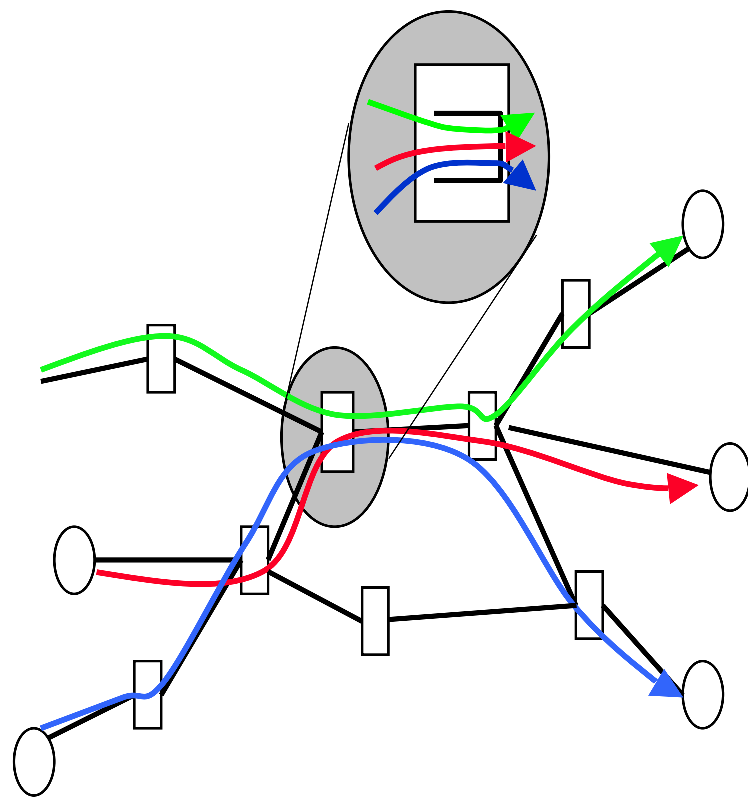
CS 268: Lecture 8 (Router Support for Congestion Control)

Ion Stoica

February 19, 2002

Router Support For Congestion Management

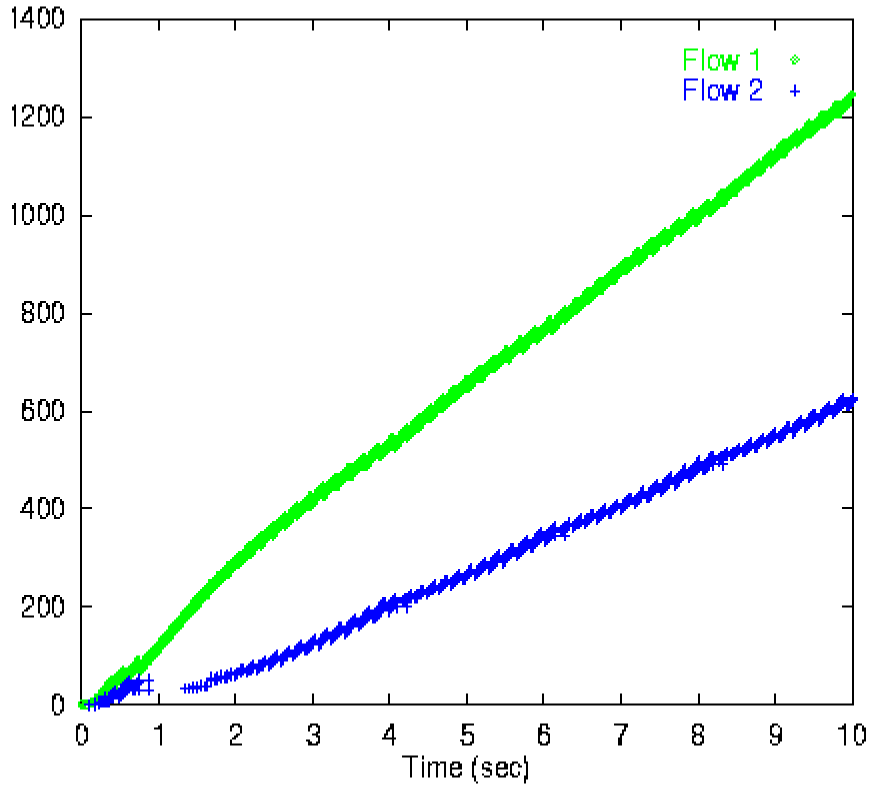
- Traditional Internet
  - Congestion control mechanisms at end-systems, mainly implemented in TCP
  - Routers play little role
- Router mechanisms affecting congestion management
  - Scheduling
  - Buffer management
- Traditional routers
  - FIFO
  - Tail drop

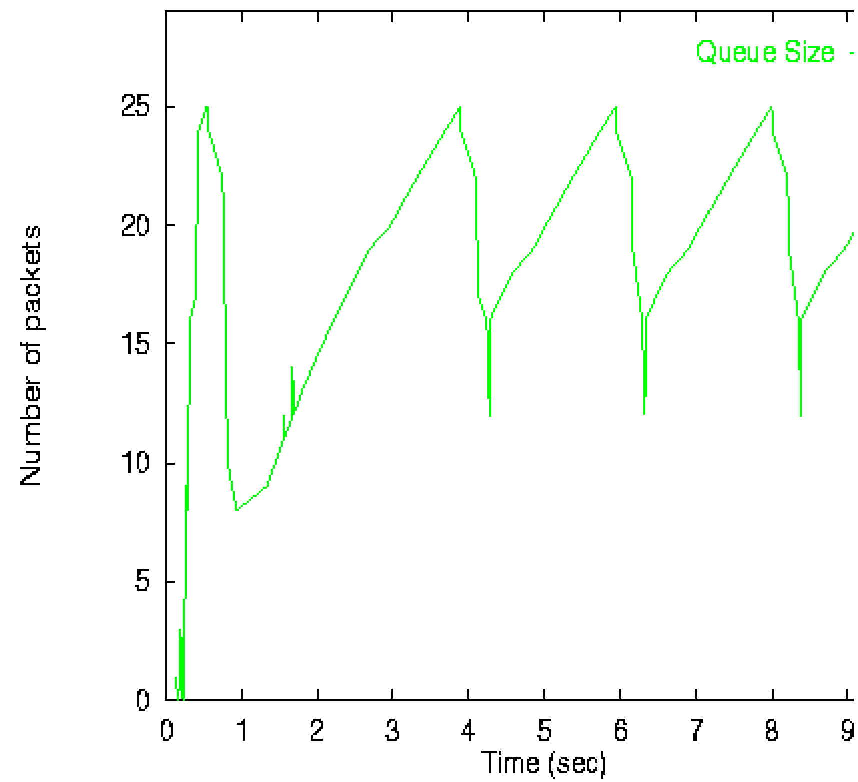


Drawbacks of FIFO with Tail-drop

- Buffer lock out by misbehaving flows
- Synchronizing effect for multiple TCP flows
- Burst or multiple consecutive packet drops
  - Bad for TCP fast recovery

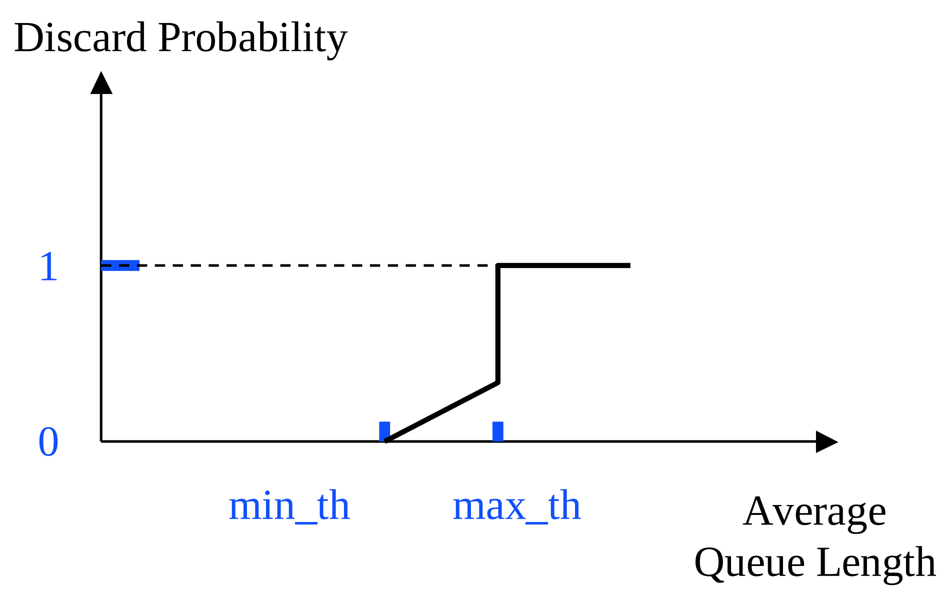
FIFO Router with Two TCP Sessions





Random Early Detection (RED) Routers (Floyd & Fall 93]

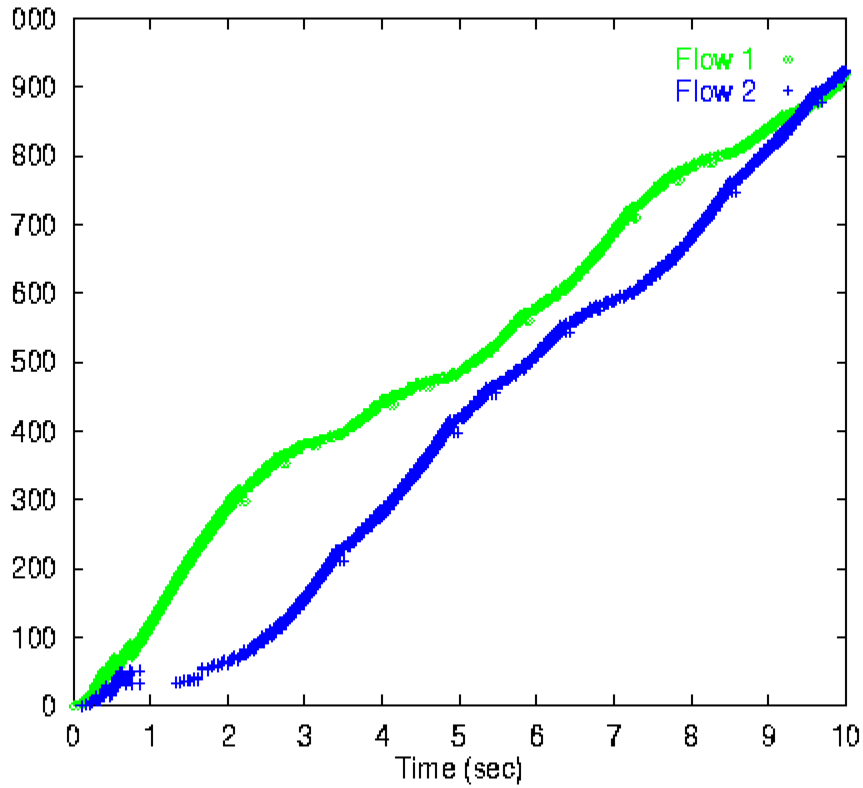
- Probabilistically discard packets
- Probability is computed as a function of average queue length (why average?)
  - Use exponential averaging

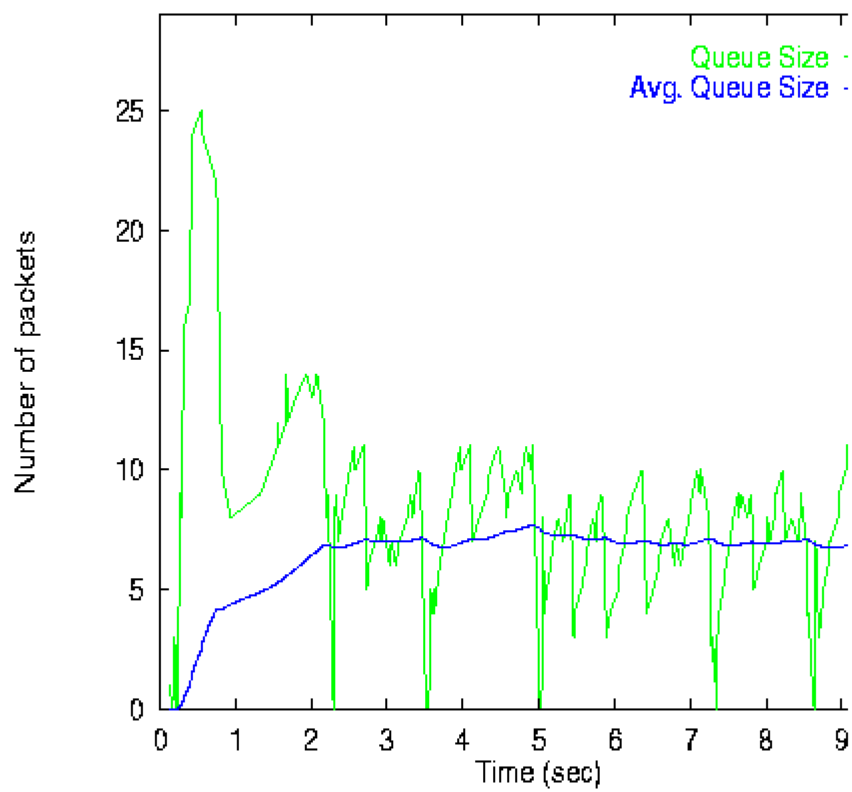


RED Advantages

- Absorb burst better
- Avoids synchronization
- Signal end systems earlier

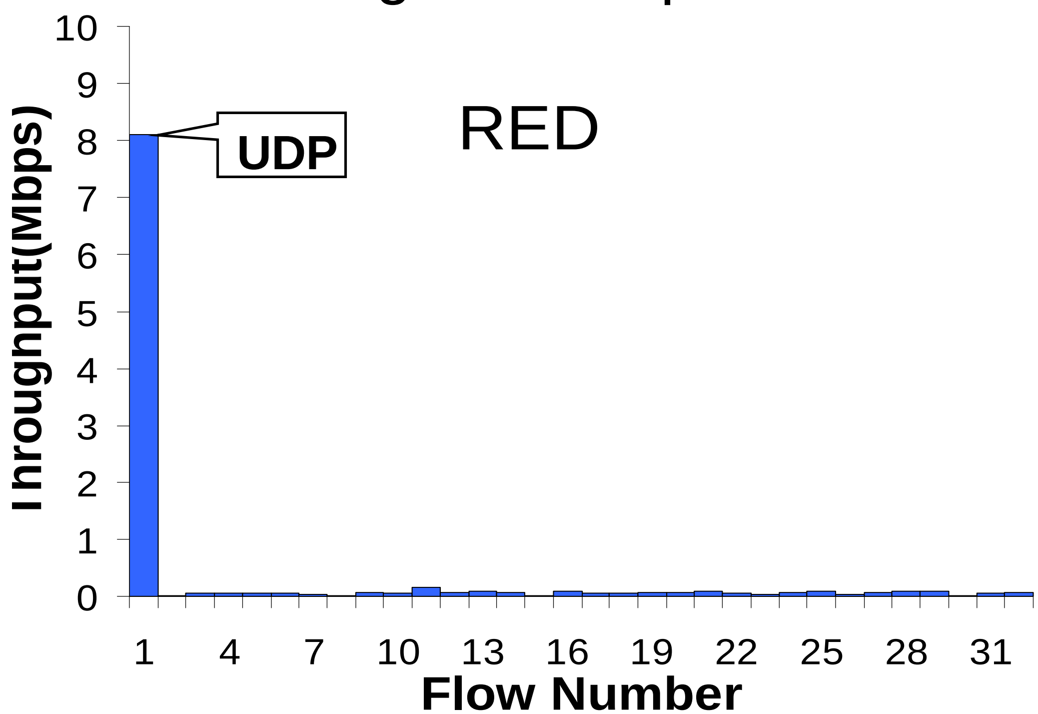
RED Router with Two TCP Sessions





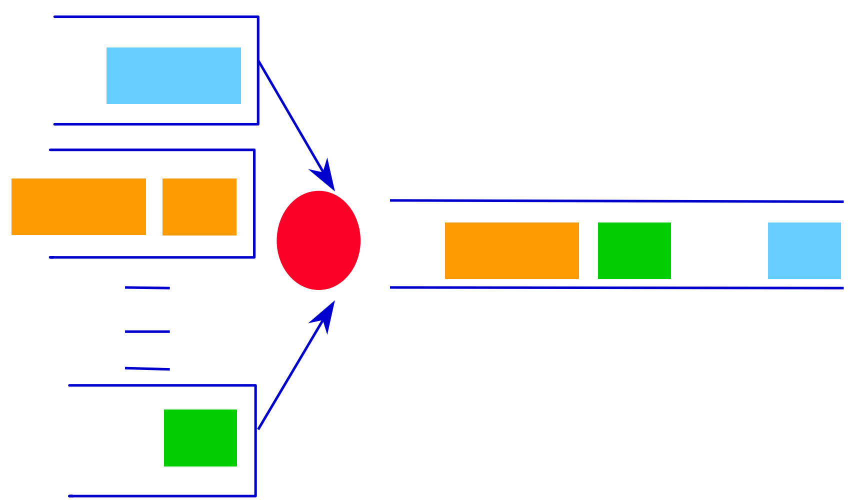
Problems with RED

- No protection: if a flow misbehaves it will hurt the other flows
- Example: 1 UDP (10 Mbps) and 31 TCP's sharing a 10 Mbps link



Solution?

- Round-robin among different flows [Nagle '87]
  - One queue per flow



Round-Robin Discussion

- Advantages: protection among flows
  - Misbehaving flows will not affect the performance of well-behaving flows
  - FIFO does not have such a property
- Disadvantages:
  - More complex than FIFO: per flow queue/state
  - Biased toward large packets – a flow receives service proportional to the number of packets (When is this bad?)

Solution?

- Bit-by-bit round robin
- Can you do this in practice?
- No, packets cannot be preempted (why?)
- …we can only approximate it

Fair Queueing (FQ) [DKS'89]

- Define a fluid flow system: a system in which flows are served bit-by-bit
- Then serve packets in the increasing order of their deadlines

- Advantages
  - Each flow will receive exactly its fair rate

- Note:
  - FQ achieves max-min fairness

Max-Min Fairness

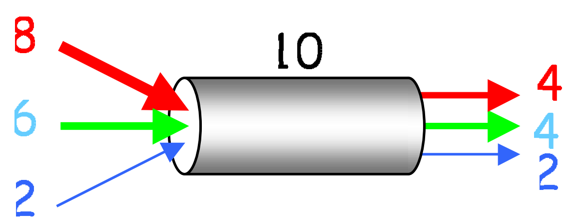
- Denote
  - C – link capacity
  - N – number of flows
  - r_i – arrival rate
- Max-min fair rate computation:
  1. compute C/N
  2. if there are flows i such that $r_i <= C/N$, update C and N

$$C = C - \sum_{i\ s.t\ r_i \leq C} r_i$$

  3. if no, $f = C/N$; terminate
  4. go to 1
- A flow can receive at most the fair rate, i.e., min(f, r_i)

Example

- $C = 10$; $r_1 = 8$, $r_2 = 6$, $r_3 = 2$; $N = 3$
- $C/3 = 3.33 \rightarrow C = C - r3 = 8$; $N = 2$
- $C/2 = 4$; $f = 4$

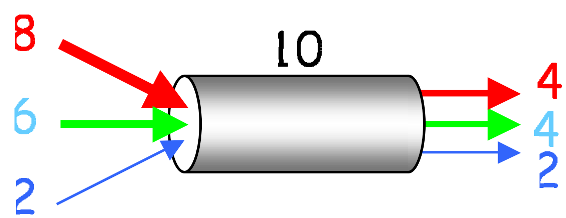


$f = 4$:
min(8, 4) = 4
min(6, 4) = 4
min(2, 4) = 2

Alternate Way to Compute Fair Rate

- If link congested, compute *f* such that

$$\sum_i \min(r_i, f) = C$$

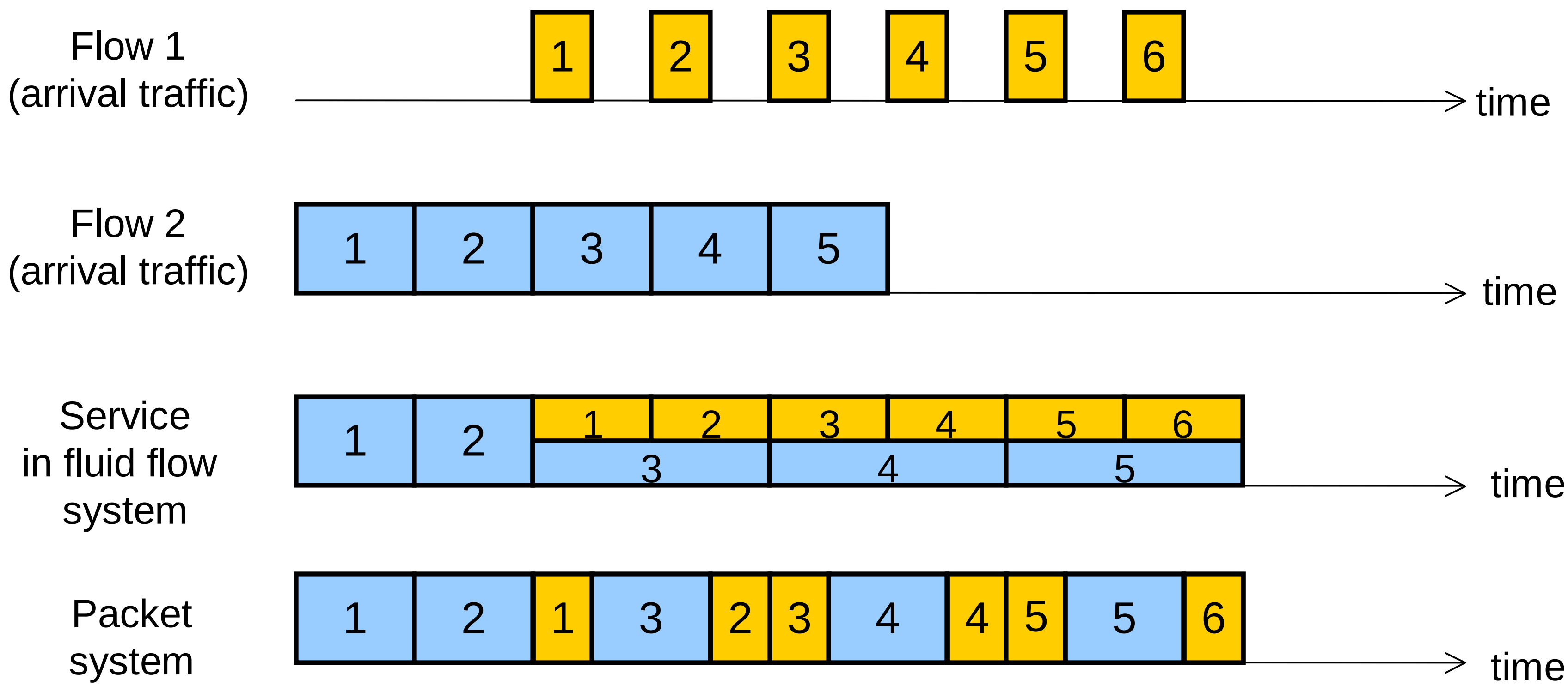


f = 4:
min(8, 4) = 4
min(6, 4) = 4
min(2, 4) = 2

Implementing Fair Queueing

- Idea: serve packets in the order in which they would have finished transmission in the fluid flow system

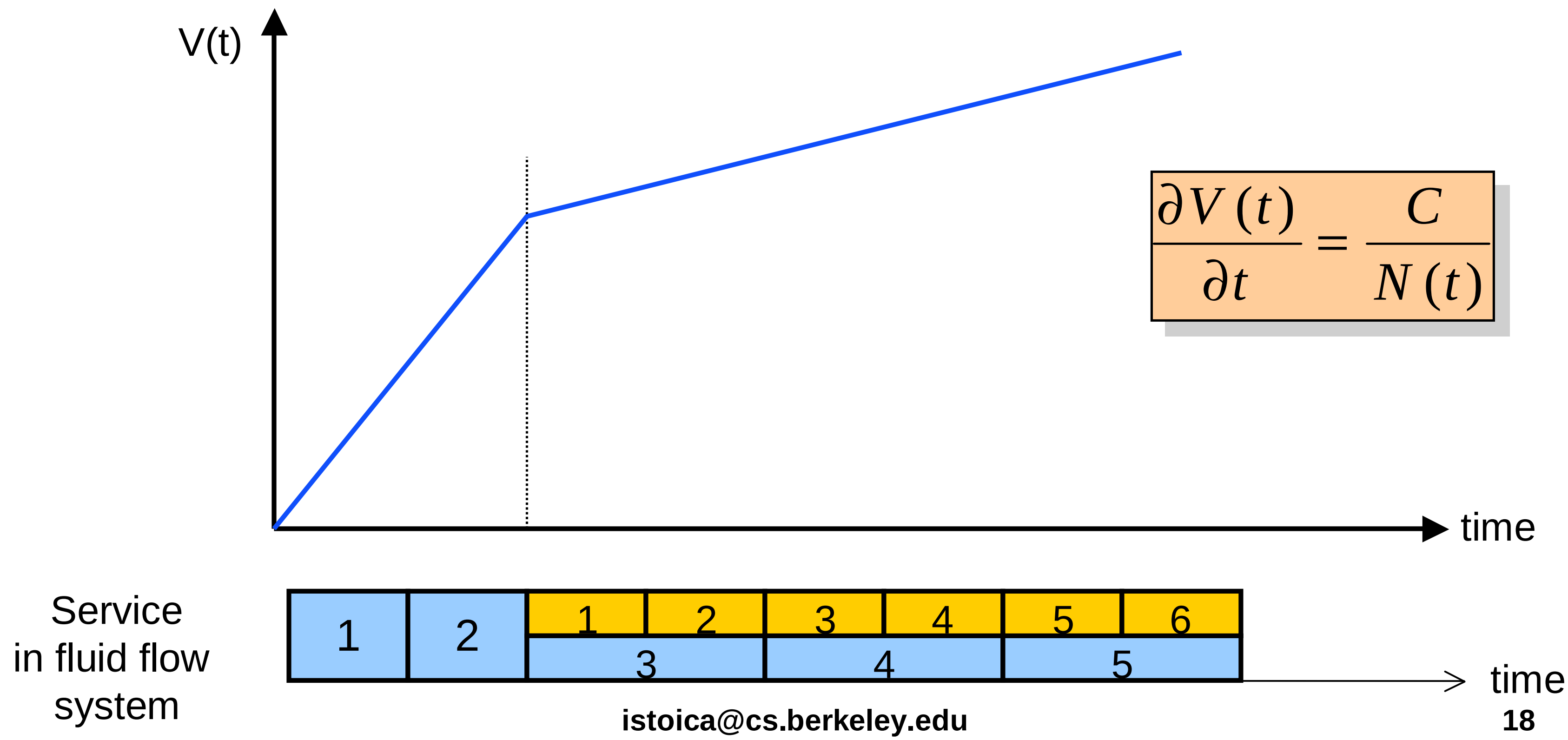
Example



Flow 1
(arrival traffic)
1
2
3
4
5
6
time
Flow 2
(arrival traffic)
1
2
3
4
5
time
Service
in fluid flow
system
1
2
1
2
3
4
5
6
3
4
5
time
Packet
system
1
2
1
3
2
3
4
4
5
5
6
time

System Virtual Time: *V*(t)

- Measure service, instead of time
- V(t) slope – rate at which every active flow receives service
  - *C* – link capacity
  - *N*(t) – number of active flows in fluid flow system at time t



$$\frac{\partial V(t)}{\partial t} = \frac{C}{N(t)}$$

Fair Queueing Implementation

- Define
  - F_i^k - finishing time of packet k of flow i (in system virtual time reference system)
  - a_i^k - arrival time of packet k of flow i
  - L_i^k - length of packet k of flow i

- The finishing time of packet $k+1$ of flow i is

$$F_i^{k+1} = \max(V(a_i^k), F_i^k) + L_i^{k+1}$$

“Weighted Fair Queueing” (WFQ)

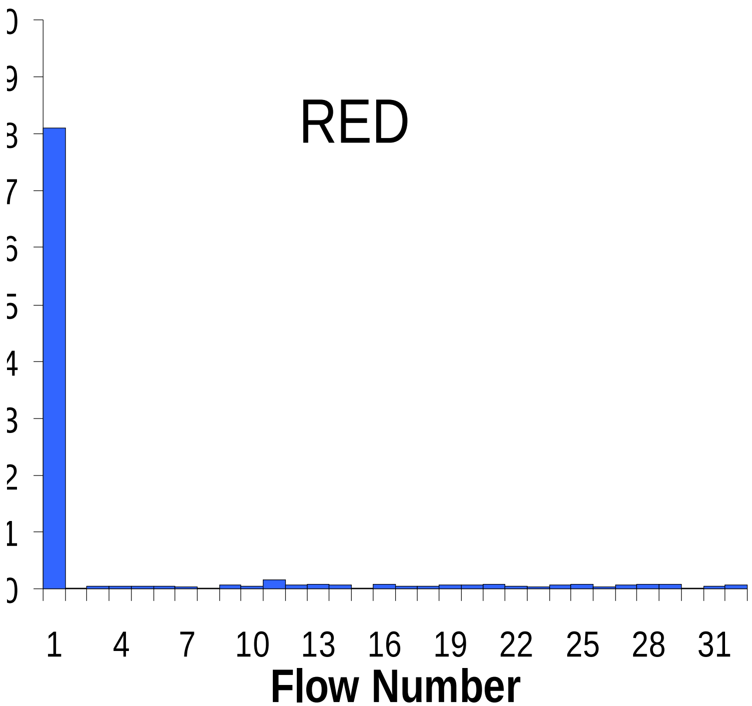
- What if we don't want exact fairness?
  - ex: file servers
- Assign weight w_i to each flow i
- And change virtual finishing time

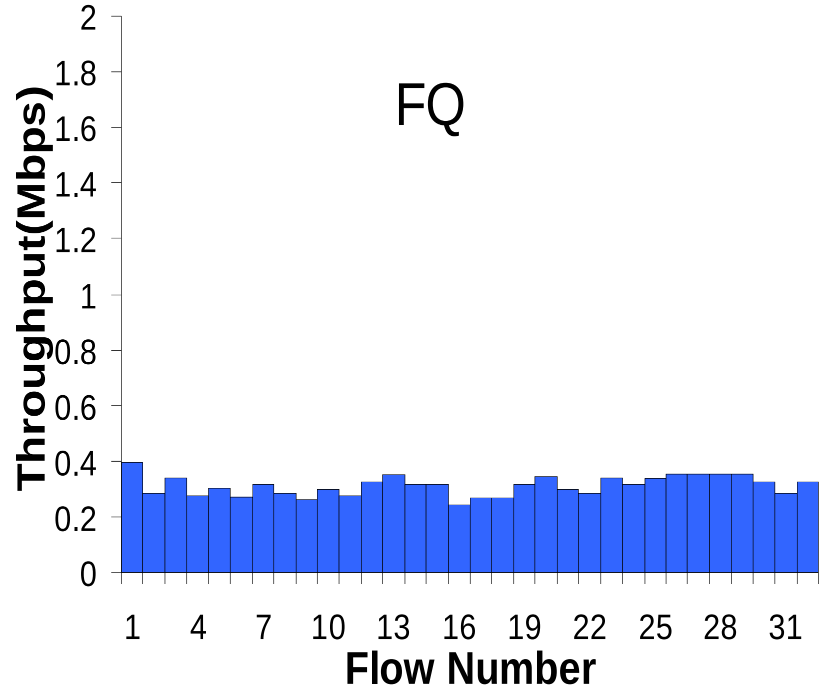
$$F_i^{k+1} = \max(V(a_i^k), F_i^k) + \frac{L_i^{k+1}}{w_i}$$

FQ Advantages

FQ protect well-behaved flows from ill-behaved flows

Example: 1 UDP (10 Mbps) and 31 TCP's sharing a 10 Mbp link





Summary

- FQ does not eliminate congestion → it just manages the congestion
- You need both end-host congestion control and router support for congestion control
  - End-host congestion control to adapt
  - Router congestion control to protect/isolate
- Don’t forget buffer management: you still need to drop in case of congestion. Which packet’s would you drop in FQ?
  - One possibility: packet from the longest queue