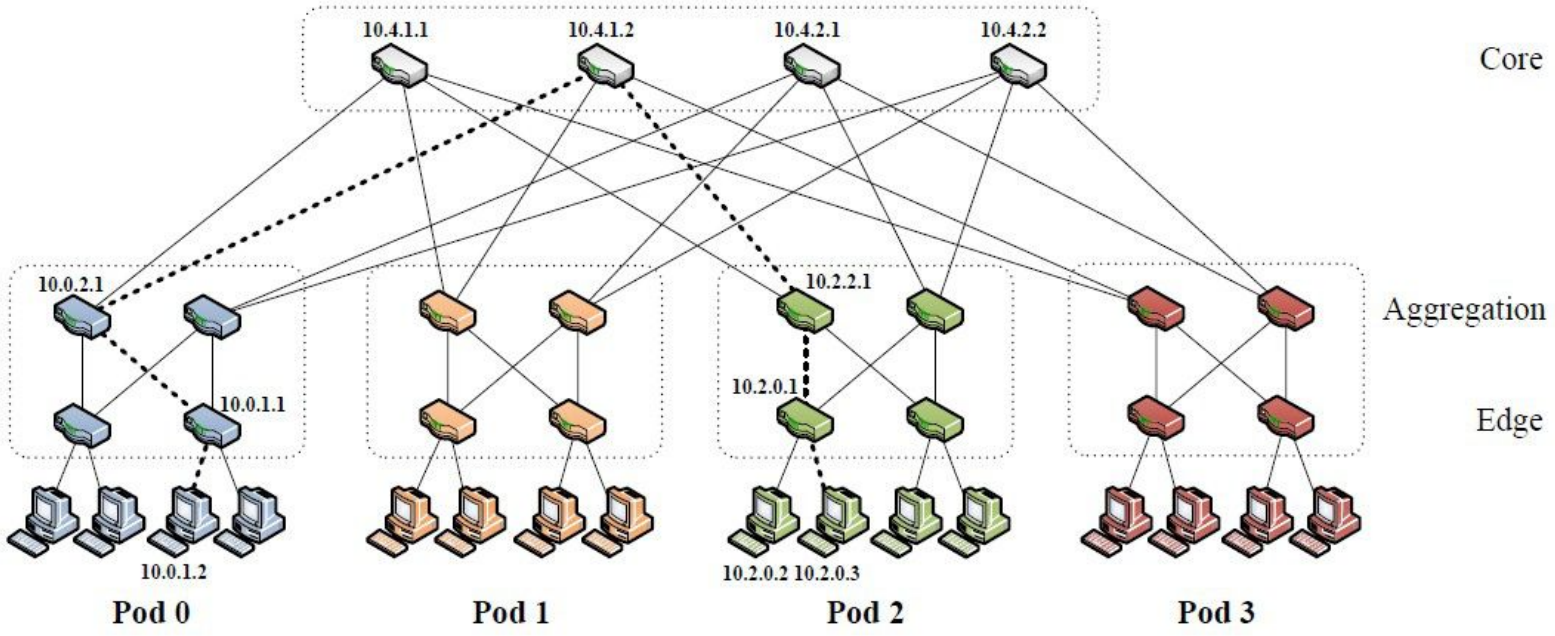
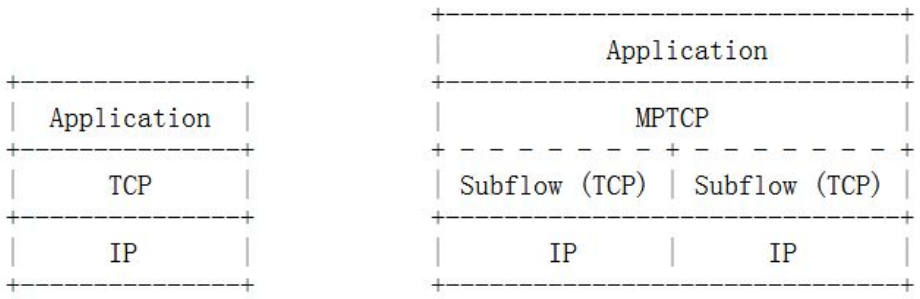
Multipath TCP and its Application in Datacenter Network

Datacenter Network

- Good network condition. Short RTT. Homogeneous topology.
- Provisioned with redundant paths.
- Small flows are latency sensitive, large flows want more throughput.
- How to efficiently utilize the network resources of the datacenter network.



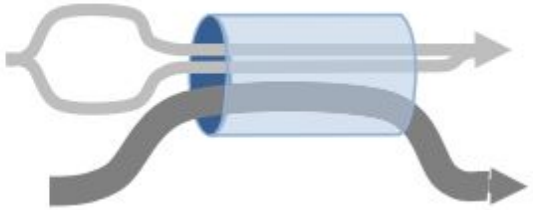
Multipath TCP



- Extend a single TCP connection into several subflow connection.
- Why use MPTCP: more multihomed hosts (host with multiple NICs or multiple IP addresses), network infrastructure is provisioned with multiple path betwen two hosts.
- Problems of designing MPTCP: congestion control, sequential number, middlebox interference.

MPTCP Congestion Control

- The AIMD congestion control method of traditional TCP:
  - Each ACK, increase the congestion window *w* by 1/*w*, resulting in an increase of one packet per RTT.
  - Each loss, decrease *w* by *w*/2.
- Can't use AIMD for MPTCP.
- AIMD leads to the fair sharing of all the flows using the same link. If two subflows of a MPTCP connection go through the same link, this MPTCP flow will get twice the transmission rate of a single TCP flow which shares that link with the MPTCP flow.
- What we need is something like this:



MPTCP Congestion Control

ALGORITHM

- Each ACK on subflow r, increase the window w_r by $\min(a/w_{\text{total}}, 1/w_r)$.
- Each loss on subflow r, decrease the window w_r by $w_r/2$.

Here

$$a = \hat{w}_{\text{total}} \frac{\max_r \hat{w}_r/\text{RTT}_r^2}{(\sum_r \hat{w}_r/\text{RTT}_r)^2}, \tag{5}$$

w_r is the current window size on path r and $\hat{w}_r$ is the equilibrium window size on path r, and similarly for w_{total} and $\hat{w}_{\text{total}}$.

MPTCP Congestion Control

- A multipath flow should give a connection at least as much throughput as it would get with single-path TCP on the best of its paths. (Incentive for deploying MPTCP)

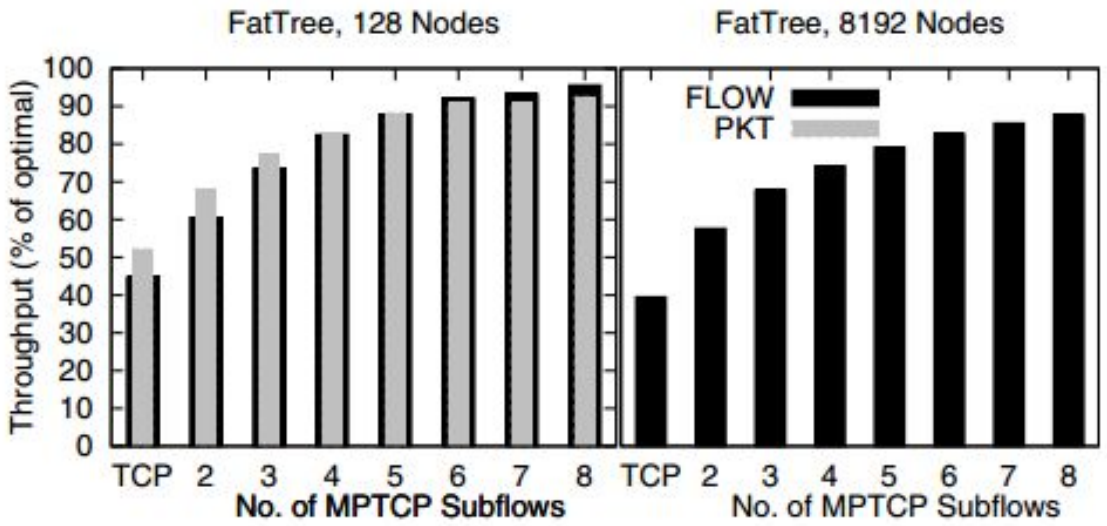
$$\sum_{r \in R} \frac{\hat{w}_r}{RTT_r} \geq \max_{r \in R} \frac{\hat{w}_r^{TCP}}{RTT_r}$$

- A multipath flow should take no more capacity on any path or collection of paths than if it was a single path TCP flow using the best of those paths.

$$\sum_{r \in S} \frac{\hat{w}_r}{RTT_r} \leq \max_{r \in S} \frac{\hat{w}_r^{TCP}}{RTT_r}, \textit{for all } S \in R$$

MPTCP in Datacenter

- Datacenter has redundant paths, MPTCP is an efficient load balancer.
- Open up several subflows, choose the path of each subflow using ECMP, let MPTCP push more data onto less congested paths, no more complicated scheduling.
- The throughput comparison of TCP with MPTCP, using a one to one permutation workload:



XMP

- XMP is an Explicit Multipath Congestion Control Especially for Datacenter Networks.
- Datacenter has a mixture of small flows and large flows.
  - Small flows are latency sensitive.
  - Large flows want more throughput.
  - The RTT is small. Major contribution to RTT is link queue length in switch.
- MPTCP is designed for Internet, it needs to be improved for datacenter network.
- XMP is used to transmit large flows in datacenter. It consists of two parts.
  - Buffer Occupancy Suppression (BOS) algorithm employs the Explicit Congestion Notification (ECN) mechanism to control link buffer occupancy.
  - Traffic Shifting (TraSh) algorithm couples the subflows so as to move the traffic of a multipath TCP flow from its more congested path to less congested path.

Buffer Occupancy Suppression

- Switch typically implements Active Queue Management algorithm, such as Random Early Detection (RED).
- RED performs Exponentially Weighted Moving Average (EWMA) to estimate the average link buffer occupancy.
- According to the link buffer occupancy, switch will mark the arriving packet with Congestion Experienced (CE) codepoint in IP header.
- The destination that receives the CE signal will feed back to the source by marking the ECN Echo (ECE) codepoint in IP header of the acknowledgement packet.
- The source decreases its congestion window in response to the ECE signal and mark the congestion Window Reduced (CWR) codepoint in subsequent packet to stop the destination from feeding back the ECE signal.

The Definition of A Round

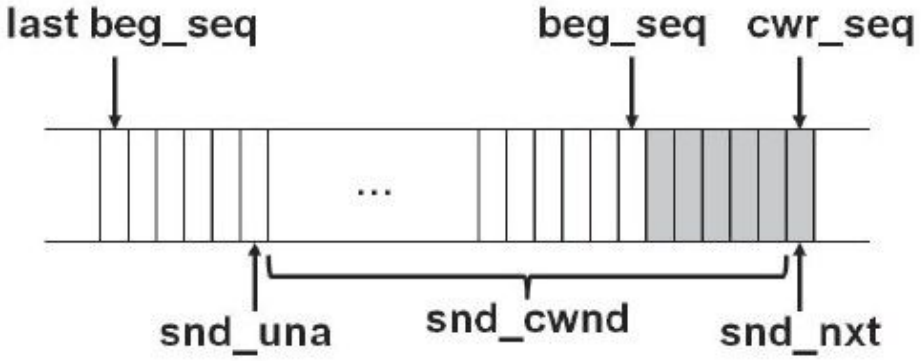


Figure 2: A round. *snd_una* **is the highest unacknowledged sequence number.** *snd_nxt* **is the sequence number of the next packet that will be sent.** *beg_seq* **records a specified packet sequence number. When** $snd\_una > beg\_seq$**, a round ends. And meanwhile,** *beg_seq* **is updated using** *snd_nxt*. *snd_cwnd* **is the congestion window.** *cwr_seq* **is used to avoid undesired reduction in** *snd_cwnd* **when ECN signals are received.**

Buffer Occupancy Suppression

- **Packet Marking Rules**: The switch marks the arriving packet with the CE codepoint if the queue length of the outgoing interface is larger than K packets.
- **ECN Echo**: The destination feeds its received CEs back to the source through the ECE and the CWR codepoint of acknowledgement packets. The two bits can encode at most 3 CEs.
- **Slow Start**: The source increases cwnd by one if receiving an acknowledgement packet whose ECE and CWR are both zero; otherwise, it stops increasing cwnd and enters congestion avoidance.
- **Congestion Avoidance**: The source decreases cwnd by a factor of $1/\beta$ if receiving an acknowledgement packet whose ECE and CWR are not both zero; otherwise, it increases cwnd by δ on the end of a round. The source decreases cwnd only once in a round.

Parameter Estimation (BOS)

- In order to achieve a full link utilization, the marking threshold K should satisfy: $(K + BDP)/\beta \le K$, which means that :

$$K \ge \frac{BDP}{\beta - 1}, \beta \ge 2$$

- The BDP of 1Gbps link and 400μs RTT is 33 packets. If β is too large, then the cwnd is decreased too much when encoutering congestion, which will increase the convergence time. Thus choose β to be 4 and K to be 10.

Traffic Shifting

- The differential equation of the cwnd evolution:

$$\frac{dw(t)}{t} = \frac{\delta}{T}(1 - p(t)) - \frac{w(t)}{T\beta}p(t)$$

- Equilibrium point:

$$\tilde{p} = \frac{1}{1 + \tilde{w}/\delta\beta}$$

- Utility function:

$$U(x) = \frac{\delta\beta}{T}\log(1 + \frac{T}{\delta\beta}x)$$

Traffic Shifting

- The network utility maximization model is used to couple subflows.

$$\max_{X \geq 0} \sum_{s \in S} U(y_s)$$

$$s.t. \;\; Y = BX$$

$$AX \leq C$$

- Based on this model, design a general framework for multipath congestion control.
  - Congestion Metric determines how the source quantifies the congestion extent of the path using congestion signals.
  - Adjustable Parameters are used by the source to control the bandwidth obtained by each flow.
  - Congestion Equality Principle means that if each subflow tries to equalize the congestion extent, network resources will be fairly and efficiently shared by all flows.

Traffic Shifting

- Construct a utility function as follows:

$$U(y_s) = \frac{\beta}{T_s} \log\left(1 + \frac{T_s}{\beta} y_s\right)$$

- where $T_s = \min\{T_{s,r}, r \in R_s\}$ and $T_{s,r}$ is the smoothed RTT of path r measured by flow s.
- The derivative of the utility function can be expected congestion extent of flow s.

$$U'(y_s) = \frac{1}{1 + y_s T_s / \beta}$$

Traffic Shifting

- Subflow r changes its cwnd according to the BOS algorithm with parameter
- $\delta_{s,r}$ and β . So at equilibrium point we have $\tilde{p}_{s,r}$,which can be perceived as the congestion extent of subflow r of flow s.

$$\tilde{p}_{s,r} = \frac{1}{1 + x_{s,r} T_{s,r} / \delta_{s,r} \beta}$$

- A larger $\delta_{s,r}$ can help flow s obtain more bandwidth on path r. A smaller $\delta_{s,r}$ can lead to less bandwidth. $\delta_{s,r}$ indicates how agressively flow s competes for bandwidth with other flows on path r. So $\delta_{s,r}$ is choosed as the parameter to control the bandwidth obtained by each subflow.

Traffic Shifting

- Remember that $\tilde{p}_{s,r}$ is defined as the congestion extent of subflow r of flow s. $U'(y_s)$ is defined as the congestion extent of flow s.
- According to the network utility maximization model, when $\tilde{p}_{s,r} < U'(y_s)$, which means that the congestion on subflow r is less than the congestion of flow s, subflow r should increase its congestion window by increasing $\delta_{s,r}$.
- Conversely, when $\tilde{p}_{s,r} > U'(y_s)$, which means that the congestion on subflow r is greater than the congestion of flow s, subflow r should decrease its congestion window by increasing $\delta_{s,r}$.
- $\delta_{s,r}$ is updated using the following formula:

$$\delta_{s,r} = \frac{T_{s,r} x_{s,r}}{T_s y_s}$$

- The algorithm is operated in a time-slotted fashion. If $\tilde{p}_{s,r} < U'(y_s)$, we are going to have:

$$\delta_{s,r}(t+1) = \frac{T_{s,r} x_{s,r}(t)}{T_s y_s(t)} > \delta_{s,r}(t)$$

Traffic Shifting Algorithm

1) **Initialization**: At step $t = 0$, flow s sets $\delta_{s,r}(0) = 1$ for each $r \in R_s$; go to 2).

2) **Rate Convergence**: At step t, flow s performs the BOS algorithm independently on each subflow until achieving rate convergence, namely $x_{s,r}(t) = \frac{\beta\delta_{s,r}(t)\left(1-p_{s,r}(t)\right)}{T_{s,r}p_{s,r}(t)}$ for each $r \in R_s$; go to 3).

3) **Parameter Adjustment**: At step t, flow s calculates the total rate $y_s(t) = \sum_{r\in R_s} x_{s,r}(t)$ and finds out $T_s = \min\{T_{s,r}, r \in R_s\}$. It then adjusts $\delta_{s,r}$ for each $r \in R_s$ using $\delta_{s,r}(t+1) = \frac{T_{s,r}x_{s,r}(t)}{T_s y_s(t)}$; go to 4).

4) **Iteration**: $t \leftarrow t + 1$; go to 2).

The Pseudo-code of XMP

Algorithm 1: The pseudo-code of XMP

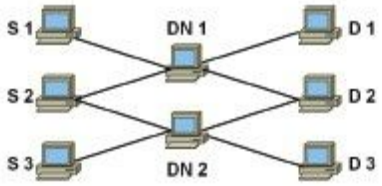
At receiving a new ack **on subflow** r**:**

```
/* perform per-round-operations                                   */
if ack > beg_seq[r] then
    instant_rate[r] ← snd_cwnd[r]/srtt_us[r];
    total_rate ← ∑{instant_rate};
    min_rtt ← min{srtt_us};
    delta[r] ← snd_cwnd[r]/(total_rate × min_rtt);
    if state[r] = NORMAL and
    snd_cwnd[r] > snd_ssthresh[r] then
        /* congestion avoidance                                   */
        adder[r] ← adder[r] + delta[r];
        snd_cwnd[r] ← snd_cwnd[r] + ⌊adder[r]⌋;
        adder[r] ← adder[r] − ⌊adder[r]⌋;
    beg_seq[r] ← snd_nxt[r]; // for next round
/* perform per-ack-operations                                     */
if state[r] = NORMAL and
snd_cwnd[r] ≤ snd_ssthresh[r] then
    snd_cwnd[r] ← snd_cwnd[r] + 1; // slow start
if state[r] ≠ NORMAL and ack ≥ cwr_seq[r] then
    state[r] ← NORMAL;
```

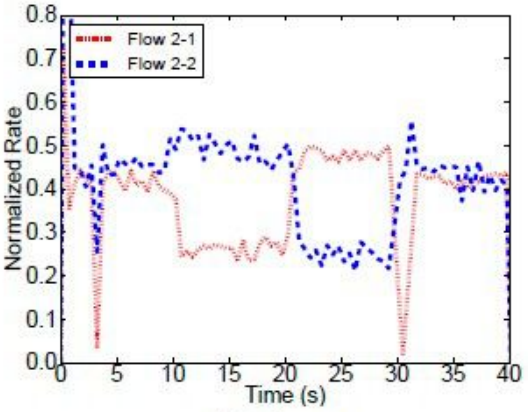
At receiving ECE or CWR on subflow r**:**

```
if state[r] = NORMAL then
    state[r] ← REDUCED;
    cwr_seq[r] ← snd_nxt[r];
    /* reduce the congestion window                               */
    if snd_cwnd[r] > snd_ssthresh[r] then
        tmp ← snd_cwnd[r]/mptcp_xmp_reducer;
        snd_cwnd[r] ← snd_cwnd[r] − max{tmp, 1};
        snd_cwnd[r] ← max{snd_cwnd[r], 2};
    /* avoid re-entering slow start                               */
    snd_ssthresh[r] ← snd_cwnd[r] − 1;
```

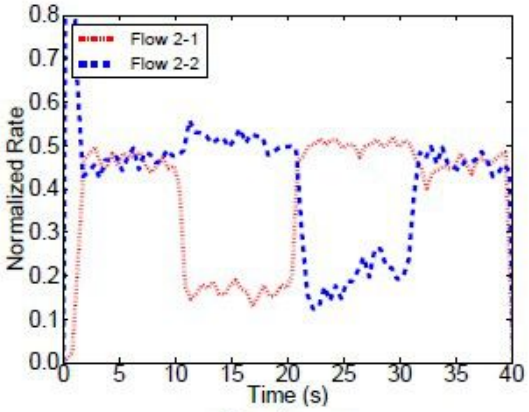
Experiment Result



(a) Traffic Shifting



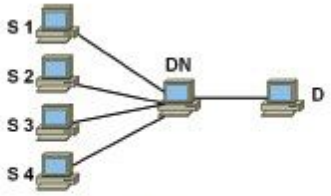
(a) $\beta = 4$



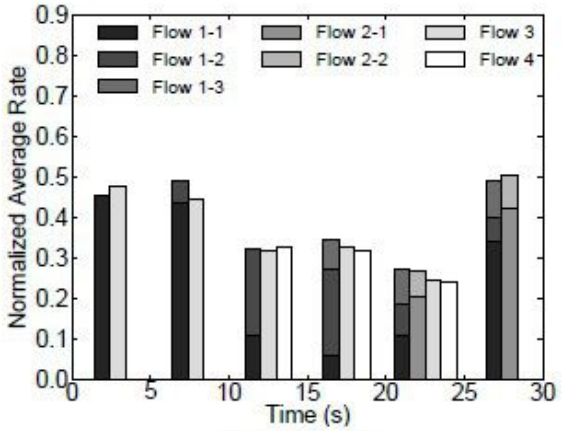
(b) $\beta = 6$

Figure 4: Two background flows run on $DN1$ from $10s$ to $20s$ and on $DN2$ from $20s$ to $30s$, respectively. They make Flow 2 dynamically shift its traffic from one subflow to the other.

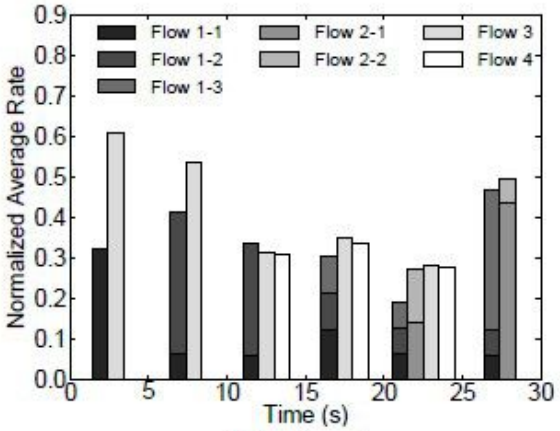
Experiment Result



(b) Fairness



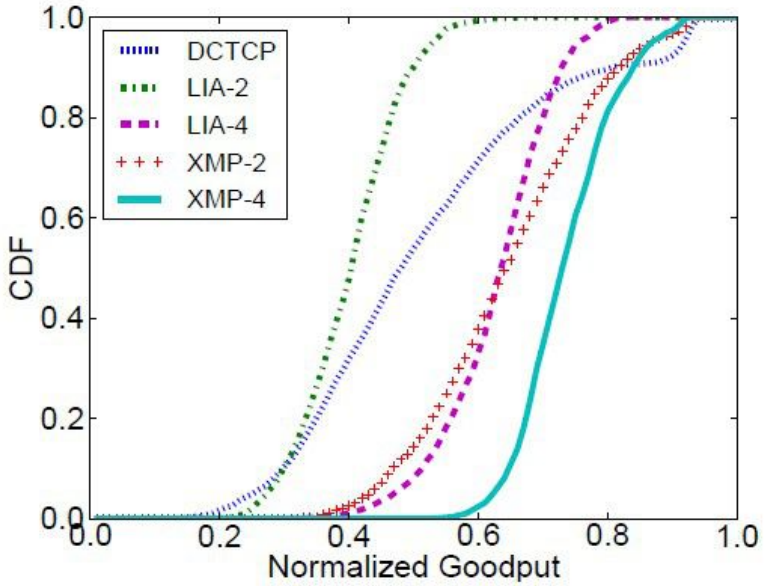
(a) $\beta = 4$



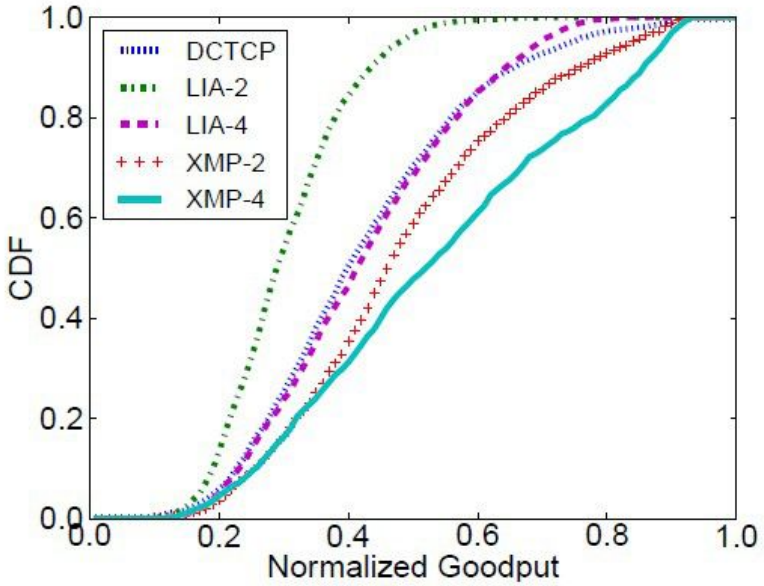
(b) $\beta = 6$

Figure 6: Four flows compete for one link.

Experiment Result



(a) Permutation



(b) Incast