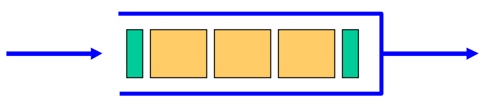
Fair Queueing

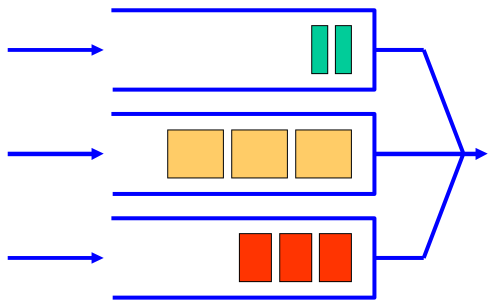
First-Come-First Served (FIFO)

- Packets are transmitted in the order of their arrival
- **Advantage:**
  - Very simple to implement
- **Disadvantage:**
  - Cannot give different service to different types of connections
  - Each flow (even with low data rate) can experience long delays



Static Priority

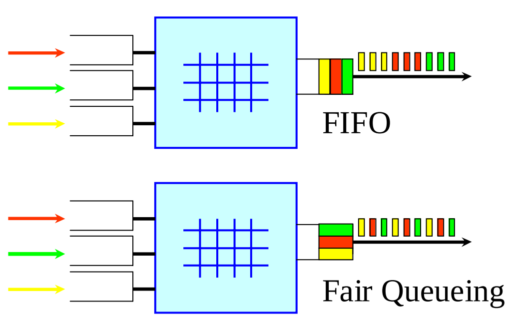
- Also called Head-of-Line (HOL) queueing:
  - Each traffic flow belong to a class
  - Each class has a priority
  - One FIFO queue for each class
  - Transmit from the highest priority queue with a backlog
- **Advantage:**
  - Simple
- **Disadvantage:**
  - Tends to “starve” the lower priority classes



3

Fair Queueing

- Attempts to implement a scheduler that simultaneously serves all flows with a backlog at the same rate
- Not easy to implement Fair Queuing in a packet network



4

Discussion of Fair Rate Allocation

- See class notes

Fair Scheduling Algorithms

- **Fair Queueing** (FQ),
  a.k.a as **Processor Sharing** (PS)
  → Objective: Achieve fair rate allocation

- **Weighted Fair Queueing** (WFQ),
  a.k.a. **Generalized Processor Sharing** (GPS)
  → Objective: Achieve weighted fair rate allocation

Problem: How to realize a fair rate allocation when

1. Traffic is transmitted in packet of variable size
2. Transmission of a packet cannot be interrupted

Fair Queuing in packet networks

- **Approach:**
  1. Take a **fluid-flow view** of traffic
     - View output link as a “pipe” with a given width
     - Transmitted traffic flows like a fluid through the pipe

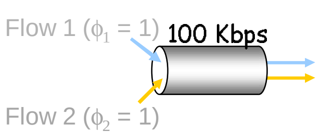
       → *Scheduler can transmit traffic from multiple flows at the same time*
     - Scheduler controls the “output rate” for each flow

       → Output rates are set to satisfy “fairness”
     - Result is a *fluid flow schedule*

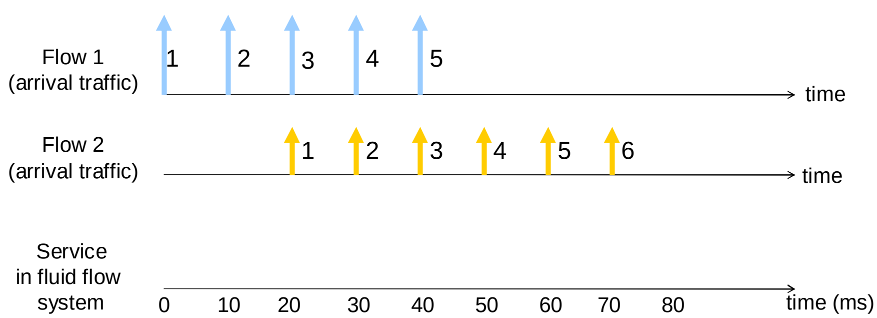
  2. Approximate fluid-flow schedule by a packet-level scheduling algorithm

7

Fair Queuing (FQ): From fluid to packets

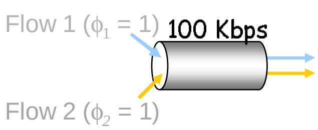


	Packet Size (bits)	Packet inter-arrival time (ms)	Rate (Kbps)
Flow 1	1000	10	100
Flow 2	500	10	50

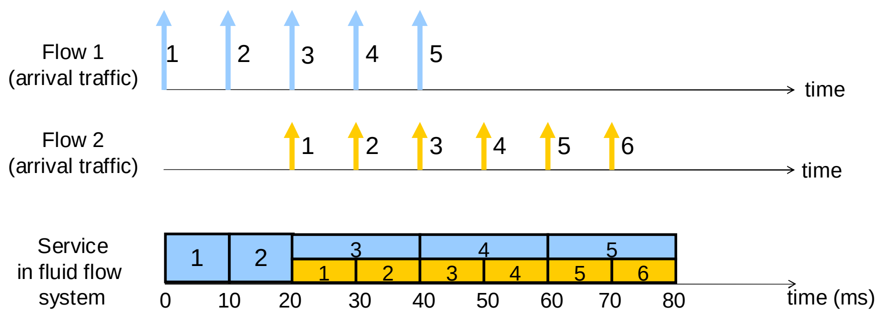


Slide from Ion Stoica 8

Fair Queueing (FQ): From fluid to packets *(complete)*

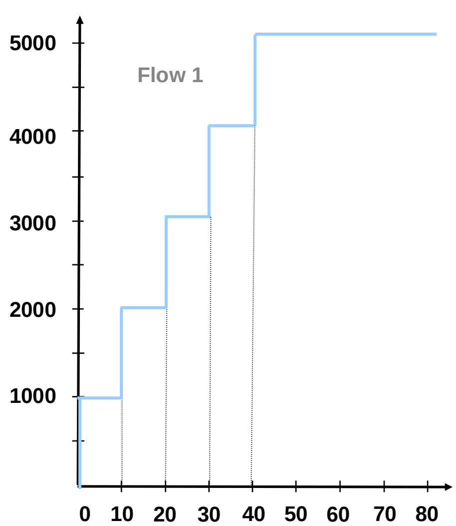


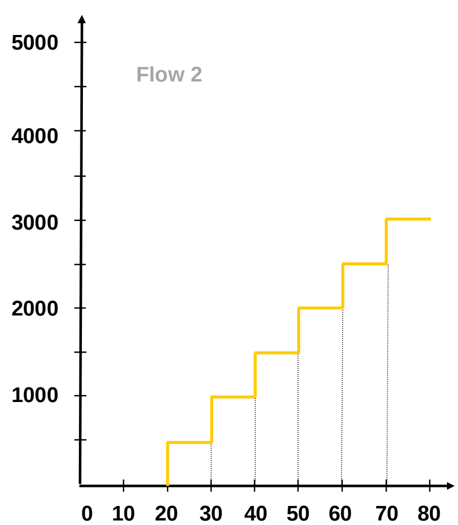
	Packet Size (bits)	Packet inter-arrival time (ms)	Rate (Kbps)
Flow 1	1000	10	100
Flow 2	500	10	50



Fair Scheduling

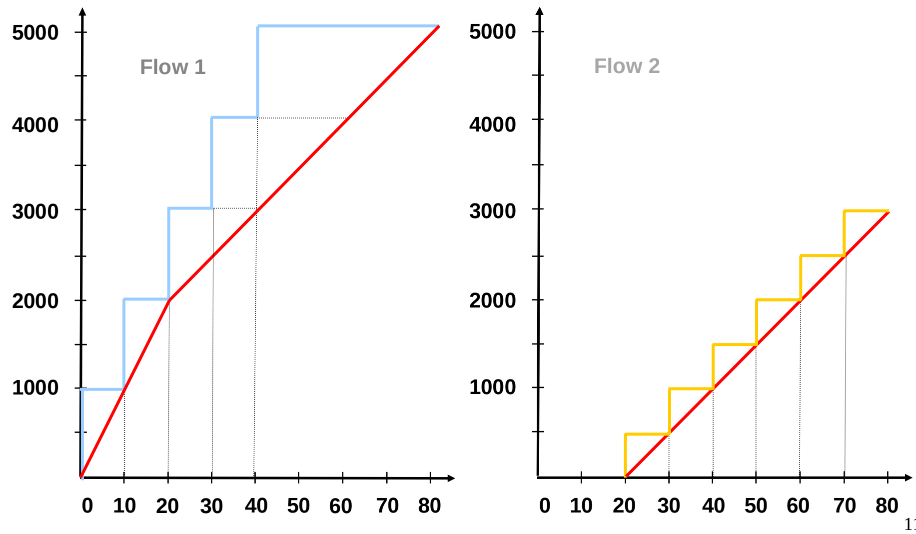
(fluid flow)





Fair Scheduling

(fluid flow)



11

Fair Scheduling (FQ)

(fluid flow view)

- There are N flows
- At any time t, all backlogged flows are served at the same rate of:

$$\frac{C}{|B(t)|}$$

where $B(t)$ is the set of backlogged flows at time t
C is the capacity of the link

- The total rate guarantee to a flow j is:

$$g_j = \frac{C}{N}$$

12

Weighted Fair Scheduling (WFQ)

(fluid flow view)

- There are N flows with weights $\phi_1, \phi_2, \ldots, \phi_N$
- The service given to two backlogged flows is proportional to their weights
- At any time *t* , the rate allocated to a backlogged flow *i* is:

$$\frac{\phi_i}{\sum_{j \in B(t)} \phi_j} C$$

where *B(t)* is the set of backlogged flows at time *t* in the fluid-flow system
C is the capacity of the link

- The total rate guarantee to a flow is:

$$g_j = \frac{\phi_j}{\sum_k \phi_k} C$$

13

FQ/WFQ Scheduling for Packets

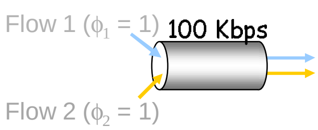
(packet-level view)

- Packet-level implementation of FQ and WFQ tries to emulate the fluid-flow version

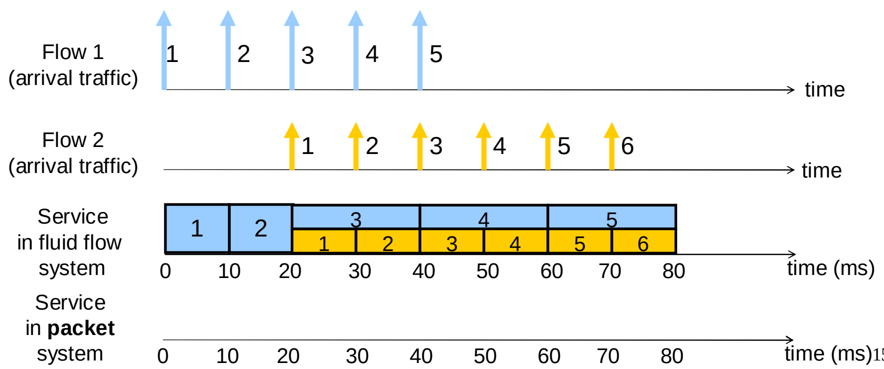
- **Scheduling decision**:
  Always select the packet that will finish next in the ideal fluid-flow FQ/WFQ system

14

WFQ scheduling *(with place holder)*

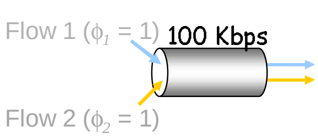


	Packet Size (bits)	Packet inter-arrival time (ms)	Rate (Kbps)
Flow 1	1000	10	100
Flow 2	500	10	50

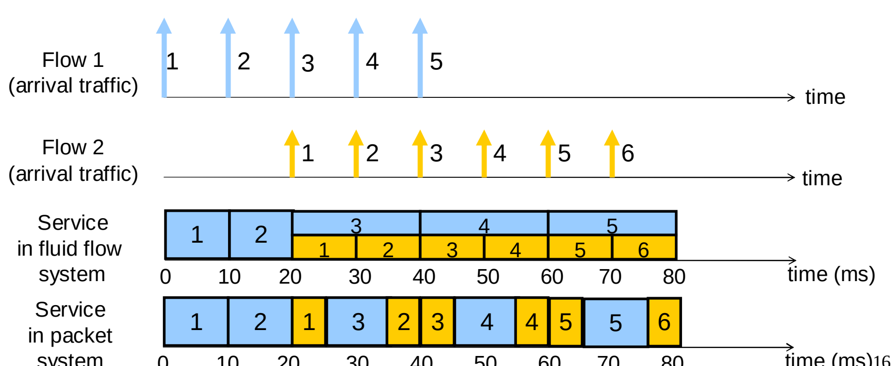


15

WFQ scheduling *(complete)*



	Packet Size (bits)	Packet inter-arrival time (ms)	Rate (Kbps)
Flow 1	1000	10	100
Flow 2	500	10	50



16

Slide from Ion Stoica

Packet-level Implementation of WFQ

Problems to deal with:

- The finishing time of a packet in the fluid-flow system may depend on arrivals after a packet has been selected
  →packet-level version of WFQ cannot be 100% accurate
- Once started, packet transmission cannot be preemtped

- **Implementation:**
  - When a packet arrives, it is assigned a “virtual finishing time”
    - This is the finishing time in the fluid flow system if the set of backlogged flows does not change after packet arrival
  - Orders packets in increasing order of virtual finishing times
  - Compute virtual finishing time with the help of a **system virtual time**

17

System Virtual Time

- See notes on System Virtual Time

18

WFQ: Implementation

- WFQ uses a **Virtual Time** that tracks the progress of GPS system
- Suppose the times when the set *B(t)* changes are $0 \leq t_1 \leq t_2 \leq$
- Let B_l be the set of backlogged flows in time interval $[t_{l-1}, t_l)$
- Then we have

$$V(0) = 0$$
$$V(t_{l-1} + \tau) = V(t_{l-1}) + \frac{C\tau}{\sum_{j \in B_l} \phi_j} \qquad \text{for } \tau \leq t_l - t_{l-1}$$

- When fewer flows are active, virtual time moves faster

19

WFQ: Implementation

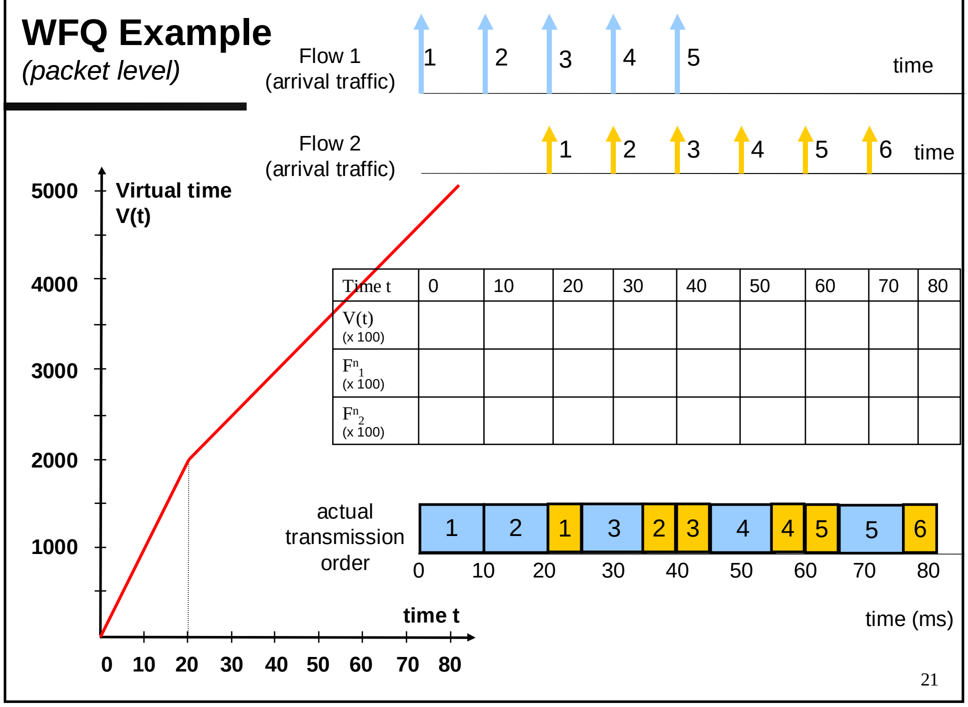
- Virtual finish time of k-th packet from flow j

$$F_j^k = \max\{F_j^{k-1}, V(a_j^k)\} + \frac{L_j^k}{\phi_j}$$

- a_j^k is the arrival time and L^k_j is thesize of the k-th packet from flow j
- Packets are sorted and transmitted in the order of virtual finishing times
  - Virtual times needs to be computed only at arrival time of packets
  - must keep track of the busy set B_l

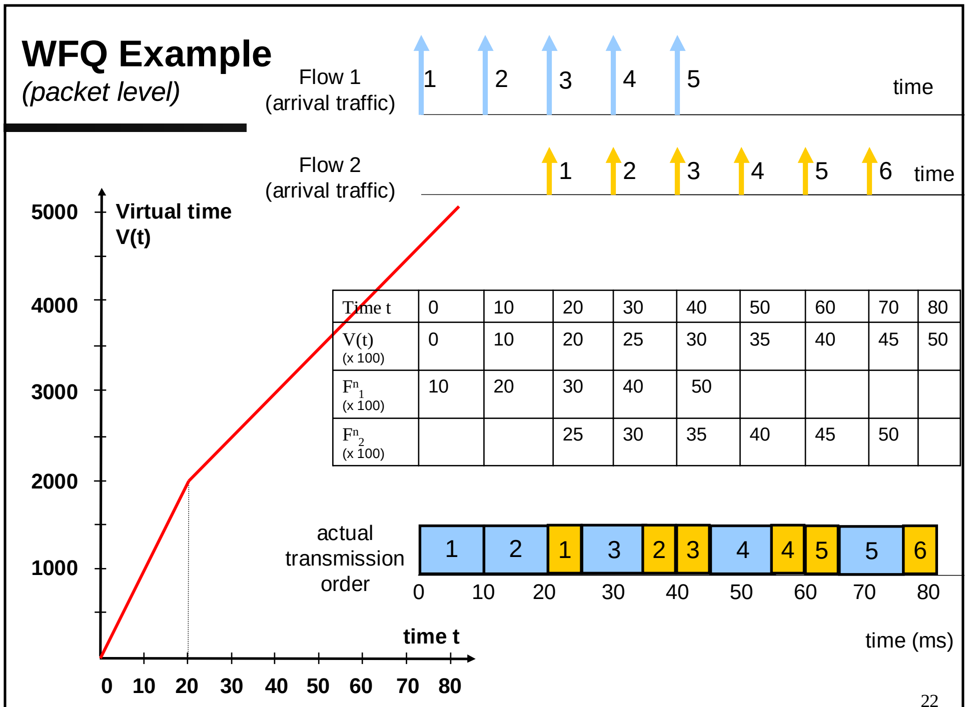
20

WFQ Example
(packet level)



Time t	0	10	20	30	40	50	60	70	80
V(t) (x 100)									
F^n_1 (x 100)									
F^n_2 (x 100)									

WFQ Example
(packet level)



Time t	0	10	20	30	40	50	60	70	80
V(t) (x 100)	0	10	20	25	30	35	40	45	50
F^n_1 (x 100)	10	20	30	40	50				
F^n_2 (x 100)			25	30	35	40	45	50	

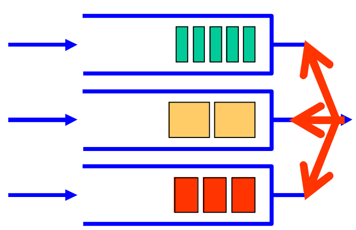
Approximations of Fair Queueing

- Since the packet implementation of WFQ is complex, packet switches often use approximations:
  - Weighted Round Robin (WRR)
  - Virtual Clock (VC)
  - Many others

23

Weighted Round Robin (WRR)

- Simple emulation of GPS
  - Operates in "rounds"
  - L_i is the average packet size of flow i
- Calculate the number of packets to be served in each round:
  - *For each flow i:* $x_i = w_i / L_i$
  - $x = min_i \{ x_i \}$
  - *For each flow i:* $packets\_per\_round_i = x_i / x$
- WRR is a good approximation of GPS if
  - All flows are active
  - Over long periods of time



24

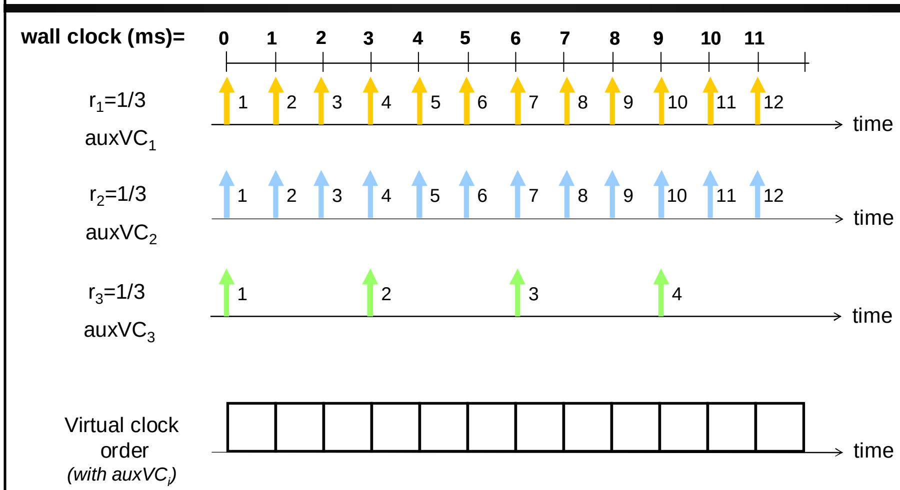
Virtual Clock (VC)

- Emulates a system with transmissions in periodic intervals
- Two state variables for each flow j:
  - $auxVC_j$ virtual transmission time of the flow
  - r_j reserved rate
- The variable $auxVC_j$ keeps track of hypothetical departure times. If all traffic from flow j is limited to the reserved rate, then $auxVC_j$ is the departure time of an arrival.
- Upon arrival of a packet from flow j *with size* L_j^k *at time* a_j^k:
  - $auxVC_j = max\,(auxVC_j\,, a_j^k) + L_j^k / r_j$
  - Stamp $auxVC_j$ in packet header
  - Packet are transmitted in increasing order of virtual transmission times

25

Example: Virtual Clock *(with place holder)*

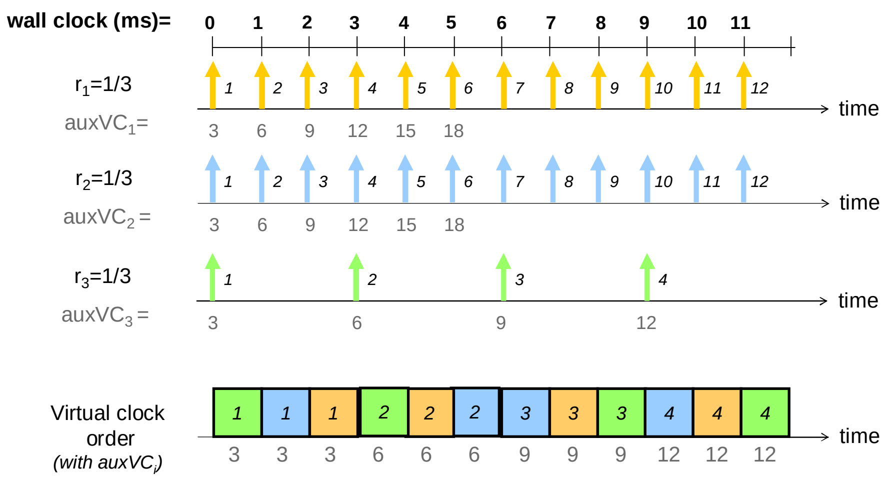
C = 1 Mbps, $r_1=r_2=r_3=1/3$ *Mbps, L=1000 bits*



26

Example: Virtual Clock

C = 1 Mbps, r_1=r_2=r_3=1/3 Mbps, L=1000 bits

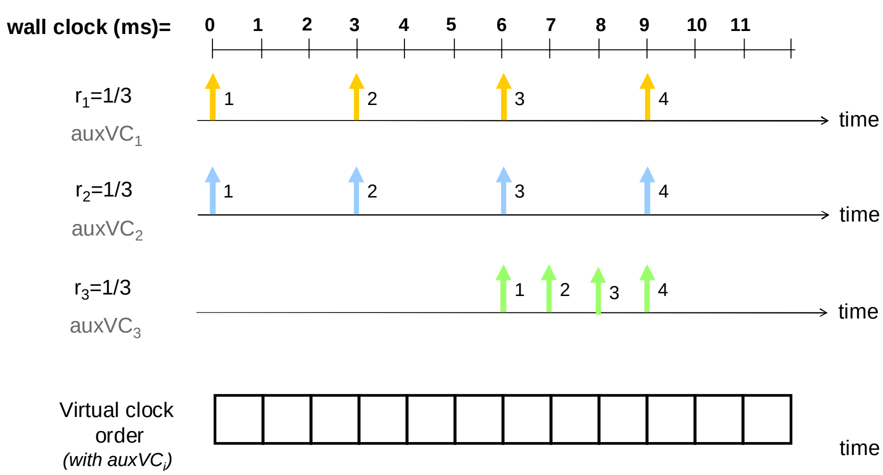


27

Example: Virtual Clock *(with place holder)*

C = 1 Mbps, r_1=r_2=r_3=1/3 Mbps, L=1000 bits

- "$auxVC_j = max\ (auxVC_j\ ,\ a_j^k) + L_j^k / r_j$" prevents credit accumulation of idle flows

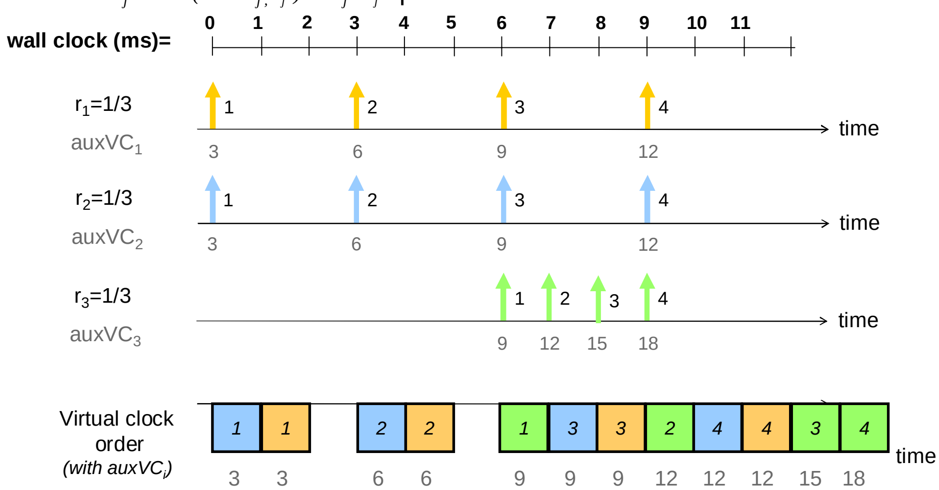


28

Example: Virtual Clock *(complete)*

C = 1 Mbps, r_1=r_2=r_3=1/3 Mbps, L=1000 bits

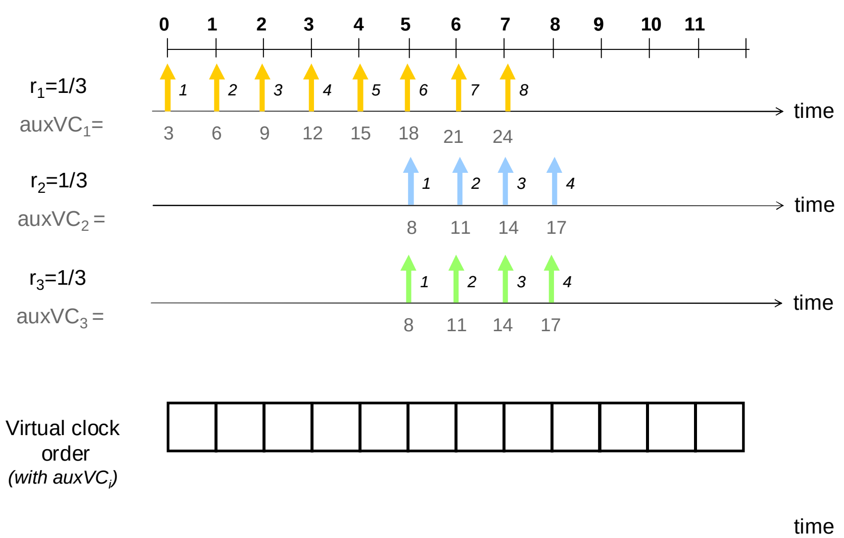
- "$auxVC_j = max(auxVC_j, a_j^k) + L_j^k / r_j$" prevents credit accumulation of idle flows



29

Problem with Virtual Clock *(with place holder)*

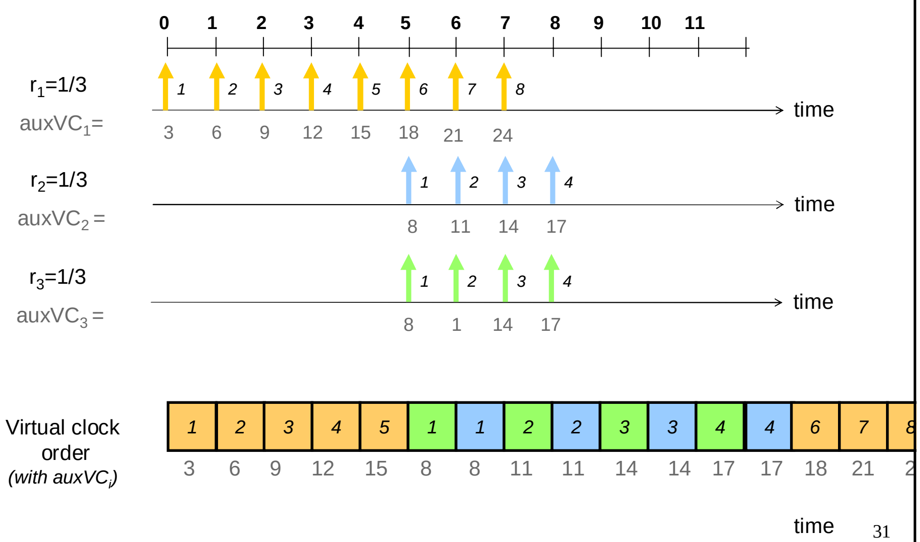
- Flow that gets more than reserved rate may be penalized in the future



30

Problem with Virtual Clock *(with place holder)*

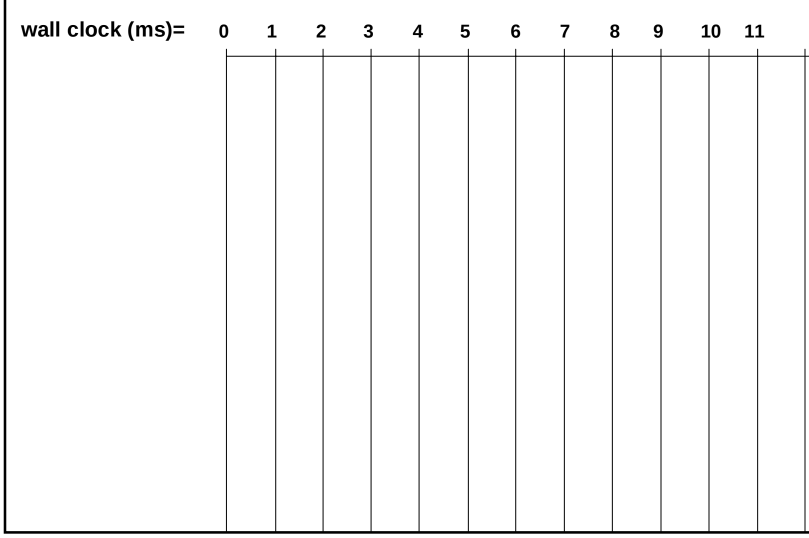
- Flow that gets more than reserved rate may be penalized in the future



31

Problem with Virtual Clock *(complete)*

- Flow that gets more than reserved rate may be penalized in the future



32