Multi-Resource Round Robin: A Low Complexity Packet Scheduler with Dominant Resource Fairness

Wei Wang, Baochun Li, Ben Liang
Department of Electrical and Computer Engineering
University of Toronto
October 10, 2013

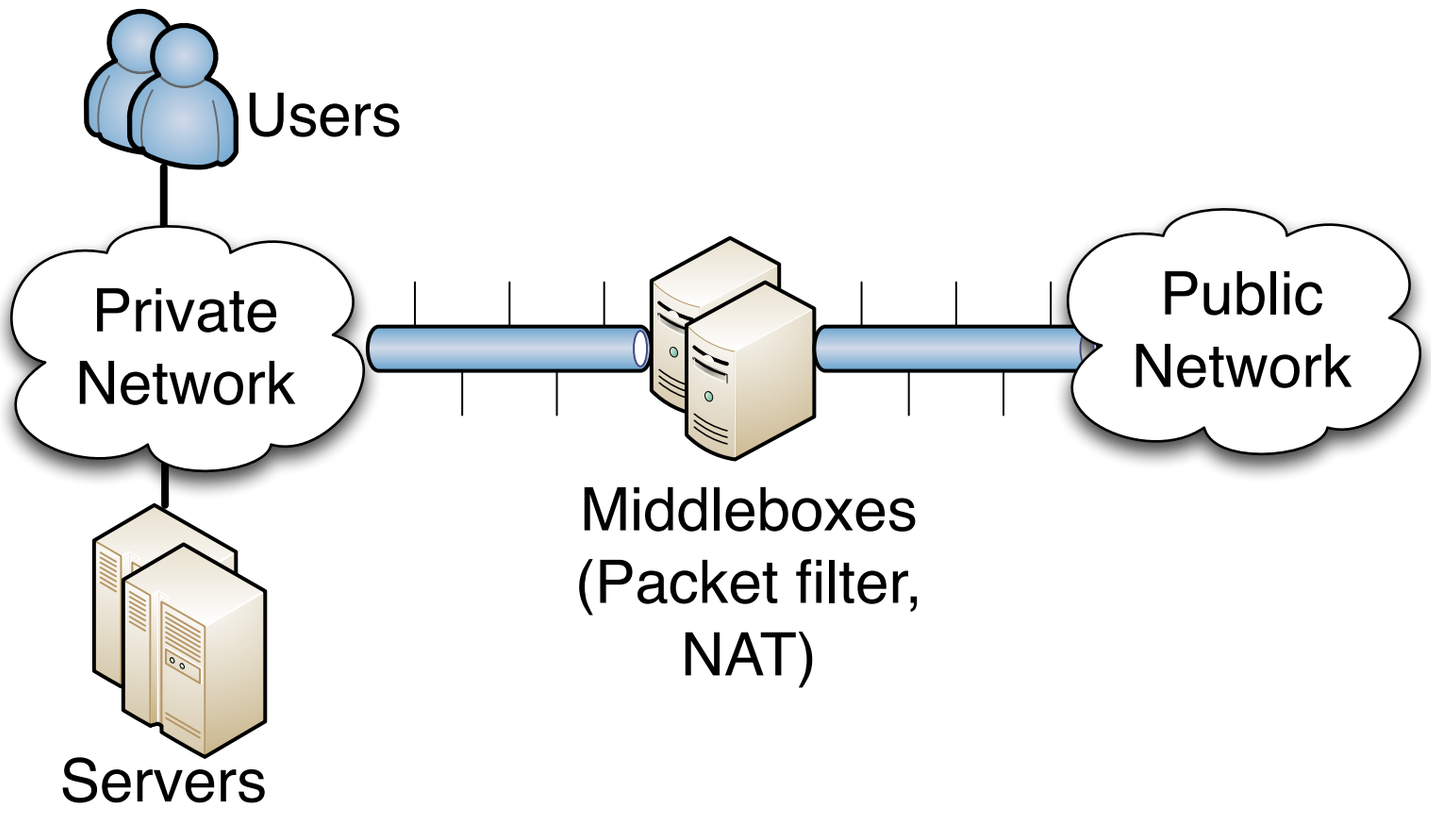
Background

Middleboxes (MBs) are ubiquitous in today's networks

The sheer number is on par with the L2/L3 infrastructures [Sherry12]

Perform a wide range of critical network functionalities

E.g., WAN optimization, intrusion detection and prevention, etc.



Background

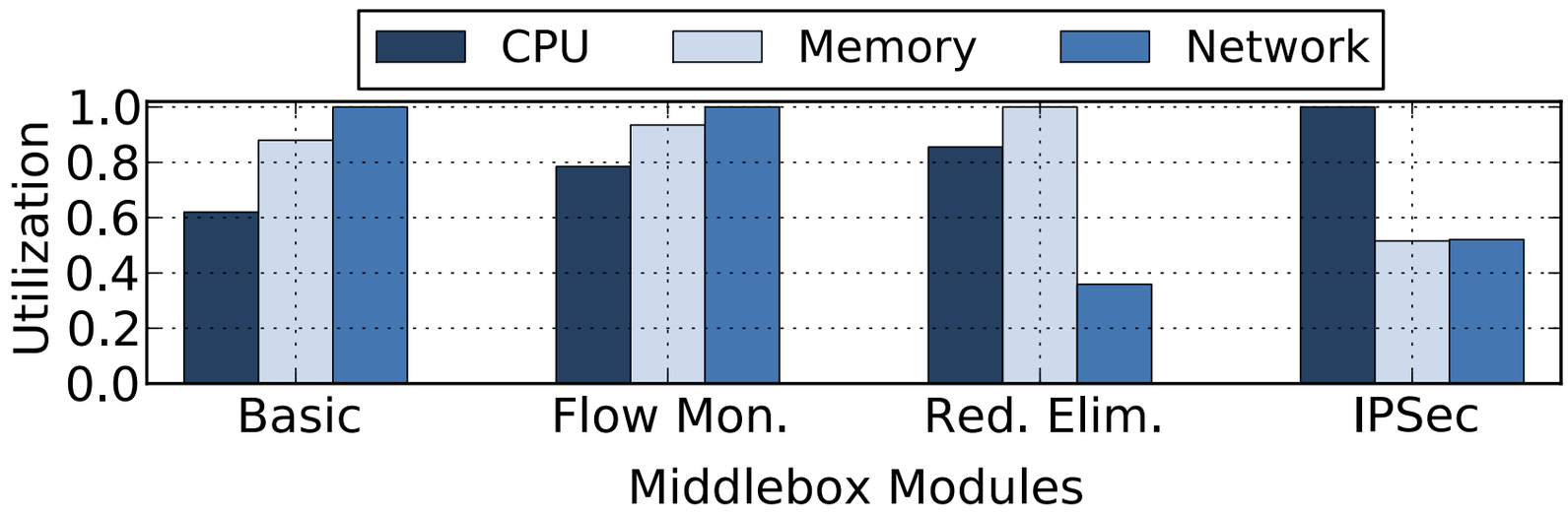
MBs perform deep packet processing based on packet contents

Require ***multiple*** MB resources, e.g., CPU, link bandwidth

Flows may have heterogenous resource demands

Basic Forwarding: Bandwidth intensive

IP Security Encryption: CPU intensive



Ghodsi et al. *SIGCOMM'12*

How to let flows *fairly* share *multiple resources* for packet processing?

Wei Wang, Department of Electrical and Computer Engineering, University of Toronto

Desired Fair Queueing Algorithm

Fairness

Each flow should receive service (i.e., throughput) at least at the level when ***every resource*** is equally allocated

Low Complexity

To schedule packets at high speeds, the scheduling decision has to be made at low time complexity

Implementation

The scheduling algorithm should also be simple enough so that it can be easily implemented in practice

The Status Quo

Traditional fair queueing algorithms have only a *single* resource to schedule, i.e., output bandwidth

- Switches simply forward the packet to the next hop
- WFQ, WF^{2}Q, SFQ, DRR, etc.

Simply extending single-resource fair queueing fails to achieve fairness in the multi-resource setting [Ghodsi12]

- Per-resource fairness
- Bottleneck fairness

Dominant Resource Fair Queueing (DRFQ) [Ghodsi12]

- Implements near-perfect **Dominant Resource Fairness** (DRF) in the time domain

However...

DRFQ is expensive to implement at high speeds

- Requires O(log *n*) time complexity per packet
- *n* could be very large
  - Given the ever growing line rate and the increasing volume of traffic passing through MBs

Recent software-defined MB innovations further aggravate this scalability problem

- More software-defined MBs are consolidated onto the same commodity servers
  - They will see an increasing amount of traffic passing through them

Our Contributions

A new multi-resource fair queueing algorithm

- Multi-Resource Round Robin (MR^3)
- Near-perfect fairness across flows
- O(1) time complexity per packet
- Very easy to implement in practice

MR^3 is the first multi-resource fair queueing algorithm that achieves nearly perfect fairness with O(1) time complexity

Preliminaries

Wei Wang, Department of Electrical and Computer Engineering, University of Toronto

Dominant Resource Fairness (DRF)

Dominant Resource

The resource that requires the most processing time to process a packet

$$d(p) = \arg\max_r \{\tau_r(p)\}$$

For example

A packet *p* requiring 1 CPU time and 3 transmission time

Dominant resource is the link bandwidth

DRF

Flows receive the ***same processing time*** on their dominant resources

Max-min fairness on flows' dominant resources

Design Objective

We use Relative Fairness Bound (RFB) to measure the fairness of a scheduling algorithm

$$\mathrm{RFB} = \sup_{t_1,t_2;i,j\in\mathcal{B}(t_1,t_2)} |T_i(t_1,t_2) - T_j(t_1,t_2)|$$

$T_i(t_1,t_2)$: the packet processing time flow *i* receives on its dominant resource in the time interval (t_1,t_2)

Referred to as the **dominant services**

Objective

RFB as a small constant

O(1) scheduling complexity per packet

When there is a single resource to schedule...

Fair Queueing Based on Round Robin

Round-robin (RR) scheme

Flows are served in rounds

In each round, each flow transmits roughly the same amounts of bits

A ***credit system*** is maintained to track the amounts of bits transmitted

RR is an ideal single-resource packet scheduler

Nearly perfect fairness with O(1) RFB

O(1) time complexity

Simple, and widely implemented in high-speed routers

E.g., Cisco GSR

Will the attractiveness of RR extend to the multi-resource setting?

Wei Wang, Department of Electrical and Computer Engineering, University of Toronto

The First Try

Intuition

DRF implements max-min fairness on flows' dominant resources

Simply applying RR to flows' dominant resources

Approach

Maintain a **credit system** to track the dominant service a flow has received

Ensure that flows receive roughly the same processing time on their dominant resources in each round

Credit System

Active flows are served in rounds

Each flow *i* maintains a credit account

Balance: B_i

The dominant service flow *i* is allowed to consume in one round

Whenever a packet *p* is processed, $B_i = B_i - \mathrm{domProcTime}(p)$

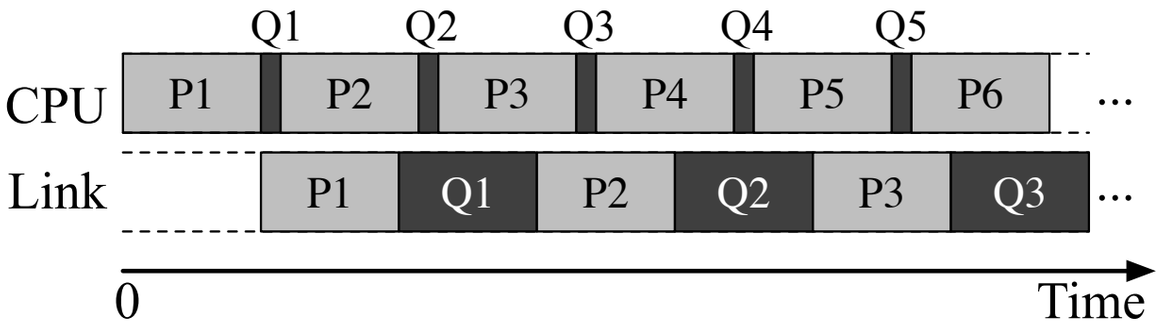
Flow *i* is allowed to process packet, as long as $B_i \geq 0$

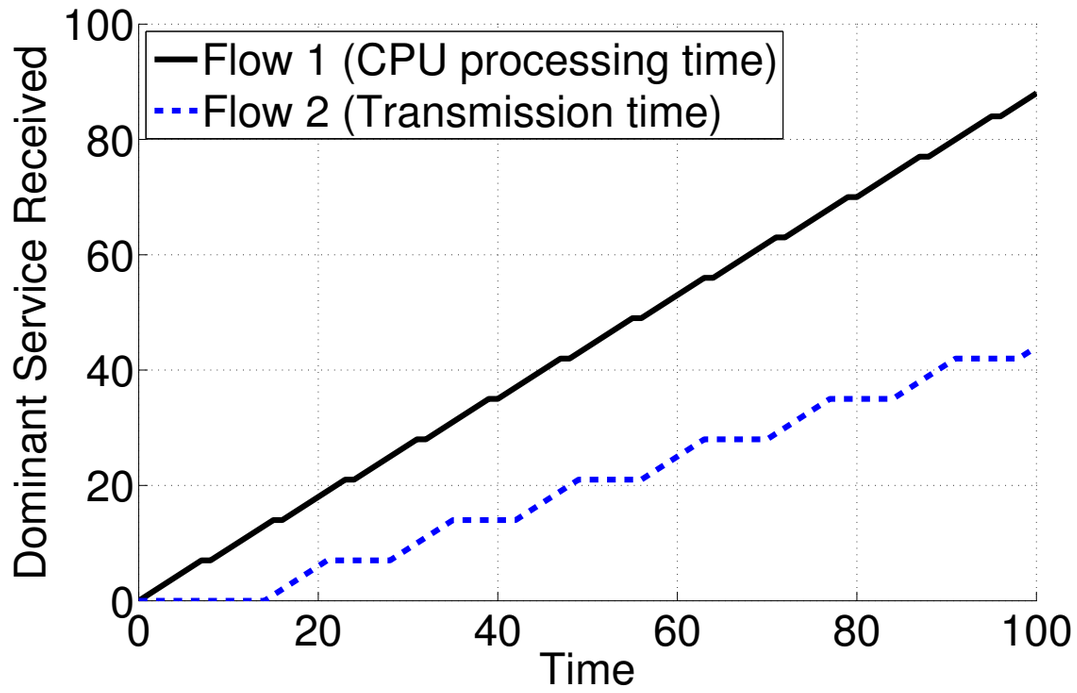
A new round begins when all active flows have been served in the previous round

All flows receive a credit, i.e., $B_i = B_i + c$, after which $B_i \geq 0$ for all *i*

However...

Such a simple extension may lead to *arbitrary unfairness*!





Root cause

Heterogeneous resource demand

Inconsistent work progress on different resources

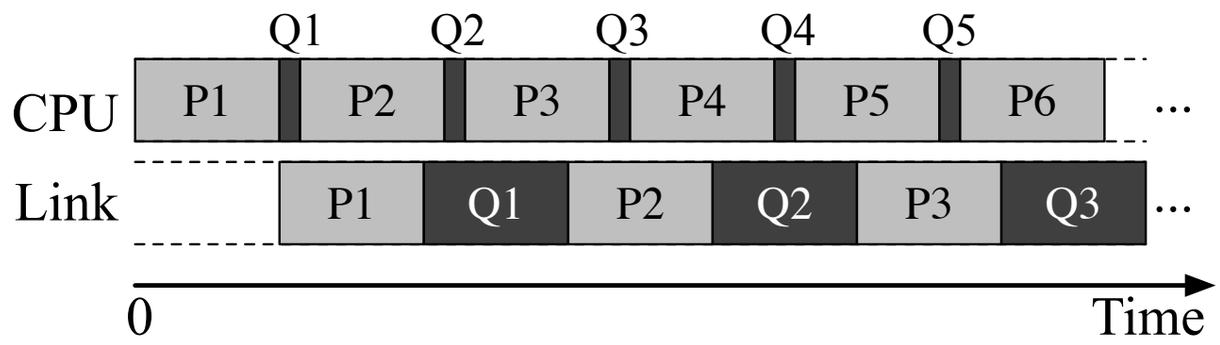
The Second Try

Credit system gives the *right* order, but *wrong* timing

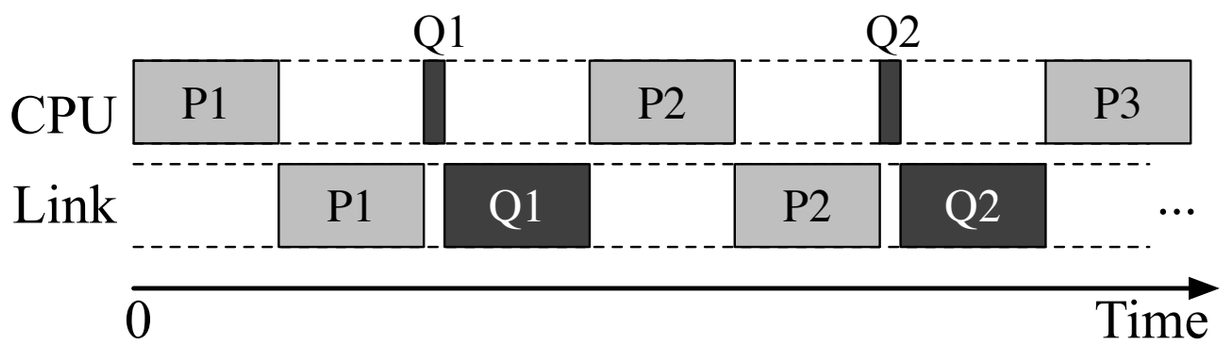
Enforce consistent work progress across resources

Allow only one packet to be processed at one time

Unfair



Fair



Significantly high delay with low resource utilization

The Right Timing for Scheduling

Enforce a roughly consistent work progress without sacrificing delays

Progress control mechanism

Bound the progress gap between any two resources by 1 round

A packet *p* of flow *i* is ready to be processed in round *k*

Check the work progress on the last resource

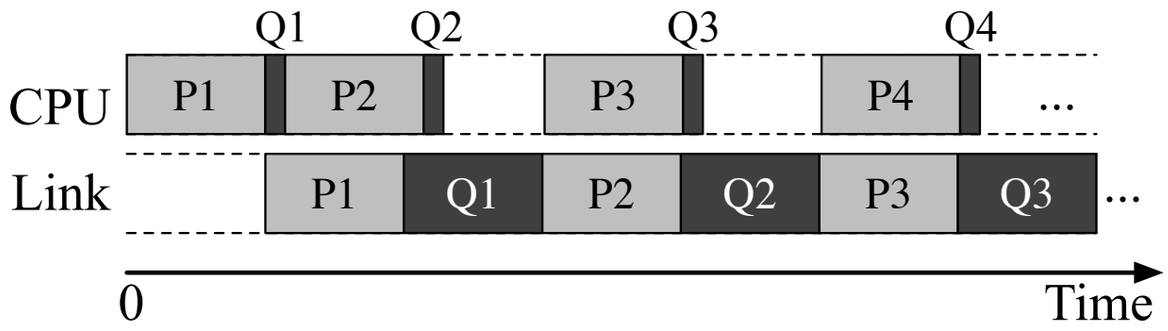
Process *p* immediately if

Flow *i* is a new arrival

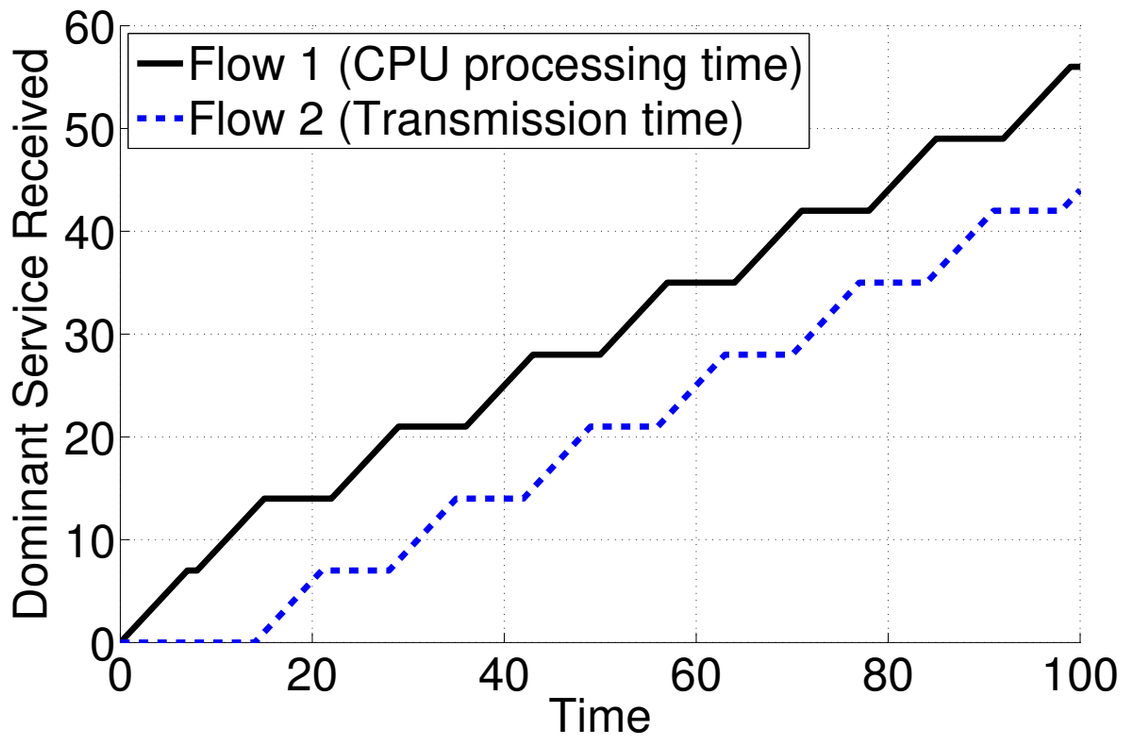
i has already received services on the last resource in the previous round *k*-1

Otherwise, withhold *p* until the condition above is satisfied

An example



(a) Schedule by MR^3.



(b) The dominant services received by two flows.

Fig. 4. Illustration of a schedule by MR^3.

MR³ Recap

Flows are served in rounds

Credit system

- Applied to flows' dominant resources
  - Track the dominant services a flow has received
- Decide the scheduling order
- Similar to the single-resource scenario

Progress control mechanism

- Ensure a roughly consistent work progress across resources
- Decide the right timing for scheduling
- Unique to the multi-resource scenario

Simple idea, easy to implement, yet is sufficient to lead to near-perfect fairness

Wei Wang, Department of Electrical and Computer Engineering, University of Toronto

Analytical Results

Wei Wang, Department of Electrical and Computer Engineering, University of Toronto

Properties of MR^3

O(1) time complexity

Nearly perfect fairness with O(1) RFB

Slight delay increase as compared with DRFQ

Easy to implement

No *a priori* information about packet processing is required

TABLE I
PERFORMANCE COMPARISON BETWEEN MR^3 AND DRFQ, WHERE L IS THE MAXIMUM PACKET PROCESSING TIME; m IS THE NUMBER OF RESOURCES; AND n IS THE NUMBER OF BACKLOGGED FLOWS.

Performance	**MR^3**	**DRFQ** [11]
Complexity	$O(1)$	$O(\log n)$
Fairness (RFB)	$4L$	$2L$
Startup Latency	$2(m+n-1)L$	nL
Single Packet Delay	$(4m+4n-2)L$	Unknown

Simulation Results

Wei Wang, Department of Electrical and Computer Engineering, University of Toronto

General Setup

3 MB modules

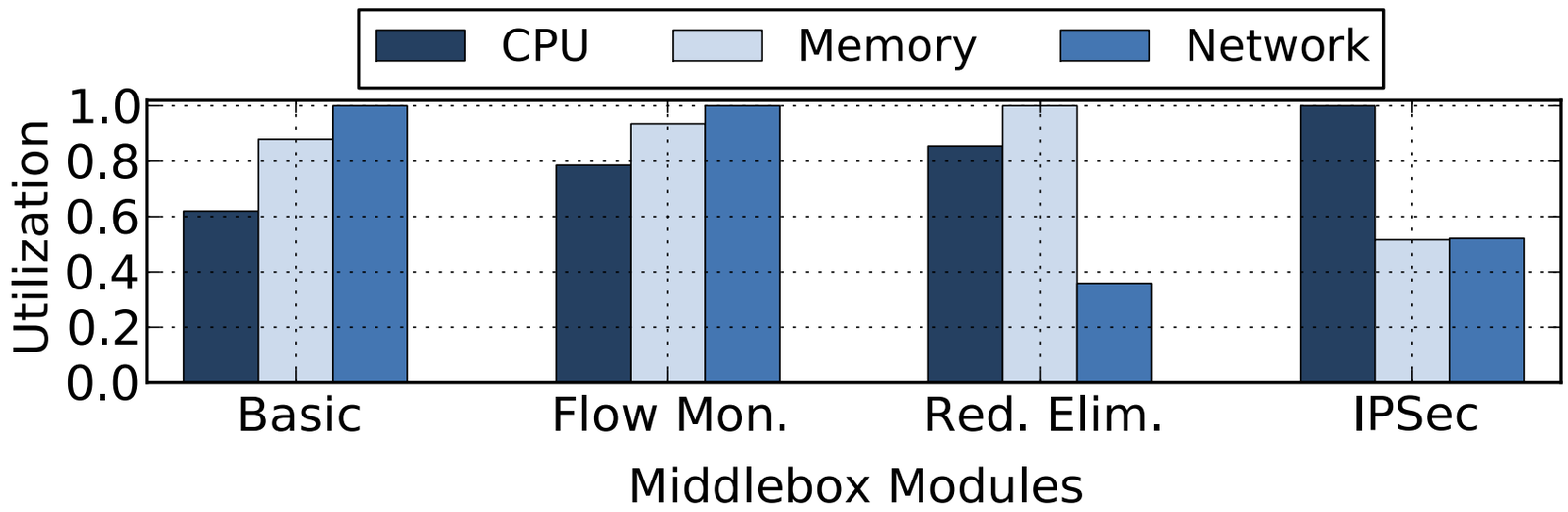
Basic forwarding

Statistical monitoring

IPSec Encryption

Bandwidth intensive

CPU intensive



Ghodsi et al. SIGCOMM'12

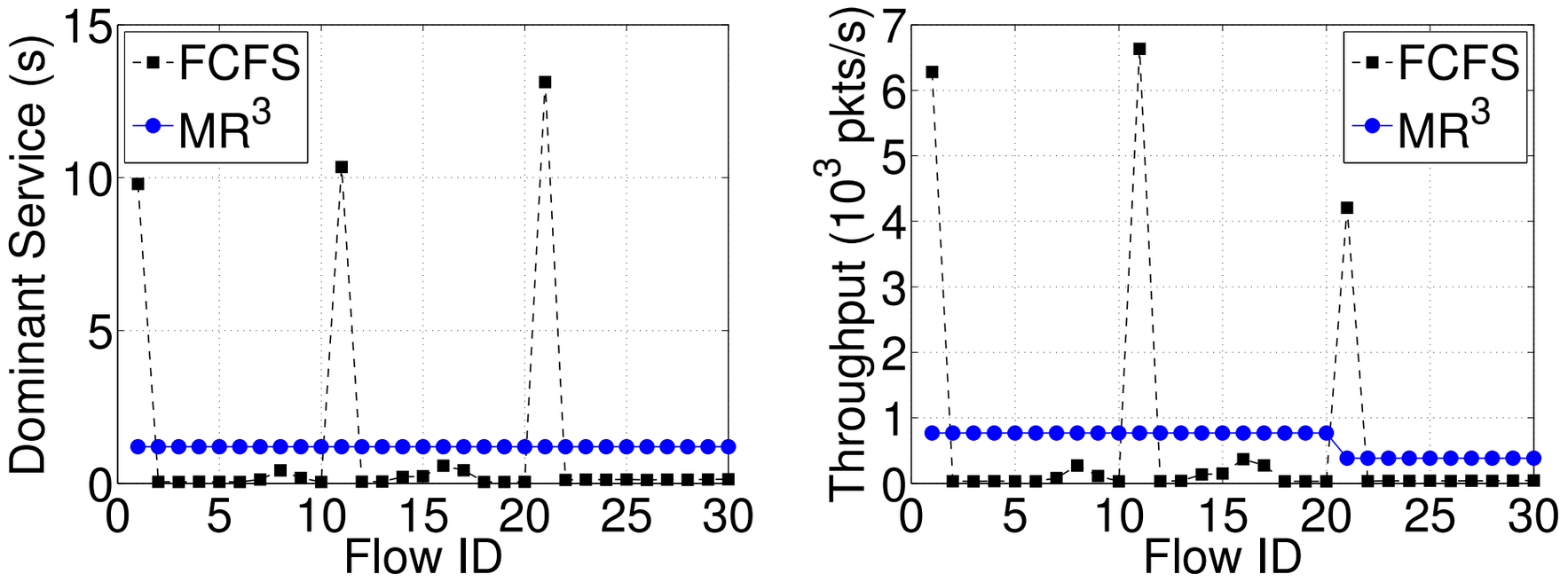
Fairness (Service Isolation)

30 UDP flows with 3 rogue flows

Flows 1 to 10 require basic forwarding

Flows 11 to 20 require statistical monitoring

Flows 21 to 30 require IPSec encryption



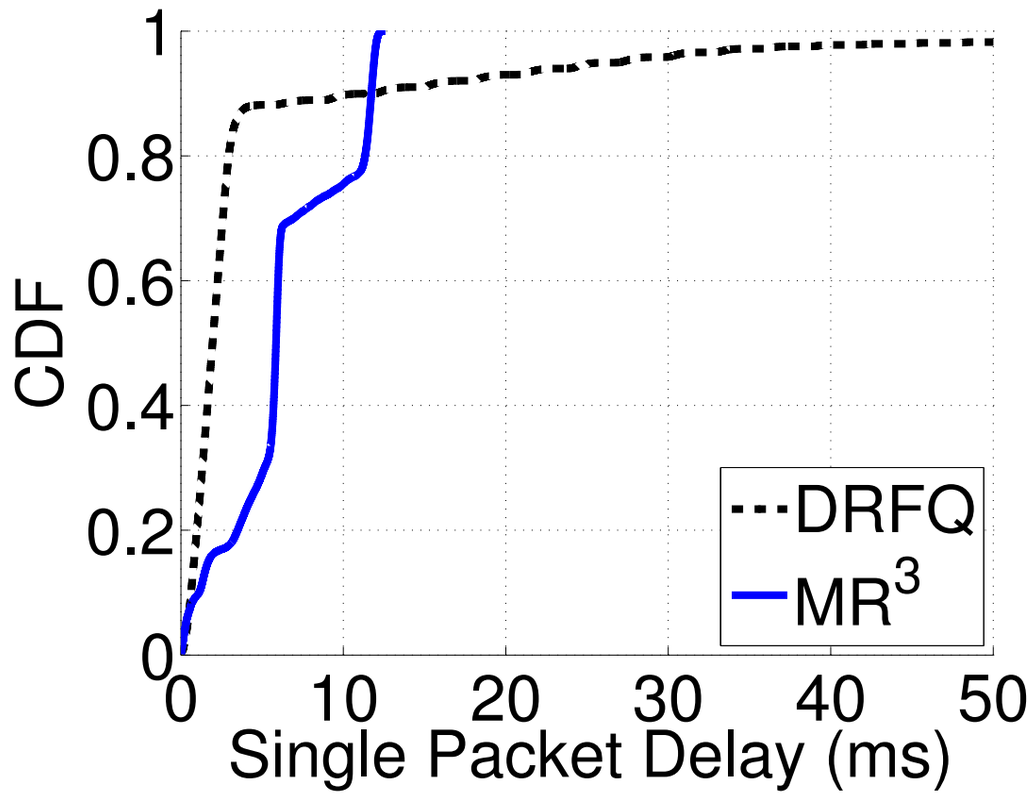
(a) Dominant service received. (b) Packet throughput of flows.

Fig. 6. Dominant services and packet throughput received by different flows under FCFS and MR^3. Flows 1, 11 and 21 are ill-behaving.

Latency

150 UDP flows

Slight latency increase as compared with DRFQ



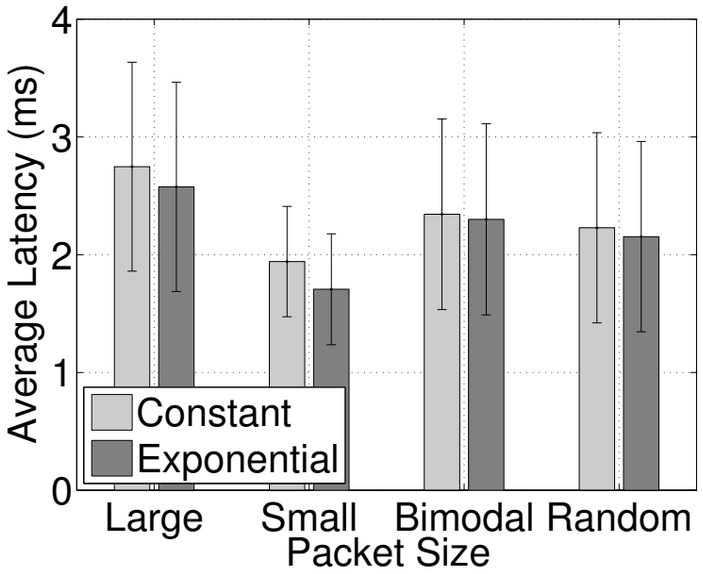
(b) CDF of the single packet delay.

Stability

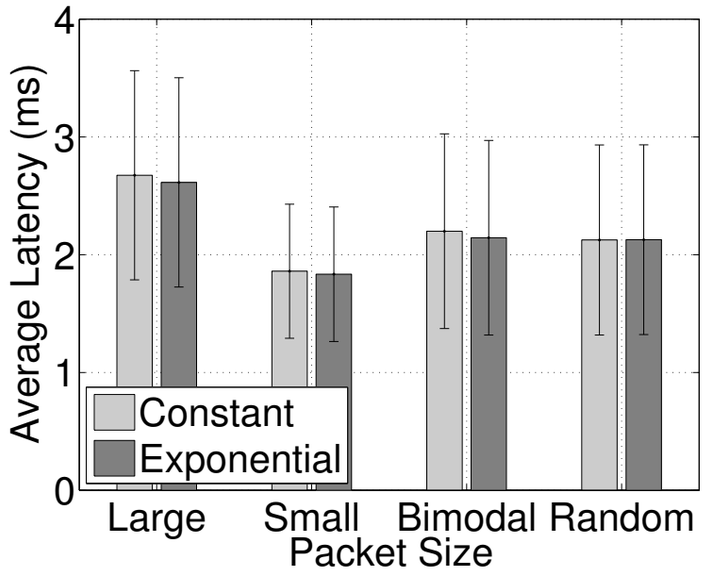
Different traffic arrival patterns

Different packet size distributions

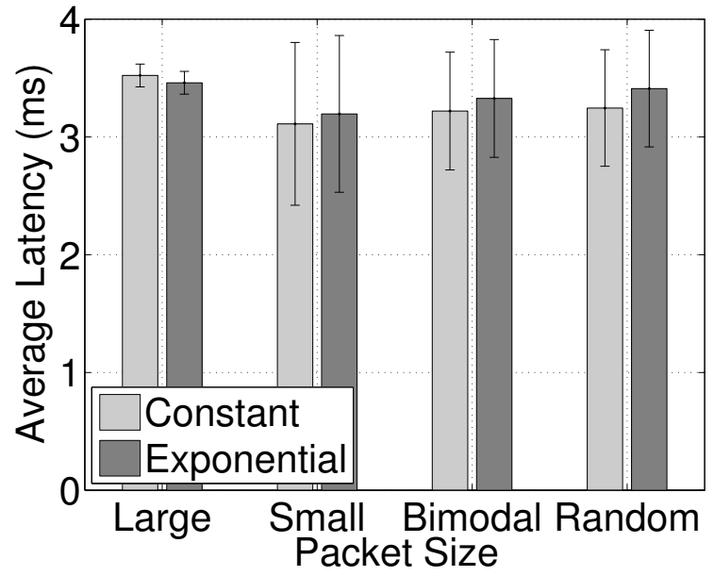
Different network functionalities applied to flows



(b) Basic forwarding.



(c) Statistical monitoring.



(d) IPSec encryption.

Conclusions

We propose MR³ and evaluate its performance both analytically and experimentally

The first multi-resource fair queueing algorithm

- achieves nearly perfect fairness
- O(1) time complexity
- Slight increase of packet latency as compared with DRFQ

MR³ could be easily extended to some other multi-resource scheduling contexts

- E.g., VM scheduling inside a hypervisor

weiwang@eecg.toronto.edu
http://iqua.ece.toronto.edu/~weiwang/