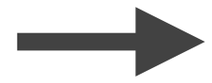
6.033 Spring 2019

Lecture #12

- **In-network resource management**
  - **Queue management schemes**
  - **Traffic differentiation**

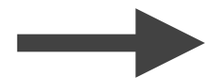
Internet of Problems

How do we **route** (and address) scalably, while dealing with issues of policy and economy?



BGP

How do we **transport** data scalably, while dealing with varying application demands?



TCP,
in-network resource management

How do we **adapt** new applications and technologies to an inflexible architecture?

problem: TCP reacts to drops, and packets aren't dropped until queues are full

Queue Management

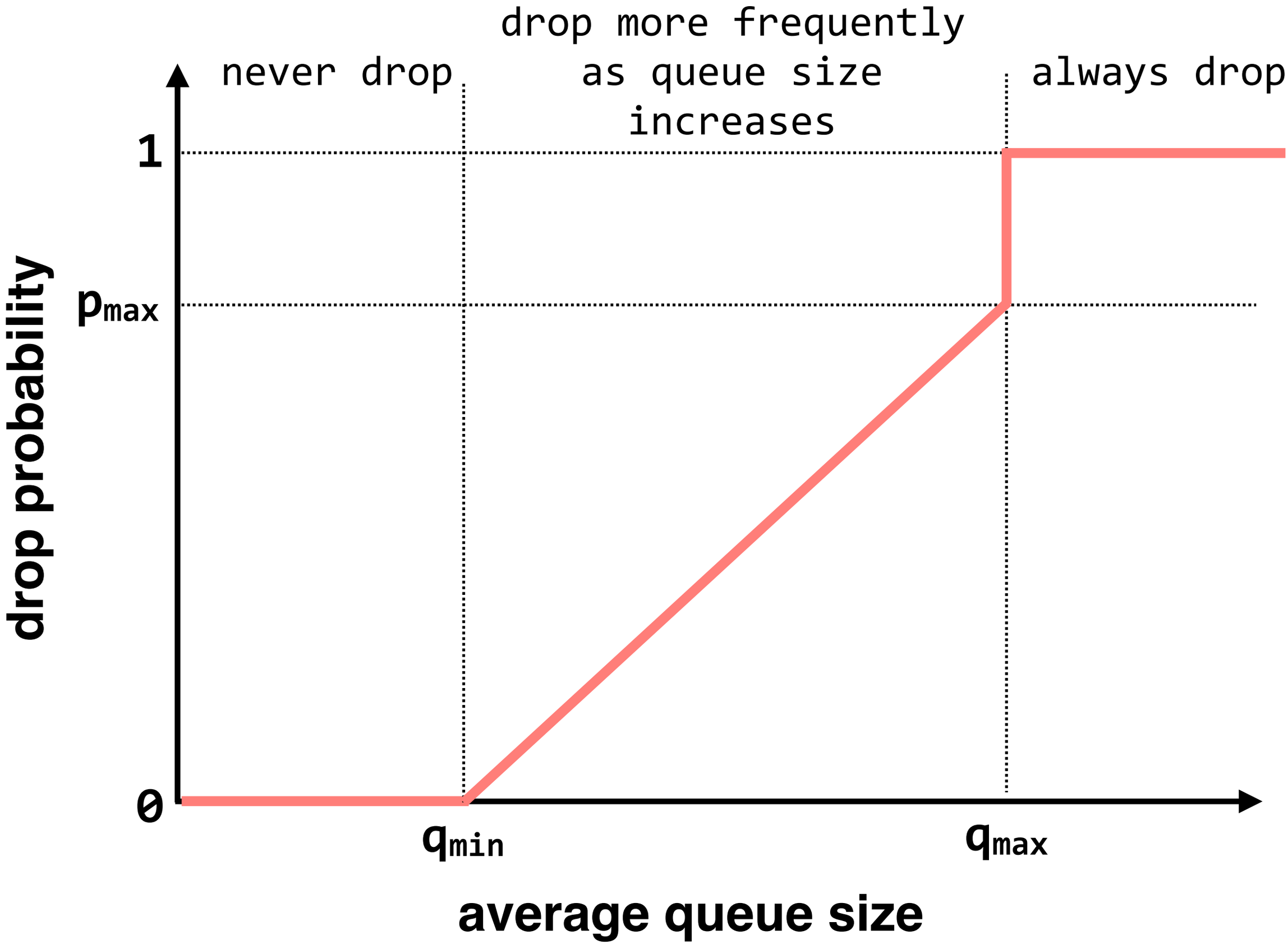
given a queue, when do we drop packets?

1. droptail

drop packets only when the queue is full. simple, but leads to high delays and synchronizes flows.

2. RED

drop packets before the queue is full


never drop
drop more frequently
as queue size
increases
always drop
1
p_{max}
0
drop probability
q_{min}
q_{max}
average queue size

Queue Management

given a queue, when do we drop packets?

1. droptail

drop packets only when the queue is full. simple, but leads to high delays and synchronizes flows.

2. RED

drop packets before the queue is full: with increasing probability as the queue grows. prevents queue lengths from oscillating, decreases delay, flows don’t synchronize

Queue Management

given a queue, when do we drop (or mark) packets?

1. droptail

drop packets only when the queue is full. simple, but leads to high delays and synchronizes flows.

2. RED (drops) / ECN (marks)

drop (or mark) packets before the queue is full: with increasing probability as the queue grows. prevents queue lengths from oscillating, decreases delay, flows don’t synchronize, but complex and hard to pick parameters

what if we want to give latency guarantees to certain types of traffic?

(or at least try to prioritize latency-sensitive traffic)

Delay-based Scheduling

how could we give latency guarantees for some traffic?

1. priority queueing

put latency-sensitive traffic in its own queue and serve that queue first. does not prevent the latency-sensitive traffic from “starving out” the other traffic (in other queues).

what if we want to allocate different amounts of bandwidth to different types of traffic?

Bandwidth-based Scheduling

how can we allocate a specific amount of network bandwidth to some traffic?

1. round-robin

can't handle variable packet sizes (and in its most basic form doesn't allow us to weight traffic differently)

2. weighted round-robin

can set weights and deal with variable packet sizes

Weighted Round Robin

```
in each round:

   for each queue q:
      q.norm = q.weight / q.mean_packet_size

   min = min of q.norm's over all flows

   for each queue q:
      q.n_packets = q.norm / min
      send q.n_packets from queue q
```

Bandwidth-based Scheduling

how can we allocate a specific amount of network bandwidth to some traffic?

1. round-robin

can't handle variable packet sizes (and in its most basic form doesn't allow us to weight traffic differently)

2. weighted round-robin

can set weights and deal with variable packet sizes, but needs to know mean packet sizes

3. deficit round-robin

Deficit Round Robin

```
in each round:
   for each queue q:
      q.credit += q.quantum
      while q.credit >= size of next packet p:
         q.credit -= size of p
         send p
```

Bandwidth-based Scheduling

how can we allocate a specific amount of network bandwidth to some traffic?

1. round-robin

can't handle variable packet sizes (and in its most basic form doesn't allow us to weight traffic differently)

2. weighted round-robin

can set weights and deal with variable packet sizes, but needs to know mean packet sizes

3. deficit round-robin

doesn't need mean packet sizes. near-perfect fairness and low packet processing overhead

Delay-based Scheduling

how could we give latency guarantees for some traffic?

1. priority queueing

put latency-sensitive traffic in its own queue and serve that queue first. does not prevent the latency-sensitive traffic from "starving out" the other traffic (in other queues).

can solve this problem by doing something similar to bandwidth-based scheduling across the two queues

In-network Resource Management

Queue Management	switches can signal congestion before queues are full	`DropTail`
		`RED`
		`ECN`

Delay-based Scheduling	switches can prioritize latency-sensitive traffic	`Priority Queueing`

Bandwidth-based Scheduling	switches can enforce (weighted) fairness among different types of traffic	`Round-robin`
		`Weighted Round-robin`
		`Deficit Round-robin`

in-network resource management: a good idea?

- Active **queue management schemes**, such as **RED** or **ECN**, drop or mark packets before a queue is full, in hopes of getting TCP senders to react earlier to congestion. They are difficult to get to work on the Internet-at-large, but the ideas can be useful in other types of networks.

- **Traffic differentiation** requires a scheduling discipline, such as **weighted round robin** or **deficit round robin**. The goal of these schemes is to give weighted fairness in the face of variable packet sizes while having low processing overhead

- Both of these are examples of **in-network resource management**