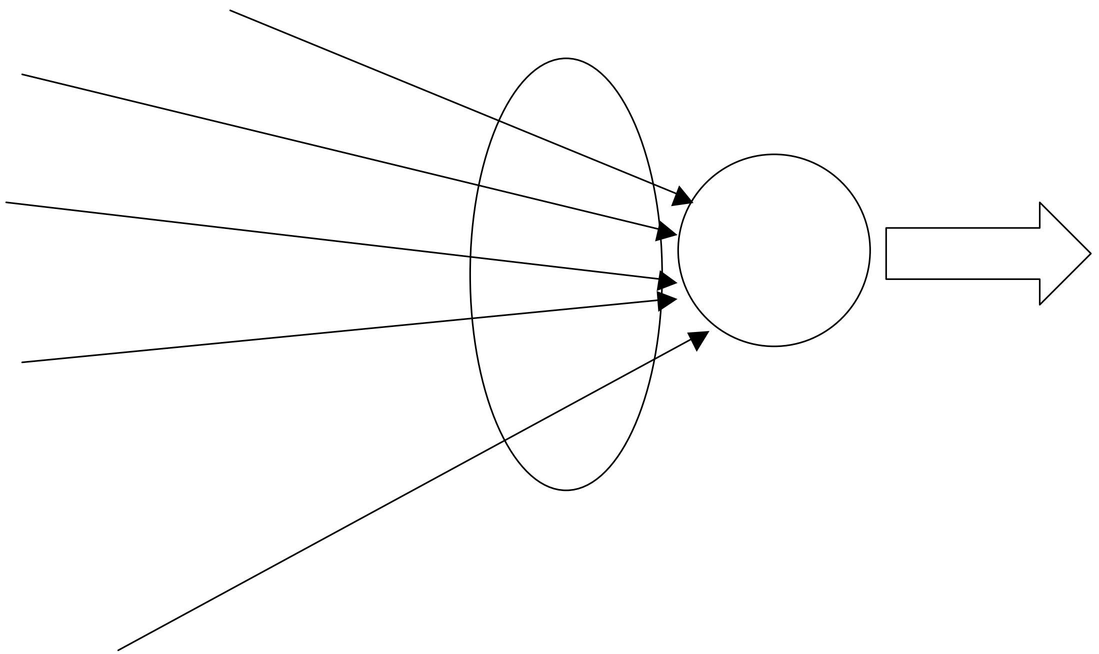
Scheduling

CS 218 Fall 02 - Keshav Chpt 9

Nov 5, 2003

Problem: given N packet streams contending for the same channel, how to schedule pkt transmissions?



The ingredients of QoS support

- QoS routing
- **Scheduling**
- Policing
- Call Admission Control

Scheduling - references

- Keshav, Chpt 9
- Ed Knightly et al: **Coordinated Scheduling:A Mechanism for Efficient Multi-Node Communication**, **http://www.ece.rice.edu/networks**
- Stoica, Shanker and Zhang: **Core Stateless Fair Queueing, SIGCOMM 98**

Scheduling Features/Requirements

- **Easy to implement (eg, per flow vs per class)**
- **Fair (for best effort sources)**
- **Protected against abusive sources (for best effort)**
- **Performance bounds (for guaranteed service)**
- **Admission control (for guaranteed service)**

Note: Features differ depending on whether we schedule best effort or guaranteed service traffic

Control Parameters/Measures

Control “knobs”	Perf. Measures
• priority ranking	avg delay; bdw share
• polling frequency	bandwidth
• buffer allocation/pkt drop	loss rate
• polling frq/buffer alloc	fairness

Performance Bounds

- **Deterministic bounds**: satisfied by **ALL** packets
- **Statistical bounds**:satisfied by a fraction **R** of packets

(a) Bandwidth: important for real time applications (eg, video on demand)

(b) Delay: avg., worst case, 99% (important for interactive, eg IP telephony)

(c) Delay jitter: important for interactive appl. (reconstruction buffer for playback)

(d) Loss: important for both real time and interactive

Max-Min Fairness

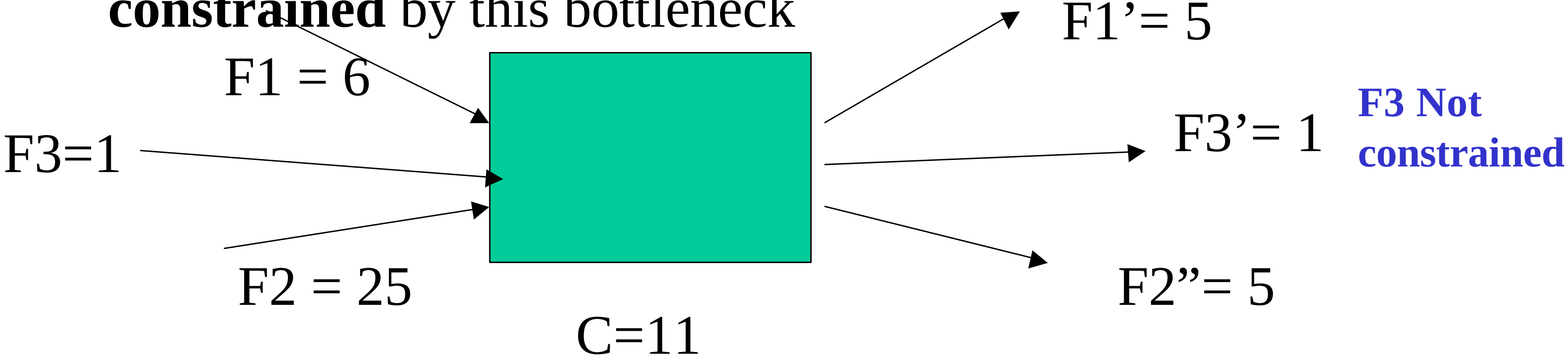
(ie, must maximize the minimum)

The **min** of the flows should be **as large as it would like to be (ie,max)**

Max-Min fairness condition for single resource:

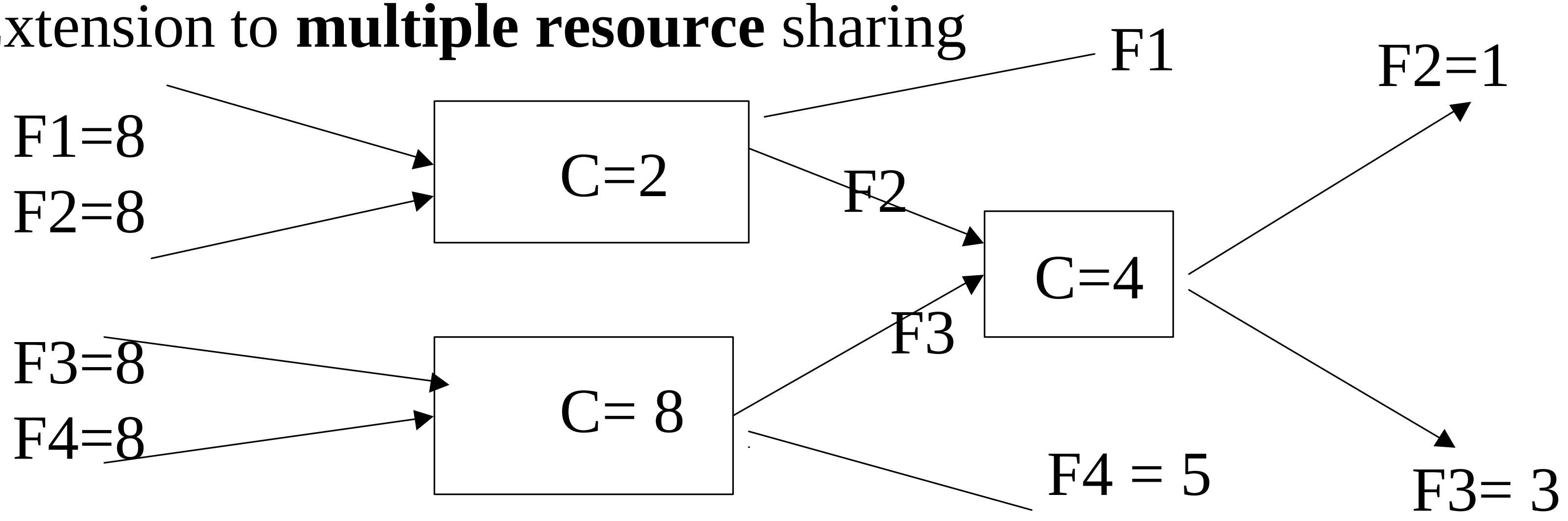
Bottlenecked (unsatisfied**)** connections share the residual bandwidth equally

Their share is > = the share held by the connections **not constrained** by this bottleneck



Max-Min Fairness (cont)

Extension to **multiple resource** sharing



Iterative construction approach (given the routing):

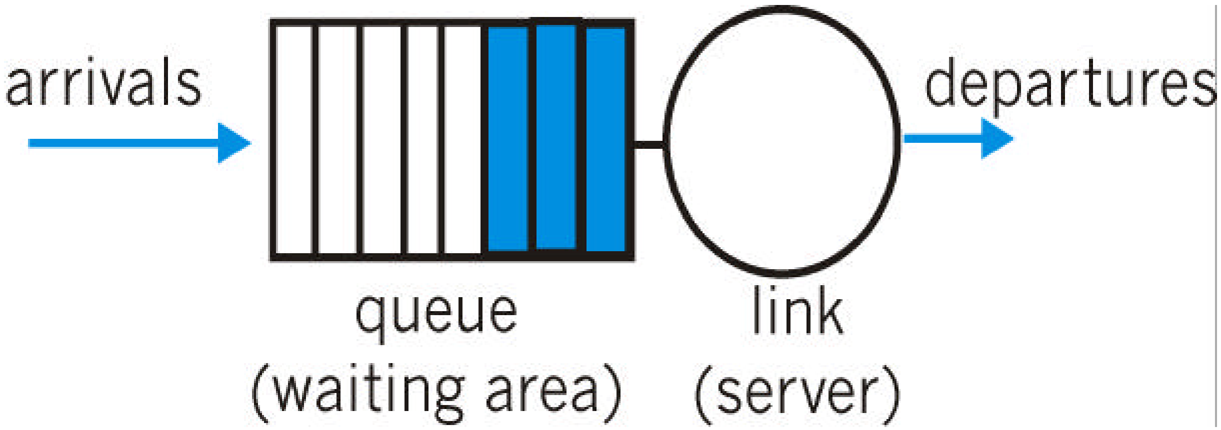
at each iteration increase the flow of non saturated connections by an increment DF

More Scheduling Issues (Keshav)

- **Work conserving** vs not work conserving (waste) schedule (the issue mainly concerns jitter control)
- **Per flow** vs **per class** (a la DiffServ) queueing: “per flow” does not scale, has bad reputation..
- Per-flow **service tag implementation** using two Heaps (for smallest tag and for largest tag): service tag as opposed to FCFS – pkt assigned a tag upon arrival and smallest tag served first
- **Schedulable region** (in space C1xC2): numbers of connections C1 and C2 that can be supported simultaneously, meeting the respective QoS req.ts

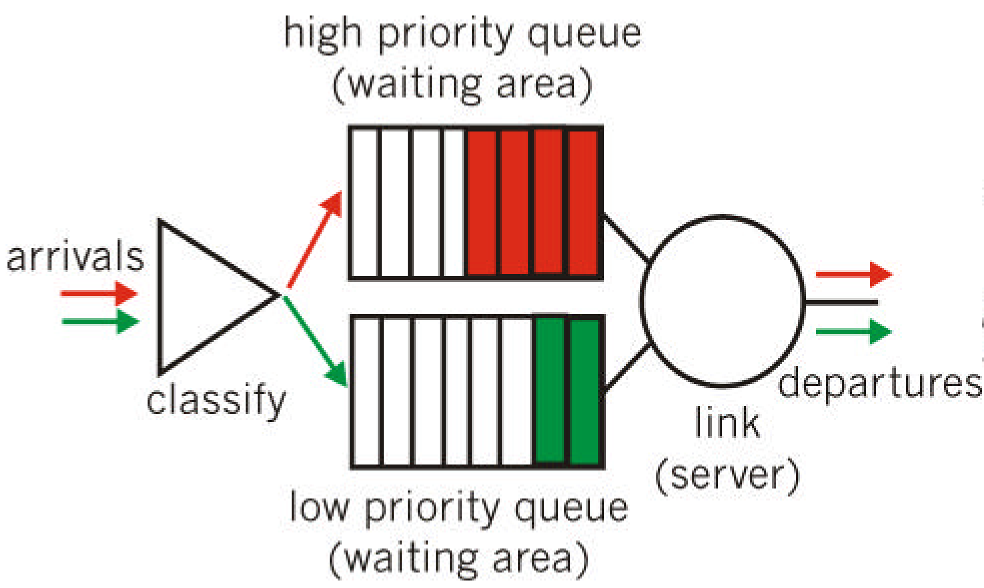
Schedule review: Best Effort Traffic

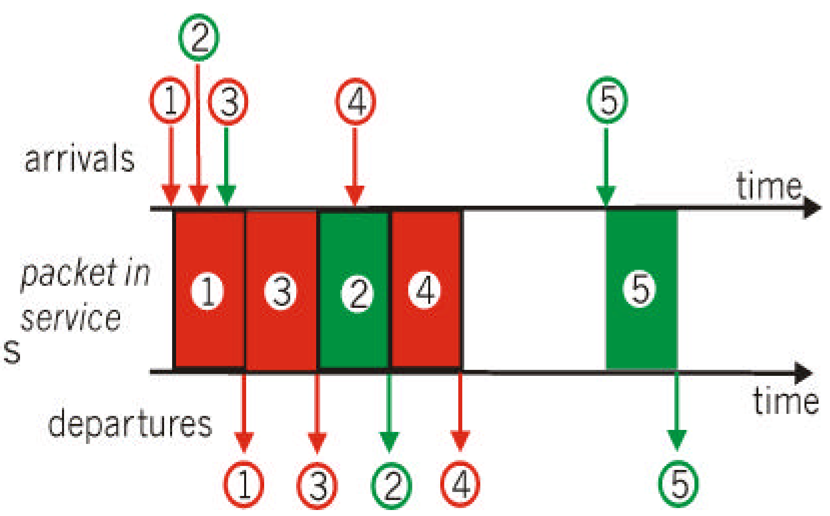
- **FIFO**: in order of arrival to the queue; packets that arrive to a full buffer are either discarded, or a discard policy is used to determine which packet to discard among the arriving pkt and those already queued



Scheduling (cont)

- **Priority Queuing**: classes have different priorities; class may depend on explicit marking or other header info, eg IP source or destination, TCP Port numbers, etc.
- Transmit a packet from the highest priority class with a non-empty queue
- **Preemptive and non-preemptive** versions



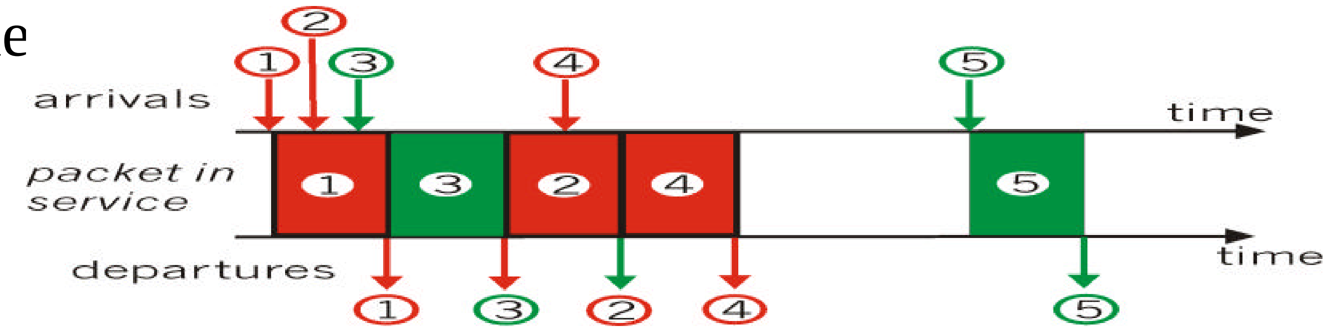


Scheduling (cont)

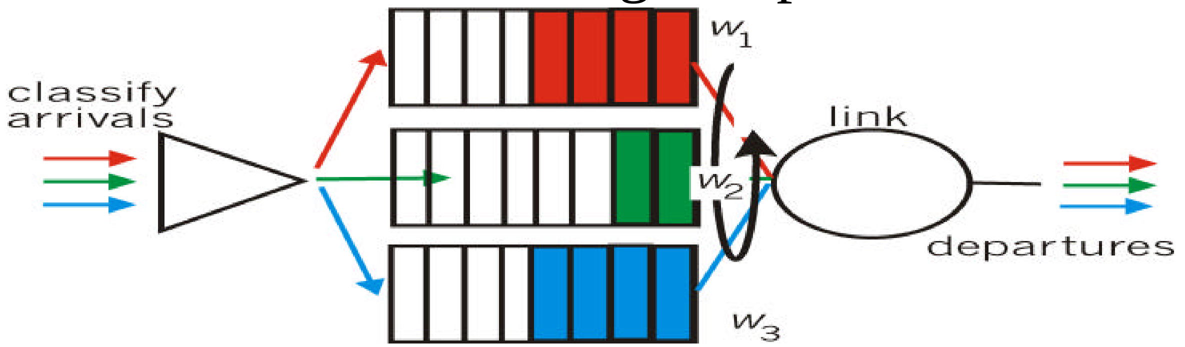
- Within the same priority class, need to schedule packet transmissions so as to achieve max-min fairness
- **Generalized Processor Sharing**: visit queues in turns, serving an infinitesimal increment from each – ideal, non implementable

Scheduling best effort (Cont.)

- **Round Robin**: scan class queues serving one from each class that has a non-empty queue; **max-min fair** (single queue



- **Weighted Round Robin**: is a generalized Round Robin in which an attempt is made to provide a class with a differentiated (ie **different weight** eg based on pkt size) amount of service over a given period of time



Scheduling Best Effort Traffic (cont)

“Deficit” RR:

- achieves same effect as WRR, but does not require avg pkt size knowledge
- a quantum size, say Q is defined (eg, Q= Pkt avg)
- a Deficit Counter DC is initialized to Q
- queues are served RR; if queue is empty, DC <= Q
- if HOL packet length is P < DC , it is served;
  DC <= DC + (Q-P)
- else, packet is queued and DC <= DC + Q

Scheduling Best Effort Traffic (cont)

Weighted Fair Queueing:

- compute the packet **finishing time**, ie,the time when the packet would be served by Generalized Proc Sharing (you “simulate” GPS on the side)
- rank packets according to **finishing times**
- the resulting sequence number (**finishing number**) is the packet’s turn to be transmitted.
- very complex to implement (can use pkt tags and heaps..)

Scheduling real time traffic

Weighted Fair Queueing:

- **Assume: G(j,k)= portion of link rate R(k) allocated to flow j**
- **elegant (but conservative) path delay bound D applies (Parekh & Gallager)**
- **D(j) = S(j)/Gmin(j) + Sum {Pmax(j)/G(j,k); over k on path} + Sum {Pmax/R(k); over k on path}**
- **S(j)= max burst for flow j admitted by leaky bucket**
- **Gmin(j) = lowest rate allocation to flow j on path**
- **Pmax (j)= max packet size for flow j**
- **Pmax = max pkt size over all flows**

Scheduling real time traffic (cont)

Virtual Clock

- arriving packets in a flow are tagged using a "virtual clock"; lowest tag served
- virtual clock ticks with the predefined flow rate
- it emulates Time Division Multiplexing

Earliest Due Date (or Earliest Deadline First)

- arriving packet tagged with deadline
- earliest deadline tag served first