Fork: A Dual Congestion Control Loop for Small and Large Flows in Datacenters

EuroSys'25

Yuan Liu[1], *Wenxin Li*[1,2], *Yulong Li*[1], *Lide Suo*[1], *Xuan Gao*[1], *Xin Xie*[1], *Sheng Chen*[1,2], *Ziqi Fan*[1], *Wenyu Qu*[1], *Guyue Liu*[3]

1. Tianjin Key Laboratory of Advanced Networking, Tianjin University

2. Huaxiahaorui Technology (Tianjin) Co., Ltd.

3. Peking University

Presenter: Huang Kai

2025.12.8

Research Interests

研究方向

- 云计算与数据中心网络
- 物联网、智慧城市

近几年相关论文

- Fork: A Dual Congestion Control Loop for Small and Large Flows in Datacenters. (EuroSys 2025)
- FUYAO: DPU-enabled Direct Data Transfer for Serverless Computing. (ASPLOS 2024)
- Efficient Coflow Transmission for Distributed Stream Processing. (INFOCOM 2020)

曲雯毓，天津大学智能与计算学部，教授

Contents

- Background
- Design
- Evaluation
- Conclusion

Background

数据中心网络低延迟和高带宽的背景

- 随着网络传输技术革新，端口带宽不断增加，而缓冲区容量的增加速度相对较慢
- 为了有效管理拥塞，需要更好的流量控制机制，以减少数据包丢失和延迟

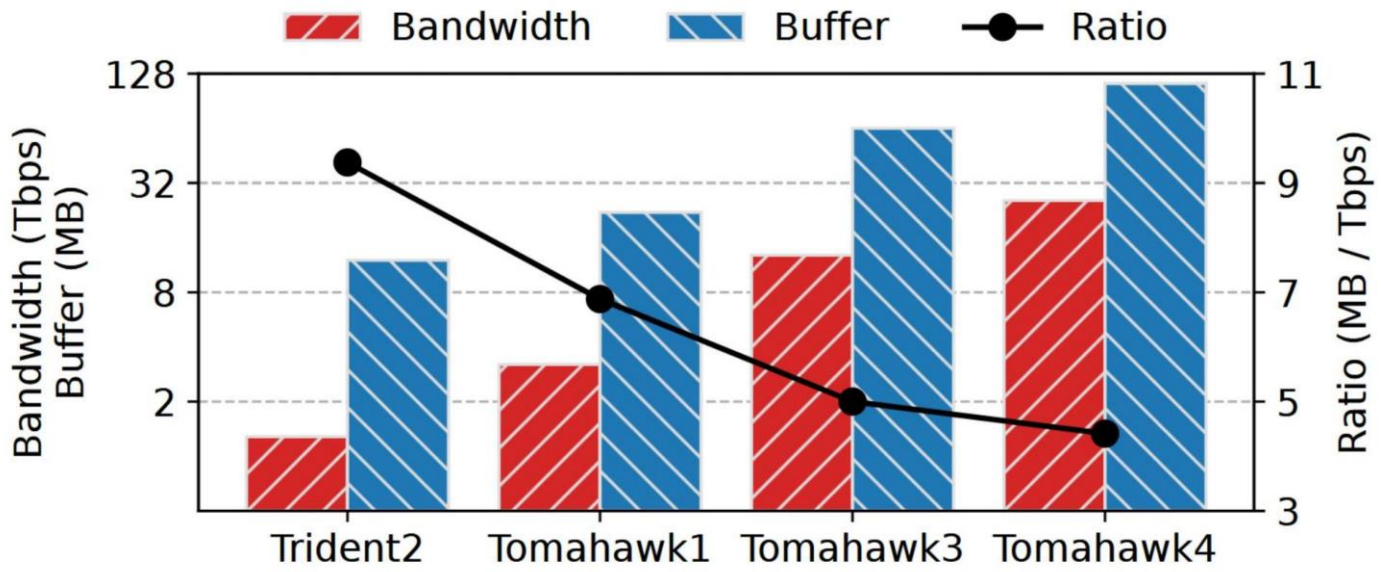
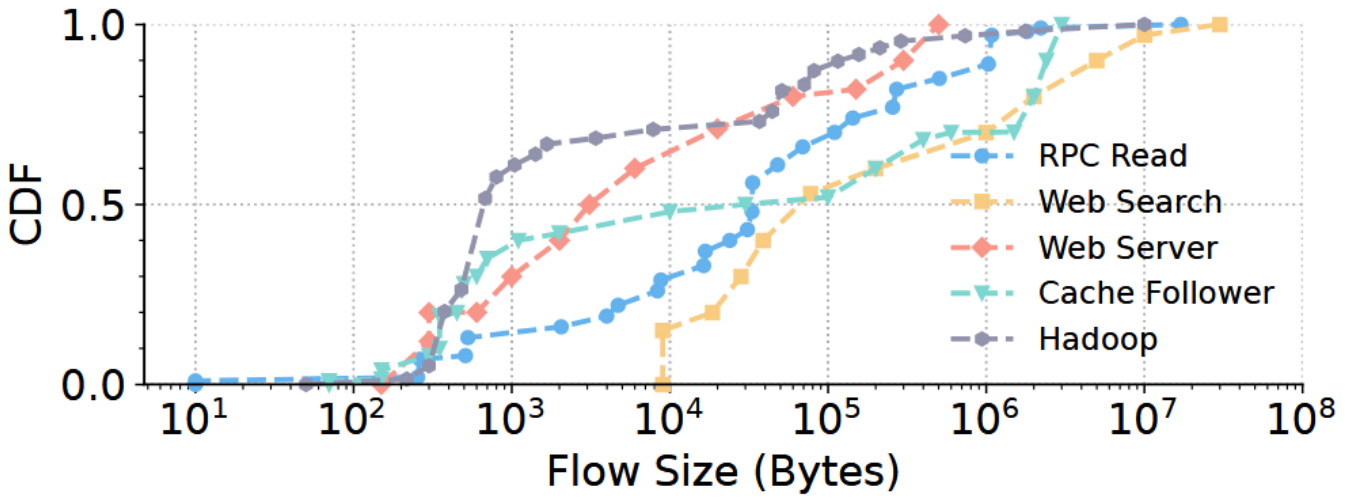


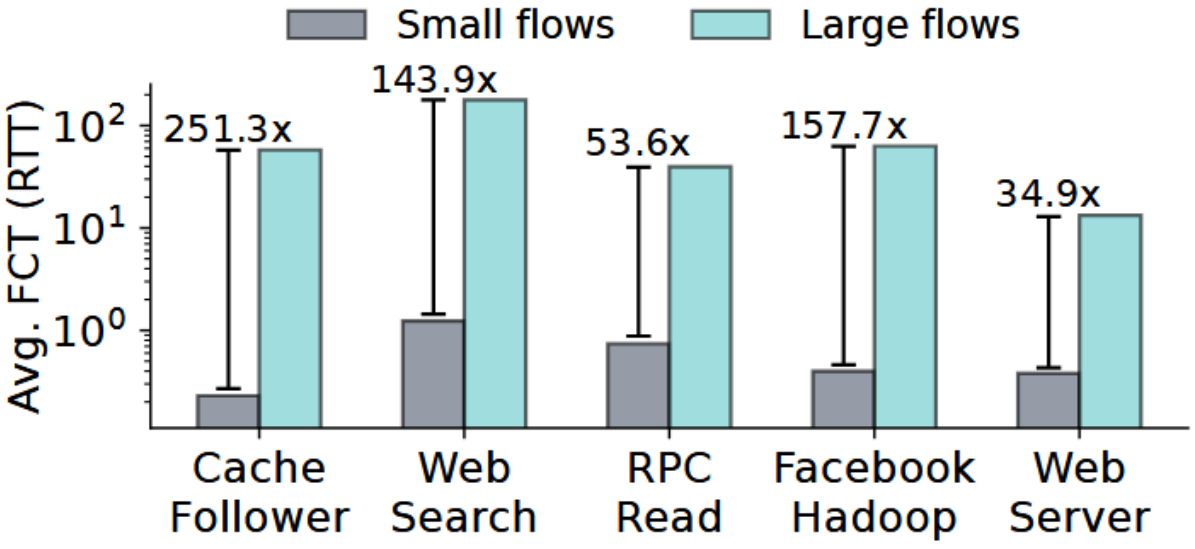
Figure source: PrioPlus@EuroSys '25

Background

□ 数据中心网络低延迟和高带宽的背景

- 在大部分数据中心网络请求中，都存在大小流混杂的现象
- 由于小流的平均流完成时间(Flow Completion Time, FCT, 一个流从开始传输到结束的时间) 很短，若与大流等待相同时间，会更加影响用户体验（时延敏感）





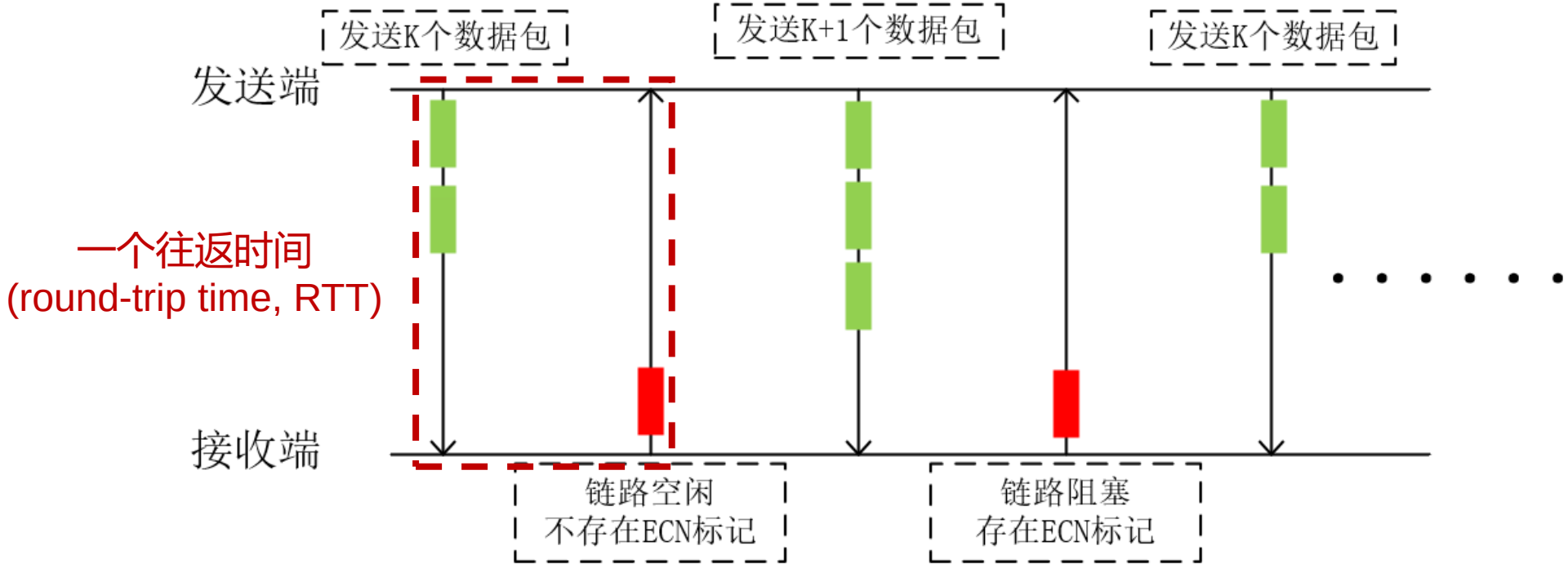
Background

- 现有的流量控制方式：
  - 大部分流量控制机制将大小流控制交织在一起，使用相同的控制方式，没有顾及到大小流各自的特性
  - 部分大小流分开控制的方式，如：
    - Hedera(NSDI'10) 依赖复杂的控制器来计算和部署路径，不满足低延迟需求。
    - Andromeda(NSDI'18) 需要硬件支持，泛化性不足，且只考虑了调度优先，没有考虑流量控制的问题。

- 构建一套大小流分开控制，且不需要额外硬件支持，并具有一定实时性的流量控制算法
  - 控制目标：保护小流高优先级低延迟的同时，让大流充分利用剩余带宽。

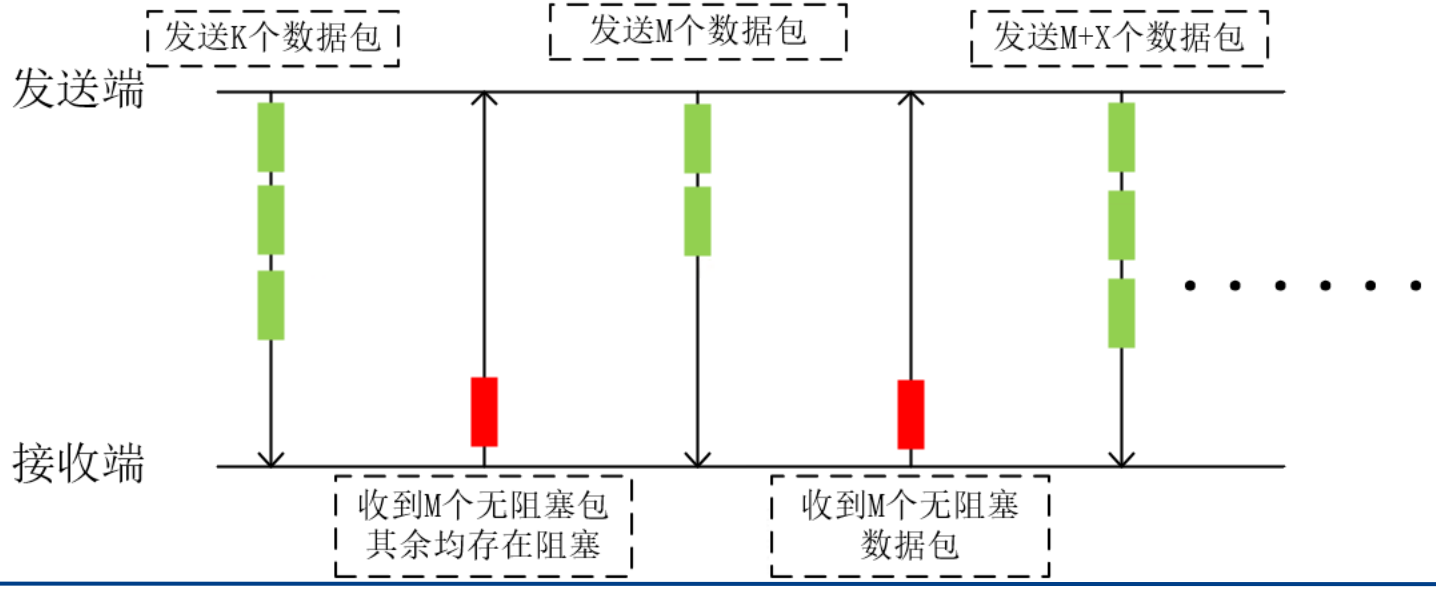
Background

- 反应式传输：
  - 如：DCTCP(SIGCOMM'10), DCQCN(SIGCOMM'15), HPCC(SIGCOMM'19)
  - 由发送端驱动，通过ECN等多种拥塞信号调整发送窗口大小，实现较小的延迟。
  - 缺点：需要多个RTT才能确定合适的窗口大小，无法充分利用带宽（不适用于大流）



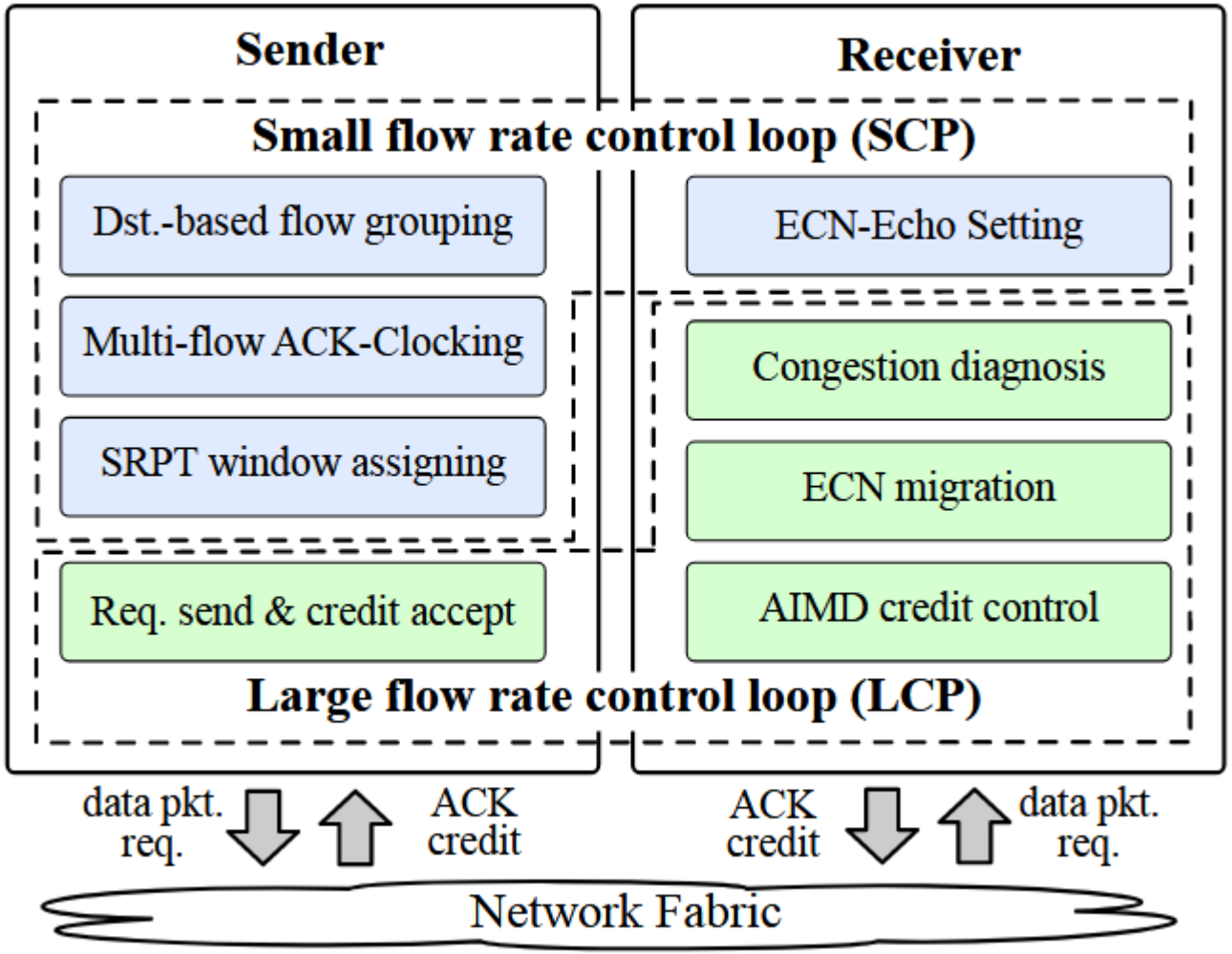
Background

- 主动传输：
  - 如：ExpressPass(SIGCOMM'17), Homa(SIGCOMM'18), dcPIM(SIGCOMM'22)
  - 由接收方分配瓶颈链路带宽作为信用，使发送端以最佳速率发送流量，实现高带宽。
  - 缺点：起始时发送端盲目发送数据包，等待一个 RTT 后才能接收到接收端分配的带宽信用，存在额外时延和丢包的可能（不适用于小流）



Design —— 总体结构

- 小流控制环路(SCP)
  - 基于ECN调整发送窗的反应式传输
  - 流组聚合及逐ACK更新策略
  - SRPT窗口资源分配策略
- 大流控制环路(LCP)
  - 基于接收方给予credit的主动传输
  - 拥塞诊断与ECN迁移
  - AIMD信用控制方法



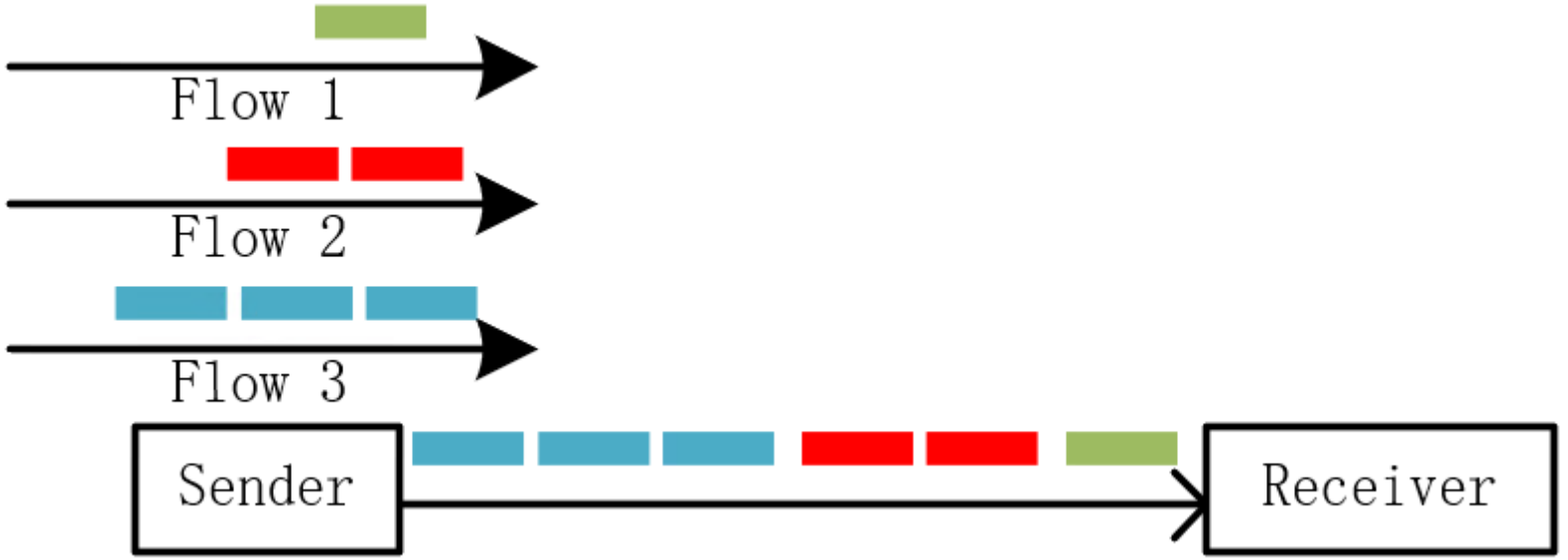
Design —— 小流控制环路(SCP)

- 小流控制环路(SCP)：流组聚合及逐ACK更新策略
  - 反应式传输缺点：需要多个RTT才能确定合适的窗口大小，而小流的流完成时间FCT很小
  - 1. 流组聚合：将一段时间内发往同一目的地址的流作为一个流组，共享发送窗口，避免每个流 warm-up 阶段的带宽占用不足
  - 2. 逐包ACK更新：逐RTT更新需要等待一个窗口包全部收齐后才发送ACK更新窗口大小，更新频率低，为了更精细化地控制，使用逐包ACK进行控制

ECN状态	00(无拥塞)	10(拥塞缓解)	01(拥塞开始)	11(正在拥塞)
窗口状态	$\alpha = 0, cwnd \mathrel{+}= 1$	$\alpha = 0, cwnd \mathrel{+}= 1$	$\alpha = 1, cwnd \mathrel{-}= 1$	$\alpha{+}{+}, cwnd \mathrel{-}= \alpha$

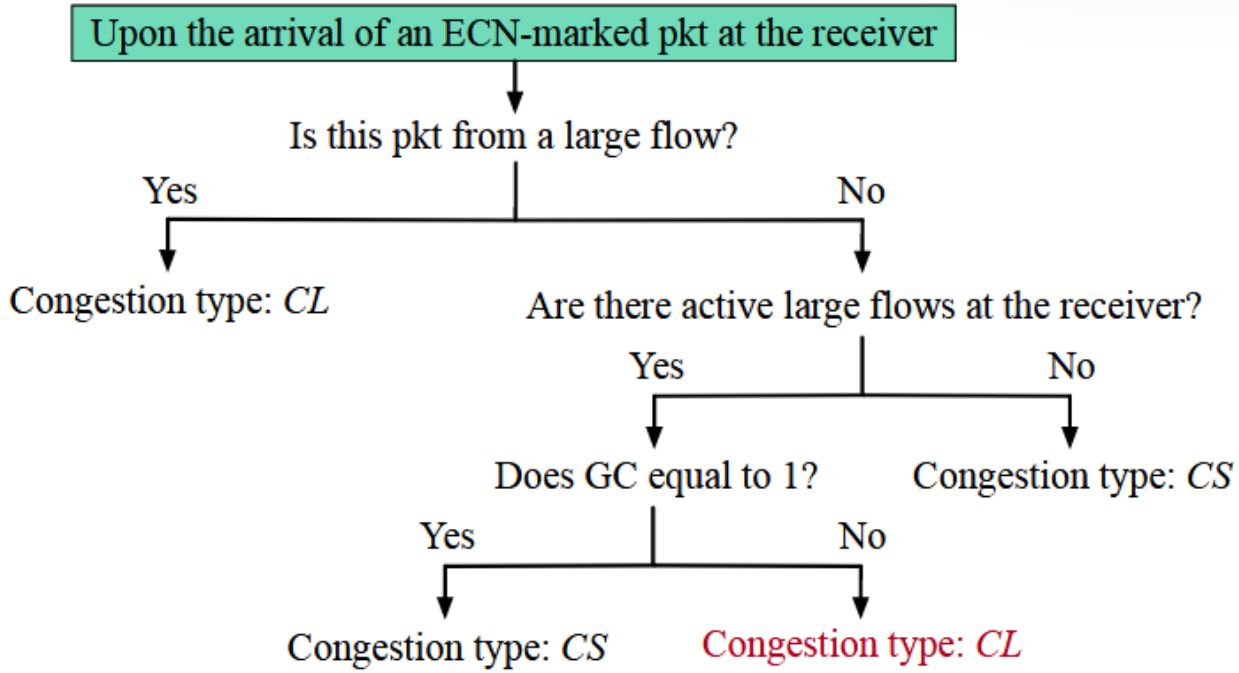
Design —— 小流控制环路(SCP)

- 小流控制环路(SCP)：SRPT窗口资源分配策略
  - 针对每个流组实行SRPT(shortest remaining processing time first) 窗口资源分配方式，发送剩余数据最少的流



Design —— 大流控制环路(LCP)

- 大流控制环路(LCP)：拥塞诊断与ECN迁移
  - 1. 拥塞诊断：区分是由于大流引起的拥塞，还是由小流引起的拥塞
  - 2. 主动传输缺点：起始时发送端盲目发送数据包造成丢包以及阻塞 ==> ECN迁移：通过小流的数据包判断拥塞情况，不需要盲目发送数据包



CL: 拥塞由大流引起

CS: 拥塞由小流引起

GC: 大流所分配到的信用额

Design —— 大流控制环路(LCP)

- 大流控制环路(LCP)：AIMD信用控制方法
  - AIMD(additive increase & multiplicative decrease) 加性增加，乘性减少策略。经过一个大流RTT时间后，若都没收到CL大流阻塞ECN，那么大流信用GC + 1，如果一个大流RTT时间内均是CL大流阻塞ECN，那么大流信用GC减半

$$GC = \begin{cases} GC + 1/GC & \text{if no CL event is detected} \\ GC - 1/2 & \text{if CL event is detected} \end{cases}$$

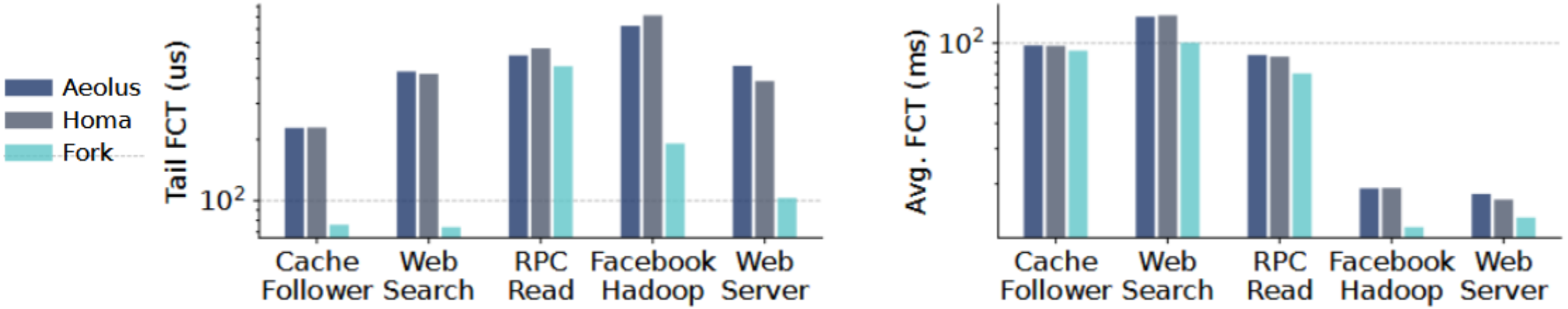
Evaluation —— 实验设置

- 小规模物理实验
  - Testbed：8台服务器连接一台交换机，交换机54 个端口共享22MB缓存，链路速率为100Gbps
  - Workload：五种真实DCN负载：Web Server, Cache Follower, Web Search, Facebook Hadoop, and RPC Read。在60%负载下进行All-to-All流量分发
  - Comparison：
    - Homa(SIGCOMM'18)：小流优先调度
    - Aeolus(SIGCOMM'20) + Homa：受信用数据包优先，同时让盲目数据包充分利用空闲带宽

Evaluation —— 小规模物理实验结果

- 流完成时间(Flow Completion Time, FCT) 代表一个流从开始到结束的时间
- 尾流完成时间(Tail FCT) 代表一系列流中，最晚到达的流完成时间 [延迟]
- 平均流完成时间(avg. FCT) 代表一系列流中，所有流完成时间的平均值 [带宽]

$$\frac{\text{流量大小}}{\text{平均流完成时间}} \approx \text{带宽占用量}$$



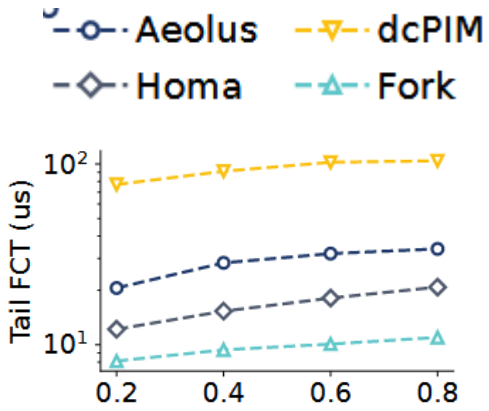
(b) Tail FCT of ≤100KB flows　　(c) Avg. FCT of >100KB flows

60%负载下：小流尾FCT最短+大流平均FCT降低

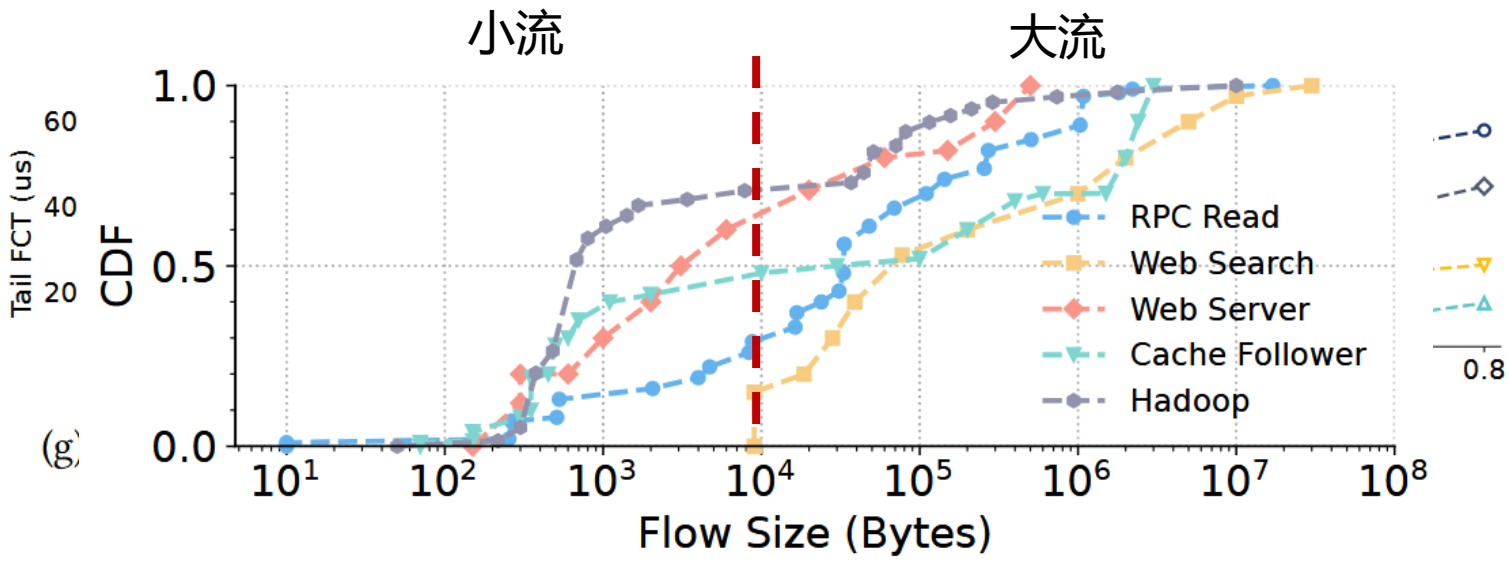
Evaluation —— 实验设置

- 大规模仿真实验
  - Testbed： 8个叶交换机+8个脊交换机和64台服务器的叶脊clos架构，链路均为100Gbps，交换机每个端口360KB缓存
  - Workload：与小规模实验一致，测试了20%~80%负载下的实验结果
  - Comparison：
    - Homa(SIGCOMM'18)：小流优先调度
    - Aeolus(SIGCOMM'20)+Homa：受信用数据包优先，同时让盲目数据包充分利用空闲带宽
    - dcPIM(SIGCOMM'22)：将输入输出端口进行二分匹配，从而实现最佳的带宽利用

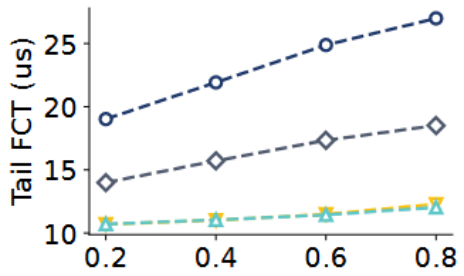
Evaluation —— 大规模仿真实验结果



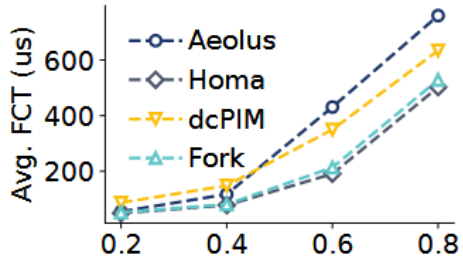
(f) Web Server: tail



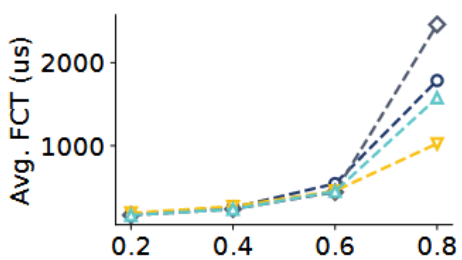
(g)



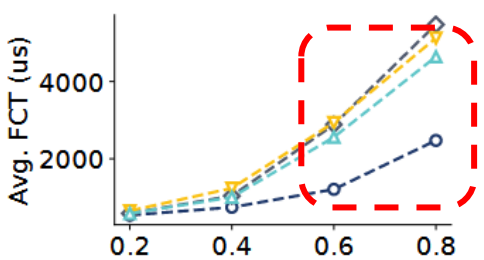
(j) RPC Read: tail



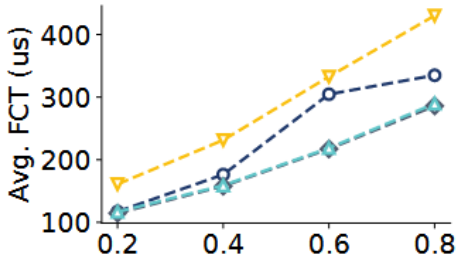
(a) Web Server: avg.



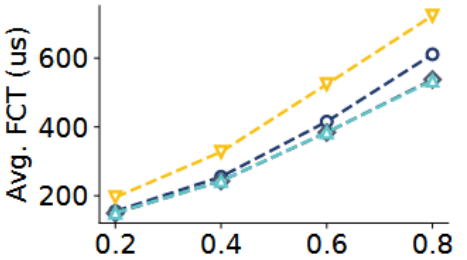
(b) Cache Follower: avg.



(c) Web Search: avg.



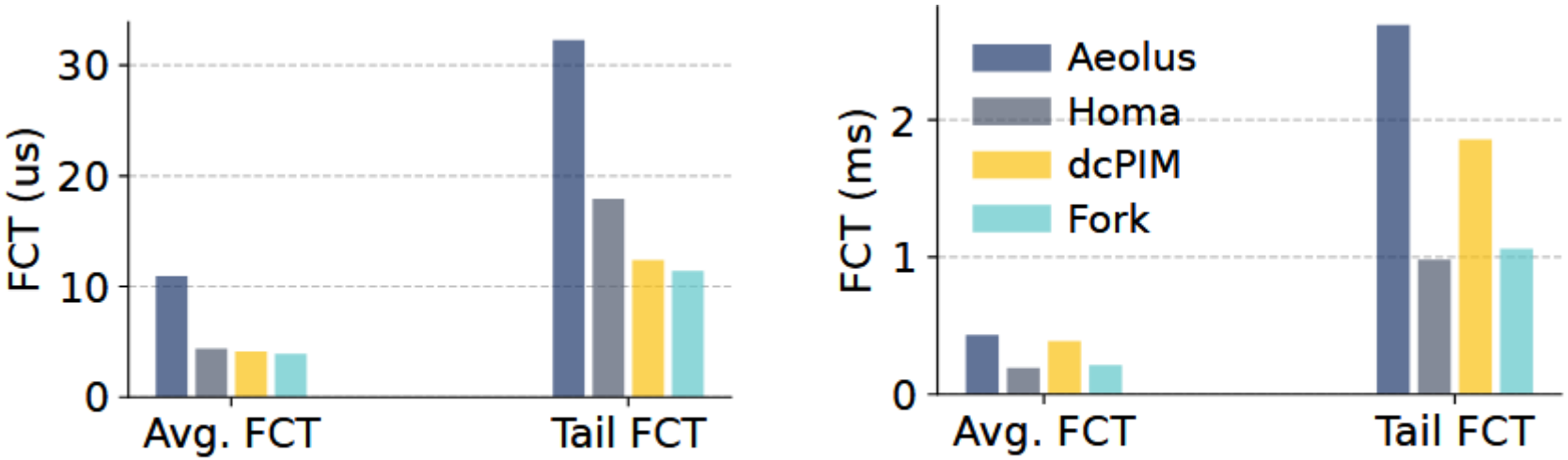
(d) Hadoop: avg.



(e) RPC Read: avg.

Web Search 负载下小流较少，大流带宽占用收敛慢

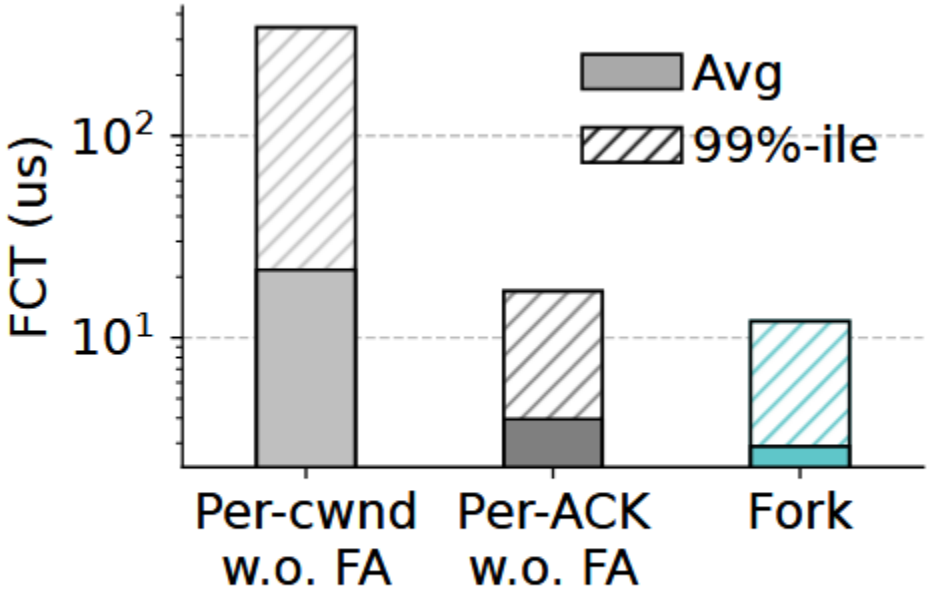
Evaluation —— 大规模仿真实验结果



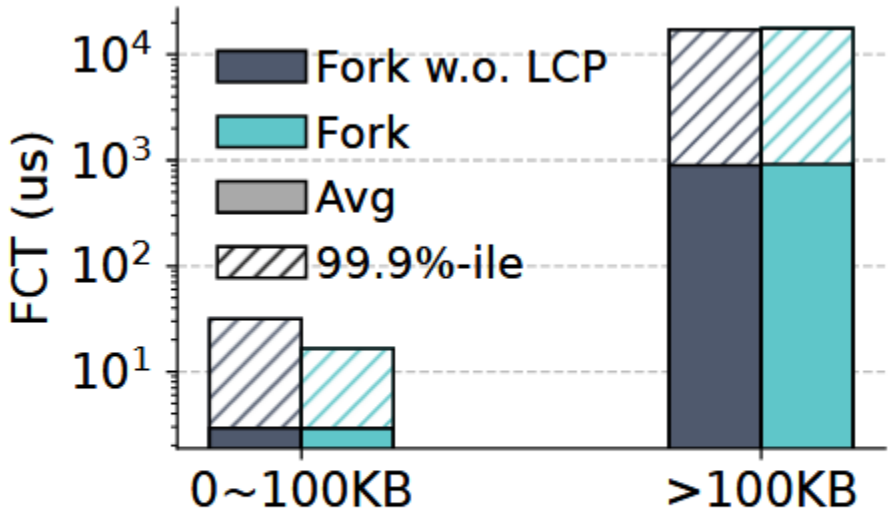
(a) FCT(≤100KB)　　(b) FCT(>100KB)

32-1 Incast场景下：小流尾FCT最短+大流平均FCT降低

Evaluation —— 大规模仿真实验结果



(a) Effect of SCP.



(b) Effect of LCP.

消融实验：小流控制环路和大流控制环路都有很好的效果

Conclusion

- 总结：本文将小流与大流的控制解耦，在保护小流传输的同时，让大流利用网络中的剩余带宽传输，减小了 [时延敏感的] 小流的尾FCT，并且不影响 [带宽敏感的] 大流传输。最重要的是，Fork 不需要对网络结构做任何更改，易于部署。
- Thinking:
  - 有没有可以改进的地方
    - 如何区分大小流：没有一个合适的阈值可以静态划分不同工作负载下的大小流，可以对往期流量进行总结并动态划分
  - 能否应用到我们的工作中、能否泛化
    - 在超节点网络中，路径复杂，等价路径组很多，可以将控制解耦的思想用在对每个端口的控制中，从原先发送端到接收端的控制变为端口到端口的控制

Q & A

Presenter: Huang Kai

2025.12.8