Prioritization of Flows

- **Goals:**
  - prioritization
  - independent of packet length
  - simple to implement

- Let’s start with fair scheduling of equal-priority flows:
  - single queue (shared fate)
  - idea: a queue per flow
Prioritization of Flows

- Equal-priority flows:
  - multiple per-flow queues served in round-robin fashion

- what if flows consist of packets of consistently different lengths?
Prioritization of Flows

- Need a method to account for the amount of traffic sent and serve queues in an order that reflects the amount of traffic sent
  - this is easier said than done

- The solution is based on theoretical approach called Processor Sharing (PS):
  - assuming packets can be fragmented into small equal-size fragments
  - if we serve packet fragments in round-robin fashion, there is no packet size bias: A B1 A B2 A B3
  - except that we cannot break packets into small fragments (!)
Fair Queuing (FQ)

- **Simulate** Processor Sharing to **find the order** of finish times of packet transmissions

- **Schedule** (full) packets in that **order**
  - surprisingly, this can be done by a simple algorithm
Weighed Fair Queueing

- **Fair Queuing** does not support prioritization

- **Idea:**
  - **Generalized Processor Sharing (GPS):** adjust fragment sizes of PS to reflect priority of the flow
  - **Weighted Fair Queuing (WFQ):** use simulation of GPS to schedule packets
Deficit Round Robin

- An improvement on WFQ
- Each queue has a **deficit counter**
- Queues with deficit counter values higher than the packet length are served in round robin fashion (and the deficit counter is reduced accordingly)
- A *quantum* is added to deficit counter of a queue that is skipped
- Complexity: $O(1)$ vs $O(\log N)$ for WFQ
Random Early Detection

- TCP flow control
  - packet loss triggers back-off (rate reduction)
  - it takes time to recognize that packet was lost (network latency, timeouts)

- Possible outcome network synchronization
  - periods of congestion followed by periods of low load caused by a TCP flows backing off

- Solution: 1993 Sally Floyd
  - RED (Random Early Detection)
RED - Goals

- **Goals:**
  - Avoid congestion and global synchronization
  - Avoid bias against bursty traffic
  - Bound on queuing delay

- **Method**
  - calculate *average queue size*
  - set two thresholds ($TH_{max}$ and $TH_{min}$) within the queue size
  - enqueue or drop packets based on the relation between the average queue size and the thresholds