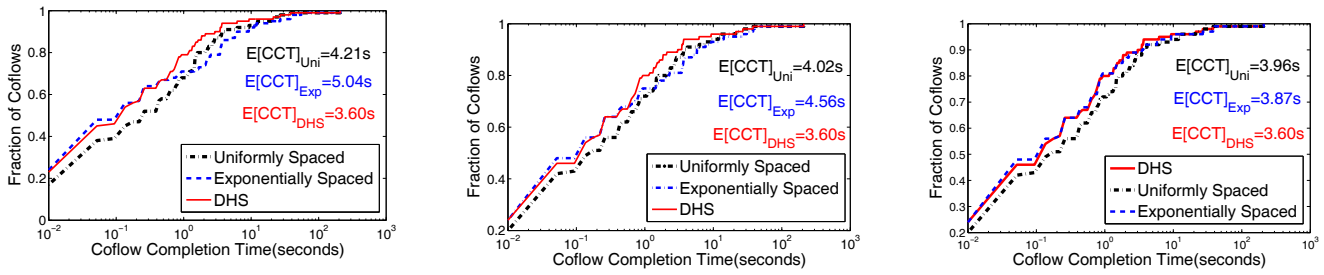


(b) $\rho = 0.177$

Fig. 2. Down-hill searching the optimal demotion threshold based on a real coflow size distribution.

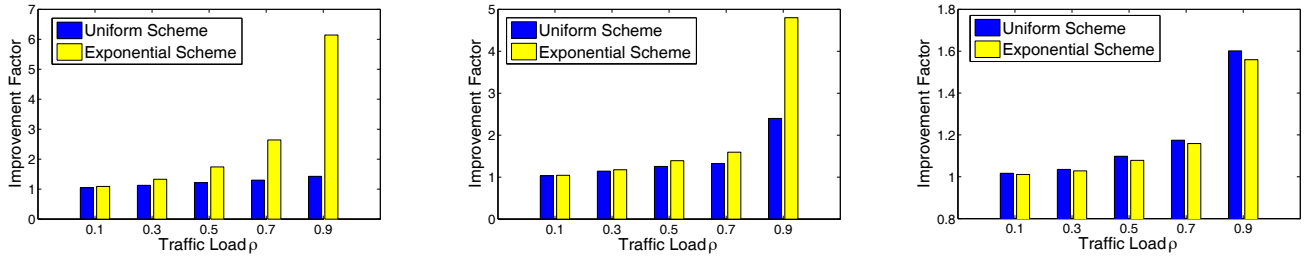
V. Experiment Results

In this section, we first apply the DHS algorithm to a real DCN coflow trace [10] in Section V-A. In Section V-



(a) P=2 (b) P=3 (c) P=4

Fig. 3. CCT distributions: uniform thresholds prefer large sized coflows and exponential thresholds prefer small sized coflows; DHS benefits for both.



(a) Two priorities, P=2 (b) Four Priorities, P=4 (c) Eight Priorities, P=8

Fig. 4. Impacts of traffic load ρ and P: the improvements of DHS increases with increasing traffic load and decreases with increasing number of priorities.

B, we introduce our Matlab-based flow-level simulator, which evaluates the average CCT improvements of DHS over Aalo through replaying the real DCN trace in a D-CLAS manner. Simulation results in Section V-C show that the improvements of DHS over Aalo is 1.08 $\times$ to 6.20 $\times$.

A. Demotion Thresholds Settings

Optimal-demotion-thresholds: In this subpart, we apply the DHS algorithm to the real DCN coflow size distribution extracted from the real DCN coflow trace [10]. The coflow size distribution ranges from 1 (MB) to 1.02×10^6 (MB). The mean coflow size is 1.25×10^4 (MB). Taking the real coflow size distribution as its input, how the DHS algorithm works is illustrated in Fig. 2 where the curves of *improvement ratios function* under $\rho = 0.5$ and $\rho = 0.177$ (estimated from the real DCN trace) is plotted respectively. Those red solid notations on the curves annotate the optimal normalized demotion threshold $x_1^*/E[X]$ which is captured by the DHS algorithm. For clearness, we only depict a narrow range ([0,4]) of possible normalized threshold $x_1/E[X]$, $R(x_1)$ and $R_G(x_1, \beta)$ increase monotonously with $x_1/E[X]$ on the range ([4,81.6]) missing in Fig. 2, which corresponds to Theorem 1.

Demotion thresholds in Aalo: Aalo utilizes two simple demotion thresholds settings. First, the exponentially-spaced thresholds, where the p-th threshold $x_p = x_0 \times \beta^p$, $p = 1, 2, ..., (P-1)$. Typically, Aalo adopts $x_0 = 10$, $\beta = 10$. Second, the uniformly-spaced thresholds, where the spaces between $(P-1)$ demotion thresholds are set as $\frac{1}{P} \times x_P$.

B. Simulation Settings

(1). **Dataset:** Our dataset is based on a real trace collected by [10] on a 3000-machine, 150-racks Facebook cluster, which contains all flows belonging to 500 coflows.

(2). **Network Model:** The network model adopted is the DC fabric (150 by 150) [4], [8] as Fig. 1(a) shows. The capacity of each ingress and egress port is 800Mbps.

(3). **Arrival Process:** The coflows arrive in a Poisson process with parameter λ. The traffic load ρ is defined as $\rho = \frac{\lambda \times AvgCoflowSize}{NetworkThroughput}$. The *NetworkThroughput* in our simulation is $150 \times 800Mbps = 120$Gbps.

Metrics The primary metric adopted in our evaluations is the improvements of average CCT and the improvement factor is defined as,

$$Improvement\ Factor = \frac{Compared\ Duration}{DHS\ Duration}. \quad (8)$$

C. Simulation Results

CCT distributions under different schemes: As different CCT distributions in Fig. 3(a)$\sim$(c) show, the uniform scheme harms CCTs of those smaller sized coflows while the exponential scheme hurts CCTs of those larger sized coflows. Differently, DHS achieves small CCTs for both smaller sized coflows and larger sized coflows. In terms of improvement factor metric, DHS is 1.17$\times$, 1.12$\times$ and 1.10$\times$ better than uniform thresholds for $P = 2$, $P = 3$ and $P = 4$ respectively. Besides, DHS outperforms exponential thresholds by 1.40$\times$, 1.27$\times$ and 1.08$\times$ for $P = 2$, $P = 3$ and $P = 4$ respectively.

The impacts: We further investigate the impacts of ρ and P on the CCT performance of DHS as shown in Fig. 4. There are two lessons we learned from Fig. 4,

- Given the number of priorities P, the improvements of optimal-demotion-thresholds over simple thresholds schemes tends to increase as traffic load ρ increases.
- For fixed traffic load ρ, the improvements of optimal-demotion-thresholds over simple thresholds schemes tends to decrease as the number of priorities P increases.

Takeaways: *On one hand, when the DCN is heavy-loaded, the optimal demotion-thresholds-vector should be exploited which brings much performance gain (up to 6.20×) compare to current simple thresholds settings. On another hand, when the DCN is light-loaded, current D-CLAS system could decrease its average CCT by increasing the number of priorities. However, the DHS always guarantees a smaller average CCT regardless of the network traffic load.*

VI. Related Work

There is a large spectrum of related work about flow scheduling in DCNs. We classify previous flow scheduling work in DCNs into three categories:

Per-flow scheduling: There are many existing work on minimizing the average flow completion time (FCT) such as pFabric [8] and PIAS [9]. However, they do not take into consideration the semantics of flows belonging to a coflow hence fall short in minimizing average CCT.

Information-aware coflow-aware scheduling: FIFO-based Ochestra [2] is the very first coflow scheduling design with inferior performance. After Orchestra, Varys [4], D-CAS [5], [11] and RAPIER [6] improve average CCT by heuristics like smallest-effective-bottleneck-first (SEBF) or smallest-remaining-size-first (SRSF) with prior coflow information. However the coflow characteristics are unknown a priori in most cases [10]. Consequently, they are hard to implement.

Information-agnostic coflow-aware scheduling: Without prior knowledge of coflows, FIFO-LM (Limit Multiplexing) proposal Barrats [3] compromises on performance to avoid head-of-line blocking. State-of-art proposal Aalo improves the average CCT performance by invoking D-CLAS [10].

VII. Conclusions

Motivated by the insight that crude settings of demotion thresholds which ignore the statistic characteristics of coflows can severely penalize average CCTs of any D-CLAS system, we model the D-CLAS system by an M/G/1 priority queue and design the DHS algorithm searching a set of statistic-related optimal demotion thresholds for current D-CLAS system. Through trace-driven simulations, DHS improves the average CCT of the state-of-art D-CLAS system Aalo by up to 6.20×.

References

[1] M. Chowdhury and I. Stoica, "Coflow: A networking abstraction for cluster applications," in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks (HotNets)*. ACM, 2012, pp. 31-36.

[2] M. Chowdhury, M. Zaharia, J. Ma, M. I. Jordan, and I. Stoica, "Managing data transfers in computer clusters with orchestra." in *Proc. ACM SIGCOMM*, 2011, pp. 98-109.

[3] F. R. Dogar, T. Karagiannis, H. Ballani, and A. Rowstron, "Decentralized task-aware scheduling for data center networks," in *Proc. ACM SIGCOMM*, 2014, pp. 431-442.

[4] M. Chowdhury, Y. Zhong, and I. Stoica, "Efficient coflow scheduling with varys," in *Proc. ACM SIGCOMM*, 2014, pp. 443-454.

[5] S. Luo, H. Yu, Y. Zhao, B. Wu, S. Wang, and L. Li, "Minimizing average coflow completion time with decentralized scheduling." In *Proc. IEEE International Conference on Communications (ICC)*, 2015, pp. 307-312.

[6] Y. Zhao, K. Chen, W. Bai, M. Yu, C. Tian, Y. Geng, Y. Zhang, D. Li, and S. Wang, "Rapier: Integrating routing and scheduling for coflow-aware data center networks," in *Proc. IEEE INFOCOM*, 2015, pp. 424-432.

[7] S. Luo and H. Yu and L. Li, "Decentralized Deadline-Aware Coflow Scheduling for Datacenter Networks," In *Proc. IEEE International Conference on Communications (ICC)*, 2016.

[8] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker, "pfabric: Minimal near-optimal datacenter transport," in *Proc. ACM SIGCOMM*, 2013, pp. 435-446.

[9] W. Bai, L. Chen, K. Chen, D. Han, C. Tian, and H. Wang, "Information-agnostic flow scheduling for commodity data centers," in *NSDI*, 2015.

[10] M. Chowdhury, and I. Stoica, "Efficient Coflow Scheduling Without Prior Knowledge," in *Proc. ACM SIGCOMM*, 2015, pp. 393-406.

[11] S. Luo and H. Yu and Y. Zhao and S. Wang and S. Yu and L. Li,"Towards Practical and Near-optimal Coflow Scheduling for Data Center Networks," *Parallel and Distributed Systems, IEEE Transactions on*, 2016.

[12] Mor Harchol-Balter, *Performance Modeling and Design of Computer Systems: Queueing Theory in Action*. Cambridge University Press, 2013.

[13] Richard L. Burden and J. Douglas Faires, *Numerical Analysis*. Brooks Cole Press, 9th edition, 2010.

Appendix

Proof of Theorem 1: For a system with two priorities, given λ, we first denote the numerator and denominator of (3) as $f(x_1)$ and $g(x_1)$ respectively. Through taking derivatives to both of them, one can obtain,

$$\frac{f'(x_1)}{g'(x_1)} = \frac{E[X]}{x_1}. \tag{9}$$

Since the equality of (4) holds only when either $x_1 = x_0$ (means $\alpha_1 = 0$) or $x_1 = x_2$ (means $E[X_1] = E[X]$), we have $R(x_0) = R(x_2) = 1$. In addition, the inequality $x_0 < E[X] < x_2$ holds. Hereby, $R(x_0) < \frac{E[X]}{x_0}$ and $R(x_2) > \frac{E[X]}{x_2}$. Thus, $(\frac{E[X]}{x_0} - R(x_0))(\frac{E[X]}{x_2} - R(x_2)) < 0$ holds. According to *intermediate value theorem* [13], $\frac{E[X]}{x_1} - R(x_1)$ intersects with x_1-axis at least once on the range (x_0, x_2) hence $\frac{E[X]}{x_1}$ intersects $R(x_1)$ at least once on the range (x_0, x_2).

Next, we assume the first intersection between $R(x_1)$ and $\frac{E[X]}{x_1}$ happens when $x_1 = x_f$. Then, we can decompose the trend of $R(x_1)$ into three stages as follows. **(1). Down-hill stage:** for $x_0 \leq x_1 < x_f$, since $R(x_0) < \frac{E[X]}{x_0}$ and $R(x_1)$ has not yet intersected with $\frac{E[X]}{x_1}$, $R(x_1) < \frac{E[X]}{x_1}$ holds. Furthermore, due to (9), we have $R(x_1) < \frac{f'(x_1)}{g'(x_1)}$ hence $\frac{f(x_1)}{g(x_1)} < \frac{f'(x_1)}{g'(x_1)}$, which results in $\frac{f'(x_1)g(x_1) - f(x_1)g'(x_1)}{g^2(x_1)} < 0$, namely, $R'(x_1) < 0$. In summary, $R(x_1)$ decreases monotonously on the range $[x_0, x_f)$; **(2). Minimal stage:** for $x_1 = x_f$, $R(x_1)$ intersects with $\frac{E[X]}{x_1}$ hence $R(x_f) = \frac{E[X]}{x_f}$, which leads to $R'(x_f) = 0$ by similar derivations in Down-hill stage. Thus, $R(x_1)$ intersects with $\frac{E[X]}{x_1}$ at its critical point (where $R'(x_f) = 0$); **(3). Uphill stage:** for $x_f < x_1 \leq x_2$, since $\frac{E[X]}{x_1}$ decreases monotonously, $R(x_1) > \frac{E[X]}{x_1}$ holds hence $R'(x_1) > 0$ is true following similar derivations in Down-hill stage. Thus $R(x_1)$ increases monotonously on the range $(x_f, x_2]$. Hereby, x_f is the minimum of $R(x_1)$ hence $x_f = x_1^*$.