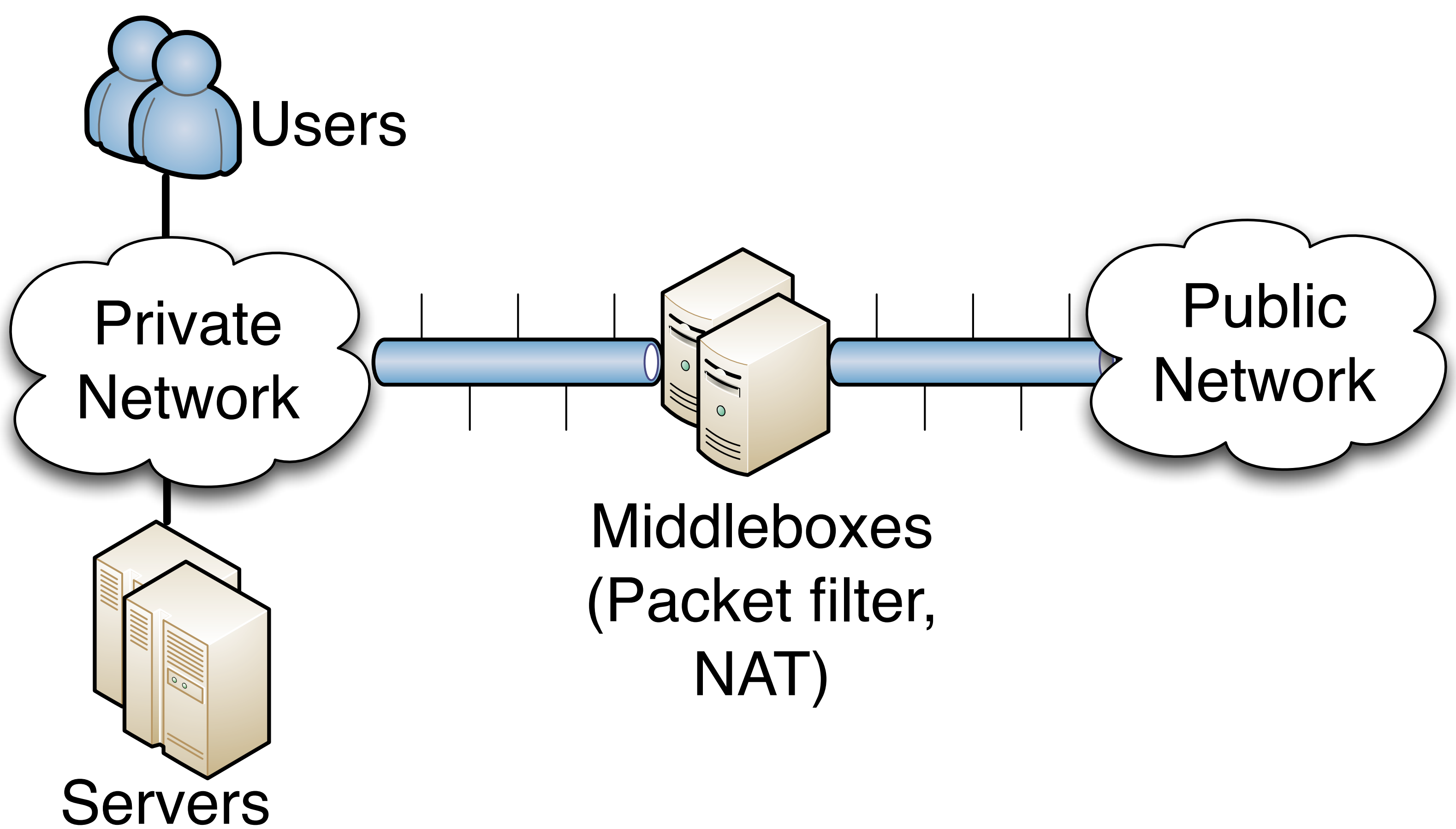
On the Fairness-Efficiency Tradeoff for Packet Processing with Multiple Resources

Wei Wang, Chen Feng, Baochun Li, Ben Liang
Department of Electrical and Computer Engineering
University of Toronto
December 4th, 2014

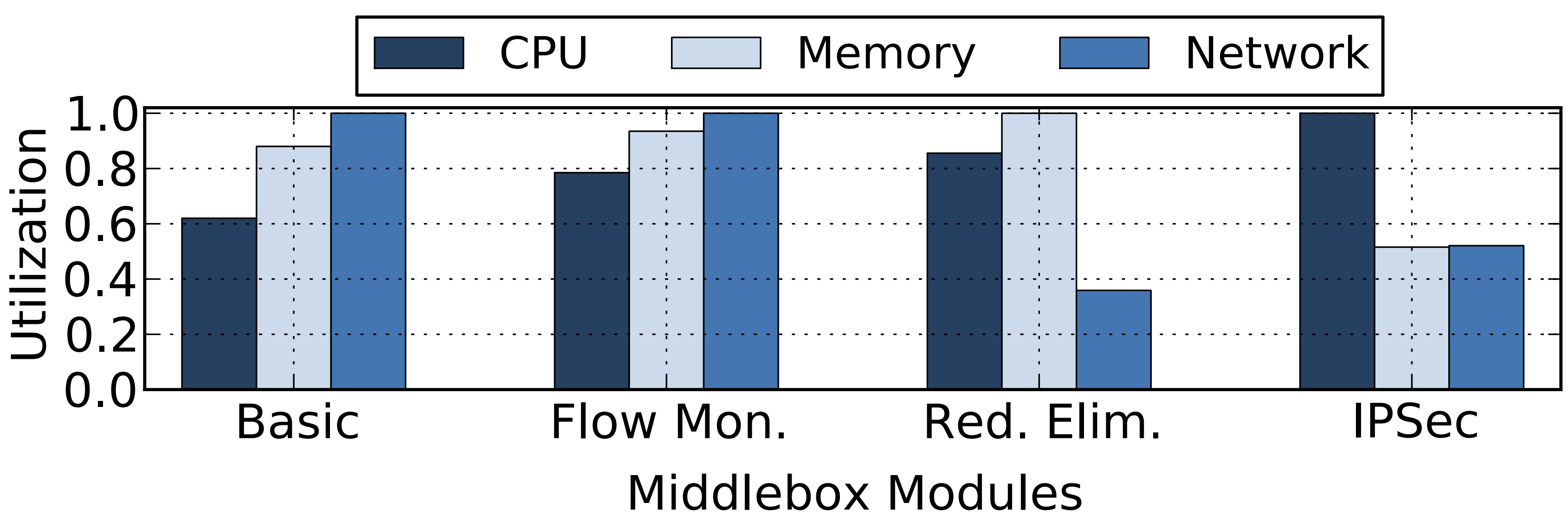
Middleboxes and Deep Packet Inspection

- Process packets based on payload
  - IPsec, Monitoring, Firewalls, WAN optimization, etc



Consumption of Multiple Resources

- Packet processing requires multiple types of resources (e.g., CPU, memory b/w, link b/w)
- Different middlebox (MB) modules consume different amounts of resources



Ghodsi et al. SIGCOMM'12

Resources should be shared **fairly** and **efficiently** among flows

Fairness

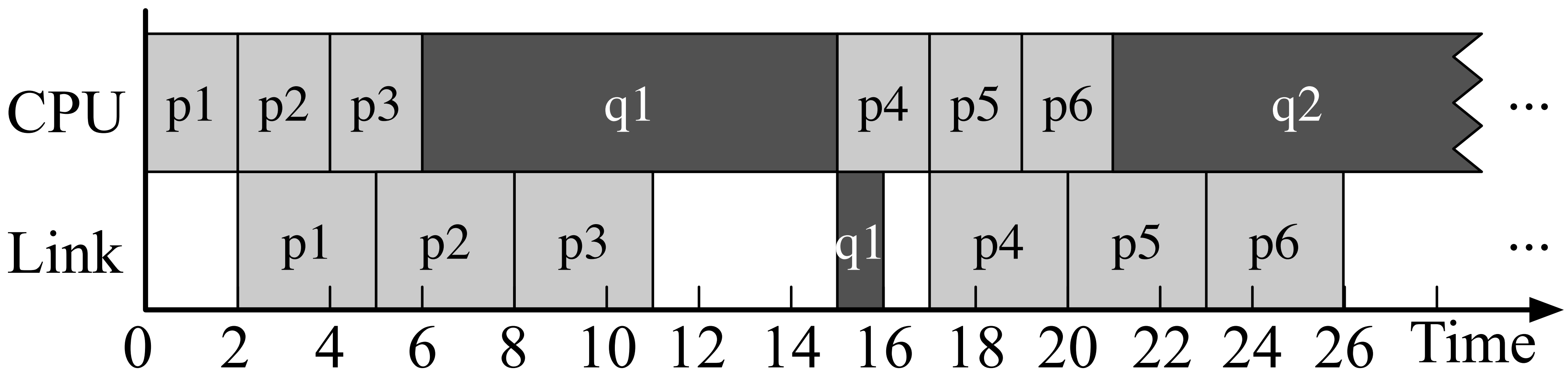
- Predictable service isolation
  - The service a flow receives in an *n*-flow system is at least 1/*n* of that it achieves when the flow monopolizes all resources
- Dominant Resource Fairness (DRF)
  - Flows receive *approximately the same* processing time on the *dominant resources* of their packets

Efficiency

- High resource utilization given a non-empty system, with high traffic throughput
  - Important in today's enterprise networks, as a surging volume of traffic now passes through MBs

However, fairness and efficiency are **conflicting** objectives in the presence of multiple resources

Fair but Inefficient

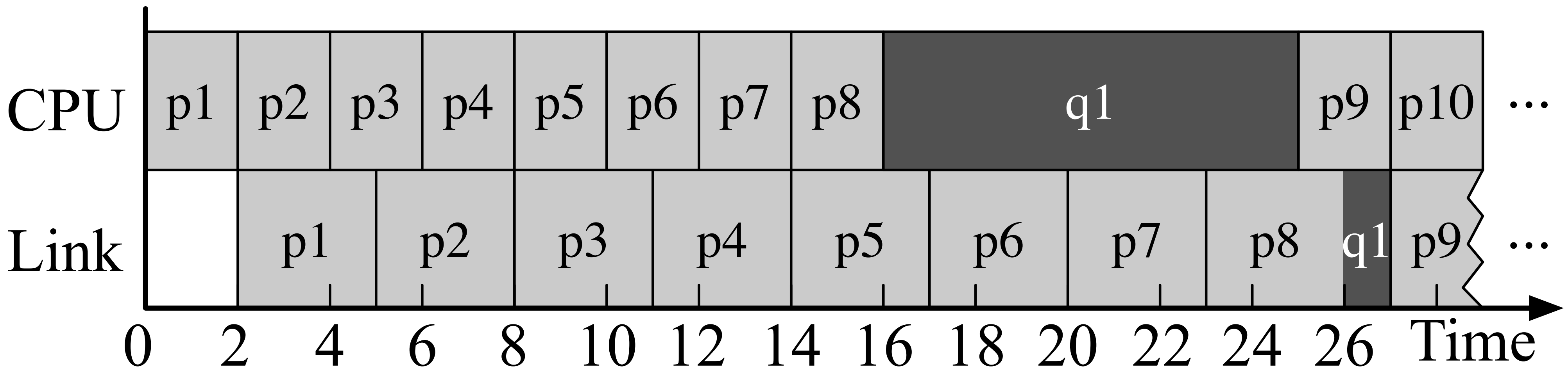


(a) A packet schedule that is fair but inefficient.

- Fair: both flows receive 9 time units to process on their dominant resources in each round

- Inefficient: link is idle at 1/3 of time

Efficient but Unfair



(b) A packet schedule that is efficient but unfair.

- Unfair: Flow 1 receives 96% of the link bandwidth; Flow 2 receives 36% of the CPU time
- Efficient: 100% CPU and link utilization given a non-empty system

Ideally…

- Allow the network operator to flexibly specify the tradeoff preference
    - Many applications may have loose fairness requirements
- Implement the specified tradeoff via a queueing algorithm

However…

- Existing multi-resource queueing algorithms focus only on fairness, without efficiency consideration
  - The tradeoff problem has never been mentioned before, and is **unique** to multi-resource scheduling
- Even the efficiency measure is unclear!

The Efficiency Measure

Schedule Makespan

- Time elapsed from the arrival of the first packet to the time when all packets finish processing on all resources
  - The completion time of the last flow

Max efficiency = Min makespan

Quantifying the Efficiency Loss

- Theoretical results
  - *m*: # of resource types concerned
  - the makespan of fair queueing could be up to *m* times the optimal makespan
- Experiment confirms 20% throughput loss of existing multi-resource fair queueing

Makespan minimization is notoriously hard, especially when there are more than two types of resources concerned (NP-hard)

We limit our discussion to the two most concerned types of resources for packet processing—CPU and link bandwidth

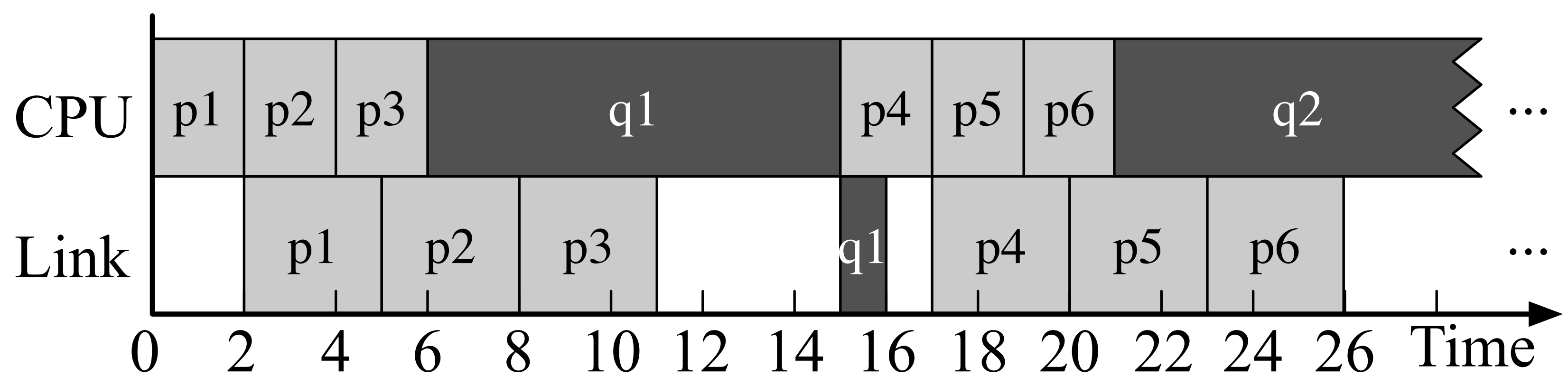
Our Approach

- Relax the scheduling problem to an idealized *fluid model*
- Discuss the tradeoff between fairness and efficiency in the fluid model
- Implement the fluid model in the real world via a packet-by-packet tracking algorithm

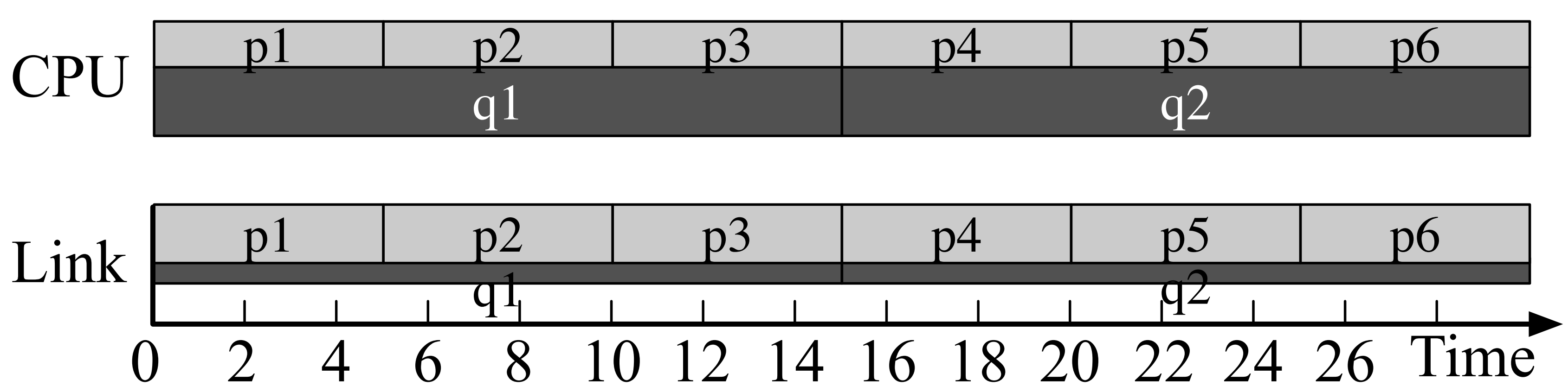
The Fluid Relaxation: packets are assumed to receive services in arbitrarily small increments on all resources

Fluid Relaxation

- Discrete schedule



- Fluid relaxation



Fluid w/ the Perfect Fairness

- Implement the strict DRF allocation at all times

Max-min flow's dominant share

$$\max_{d_i} \quad \min_{i \in \mathcal{B}} d_i$$
$$\text{s.t.} \quad \sum_{i \in \mathcal{B}} \bar{\tau}_{i,r} d_i \leq 1, \quad r = 1, 2 \ .$$

Resource constraints

- All flows receive the same *fair dominant share*

$$\bar{d} = 1 / \max \left\{ \sum_i \bar{\tau}_{i,1}, \sum_i \bar{\tau}_{i,2} \right\}$$

Fluid w/ the Optimal Efficiency

- Greedily maximizes the *system dominant share* at all times

Maximize system dominant share

$$\max_{d_i \geq 0} \quad \sum_{i \in \mathcal{B}_t} d_i$$

$$\text{s.t.} \quad \sum_{i \in \mathcal{B}_t} \bar{\tau}_{i,r} d_i \leq 1, \quad r = 1, 2.$$

Resource constraints

Fairness-Efficiency Tradeoff

Specifying Fairness Requirement

- Let $\bar{d}$ be the *fair dominant share* under DRF
- Let $\alpha \in [0, 1]$ be a *fairness knob* specified by the operator
- **Fairness constraint**: flows receive at least α-portion of fair dominant share

Fair share under DRF

$$d_i \geq \alpha \bar{d}, \quad \forall i \in \mathcal{B},$$

Dominant share of flow *i*

Fairness-Efficiency Tradeoffs

- Maximize the *system dominant share* under a specified tradeoff level (quantified by fairness knob $\alpha \in [0, 1]$)

$$\begin{aligned} \max_{d_i} \quad & \sum_{i \in \mathcal{B}_t} d_i \\ \text{s.t.} \quad & \sum_{i \in \mathcal{B}_t} \bar{\tau}_{i,r} d_i \leq 1, \quad r = 1, 2, \\ & d_i \geq \alpha \bar{d}, \quad \forall i \in \mathcal{B}_t \ . \end{aligned}$$

Resource constraint

Fairness constraint

Implement the fluid model via packet-by-packet tracking

Start-Time Tracking

- Maintain the Tradeoff Fluid as a reference system in the background
- In the real world, whenever there is a packet scheduling opportunity, the one that starts the **earliest** in the Tradeoff Fluid is scheduled first
- An O(log n) implementation based on a *special structure* of the Tradeoff Fluid
- **Asymptotically close** to the fluid model in terms of both makespan and fairness guarantee

Evaluation

Experiment Setup

- Prototype implementation in Click modular router
- 60 UDP flows each sending 2,000 800-byte pkts/s
- Three middlebox processing modules
  - Packet checking (bandwidth-bound): Flows 1~20
  - Statistical monitoring (bandwidth-bound): Flow 21~40
  - IPsec (CPU-bound): Flows 41~60

Scenario 1: No packet drop

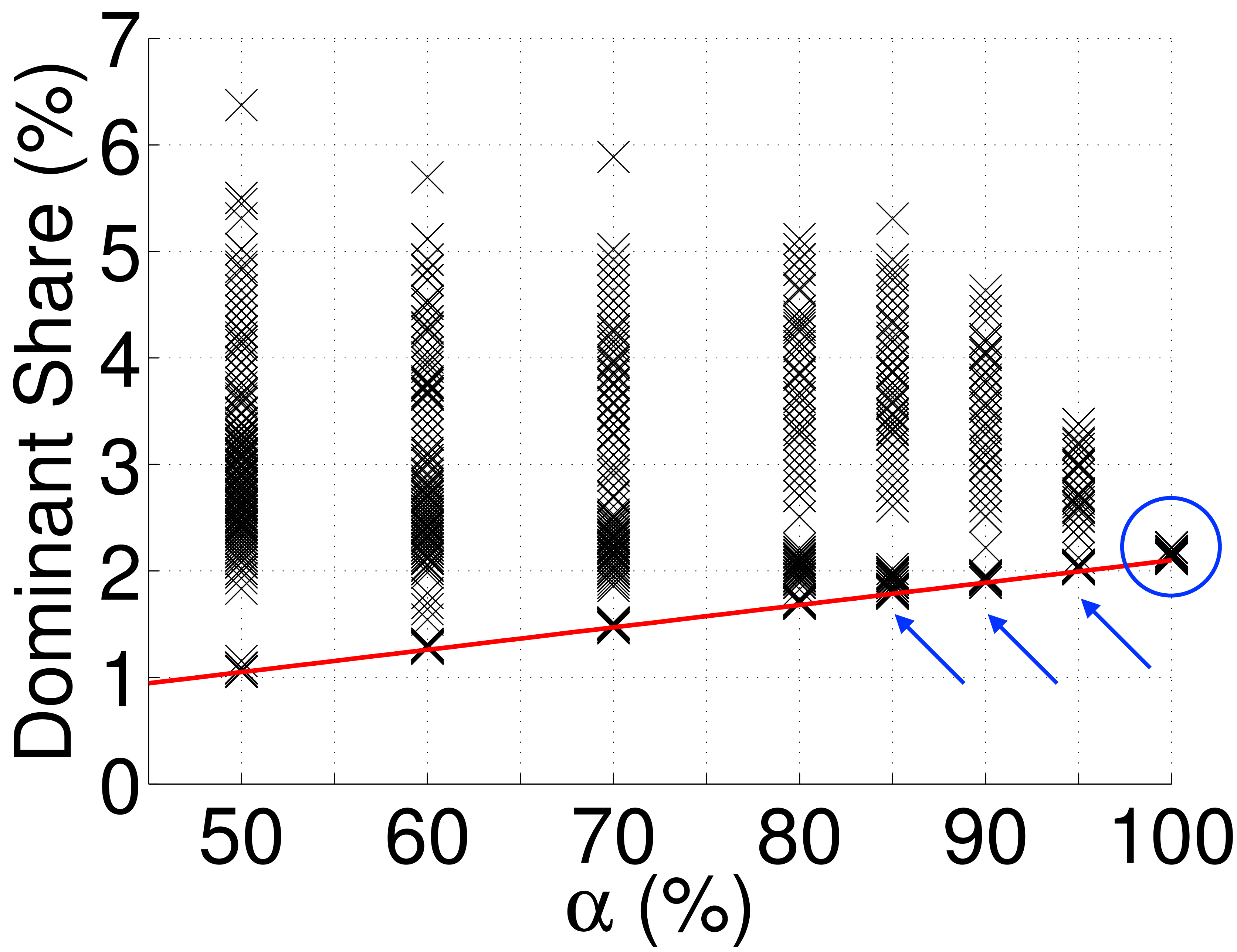
Makespan

- Each flow sends 10s traffic

α	**Makespan (s)**	**Normalized Makespan (%)**
1.00	55.68	100.00
0.95	52.50	94.28
0.90	48.97	87.95
0.85	47.17	84.72
0.70	47.13	84.64
0.60	47.07	84.54
0.50	47.07	84.54

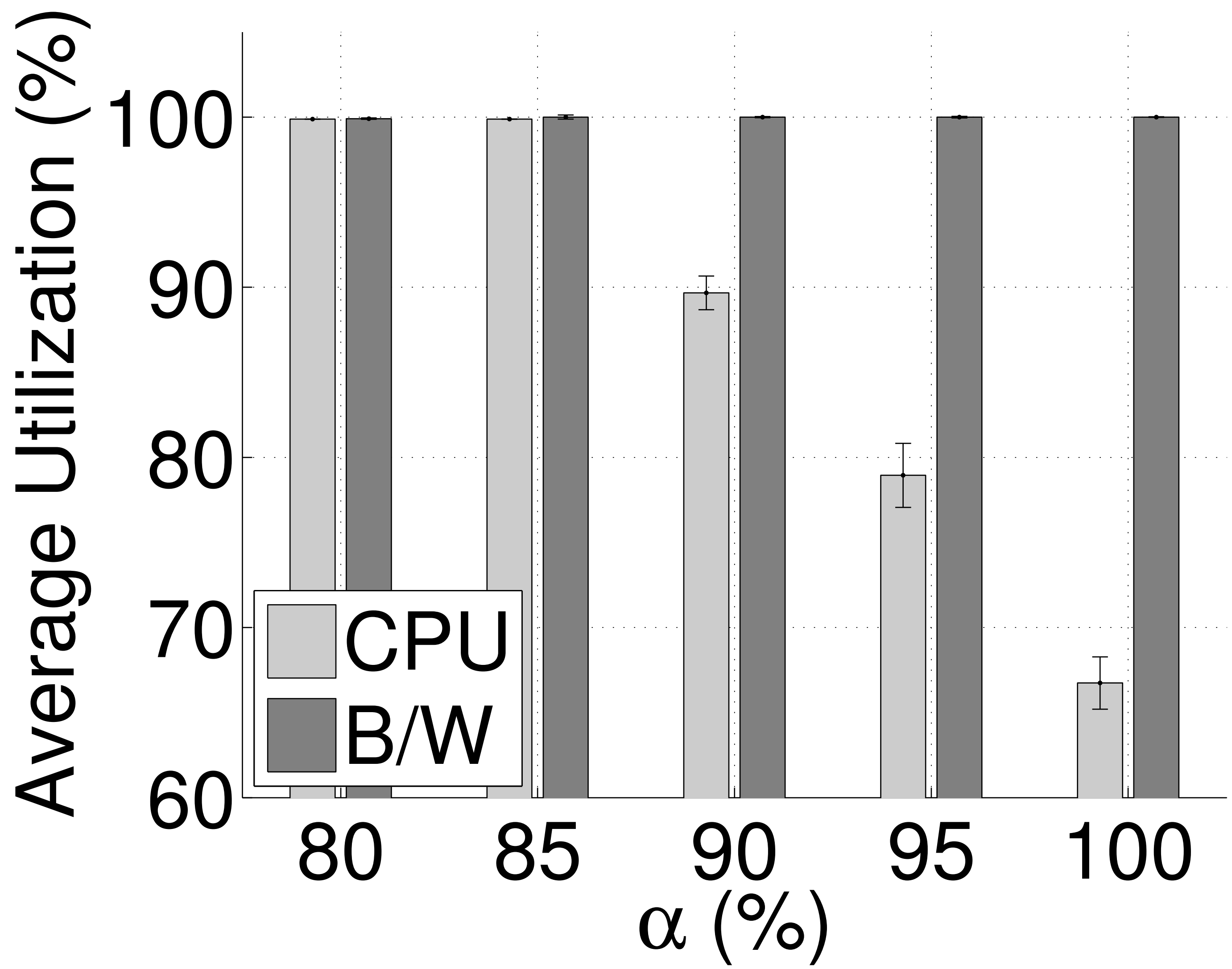
- Trading off 15% of fairness is sufficient to achieve the shortest makespan (20% throughput improvement)

Fairness

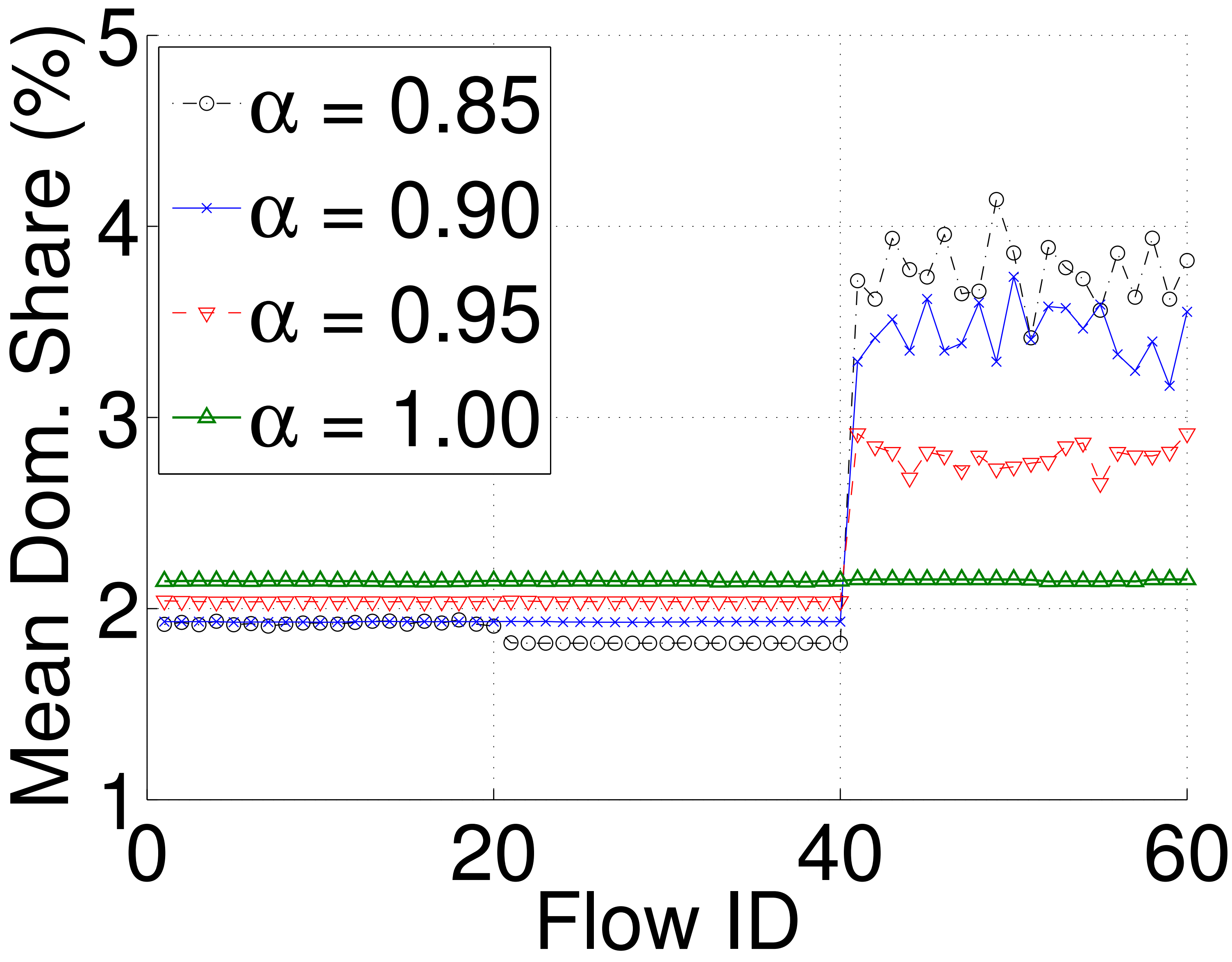


Scenario 2: buffer size=200

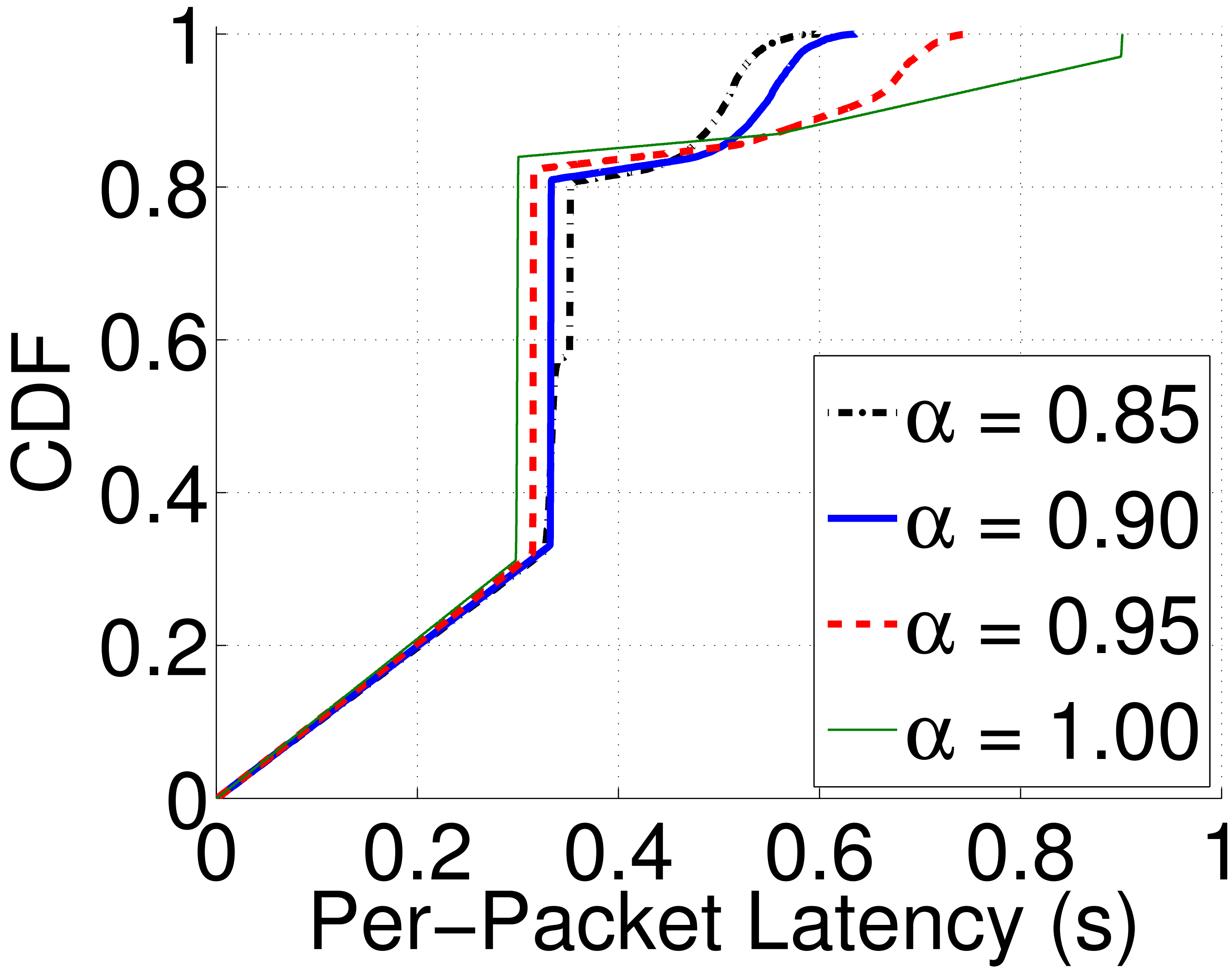
Resource Utilization



Dominant Share



Per-Packet Latency



Conclusions

- We have identified the problem of fairness-efficiency tradeoffs for multi-resource packet scheduling
- We have designed a scheduling algorithm to achieve a flexible tradeoff between fairness and efficiency for packet processing that requires both CPU and link bandwidth
- We have prototyped the tradeoff algorithm in Click. Experimental results show that slight fairness tradeoff is sufficient to achieve the highest efficiency

Thank you!

weiwang@ece.utoronto.ca
http://iqua.ece.toronto.edu/~weiwang/