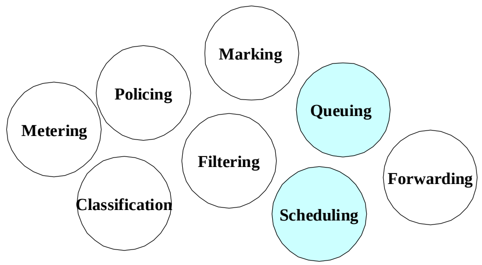
S−38.180: Quality of Service in Internet

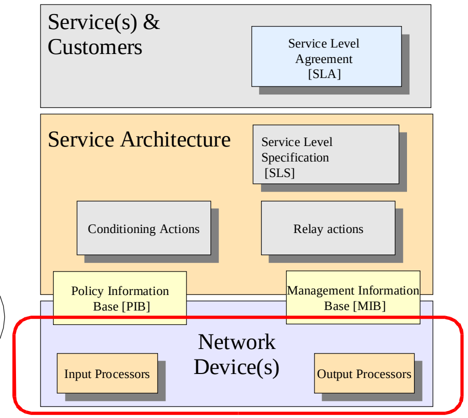
Lecture I: Egress Traffic Processing

25.9.2003

Today’s Topic

- This lecture is about functional mechanisms which can be found from the output processors of network devices





Egress Processing

- **Scheduling**
  - Decission of ordering of packets
    - Which packet is going to be sent out to the link next
  - Control of network link resource
    - Differentiation of 1st resource
    - Partial control over 2nd resource
- **Queue Management**
  - Decission of when packet should be dropped from the queue and which packet it is going to be
  - Control of delays and packet losses
    - Differentiation of 3rd resource
    - Partial control over 2nd resource

Queues

- Queues are used to store **contending** packets
  - Contention is temporary event rising from statistical multiplexing
    - Packets from different input links of a router attempt to the same output link at certain time
    - Packets from a higher speed link arrive temporarily too fast for a slow speed link
- If contention is permanent queues overflow i.e. network is **congested**
- **Difference:**
  - Contention − packets are not lost only delayed
  - Congestion − packets are not only delayed but also lost

Queues

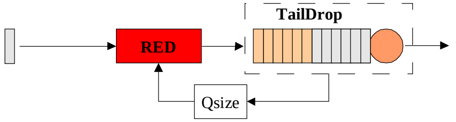
- Congestion situations demand **queue management** to decide
  - When packets should be discarded
  - Which are the packets that should be discarded
- Prevalent solutions
  - Tail Drop
  - Random Early Detection (RED)
  - Random Early Detection In/Out (RIO)

Tail Drop

- Simple algorithm:
  - If arriving packets sees a full queue it is discarded
  - Otherwise it is accepted to the queue
- Problem:
  - Poor fairness in distribution of buffer space
  - Unable to accommodate short transients when queue is almost full
    - Bursty discarding leading global syncronisation
- Global syncronisation is a process where large number of TCP connections syncronise their window control due to concurrent packet losses.
  - Packet losses are bursty, therefore window decreases to one and halts the communication

Random Early Detection

- RED is an active queue management algorithm (AQM), which aims to
  - Prevent global syncronisation
  - Offer better fairness among competing connections
  - Allow transient burst without packet loss
- Algorithm operates on the knowledge of current Qsize
  - Updated on every arrival and departure from the actual queue



RED

- Qsize is used to calculate average length of the queue:

Initial condition:

$avg(0) = 0$

$Count = -1$

When Qsize=0:

$T_{idle} = T_{now}$

After every packet arrival:

if $Qsize(n) > 0$:

$avg(n+1) = (1-\epsilon)\cdot avg(n) + \epsilon \cdot Qsize(n)$

else:

$avg(n+1) = avg(n)\cdot(1-\epsilon)^{f(T_{now}-T_{idle})}$

If queue is empty, averaging is done based on the assumption that N packets have passed the algorithm before actual packet arrival.
-> Decay of average during idle times

- Packets are discarded based on the average queue length:

if $avg(n+1) < min_{th}$:

$Count = -1$

else if $min_{th} \le avg(n+1) < max_{th}$:

$count = count + 1$

$$P_b(n+1) = max_p \cdot \frac{avg(n+1) - min_{th}}{max_{th} - min_{th}}$$

$$P_a(n+1) = \frac{P_b(n+1)}{1 - count \cdot P_b(n+1)}$$

With probability $P_a(n+1)$:

Discard packet

$Count = 0$

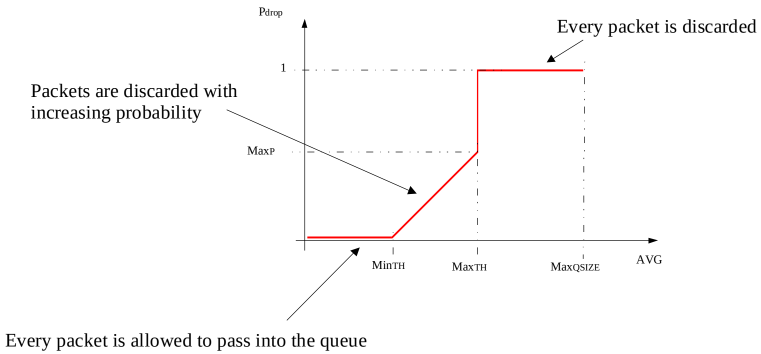
else if $max_{th} \le avg(n+1)$

Discard packet

$Count = 0$

Stochastic packet discard

RED

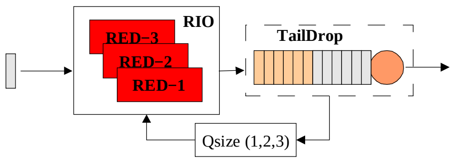


Achievements of RED

- **Some packets are discarded even before overflow of the actual buffer**
  - Is it good or bad ?
    - Bad: A part of buffer space is in some occasions wasted
    - Good: A signal is sent to co−operating sources that they should decrease their sending rate or congestion will occure
- **On the average early packet discards will hit connections which use more than their fair share of capacity in contending link**
  - Is it good or bad ?
    - Bad: Makes differentiation impossible
    - Good: Is consistent policy and withing the goal of conventional Best Effort model

RED In/Out − WRED

- When we aim for differentiation of resources we must also allow different shares of resources in contending link or buffer
- One way to do it is to use RED with several parallel algorithms and thresholds
  - RED In/Out −> RIO or WRED
  - Popular implementations use two or three parallel algorithms
- This requires that packets are marked
  - One algorithm is responsible of one or several marks

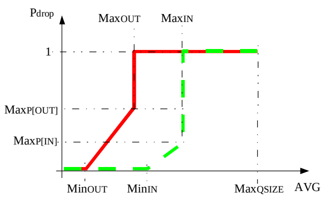


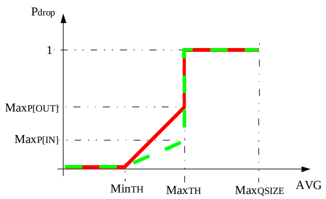
RIO

- Operation is usually based on following idea:
  - Customer has contracted capacity of X bps
  - He sends packets with rate Y bps
  - If Y is greater than X, some packets are marked as out of profile.
    - Out of profile packets usually experience harsh treatment on contending situations
- Calculation of the average queue length is modified to take into accout number of packets with different markings:
  - In (green): Only green packets
  - In/Out (yellow): Green and yellow packets
  - Out (red): All packets in the queue

Parameters in WRED

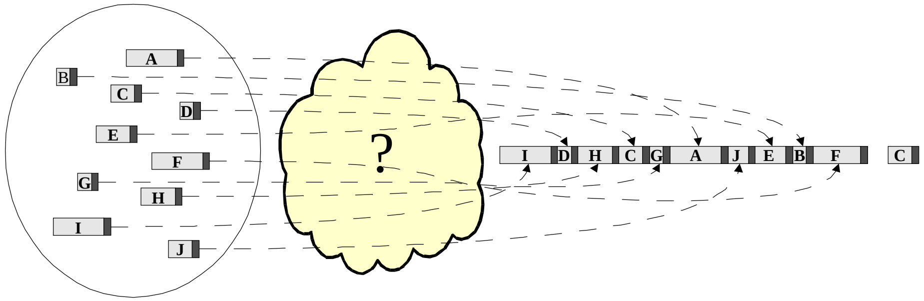
- All parameres are independent for different markings
  - More dimensions in creating differentiation
- Some parameters are common for different markings
  - Less dimensions but more understandable





Scheduling

- Task of a scheduler is to decide the order of packets which are transmitted from the queue

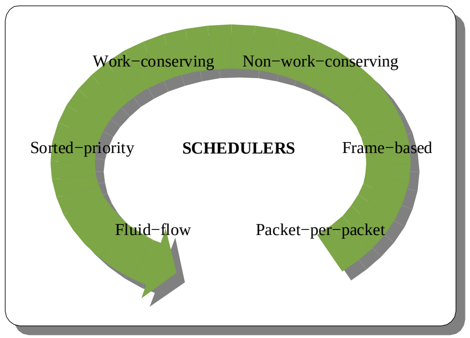


Scheduling

- Selecting the order of packets means that resource sharing is controlled with predefined policy.
- Policy defines the amount of resources which are allocated to the connections / classes for which single packets belong to.
- One end in this continum is that predefined amount of resources are allocated to the connections.
- Other end is that no allocation is done and resources are shared on the basis of the need

Scheduling

- There are vast amount of schedulers developed for different purposes
- Generally they can be divided into categories of
  - Work−conserving vs non−work−conserving
  - Time−based vs frame−based
  - Continuous vs packetized
  - Priority vs no priority

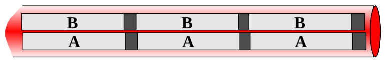


Scheduling

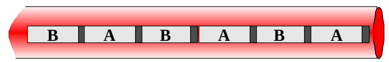
- **Conservation of work** means that scheduler is executing its task as long as it has some work to do.
- Technically this means that there are packets in the queue which has to be sent into the link before scheduler can take a break i.e. change to the idle state.
- Non−work conserving scheduler can idle even though it has packets in the queue.
- Why we would want to have non−work conserving scheduler ?
- Conservation of work means that packets are sent to the link even though receiver would prefere them to come a little bit later.
- This can happen with real−time applications which send packets with constant time intervals. However, network can multiplex them so that they form bursts. Non−work conserving scheduler may delay packets so that intervals structure is maintained throughout the network.

Scheduling

- **Continuous time**
  - Scheduling decissions and calculations are done based on continuous time units
  - Fluid−Flow modeling − packets are infinitesimally small
  - Assumes that number of packets could be served on same time (not possible)



- **Packetized**
  - Scheduling decissions and calculations are based on packet per packet analysis
  - Distorts fluid flow model



Scheduling

- **Time based scheduling**
  - Uses either arrival time or finishing time as a criteria for ordering
  - Time may be virtual or real−time depending on scheduler time
  - Virtual time is usually finishing time in ideal scheduler i.e. scheduler which is not packetized
- **Frame based scheduling**
  - Uses fixed frame which is partitioned for the scheduled packets based on their weights.
  - During a rotation,
    - If there are enough tokens (partition + left overs), then packet is served.
    - Otherwise tokens are added for the next round.
  - A number of packets may be served from a single class if frame is big.

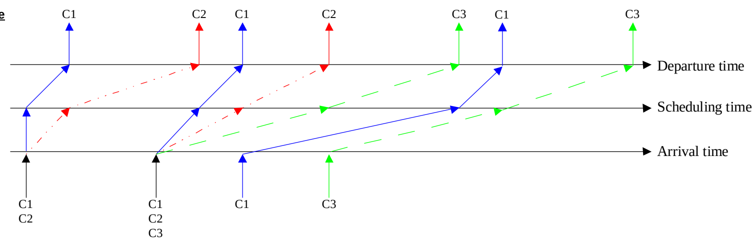
Scheduling

- Scheduling can happen:
  - **Within one queue**, sorting packets inside queue to appropriate transmission order
  - **Between several queues**, dispatching head of line packets from different queues
  - **Hierarchically over several schedulers**, combination of previous ones
- Many of scheduling algorithms can be used to produce QoS in each of these cases

Scheduling

- **First Come First Served** (FCFS) is prevalent scheduling method in routers.
- FCFS uses arrival time information as sorting criteria for packet dispatching.
- FCFS is not able to offer any QoS as arrival time is the only parameter that has influence to the order of packets.

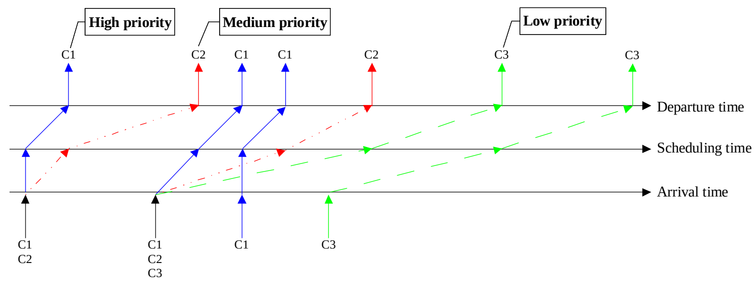
Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3



Scheduling

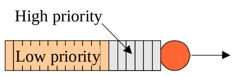
- **Simple priority** scheduler extends FCFS to be able to distinguish between more and less important traffic.
- Packets are ordered first based on their priority and second on their arrival time.

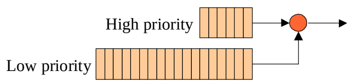
Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3



Scheduling

- Prioritized ordering may lead to starvation of resources in low priority classes if traffic in high priority classes is not limited.
- This can be accomplished by using
  - Connection admission control
  - Over provisioning
  - Rate control
  - Modifying priority scheduler to take class rates into account (token based operation)

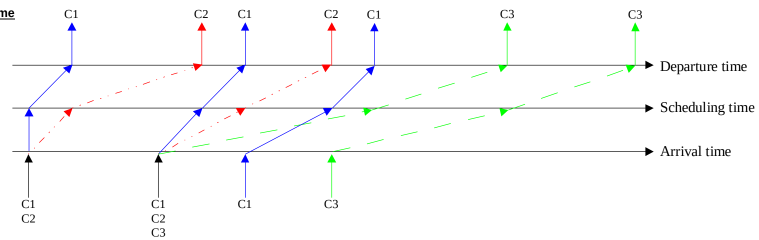




Scheduling

- **Deadline based** scheduling schemes (e.q. Earlies Due Date) are based on the calculation of finishing time (i.e. time when a packet would have been transmitted if it arrived to empty system).
- Packets are transmitted on the order of finishing times.

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3

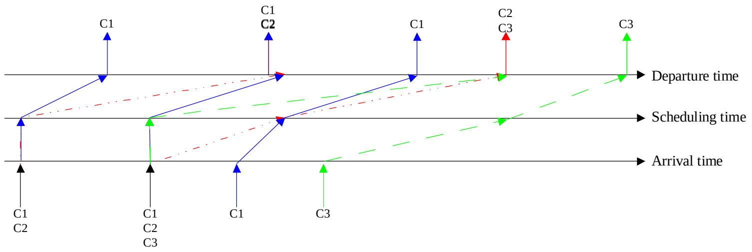


Scheduling

- **Generalized Processor Sharing** is ideal fair queueing algorithm which is based on fluid flow model.
- GPS provides service to the individual connections based on their weights.
- GPS is work conserving scheduler and thus distributes excess capacity to connections which are able to utilize it.

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3

Weights are all equal



Scheduling

- Disadvantages of GPS are:
  - Departures from GPS are colliding which makes the use of GPS based scheduler impossible
    - However it may be used as backgroud scheduler if collisions are resolved in some manner
  - Heavy calculation of departure times
    - Departure time of every packet in scheduler changes whenever a packet arrives or departs the scheduler

Scheduling

- Advantages of GPS are:
  - Fairness which it provides for the sharing connections

$$\frac{|Service(t, t+\Delta t)|_i}{|Service(t, t+\Delta t)|_j} \geq \frac{Weight_i}{Weight_j}$$

  - Strict delay bound caused by scheduling when traffic is constrained by a token bucket of token rate *r* and bucket depth *b*

$$Service\ rate\ for\ connection\ i:\ r_i \geq \frac{Weight_i}{\sum_j Weighth_j} \cdot LinkRate$$

$$Delay\ for\ connection\ i:\ D_i \leq \frac{b_i}{r_i}$$

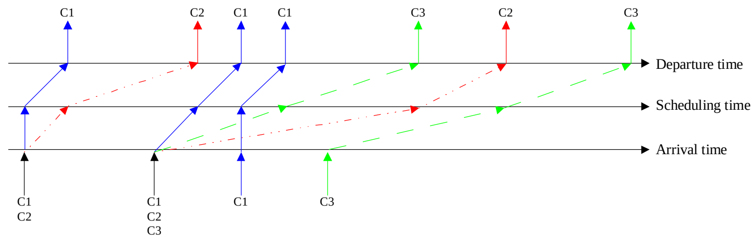
Remember these results were derived from the assumption that packets flow like fluid through the system i.e. there would be a dedicated link with capacity ***r*** between endpoints.

Scheduling

- **Packetized Generalized Processor Sharing** is packet per packet approximation of GPS scheduling.
- Most prevalent implementation of PGPS is weighted fair queueing (WFQ)
- WFQ uses calculation of finishing time in correspongina GPS system as a criteria for sorting the packets.

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3

Weights are all equal



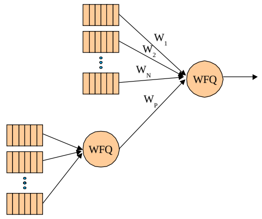
Scheduling

- Delay bound of WFQ system differs the one of GPS system with two extra components:
  - $\frac{(K-1)L'_{max}}{r_i}$ which represents extra delay caused if packet arrives a moment later it would have been served in corresponding GPS system. L is the maximum packet length and K is the number of hops.
  - $\sum_{m=1}^{K}\frac{L_{max}}{R^m}$ which represents the fact that packets are served one by one. In backlogged system, packet must wait that previous packet is served, before it gets to be scheduled.

$$D_i \le \frac{b_i}{r_i} + \frac{(K-1)L'_{max}}{r_i} + \sum_{m=1}^{K}\frac{L_{max}}{R^m}$$

Scheduling

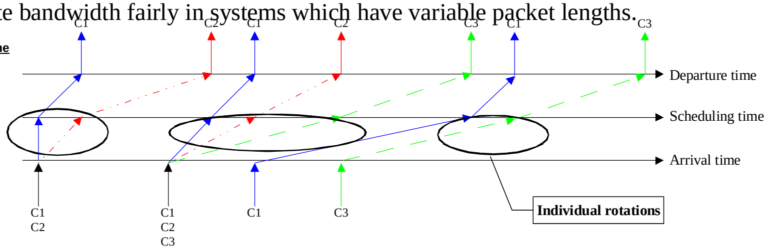
- WFQ scheduling has number of variant which aim:
  - Ease the calculation of finishing time in corresponding GPS system
    - By replacing the idle time function with the finishing time of packet which was in service when backlogging packet arrived to the system.
    - By replacing the time calculation with frame based operation
  - Make the fairness packetized system as good as continuous system
  - Allow hierarchical construction of service



Scheduling

- **Weighted Round Robin** is popular implementation of frame based fair queueing.
- WRR uses a rotation where each individual connection is served in relation of their weights.
- Service is usually based on packets, which causes WRR to be not able to distribute bandwidth fairly in systems which have variable packet lengths.

Class	Arrival Time	Service Time
C1	0	1
C2	0	3
C1	3	1
C3	3	3
C2	3	2
C1	5	1
C3	7	3



Scheduling

- **Deficit Round Robin** is extention of WRR which takes account the packet size
- DRR uses a rotation where a frame of *N* bits is divided to indivivual connections in relation to their weights (quantums).
- Quantums which individual connections receive serve packets
  - If the quantum is small, many rotations are required to serve backlogged connection
  - If the quantum is big, many packets can be served on one rotation
- DRR uses special counter for each backlogged connection which stores the information of received bits.
  - If connection gets to non backlogged state counter is cleared

Scheduling

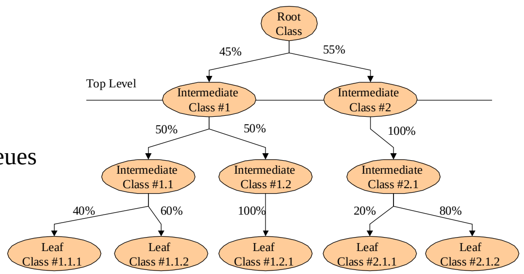
- **Class Based Queueing** is one form hierarchical scheduling
  - In CBQ scheduling is divided into two cases:
    - Unregulated: When a class is scheduled by **general scheduler**
    - Regulated: When a class is scheduled by **link share scheduler**
  - Class is regulated in situations when network is persistently contended and class has run over its limits
- Actual implementation of scheduling is uniform
  - Both schedulers manipulate HOL packets time to send information which is then examined by actual dispatcher.
- CBQ uses different variants of round robin schedulers as a general scheduler
- Link share scheduler is based on general rules supplied by user

Scheduling

- Advantage of CBQ is that scheduling during contention is easily manipulated to produce outcome which is not only based on time and priority information
- Disadvantage is that CBQ requires a lot of processing time when there are a lot of independent connections / classes

Scheduling

- Link sharing guidelines are based on tree like structure
  - Link resources are on Root Class
  - Intermediate Classes form logical groupings
    - Organisations
    - Protocols
  - Leaf classes are actual queues with distinct traffic



Scheduling

- CBQ has concept of **borrowing:**
  - If class has run over its limit but it has parent class which is not over its limit, it may borrow capacity from the parent
  - Borrowing may be limited to some level in link sharing tree (Top Level)
- Formal definition between regulated and un regulated follows from borrowing:
  - Class is unregulated if:
    - It is under its limit
      or
    - It has parent below Top Level which is under its limit