Congestion Control
The Router's View

14-740: Fundamentals of Computer Networks
Bill Nace

Material from Computer Networking: A Top Down Approach. J.F. Kurose and K.W. Ross

traceroute

- Queuing Disciplines
- Random Early Drop (RED) Gateways

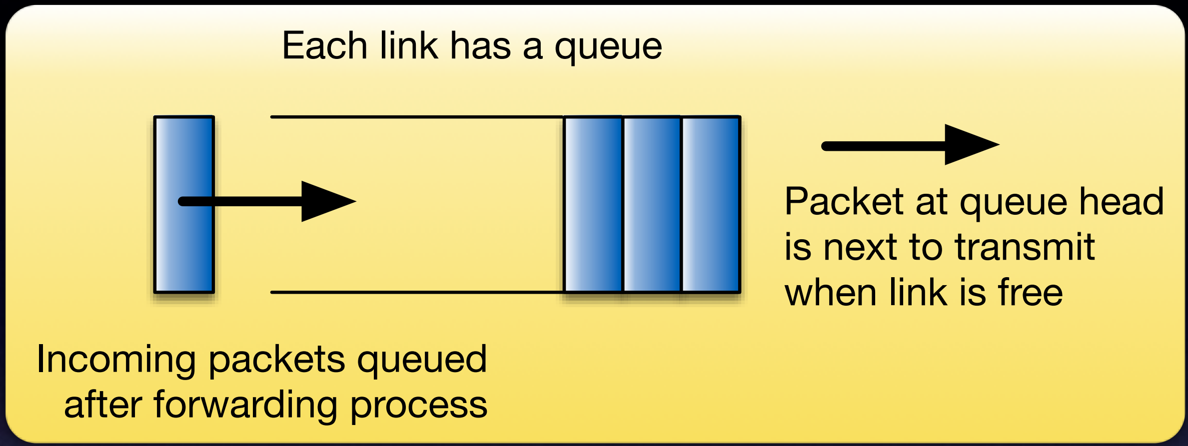
Queuing

- Queue: A buffer with packets awaiting transmission
- Queuing algorithm determines
  - Packet scheduling: ordering of transmission
    - Allocates bandwidth
  - Drop policy: which packet gets discarded
    - Allocates buffer space
- Affects latency experienced by a packet

Queuing Algorithms

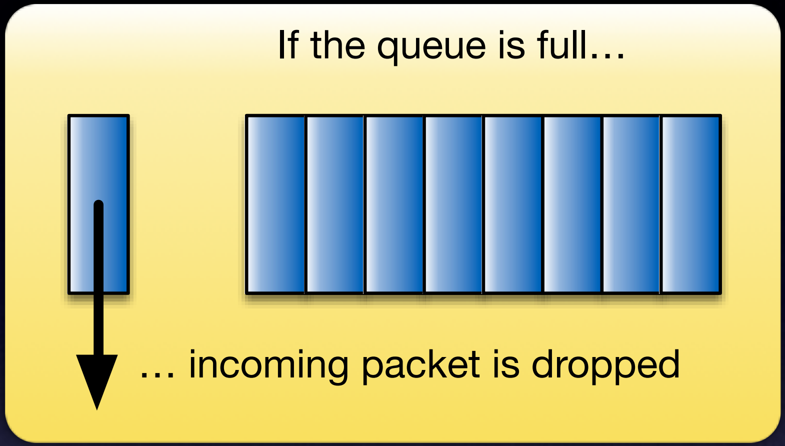
- FIFO
- Priority Queueing
- Round Robin Queuing
- Fair Queuing (FQ)
- Weighted Fair Queuing (WFQ)

First in, First Out



- No priority scheme
  - Importance or source of the packet doesn't matter
- FIFO is a packet scheduling algorithm
  - It determines the order of transmission

Drop Tail

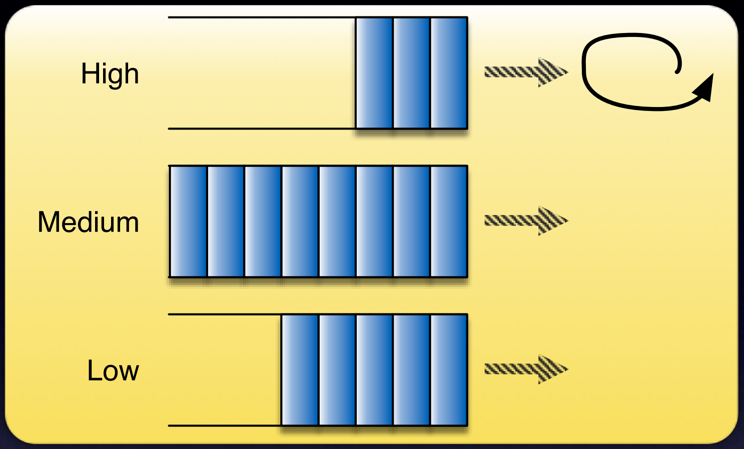


- Drop Tail is a drop policy
  - It determines which packet is dropped
- "FIFO with drop tail" is the simplest of all packet scheduling algorithms/drop policies
  - Most widely used in Internet routers

FIFO-DT Problems

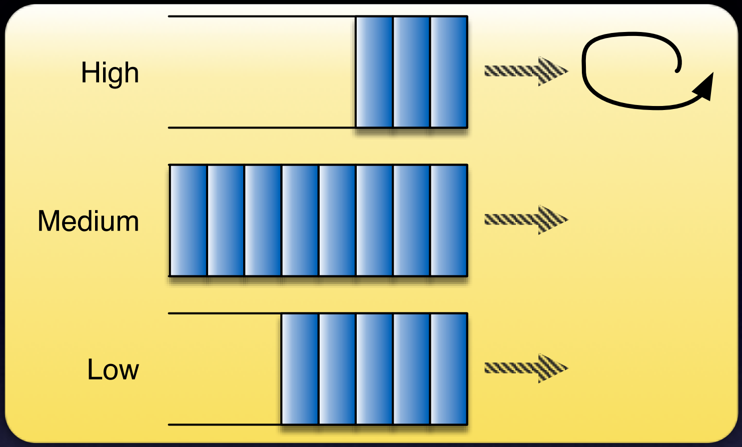
- Does not discriminate – packets from all flows go in the same queue
  - Importance or source of the packet doesn’t matter
- Relies on end-hosts to implement congestion control – but does not police the sources
  - An ill-behaved end-host can send as fast as possible – causing loss in other flows

Priority Queuing



- FIFO + label each packet with a priority
- Router uses one queue for each priority
  - Sends from highest priority queues first
  - Drops from lowest priority queues first

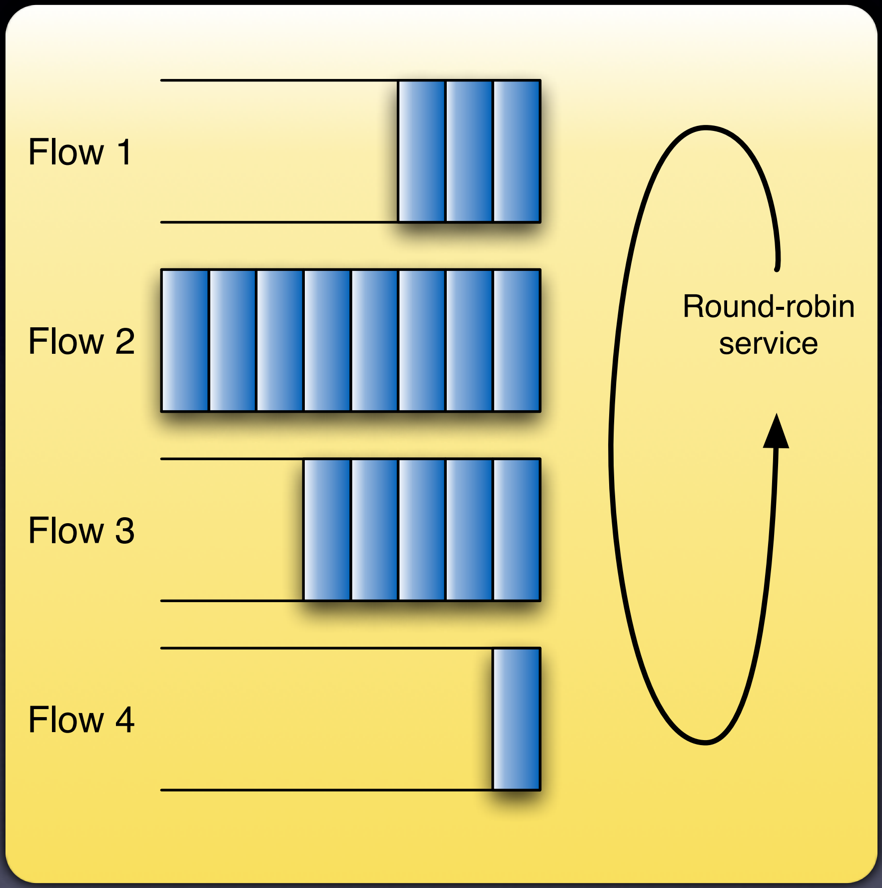
Priority Queuing



- Problem: High priority will starve low priority
  - Economics (i.e. $$) could solve, but Internet is decentralized

Round Robin Queuing

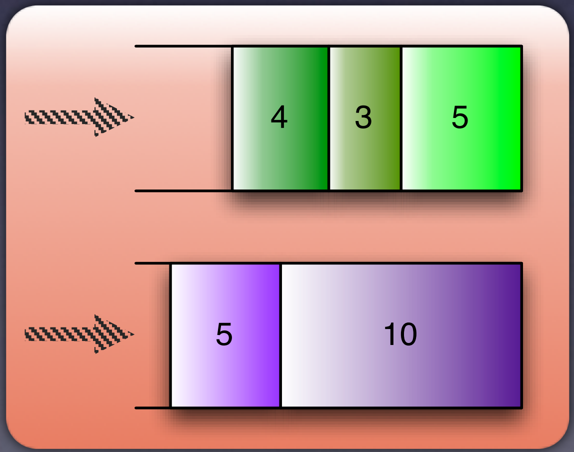
- Separate queue for each flow
- Router services each queue in turn (round-robin)
- If a flow sends too fast, its own queue would fill up
  - Drop the packets?
  - Separate drop policy can be implemented

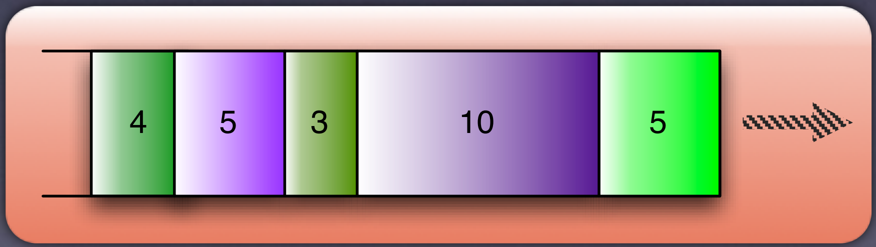


- Multiple queues doesn't necessarily mean more wasted space

Fair Queuing (FQ)

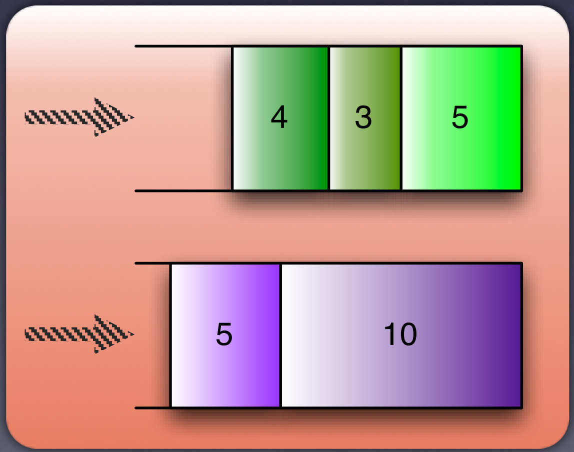
- Problem with Round Robin: Packets are not same length
- A flow with bigger packets will get more bandwidth in a packet-by-packet round-robin router

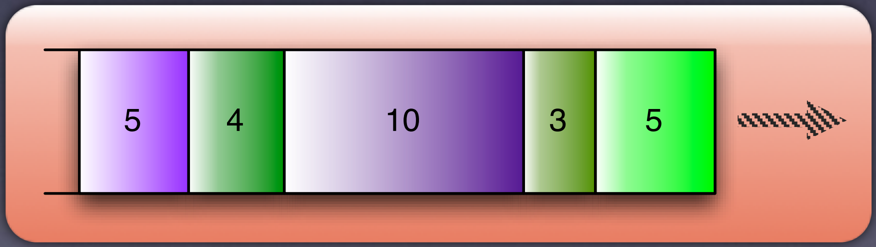




FQ Algorithm

- Desired: Bit-by-bit round-robin
  - But cannot send a bit at a time!
- Calculate transmission finishing time for each packet
- Transmit in order of finishing time





FQ Characteristics

- Work conserving: Link is never left idle if there is at least one packet in a queue
- Effectively share a link with multiple flows
  - For n flows, each uses 1/n of bandwidth
- Achieves max-min fairness
  - Maximizes the min data rate of any flow
  - ... thus no starvation of any flow

Weighted FQ

- Assigns a weight (priority) to each queue
  - From where? Manual config or source signaling
- Weight is proportional to number of bits to transmit each time the router services that queue
  - FQ same as WFQ with weight = 1 for each queue
- Not exactly reservation-based, but can be used as a component of reservation-based resource allocation
  - Differentiated Services Architecture

traceroute

- Queuing Disciplines
- Random Early Drop (RED) Gateways

Congestion Control

- TCP *controls* congestion once it happens
  - Strategy: increase load to find the point where congestion happens
    - Then back off
    - Rinse, repeat
  - TCP needs to create loss to discover available bandwidth

Congestion Avoidance

- Can we *avoid* congestion before it occurs?
  - Sender could reduce send rate just before packets start being dropped
  - Router knows congestion state (based on queue lengths) and could signal end-hosts
    - DECbit, ATM ABR, TCP ECN
    - Random Early Drop (RED) gateways

What is RED?

- Router-centric congestion control
  - But still leverages TCP end-hosts
  - Active Queue Management (AQM) scheme
- Decides which connection to notify of congestion
  - Randomly chooses a packet ...
  - ... before buffer overflow, thus early
  - Chosen packet is dropped to inform sender
- Also can mark with explicit congestion notification (ECN) flag without generating packet loss

RED Design Guidelines

- Goals
  - Reduce persistent queueing delay
  - Reduce unnecessary packet drops in some cases
- Control average queue size
  - Stay to left of “knee point” – region of high throughput and low delay

Design Guidelines (2)

- Avoid global synchronization
  - Don’t want all connections to back off and increase at the same time
- Avoid bias against bursty traffic which otherwise uses low bandwidth
  - Don’t want to drop packets mostly from bursty connections

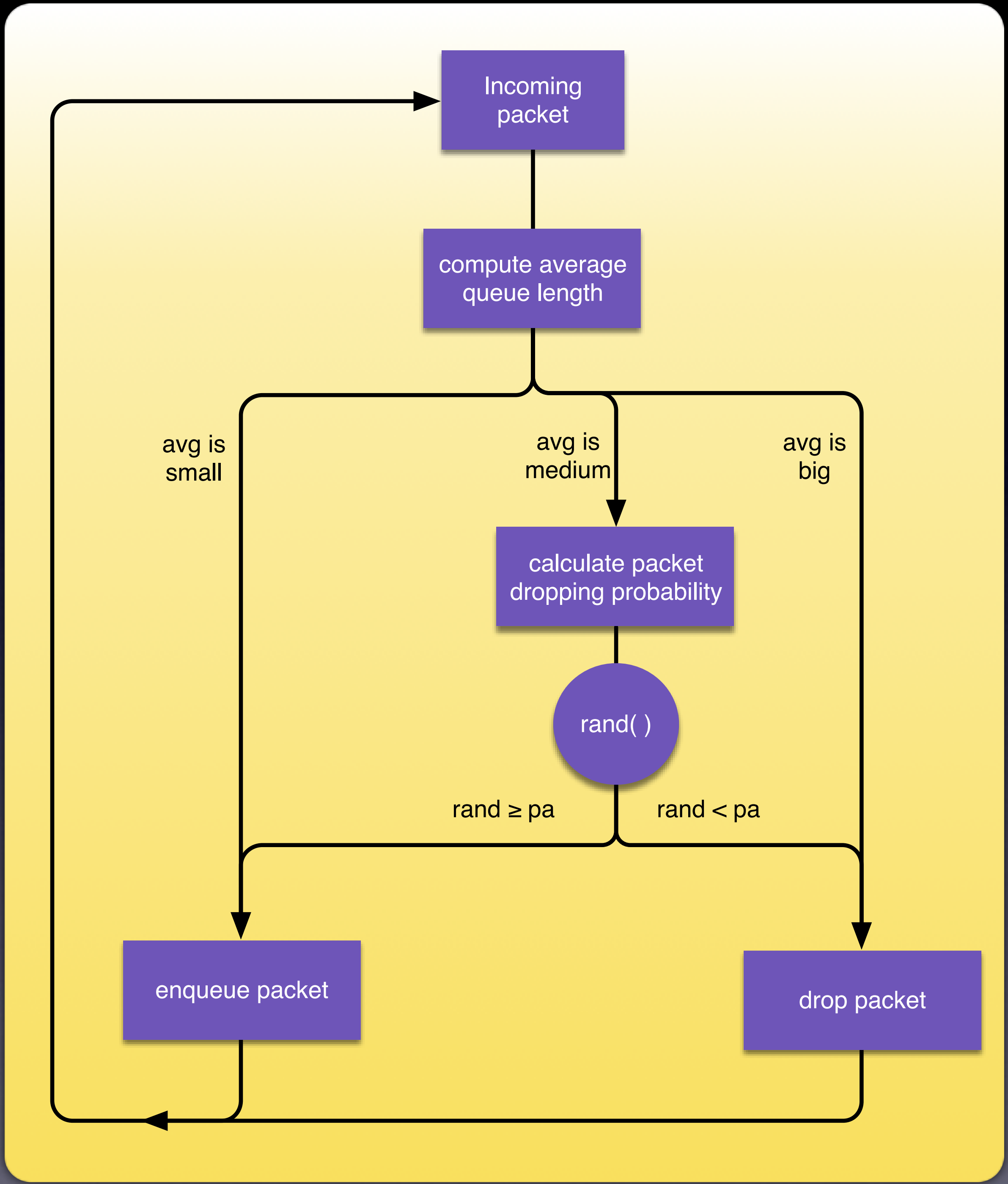
Algorithm

```
for each packet arrival
  calculate average queue size avg
  if avg < min_th
    add packet to queue
  if max_th ≤ avg
    mark(drop) the arriving packet
  if min_th ≤ avg < max_th
    calculate probability p_a
    rand < p_a?  if so:
      mark the arriving packet
```

Another look

Details:
“Compute”
“Calculate”
How?

How big is “big?”



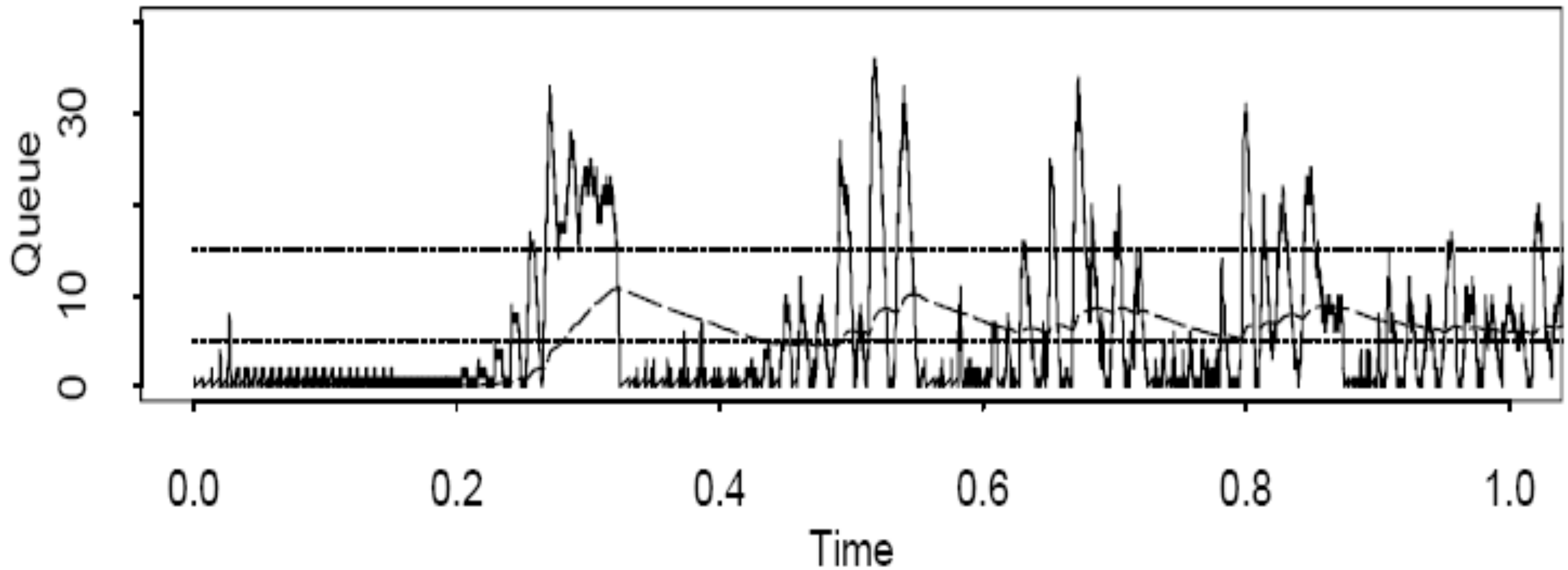
Compute Avg Queue Length

- Determines degree of burstiness that will be allowed in the router queue
- Use our old friend EWMA

  $avg \leftarrow (1-\beta) \cdot avg + \beta \cdot q\_length$
- Short-term increases in queue size do not result in significant increase in avg
- Focuses on long-lived congestion

Example

- Queue size vs average queue size



Queue size (solid line) and average queue size (dashed line).

Short Term Variations?

- Why filter out the short-term variations?
- Isn't it good to react to congestion ASAP?

How big is big?

- Set some thresholds
- min_{th}: if avg queue length is less than this, there is no congestion
- max_{th}: if greater than this, there is congestion
- if avg is between min_{th} and max_{th}, then there is growing concern

Calculate Probability

- Want to mark (i.e. drop) packets if the queue length is long ...

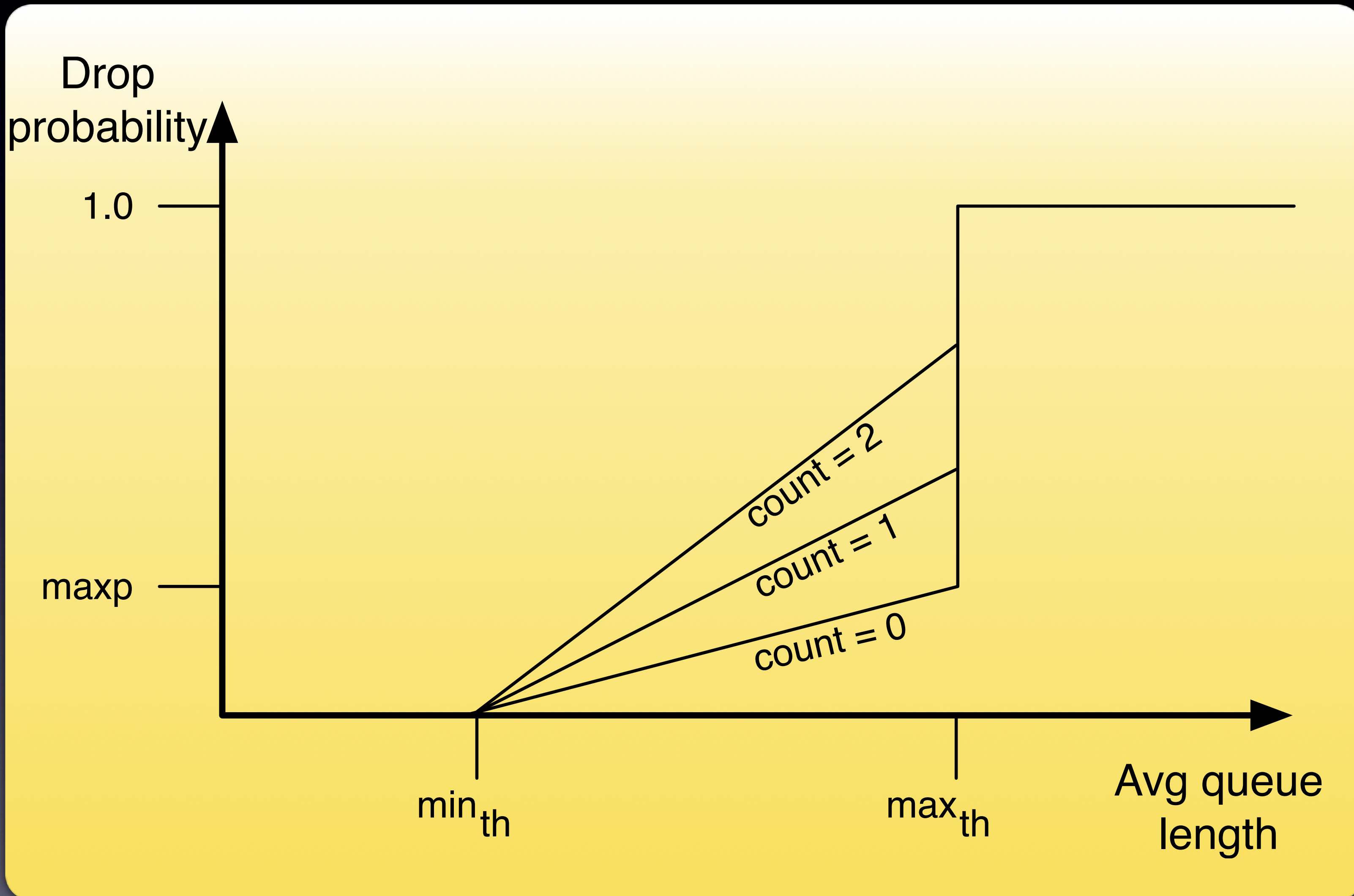
  $p_b \leftarrow max_p \cdot (Avg - min_{th}) / (max_{th} - min_{th})$

- ... and not in clusters

  $p_a \leftarrow p_b / (1 - count \bullet p_b)$

- count is the number of packets not dropped while Avg has been medium
- p_a is the actual drop probability

Drop Probability



What is wrong with Drop Tail?

- RED specifically tries to keep the router from Drop Tail regime
  - Random and Early
- Biased against bursty connection
  - When packets from a bursty connection arrive, highly likely the queue will overflow – dropping those packets
- Global synchronization
  - Drop packets from multiple connections at the same time – causing them all to enter slow start

Is RED Fair?

- Probability of a particular flow’s packets being dropped is roughly proportional to the share of bandwidth that flow is getting at the router
- But an ill-behaved flow is not limited to its fair share
  - Can identify bad flows, but additional mechanisms on top of RED needed to deal with them

Problems with RED

- Relies on end-hosts reacting to ECN or packet loss
- Unresponsive flows may ...
  - Ignore RED signals
  - Use more than fair share of bandwidth
- Different application and traffic mix than in 1993

Scaling problem

- Floyd93 used simulations of small networks to justify RED algorithm
- But, when scaling to large network, large propagation delays cause RED to update estimated avg queue length too slowly
  - New variant: SPRED has emerged

RED Status

- How widely is RED deployed in the Internet?
  - Fairly common in modern routers
- Research Extensions
  - Weighted RED
  - Adaptive / Active RED (ARED)
    - infers if RED should be more / less aggressive based on observations of avg queue length

Lesson Objectives

- Now, you should be able to:
  - describe the differences between packet scheduling algorithms and drop policies
  - describe the algorithm and issues with the following queueing algorithms: FIFO, Priority Queueing, Round Robin, Fair Queuing and Weighted Fair Queueing
  - analyze a scenario using one of the following queueing algorithms: FIFO, Priority Queueing, Round Robin, Fair Queuing and Weighted Fair Queueing

- You should be able to:
  - describe the opportunities for a router to do congestion control
  - describe the goals and details of the RED Gateway algorithm, as well as its advantages when compared to FIFO or other queueing algorithsm
  - calculate average queue length and drop probability as the RED algorithm does