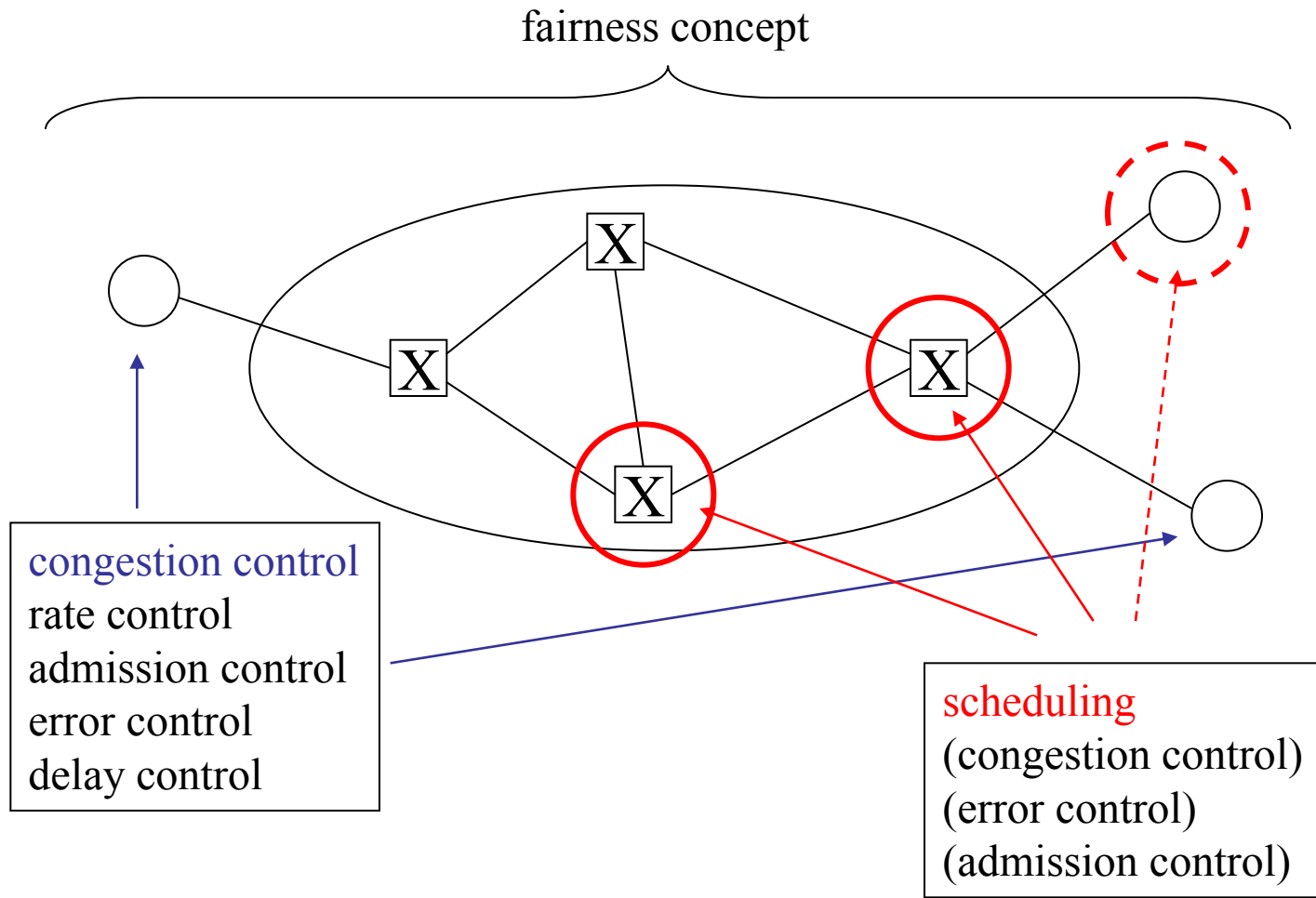


EP2210

Scheduling

- Lecture material:
 - Bertsekas, Gallager, 6.1.2.
 - MIT OpenCourseWare, 6.829
 - A. Parekh, R. Gallager, "A generalized Processor Sharing Approach to Flow Control - The Single Node Case," IEEE Infocom 1992

Scheduling



Scheduling - Problem definition

- Scheduling happens at the routers (switches) – or at user nodes if there are many simultaneous connections
 - many flows transmitted simultaneously at an output link
 - packets waiting for transmission are buffered
- Question: which packet to send, and when?
- Simplest case: FIFO
 - packets of all flows stored in the same buffer in arrival order
 - first packet in the buffer transmitted when the previous transmission is complete
 - packet transmission in the order of packet arrival
 - packet arriving when buffer is full dropped
- Complex cases: separate queues for flows (or set of flows)
 - one of the first packets in the queues transmitted
 - according to some policy
 - needs separate queues and policy specific variable for each flow
 - PER FLOW STATE

Scheduling - Requirements

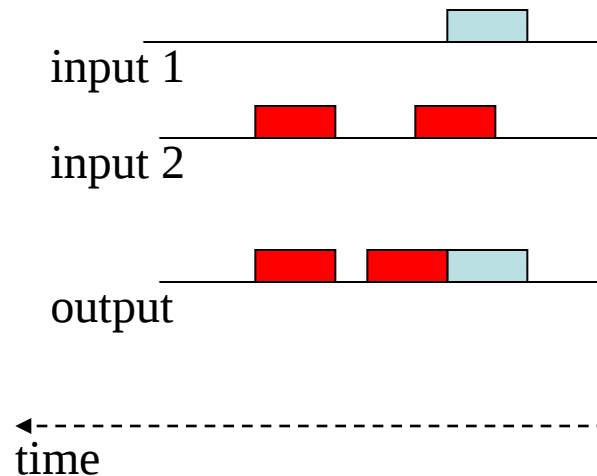
- Fair bandwidth allocation
 - for elastic (or best effort) traffic
 - all competing flows receive the some “fair” amount of resources
- Provide performance guarantees for flows or aggregates
 - service provisioning in the Internet (guaranteed service per flow)
 - guaranteed bandwidth for SLA, MPLS, VPN (guaranteed service for aggregates)
 - integrated services in mobile networks (UMTS, 4G)
- Performance guarantees
 - throughput, delay, delay variation, packet loss probability
 - performance guarantees should be de-coupled (coupled e.g., high throughput -> low delay variation)
- Easy implementation
 - has to operate on a per packet basis at high speed routers

Scheduling – Implementation issues

- Scheduling discipline has to make a decision before each packet transmission – every few microseconds
- Decision complexity should increase slower than linearly with the number of flows scheduled
 - e.g., complexity of FIFO is 1
 - scheduling where all flows have to be compared scales linearly
- Information to be stored and managed should scale with the number of flows
 - e.g., with per flow state requirement it scales linearly (e.g., queue length or packet arrival time)
- Scheduling disciplines make different trade-off among the requirements on fairness, performance provisioning and complexity
 - e.g., FIFO has low complexity, but can not provide fair bandwidth share for flows

Scheduling classes

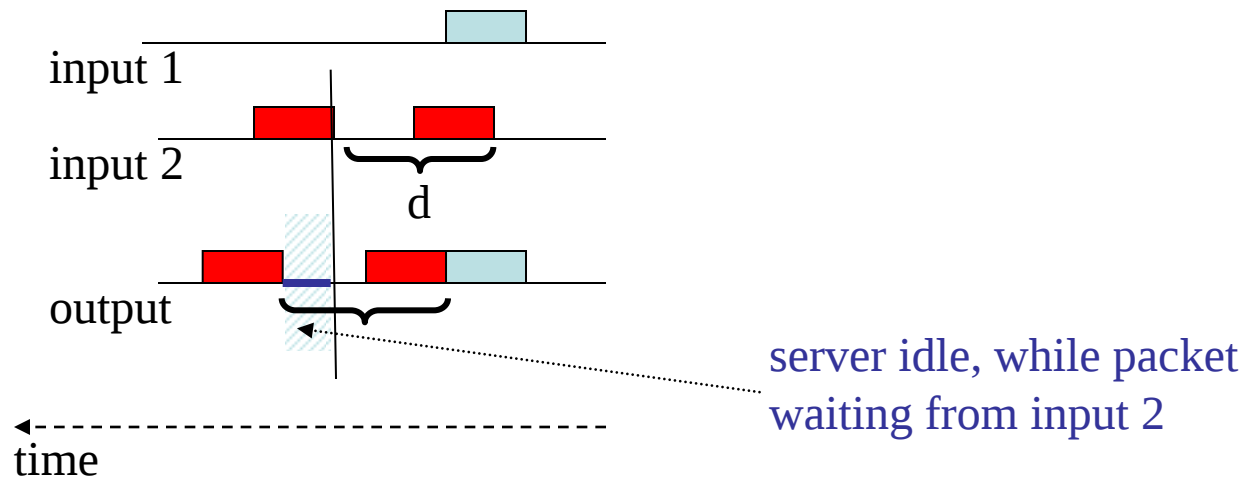
- Work-conserving
 - server (output link) is never idle when there is packet waiting



- utilizes output bandwidth efficiently
- burstiness of flows may increase → loss probability at the network nodes on the transmission path increases
- latency variations at each switch → may disturb delay sensitive traffic

Scheduling classes

- Nonwork-conserving
 - add rate control for each flow
 - each packet assigned an eligibility time when it can be transmitted
 - e.g, based on minimum d gap between packets
 - server can be idle if no packet is eligible



- burstiness and delay variations are controlled
- some bandwidth is lost
- can be useful for transmission with service guarantees.

Scheduling for fairness

- The goal is to share the bandwidth among the flows in a “fair” way
 - fairness can be defined a number of ways (see lectures later)
 - here fairness is considered for one single link, not for the whole transmission path
- Max-min fairness
 - *Maximize the minimum* bandwidth provided to any flow not receiving all bandwidth it requests
 - E.g.: no maximum requirement, single node – the flows should receive the same bandwidth
 - Specific cases: weighted flows and maximum requirements

Max-min fairness

- *Maximize the minimum* bandwidth provided to any flow not receiving all bandwidth it requests

C : link capacity

$B(t)$: set of flows with data to transmit at time t
(backlogged (saturated) flows)

$n(t)$: number of backlogged flows at time t

$C_i(t)$: bandwidth received by flow i at time t

Case: without weights or max. requirements

$$C_i(t) = \frac{C}{n(t)}$$

Case: weights

w_i : relative weight of flow i

$$C_i(t) = \frac{w_i}{\sum_{j \in B(t)} w_j} C$$

Case: max. requirements

r_i : max. bandwidth requirement for flow i

$\alpha(t)$: fair share at time t

$$C_i(t) = \min(r_i, \alpha(t))$$

$$\alpha(t) : \sum_{j \in B(t)} \min(r_j, \alpha(t)) = C$$

Max-min fairness

C : link capacity

$B(t)$: set of backlogged flows at time t

$C_i(t)$: bandwidth received by flow i at time t

Case: weights

w_i : relative weight of flow i

$$C_i(t) = \frac{w_i}{\sum_{j \in B(t)} w_j} C$$

Case: max. requirements

r_i : max. bandwidth requirement for flow i

$\alpha(t)$: fair share at time t

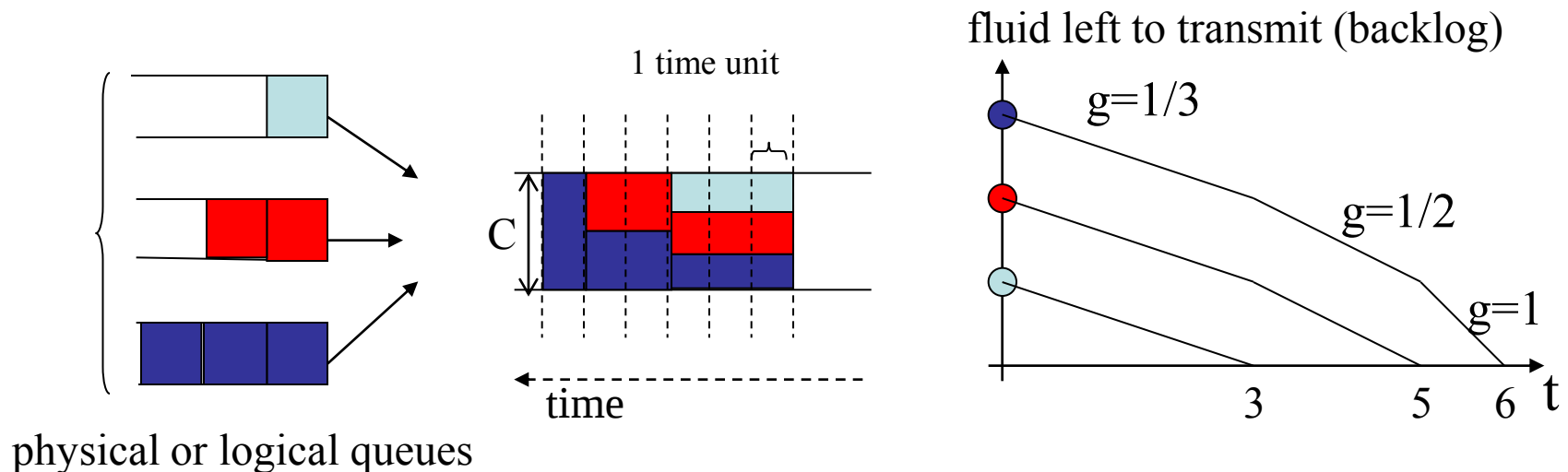
$$C_i(t) = \min(r_i, \alpha(t))$$

$$\alpha(t) : \sum_{j \in B(t)} \min(r_j, \alpha(t)) = C$$

- Calculate fair shares:
 - 3 backlogged (saturated) flows, equal weights, link capacity 10.
 - 3 backlogged flows, weights 1,2,2 link capacity 10
 - 4 backlogged flows, max requirements: 2, 3, 4, 5, link capacity 11.
 - 3 backlogged flows, rate requirements: 2,4,5, the link capacity is 11. What are the fair shares now?

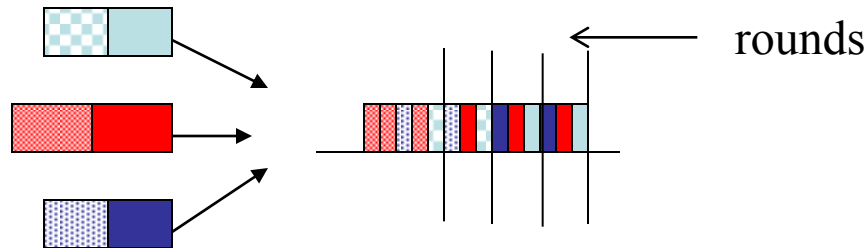
Fair queuing-for max-min fairness

- Fluid approximation
 - fluid fair queuing (FFQ) or generalized processor sharing (GPS)
 - idealized policy to split bandwidth
 - assumption: dedicated buffer per flow
 - assumption: flows from backlogged queues served simultaneously (like fluid)
 - not implementable, used to evaluate real approaches
 - used for performance analysis if per packet performance is not interesting

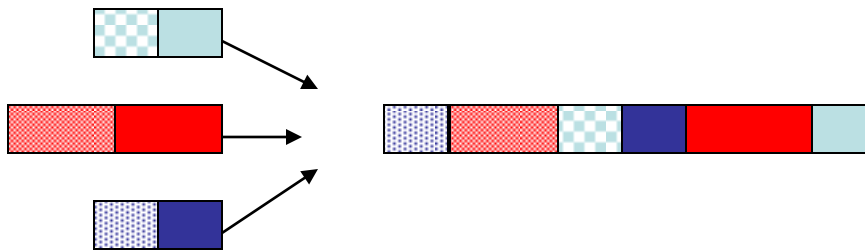


Packet-level Fair queuing

- How to realize GPS/FFQ?
- Bit-by-bit fair queuing
 - one bit from each backlogged queue in rounds (round robin) – still not possible to implement



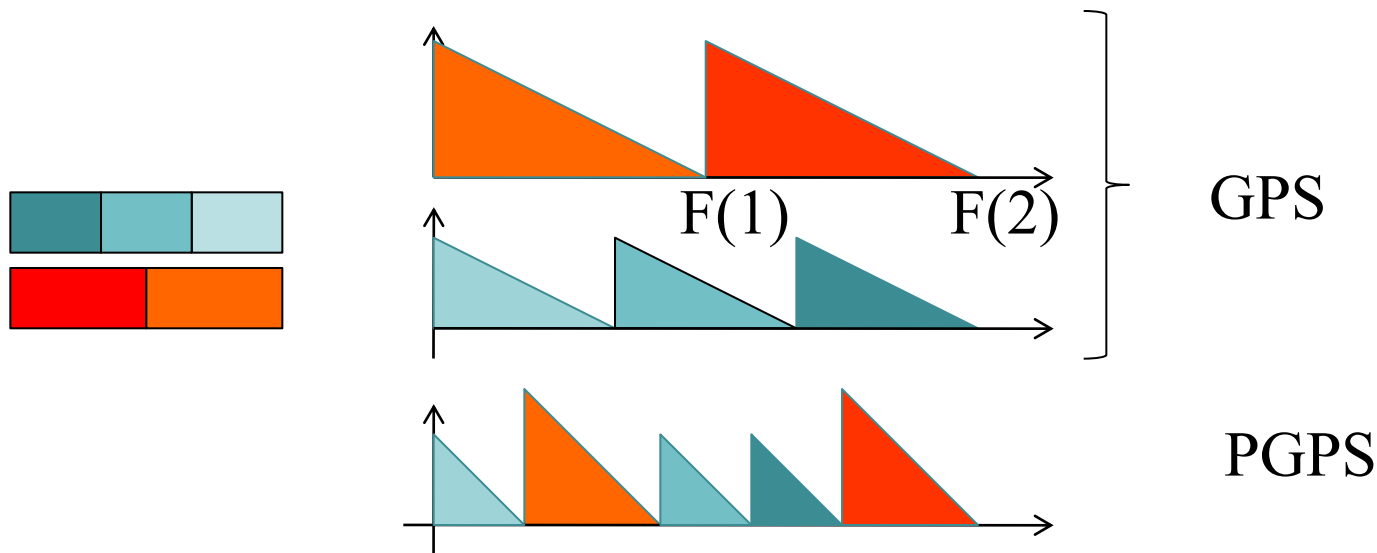
- Packet-level fair queuing
 - one packet from each backlogged queue in rounds ???



Flows with large packets
get more bandwidth!
More sophisticated schemes
required!

Packetized GPS (PGPS)

- How to realize GPS/FFQ?
- Try to mimic GPS
- Transmit packets that would arrive earliest with GPS
 - Finishing time ($F(p)$)
- Quantify the difference between GPS and PGPS



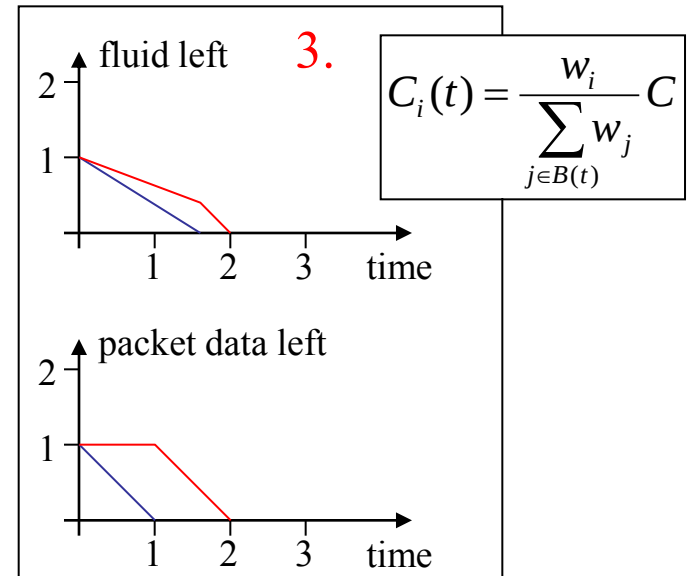
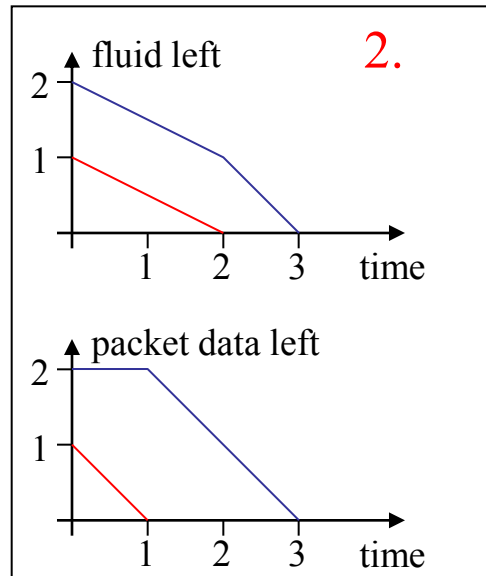
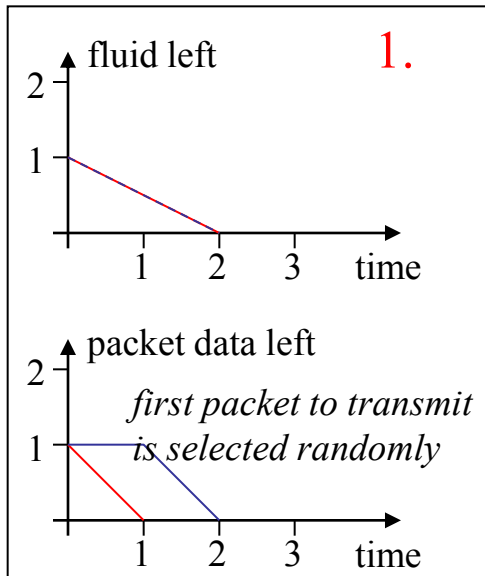
Fair queuing – group work

- Packet-by-packet GPS (PGPS)
 - Compare GPS (fluid) and PGPS (packetized) in the following scenarios – draw diagrams “backlogged traffic per flow vs. time”.
 - Consider one packet in each queue. $C=1$ unit/sec
1. Two flows, equal size packets, same weight, $L1=L2=1$ unit
 2. Two flows, different size packets, same weight $L1=1, L2=2$ units
 3. Two flows, same packet size, different weight, $L1=L2=1$ unit, $w1=1, w2=2$

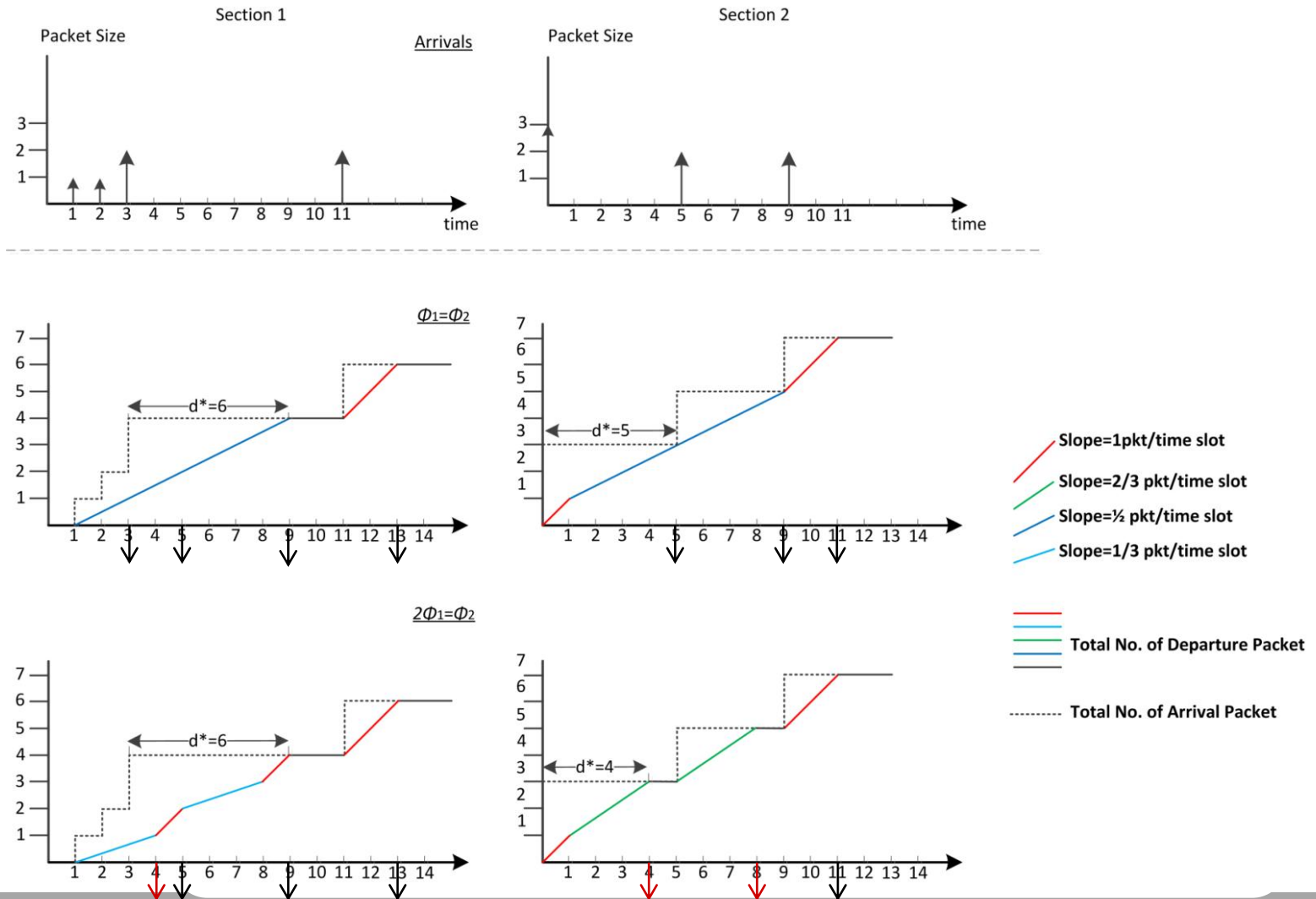
$$C_i(t) = \frac{w_i}{\sum_{j \in B(t)} w_j} C$$

Fair queuing – group work

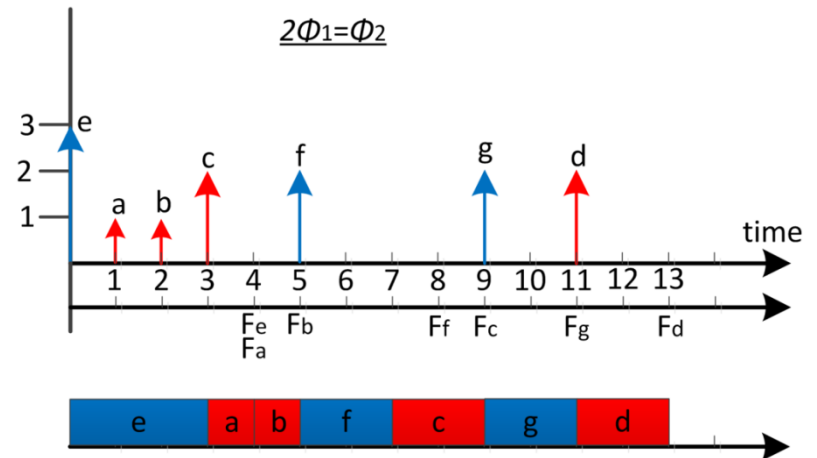
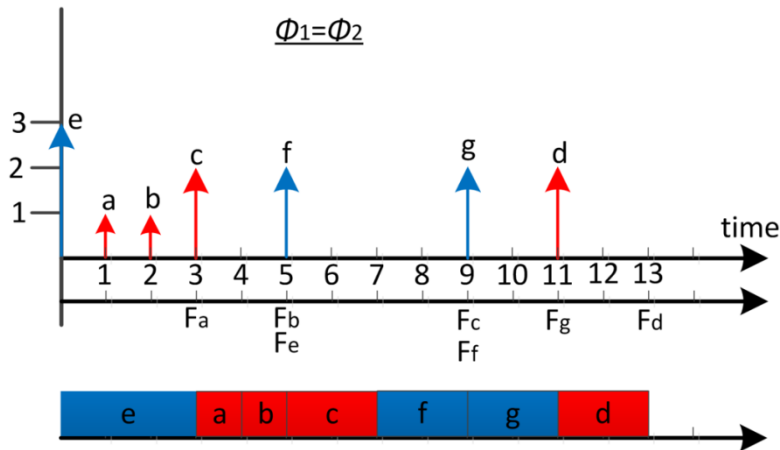
- Compare GPS (fluid) and PGPS (packetized) in the following scenarios – draw diagrams “backlogged traffic per flow vs. time”.
- Consider one packet in each queue. $C=1$ unit/sec
- 1. Two flows, equal size packets, same weight, $L_1=L_2=1$ unit
- 2. Two flows, different size packets, same weight $L_1=1$, $L_2=2$ units
- 3. Two flows, same packet size, different weight, $L_1=L_2=1$ unit, $w_1=1$, $w_2=2$



GPS vs. PGPS dynamic packet arrival



GPS vs. PGPS main results



		Section 1				Section 2		
Pkt. Info	Arrival	1	2	3	11	0	5	9
	Size	1	1	2	2	3	2	2
	Label	a	b	c	d	e	f	g
$\phi_1 = \phi_2$	GPS	3	5	9	13	5	9	11
	PGPS	4	5	7	13	3	9	11
$2\phi_1 = \phi_2$	GPS	4	5	9	13	4	8	11
	PGPS	4	5	9	13	3	7	11

GPS vs. PGPS main results

- Only packets that arrive “too late” to be scheduled in the GPS order are delayed more in PGPS than in GPS.
- Theorem 1: $F^*(p) - F(p) \leq L(\max)/r$
 - $F^*(p)$, $F(p)$: finishing time under PGPS and GPS
- Theorem 2: $S(0,t) - S^*(0,t) \leq L(\max)$
 - $S(0,t)$, $S^*(0,t)$: amount of traffic transmitted under GPS and PGPS
- Theorem 3: $Q^*(t) - Q(t) \leq L(\max)$
 - $Q(t)$, $Q^*(t)$: amount of traffic still in the queue in GPS and PGPS

GPS vs. PGPS main results

- Packet-by-packet GPS (PGPS)
 - transmit packet from buffers with earliest GPS finishing time.
- Question 1: how much later can a packet be “finished” in PGPS compared to GPS (reason, not in buffer at time of decision).
- Lemma 1: consider p and p' in buffer at time t , p completing service before p' - then the same happens under all future arrival pattern (since future arrivals delay the service of p and p' the same way).
- Theorem 1: $F^*(p) - F(p) \leq L(\max)/r$
 - $F^*(p)$, $F(p)$: finishing time under PGPS and GPS

GPS vs. PGPS main results

- Theorem 1: $F^*(p) - F(p) \leq L(\max)/r$
 - $F^*(p)$, $F(p)$: finishing time under PGPS and GPS

- Proof

$$t_k \leq u_k + \frac{L_{\max}}{r}$$

finishing GPS

finishing PGPS

- consider a busy period
- consider the last packet m that arrives before but leaves after packet k under GPS $u_m > u_k \geq u_i$ for $m < i < k$.
(packet k is scheduled after packet m in PGPS)
- start service time for m in PGPS is before arrivals $m+1 \dots k$

$$\min\{a_{m+1}, \dots, a_k\} > t_m - \frac{L_m}{r}$$

start service in PGPS

- finish time of k in GPS is larger then the time of fluid service of $m+1 \dots k$ plus the earliest arrival time of $m+1$

$$u_k \geq \underbrace{\frac{1}{r}(L_k + L_{k-1} + L_{k-2} + \dots + L_{m+1}) + t_m}_{t_k \text{ in PGPS}} - \frac{L_m}{r}$$

Scheduling summary

- Scheduling:
 - At the network nodes and at the edge
 - To provide quality guarantees or fairness
 - Work-conserving and non-work-conserving
- Max-min fairness in a single link, with weights and max. rate requirement
- GPS for max-min fairness in a fluid model
- PGPS (or WFQ) in the packetized version
 - Schedule according to finish time in GPS
 - Guaranteed performance compared to GPS
- Next lecture: work-conserving and non-work-conserving scheduling

Reading assignment

- For next lecture: A. Parekh, R. Gallager, "A Generalized Processor Sharing Approach to Flow Control - The Single Node Case," IEEE Transaction on Networking, 1993, Vol.1, No.3.
 - Read I-III-before part A
- For next week: H. Zhang, "Service Disciplines for Guaranteed Performance Service in Packet-Switching Networks," Proceedings of the IEEE, Oct, 1995, pp. 1374-1396
 - Read sections I and II, III.A,B,F,G