CS 181AG
Lecture 19

Output Scheduling

Arthi Padmanabhan

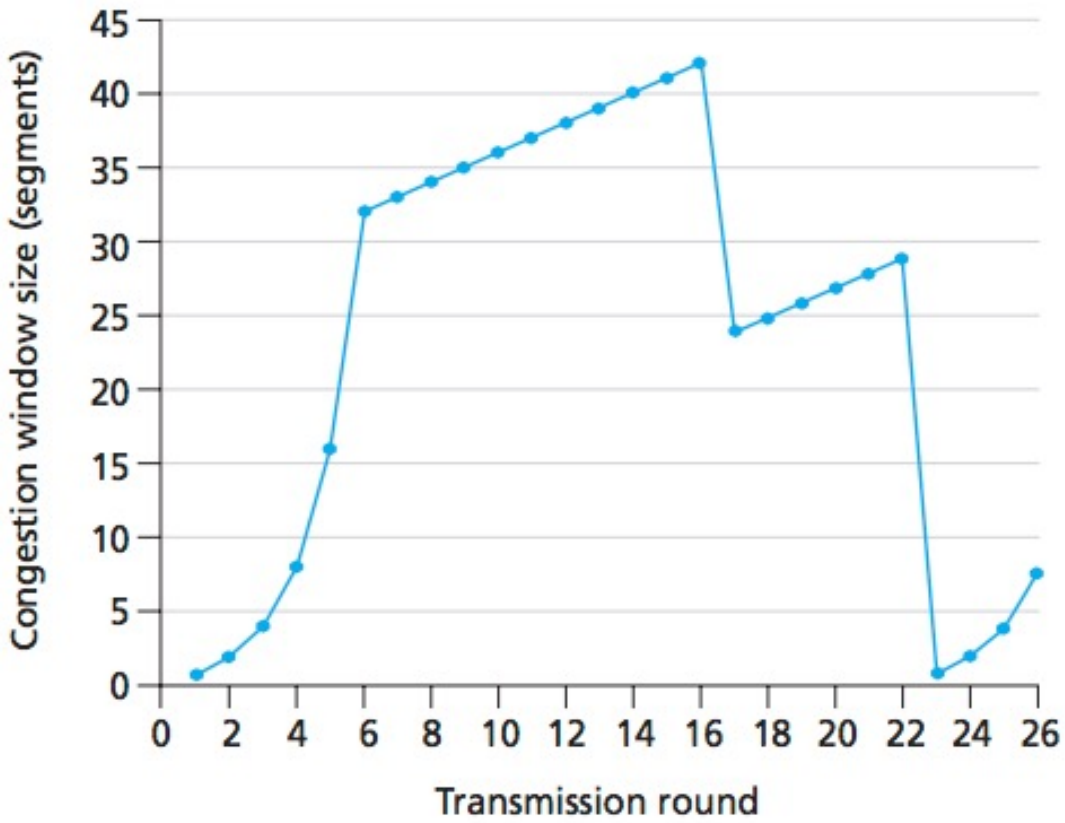
Nov 16, 2022

Upcoming Schedule

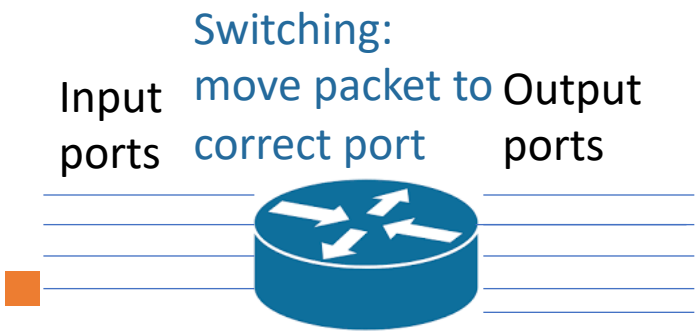
- Assignment 9 goes out today, due next Tuesday
- Assignment 8 (reading assignment) went out on Sunday, due on the last day of classes (Dec. 9)
- Last assignment goes out Nov 30, due Dec 7
- Midterm was returned this morning. If you have any questions, please schedule an appt with me (tomorrow and Friday mornings are blocked off for you)

Recap: TCP Congestion Control

- Window size is adjusted based on lost packets (how it is adjusted depends on whether it was a timeout or duplicate ACKs)

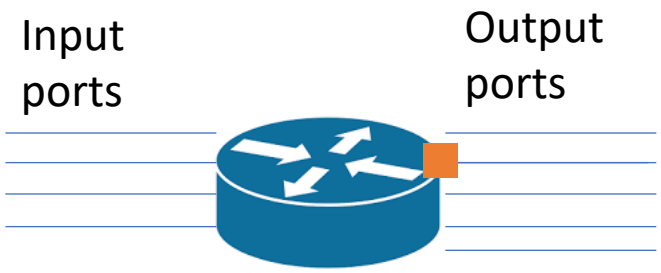


Big Picture: Router Functionality



Longest Matching Prefix to decide which output port, Packet Classification to decide matching rule for packet

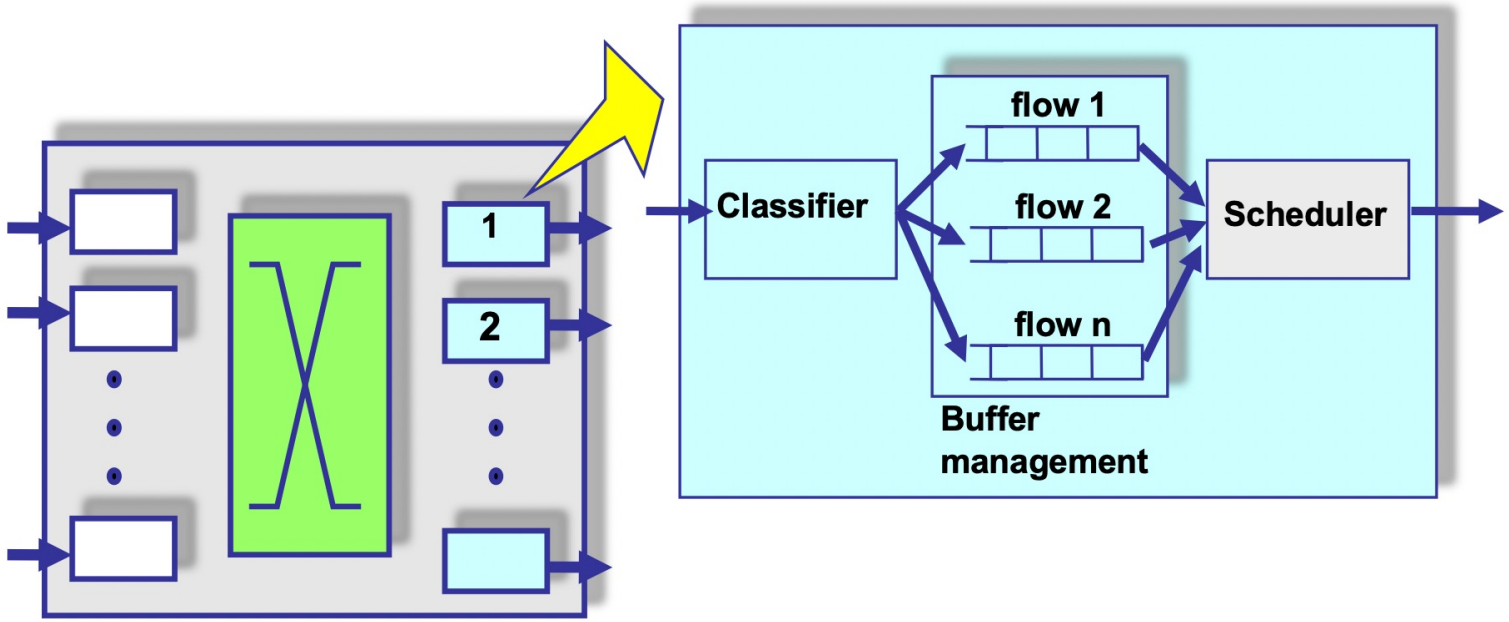
Big Picture: Router Functionality



Schedule packets in output port queue

Output Scheduling

- Flow: group of packets that follow the same path from source to destination



Buffer Management

- Simplest option: FIFO with tail-drop
- FIFO – scheduling policy
- Tail-drop – drop policy
- Properties
  - No separation between flows
  - No policing: send more packets = get more service
  - Upon full buffer, all subsequent packets are dropped

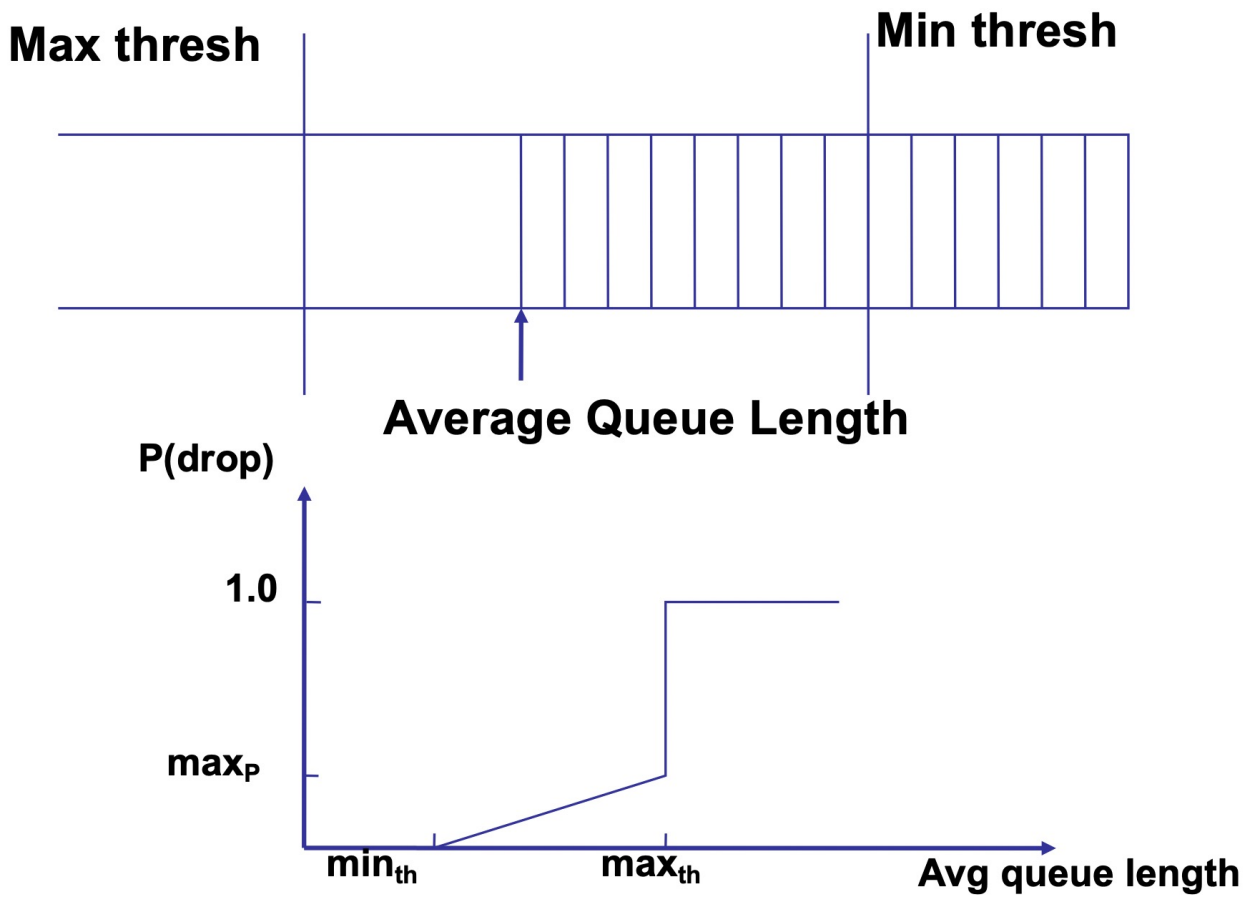
Tail Drop Alternatives?

- Why might we want to not simply drop all packets from the tail?
  - Remember how TCP congestion control adjusts window size

Random Early Detect (RED)

- Maintain running average of queue length
- If avg > thresh:
  - Drop incoming packets to queue with probability p
  - p increases linearly with avg until it reaches max thresh, when all incoming packets to queue are dropped
- Else:
  - Do nothing

Random Early Detect (RED)



Providing Bandwidth Guarantees

- We might want to limit certain type of traffic to a certain bandwidth
- Not all senders will pull back based on congestion – why?

Goals

- For a particular flow, we want an average rate of r bytes/sec with a maximum burst size of b bytes
- How can we do this?
  - You may think about this per-flow (one queue per flow)