Universal Scheduling Mechanisms

Rachit Agarwal
Cornell University (starting Fall)
Network Task

Delivering data from source(s) to corresponding destination(s)

- Policies (e.g., resource sharing)
- Performance metrics (e.g., latency, throughput)

Switching fabric

Which packet to schedule next? (Packet scheduling)

Network fabric

Which flow to schedule next? (Flow scheduling)
Great area to write papers do research

- Ever-evolving metrics (latency, tail latency, deadlines, ....)
- … workloads (flows, short flows, coFlows, ....)
- … contexts (WAN, datacenters, cellular, ....)
- … trends (new/commodity hardware, centralized, distributed, ....)

Choose a random permutation!
Questions ...

Question 1: new near-optimal mechanism for a new objective?
   • FIFO, FIFO+, LIFO, Round Robin, FQ, ....
   • TCP, DCTCP, D2TCP, D3, PDQ, Fastpass, pFabric, pHost, PIAS, PASE, ....

Question 2: how to support different mechanisms?
   • Change hardware for each new mechanism?
   • Programmable network hardware?
     • The set of abstractions for each and every possible mechanism?
Questions ...

Question 1: new near-optimal mechanism for a new objective?

Question 2: how to support different mechanisms for different objectives?

Question:

one near-optimal mechanism for all objectives?
**Question:**

one near-optimal mechanism for all objectives?

**P4 and the RMT switch**

- no abstractions and implementations for scheduling
- **Right time** to ask the question!

**NO:** we need to design new mechanisms with new objective

- Programmable *packet schedulers* a necessity
- Focus on designing the set of abstractions

**YES:** changes the **lens through which we view** scheduling mechanisms

- **one abstraction:** simplified network programming
- is it cheaper to implement than the universal mechanism?
Universal Scheduling Mechanism: Feasibility?
Universal Scheduling Mechanism: Feasibility?
Universal Scheduling Mechanism: Feasibility?

Packet header initialization
(Priority, flow size, flow weight, .. —based on the objective)

Packet output timing
(at egress)
Universal Scheduling Mechanism: Feasibility?

Packet scheduling in A (switch decisions — potentially different for each switch & based on the objective)

Packet header initialization (Priority, flow size, flow weight, .. — based on the objective)

Packet output timing (at egress)
Packet scheduling in A (switch decisions — potentially different for each switch & based on the objective)

Packet header initialization (Priority, flow size, flow weight, .. —based on the objective)

Packet output timing (at egress)
Universal Scheduling Mechanism: Definition

Given

• any network topology
• any workload (set of flows, their arrival times, and paths)
• any objective function
• each switch may keep infinite state, and implement any logic
• An optimal scheduling algorithm A
**Universal Scheduling Mechanism: Definition**

Given

- any network topology
- any workload (set of flows, their arrival times, and paths)
- any objective function
- each switch may keep infinite state, and implement any logic
- An optimal scheduling algorithm A

The Universal Scheduling Mechanism should

- produce packet timing at the egress same as A for each packet
- with each switch implementing the same logic
- without keeping any additional state
There exists a Universal Scheduling Mechanism

• except for flows that encounter more than two congestion points
Universal Scheduling Mechanism: Preliminary results

There exists a Universal Scheduling Mechanism

• except for flows that encounter more than two congestion points

Header initialization

• “updatable” slack, at ingress based on objective
Universal Scheduling Mechanism: Preliminary results

There exists a Universal Scheduling Mechanism

- except for flows that encounter more than two congestion points

Header initialization

- “updatable” slack, at ingress based on objective

Switch logic

- Enqueue
- Dequeue: Least Slack First
- Reduce slack value by “waiting time”
- Just need Pipelined-heap implementation
Our current constraints:

- any network topology
- any workload (set of flows, their arrival times, and paths)
- any objective function
- each switch may keep infinite state, and implement any logic
- An optimal scheduling algorithm A

Relax the constraints

- What if a few packets are allowed to be delayed?
- Approximation algorithms?
- Randomized algorithms?
- What if the switches have limited buffers?
Other questions

• Active Queue Management

• Multiple objective functions sharing the fabric
  • Each flow may have its own goal

• Guarantees in presence of failures

• Implementation issues?

Feedback?

rachit @ cornell