DA&FD–Deadline-Aware and Flow Duration-Based Rate Control for Mixed Flows in DCNs

Han Zhang, *Member, IEEE*, Haijun Geng, Yahui Li, *Member, IEEE*, Xia Yin, *Senior Member, IEEE,* Xingang Shi, *Member, IEEE,* Zhiliang Wang, Qianhong Wu, and Jianwei Liu

***Abstract*— Data center has become an important facility for hosting various applications. For data center networks, deadline missing rate and average flow completion time are two main metrics for the performance of applications. In this paper, we find deadline-aware methods can only reduce the percentage of flows missing deadline, while flowsize-aware and information-cumulative methods can only optimize the average flow completion time. However, traffic in data center is the mixture of various flows and focusing on the single goal is not enough. We advocate to incorporate deadline and flow duration time into flow rate control. Then we design DA&FD (Deadline-Aware and Flow Duration) based rate control mechanism and analyze its performance in theory. At last, we evaluate DA&FD under different topologies, real world traffic and load scenarios, both by simulation and in real testbed. Our results show that DA&FD performs close to D^2TCP and about 15%, 25%, 30%, 35% better than Ameon, L^2DCT, Karuna, DCTCP on deadline missing rate. For average FCT, the performance of DA&FD is similar to L^2DCT and compared with Ameon, D^2TCP, Karuna, DCTCP, DA&FD can reduce average FCT by 10%, 15%, 20%, 25%.**

***Index Terms*— Data center, rate control, deadline, average flow completion time.**

I. Introduction

DATA center has become an important infrastructure for applications. Especially, with the existing of cloud computing, tenants can rent data center to satisfy their demands. There are frequent interactions between the components of distributed applications. Many technologies have been proposed to improve the QoS of network. MPTCP [8], [14], [27] extends standard TCP, so that a single connection between two hosts can transmit segments over multiple paths simultaneously. It can provide redundancy against link failures and reduce the load on congested paths. QoS can also benefit from advantages of SDN [11], [19], [34], which adopts separation of data plane and control plane for networks. Recently, mixed flow control technology has attracted much attention, since it can reduce deadline missing rate and average flow completion time (FCT) simultaneously, thus affecting user experience and revenue [36].

Flows can be divided into two categories by whether explicit deadline can be known in advance. **Type 1: flows with deadlines.** Applications such as web search, recommendation system will generate these flows. Users should set deadline to applications and missing deadline will lead to performance loss. For these applications, the primary performance metric is to reduce the percentage of flows missing deadline. **Type 2: flows without deadlines.** Most of these flows come from data transfer applications, e.g, VM migration, HTTP chunk transfer, database access. These flows require small average flow completion time (AFCT). The traditional transport mechanism TCP falls short on two sides. Firstly, TCP can cause buffers overflow and this will lead to high latency, thus user experience. Secondly, the behavior of TCP congestion aviodance will cause link utilization low. Due to the faultiness of TCP, DCTCP [4] is proposed. DCTCP uses ECN marking mechanism [4] which is supported by most modern switches to assist senders adjust congestion window. Instead of reducing congestion window by half, DCTCP modifies window with congestion level. DCTCP can keep switch queue shallow and maintain link utilization high. However, DCTCP is a deadline-agnostic and fair-share method [4]. In reality, flows have different level of emergency and fair allocation is not a good choice since this can lead delay-sensitive flows large latency [29], [30]. Still now, three types of methods have been raised to make up the deficiencies:

Deadline-aware methods: They can be used to reduce the percentage of flows missing deadline. For type 1 flows, users can set explicit deadline from user space and deadline-aware methods (D^3 [30], D^2TCP [29], LPD [36], etc.) incorporate deadline into rate control. As a result, urgent flows will get larger bandwidth than tardy flows.

Flowsize-aware methods: In DCNs, the urgent flows are always short [13] and size of some type 2 flows (VM migration, data backup, etc.) can be known in advance. The flowsize-aware methods (PDQ [13], L^2DCT [23], pFabric [6], etc.) that can be applied to type 2 flows whose length can be known beforehand. They prefer short flows and realize Least Attained Service (LAS) strategy. Minimizing average

Manuscript received February 28, 2019; revised September 1, 2019 and October 20, 2019; accepted October 21, 2019; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor I.-H. Hou. Date of publication November 22, 2019; date of current version December 17, 2019. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFB1800401, in part by the Jiangsu Key Laboratory of Big Data Security and Intelligent Processing, NJUPT, under Grant BDSIP1912, and in part by the Natural Science Foundation of China under Project 61702315. *(Corresponding author: Xingang Shi.)*

H. Zhang, Q. Wu, and J. Liu are with the School of Cyber Science and Technology, Beihang University, Beijing 100191, China.

H. Geng is with the School of Software Engineering, Shanxi University, Taiyuan 030006, China.

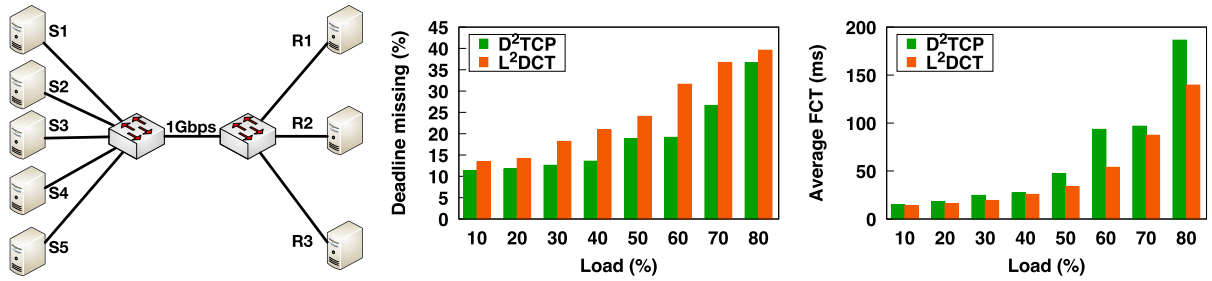
Y. Li and X. Yin are with the Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.

X. Shi and Z. Wang are with the Beijing National Research Center for Information Science and Technology and the Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China (e-mail: shixg@cernet.edu.cn).

Digital Object Identifier 10.1109/TNET.2019.2951925

1063-6692 © 2019 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) Dumbbell topology (b) Flows missing deadline (type 1 flows) (c) Average FCT (type 2 flows)

Fig. 1. [Simulation] Performance of deadline-aware and flowsize-aware methods (30% flows have deadline).

flow completion time is the optimization goal for these methods.

Information-cumulative methods: Neither deadline nor flow size of some type 2 flows (database access, etc.) can be learned by applications before sending. To optimize the performance of such applications, information-cumulative methods (FDRC [37], PIAS [7], Aemon [28], etc.) accumulate data volume to approximate flow size. They nearly realize LAS and average flow completion time decreases.

From the technical perspective, the scheduling methods (PDQ, D^3, etc.) try to set and enforce the priority at the ends or switches, while the rate control approaches (DCTCP, L^2DCT, LPD, etc.) have no explicit priority settings. Although they have good performance under certain scenarios, in reality, no rate control method can reduce the percentage of flows missing deadline and the average flow completion time simultaneously when all kinds of flows coexist in the data center. We find deadline-aware methods can significantly reduce the percentage of flows missing deadline but it will cause a large average FCT, while flowsize-aware and information-cumulative methods can diminish the average FCT but with a great deadline missing rate. In this paper, we advocate to incorporate both deadline and flow duration time into flow rate control and propose DA&FD (Deadline-Aware and Flow Duration) based rate control. For a flow with deadline, its deadline is used when updating its congestion window size. Otherwise, its flow duration is used. DA&FD puts duration and deadline into one framework, so that latency of all flows can be optimized with the same threshold (T). Besides, DA&FD is a rate control method and can work in a fully distributed manner. DA&FD extension can provide more QoS options to users by adjusting the importance ratio factor.

Thus the contributions we make in this paper are summarized as follows:

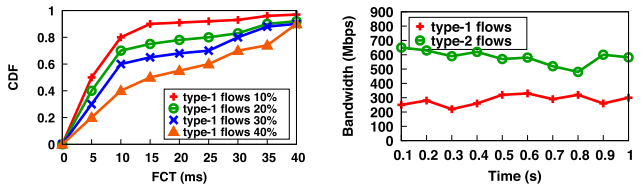
- We find that modern rate control methods can't meet the demands of applications and propose to incorporate deadline and flow duration time into congestion window computation.
- We design and analyze DA&FD which can reduce the percentage of flows missing deadline and average flow completion time simultaneously.
- We evaluate DA&FD in ns-3 [2], then use the real data center network traffic to compare DA&FD with D^2TCP, L^2DCT, DCTCP, Ameon and Karuna. Large scale simulations show that DA&FD performs close to D^2TCP and about 15%, 25%, 30%, 35% better than Ameon, L^2DCT, Karuna and DCTCP on deadline missing rate. For average FCT, the performance of DA&FD is similar to L^2DCT and compared with Ameon, D^2TCP, Karuna, DCTCP, DA&FD can reduce average FCT by 10%, 15%, 20%, 25% on average.
- We evaluate DA&FD into Linux kernel and use small data center testbed to test the performance of DA&FD. We find the testbed results match well with simulations.

II. Motivation and Related Work

For applications (web search, recommendation system, etc.) with deadlines, the performance metric is to reduce the percentage of flows missing deadline. Other applications (VM migration, data backup, database access and HTTP chunked transfer, etc.) expect short average flow completion time. Nowadays, a lot of methods [23], [29], [36] focus on only single performance metric. All applications share the data center network and only considering deadline or average flow completion time will lead to poor performance.

We construct a simple experiment to show that deadline-aware, flowsize-aware and information-cumulative methods can hurt the performance of applications in the mixed scenarios with ns-3 simulator [2]. Fig. 1(a) shows the topology. In the topology, 5 senders and 3 receivers connect the switch with 1Gbps link. S1-S4 run web search workload [4], [6] and S5 sends long-live background flow. The bottleneck link capacity between the two switches is also 1Gbps. We set 30% of the flows with uniform distribution deadline between 30ms and 50ms to simulate type 1 flows. The other 70% flows don't have deadline and they simulate the type 2 flows. Fig. 1(b) shows that D^2TCP has about 30% less flows missing deadline than L^2DCT. Fig. 1(c) shows that L^2DCT performs about 20% better than D^2TCP on average flow completion time. D^2TCP can only reduce the deadline missing ratio while L^2DCT focuses on average FCT. The reason for this is that D^2TCP (deadline-aware method) only considers deadline and congestion level, while flowsize-aware and information-cumulative methods prioritize short flows, but ignoring deadline flows. However, traffic in data center is the mixture of deadline and non-deadline flows, so new strategies are needed for the mixed traffic scenario to meet the demands of all flows. Moreover, D^2TCP and L^2DCT

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) FCT distribution (type 2 flows) (b) Bandwidth (type 1 flows 30%)

Fig. 2. [Simulation] D^{2}TCP hurts type 2 flows and the hybrid method has low link utilization.

can be used in parallel and we name this method as hybrid, where type 1 flows will use D^{2}TCP and L^{2}DCT can be used for type 2 flows. The simple combination of D^{2}TCP and L^{2}DCT is also not a good choice, since type 1 flows complete quickly with small FCT, which is unnecessary for meeting their deadlines, at the cost of higher latency for type 2 flows. To prove this, in the simple dumbbell topology, S1-S4 run data mining workload without deadline and S5 run type 1 flows whose sizes are 5KB, and their deadlines are drawn from an exponential distribution with mean 15ms. We also vary the percentage of flows with deadline, and collect the FCT of type 2 flows. Fig. 2(a) shows the results. We can see that when there are more type 1 flows, the number of type 2 flows with small average FCT (< 20ms) drops heavily. This will hurt applications' performance since type 2 flows (hadoop, VM migration, etc) often call for small average FCT in data center [9], [13], [23]. Fig. 2(b) demonstrates the bandwidth of all type 1 and type 2 flows when there are 30% flows with deadline. We can see that the average link utilization of hybrid method is only about 90%. The competition of the two rate control methods could induce higher fluctuations, and lowers the overall bandwidth utilization with naive hybrid method.

Still now, many strategies have emerged to optimize the transfer latency of applications [24]. DCTCP [4] tries to reduce queue latency. When packets arrive at the switch, if queue length is larger than the marking threshold K, the new arrived packets will be marked CE. Senders compute congestion degree every RTT. When congestion occurs, congestion window will be computed as $w = w * (1 - \alpha/2)$, where α is the congestion degree. DCTCP is a deadline-agnostic method and it is fair sharing. However, in data center, applications call for different latency and fair sharing can lead to performance loss [30].

As the deadline of some flows can be known in advance, deadline-aware methods try to set explicit deadline constrains to flows to differentiate them. D3 [30] allocates bandwidth to far-deadline requests arriving slightly ahead of near-deadline requests. D3 has significant performance and practical shortcomings. D3 can not coexist with TCP and it works bad because of its greedy and first-come-first-service policy. D^{2}TCP [29] modifies DCTCP. When packets are marked, congestion window is computed as $w = w * (1 - \alpha^d)$, where α is the congestion level and $d = TC/D$ is deadline factor. D^{2}TCP can coexist with TCP and does not need to change switch hardware. However, D^{2}TCP's performance declines when load is heavy. LPD [36] finds defects of D^{2}TCP and advocates that when designing congestion control algorithm, the principle –more load, more differentiation should be followed. LPD does not need to change switch hardware and can keep low latency.

The deadline-aware methods take deadline and congestion level into rate allocation. However, deadline of some flows can not be known prior. In this case, some methods try to reduce average flow completion time or tail to optimize the transfer latency.

Applications such as VM migration, data backup will generate flows with prior-known length, so flowsize-aware methods take advantage of this and try to reduce the average flow completion time (AFCT). L^{2}DCT [23] alters both the additive increase and the multiplicative decrease part. In L^{2}DCT, short flows have larger increase and smaller decrease values than long flows, so that average flow completion time decreases. PDQ [13] is a schedule method and it works in distributed fashion. PDQ can emulate different scheduling policies like Earliest Deadline First (EDF) or Shortest Job First (SJF). PDQ changes switch hardware and can not co-exist with TCP. FastPass [26] advocates that each sender should delegate control to a centralized arbiter of when each packet should be transmitted and what path it should follow. FastPass tries to keep the switch queue shallow. As a result, the transfer latency of flows reduces. In pFabric [6], datacenter transport should decouple flow scheduling from rate control. For flow scheduling, it uses Shortest Remaining Processing Time (SRPT) policy. Flows start at line rate and throttle back only under high and persistent packet loss. Combining these two characters, pFabric achieve near-optimal performance for flow completion time. PASE [22] claims protocols (DCTCP, PDQ, pFabric, etc) are not competitors, they are complementary. PASE does not change network elements and its performance is better than those state-of-art protocols [22]. Repflow [31] reproduces short flows when transferring data and with the help of multipath, completion time of short flows reduces. Detail [33] emphasizes on tail completion time of flows.

Neither deadline nor flowsize of some applications (database access, etc) can be known in advance. Infromation-cumulative methods collect flow information (volume of data, duration time, etc) to change the priority of flows. FDRC [37] and Aemon [28] incorporate flow duration time into rate control, flows last long will have low priority, so that average flow completion time reduce. PIAS [7] leverages multiple priority queues available in commodity switches to implement a Multiple Level Feedback Queue (MLFQ), in which a PIAS flow gradually demotes from higher priority queues to lower based on the bytes it has sent. In Karuna [9], the key insight is that deadline flows should meet their deadlines while minimally impacting the FCT of non-deadline flows. Although Karuna can reduce the percentage of flows missing deadline and average FCT simultaneously, it will lead link utilization low. There are also some task level transfer methods [15], [16], [32]. [10], [12], [17] optimize the transfer latency by queue management. Homa [21] is a receiver-driven low-latency transport protocol using network priorities. [18] proposes Explicit Priority Notification (EPN), which emulates fine-grained priorities (i.e., desired priorities or DP) using only two switch priority queues. HybridPass [25] uses a loosely synchronized time-triggered

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.

TABLE I
COMPARISON AMONG DIFFERENT METHODS

Methods	Objectives		Information Prior		Modifications			Technology
	FCT	Deadline	Flow Size	Deadline	Switch	Kernel	Application	
DCTCP[4]	mean	×	×	×	×	√	×	Implicit
D^{2}TCP[30]	×	√	√	√	×	√	√	Implicit
D3[31]	×	√	√	√	√	√	√	Explicit
LPD[37]	×	√	√	√	×	√	√	Implicit
L^{2}DCT[24]	mean	×	√	×	×	√	×	Implicit
PDQ[14]	mean&tail	×	√	×	√	√	√	Explicit
pFabric[7]	mean&tail	×	√	×	√	√	√	Implicit
Repflow[32]	mean&tail	×	×	×	×	×	√	Implicit
Detail[34]	tail	×	√	×	√	√	√	Implicit
PIAS[8]	mean&tail	×	×	×	√	√	×	Implicit
PASE[23]	mean&tail	×	√	×	√	√	√	Implicit
FastPass[27]	mean&tail	×	√	×	√	√	×	Explicit
FDRC[38]	mean	×	×	×	×	√	×	Implicit
Karuna[10]	mean	√	×	×	×	√	×	Implicit
Aemon[29]	mean	√	×	×	×	√	×	Implicit
DA&FD	mean	√	×	×	×	√	×	Implicit

manner to allocate the network bandwidth for latency-sensitive and throughput intensive flows from a global perspective.

According to whether rate can be precisely determined, the rate control methods can be divided into implicit and explicit ones, where the explicit technique can compute and assign rates from the network to each flow and the implicit technique can't know the right flow rates. TABLE I shows the comparison of the most related works from optimization objectives, information should be known prior and modifications to hardware. We can see that most of the methods need to know some flow information (flow size, deadline, remaining flow size, remaining time, etc.) beforehand or change network hardware, which limits their usage and makes them hard to deploy. We also include DA&FD in the table, so that the readers would have an intuitive understanding for DA&FD.

III. DA&FD PROTOCOL

The data center traffic is the mixture of deadline and non-deadline flows. They come from different applications and have different optimization goals. Due to the low performance of existing methods for the mixed flows in data center network, the high efficiency and deployable methods are needed to make flows satisfy their demands. In this section, we first introduce the optimization goal. Then we design DA&FD (Deadline-Aware and Flow Duration) based rate control mechanism to minimize the percentage of flows missing deadline and average flow completion time together. At last, we analyze the performance of DA&FD in theory.

A. Optimization Goal

Flows with or without deadlines co-exist in data centers. Only trying to reduce the deadline missing rate may increase the flow completion time of type 2 flows, i.e., those without deadlines. Similarly, only focusing on optimizing the average FCT may cause many type 1 flows to miss their deadlines. Since both flows need QoS optimization, we study how to reduce the deadline missing rate and the average FCT simultaneously. Moreover, the major culprit of high latency is long queueing delay in switches, caused by the traffic characteristics of data centers. To obtain low flow latency in data center, queue length should be maintained in low and steady state at the same time.

B. DA&FD Protocol

To achieve the goals mentioned above, we adopt the implicit rate control instead of the explicit flow scheduling. The reason for this is that compared with the scheduling methods that work in the centralized manner, the TCP-based rate control technology has small latency overhead when deciding the transfer speed and can work in the distributed way. Based on this direction, we design DA&FD, which denotes Deadline-Aware and Flow Duration rate control mechanism for data center network. For flows with deadline, applications send deadline to transport layer through socket option. Otherwise, flow duration time is used. The congestion window updates with the computation result through the option and congestion level.

DA&FD has three components: **(1) Marking CE at swtiches.** When packets arrive the switch, they will be marked CE if queue length is larger than the threshold K, otherwise, packets will not be marked. The RED marking scheme implemented by most modern switches can be re-purposed for DA&FD. We just need to set both the low and high thresholds to K, and mark based on instantaneous, instead of average

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.

queue length. **(2) Receiver marks ECN.** DA&FD has the same way in CE replying to sender as DCTCP [4]. According to RFC 3168 [3], the receivers can send ECN-Echo in ACK if they receive CE. This can help the sender to notify congestion in the network. **(3) Rate control.** Every sender maintains an estimate of the fraction of packets that are marked CE, named α, which denotes congestion level and it updates once for every window of data (roughly one RTT) as follows:

$$\alpha = (1 - g) \times \alpha + g \times F \tag{1}$$

F is the fraction of packets that are marked according to the last window in a moving average estimation on α, where g $(0 < g < 1)$ is the weight given to new samples against the past. Actually, equation (1) can be regarded as the probability that the queue size is greater than K, and α presents the congestion degree of the flow. When α is close to 0, congestion is likely to be low, while if α is close to 1, a high level of congestion is more likely to happen. DA&FD uses penalty function f and α to adjust congestion window:

$$cwnd \leftarrow \begin{cases} cwnd + (1 - \dfrac{f}{2}) & \text{Without congestion} \\ cwnd \times (1 - \alpha \times \dfrac{f}{2}) & \text{With congestion} \end{cases} \tag{2}$$

Thus, when there is no congestion, DA&FD adds $(1 - \frac{f}{2})$ per RTT. When congestion occurs, it reduces by $(1 - \alpha \times \frac{f}{2})$ times. If α is near 0 (low congestion), the congestion window only slightly reduced. This means, DA&FD begins to reduce congestion window when queue exceeds K. This can guarantee the low delay. If α is larger (high congestion), DA&FD reduces congestion window by the flow congestion level α and the penalty function f. We now define the flow duration (FD) time of a flow i as follows:

$$FD_i = t - s_i \tag{3}$$

t is current time and s_i is the start time of flow i. With FD_i, we now define DA&FD:

$$f = \begin{cases} \dfrac{D_i}{T} & \text{Flow has deadline } D \\ \dfrac{FD_i}{T} & \text{Otherwise} \end{cases} \tag{4}$$

For flows have deadline, the penalty is computed with deadline D_i. For others, f uses flow duration (FD) time. In reality, flow duration time of long ones might be large, we cap the maximum value of f at 1.0 to prevent the long flows hungry. DA&FD doesn't need to know flow size. We can also extend DA&FD to make it provide more QoS options, which we named DA&FD-extension:

$$f = \begin{cases} \dfrac{D_i}{A \times T} & \text{Flow } i \text{ has deadline } D_i \\ \dfrac{FD_i}{T} & \text{Otherwise} \end{cases} \tag{5}$$

In DA&FD-extension, A is used to adjust the importance of the two goals. For data centers that prefer small deadline missing rate, a large value is needed and for those focused on short average FCT, we should set it small. Default value of A is 1.

We now conclude the novelties of DA&FD. Firstly, compared with the hybrid methods that use separated schemas, time is the only criterion used and the performance of all flows could be optimized under one framework in DA&FD. Secondly, with the importance ratio factor A, DA&FD can adjust the effectiveness of the two goals. As a result, more QoS options will be provided. Thirdly, DA&FD can work well even under heavy congestion (and we will prove this later). Compared with the least attained service (LAS) methods, DA&FD has the following advantages. Firstly, DA&FD can reduce the average FCT of type 2 flows without knowing flow size beforehand. However, LAS methods can only cut down the average transfer latency of flows without deadline, and most of them (L^2DCT, PDQ, etc.) need to know flow sizes beforehand, but a large proportion of applications (hadoop, etc.) can't obtain this information beforehand. Secondly, different from LAS methods that only focus on type 2 flows, as equation (4) shown, DA&FD unifies duration and deadline into one framework, so that latency of both type 1 and type 2 flows can be optimized with the same threshold (T). Thirdly, the penalty function of DA&FD can differentiate flows even in heavy congestion, while LAS methods perform close to DCTCP under high load [36]. Especially, different from Aemon [28] that incorporates flow duration time to compute the urgency of deadline flows, DA&FD uses FD on non-deadline flows.

C. DA&FD Analysis

In DA&FD, for non-deadline flows, flows start with highest priority and with time goes by, priorities drop. For deadline flows, the emergency ones have higher priority than the lax ones. In this section, we build sawtooth [4], [23], [29] and fluid model [5], [35] to analyze the performance of DA&FD.

1) DA&FD Property: DA&FD performs well even when the congestion level is heavy. In TCP-like schemes, flow rate is implicitly adjusted by regulating congestion window. Thus we can roughly regard the change of congestion window size as some kind of reflection of flow rate. [36] and [35] advocate that when designing congestion control algorithm, a principle–more load, more differentiation should be followed. Based on this, they propose PI and PD [36] to describe this. Assume all flows have the same threshold. In this sense, we have the following proposition.

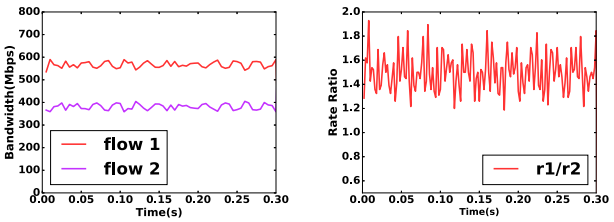
Proposition 1: Let $d = \frac{T}{D}$, $f' = \frac{\alpha}{d}$, then DA&FD respects PI and PD.

Proof: $\frac{\partial(1-f')}{\partial d} = \frac{\partial(1-\alpha/d)}{\partial d} = \frac{\alpha}{d^2} > 0$, and $\frac{\partial^2(1-f')}{\partial d \partial \alpha} = \frac{1}{d^2} > 0$, so the regulation on congestion windows size enforced by DA&FD satisfies the conditions respects PI and PD. ■

From the geometric view. Since f is proportional to α, a scatter plot of (α, f) will be a straight line across the origin. The distance between two such lines, which correspond to different f under different d, just increases with α. DA&FD has another property under some wild assumptions.

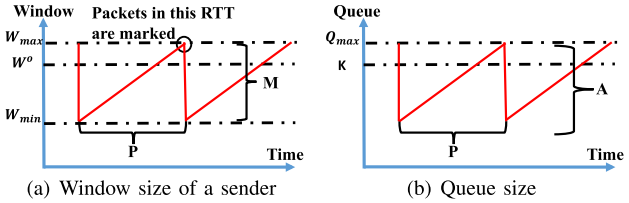
Proposition 2: Assume all flows have deadline and suppose DA&FD can lead to a long term steady state, that is the congestion window of each flow steadily (and synchronously) cycles within a small fixed interval, implying a steady flow

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) Individual bandwidth (b) Bandwidth ratio

Fig. 3. Two flows start simultaneously, where flow1's deadline is 300ms and flow2's deadline is 450ms.



(a) Window size of a sender (b) Queue size

Fig. 4. Window size of a sender and the queue size.

rate and congestion extent. If $D \ll T$, where D and T are parameters denoted by deadline and threshold, then approximately, steady rate a flow achieves is inversely proportional to its corresponding deadline.

Proof: Fig. 4(a) shows the window of a single sender with deadline. We can see, in a steady state, the congestion window size of a flow behaves like a periodic sawtooth with a fixed period P. Using W_{min} and W_{max} to represent the minimum and maximum window size, then according to equation (2), we have $W_{max} = W_{min} + P \times (1 - \frac{D}{T})$, and $W_{min} = W_{max} \times (1 - \alpha \times \frac{D}{T})$, from which we get $W_{max} = P \times (1 - \frac{D}{2T})/(\alpha \times \frac{D}{2T})$, and the average window size $W_{av} = (1 - \alpha \times \frac{D}{4T}) \times W_{max}$.

For two flows f_1 and f_2 with deadline D_1 and D_2, suppose their maximum window sizes are W_1 and W_2, and their steady flow rates are r_1 and r_2, respectively. Then the ratio between r_1 and r_2 is

$$\frac{r_1}{r_2} = \frac{(1 - \alpha \times \frac{D_1}{4T}) \times W_1}{(1 - \alpha \times \frac{D_2}{4T}) \times W_2} = \frac{D_2}{D_1} \times \frac{4T - \alpha \times D_1}{4T - \alpha \times D_2} \times \frac{2T - D_1}{2T - D_2}$$

If $D \ll T$, then $r_1/r_2 \approx D_2/D_1$. ■

To verify proposition 2, we start 2 flows in ns-2 simulator, where flow1's deadline is 600ms and flow2 's deadline is 900ms. Set T to 20s (this guarantees $D \ll T$) and bottleneck link capacity is 1Gbps. Then Fig. 3(a) shows the individual bandwidth of the two flows. We can see bandwidth of flow1 is about 600Mbps and bandwidth of flow2 is about 400Mbps. Fig. 3(b) shows bandwidth ratio of the two flows. We can see that rate ratio of the two flows fluctuates around 1.5, which is inversely proportional to their deadline. Simulation results are consistent with proposition 2.

Proposition 3: Amplitude of the queue oscillations is in $O(C \times T)^{2.5}$when N is small.

Proof: We now consider one of the flows and its penalty function. Let $X(W_{min}, W_{max})$ denote number of packets when window increases from W_{min} to W_{max} (see Fig. 4). Set $b = \alpha \times \frac{f}{2}$ and $e = 1 - \frac{f}{2}$. The average congestion window is $W_{av} = \frac{W_{min} + W_{max}}{2}$ and it takes $(W_{max} - W_{min})/e$ RTTs. So X(W_{min}, W_{max}) can be computed as :

$$X(W_{min}, W_{max}) = \frac{W_{max}^2 - W_{min}^2}{2 \times e} \tag{6}$$

Let $W^o = (CT + K)/N$ denote the window size when queue length of switch reaches K, where C is the bottleneck capacity and N is the number of concurrent flows. After this, all the packets will be marked CE. Then next RTT, another number of W^o packets have been sent, so that we can compute the number of packets that are marked ECN:

$$\alpha = \frac{X(W^o, W^o + e)}{X((W^o + e) \times (1 - b), W^o + e)} = \frac{((W^o + e)^2 - (W^o)^2)}{((W^o + e)^2 - (W^o + e)^2 \times (1 - b)^2)}$$

Simplifying the equation, we can get:

$$\alpha \times (2b - b^2) = (2W^o e + e^2)/(W^o + e)^2 \tag{7}$$

Assume b is small, we have:

$$2\alpha \times b = (2W^o e + e^2)/(W^o + e)^2 \tag{8}$$

As $b = \alpha \times \frac{f}{2}$, plugging this into (8), we get:

$$\alpha = (W^o + e)\sqrt{\frac{2W^o + e^2}{f}}$$
$$b = \frac{(W^o + e)}{2}\sqrt{2fW^o + fe^2} \tag{9}$$

The amplitude of oscillation in the window size of a single flow, M, is given by,

$$M = (W^o + e) - (W^o + e) \times (1 - b) = b \times (W^o + e) \tag{10}$$

Assume there are N flows, then the amplitude of oscillation of the queue is:

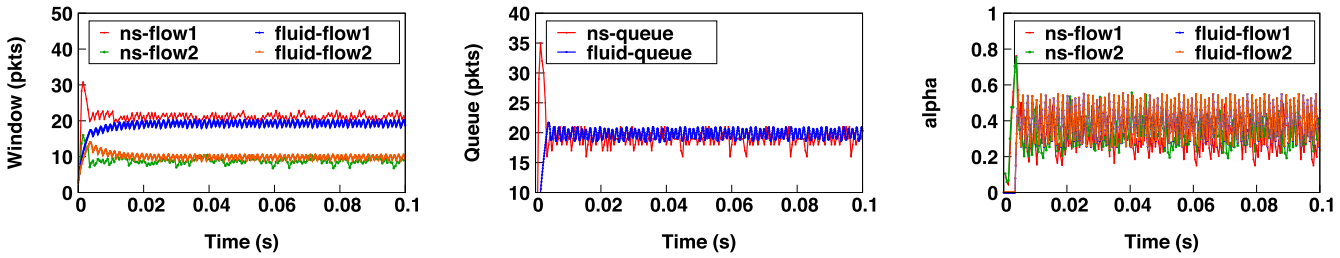
$$A = NM = Nb(W^o + e) = N \times \frac{(W^o + e)^2}{2}\sqrt{2fW^o + fe^2} \tag{11}$$

As f < 1 and e < 1, so $\sqrt{2fW^o + fe^2} < \sqrt{2W^o + 1}$. When $N$is small, $W^o \gg 1$, so that $\sqrt{2W^o + 1} \simeq \sqrt{2W^o}$. As a result, the following holds:

$$A < N \times \frac{(W^o + e)^2}{2}\sqrt{2W^o + 1} \simeq N \times \frac{(W^o)^2}{2}\sqrt{2W^o} \tag{12}$$

$W^o = (CT + K)/N$, so amplitude of the queue oscillations is in $O(C \times T)^{2.5}$ ■

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) window (b) queue length (c) α

Fig. 5. [Simulation] Fluid model and ns-3 simulation comparison.

2) DA&FD Fluid Model: In this section, we build fluid model to understand the behavior of DA&FD [5], [20]. Assume N flows transfer to the same receiver node through a switch and a bottleneck link. All flows start at $t = 0$ and the bottleneck link capacity is C. Let $D_i(t)$ denote the duration time of the i-th flow and $T_i(t)$ denote its threshold. Then DA&FD fluid model can be described as:

$$\frac{d\alpha_i}{dt} = \frac{g}{R(t)}(p(t - R^*) - \alpha_i(t)) \tag{13}$$

$$\frac{dq}{dt} = \sum_{i=1}^{N} \frac{w_i(t)}{R(t)} - C \tag{14}$$

$$\frac{dw_i}{dt} = \frac{1-\frac{D_i(t)}{2T_i(t)}}{R(t)} - \frac{w_i(t)\times\alpha_i(t)\times D_i(t)}{2R(t)T_i(t)}p(t-R^*) \tag{15}$$

Here $R(t) = d + q(t)/C$ is the round-trip time (RTT), where d is the propagation delay (assumed to be equal for all flows), and $q(t)/C$ is the queueing delay. $\widehat{p}(t)$ indicates the packet marking process at the switch and it is computed as:

$$\widehat{p}(t) = \begin{cases} 1 & \widehat{q}(t) \geq K \\ 0 & \widehat{q}(t) < K \end{cases} \tag{16}$$

Equation (13) and equation (16) model the marking process of the switch. Equation (14) models the queue. Equation (15) has two parts, the first part $\frac{1-\frac{D_i(t)}{2T_i(t)}}{R(t)}$ describes the process of window additive increase and the second part $\frac{w_i(t)\times\alpha_i(t)\times D_i(t)}{2R(t)T_i(t)}p(t - R^*)$ describes multiplicative decrease.

We start two flows to verify the accuracy of fluid model. Assume flow1's deadline is 300ms and flow2's deadline is 500ms. Set the time threshold T to 600ms and queue threshold K to 20. DA&FD fluid model can be used to describe congestion window, α and queue. Also, under the same parameters, the ns-2 simulator can also trace congestion window, α and queue. Fig. 5 shows the result of fluid model and simulation under the same configuration. From Fig. 5(a), we can see flow1's window is about 20 and flow2's window is about 12 under the fluid model and ns-3. The average window ratio of flow1 and flow2 is about 5/3, which is inverse proportion to deadline. This also demonstrates proposition 2 from another aspect. Fig. 5(b) shows queue size of the switch. We can see switch queue is about 20 (switch threshold K) under fluid model and ns-3. Fig. 5(c) depicts congestion level, which ranges 0.2 to 0.4. Considering the fluid model results match the simulation results, so that we can use fluid model to analyze DA&FD.

3) How to Set T in Practice: In DA&FD, threshold T can differentiate flows with various deadlines and duration time. In this part, we analyze the influence of T.

Firstly, we discuss the existence of possible fixed points for the fluid model. A possible fixed point $(\tilde{w}, \tilde{\alpha}, \tilde{q})$ of the system should satisfy the following equations:

$$\tilde{p} - \tilde{\alpha} = 0, \sum_{i=1}^{N} \frac{\tilde{w}_i}{\tilde{R}} - C = 0, \quad \tilde{w}_i - \frac{2T_i}{D_i} + 1 = 0 \tag{17}$$

The above equations have a solution only if there are large number of concurrent long flows and all packets are marked CE. Therefore, we have the following two operating regimes: (1) All packets are marked CE. In this regime, the fluid model has a fixed point, namely $(\tilde{w}_i, \tilde{\alpha}_i, \tilde{q}) = (\frac{2T_i}{D_i} - 1, 1, \sum_{i=1}^{N} \tilde{w}_i - Cd)$. This regime corresponds to the large N case, where the system has a very simple steady state behavior: each source transmits fixed number packets per RTT, of which Cd fill the link capacity, and the remaining $\sum_{i=1}^{N} \tilde{w}_i - Cd$ builds up a queue. (2) When concurrent number of flows is small. The system does not have a fixed point and it has a periodic solution or limit cycle. We now use the fixed point to discuss T.

Assume there are N flows ($\mathrm{D}_1 > \mathrm{D}_2 > \mathrm{D}_3 \ldots > \mathrm{D}_n$) starting simultaneously and threshold of all flows are the same. Let $q(t)$ denote queue length at time t, so that we have:

$$\frac{2T}{D_1} - 1 + \frac{2T}{D_2} - 1 + \ldots \frac{2T}{D_n} - 1 < q(t) + C \times RTT \tag{18}$$

As flow deadline have the relationship $\mathrm{D}_1 > \mathrm{D}_2 > \mathrm{D}_3 \ldots . > \mathrm{D}_n$, so we have:

$$N \times \frac{2T}{D_1} - N < q(t) + C \times RTT \tag{19}$$

where C is bottleneck link capacity and K denotes the marking threshold. Window increment is smaller than 1 every RTT, so that $q(t) < K + N$. Take this into (19), we can get:

$$D_1 < T < \frac{D_1 \times (K + 2 \times N + C \times RTT)}{2 \times N} \tag{20}$$

From (20), we can see the upper bound of T is decided by concurrent number of flows, RTT, network bottleneck link capacity and largest flow deadline. In reality, bottleneck capacity(C) can be known by network administrator

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.

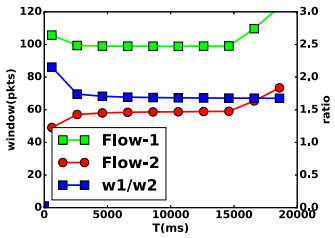


Fig. 6. Two flow example.

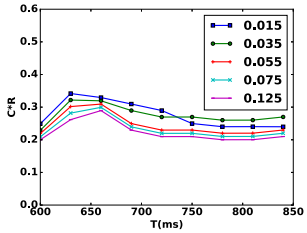


Fig. 7. Queue undershoot.

in advance. Switch threshold K can also be set. From [4], we know concurrent number of flows is always smaller than 100 for data center network. RTT is less than 100ms for most flows [30].

To test the accuracy of the analysis, we start 2 flows, whose deadline are 200ms and 400ms, respectively. If the bottleneck link capacity is 10Gbps and packet size is 1500B. Set RTT to 100us and K to 65. According to (20), T can be set as:

$$400ms < T < 15200ms \tag{21}$$

In each group of experiment, we vary T from 400ms to 20000ms and Fig. 6 shows average window of the two flows when they reach a steady state (left axis). We can see that, when T <15000ms, average window of flow1 is about 100 and average window of flow2 is about 50. Fig. 6 also shows the steady window ratio of the two flows (right axis). We can see when T >15000ms($\simeq$15200ms), window ratio of the two flows is less than 2, which means DA&FD loses efficacy. This inspires us to set reasonable value of T in practice, otherwise, performance of the system will be affected. Further discussion about T will be shown at section IV.

4) How to Set K to Achieve High Link Utilization: According to previous literature [5], a suitable K will make the link utilization high. In this part, we determine how large the queue size oscillations are about the operating point K to achieve 100% link usage.

Assume $qmax$ and $qmin$ denote the maximal and minimal queue size. K should be larger than the minimal queue oscillation (otherwise, link utilization would be less than 100%). Vary bottleneck link from 1Gbps to 10Gbps, then set flow deadline within 400ms. Change T from 600ms to 850ms, we write down the minimal queue size undershoot for each group of T with fluid model. Fig. 7 shows the result. We can see that even for the worst case (T = 620ms,g = 0.015), K should satisfy:

$$K \geq 0.35 \times C \times R \tag{22}$$

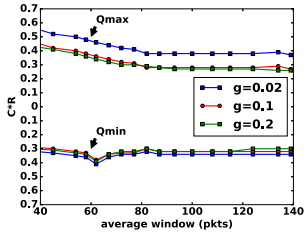


Fig. 8. Under and overshoot.

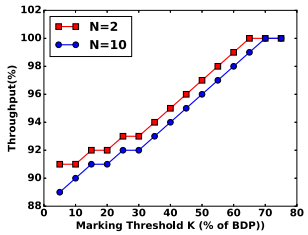


Fig. 9. Utilization vs. BDP.

Let d denote propagation delay of links, so RTT can be computed as $R = d + K/C$. For the worst case, we have $K \geq 0.54Cd$. When C = 10Gbps, d = 100us and packet size is $1500B$, K should at least be 45.

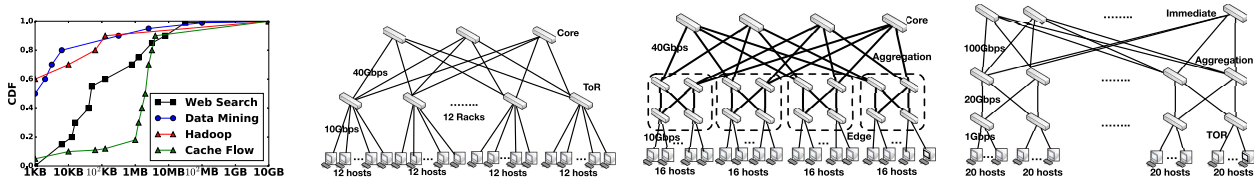
For a common case in data center, where link propagation delay is 100 us and link capacity is 10 Gbps. Set T to 600ms and generate flow deadline within 400ms. Fig. 8 shows $qmin$ and $qmax$ when they are against average window size of flow1 for some values of g with fluid model. We can see that $K \geq$ 0.4CR, so we have $K \geq$0.67Cd. This means 67% of bandwidth-delay product of buffering is needed for 100% throughput. Let flows transfer the bottleneck link with 1Gbps capacity and set propagation delay to 100ms. Vary K from 5% BDP to 80% BDP and Fig. 9 shows link utilization of the bottleneck link when there are 2 and 10 synchronous flows . We can see throughput remains higher than 90% even for K as small as 5% of the BDP. Specially, when K is larger than 60% BDP, link utilization is 100%. The result matches well with the analysis beforehand.

IV. Evaluation

We evaluate the performance of DA&FD using at-scale trace driven simulations and real Linux implementation. Firstly, we evaluate DA&FD in ns-3 and show that it can optimize all type of flows. Secondly, we evaluate DA&FD in real testbed and test its throughput and queuing behavior in the multi-hop environments. Thirdly, we test the performance of DA&FD with real data center traffic under different topologies.

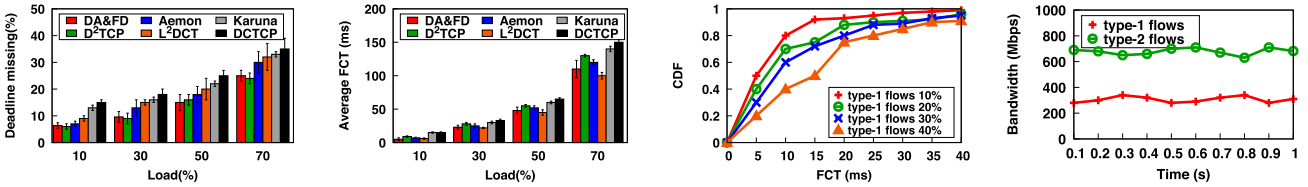
Traffic Workload: For the large scale simulations, we use 4 kinds of traffic trace: data mining, web search, hadoop and cache flow. Web search and data mining are widely used in previous work [4], [6]. Hadoop and cache flow are measured in a small data center by ourself. Fig. 10(a) shows the traffic distribution. We can see that more than 80% of flows are shorter than 10KB for data mining. For web search, more than 80% of flows are between between 1KB and 1MB. For cache flows, we can see most flows are between 1MB and 10MB. While for hadoop, more than 80% flows are shorter than 1MB.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) Traffic distribution (b) Spine-leaf topology (c) Fat-tree topology (d) VL2 topology

Fig. 10. Traffic trace and the data center topology.



(a) Missing deadline (type 1 flows) (b) Average FCT (type 2 flows) (c) Latency distribution (d) Bandwidth (type 1 flows 30%)

Fig. 11. [Simulation] DA&FD performs well with the same parameters.

Hahoop and data mining have similar traffic distribution. This is because many machine learning algorithms are implemented on the top of big data platform.

Performance Metric: Two metrics are used in our experiments. We use the percentage of flows missing deadline to assess the performance of flows with deadline. For other flows, we use average flow completion time. We also implement DCTCP, D^{2}TCP, L^{2}DCT, Aemon, Karuna both in simulations and real testbed to make comparison.

Data Center Topology Three topologies are used in our simulations. Fig. 10(b) shows the details of spine-leaf topology. In the topology, 12 ToR switches connect the core switches through the link with the capacity of 40Gbps. Each ToR switch also connects 12 hosts trough the 10Gbps link. Fig. 10(c) shows the topology of fat-tree. In the topology, 4 aggregation layers connect the 4 core switches through the link with the capacity of 40Gbps. Each edge switch connects 16 hosts with the 10Gbps links. Fig. 10(d) shows the topology of VL2, where 20 hosts connect the edge switch with the 1Gbps links. The aggregation layers connect the immediate layer with the 100Gbps links. In all the topologies, propagation delay between each component is 200us. We also set 250us in the hosts to denote the computation time. Connection number between each host is 8.

Parameters: Proper T should be set to guarantee the performance of DA&FD. According to (20), T should be larger than the maximal value of deadline. In data center, deadline can be tight (20ms), mild (30ms), lax (40ms). The default value of T is 100ms in our experiment. Flow duration begins to count from the time the first packet leaves the network interface of the sender. We cap deadline and flow duration time to 500ms when they are larger than 500ms. When deadline and flow duration are smaller than 20ms, we limit them to 20ms, so applications will not drive the network into congestive collapse. The default value of K is 65 in all the experiments.

Main results of our experiments are:

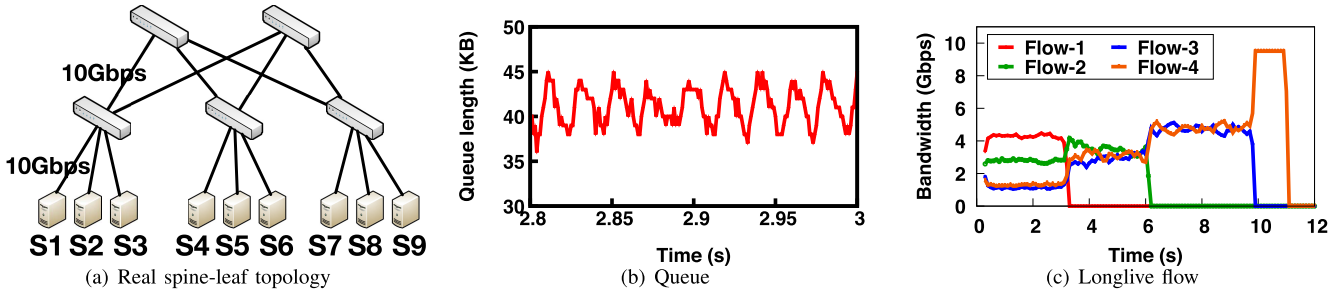
- Trace driven simulations show DA&FD can optimize the performance of both type 1 and type 2 flows simultaneously. For deadline missing rate, DA&FD performs close to D^{2}TCP and about 15%, 25%, 30%, 35% better than Ameon, L^{2}DCT, Karuna and DCTCP. For average FCT, the performance of DA&FD is close to L^{2}DCT and compared with Ameon, D^{2}TCP, Karuna and DCTCP, DA&FD can reduce average FCT by 10%, 15%, 20% and 25%.
- We evaluate DA&FD into linux kernel and use small data center testbed to test the performance of DA&FD. We find real testbed results match the ns-3 simulation results quite well.
- Compared with other mixed flow optimization methods such as Karuna, Ameon, and hybrid, DA&FD performs better on deadline missing and average FCT.
- DA&FD-extension can provide more QoS options by setting different importance factor.

A. Revisiting the Motivating Example

We first apply DA&FD to our motivating example presented in section II. Although this example is not a real scenario, we just use it to show DA&FD can reduce the percentage of flows missing deadline and average flow completion time simultaneously. With the same network topology and network configuration, then Fig. 11 shows the results. Repeat the deadline setting process 100 times, where the error bar paints the mean and mean $\pm$ standard deviation value. We also draw Ameon, L^{2}DCT, Karuna and DCTCP in the picture to make comparison.

From Fig. 11(a), we can see when there are 30% type 1 flows, the deadline missing rate of DA&FD is 6%, 9%, 14%, 23% on average. Compared with D^{2}TCP, L^{2}DCT, Ameon, Karuna and DCTCP under the same network configuration, DA&FD performs close to D^{2}TCP and is about 15%, 20%, 25% 30% better than Ameon, L^{2}DCT, Karuna and DCTCP. Fig. 11(b) shows average FCT (30% flows with deadline) for type 2 flows. Recalling the average FCT of L^{2}DCT and D^{2}TCP (Fig. 1(c) shows the results), DA&FD is about the same as L^{2}DCT and about 10%, 15%, 20%, 25% better than Ameon, D^{2}TCP, Karuna, DCTCP. From the results, we can draw the conclusion that DA&FD can reduce the percentage of

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) Real spine-leaf topology (b) Queue (c) Longlive flow

Fig. 12. [Testbed] Topologies, queue length and flow bandwidth.

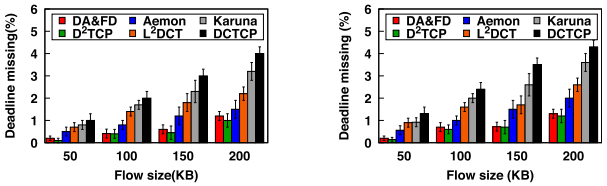
flows missing deadline and average FCT simultaneously. In the simple dumbbell topology, let S1-S4 run data mining workload without deadline and S5 run type 1 flows whose sizes are 5KB. The deadlines are drawn from an exponential distribution with mean 15ms. Set T to 20ms and Fig. 11(c) shows the results. Compared with the hybrid method (shown in Fig. 2), about 20% more flows whose FCT are short ($< 20ms$). DA&FD achieves this by setting the proper threshold T and flows have the highest throughput at the first 20ms. Fig. 11(d) shows the bandwidth of all type 1 and type 2 flows when there are 30% flows with deadline. We can see that the DA&FD almost fully utilize the link capacity, which is about 10% larger than the naive hybrid method.

B. Real Testbed Experiments

In this part, we built a small-scale testbed to evaluate the performance of DA&FD in real network settings. Fig. 12(a) shows the small spine-leaf topology of our experiments. In this topology, 9 servers connect three ToR switches that support ECN marking with the capacity of 10Gbps. The three ToR switches connect the two core switches with the capacity of 10Gbps link. For realizing DA&FD's switch, we use the RED queue implementation in Linux. We set the low and high thresholds of RED queue to K and perform marking based on the instantaneous rather than the average queue length as done in RED.

For the real implementation of DA&FD, deadline or flow duration time should be set to linux kernel. Passing information from user space to kernel space is a challenge. To address this, in the evaluation of DA&FD, we use *setsockopt* to set *mask* for each packet through the socket. The *mask* is a 32 bit integer belonging to *sk_buff*. We use 13 bits for deadline (in ms) or duration and the remaining 19 bits for size information (in KB) in the implementation. Therefore, mark can represent 512 GB (2^{19}KB) flow size and 8192ms (2^{13}ms) time information at most, which can meet the requirements of most data center applications [9]. Specially, flow duration begins to compute when the first packet passes into the traffic control module of the sender. We cap flow size within 1KB and 512GB, flow duration within 1ms and 8192ms in our evaluations. We also evaluate DA&FD, L^{2}DCT, D^{2}TCP, Ameon, Karuna and DCTCP on Linux 3.16 to make comparison.

We first test the performance of DA&FD for long-live flows. Let flow-1 send from S1, flow-2 send from S2, flow-3 send from S4 and flow-4 send from S5. The destination of all flows



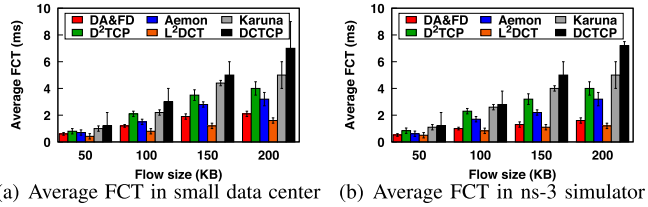
(a) Small data center (b) Ns-3 simulator

Fig. 13. [Testbed] Deadline missing comparison with different mean flow size for type 1 flows.

are S7. Set T to 10000ms and deadline of flow-1, flow-2, flow-3, flow-4 to 3000ms, 7000ms, 8000ms and 20000ms respectively. Flow size of the four flows are 150MB, 220MB, 350MB, 500MB. Note that deadline of flow-4 is 20000ms which is larger than the maximum deadline value (8192ms), so we cap the deadline of flow-4 to 8192ms. The marking threshold on the switch is 40KB. Fig. 12(b) shows queue length of ToR switch connects S7. We can see queue length fluctuates around 40KB and maintains a steady state. Fig. 12(c) shows the bandwidth of the four flows. We can see that for throughput: flow-1>flow-2>flow-3>flow-4. Also, from the picture, we can see the bandwidth of the flows reverses deadline, which matches the analysis in the previous section.

In the following experiment, we imitate the real situation of the data center network. In the topology, each server Si first builds up a connection with a random server. Each connection lasts L min. During the L min, server Si sends short flows with exponential distribution to receiver. After L min, the server Si will choose another random server and repeat the process. We set L to 1 and repeat the process 100 times. Let half of the flows have deadline and we randomly generate deadline between 10ms and 30ms. Set T to 100ms and K to 40KB. Fig. 13 (a) shows percentage of flows missing deadline with different mean flow size. We can observe that for percentage of flows missing deadline, DA&FD performs close to D^{2}TCP and about 15%, 20%, 25% 30% better than Ameon, L^{2}DCT, Karuna and DCTCP. Fig. 14(a) shows average FCT with different mean flow size. We can see that for average FCT, DA&FD performs close to L^{2}DCT and about 25%, 30%, 32%, 35% better than Ameon, D^{2}TCP, Karuna, DCTCP. We also evaluate DA&FD, D^{2}TCP, L^{2}DCT, DCTCP, Karuna, Ameon in ns-3 simulator. Fig. 13(b) and Fig. 14(b) show the corresponding result in ns-3 with the same environment (topology, network capacity and applications, etc.). From the comparison, we can see that the simulation matches the testbed results quite well.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) Average FCT in small data center (b) Average FCT in ns-3 simulator

Fig. 14. [Testbed] Average FCT comparison with different mean flow size for type 2 flows.

C. Large Scale Simulation

From the comparison between the evaluation and testbed results, we can see that the discrete-event network simulator ns-3 [2]simulates the real world well, so that in the following evaluations, we use ns-3 to perform the large scale evaluations.

1) Performance Under Different Traffic Traces and Network Topologies: We first test deadline missing rate under different traffic. The original traffic shown in Fig. 10(a) doesn't contain deadline, so we set half of the flows with uniform distribution deadline between 30ms and 50ms under spine-leaf topology. Repeat the deadline setting process 100 times, where the error bar paints the mean and mean $\pm$ standard deviation value. Fig. 15 (a) shows the comparison of deadline missing and Fig. 15 (b) shows the result of average FCT. We can see that for deadline missing rate, DA&FD performs close to D^{2}TCP and about 15%, 22%, 23%, 28% better than Ameon, L^{2}DCT, Karuna, DCTCP. Compared with Ameon, D^{2}TCP, Karuna, DCTCP, DA&FD reduces average FCT by 13%, 20%, 24%, 30%. Especially, under the cache flow case, the superiority of DA&FD is larger, since many long flows make this scenario heavy congestion and DA&FD performs even better when load is heavy, just as the analysis in the previous section. Fig. 15(c) and Fig. 15(d) show the results of web search under different topologies when there are 50% flows with uniform distribution deadline between 30ms and 50ms. We can see that for percentage of flows missing deadline, DA&FD performs close to D^{2}TCP and better than other methods. The average FCT under DA&FD is almost the same as L^{2}DCT. The results show that DA&FD can optimize the performance of both type 1 and type 2 flows simultaneously.

2) Performance Under Different Settings: In this part, we test the performance of DA&FD under different settings in ns-3 with the spine-leaf topology and web search traffic. Fig. 16(a) shows that percentage of flows missing deadline at first increases and then decreases with type 1 flows increasing. Because when small or large percentages of flows with deadline, priority gap between flows are large and percentage of flows missing deadline reduces. Average FCT for type 2 flows increases as the percentage of type 1 flows increases. From Fig. 10(b), we can see that 12 hosts connect the ToR switch with 10Gbps link in the spine-leaf topology. In modern data center, oversubscription is common and link utilization will be fully used, however, congestion could occur. We assume the link capacity between the core and the aggregate switch is 40Gbps/x, where x denotes the oversubscription factor. Fig. 16(b) shows the performance when we vary the oversubscription factor from 1 to 9. We can see when oversubscription factor is larger, both percentage of flows missing deadline and average flow completion time become larger. In the original spine-leaf topology, the concurrent number of servers is 12. We now test the performance of DA&FD when more servers connect the ToR switch. Fig. 16(c) shows the performance, which is similar to the result of oversubscription factor. When the concurrent number of servers becomes larger, deadline missing rate increases slowly. The reason for this is that the penalty of DA&FD makes it perform better under heavy load.

In DA&FD, parameter T has huge influence on the performance. According to the analysis in section III, T can be set based on largest flow deadline, K, RTT and the concurrent number of flows. Assume the concurrent number of flows is 20 and packet size is 1500B in the data center network. Average link capacity is 10Gbps, then we can use (20) to compute the value of T in practice. If we use D_1 to present the maximal value of deadline, then T can be set as :

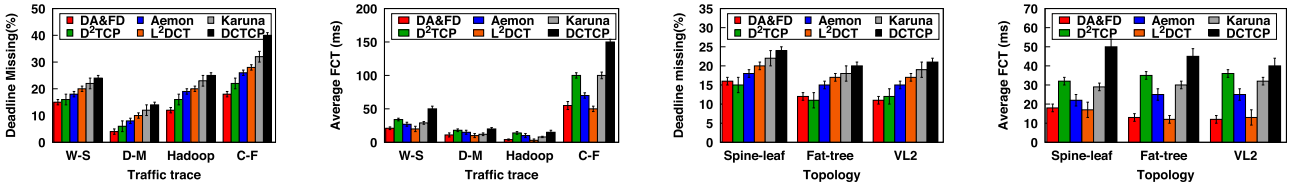
$$D_1 < T < 4.7 \times D_1 \tag{23}$$

Fig. 16(d) shows the result of deadline missing with different maximal flow deadline. For the case of 10ms, percentage of flows missing deadline is less than 7% when T is less than 50ms ($5\times D_1$). However, when T is larger than 50ms, percentage of flows missing deadline becomes larger with the increasing of T. When T is 160ms ($16\times D_1$), percentage of flows missing deadline reaches 9.2%. Similar performance could be seen for the case of 15ms and 20ms. Fig. 16(e) shows the performance with different g. We can see that even $g = 1/2$, DA&FD performances better than D^{2}TCP (8.8%). Note, large or small g could cause performance degradation. The reason for this when g is too large, congestion level (α) is affected by instantaneous network. However, when g is too small, congestion level (α) remains almost the same. According to the analysis in section III-C.4, K should be large enough to occupy the link capacity. Fig. 16(f) shows the performance with different K. We can see that though $K = 10$, link utilization can't reach full and DA&FD still performances better than D^{2}TCP (8.8%) on deadline optimization.

3) Compared With Other Mixed Flow Technologies: We now compare the performance of DA&FD with other mixed flow methods using web search traffic under spine-leaf topology. Let flows have deadline with uniform distribution between 20ms and 50ms. Fig. 17(a) shows the result of deadline missing for type 1 flows when varying the percentage of deadline flows. We can see that DA&FD performs best and it is about 15%, 20% and 25% better than Ameon, Karuna and Hybird method. The hybrid method performs worse than other mixed flow methods since L^{2}DCT hurts the performance of type 1 flows with urgent deadline. Fig. 17(b) shows the result of average FCT for type 2 flows. We can see that DA&FD has the best performance, which allows non-deadline flows to reduce their FCT in presence of deadline flows, while achieving comparable deadline completion to previous deadline-aware schemes. Fig. 17(c) shows the FCT distribution when there are 50% type 1 flows. We can see that FCT of about 80% type 2 flows are less than 80ms with DA&FD, which is about 20%, 25%, 30% better than Aemon, Karuna and Hybrid.

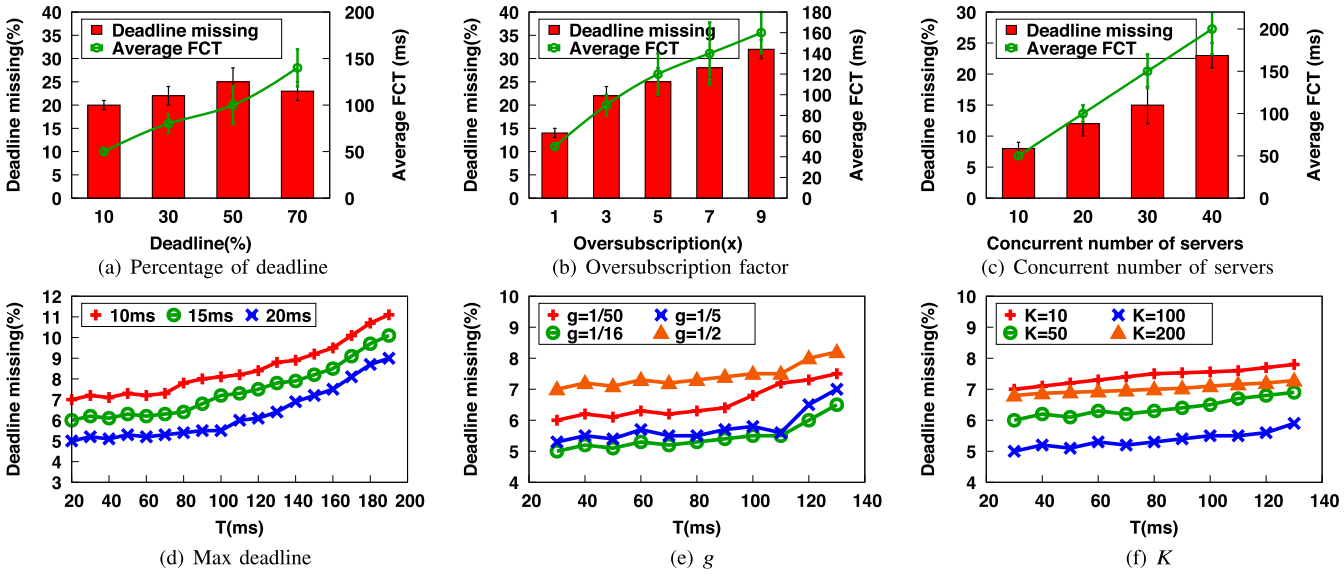
We now compare the performance of DA&FD with the other mixed flow methods in the real spine-leaf topology (Fig. 12(a)

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.



(a) Deadline missing and traces (b) Average FCT and traces (c) Deadline missing and topologies (d) Average FCT and topologies

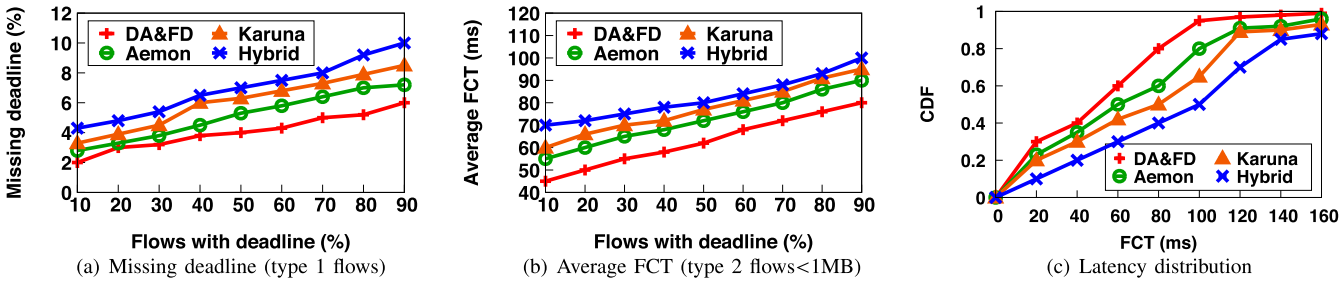
Fig. 15. [Simulation] Deadline missing and average flow completion time under different traffic traces and network topologies, where W-S, D-M and C-F are abbreviations for web search, data mining and cache flow.



(a) Percentage of deadline (b) Oversubscription factor (c) Concurrent number of servers

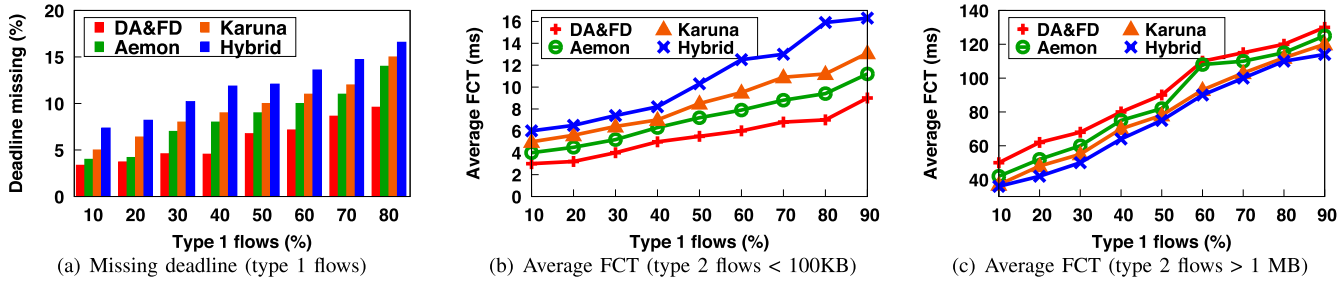
(d) Max deadline (e) g (f) K

Fig. 16. [Simulation] Performance of DA&FD under different settings.



(a) Missing deadline (type 1 flows) (b) Average FCT (type 2 flows<1MB) (c) Latency distribution

Fig. 17. [Simulation] Performance comparison between DA&FD and other mixed flow algorithms.



(a) Missing deadline (type 1 flows) (b) Average FCT (type 2 flows < 100KB) (c) Average FCT (type 2 flows > 1 MB)

Fig. 18. [Testbed] DA&FD and hybrid comparison. Short flows in DA&FD have small average FCT.

shows). S1 and S2 send short flows without deadline and the size of flows are randomly generated with exponential distribution (mean = 100KB). S3 sends type 1 flows whose sizes are 10KB, and their deadlines are drawn from an exponential distribution with mean 15ms. S4 is the destination of all flows. We vary the percentage of type 1 flows, and collect the percentage of flows missing deadline and FCT of type 2 flows. Fig. 18(a) shows the deadline missing rate of type 1 flows and we can see that DA&FD performs about about 22%, 23%, 35% better than Aemon, Karuna and Hybrid on deadline missing. Fig. 18(b) shows the average FCT for type 2 flows whose flow size are shorter than 100KB and Fig. 18(c)

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.

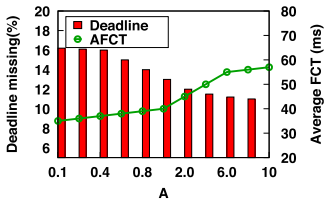


Fig. 19. DA&FD-extension.

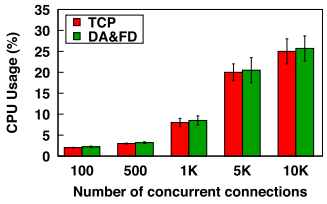


Fig. 20. DA&FD overhead.

shows the result of long flows. We can see that for short flows ($< 100KB$), DA&FD performs at least 15% better than the other methods. Long flows ($> 1MB$) in DA&FD have larger average FCT, since short flows with DA&FD have stronger abilities to scramble bandwidth than other methods.

4) More QoS Options and Overhead: A is 1 in DA&FD default version, where deadline missing rate and average FCT have the same level of importance. When deploying DA&FD in reality, a large A will prefer reducing deadline missing rate and a small value is just the opposite. Fig. 19 shows the performance of DA&FD when we choose A from {0.1, 0.2, 0.4, 0.6, 0.8, 1.0, 2.0, 4.0, 6.0, 8.0, 10} with web search traffic trace under spine-leaf topology. Let half of the flows have deadline ranging from 10ms to 30ms with uniform distribution. We can see that when A is small (< 0.2), DA&FD emphasizes on average FCT optimization. When A is large(> 8), deadline optimization plays a more important role. Fig. 19 also demonstrates that many QoS options can be provided when we choose different A in reality.

Compared with TCP, DA&FD needs to update the congestion estimation (i.e., α) on every packet arrival, so we measure this overhead of DA&FD in our real testbed. We connect two servers to a single switch through the 10Gbps bottleneck link. Multiple simultaneous flows are started from one server to the other and the total CPU utilization is measured on the sender side using sar [1] at the system level. Each flow is given time to perform the handshake and when all are connected, each client sends with a demand of 10 Mbps by sending 128KB bursts every 100 milliseconds (so 1,000 connections saturate the 10 Gbps link). Fig. 20 shows the CPU usage between DA&FD and TCP, where the mean and the standard deviation are computed from 100 trials and plotted. We can see that although DA&FD increases the CPU usage in all cases, the overhead is almost negligible, since it does not vary much under different traffic load (i.e., different number of concurrent flows), and never exceeds 3% in all our experiments.

V. Conclusion

In this paper, we incorporate deadline and flow duration time into flow congestion window updates to satisfy the demands of mix traffic in data center. We design and implement DA&FD (Deadline-Aware and Flow Duration) based rate control mechanism and evaluate it both in ns-3 simulator and Linux kernel. The results show DA&FD can optimize deadline and non-deadline flows simultaneously. We also analyze the performance of DA&FD through fluid models and gains its parameters settings in theory. Although DA&FD performs well both on real testbed and trace driven simulations, more rigorous analysis on DA&FD, such as its stability and convergence properties, requires further studies in the future.

References

[1] *Document of SAR*. Accessed: May 1, 2019. [Online]. Available: https://linux.die.net/man/1/sar
[2] *ns-3*. Accessed: May 1, 2019. [Online]. Available: https://www.nsnam.org/
[3] *The Addition of Explicit Congestion Notification (ECN) to IP*, Standard RFC 3168, IETF, Sep. 2001. [Online]. Available: https://tools.ietf.org/html/rfc3168
[4] M. Alizadeh *et al.*, "Data center TCP (DCTCP)," in *Proc. ACM SIGCOMM Conf.*, Sep. 2010, pp. 63–74.
[5] M. Alizadeh, A. Javanmard, and B. Prabhakar, "Analysis of DCTCP: Stability, convergence, and fairness," in *Proc. ACM SIGMETRICS Joint Int. Conf. Meas. Modeling Comput. Syst.*, Jun. 2011, pp. 73–84.
[6] M. Alizadeh *et al.*, "pFabric: Minimal near-optimal datacenter transport," in *Proc. SIGCOMM*, Aug. 2013, pp. 435–446.
[7] W. Bai, L. Chen, K. Chen, D. Han, C. Tian, and W. Sun, "PIAS: Practical information-agnostic flow scheduling for data center networks," in *Proc. ACM Workshop Hot Topics Netw.*, Oct. 2014, pp. 1–7.
[8] Y. Cao, M. Xu, X. Fu, and E. Dong, "Explicit multipath congestion control for data center networks," in *Proc. 9th ACM Conf. Emerg. Netw. Exp. Technol.*, Dec. 2013, pp. 73–84.
[9] L. Chen, K. Chen, W. Bai, and M. Alizadeh, "Scheduling mix-flows in commodity datacenters with karuna," in *Proc. ACM Conf. (SIGCOMM)*, Aug. 2016, pp. 174–187.
[10] L. Chen, J. Xia, B. Yi, and K. Chen, "PowerMan: An out-of-band management network for datacenters using power line communication," in *Proc. 15th USENIX Symp. Networked Syst. Design Implement. (NSDI)*, 2018, pp. 561–578.
[11] L. Cui, F. R. Yu, and Q. Yan, "When big data meets software-defined networking: SDN for big data and big data for SDN," *IEEE Netw.*, vol. 30, no. 1, pp. 58–65, Jan./Feb. 2016.
[12] C. Gao, V. Lee, and K. Li, "D-SRTF: Distributed shortest remaining time first scheduling for data center networks," *IEEE Trans. Cloud Comput.*, to be published, doi: 10.1109/TCC.2018.2879313.
[13] C.-Y. Hong, M. Caesar, and P. Godfrey, "Finishing flows quickly with preemptive scheduling," in *Proc. ACM Comput. Commun. Rev. (SIGCOMM)*, Aug. 2012, vol. 42, no. 4, pp. 127–138.
[14] M. Kheirkhah, I. Wakeman, and G. Parisis, "MMPTCP: A multipath transport protocol for data centers," in *Proc. IEEE 35th Annu. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Apr. 2016, pp. 1–9.
[15] W. Li, X. Yuan, K. Li, H. Qi, and X. Zhou, "Leveraging endpoint flexibility when scheduling coflows across geo-distributed datacenters," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 873–881.
[16] S. Liu, J. Huang, Y. Zhou, J. Wang, and T. He, "Task-aware TCP in data center networks," *IEEE/ACM Trans. Netw.*, vol. 27, no. 1, pp. 389–404, Feb. 2019.
[17] Z. Liu *et al.*, "Enabling work-conserving bandwidth guarantees for multi-tenant datacenters via dynamic tenant-queue binding," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr. 2018, pp. 1–9.
[18] Y. Lu *et al.*, "One more queue is enough: Minimizing flow completion time with explicit priority notification," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, May 2017, pp. 1–9.
[19] W. Miao *et al.*, "Sdn-enabled OPS with QoS guarantee for reconfigurable virtual data center networks," *IEEE/OSA J. Opt. Commun. Netw.*, vol. 7, no. 7, pp. 634–643, Jul. 2015.
[20] V. Misra, W.-B. Gong, and D. Towsley, "Fluid-based analysis of a network of AQM routers supporting TCP flows with an application to RED," *ACM Comput. Commun. Rev. (SIGCOMM)*, vol. 30, no. 4, pp. 151–160, 2000.
[21] B. Montazeri, Y. Li, M. Alizadeh, and J. Ousterhout, "Homa: A receiver-driven low-latency transport protocol using network priorities," in *Proc. Conf. ACM Special Interest Group Data Commun.*, Aug. 2018, pp. 221–235.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.

[22] A. Munir, G. Baig, S. M. Irteza, I. A. Qazi, A. X. Liu, and F. R. Dogar, "Friends, not foes: Synthesizing existing transport strategies for data center networks," in *Proc. ACM Conf. SIGCOMM*, Aug. 2014, pp. 491–502.
[23] A. Munir *et al.*, "Minimizing flow completion times in data centers," in *Proc. IEEE INFOCOM*, Apr. 2013, pp. 2157–2165.
[24] M. Noormohammadpour and C. S. Raghavendra, "Datacenter traffic control: Understanding techniques and tradeoffs," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1492–1525, 2nd Quart., 2017.
[25] B. Peng, J. Yao, Z. Qi, and H. Guan, "HybridPass: Hybrid scheduling for mixed flows in datacenter networks," in *Proc. IEEE Int. Parallel Distrib. Process. Symp.*, May 2018, pp. 1000–1009.
[26] J. Perry, A. Ousterhout, H. Balakrishnan, D. Shah, and H. Fugal, "Fastpass: A centralized 'zero-queue' datacenter network," in *Proc. ACM SIGCOMM*, Aug. 2014, pp. 307–318.
[27] C. Raiciu, S. Barre, C. Pluntke, A. Greenhalgh, D. Wischik, and M. Handley, "Improving datacenter performance and robustness with multipath TCP," *ACM Comput. Commun. Rev. (SIGCOMM)*, vol. 41, no. 4, pp. 266–277, 2011.
[28] W. Tao, X. Hong, and F. Liu, "Aemon: Information-agnostic mix-flow scheduling in data center networks," in *Proc. Asia–Pacific Workshop Netw.*, Aug. 2017, pp. 106–112.
[29] B. Vamanan, J. Hasan, and T. N. Vijaykumar, "Deadline-aware datacenter tcp (D2TCP)," *SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 115–126, 2012.
[30] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron, "Better never than late: Meeting deadlines in datacenter networks," in *Proc. SIGCOMM*, Aug. 2011, pp. 50–61.
[31] H. Xu and B. Li, "RepFlow: Minimizing flow completion times with replicated flows in data centers," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr./May 2014, pp. 1581–1589.
[32] R. Xu, W. Li, K. Li, and X. Zhou, "Shaping deadline coflows to accelerate non-deadline coflows," in *Proc. IEEE/ACM 26th Int. Symp. Qual. Service (IWQoS)*, Jun. 2018, pp. 1–6.
[33] D. Zats, T. Das, D. Borthakur, P. Mohan, D. Borthakur, and R. Katz, "DeTail: Reducing the flow completion time tail in datacenter networks," in *Proc. ACM Comput. Commun. Rev. (SIGCOMM)*, 2012, vol. 42, no. 4, pp. 139–150.
[34] H. Zhang *et al.*, "Guaranteeing deadlines for inter-data center transfers," *IEEE/ACM Trans. Netw.*, vol. 25, no. 1, pp. 579–595, Feb. 2017.
[35] H. Zhang, X. Shi, Y. Guo, Z. Wang, and X. Yin, "More load, more differentiation—Let more flows finish before deadline in data center networks," *Comput. Netw.*, vol. 127, pp. 352–367, Nov. 2017.
[36] H. Zhang, X. Shi, X. Yin, F. Ren, and Z. Wang, "More load, more differentiation—A design principle for deadline-aware congestion control," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Apr./May 2015, pp. 127–135.
[37] H. Zhang, X. Shi, X. Yin, Z. Wang, and Y. Guo, "FDRC—Flow duration time based rate control in data center networks," in *Proc. IEEE/ACM Int. Symp. Qual. Service*, Jun. 2016, pp. 1–10.

Han Zhang (M'15) received the B.S. degree in computer science and technology from Jilin University, and the Ph.D. degree from Tsinghua University. He is currently with the School of Cyber Science and Technology, Beihang University. He has published more than 30 articles in his research areas. His research interests include computer networks, network security, and AI.

Haijun Geng received the B.E. degree from Yantai University in 2008, the M.E. degree from Capital Normal University in 2011, and the Ph.D. degree from Tsinghua University in 2015. He is currently with the School of Software Engineering, Shanxi University. His research interests include future Internet architecture and large-scale Internet routing.

Yahui Li (M'17) received the B.S. degree in computer science and technology from Jilin University. She is currently pursuing the Ph.D. degree with Tsinghua University. Her research interests include computer networks, network security, and AI.

Xia Yin (SM'05) received the B.E., M.E., and Ph.D. degrees in computer science from Tsinghua University in 1995, 1997, and 2000, respectively. She is a Full Professor with the Department of Computer Science and Technology, Tsinghua University. Her research interests include future Internet architecture, formal method, protocol testing, and large-scale Internet routing.

Xingang Shi (M'14) received the B.S. degree from Tsinghua University and the Ph.D. degree from The Chinese University of Hong Kong. He is currently with the Institute for Network Sciences and Cyberspace, Tsinghua University. His research interests include network measurement and routing protocols.

Zhiliang Wang received the B.E., M.E., and Ph.D. degrees in computer science from Tsinghua University, China, in 2001, 2003, and 2006, respectively. He is currently an Associate Professor with the Institute for Network Sciences and Cyberspace, Tsinghua University. His research interests include formal methods and protocol testing, next-generation Internet, and network measurement.

Qianhong Wu received the Ph.D. degree in cryptography from Xidian University in 2004. Since 2004, he has been an Associate Research Fellow with Wollongong University, Australia, an Associate Professor with Wuhan University, China, and the Research Director of the Universitat Rovira i Virgili, Spain. He is currently a Professor with Beihang University, China. His research interests include cryptography, information security and privacy, VANET security, and cloud computing security. He has been a holder/co-holder of ten China/Australia/Spain-funded projects. He held 30 patents. He has authored over 130 publications. He has served as an Associate/Guest Editor of several international ISI journals and in the program committee of dozens of international conferences. He is a member of the IACR and the ACM.

Jianwei Liu received the B.S. and M.S. degrees in electronic engineering from Shandong University, China, in 1985 and 1988, respectively, and the Ph.D. degree in communication engineering from Xidian University, China, in 1998. He is currently a Professor and the Dean of the School of Cyber Science and Technology, Beihang University. His current research interests include cryptographic protocol design, security on wireless and mobile network, computer network security, and cryptography.

Authorized licensed use limited to: Tsinghua University. Downloaded on October 15,2026 at 14:42:49 UTC from IEEE Xplore. Restrictions apply.