2/11/2010

Queuing: FCFS

Ideally,

Role of router in congestion control: many, but here are a few:
- Actively monitor onset of congestion
- Provide early indication of congestion
- Isolate misbehaving flows so that end-host congestion control is effective - or unresponsive

"Active Queue Management (AQM)"

- FCFS → does this really well.
- Others that do subsets of these → discuss in the next class.

How to isolate: ideal → max-min fairness; efficient use of resources → also buffer and delay allocation.

- FEES: instantiates all three goals and measures against outcomes above
- Router backplane switching does this using starvation-free fair scheduling: simple, efficient, but...
  - Too granular: focuses on input ports as opposed to flows.
  - Parking lot problem where flows that traverse multiple routers get exponentially lesser share.

Question: How to build a fair mechanism that is easy to implement/use?

Nagle's idea: Instead of links, allocate bandwidth by isolating flows.

- Randomly select an active queue
- Fairly divide available rate
- Add to selected queue
- Roundrobin list of active queues

Bad:
- # Queues = # flows ever seen (provision queues for worst case)
- Unfairness = relative proportions of new shares received
  \[ \text{Maximum} - \text{Minimum} \rightarrow \infty \]
  \[ \text{Min} \rightarrow \infty \]

Good:
- Work complexity = \(O(1)\) enqueueing
- \(O(1)\) dequeueing

\[ \text{Many} \]
FA proposal of DKS:
- Fixes the unfairness problems
- Buffer allocation and incentives / disincentives
- But somewhat complex to implement
- Other elegant features
- Simulate bit-by-bit RR.

Upon packet arrival, calculate finish time and insert into a sorted queue based on finish times.
- per flow state

\[
\text{Finish Time} = \text{size} + \max(\text{current time}, \text{finish time of prior packet in flow})
\]

\[
\text{Finish Round} = \text{size} + \max(\text{current round}, \text{finish round in list})
\]

\[
\frac{dR(t)}{dt} \rightarrow \text{can change over time as the number of active flows grows and shrinks.}
\]

also \( R(t) \) is fractional. Packets arriving almost at the same time will see different \( R(t) \)

1. Insert \( O(\log m) \) \( m \) is the number of packets in the queue.

2. For each flow, track finish time of last packet in flow.
   \( O(\text{num active flows}) \)

FA Buffer management: Buffer has a certain maximum length.

What to do when buffer is full:
- "Buffer stealing" remove packet from flow with maximum buffer occupancy \( \frac{1}{O(\log n)} \) can be done in \( O(1) \) time.
- But don't update
Delay allocation with a \( T \)

**Important properties:**

1. Fair
2. Low delay for high flows
3. Good congestion control still important
4. Incentive to adopt good congestion control
5. Punishment to misbehaving flows

Can we achieve the same fairness as FA but at \( O(1) \) cost per packet, and lower space requirement, and low amount of state tracking.

RR of Nagle \( \Rightarrow O(1) \) but unfairness.

RR + deficit

**Quantum:** \( Q \)

**Deficit per queue:** \( D_i \)

At round \( i \), send \( B_i + D_i \) bytes from queue \( i \)

**Round Robin:** when servicing queue \( i \)

- Check if size of packet at head, \( H_i \), \( \leq B_i + D_i \)
- If yes, send \( B_i = \lceil H_i \rceil \) bytes
- If no, send \( B_i = 0 \)
- \( D_i \leftarrow B_i + D_i \)

**Difference between FA and DR:**

- \( FA \) - maximum fine-grained difference across flows \( \max \)
- \( DR \) - "" \( = 2 \max \)

**Another difference:**

**Complexity:**

- **Enqueue:** finding the queue, append to tail of queue
- **Dequeue:** \( \geq \max \Rightarrow \text{guaranteed to send on packet every} \)

**Next active update active queue:** counting time a queue is visited
Unfairness: \( s_{\text{MAX}} \)

Problem: Queues are way too many.

How to bound the number of Queues: SFG.

If the Hash index is larger than the number of active flows at any given point then fairness guarantees become static.

---

From the last lecture:

- Buffer sizing
- TCP Algebra
- TFRC

---

QOS