6.033 Spring 2016
Lecture #11

- 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?

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

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

→ BGP

→ TCP, in-network resource management
problem: TCP reacts to drops, and packets aren’t dropped until queues are full
solution (?): drop packets before queues are full
The diagram illustrates the relationship between the average queue size and the drop probability. The x-axis represents the average queue size, while the y-axis represents the drop probability. For queue sizes below $q_{min}$, the drop probability is 0, indicating that packets are never dropped. As the queue size increases towards $q_{max}$, the drop probability increases linearly until it reaches 1, indicating that packets are always dropped. The drop probability is always 0 for queue sizes below $q_{min}$ and always 1 for queue sizes above $q_{max}$. The maximum drop probability, $p_{max}$, is reached at $q_{max}$. The diagram shows how packet drops become more frequent as the queue size increases.
are RED and ECN better than DropTail?
what if we want to give latency guarantees to certain types of traffic?
(or at least try to prioritize latency-sensitive traffic)
what if we want to allocate different amounts of bandwidth to different types of 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
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
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
in-network resource management: a good idea?
• **Queue management schemes**
  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**
  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**