IEICE TRANS. INF. & SYST., VOL.E100–D, NO.7 JULY 2017

LETTER

Maximizing the Profit of Datacenter Networks with HPFF*

Bo LIU[†a)], **Hui HU**[†], **Chao HU**[†,††], **Bo XU**[†], ***Nonmembers***, ***and*** **Bing XU**[†], ***Member***

SUMMARY Maximizing the profit of datacenter networks (DCNs) demands to satisfy more flows' requirements simultaneously, but existing schemes always allocate resource based on single flow attribute, which cannot carry out accurate resource allocation and make many flows failed. In this letter, we propose Highest Priority Flow First (HPFF) to maximize DCN profit, which allocates resource for flows according to the priority. HPFF employs a utility function that considers multiple flow attributes, including flow size, deadline and demanded bandwidth, to calculate the priority for each flow. The experiments on the testbed show that HPFF can improve the network profit by 6.75%-19.7% and decrease the number of failed flow by 26.3%-83.3% compared with existing schemes under real DCN workloads.

key words: *datacenter network, network profit, multiple attributes*

1. Introduction

Maximizing the DCN profit is the basic goal for cloud providers, which demands to maximize number of flows whose requirements are met. How to maximize profit depends on the principle of resource allocation, which affects the DCN profit conversely. Currently, many schemes allocate resource for flows based on single flow attribute such as flow size and deadline. However, these greedy resource allocation schemes based on shortest job first (SJF) [1] or earliest deadline first (EDF) [2] cannot provide required resource for more flows and thus suffer poor performance. For instance, SJF prefers to allocate enough bandwidth for flows with the smallest size rather than latency-sensitive flows or throughput-sensitive flows, which will impair the performance of long flows with deadline or bandwidth requirement. Meanwhile, EDF would impair the performance of throughput-sensitive flows and elastic flows since it greedily allocates all available bandwidth for the earliest deadline flows first. We will demonstrate that single attribute based resource allocation schemes such as SJF and EDF cannot maximize the network profit in Sect. 2, and we insist on considering all available flow attributes in resource allocation to meet more flows' requirements at the same time.

In this letter, we propose HPFF to maximize DCN profit. HPFF calculates a priority for each flow based on multiple flow attributes and allocates resource according to the principle of highest priority flow first. Meanwhile, HPFF maintains real-time attributes of each flow at end hosts and achieves per-flow requirements guarantee with demand-aware rate control. The experimental results on the testbed show that HPFF can effectively improve the network profit and decrease the number of failed flow, which is presented in Sects. 3 and 4, respectively. Moreover, HPFF is easy to deploy since it does not require any modification on DCN applications and switches.

2. Problem Statement

Profit is used to describe the utilization ratio of network resource. We calculate the profit of the whole network as the ratio between the sum of revenue from serving the flows F and the number of flow in the network, which is formulated as

$$Profit = \frac{\sum_{f \in F} R(f)}{num(F)} \tag{1}$$

where $R(f)$ is the revenue of flow f. We describe the flow revenue as follows: for a given flow f with requirements such as demanded bandwidth db and deadline dt, the network profit would be 1 when DCNs can meet these demands, otherwise, the profit is 0. Specifically, for the latency-sensitive flow, the network profit will be 1 if $FCT \leq dt$; while for the throughput-sensitive flow, the network profit will be 1 if $b \geq db$, where FCT and b are the flow completion time (FCT) and transmitting bandwidth of the flow, respectively. For the elastic flow, smaller FCT devotes to higher application performance, thus we use Eq. (2) to calculate the revenue of a given flow:

$$R(f) = Sigmoid_{\alpha}(db - b) \times Sigmoid_{\alpha}(FCT - dt) \times e^{(1 - \frac{FCT \times B}{Size})} \tag{2}$$

where B is the maximum sending rate of the flow and $size$ is the flow size. $Sigmoid(x)$ would be expressed as $Sigmoid_{\alpha}(x) = \frac{1}{1+e^{\alpha x}}$ in which $\alpha = 100$ according to [3]. Obviously, $R(f)$ will be 0 when $db > b$ or $FCT > dt$ for any flow. For elastic flows, we set the demanded bandwidth db

Manuscript received December 13, 2016.
Manuscript revised March 15, 2017.
Manuscript publicized April 5, 2017.

[†]The authors are with the College of Command Information Systems, PLA University of Science and Technology, Nanjing, 210007, China.
[††]The author is also with the Key Laboratory of Computer Network and Information Integration (Southeast University), Ministry of Education, Nanjing, China.

*This work is supported by the State Key Development Program for Basic Research of China under Grant No. 2012CB315806, the National Natural Science Foundation of China under Grant No. 61379149, Jiangsu Province Natural Science Foundation of China under Grant No. BK20140070.

a) E-mail: lbo.xidian@163.com

DOI: 10.1587/transinf.2016EDL8241

Table 1 Flow information

Flow	$f1$	$f2$	$f3$	$f4$
Size(Mb)	40	10	20	30
demand	elastic	$dt \leq 1s$	elastic	$db \geq$10Mbps

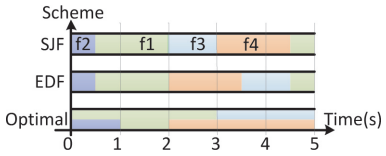


Fig. 1 Bandwidth allocation in different schemes.

Table 2 Results of different schemes

Flow	E-AFCT(s)	Profit	Failure
SJF	3	0.556	25%
EDF	3.75	0.612	0
Optimal	3	0.686	0

to be $b-1$ and the deadline dt to be $FCT+1$, and then the revenue of an elastic flow is mainly decided by $e^{(1-\frac{FCT\times B}{Size})}$, which demonstrates the network should allocate more bandwidth for elastic flows to increase the network profit.

We will use a example to demonstrate two observations: 1) single flow attribute based resource allocation will impair the network profit, and 2) Profit is a efficient metric in evaluating the network performance. There are four flows $f1$, $f2$, $f3$ and $f4$ transmitting on a bottleneck link with 20Mbps available bandwidth, and the requirements of these flows are shown in Table 1, where $f1$ and $f2$ arrive at the same time while $f3$ and $f4$ arrive two seconds later. We compare SJF and EDF with the optimal scheme in bandwidth allocation, where the optimal scheme allocates bandwidth for each flow based on all attributes of the flow.

The results of scheduling and bandwidth allocation are displayed in Fig. 1, and the average flow completion time of elastic flows (E-AFCT), Profit and Failure of the network is exhibited in Table 2.

From Table 2, we get the following observations:

1) SJF and EDF that is based on single flow attribute cannot maximize network profit. Specifically, SJF fails to provide required bandwidth for $f4$, while EDF prolongs the AFCT of elastic flows $f1$ and $f3$ by 25% compared with the optimal scheme. Therefore, all available flow attributes should be considered in resources allocation.

2) The Profit metric is useful in evaluating the performance of overall network. Although EDF can satisfy all flows' requirements in the example as the optimal scheme, it prolongs the AFCT of elastic flows by more than 25%, and we can learn that Profit can distinguish this distinctness while Failure cannot. Therefore, we believe Profit is a good metric in evaluating the network efficiency.

In conclusion, it is urgent to consider all available flow attributes in resource allocation, which can provide comprehensive evaluation on each flow and thus satisfy more flows' requirements. Moreover, a better resource allocation scheme is needed to maximize network profit. In this letter, HPFF is proposed to achieve above objectives.

3. The HPFF Design

In HPFF, flows with the highest priority take precedence in obtaining anticipant bandwidth for maximizing the network profit. Therefore, how to effectively calculate the priority for each flow is the first problem to solve. Since end host is effortless to acquire flows' real-time requirements [4], the functions of flow priority calculation, rate control and per-flow priority tagging are implemented at end hosts. Existing commodity DCN switches always support 4-8 priority queues [1], so how to guarantee per-flow requirements is another problem to solve.

3.1 The Calculation of Flow Priority

We assume that the information about each flow can be derived from the upper layer applications or using state-of-the-art prediction techniques [1], and we leverage the information entropy theory [5] to solve the priority calculation problem. Specifically, let $F=\{f_1,f_2,\cdots,f_n\}$ denote the set of flows to be evaluated, and use $U=\{u_1,u_2,\cdots,u_m\}$ to denote the multiple attributes of flows. Meanwhile, we use $A=(a_{ij})_{n\times m}$ and $V=(r_{ij})_{n\times m}$ to denote the attribute vector and normalized attribute vector, where $r_{ij}=a_{ij}/\sum_{i=1}^{n}a_{ij}$ $(\forall i\in N, j\in M)$. Then the entropy of attribute j can be calculated by the following formula:

$$E_j=-k\sum_{i=1}^{n}r_{ij}\ln r_{ij},\forall i,j \tag{3}$$

where $k=1/\ln n$. Then the weight of attribute j can be calculated as

$$w_j=\frac{1-E_j}{\sum_{k=1}^{m}(1-E_k)},\forall j \tag{4}$$

After calculating the weight of each attribute, the flow priority can be calculated as follows:

$$priority\,(f_i)=\sum_{j=1}^{m}w_j r_{ij},\forall i\in N, j\in M \tag{5}$$

The comprehensive evaluation of $priority(f)$ is in the range of [0,1]. In this letter, we consider three common flow attributes, namely flow size, deadline and demanded bandwidth. For elastic flows, their demanded bandwidth would be set to be the sender NIC (network interface card) rate, which is the maximum sending rate of elastic flows. Meanwhile, we set the deadline of elastic flows to be 1000s, since they have no definite demand on deadline. After that, The priorities of elastic flows are mainly dependent upon their flow size, then we can realize SJF that is the optimal scheduling in decreasing the average FCT for elastic flows. With considering more flow attributes, HPFF can make a accurate decision on flow scheduling compared with SJF and

EDF, which will contribute to higher network profit.

3.2 Resource Allocation

To maximize the network profit, HPFF allocates resource for flows according to the priority, and HPFF implements strict priority scheduling at DCN switches to guarantee the bandwidth reservation for high priority flows. We use four priority queues $\{q_0, q_1, q_2, q_3\}$ in this letter since common switches always support 4-8 priority queues [1], which match to four priority classes $\{p_0, p_1, p_2, p_3\}$ satisfied $p_0 > p_1 > p_2 > p_3$. HPFF conducts multi-level tagging and resource allocation. Specifically, for each flow with its attributes *size*, deadline *dt* and demanded bandwidth *db*, HPFF assigns the highest priority class p_0 to short flows ($size \leq 100$KB) since they are sensitive to queuing delays, which will be forwarded to q_0; while for each throughput-sensitive flow, HPFF allocates required bandwidth for it and sets its priority class to p_1 if *db* is less than the residual bandwidth B of transmitting path, otherwise terminates it. Meanwhile, HPFF allocates $size/dt$ bandwidth to each latency-sensitive flow if $size/dt \leq B$; otherwise, HPFF allocates all residual bandwidth B for it if its deadline is larger than the control peirod T. However, if $size/dt > B$ and $dt \leq T$ for any latency-sensitive flow, HPFF terminates it since its deadline cannot be met. When the latency-sensitive flow is permitted to transmit, HPFF set its priority class to be p_2. Finally, HPFF allocates bandwidth for elastic flows according to the priority and sets the priority class to be p_3.

In this letter, we use the DSCP field in IPv4 header to identify the priority class of a given flow and DCN switches supports DSCP matching. In HPFF, we realize packet-tagging function according to [6], which does not need to modify the applications in the end host. For per-flow rate control, we realize it with Linux traffic control (TC) technology.

3.3 Per-Flow Requirements Guarantee

To guarantee per-flow requirements, HPFF monitors the performance of each flow with the period T. When a flow with higher priority suffers bad performance in a given queue, HPFF will degrade the DSCP of flows with lower priority competing for the same priority queue, and then these priority-degraded flows will be forwarded to a lower priority queue, which will relinquish more bandwidth for higher priority flows. For those flows with degraded priority, strict priority based traffic control mechanism can guarantee no packet disordering. In a word, HPFF maximizes the DCN profit through guaranteeing the requirements of high priority flows and accommodates more flows with low priority.

4. Evaluation

4.1 Methodology

This letter mainly focuses on flow scheduling, so we implement HPFF in a single-switch testbed as [2], [6] to escape the routing problem in high interconnectivity DCNs such as FatTree [7]. There are five Lenovo PCs, which run Ubuntu 12.04.5 LTS operating system. The switch is Pica8, and the bandwidth of each link is 1Gbps. In order to ensure the accuracy of experiments, each set of experiments is executed for ten runs lasting eight hours, and we choose the average value as the experimental results.

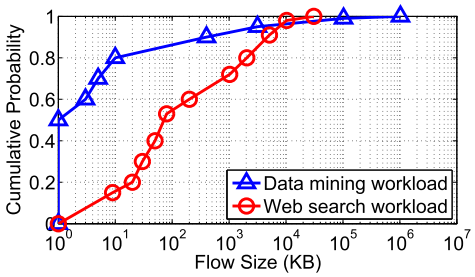


Fig. 2 Workloads information.

Benchmark workloads: We consider two flow size distributions. The first is from a cluster running web search [8], and the second is from a data center mostly running data mining jobs [1]. Both workloads exhibit heavy-tailed characteristics with mixing of short and long flows. In the web search workload, over 95% of the bytes are from 30% of flows larger than 1MB. In the data mining workload, 95% of all bytes are from 3.6% flows that are larger than 35MB while more than 80% of flows are less than 10KB. The flow size distributions of the two set workloads is shown in Fig. 2. Flows are generated between each pair of hosts with iperf traffic generator based on random communication pattern following a poisson process, which includes one-to-one, one-to-multi and all-to-all communication patterns. We set the requirements of flows in accordance with [2].

Schemes Compared

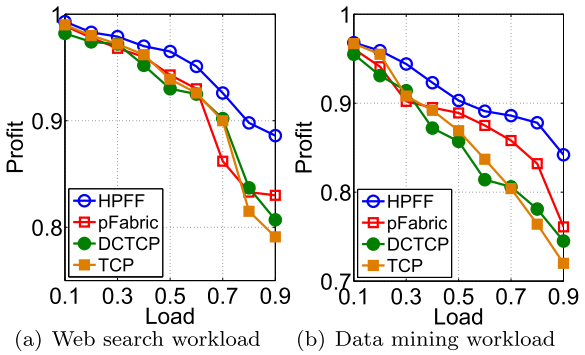
TCP: TCP CUBIC is used as the baseline of our evaluation. The initial window is set to 12 packets [1], and Pica8 switch uses DropTail queues and FCFS (first come first serve) scheduling with a buffer size of 150 packets as [1].

DCTCP: The DCTCP protocol with ECN marking at DropTail queues [8]. The ECN marking threshold is set to K=20 packets. Other parameters are set following [8].

pFabric: We realize pFabric following [1] and set the priority of packets based on flow size. Meanwhile, we set DropTail queue size to be 36KB at Pica8 switch port for best performance as [1] suggested. We set eight priority queues in Pica8 switch port since it only supports eight priority queues.

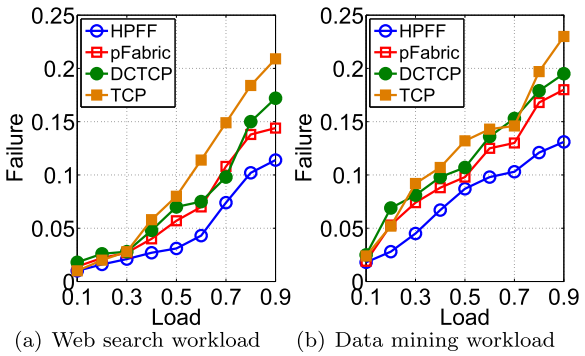
HPFF: Our design is described in Sect. 3. We still utilize TCP CUBIC in end hosts as mentioned, and set T to be 4×RTT.

Metrics: We use Profit and Failure to evaluate the performance of above schemes, where $Failure = \frac{num(failed\_flows)}{num(all\_flows)}$. Failed flows include deadline-missing, terminated and flows that fail to get the required transmitting



(a) Web search workload (b) Data mining workload

Fig. 3 Profit of different schemes under the web search workload and data mining workload.



(a) Web search workload (b) Data mining workload

Fig. 4 Failure of different schemes under the web search workload and data mining workload.

bandwidth.

4.2 Results

Profit. It is apparent that HPFF achieves the highest profit under the two workloads in Fig. 3. Especially in heavy load, HPFF still works well and provides 6.75%-12% improvement in the web search workload and 10.6%-16.9% improvement in the data mining workload compared with other three schemes. Since only considering flow size, pFabric even gets lower profit than DCTCP and TCP when *load*=0.7 under the web search workload, which demonstrates that it is essential to consider multiple flow attributes in improving network profit. DCTCP decreases the queuing delay for (short) flows but fails to guarantee their requirements, which impairs the network profit. Additionally, DCTCP performs worse than TCP sometimes since small buffer causes throughput degradation. TCP allocates bandwidth for flows based on statistic multiplexing which leads to bad performance especially in heavy load. In summary, HPFF achieves the best performance in improving network profit by superior resource allocation principle.

Failure. The failure of different schemes is presented in Fig. 4, and HPFF suffers less failure than other schemes, which means HPFF can guarantee more flows' requirements. Specifically, HPFF decreases failure by 26.3%-83.3% in the web search workload and 37.4%-75.6% in the data mining workload. TCP neither provides more bandwidth for latency-sensitive flows nor schedules the highest profit flow first and thus leads to more than 21% flows failed when *load*=0.9. DCTCP decreases queuing delay and pFabric allocates more bandwidth for small flows, so they can meet more short flows' requirements. However, there still have wide performance gap compared with HPFF due to lacking of considering multiple flow attributes that impairs flow performance. In conclusion, HPFF can maximize the network profit and guarantee more flows' requirements under two DCNs workloads.

5. Conclusions

In this paper, we propose HPFF to improve the profit of DCNs. HPFF calculates priority for each flow according to multiple flow attributes, and allocates required bandwidth for the highest priority flow first to maximize network profit. The experimental results on the testbed demonstrate that HPFF can efficiently improve network profit and guarantee more flows' requirements. Moreover, HPFF does not require any modifications on DCN switches or applications that could be easily deployed. In future work, we prepare to deploy HPFF in real DCN topologies such as FatTree [7] and further optimize its performance.

References

[1] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker, "pFabric: Minimal near-optimal datacenter transport," Proc. SIGCOMM, vol.43, no.4, pp.435–446, 2013.
[2] C.-Y. Hong, M. Caesar, and P.B. Godfrey, "Finishing flows quickly with preemptive scheduling," Proc. SIGCOMM, vol.42, no.4, pp.127–138, 2012.
[3] M. Dong, Q. Li, D. Zarchy, P.B. Godfrey, and M. Schapira, "PCC: Re-architecting Congestion Control for Consistent High Performance," Proc. NSDI, pp.395–408, 2014.
[4] H. Ballani, P. Costa, C. Gkantsidis, M.P. Grosvenor, T. Karagiannis, L. Koromilas, and G. O'Shea, "Enabling End-host Network Functions," Proc. SIGCOMM, vol.45, no.5, pp.493–507, 2015.
[5] C.L. Hwang and K. Yoon, Multiple attribute decision making: Methods and applications a state-of-the-art survey, Springer Science & Business Media, 2012.
[6] W. Bai, L. Chen, K. Chen, D. Han, C. Tian, and H. Wang, "Information-agnostic flow scheduling for commodity data centers," Proc. NSDI, pp.455–468, 2015.
[7] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," Proc. SIGCOMM, pp.63–74, 2008.
[8] M. Alizadeh, A. Greenberg, D.A. Maltz, J. Padhye, P. Patel, B. Prabhakar, S. Sengupta, and M. Sridharan, "Datacenter TCP (DCTCP)," Proc. SIGCOMM, vol.40, no.4, pp.63–74, 2010.