部分先后分别进入到两个队列 Queue1_1 和 Queue2_1 中.f2 较短,只进入到队列 Queue1_2 中.Queue1_1 和 Queue1_2 的优先级最高,Queue2_1 优先级次之.f1_1 和 f2_1 轮流发送预先已设置好的长度,直到发送完毕.然后 f1_2 才被发送.

FISH 对传统 MLFQ 进行了多维扩展,使得同一优先级中的每个流可以被及时地服务,同时也实现了 SJF 算法,保证了小流优先于大流被调度.

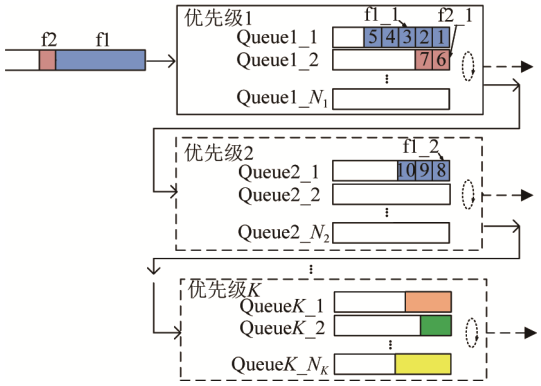


Fig.3 FISH scheduling graph

图 3 FISH 调度示意图

3.2 FISH 的具体实现

FISH 方案主要由交换机完成.我们把流的识别、分配以及调度全部放在交换机上加以实现.如图 4 所示,当数据包进入交换机后,交换机通过对数据包的五元组(源 IP 地址、目的 IP 地址、协议号、源端口、目的端口)的识别来判断该数据包所属的流.对进入交换机的每一条流,计数器会记录这条流有已经进入到交换机中数据包的数目.当一条流的计数器达到某个阈值时,该流的数据包的 DSCP 值会被修改为相应的优先级,然后数据包被分配到相应的优先级下的队列中.

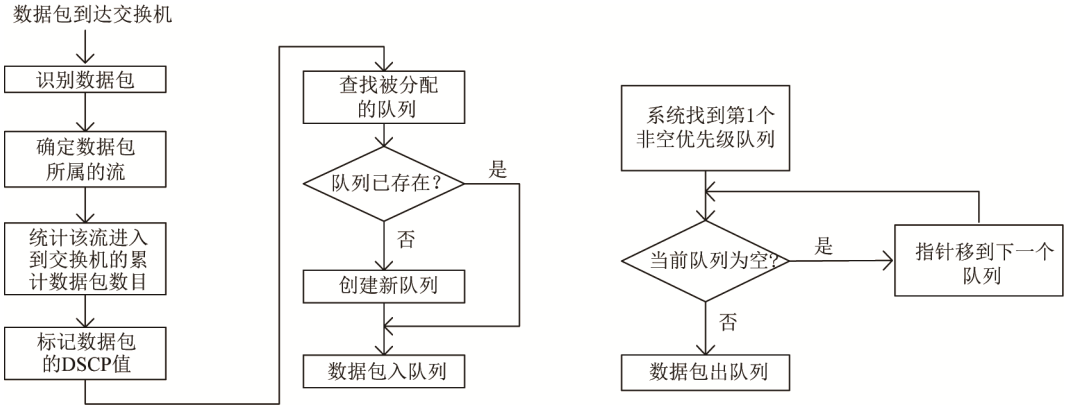


Fig.4 FISH data packet processing

图 4 FISH 数据包处理流程

为了实现 FISH,我们在交换机上将 per-flow 调度与优先级队列调度相结合,实现了一个全新的设计.Per-Flow 调度在交换机的具体实现上可以分为队列的静态划分和动态划分.静态划分简单、直观,但却无法完全满足许多应用的性能需求.动态划分根据流的大小和数量为其分配队列,它对缓存的利用更加充分和灵活,而且可以满足不同应用对性能的潜在需求.

为了在交换机上实现了 per-flow 调度,我们借鉴了动态队列共享(dynamic queue sharing,简称 DQS)机制[11],为具有相同优先级的每个流创建属于自己的队列.当一条新流进入交换机的缓存后,系统会自动地为该流创建队列.当一个队列长时间为空时,队列就会被清除.队列的数量由该流的长度决定.如果流很长,那么该流会从高优先级队列中逐渐降到低优先级队列,所以系统会为它创建多个队列;如果流很短,那么系统很可能只为它创建一个队列.

以图 3 中的流 f1 和 f2 为例,大流 f1 进入交换机后,随着它发送的字节数的增加,系统先后为它创建了两个队列 Queue1_1 和 Queue2_1.同理,由于流 f2 比较短,系统为它创建了一个队列 Queue1_2.队列 Queue1_1 和 Queue1_2 在最高优先级中,所以队列 Queue1_1 和 Queue1_2 轮循发送它们中的数据包,假设每个队列每次只有一个数据包出队,根据本方案中的队列调度算法,数据包发送顺序为 P1→P6→P2→P7→P3→P4→P5.当最高优先级队列中的数据包发送完毕后,开始发送第 2 个优先级队列中的数据.队列 Queue2_1 的优先级次之,所以只能依次发送该队列中的流 f2 的数据包 P8→P9→P10.当一个队列长时间为空时,队列就会被清除.

以下给出了队列调度算法.

算法 1. Queue scheduling.

```
1 while All_queue_length>0
2     do for prio=1 to Prio_queue_num
3         do while Prio_queue_length(prio)>0      //判断同一优先级队列是否为空
4             do if queue[queid]_length()>0
5                 then queue_deque()              //数据包出队
6                     queid←queid+1               //指针移动到下一个队列
7                 else queid←queid+1
8 return NULL
```

默认情况下,系统优先调度高优先级队列中的数据包.但当高优先级队列中不断有小流存在时,由于小流一直被调度,那么处于低优先级队列中的大流就会“挨饿”.为了解决挨饿问题,我们打破了这种严格的优先级限制,将带宽分配给了不同的优先级,即发送一部分高优先级中的流,然后再发送一部分低优先级中的流,这样可以保证低优先级中的大流不会被饿死.

4 实验评估

我们用 NS-2 评估了 FISH.围绕两个关键问题来评估.

(1) FISH 在多种复杂流量环境下表现如何?通过 NS-2 模拟,与 PIAS 相比,在一般的流量环境下,FISH 与 PIAS 的性能相差不大;在流量突发的环境下,FISH 与 PIAS 相比,在小流平均的完成时间上最大降低了 3.23%,在 99%的小流完成时间上最大降低了 19.92%.

(2) FISH 在大型数据中心网络中表现如何?使用大规模 NS-2 模拟,结果表明,FISH 可以扩展到大型数据中心网络中.与 PIAS 相比,可以在很大程度上减小小流的 FCT.经过测试,在网络搜索负载下,降低了小流的平均 FCT 可达 8.6%,对于 99%的小流平均 FCT 最高可以减小 20.9%.

4.1 简单拓扑实验

我们使用真实的数据中心负载,在简单的拓扑环境下测试了 FISH 的表现.

拓扑结构:为了验证 FISH 的有效性,用 NS-2 搭建了一个简单的拓扑,如图 1 所示.2 台服务器连接 1 台交换机,2 台服务器分别为流量的发送端和接收端.服务器到交换机的链接采用 10Gbps 链路.

流量负载:使用真实的网络搜索业务[6]作为流量负载,发送端向接收端发送 100 条流,流的大小基本分布在 1KB~10M 之间,流大小的概率分布函数如图 5 所示.本实验中,设置网络搜索负载大小为 80%,根据流的大小把流划分为 3 类:小流(0,100KB]、中型流(100KB,10MB]和大流(10MB,∞).

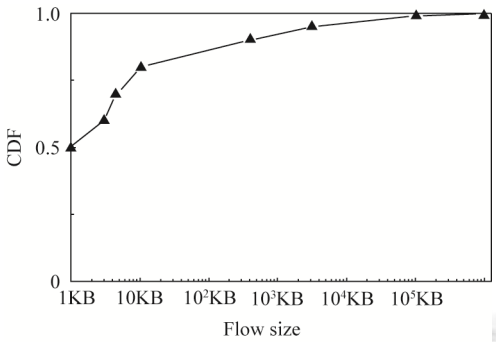
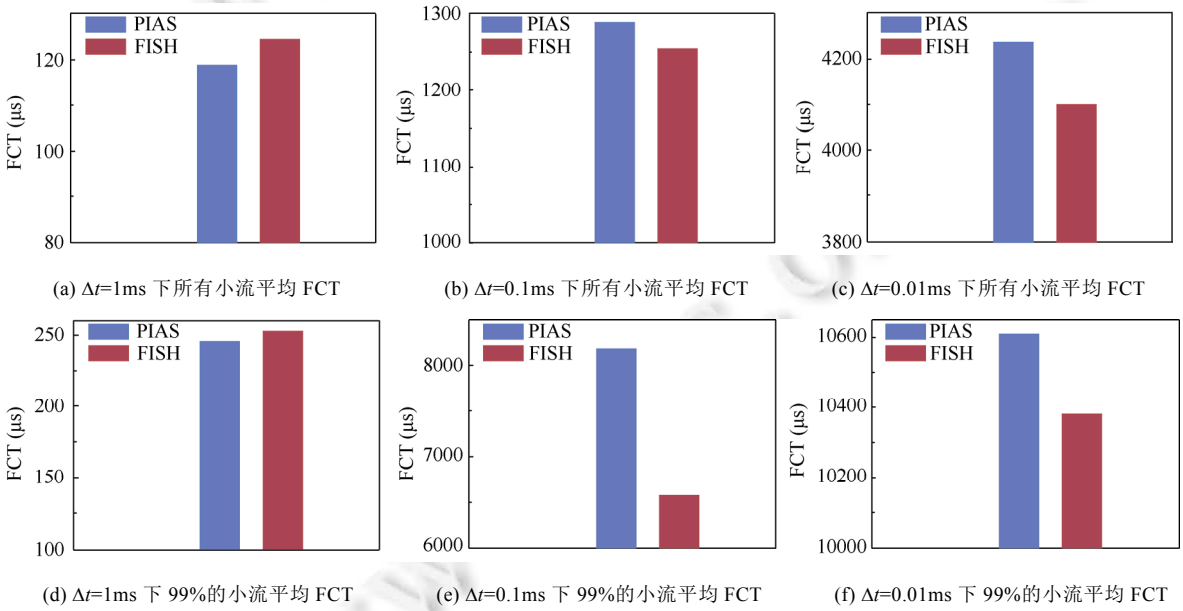


Fig.5 Flow distribution

图 5 流量分布

我们测试了 FISH 在一般的流量环境和流量突发环境下的性能.改变流的平均到达间隔(Δt)来模拟流量突发的情况,对比了 FISH 和 PIAS 在这些环境下的表现.在本文中,设置每个队列每次只发送一个数据包.FISH 和 PIAS 的性能表现如图 6 所示.经过统计,在一般流量环境下,流到达间隔为 1ms,在这种环境下,FISH 对各种大小的流的完成时间与 PIAS 相差不大,如图 6(a)所示.但是随着流达到间隔的不断减小,FISH 的性能优势逐渐显示出来.在流到达间隔为 0.1ms 和 0.01ms 时,如图 6(b)和图 6(c)所示,FISH 比 PIAS 在小流平均完成时间上分别减小了 2.64%和 3.23%;FISH 比 PIAS 在 99%的小流平均完成时间上分别减小了 19.92%和 2.2%.一般流量环境中的流到达间隔比较大,PIAS 中的队列同时被多条流占用的概率降低,而这种情况很难让 FISH 发挥它的作用,所以此时 FISH 和 PIAS 的性能差距不大.而流量突发会使队列被多条流占用的概率增加,此时 FISH 就可以完全发挥它的调度优势,所以 FISH 的性能就会比 PIAS 要好.

针对所有的小流、中型流和大流的完成时间,FISH 与 PIAS 相差不超过 0.6%,见表 1 和表 2.这是由于 FISH 使小流较早被调度,导致大流或者中型流在更晚一些时刻才能被调度完.



(a) Δt=1ms 下所有小流平均 FCT (b) Δt=0.1ms 下所有小流平均 FCT (c) Δt=0.01ms 下所有小流平均 FCT

(d) Δt=1ms 下 99%的小流平均 FCT (e) Δt=0.1ms 下 99%的小流平均 FCT (f) Δt=0.01ms 下 99%的小流平均 FCT

Fig.6 FCT of small flow in the Web search workload

图 6 网络搜索负载下的小流完成时间

Table 1 FCT of medium flows

表 1 中型流的平均 FCT

流到达的平均间隔(ms)	中型流的平均 FCT(ms)	
	PIAS	FISH
1	11.133	11.157
0.1	37.907	38.133
0.01	43.296	43.534

Table 2 FCT of large flows

表 2 大流的平均 FCT

流到达的平均间隔(ms)	大流的平均 FCT(ms)	
	PIAS	FISH
1	79.672	79.450
0.1	122.966	123.287
0.01	128.476	128.198

4.2 数据中心网络拓扑实验

为了观察 FISH 在数据中心网络中的性能,我们在 NS-2 上构建了一个数据中心的网络拓扑来加以评估.使用了真实的数据中心网络负载评估了 FISH.数据中心网络拓扑采用现在比较常用的叶脊(leaf-spine)结构[12],如图 7 所示,一共有 4 台叶交换机、4 台脊交换机和 16 台服务器,叶交换机的上行链路为 20Gbps,下行链路为 10Gbps;服务器两两相互发送数据流共 800 条.设置网络搜索负载大小为 80%.我们选择第 4.1 节给出的实验中的流量作为流量负载,调整流到达间隔,测试 FISH 在各种流量环境下的性能.

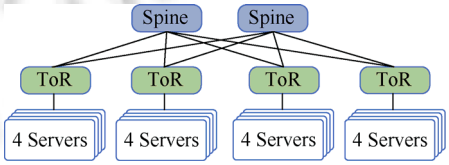


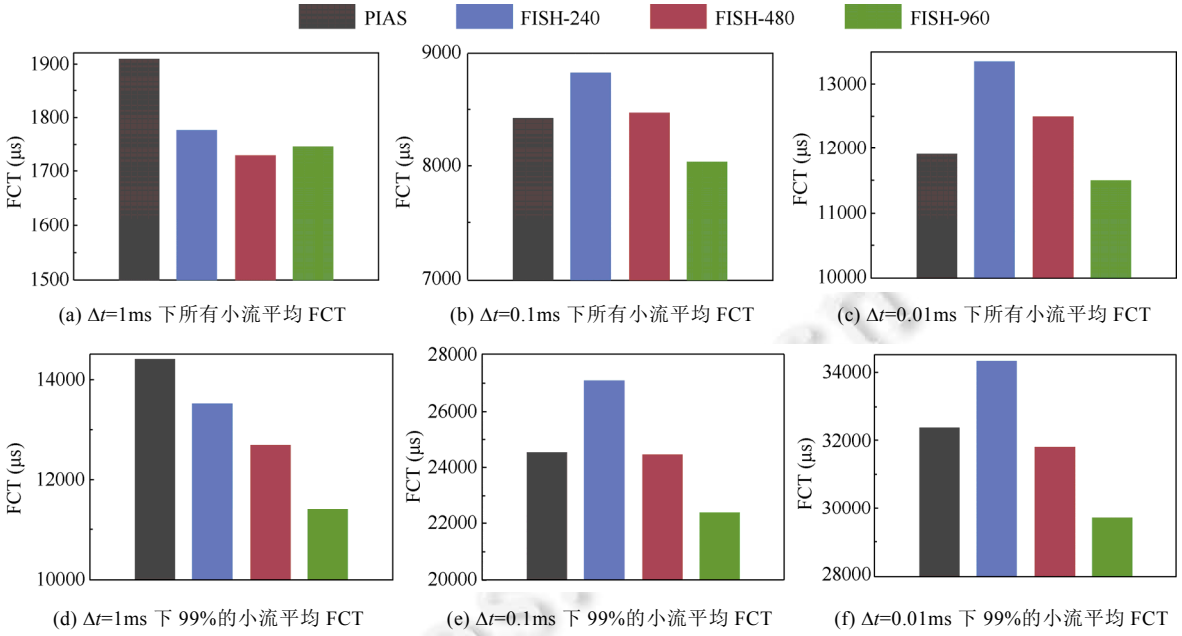
Fig.7 Data center network topology

图 7 数据中心网络拓扑

实验结果显示,在小流的处理上,FISH 明显优于 PIAS.FISH 降低了小流平均完成时间高达 8.6%,降低了 99%的小流平均完成时间最高可达 20.9%.对于中型流和大流,FISH 的性能没有 PIAS 那么好,但是 FISH 与 PIAS 相差不超过 3.8%.

影响 FISH 性能的不但与其机制本身有关,而且与硬件条件也有很大的关系.队列容量指的是所有队列可以容纳的数据包数量.我们比较了不同队列容量的下 FISH 与 PIAS 在不同流到达间隔(Δt)下对小流的处理能力,实验结果如图 8 和图 9 所示,其中,FISH-240、FISH-480 和 FISH-960 分别表示 FISH 中的队列最大可以容纳 240 个、480 个和 960 个包.在实验中我们发现,队列容量对 FISH 的影响比较大.在流量突发情况下,FISH 的性能好坏与队列容量的大小呈正相关性.对于小流,在流量没有发生突发时,如图 8(a)所示,不同容量下的 FISH 比 PIAS 性能都好,最高可以提高 9.5%.这是因为,由于流的到达间隔比较大,FISH 中的队列被数据包占满的概率比较低,队列拥塞丢包出现的情况比较少,所以队列容量的变化对 FISH 影响并不大.在流量突发时,如图 8(b)和图 8(c)所示,FISH 在队列容量为 240 个包时的性能没有 PIAS 好,随着队列容量逐渐变大,FISH 的性能才逐渐超过 PIAS.出现这种现象的原因在于,当 A 的队列容量为 240 个包时,突发的流量使较短的队列在很短的时间内被占满,拥塞丢包导致大量的数据包重传,所以 FISH 在队列容量不够大时性能可能没有 PIAS 好.但是,如图 8(b)~图 8(g)所示,随着 FISH 的队列容量的不断扩展,在降低小流平均完成时间和 99%的小流平均完成时间上,FISH 的性能越来越好.例如,在流到达间隔为 0.01ms 时,随着队列容量的不断增大,在 99%的小流平均完成时间上,在队列容量为 480 个包时,FISH 比 PIAS 性能高 1.8%;在队列容量为 960 个包时,FISH 比 PIAS 性能高 8.2%.由于队列有了足够大的空间就可以容纳更多的等待发送的数据包,FISH 在容量扩展到 960 个包时的性能明显优于

PIAS.在实际部署中,可以考虑增加队列容量以提高 FISH 的性能.



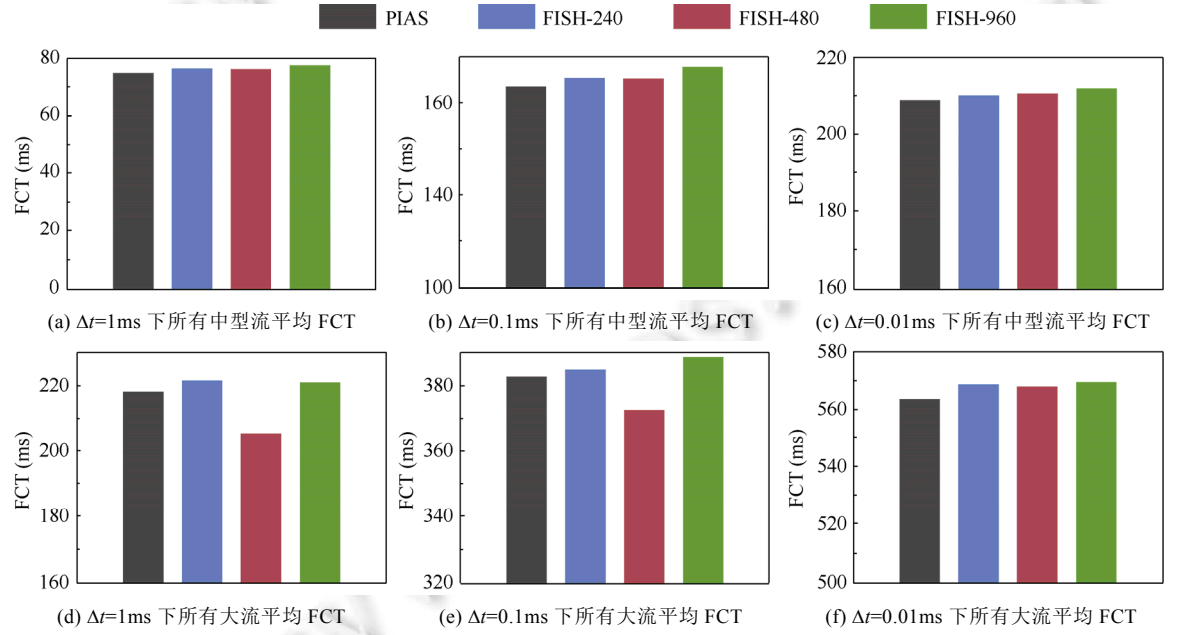
(a) Δ*t*=1ms 下所有小流平均 FCT (b) Δ*t*=0.1ms 下所有小流平均 FCT (c) Δ*t*=0.01ms 下所有小流平均 FCT

(d) Δ*t*=1ms 下 99%的小流平均 FCT (e) Δ*t*=0.1ms 下 99%的小流平均 FCT (f) Δ*t*=0.01ms 下 99%的小流平均 FCT

Fig.8 FCT for small flows by FISH and PIAS under the data center network

图 8 FISH 与 PIAS 在数据中心网络下对小流进行处理的性能比较

FISH 在处理中型流和大流方面,如图 9 所示,FISH 与 PIAS 的性能相接近,甚至 FISH 没有 PIAS 那么好,但是它们相差最多不超过 3.8%.因为 FISH 已让较短的流得到了优先的调度,所以较大的流可能会花费更多的时间才能被传输完毕.



(a) Δ*t*=1ms 下所有中型流平均 FCT (b) Δ*t*=0.1ms 下所有中型流平均 FCT (c) Δ*t*=0.01ms 下所有中型流平均 FCT

(d) Δ*t*=1ms 下所有大流平均 FCT (e) Δ*t*=0.1ms 下所有大流平均 FCT (f) Δ*t*=0.01ms 下所有大流平均 FCT

Fig.9 FCT for medium and large flows by FISH and PIAS in the data center network

图 9 FISH 在数据中心网络下中型流和大流的平均 FCT

5 结 论

我们针对数据中心网络流量突发性提供了一种流调度方案 FISH.该方案把 per-flow 调度的思想引入到 MFLQ 中,在交换机上实现了一个多维的 MFLQ,对流进行调度,解决了流量突发情况下,不同大小的流的排队干扰问题.我们通过一系列实验评估了 FISH 在不同流量环境下的性能,对比实验结果表明,在网络流量突发的情况下,FISH 在减小所有流(特别是小流)的完成时间方面更有优势.

References:

[1] Alizadeh M, Greenberg A, Maltz DA, Padhye J, Patel P, Prabhakar B, Sengupta S, Sridharan M. Data center TCP (DCTCP). In: Bonaventure O, ed. Computer Communication Review. New York: ACM, 2010. 63−74.

[2] Alizadeh M, Kabbani A, Edsall T, Prabhakar B, Vahdat A, Yasuda M. Less is more: Trading a little bandwidth for ultra-low latency in the data center. In: Proc. of the 10th USENIX Symp. on Networked Systems Design and Implementation (NSDI). Berkeley: USENIX, 2012. 19.

[3] Hong CY, Caesar M, Godfrey P. Finishing flows quickly with preemptive scheduling. In: Bonaventure O, ed. Computer Communication Review. New York: ACM, 2012. 127−138.

[4] Alizadeh M, Yang S, Sharif M, Katti S, McKeown N, Prabhakar B, Shenker S. pFabric: Minimal nearoptimal datacenter transport. In: Bonaventure O, ed. Computer Communication Review. New York: ACM, 2013. 435−446.

[5] Munir A, Baig G, Irteza SM, Qazi IA, Liu AX, Dogar FR. Friends, not foes-synthesizing existing transport strategies for data center networks. In: Bonaventure O, ed. Computer Communication Review. New York: ACM, 2014. 491−502.

[6] Bai W, Chen L, Chen K, Han DS, Tian C, Wang H. Information-Agnostic flow scheduling for commodity data centers. In: Proc. of the 12th USENIX Symp. on Networked Systems Design and Implementation (NSDI). Berkeley: USENIX, 2015. 455−468.

[7] Chen L, Chen K, Bai W, Alizadeh M. Scheduling mix-flows in commodity datacenters with Karuna. In: Proc. of the ACM SIGCOMM. New York: ACM, 2016. 174−187.

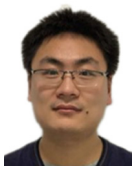
[8] Braden R, Clark D, Shenker S. Integrated services in the Internet architecture: An overview. IERF RFC 1633, 1994.

[9] Kumar VP, Lakshman TV, Stiliadis D. Beyond best effort: Router architectures for the differentiated services of tomorrow's Internet. IEEE Communications Magazine, 1998,36(5):152−164.

[10] Network Simulator NS-2. https://www.isi.edu/nsnam/ns/

[11] Hu C, Tang Y, Chen X, Liu B. Per-Flow queueing by dynamic queue sharing. In: Proc. of the 26th IEEE Int'l Conf. on Computer Communications (INFOCOM). Piscataway: IEEE, 2007. 1613−1621.

[12] Cisco Data Center Spine-and-Leaf Architecture: Design Overview. https://www.cisco.com/c/en/us/products/collateral/switches/nexus-7000-series-switches/white-paper-c11-737022.pdf



张帆(1985－),男,陕西西安人,博士生,主要研究领域为计算机网络,数据中心网络.

胡成臣(1981－),男,博士,教授,博士生导师,CCF 专业会员,主要研究领域为计算机网络,软件定义网络,云数据中心网络.