Receiver-driven Congestion Control for Content Oriented Application with Multiple Sources

Y. Hayamizu M. Yamamoto

Kansai University, Japan

Content-Oriented Application

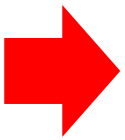
a user does not care about "**where**" a content is obtained

P2P system

- Content file (chunks) is downloaded from anywhere
- Many-to-one communication style

one TCP session is set up for each pair of a sender and a receiver

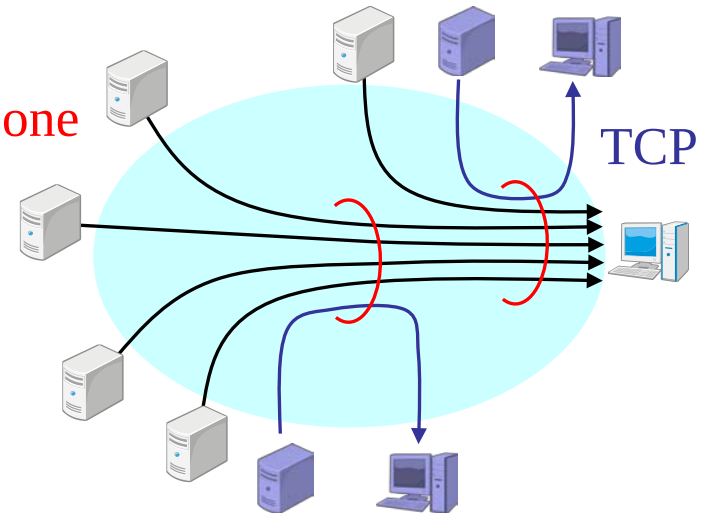
If these sessions go through the same bottleneck link



many-to-one communication might obtain unfairly large throughput

Aim of our proposal

To achieve fairness
with conventional TCP



We propose a **Receiver-driven Congestion Control** for many-to-one communications

- Window coupling in multipath TCP [1]
- Detecting flows sharing the same bottleneck-link Delay-based [2]
- Flow clustering algorithm in Flowmate [3]

[1] C. Raiciu, D. Wischik and M. Handley, “Practical Congestion Control for Multipath Transport Protocols,” UCL Technical Report, 2009.
[2] D. Rubenstein, J. Kurose and D. Towsley, “Detecting shared congestion of flows via end-to-end measurement,” IEEE/ACM Transactions on Networking, vol. 10, no. 3, pp. 381-395, Jun. 2002.
[3] O. Younis and S. Fahmy, “Flowmate: Scalable on-line flow clustering,” IEEE/ACM Transactions on Networking, vol. 13, no. 2, pp. 288-301, Apr. 2005.

Design Goal

a. Take a Receiver-driven approach.

b. Clustering bottleneck-sharing flows and window sizes of these flows are controlled in a coupled manner.

c. Window size of a flow sharing no bottleneck link with others is controlled independently.

d. Among flows in a cluster, bandwidth of a bottleneck link is effectively managed by Resource Pooling [1] policy.

e. For clustered flows, “Linked Increase” policy, i.e. only window increase phase is coupled, is applied.

[1] D. Wischik, M. Handley and M. Bagnulo, “The Resource Pooling Principle,” ACM Computer Communications Review, Vol.38, No.5,pp.47-52, Oct. 2008.

Design Goal

a. Take a Receiver-driven approach.

b. Clustering bottleneck-sharing flows and window sizes of these flows are controlled in a coupled manner.

c. Window size of a flow sharing no bottleneck link with others is controlled independently.

d. Among flows in a cluster, bandwidth of a bottleneck link is effectively managed by Resource Pooling [1] policy.

e. For clustered flows, “Linked Increase” policy, i.e. only window increase phase is coupled, is applied.

[1] D. Wischik, M. Handley and M. Bagnulo, “The Resource Pooling Principle,” ACM Computer Communications Review, Vol.38, No.5,pp.47-52, Oct. 2008.

The Reason for Receiver-driven

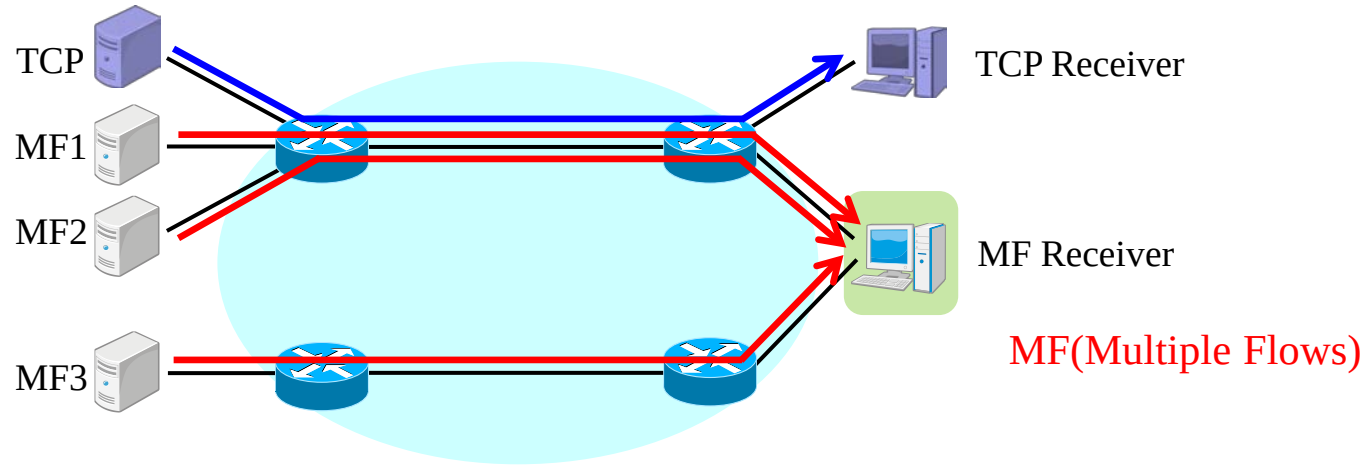
In multiple-senders and one-receiver (many-to-one) communications, Multipath TCP can not be easily applied

It is not easy for a sender to understand other senders' situation

A **receiver** can learn all the senders' status

- Receiver notifies its calculated window size to the senders by advertised window
- Sender transmission rate is controlled by advertised window

hard to understand other senders' situation



Design Goal

a. Take a Receiver-driven approach.

b. Clustering bottleneck-sharing flows and window sizes of these flows are controlled in a coupled manner.

c. Window size of a flow sharing no bottleneck link with others is controlled independently.

d. Among flows in a cluster, bandwidth of a bottleneck link is effectively managed by Resource Pooling [1] policy.

e. For clustered flows, “Linked Increase” policy, i.e. only window increase phase is coupled, is applied.

[1] D. Wischik, M. Handley and M. Bagnulo, “The Resource Pooling Principle,” ACM Computer Communications Review, Vol.38, No.5,pp.47-52, Oct. 2008.

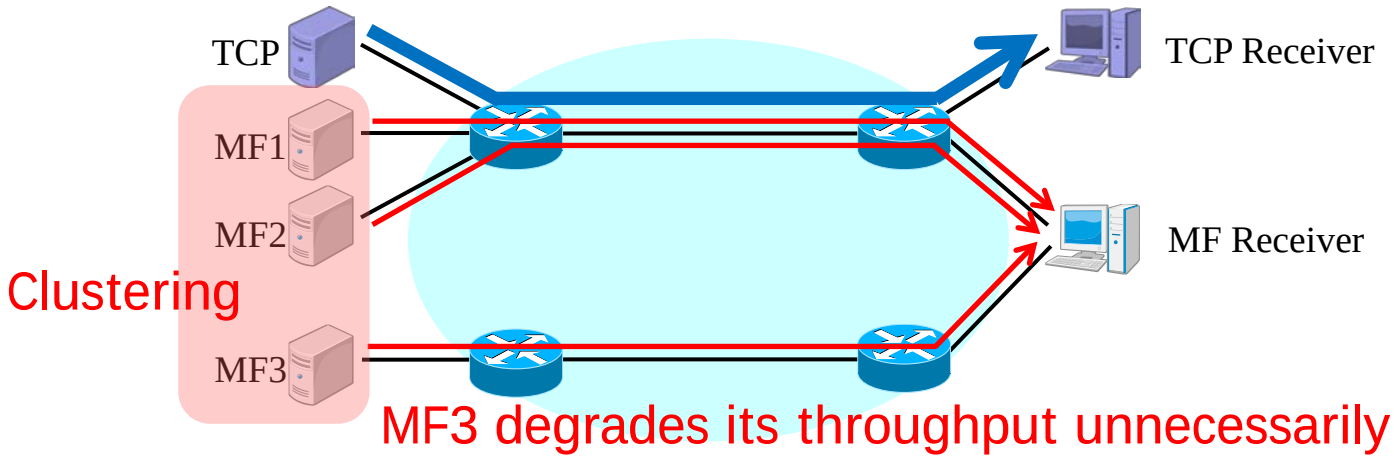
Clustering of flows

Clustering policy

1. Not degrade other competing TCP throughput
2. Prevent an unnecessarily large cluster

Policy 1 gives comprehensive control to clustered flows

With too large cluster, congestion at other path(s) may degrade throughput of a non-congested path

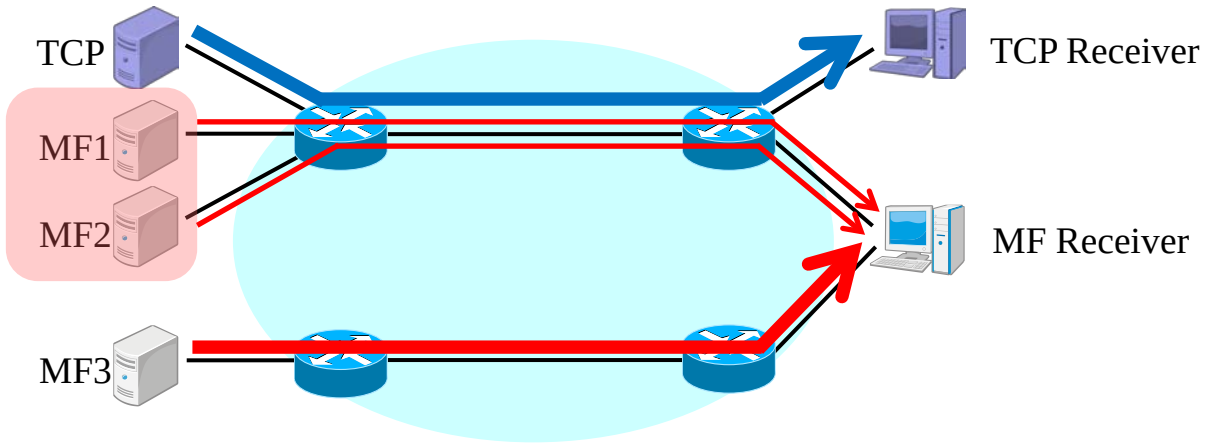


Clustering of flows

Clustering policy

1. Not degrade other competing TCP throughput
2. Prevent an unnecessarily large cluster

Window size of a flow sharing no bottleneck link with others is controlled independently



Design Goal

a. Take a Receiver-driven approach.

b. Clustering bottleneck-sharing flows and window sizes of these flows are controlled in a coupled manner.

c. Window size of a flow sharing no bottleneck link with others is controlled independently.

d. Among flows in a cluster, bandwidth of a bottleneck link is effectively managed by Resource Pooling [1] policy.

e. For clustered flows, “Linked Increase” policy, i.e. only window increase phase is coupled, is applied.

[1] D. Wischik, M. Handley and M. Bagnulo, “The Resource Pooling Principle,” ACM Computer Communications Review, Vol.38, No.5,pp.47-52, Oct. 2008.

Resource Pooling in a cluster

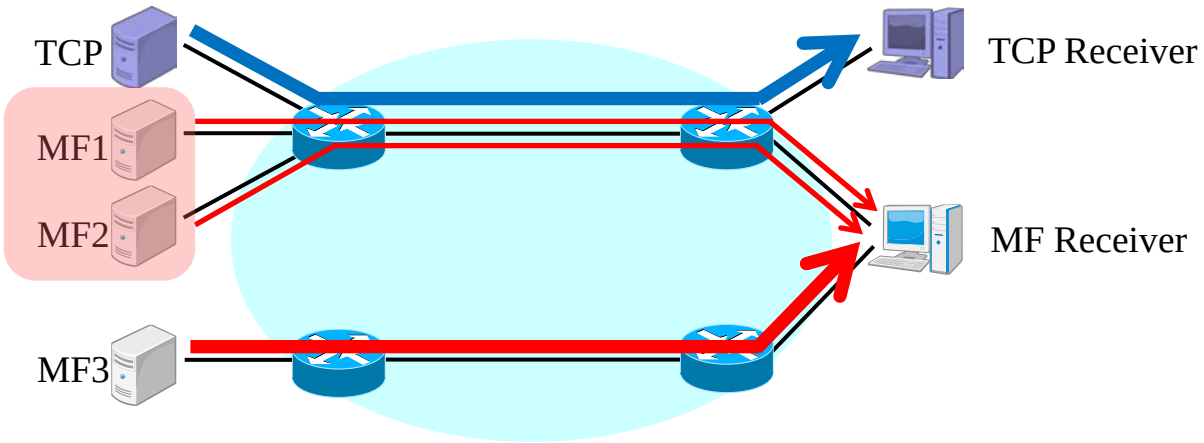
In many-to-one communications, each flow may have quite different RTT and packet loss rate

It is desirable to pool resources flexibly according to the situation

We adopt Multipath TCP [1] algorithm which realizes

Resource Pooling [2] "making a collection of resources behave like a single pooled resource"

allocate bandwidth flexibly



[1] C. Raiciu, D. Wischik and M. Handley, "Practical Congestion Control for Multipath Transport Protocols," UCL Technical Report, 2009.
[2] D. Wischik, M. Handley and M. Bagnulo, "The Resource Pooling Principle," ACM Computer Communications Review, Vol.38, No.5,pp.47-52, Oct. 2008.

Window control of our proposal

Window increase phase

- When a receiver receives a data packet of flow r ,

  w_r^{ad} is increased by $\min(\frac{a}{w^{ad}}, \frac{1}{w_r^{ad}})$ ➡ **Windows are coupled**

$$a = w^{ad} \left(\frac{\max_r \sqrt{w_r^{ad} / RTT_r}}{\sum_r w_r^{ad} / RTT_r}\right)^2$$

Senders notify this value with TCP-header field

w_r^{ad} : advertised window size of flow r

w^{ad} : total window size of clustered flows

Window decrease phase

- When a packet is lost in flow r, w_r^{ad} is decreased by $w_r^{ad} / 2$

  ➡ **Windows are <u>not</u> coupled**

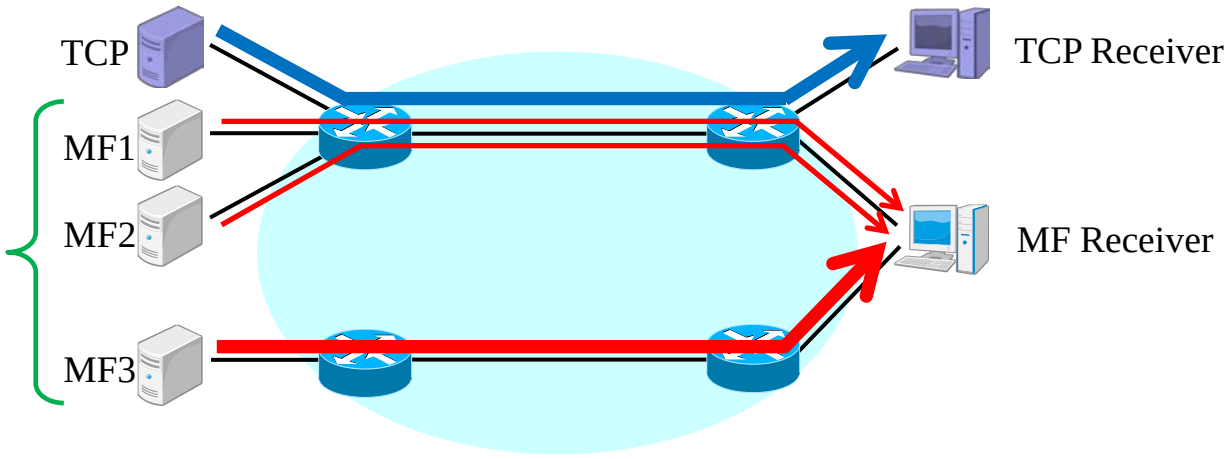
- When a timeout occurred, only flow r 's window size is decreased

Clustering Algorithm

We want to cluster flows which go through a same bottleneck link and achieve fairness with a TCP flow per bottleneck link

Need for clustering bottleneck sharing flows

A receiver must identify whether these flows go through a same bottleneck link or not



Clustering Algorithm

We want to cluster flows which go through a same bottleneck link and achieve fairness with a TCP flow per bottleneck link

Need for clustering bottleneck sharing flows

Clustering Algorithm [1]

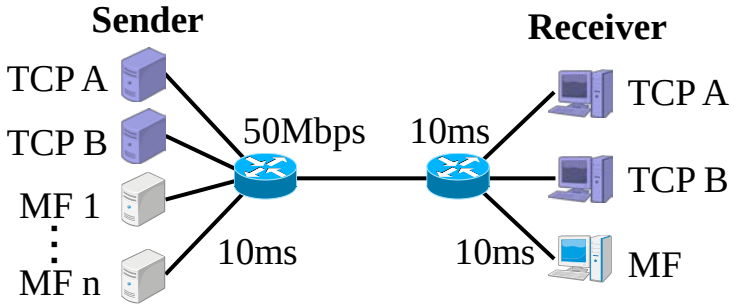
- utilize correlation among packet loss timings and also packet delays
  1. Calculates cross-correlation function M_x of 2 flows' RTT in each interval $t > 0$
  2. Calculates auto-correlation function M_a of each flow's RTT in each interval $T > t$
  3. When $M_x > M_a$ these 2 flows is detected as bottleneck-link shared

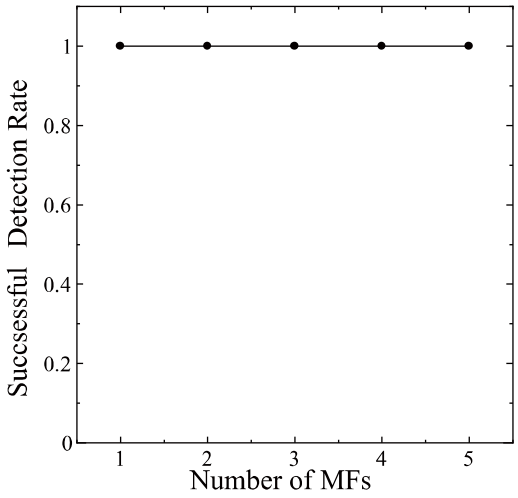
[1] D. Rubenstein, J. Kurose and D. Towsley, "Detecting shared congestion of flows via end-to-end measurement," IEEE/ACM Transactions on Networking, vol. 10, no. 3, pp. 381-395, Jun. 2002.

Preliminary Simulation

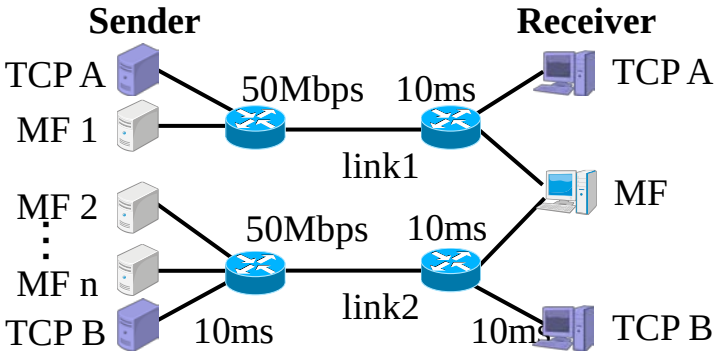
$$\text{Successful Detection Rate} = \frac{\#\ of\ successfully\ detected\ flows}{\#\ of\ total\ flows\ in\ recerver\text{-}driven\ CC}$$

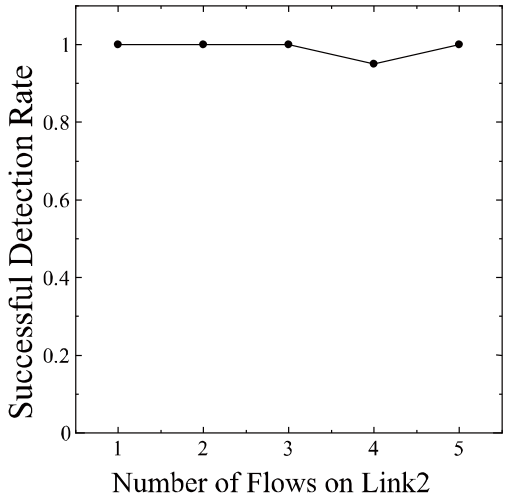
One-bottleneck sharing model





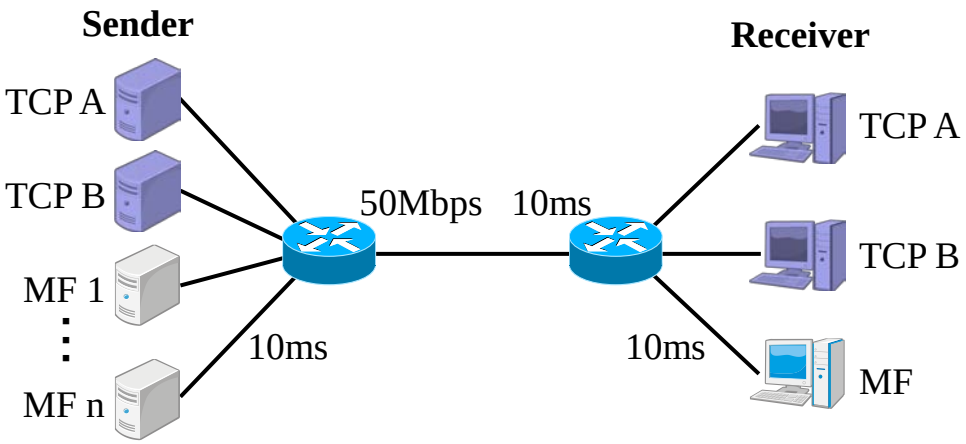
Branching model

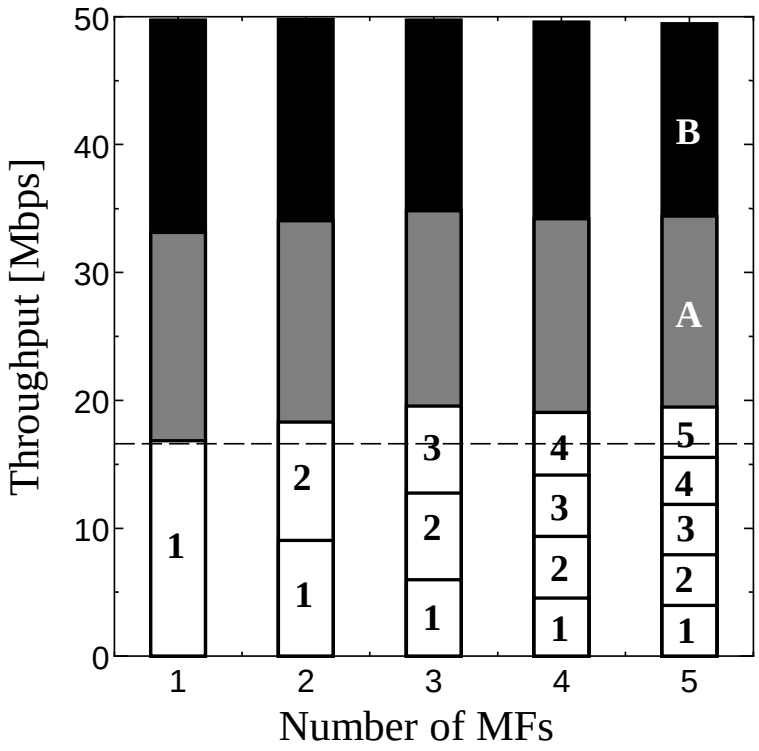




Simulation Result 1

One-bottleneck sharing model



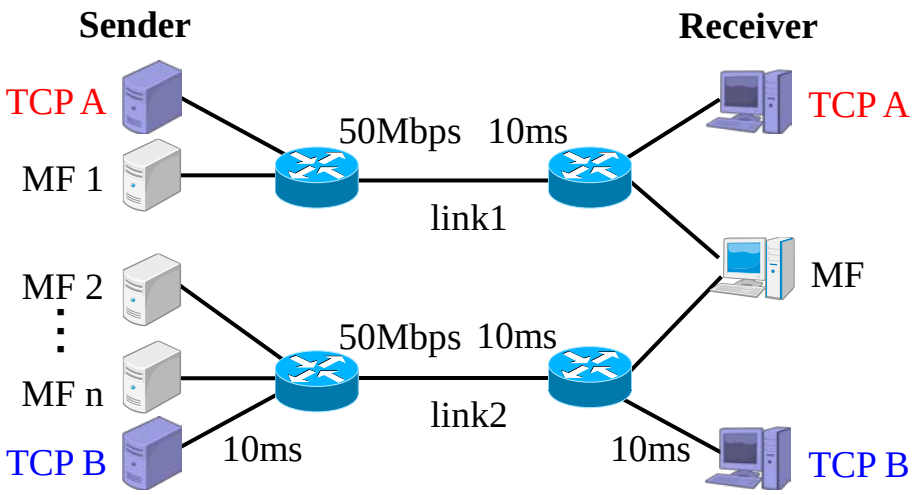


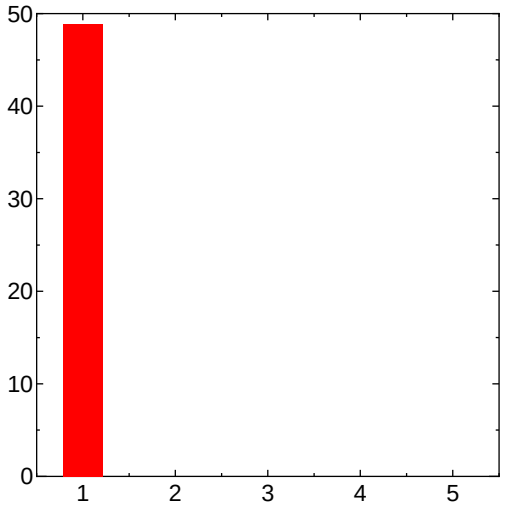
Our proposal controls multiple flows(MF) as a single cluster

aggregated throughput of MF flows(sum of white bars) is almost equal to the ideal value

Simulation Result 2

Branching model





Conclusions

- Proposal of receiver-driven congestion control for many-to-one content oriented applications
- Our proposed congestion control successfully detects bottleneck-sharing flows
- The coupled window control mechanism makes these flows share bottleneck link fairly with regular TCP sessions