# 基于松弛时间与累计发送量的数据中心网络混合流调度机制

臧韦菲,兰巨龙,胡宇翔
(国家数字交换系统工程技术研究中心,河南郑州 450002)

**摘 要:** 数据中心网络中同时存在截止时间流(deadline flow)和非截止时间流(non-deadline flow),为降低非截止时间流的平均完成时间(Average Flow Complete Time,AFCT)同时维持低截止时间错失率(Deadline Miss Rate,DMR),本文提出了一种基于松弛时间与累计发送量的混合流调度机制(Slack Time and Accumulation based Mix-flow Scheduling,STAM).首先通过引入松弛时间的概念,衡量截止时间流对非截止时间流在传输时延上的宽容度;然后根据松弛时间,通过使截止时间流尽可能接近其规定截止时间完成,降低非截止时间流的完成时间;最后,利用最小累计发送量优先策略进一步降低非截止时间流的平均完成时间.仿真结果表明,该机制能有效降低非截止时间流的平均完成时间,同时保证较低的截止时间错失率.

**关键词:** 数据中心网络; 流量调度; 截止时间; 松弛时间
**中图分类号:** TP393 **文献标识码:** A **文章编号:** 0372-2112 (2019)10-2061-08
**电子学报 URL:** http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2019.10.006

# Slack Time and Accumulation-Based Mix-flow Scheduling in Data Center Networks

ZANG Wei-fei,LAN Ju-long,HU Yu-xiang
(*National Digital Switching System Engineering & Technological Research Center,Zhengzhou,Henan* 450002,*China*)

**Abstract:** Applications deployed in data center networks generate a mix of flows with and without deadlines. To reduce the average flow complete time (AFCT) while maintain a low deadline miss rate (DMR),a slack time and accumulation-based mix-flow scheduling mechanism (STAM) is proposed in this paper. Firstly,the delay tolerance of deadline flows on non-deadline flows is measured with the introduction of Slack Time. Then the cost of complete time of non-deadline flows is reduced by completing deadline flows barely before their deadlines. Lastly,non-deadline flow is scheduled according to the number of bytes it has sent to reduce the average flow completion time. Experiment results show that the proposed mechanism can effectively reduce the average flow completion time of non-deadline flows while maintaining low deadline miss rate.

**Key words:** data center network;flow scheduling;deadline;slack time

## 1 引言

数据中心网络(Data Center Network,DCN)承载了多样的服务.网页搜索等在线服务产生的流称为截止时间流(deadline flow),当一条流错过其规定的截止时间时,会影响最终的计算结果,因此降低该类流的截止时间错失率(Deadline Miss Rate,DMR)十分重要.虚拟机迁移等服务所产生的流称为非截止时间流(non-deadline flow),流的平均完成时间(Average Flow Complete Time,AFCT)是衡量此类服务性能的一项关键指标[1].

随着服务种类和数量的增多,端到端的传输时延增加,严重影响了用户体验[2].为提高服务性能,当前研究主要从拥塞控制和流量调度两个方面进行优化设

收稿日期:2017-12-27;修回日期:2019-02-28;责任编辑:孙瑶
基金项目:国家网络空间安全专项课题(No.2017YFB0803204);国家 863 高技术研究发展计划(No.2015AA016102);国家自然科学基金创新研究群体科学基金(No.61521003)

计[3]. 拥塞控制通过调整每条流发送窗口的大小,控制交换机内队列的长度. DCTCP[4]利用显式拥塞通告机制(Explicit Congestion Notification,ECN)按比例调整每条流的发送量,缓解网络拥塞,但其忽略了不同流的传输特性,导致较高的截止时间错失率. 为了对不同需求特性的流加以区分,$D^2$TCP[5]和 $L^2$DCT[6]引入优先级系数,采用伽马校正函数对网络拥塞程度进行修正. $D^2$TCP定义了流的紧迫程度,有效降低了截止时间错失率. $L^2$DCT 借鉴最短服务时间优先(Least Attained Service,LAS)的思想,根据每条流的累计发送量动态地调整流的发送速率,实现流的平均完成时间最小. 伽马校正函数存在的问题是,当网络拥塞程度比较大时,所有优先级相似的流的窗口缩减比例相近,优化效果退化为DCTCP.

流量调度通过调整流在交换机内的排队顺序,保证不同流的传输需求. PDQ[7]设计了一种最早截止时间优先(Earliest Deadline First,EDF)和最短流优先(Shortest Job First,SJF)相结合的抢占机制,实现了流的快速调度,但其需要对交换机的硬件进行修改,无法在商用交换机上实现. 为方便部署,PIAS[8]将流调度的决策功能转移到了主机端,结合现有交换机可提供多级优先级队列的特点,在主机端设计了多级反馈队列(Multiple Level Feedback Queue,MLFQ),发送方根据每条流累计发送总量对数据包的优先级进行设定,交换机只需根据包头的优先级将包分配到指定的队列中,通过发送端和接收端主机的协作,有效的降低了流的平均完成时间.

上述研究仅能达到流的平均完成时间或截止时间错失率一项指标上的提升. 针对 DCN 混合流并存的复杂环境,Karuna[1]和 Aemon[9]利用拥塞控制和流量调度相结合的方法,实现多目标优化. Karuna 将因截止时间流引入的包传输时延(per-packet latency)定义为对非截止时间流的影响,交换机端优先转发截止时间流,发送端通过控制每轮截止时间流进入网络的包的数目,降低非截止时间流的平均完成时间. Aemon 的发送端采用基于紧迫度的拥塞控制机制,非截止时间流和截止时间流分别根据流的累计发送量和紧迫度进行调度. 拥塞控制和流量调度相结合的方法可以有效优化两项指标,但其设计的灵活性不强,且需要对 TCP 协议栈进行修改.

针对上述问题,为同时满足 DCN 中不同服务的传输需求,本文提出了一种基于松弛时间与累计发送量的混合流调度机制(Slack Time and Accumulation-based Mix-flow Scheduling,STAM). 当网络发生拥塞时,主机端通过多级反馈队列对每条流的优先级进行决策和调整,其中,截止时间流依据松弛时间大小依次进入指定队列,非截止时间流依据累计发送量按照从小到大的顺序填充剩余队列. 交换机利用严格的多级优先级队列进行转发. 本文的主要贡献和创新工作如下.

(1) 设计了一种针对混合流的优先级调度机制,该机制可以在商用交换机上灵活部署,且不需要对传输层协议进行修改.

(2) 定义了松弛时间的概念,通过牺牲截止时间流的松弛时间,减少非截止时间流的完成时间,同时降低截止时间错失率.

(3) 在仿真平台上模拟了 Incast 同步流和真实网络场景,与现有代表调度机制进行对比,验证了 STAM 的性能.

## 2 问题分析

DCN 中流的完成时间对应的效用函数[10]如图 1 所示. 从图中可以看出,截止时间流在截止时间之前完成所获取的效用值保持不变,当完成时间超出截止时间时,效用值立即下降为零. 而非截止时间流的效用值则随着完成时间的增加不断下降. 现有的流量调度机制 FCFS 采用公平性的调度方式,没有对流的传输特性加以区分,无法满足两类流的传输需求;SJF 按照流的长度从小到大的顺序进行调度,可以有效降低流的整体完成时间,却大大增加了截止时间错失率;EDF 严格优先传输截止时间流,有效缩减了截止时间流的完成时间,却牺牲了非截止时间流的传输时延.

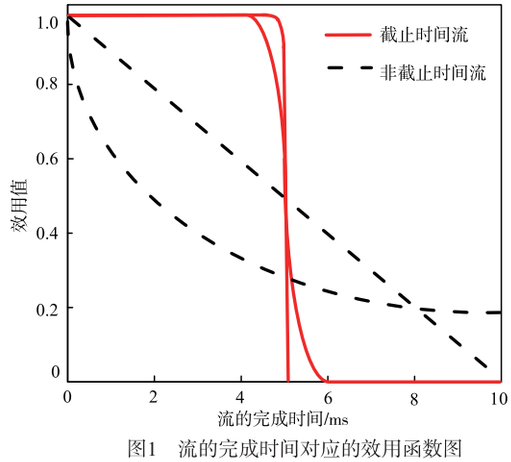



图1 流的完成时间对应的效用函数图

基于以上分析,鉴于截止时间流提前于其截止时间完成不会使效用值增加,本文定义了松弛时间的概念,计算每条截止时间流的预计完成时间较其规定截止时间的提前量,一方面评估了截止时间流的紧迫度,一方面权衡了截止时间流对非截止时间流的宽容度. 根据松弛时间的大小,使部分非截止时间流优先于具有较大松弛时间的截止时间流传输,在保证截止时间流尽可能的接近截止时间完成的前提下,减少非截止

时间流的完成时间. 针对非截止时间流,本文近似实现最短服务时间优先原则对其进行调度,进一步降低该类流的平均完成时间.

下面通过举例说明混合流情况下不同调度机制的结果,网络中流的相关信息如表 1 所示. 图 2(*a*)为 FCFS 的调度结果,由于 A 晚于 D 到达,从而导致 A 错过截止时间,B 和 D 的平均完成时间为:$\frac{(14-3)+(11-1)}{2}=10.5$;图 2(*b*)为 SJF 调度机制调度结果,当短流 A、B 到达时,抢占了 C 的传输,导致 C 错过截止时间,B 和 D 的平均完成时间为:$\frac{(5-3)+(14-1)}{2}=7.5$;图 2(*c*)为 EDF 调度结果,其中,针对非截止时间流,采用 SJF 机制. 因为优先传输非截止时间流,所以 A、C 均在截止时间内完成,B、D 的平均完成时间为:$\frac{(8-3)+(14-1)}{2}=9$;图 2(*d*)中,我们使截止时间流尽可能的接近其截止时间完成,保证 A、C 都在截止时间完成的情况下,B 和 D 的平均完成时间为:$\frac{(7-3)+(14-1)}{2}=8.5$.

**表 1　流的相关参数**

| 流 ID | 长度 | 到达时间 | 截止时间 |
|---|---|---|---|
| A | 1 | 2 | 8 |
| B | 2 | 3 | – |
| C | 5 | 0 | 6 |
| D | 6 | 1 | – |

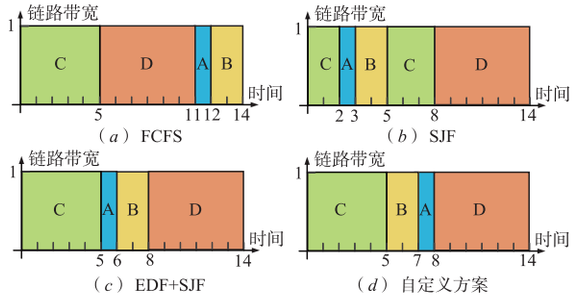



图2　不同机制的调度结果对比图

## 3　STAM 的整体结构

图 3 是 STAM 的整体结构图,主机端根据应用类型对数据流进行分类. 包标记模块负责决策流的优先级,并对 IP 包头的区分服务域(Differentiated Service Code Point,DSCP)进行标记. 交换机利用 ECN 机制将网络拥塞状况反馈给主机,并采用多级优先级队列,依据数据包的包头信息将其放入对应的队列中,严格的按照优先级从高到低进行转发.

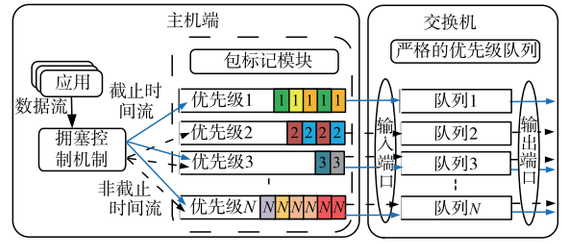



图3　STAM整体结构图

图 4 为包标记模块结构图,该模块位于 TCP/IP 协议栈和 Linux 流量控制(Linux Traffic Control,Linux TC)单元之间. 其中,包分析子模块维持一张流表,用于对进入主机的流进行统计和分析,同时对网络的状态进行判断和更新. 基于优先级的流量调度子模块内部维护一个多级反馈队列,每一轮往返时延(Round Trip Time,RTT)内,根据包分析子模块的信息,调整优先级调度策略. 包标记模块的工作流程如算法 1 所示. 若主机为接收端,当网络发生拥塞时,基于优先级的流量调度子模块调用基于松弛时间的优先级调整策略 SlackTimebasedPolicy()和基于累计发送量的优先级调整策略 AccumulationbasedPolicy(),分别对截止时间流和非截止时间流的 ACK 的返回序列做出调整,并将序列号标记到 ACK 的 IP 头部;否则,按照数据包的到达顺序,以先进先出的顺序返回 ACK 包. 若主机为发送端,初始时采用累计发送量最小优先原则进行流调度,当 ACK 包通告网络发生拥塞时,主机根据 ACK 包头的 DSCP 域调整更新下一轮流的优先级顺序.

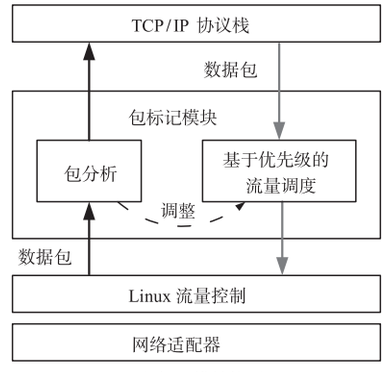



图4　包标记模结构图

**算法 1　包标记模块的工作流程**

输入:数据包 packet
输出:Queue　//调度序列
(1)　If packet. type　= SYN/FIN/data packet //主机为接收端
(2)　If packet. ECN = 11 //当网络发送拥塞时
(3)　　DeadlineQueue = SlackTimebasedPolicy();
(4)　　NonDeadlineQueue = AccumulationbasedPolicy();
(5)　　ACKqueue = DeadlineQueue ∪ NonDeadlineQueue;

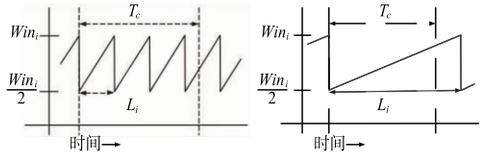
```
(6)      根据 ACK 队列标记每个 ACK 包头中 DSCP 域
(7)   Else
(8)      ACKqueue = FCFS();
(9)   End if
(10)  End if
(11)  If packet. type = ACK //主机为发送端
(12)     根据 ACK 携带的信息更新流表;
(13)     If (Queue = null)
(14)        Queue = LAS;
(15)     End if;
(16)     If packet. ECN = 11
(17)        根据 ACK 队列的序列更新包发送序列;
(18)     End if;
(19)        根据 ACK 队列标记每个数据包包头中 DSCP 域;
(20)  End if;
(21)  Return Queue;
```

## 4 优先级调整策略设计

### 4.1 松弛时间计算

**定义 1** 松弛时间 $T_{\text{slack}}$:每条截止时间流传输完成预计需要的剩余往返时延与其加权平均往返时延 $RTT_s$ 之间的差值.

截止时间流 $df_i$ 的窗口大小记作 $Win_i$,剩余长度记作 $B_i$. 当接收端检测到网络发生拥塞时,为获得流的预计完成时间的上界,假设流经历了满拥塞过程[5](即拥塞程度 $\alpha=1$),发送端基于 TCP 协议进行窗口变化,拥塞时窗口减为 $\frac{Win_i}{2}$,同时进入拥塞避免阶段,窗口以每轮 $RTT$ 内增加 1 个最大报文段长度(Maximum Segment Size,MSS)的幅度上升,且当窗口恢复到 $Win_i$ 时,再次经历满拥塞,则 $df_i$ 剩余长度传输完成所经历的窗口变化过程如图 5 所示.




(a) $B_i \geq S\left(\frac{Win_i}{2}, Win_i\right)$ (b) $B_i < S\left(\frac{Win_i}{2}, Win_i\right)$

图5 $df_i$ 剩余长度传输完成所经历的窗口变化过程

将 $df_i$ 的窗口从 $\frac{Win_i}{2}$ 上升为 $Win_i$ 的过程中,总共发送的数据包数目记作 $S\left(\frac{W_i}{2}, W_i\right)$,则该段时间内流的平均窗口大小为 $\frac{3Win_i}{4}$,共经历了 $L_i=\frac{Win_i}{2}$ 轮 $RTT$,则

$$S\left(\frac{Win_i}{2}, Win_i\right)=\frac{3Win_i}{4}\times\frac{Win_i}{2}=\frac{3Win_i}{8} \tag{1}$$

传输完成剩余长度需要的 $RTT$ 数目 $T_c$ 为:

$$T_c=\begin{cases}\dfrac{B_i}{\frac{3}{4}Win_i-\frac{1}{2}}, & \text{if } B_i\geq S\left(\dfrac{Win_i}{2}, Win_i\right)\\ \sqrt{1/4\,(Win_i-1)^2+2B_i}-\dfrac{Win_i-1}{2}, & \text{else}\end{cases} \tag{2}$$

$RTT_s$ 是 $df_i$ 传输过程所经历的每轮 $RTT$ 的加权平均值,其初始值为不考虑排队时延的情况下,DCN 的最小往返时延[11]. 每经过一轮传输,对 $RTT_s$ 进行更新,其更新公式如下:

$$RTT_s^{\text{new}}=(1-\omega)\times RTT_s^{\text{old}}+\omega\times RTT_{\text{sample}} \tag{3}$$

其中,$RTT_s^{\text{new}}$ 为下一轮的加权往返时延,$RTT_s^{\text{old}}$ 为目前为止所经过的 $RTT$ 的加权平均值,$RTT_{\text{sample}}$ 为最新一轮 $RTT$ 的采样值. $\omega\in[0,1]$ 为权重值.

将 $df_i$ 的剩余截止时间记作 Deadline,则每轮 $RTT$ 内,流的松弛时间的计算公式为

$$T_{\text{slack}}=\frac{\text{Deadline}}{T_c}-RTT_s \tag{4}$$

当存在流的松弛时间 $T_{\text{slack}}<0$,即该流无法在截止时间内完成时,则发送端返回 FIN 包终止该流的传输,以减少其余流的排队时延.

### 4.2 优先级调整策略

首先,对截止时间流进行优先级调整. 交换机采用基于端口的 ECN 机制,每个端口仅有一个阈值 $K$,当所有队列的总长度超出阈值时,才对新到的包进行标记. 设交换机内共有 $N$ 条队列 $P_j$,$1\leq j\leq N$,其中 $P_1$ 的优先级最高,所有队列公平共享内存,则每条队列可容纳的最大队列长度为 $\lfloor K/N\rfloor$[12]. 假设流进入交换机服从 $M/M/1$ 排队模型[1,8,9],其中,流到达交换机队列的平均到达速率记为 $\lambda$,交换机的平均服务速率记为 $\mu$,则 $\lambda_j$ 表示所有进入 $P_j$ 的流的平均到达速率,$\mu_j$ 表示第 $j$ 条队列的平均服务速率.

将下一轮 $RTT$ 时排列在第 $j$ 队列中的数据包总数记作 $x_j$,$x_j$ 的期望 $E[x_j]\leq\lfloor K/N\rfloor$,则进入 $P_j$ 的流的平均到达速率为:

$$\lambda_j=\lambda E[x_j]\leq\lambda\lfloor K/N\rfloor \tag{5}$$

因为交换机内采用严格的优先级调度顺序,所以 $\mu_j$ 的大小取决于链路带宽 $\mu$ 和前 $j-1$ 条队列空闲的概率,每条队列空闲的概率由该队列上的负载量 $\rho=\frac{\lambda}{\mu}$ 决定. 若 $P_1$ 的服务速率 $\mu_1=\mu$,该队列空闲的概率为 $1-\rho_1=1-\frac{\lambda_1}{\mu_1}$;当 $P_1$ 为空时,$P_2$ 获得服务,因此 $\mu_2=(1-\rho_1)\mu$;以此类推,则 $P_j$ 的服务速率为:

$$\mu_j=\prod_{y=0}^{j-1}(1-\rho_y)\mu \tag{6}$$

其中,$\rho_0=0$. 则在第 $j$ 条队列中的所有截止时间流的排队时延的分布函数为:

$$F(T_j)=\begin{cases}0, & T_j\le 0\\ 1-\rho_j e^{-\mu_j(1-\rho_j)T_j}, & T_j>0\end{cases} \tag{7}$$

根据截止时间流的排队时延分布函数,计算第 $j$ 条队列的近似最大排队时延. 在应用允许的范围内,以牺牲 $\eta$ 比例的截止时间错失率来提高非截止时间流的传输效率,$\eta\in(0,0.01]$,另 $F(T_j)=1-\eta$,则 $P_j$ 的近似最大排队时延为

$$T_j=F^{-1}(1-\eta) \tag{8}$$

根据每条截止时间流在每条队列中经历的近似最大排队时延和松弛时间,为该流选择合适的优先级队列,优先级队列 $j$ 的计算公式为:

$$j=\max_{1\le z\le N}\{z,\varepsilon T_{\text{slack}}\ge T_z\} \tag{9}$$

其中,松弛比例因子 $\varepsilon\in[0,1]$,限制了截止时间流可以牺牲的松弛时间范围. 基于松弛时间的优先级调整策略 SlackTimebasedPolicy 如算法 2 所示.

**算法 2　SlackTimebasedPolicy**

输入:DeadlineFlowSet //截止时间流集合及其相关信息
输出:DeadlineQueue //截止时间流的 ACK 调度序列
(1)　For each deadlineflow $df_i$ in DeadlineFlowSet
(2)　　/* 计算每条流的预计剩余完成时间 */
(3)　　根据式(2)计算 $T_c$;
(4)　　根据式(3)更新 $RTT_s$;
(5)　　根据式(4)计算 $T_{\text{slack}}$;
(6)　End for
(7)　/* 按照松弛时间从小到大的顺序对截止时间流进行重新排序 */
(8)　　DeadlineFlowSet' = sortAscending_$T_{\text{slack}}$();
(9)　For each deadlineflow $df_i$ in DeadlineFlowSet';
(10)　　根据式(9)计算包序列号 $j$;
(11)　　标记该包的 ACK 序列为 $j$
(12)　DeadlineQueue. insert(ACK,$j$);
　　//将 ACK 插入到第 j 条队列中
(13)　End for
(14)　Return DeadlineQueue;

然后,对非截止时间流进行优先级调整. 本文选用最小累计发送量优先策略[8]对非截止时间流进行优先级调整. 接收端首先计算下一个 *RTT* 内所有流的拥塞窗口大小 $Win_{\text{next}}$,并估计下一轮 *RTT* 内,交换机中每条队列已经被截止时间流占用的队列长度 QueueOccupied;然后,在保证每条队列的最大长度不超过$\lfloor K/N\rfloor$的前提下,依据流表中的累计发送量将非截止时间流按照从小到大的顺序依次填充 MLFQ,并且属于同一条流的数据包进入同一条队列. 基于累计发送量的优先级调整策略 AccumulationbasedPolicy 的流程如算法 3 所示.

**算法 3　AccumulationbasedPolicy**

输入:DeadlineFlowSet,Non-DeadlineFlowSet NetworkState = $<K,N,\mu,\alpha>$ //网络状态信息
输出:NonDeadlineQueue //非截止时间流 ACK 调度序列
(1)　/* 估算下一轮 *RTT* 中,每条流的发送窗口大小 */
(2)　For each flow $df_i$ in DeadlineFlowSet
(3)　　$Win_{\text{next}}(df_i)$ = CongestionControl($df_i,\alpha$);
(4)　End for
(5)　For each flow $ndf_i$ in Non-DeadlineFlowSet
(6)　　$Win_{\text{next}}(ndf_i)$ = CongestionControl($ndf_i,\alpha$);
(7)　End for
(8)　/* 计算交换机内每条队列的剩余长度 */
(9)　For each queue $j$ in SwitchQueue
(10)　　$\text{QueueOccupied}_j=0$;
(11)　　For each deadlineflow $df_i$ in queue $j$
(12)　　$\text{QueueOccupied}_j=\text{QueueOccupied}_j+Win_{\text{next}}(df_i)$
(13)　　End for
(14)　End for
(15)　/* 按照累计发送量从小到大的顺序对非截止时间流进行重新排序 */
(16) Non-DeadlineFlowSet' = sortAscending_ByteSent();
(17)　/* 将非截止时间流填充到队列中 */
(18)　$j=1$;
(19)　For each flow $ndf_i$ in Non-DeadlineFlowSet'
(20)　　while (SwitchQueue. get($j$). length + $Win_{\text{next}}(ndf_i)\ge\lfloor K/N\rfloor$)
(21)　　$j=j+1$;
(22)　　end while
(23)　标记该包的 ACK 序列为 $j$
(24)　　ACKqueue. insert(ACK,$j$);
(25)　End for
(26)　Return ACKqueue;

## 5　实验结果与分析

本节选取 NS2 搭建仿真环境,对比对象选取基于公平性的流调度机制 FCFS[4],信息无关的调度机制 PIAS[8],基于优先级的调度机制 EDF + SJF[7]以及多目标优化机制 Karuna[1]和 Aemon[9]. 其中,EDF + SJF 采用 EDF 调度截止时间流,采用 SJF 调度非截止时间流,且截止时间流的优先级高于非截止时间流. 除 Karuna 和 Aemon,其余机制均选用 DCTCP 协议进行拥塞控制.

### 5.1　Incast 同步流场景

DCN 中产生截止时间流的应用主要基于 Partition/Aggregate 模型进行传输,在 Partition/Aggregate 中,只有当 Aggregator 收到所有 Worker 的响应才开始下一次请求[13]. 多条流同时到达连接 Aggregator 的交换机导致交换机缓冲区溢出的现象即为 Incast 问题.

#### 5.1.1　仿真设置

拓扑结构如图 6 所示,网络中包含 1 台交换机和 $n+3$ 台主机($n\in[20,40]$,$n\in N$),Worker 产生的流的长度服从 100KB ~ 500KB 之间的均匀分布,截止时间服从 5ms ~ 25ms 之间的均匀分布. 另外,网络中有 2 台主机 Server1 和 Server2 每隔 80ms 向 Aggregator 发送非截止

时间流,流的长度为 10MB. 主机与 ToR 交换机之间的带宽为 10Gbps. 超时重传时延 RTO 设置为 20ms,松弛比例因子 $\varepsilon = 0.7$,截止时间流牺牲比例 $\eta = 0.01$. 每个数据包的大小为 1500Bytes.

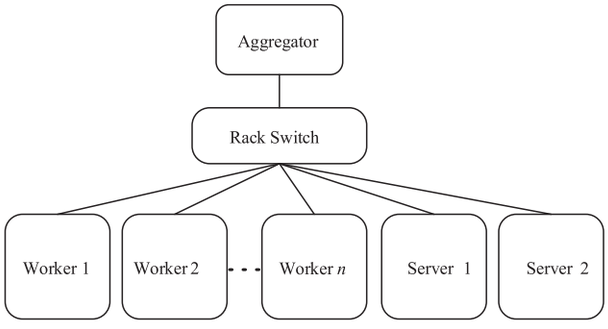



图6 Incast同步流实验场景

### 5.1.2 性能评价

图 7 为截止时间流的丢包率对比图. 在 FCFS 中,当大量非截止时间流排在截止时间流之前时,会导致截止时间流丢包,因此丢包率最高. Aemon 的同一优先级队列中,非截止时间流优先于截止时间流,因此排在后面的截止时间流会被丢弃. PIAS 中非截止时间流的优先级会随着时间的推移而下移,截止时间流的丢包率会逐渐下降. EDF + SJF 和 Karuna 中,截止时间流严格优先于非截止时间流,因此截止时间流的丢包率整体最低. STAM 同一优先级队列中,截止时间流的优先级高于非截止时间流,因此整体丢包率不高,但略高于 EDF + SJF 和 Karuna,在 Worker 数目为 40 时,STAM 的丢包率较 EDF + SJF 和 Karuna 分别高出了 17.6% 和 14.3%,较 FCFS, Aemon 和 PIAS 低了 82.4%,79.7% 和 81.8%.

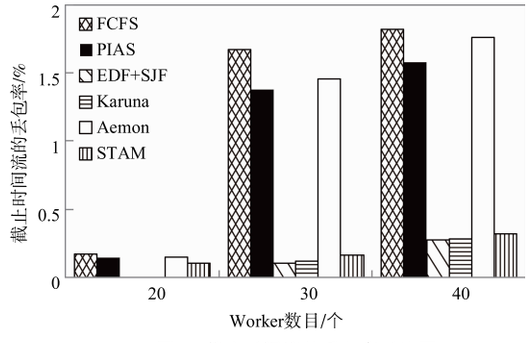



图7 截止时间流的丢包率对比图

图 8 为非截止时间流的平均吞吐量对比图. 在同一优先级队列中,Aemon 优先保证非截止时间流的传输,因此非截止时间流的平均吞吐量最高,FCFS 和 PIAS 没有对截止时间流和非截止时间流区分,先到的非截止时间流会排在截止时间流的前面,从而提高了非截止时间流的平均吞吐量. EDF + SJF 和 Karuna 严格的优先保证截止时间流,从而会产生非截止时间流“饿死”的现象,因此平均吞吐量最低. STAM 根据截止时间流的松弛时间大小,适当的将非截止时间流调度到截止时间流前面进行传输,保证了非截止时间流的传输,当 Worker 数为 40 时,STAM 机制中非截止时间流的平均吞吐量较 FCFS、PIAS、Karuna 和 EDF + SJF 分别提高了 1.8%,2.8%,21.87% 和 25.94%,较 Aemon 降低了 1.7%.

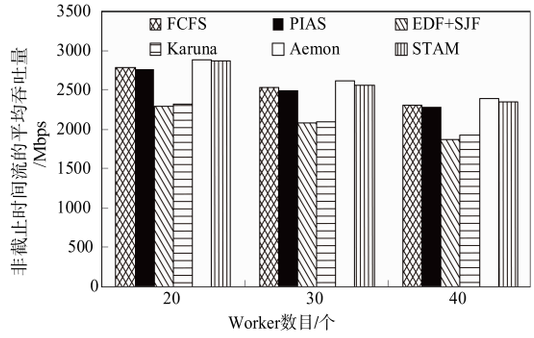



图8 非截止时间流的平均吞吐量对比图

## 5.2 真实网络场景

本小节通过生成 Web search 流量模型从截止时间流的截止时间错失率和非截止时间流的平均完成时间两个方面对 STAM 的有效性进行验证.

### 5.2.1 仿真设置

拓扑结构如图 9 所示,网络中共有 144 台主机,9 台架顶(Top of Rack, ToR)交换机和 4 台 Spine 交换机,其中,主机与 ToR 交换机之间的带宽为 10Gbps, ToR 交换机与 Spine 交换机之间通过 40Gbps 的链路全互连. 初始 RTT 值设置为 100μs. 实时的记录当前网络中截止时间流占据网络负载量的比例,当有新流产生时,若当前截止时间流的负载量小于总负载量的 80%,则该流被标记为截止时间流,且截止时间服从 5ms ~ 25ms 之间的均匀分布;否则,将该流标记为非截止时间流.

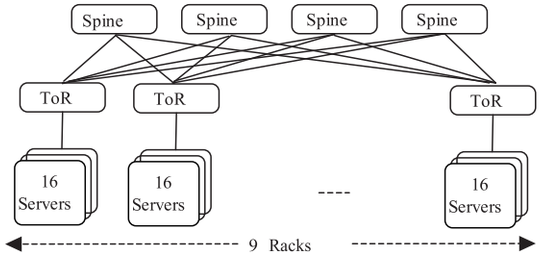



图9 叶脊(leaf-spine)架构

### 5.2.2 性能评价

图 10 为截止时间流的截止时间错失率的对比图. 在 FCFS 和 PIAS 中,先到的非截止时间流会影响后到的截止时间流的传输,因此截止时间错失率高于其它

机制. Aemon 中截止时间流的优先级会随着时间的推移不断上移,因此截止时间错失率较 FCFS 和 PIAS 略低. EDF + SJF 和 Karuna 优先传输截止时间流,且严格的优于非截止时间流,因此截止时间错失率低于其他机制. STAM 会根据网络状况和流自身信息,适当的调整两类流在交换机内的转发顺序,在保证截止时间流的传输性能的前提下,降低非截止时间流的传输时延,当网络负载为 80% 时,在 Web search 流量模型下,截止时间错失率较 EDF + SJF 和 Karuna 分别高出了 16.6% 和 6.2%,较 FCFS、PIAS 和 Aemon 分别降低了 22.7%,16.7% 和 7.4%.

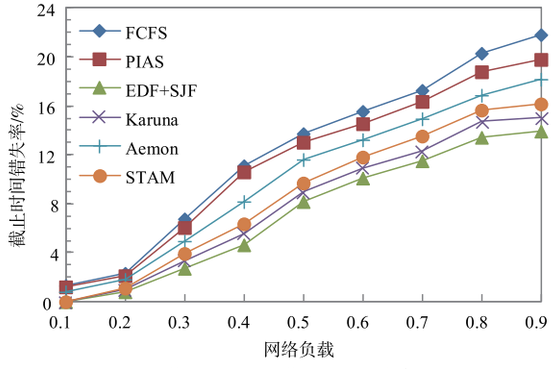
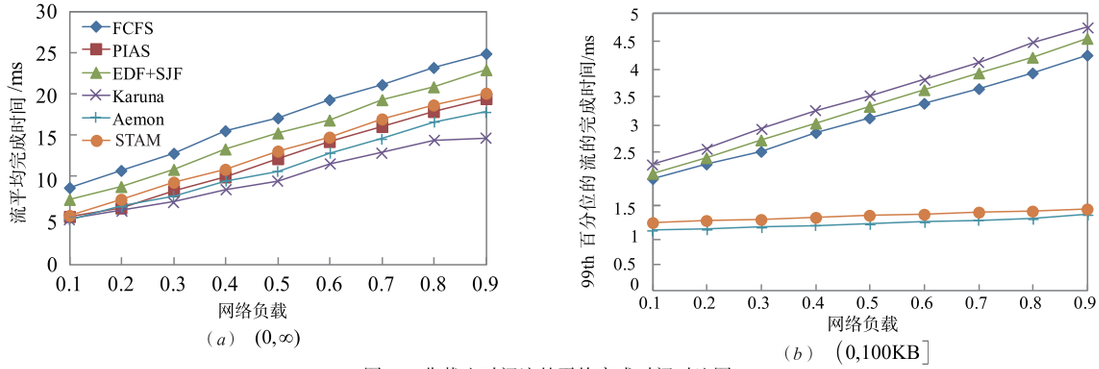



图10　截止时间流的截止时间错失率对比图

图 11(*a*)为全部非截止时间流的平均完成时间对比图. 其中,PIAS 采用累计发送量最小优先的调度策略,在不需要流自身信息的情况下,模拟了 SJF,大大减少了非截止时间流的平均完成时间. STAM 根据截止时间流的松弛时间,适当的提高非截止时间流的优先级,非截止时间流的平均完成时间略高于 PIAS. Karuna 通过调整截止时间流的发送窗口,采用向非截止时间流让步带宽的方式,降低对非截止时间流传输的影响. 当网络负载为 80% 时,在 Web search 流量模型下,STAM 的非截止时间流的平均完成时间较 Karuna、Aemon 和 PIAS 分别高出了 28.1%,11.8% 和 3.8%,较 FCFS、EDF + SJF 分别降低了 19.2% 和 11.6%.

图 11(*b*)为长度均小于 100KB 的非截止时间流的完成时间对比情况. 从图中可以看出,STAM、PIAS 和 Aemon 机制下非截止时间流的最大完成时间明显小于 Karuna、EDF + SJF 和 FCFS,这是因为,在 Karuna 和 EDF + SJF 中,截止时间流严格优先于非截止时间流传输,从而导致非截止时间流经历较长的排队时延,而针对长度小于 100KB 的流,排队时延占据了流的整体传输完成时间较高的比例,从而导致流的完成时间大大上升. 而在 STAM、PIAS 和 Aemon 中,截止时间流和非截止时间流在队列中交替排列,因此对非截止时间流的排队时延影响较小.




(*a*)　(0,∞)　　(*b*)　(0,100KB]

图11　非截止时间流的平均完成时间对比图

## 6　结束语

为保证数据中心网络多种服务的传输需求,本文提出了一种针对混合流的调度机制,主机端依据截止时间流的松弛时间和交换机多级反馈队列的排队时延,指定截止时间流的优先级,依据累计发送量设定非截止时间流的优先级,同一队列中,截止时间流优于非截止时间流. 仿真结果表明,该机制在 Incast 同步流场景下可以有效降低截止时间流的丢包率,从而减少该类流的截止时间错失率,并保证了非截止时间流的吞吐量.

**参考文献**

[1] CHEN L,CHEN K,BAI W. Scheduling mix-flows in commodity datacenters with karuna[A]. Proceedings of the 2016 ACM SIGCOMM Conference[C]. New York,USA: ACM,2016. 174 – 187.

[2] 黄建洋,兰巨龙,胡宇翔. 一种基于分段路由的多路径流传输机制[J]. 电子学报,2018,46(6):1488 – 1495.
HUANG Jian-yang,LAN Ju-long,HU Yu-xiang. Asegment routing based multipath flow transmission mechanism[J]. Acta Electronica Sinica,2018,46(6):1488 – 1495. (in

Chinese)
[3] XIA W, ZHAO P, WEN Y, et al. A survey on data center networking(DCN): Infrastructure and operations[J]. IEEE Communications Surveys & Tutorials, 2017, 19(1): 640-656.
[4] Alizadeh Mohammad, Greenberg Albert, Maltz David A, et al. Data center TCP (DCTCP)[J]. ACM SIGCOMM Computer Communication Review, 2010, 40(4): 63-74.
[5] VAMANAN B, HASAN J, VIJAYKUMAR T N. Deadline-aware datacenter tcp (D2TCP)[A]. Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication[C]. New York, USA: ACM, 2012. 115-126.
[6] MUNIR A, QAZI I A, UZMI Z A. Minimizing flow completion times in data centers[A]. Proceedings of the 32nd IEEE International Conference on Computer Communications[C]. Piscataway, NJ: IEEE, 2013. 2157-2165.
[7] Chi-Yao HONG, Matthew CAESAR, P Brighten GODFREY. Finishing flows quickly with preemptive scheduling [J]. ACM SIGCOMM Computer Communication Review, 2012, 42(4): 127-138.
[8] BAI W, CHEN L, CHEN K. PIAS: Practical information-agnostic flow scheduling for data center networks[A]. Proceedings of the 13th ACM Workshop on Hot Topics in Networks[C]. New York, USA: ACM, 2014. 25.
[9] WANG T, XU H, LIU F. Aemon: Information-agnostic mix-flow scheduling in data center networks[A]. Proceedings of the First Asia-Pacific Workshop on Networking [C]. New York, USA: ACM, 2017. 106-112.
[10] CHEN L, CUI W. Optimizing coflow completion times with utility max-min fairness[A]. Proceedings of the 35th IEEE International Conference on Computer Communications[C]. Piscataway, NJ: IEEE, 2016. 1-9.
[11] BAI W, CHEN K, WU H. PAC: Taming TCP incast congestion using proactive ACK control[A]. Proceedings of the 22nd International Conference on Network Protocols [C]. Washington, DC, USA: IEEE Computer Society, 2014. 385-396.
[12] BAI W, CHEN L. Enabling ECN in multi-service multi-queue data centers[A]. Proceedings of the 13th Usenix Conference on Networked Systems Design and Implementation[C]. Berkeley, CA, USA: USENIX Association, 2016. 537-549.
[13] 马腾,胡宇翔,张校辉. 基于深度增强学习的数据中心网络 coflow 调度机制[J]. 电子学报,2018,46(7):1617-1624.
MA Teng, HU Yu-xiang, ZHANG Xiao-hui. Deepreinforcement learning based coflow scheduling in data center networks[J]. Acta Electronica Sinica, 2018, 46(7): 1617-1624. (in Chinese)

## 作者简介

**臧韦菲** 女,1993 年出生,河南郑州人. 国家数字交换系统工程技术研究中心博士研究生. 主要研究方向为新型网络体系结构.
E-mail: zwfndsc@163.com

**兰巨龙** 男,1962 年出生,河北张北人. 国家数据交换系统工程技术研究中心教授、博士生导师. 主要从事新一代信息网络关键理论与技术的研究工作.