QoS and Packet Scheduling
Corso di Tecnologie di Infrastrutture di Reti

Carlo Augusto Grazia

Department of Engineering *Enzo Ferrari*
University of Modena and Reggio Emilia

Modena, 15th May 2017

Overview

- Quality of Service
  - What is it?
  - Why is it important?
- QoS Vs TCP/IP stack
  - Different layer $\rightarrow$ different QoS def.
- QoS in IP networks
  - Buffers
  - Packet Scheduling
  - Active Queue Management

Quality of Service

?

Quality of Service

Quality of Service: What is it?

QoS: Defined by the ITU in 1994

is the overall performance of a telephony or computer network

Quantitative measured in:

- error rates
- bandwidth
- throughput
- transmission delay
- jitter
- fairness
- ...

Quality of Service: Why is it important?

QoS

is particularly important for the transport of traffic with special requirements. (e.g VoIP, VIP, streaming, FTP)

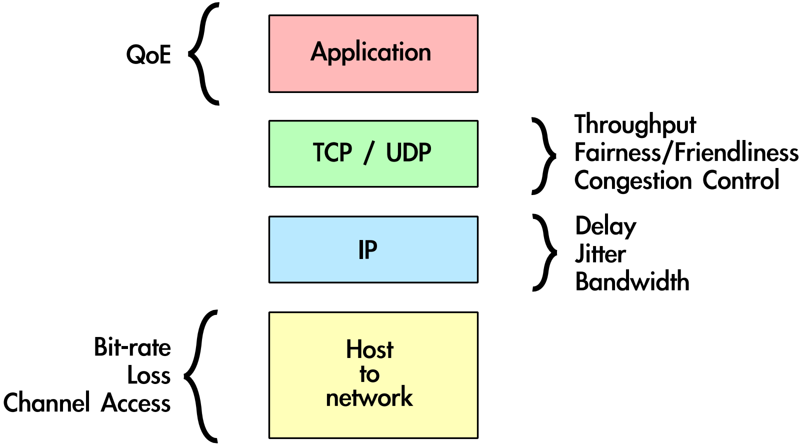
Different applications means different requirements → different QoS

Application	loss	bandwidth	time-sensitive
File transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web browsing	no loss	elastic (few kbps)	no
VoIP	loss-tolerant	[few kbps, 1 Mbps][1]	100s msec
VIP	loss-tolerant	[10 kbps, 5 Mbps][1]	100s msec
Stored audio/video	loss-tolerant	like VoIP and VIP[1]	few seconds
Gaming	loss-tolerant	[few kbps, 10 kbps]	100s msec
Chat	no loss	elastic	depends

[1] VoIP and VIP have also hard jitter constraint. Why stored audio/video not??

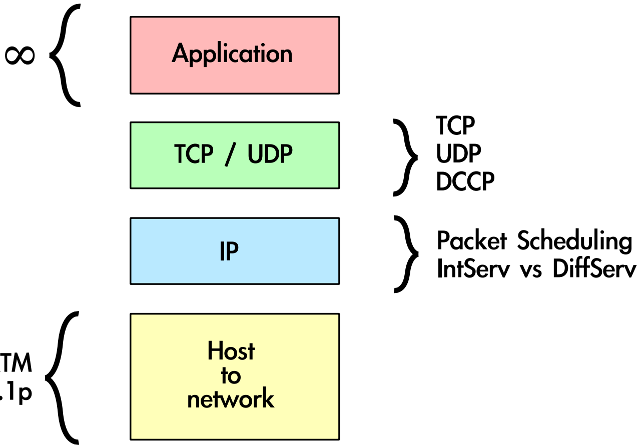
QoS vs TCP/IP

What's your QoS performance metric?



QoS vs TCP/IP

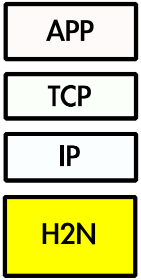
What's your QoS tech?



QoS at Layer 1

QoS is “hidden” at link layer:

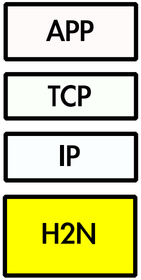
- Loss
  - channel/modulation quality
  - CRC
- Delay
  - Tx delay
  - channel bandwidth
- Time varying link
  - adaptive modulation
  - models for channel estimation



QoS at Layer 2

QoS has born for layer 2:

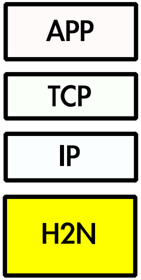
- Frame Relay
- ATM
- 802.x family



QoS at Layer 2: 802.1p

3-bit field called the Priority Code Point (PCP) within an Ethernet frame header:

PCP	Priority	Traffic Type
1	0 (lowest)	Background
0	1	Best Effort
2	2	Excellent Effort
3	3	Critical Applications
4	4	Video, <100 ms latency and jitter
5	5	Voice, <10 ms latency and jitter
6	6	Internetwork Control
7	7 (highest)	Network Control



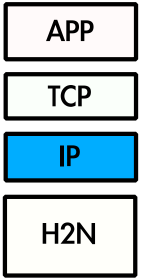
QoS at Layer 3

Encapsulate Layer 2 QoS in Layer 3 is not enough.
Module involved:

- Packet scheduler
- Routing protocol

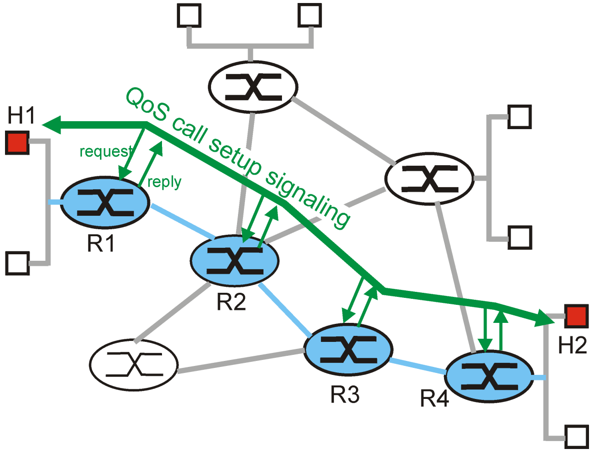
The main choice is between:

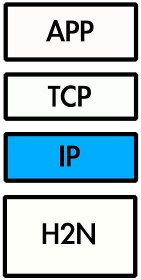
- IntServ
- DiffServ



QoS at Layer 3: IntServ protocol

Fine-grained QoS system based on RSVP:

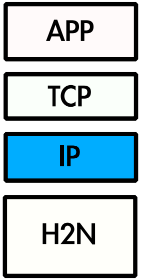




QoS at Layer 3: IntServ protocol

fine-grained QoS system based on RSVP:

- Pros
  - audio/video flow without interruption
  - easy guarantees definition
- Cons
  - all routers along the path must support it
  - no scalable
  - stateful
  - advances setup required
  - impractical for large networks (e.g. internet)
  - efficiency

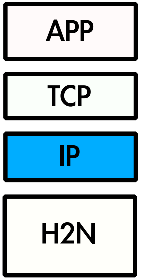


Still important and feasible for data-center or autonomous networks (e.g. bank or intranet)

QoS at Layer 3: DiffServ protocol

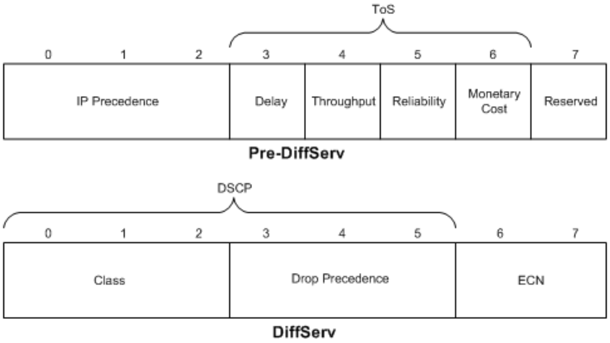
coarse-grained QoS system based on
per-hop behavior and traffic classification:

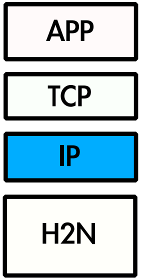
- Pros
  - low latency for audio/video
  - best effort for non-critical services
  - no advanced setup requirement
- Cons
  - different routers could have different QoS behavior
  - end2end perf $\neq \sum$ per-hop perf
  - extra protocol needed (e.g. packet scheduling)



QoS at Layer 3: DiffServ protocol

DiffServ principle → traffic classification.
Classification (and Per-Hop Behavior (PHB)) using the 6-bit DSCP of IP packet field (ToS is deprecated).



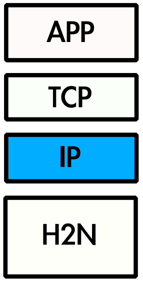


QoS at Layer 3: DiffServ protocol

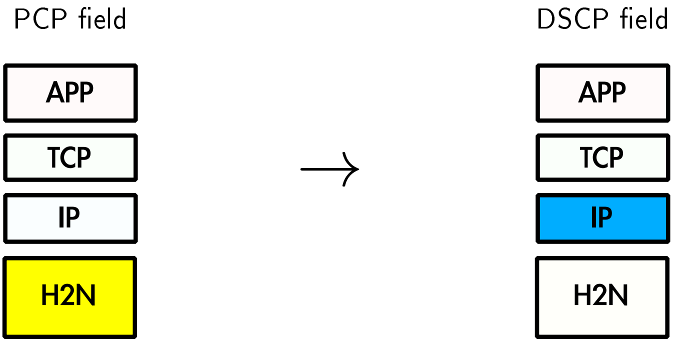
Theoretically 64 different class of service (i.e. 2^6). Intra-class division also possible, using src/dst address and service type.

Standard Per-Hop Behavior:

- Default PHB — best-effort traffic
- Expedited Forwarding (EF) PHB — low-loss, low-latency traffic
- Assured Forwarding (AF) PHB — assurance of delivery
- Class Selector PHBs — gives backward compatibility with the IP Precedence field.



Merging Layer 2 and Layer 3 QoS



Merging Layer 2 and Layer 3 QoS

Cisco Router family RV180 / RV180W

Automatic mapping between 802.1p PCP class of service and the equivalent DSCP packet field one

Merging Layer 2 and Layer 3 QoS

Standard mapping between PCP and DSCP

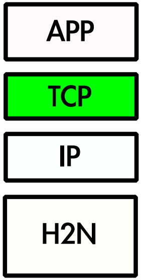
Lv2	Lv3		
PCP	**DSCP**	**PHB**	**Application**
0	0	0	Best Effort
1	8	CS1	Torrent
1	10	AF11	Bulk Data
2	16	CS2	Network Management
2	18	AF21	Transactional Data
3	24	CS3	Call Signaling
3	26	AF31	Mission-Critical Data
4	32	CS4	Streaming Video
4	34	AF41	Video Conferencing
5	46	EF	Voice
6	48	CS6	Routing
7	56	CS7	Network Control

just an example, DSCP could refine the classification (more and more)

QoS at Layer 4

Transport layer is a neglected area concerning QoS.
Two main protocols:

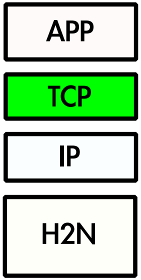
- TCP
  - Congestion Control
  - Fairness among flows
  - Friendliness among TCP algos
- UDP
  - NO Congestion Control
  - Problems delegated to level 3



QoS at Layer 4: TCP

Not created for QoS but QoS could be evaluated:

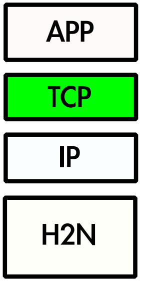
- Congestion Control
  - Agressive vs Careful
  - Avoid Congestion means avoid lot of QoS problems
- Fairness among flows
  - Flows of the same type should have the same bw
  - Flows of the same type whit different RTTs?
- Friendliness among TCP algos
  - Fairness between flows of different TCP algos



QoS at Layer 4: UDP

Not created for QoS and QoS is difficult to evaluate:

- NO Congestion Control
  - Agressive!
- Problems delegated to level 3
  - QoS is completely delegated to bottom layers
- DCCP
  - UDP + Congestion Control
  - At least avoid congestion to help bottom layers in QoS provisioning

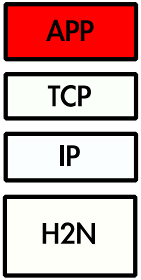


QoS at Layer 5

At the application layer the formal view of QoS is hard to achieve ...

... and QoS became ...

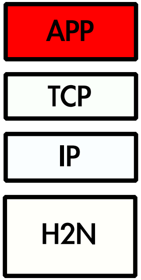
... Quality of Experience (QoE).



QoS at Layer 5: QoE

If QoS is complicated to define, QoE is worse:

- measure of a customer's experience with a service
  - completely subjective
  - NOT formal
- related to but differs from QoS
  - is the human QoS
- multidisciplinary
  - social psychology
  - cognitive science
  - economics
  - engineering science



QoS in IP network

Quality of Service
in IP networks

QoS in IP network: Buffer's role

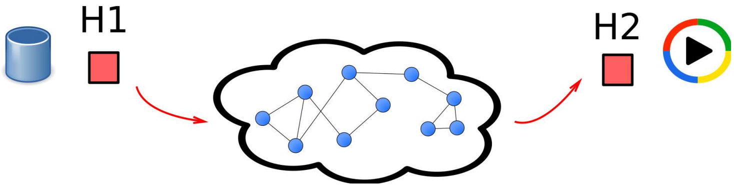
Why we need buffers?

- Sender side
    - save bursts of data to be send
    - wait for ACK (TCP)
- Receiver side
    - save bursts of data received
    - reordering problem
    - playback buffer (Audio/Video)
- Nodes on the path
    - store & forward technique
    - congestion management

Buffers: fact of life

Learn through an example

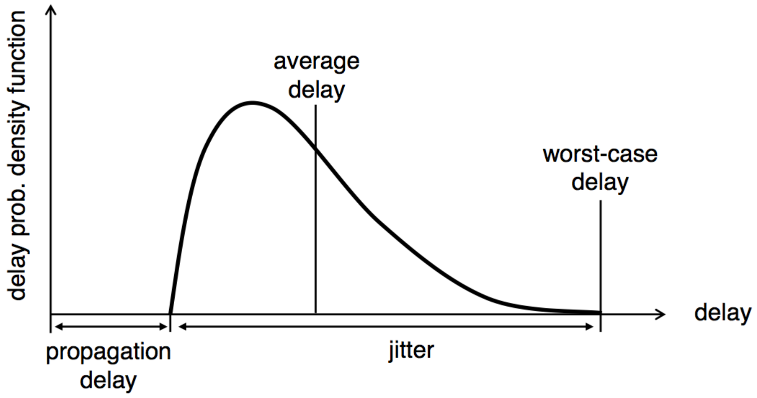
Host 2 wants to play an internet video stored in Host 1



Delay Performance at the Receiver

Stored video to play has particular performance bound (see table at slide 5)

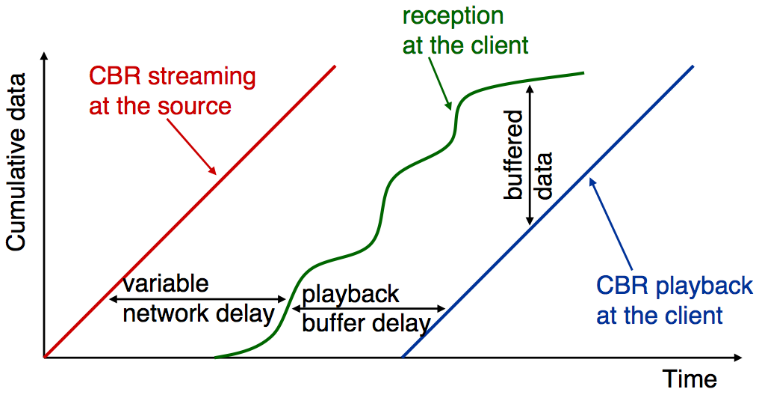
- Delay bound: video should start before a few seconds buffering
- Jitter bound: no delay variation between frames!



Delay Performance at the Receiver

The receiver buffer can compensate the delay variation (jitter) by:

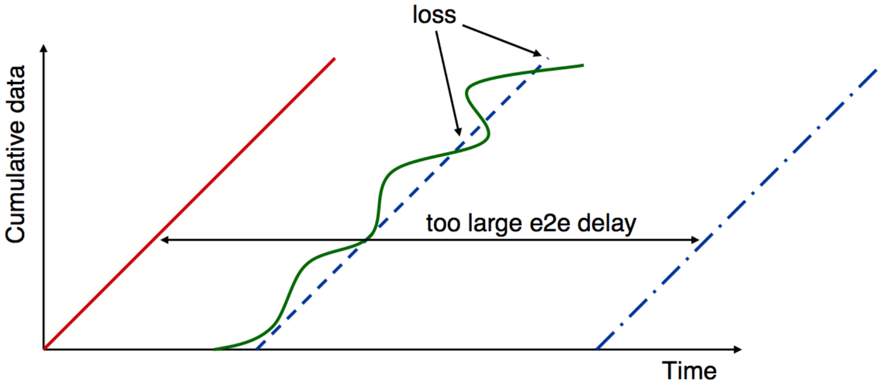
- delaying the first packet in an elasticity buffer
- playing back packets at a constant rate from the buffer (emulate the sender)



Delay Performance at the Receiver

Tuning the receiver buffer size:

- if too short, it will cause losses (frame losses)
- if too large, it will affect interactivity



Network Performance

Receiver buffer recap:

- helps in “playback” stored multimedia contents
- should be properly dimensioned
- mask delay/jitter issue for NON real time application

In case of real-time application the receiver buffer is not enough, in a network we find:

- buffers in intermediates nodes
- scheduling disciplines to choose next packet to transmit
  - fairly share the resources
  - provide performance guarantees

Packet Scheduling: a first look

Purpose:

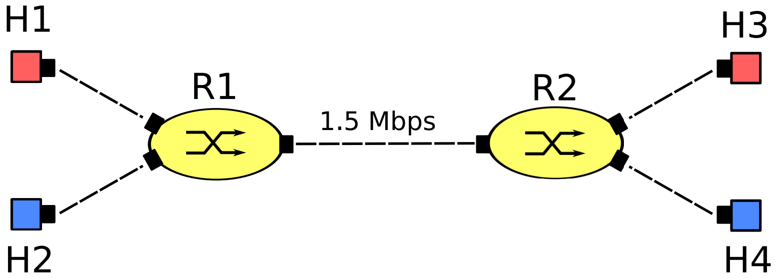
- choose next packet to send on link

Constraints for a packet scheduler:

- not too expensive in terms of required hardware
- fast!!
- scalable (independent from the connections number)
- fair (fairly share the link capacity)
- protective (malicious flows do not affect other flows' performance)

QoS in IP network: Packet Scheduling

Learn through an example[2]

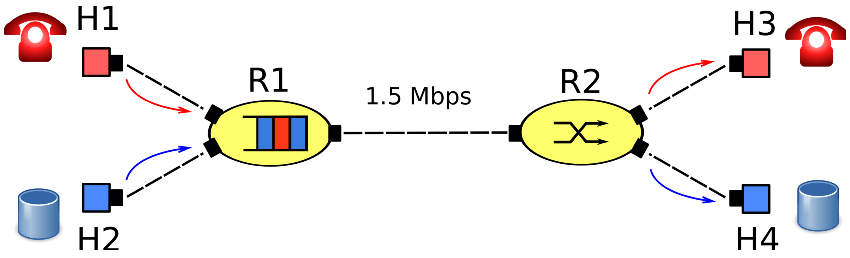


[2]Easy to deploy with ns3

QoS in IP network: Packet Scheduling

Our case-study example: 1Mbps IP phone and FTP share 1.5 Mbps link.

- only VoIP no problem ... (example of playback buffer)
- FTP could congest the network and cause:
  - delay increment
  - delay variation (jitter)
  - both problems for VoIP!!

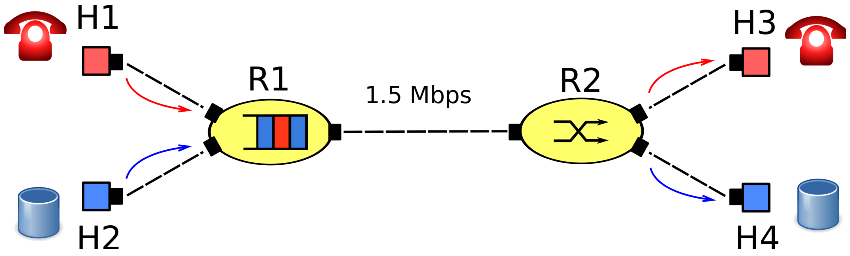


QoS in IP network: Packet Scheduling, Principle 1

Principle 1

we need to distinguish among packets belonging to different classes of traffic (VoIP vs FTP in the example), so, we need:

- a packet marker
- a router policy to treat packets accordingly (**packet scheduler**)



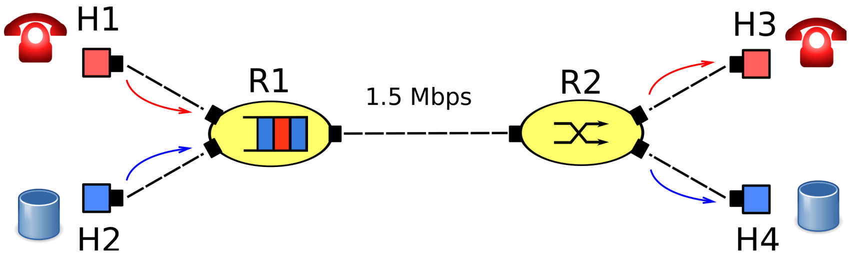
in the figure, FIFO is not enough :)

QoS in IP network: Packet Scheduling, Principle 2

Principle 2

provide protection (*isolation*) for one class from others, for example if:

- VoIP sends higher than declared rate
- FTP sends more until to congest the network!



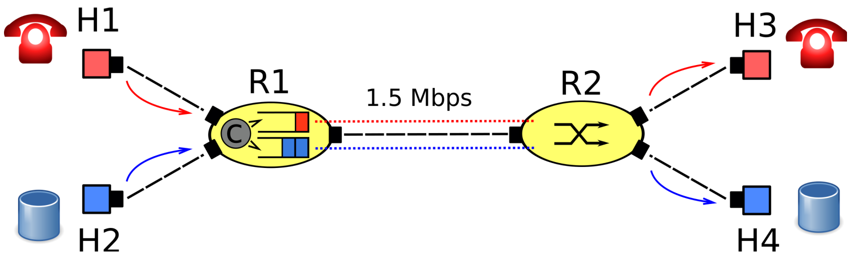
in the figure, FIFO is not enough :)

QoS in IP network: Packet Scheduling, Principle 3

Principle 3

While providing isolation among flows, it is desirable to use resources as efficiently as possible, example:

- link at 1.5 Mbps
- VoIP at 1 Mbps
- FTP with $\leq$ 0.5 Mbps is not efficient!

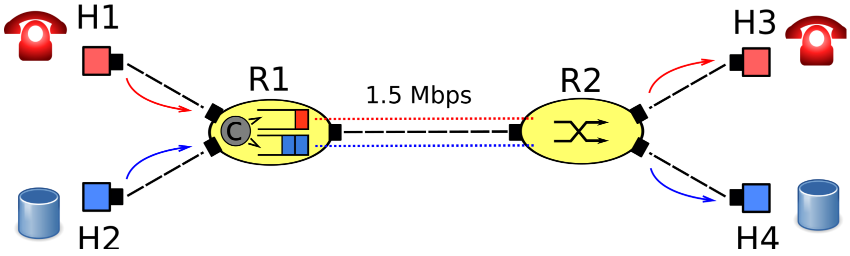


Packet Scheduling

And now?

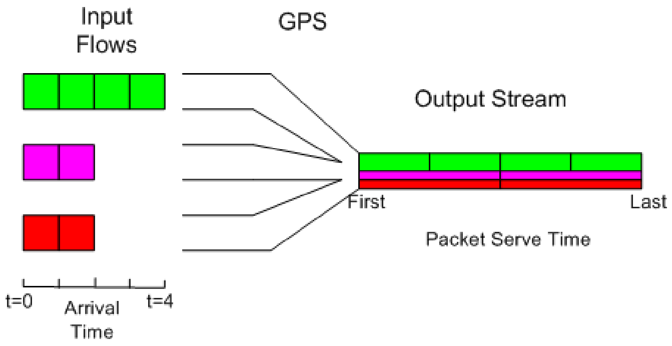
How to choose the scheduling algorithm?

How many packet schedulers exist?



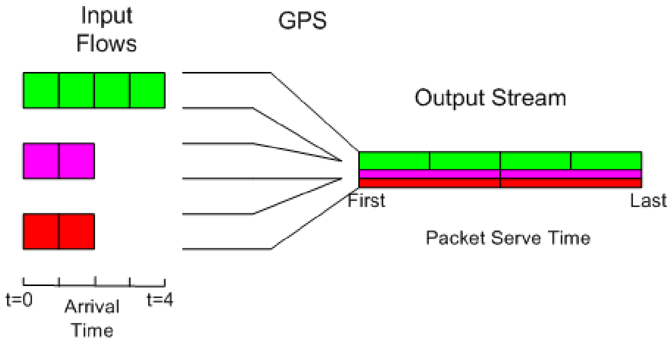
Packet Scheduling: Theory

- main requirement is fairness
- achievable using Generalized processor sharing (GPS)
  - visit each non-empty queue in turn
  - serve infinitesimal from each
  - fair like the fluid system problem



Packet Scheduling: Theory

- GPS is unimplementable! :(
  - we cannot serve infinitesimals, only packets
- **FACT**: NO packet discipline can be as fair as GPS
  - while a packet is being served, we are unfair to others



Packet Scheduling: Theory

- Degree of unfairness can be bounded

Definition: $W_i(t_1, t_2)$
number of bits transmitted by flow i in $[t_1, t_2]$ interval

- absolute fairness bound for scheduler S:

$$\max_i\{W_i^{GPS}(t_1, t_2) - W_i^S(t_1, t_2)\} \qquad \forall\, [t_1, t_2]$$

- relative fairness bound for scheduler S:

$$\max_{i,j}\{W_i^S(t_1, t_2) - W_j^S(t_1, t_2)\} \qquad \forall\, [t_1, t_2]$$

with i and j of the same weight, otherwise, normalize it

Type 1: FIFO

FIFO

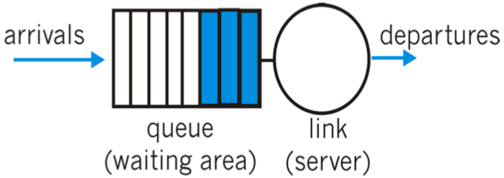
First In First Out scheduling: send in order of arrival to queue

Pros:

- fast, O(1) time complexity

Cons:

- no packet distinction (Principle 1)
- no insolation between different services (Principle 2)
- unfair: Flows of larger packets get better service



Type 2: PRIO

PRIO

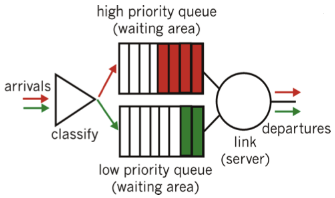
Priority scheduling: Multiple priority classes, each has its own queue

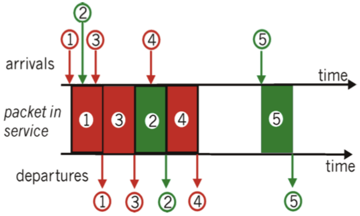
Pros:

- mark packets, multiple queue (Principle 1), based on src/dst IP or port or DSCP field
- insolation for high priority flow (Principle 2)

Cons:

- insolation/starvation for low priority flows (Principle 2)
- priority management is O(1)...O(logn)...O(n)





Type 3: RR

RR

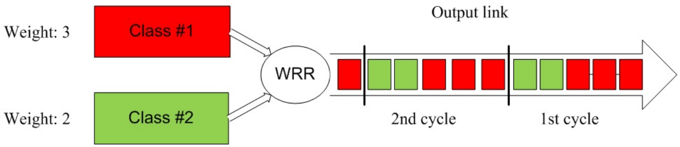
Round Robin scheduling: cyclically scan class queues, serving one packet from each class (if available)

Pros:

- fast, O(1) time complexity
- mark packets, multiple queue (Principle 1)
- no greedy advantage (Principle 2), work-conserving (Principle 3)

It looks like THE solution! ... but ... Cons:

- unfair, O(n) deviation from optimal service
- works bad with different packet sizes



Type 4: Timestamp based Schedulers

Timestamp based Schedulers

Timestamp based schedulers emulate a fluid scheduler, the GPS one, as follows:

- compute, at each time, how much service the flow would receive in the Fluid system (*Virtual Time*)
- mark packet with their **Start** and **Finish** time in the fluid system
- schedule packets according to their **Finish** times
- to reduce burstiness, do not consider packets that have not started yet in the fluid system

Type 4.1: WFQ

Weighted Fair Queueing Scheduler

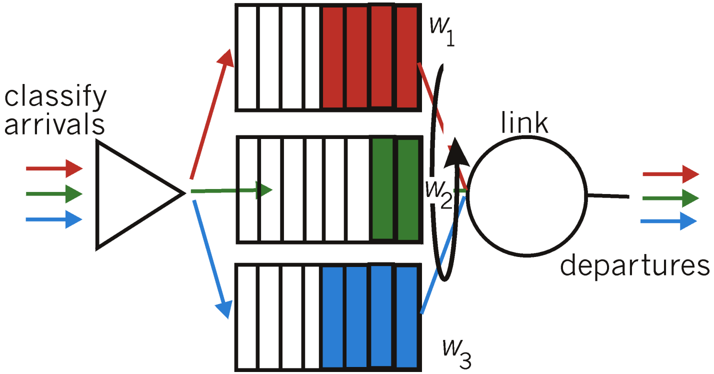
Timestamp based schedulers emulate a fluid scheduler, the GPS one, as follows:

- each flow i is given a weight w_i
- service rate received by flow i is:

$$r_i = \frac{R \cdot w_i}{w_1 + w_2 + \ldots + w_n}$$

where R is the link rate

Type 4.1: WFQ



Type 4.1: WFQ

Pros:

- looks fair: departure time of a WFQ packet is always $\leq$ of the departure time of GPS *fluid* packet plus a maximum packet service time
- gives Principle 1, 2 and 3

Cons:

- $\Omega(\log n)$ time complexity, due to timestamps (and keep it sorted)
- not good for Jitter bound

An $\Omega(\log n)$ time complexity looks, at a first glance, not too much! In our examples just 1, 2 o 3 flows are considered. Backbone routers manage several K flows!!!

Type 4.2: WF^{2}Q

Worst-case Fair Weighted Fair Queueing Scheduler

Optimal service-guarantees variant of WFQ

- departure time of a WFQ packet is always $\leq$ of the departure time of GPS *fluid* packet plus a maximum packet service time

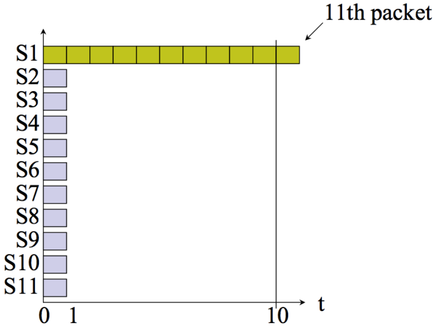
$$t_{WFQ}^{start}(pkt_i) \leq t_{GPS}^{start}(pkt_i) + t_{max} \qquad \forall i$$

- but WFQ might be well *ahead* of GPS!

Type 4.2: WFQ vs WF^{2}Q

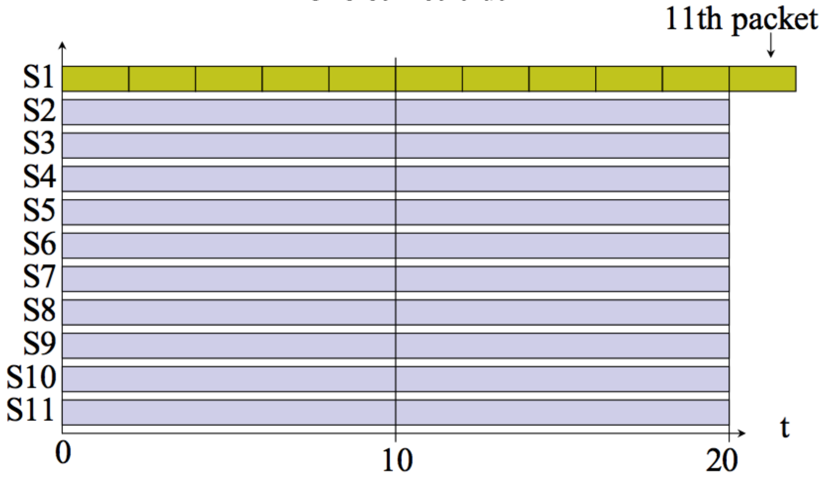
Learn through an example:

- 11 flows/services $S_1 \ldots S_{11}$
- S_1 has 0.5 of the link rate R
- $S_2 = S_3 = \cdots = S_{11}$ have 0.05 of R
- packet length of 1 second (space length / R is 1 second)



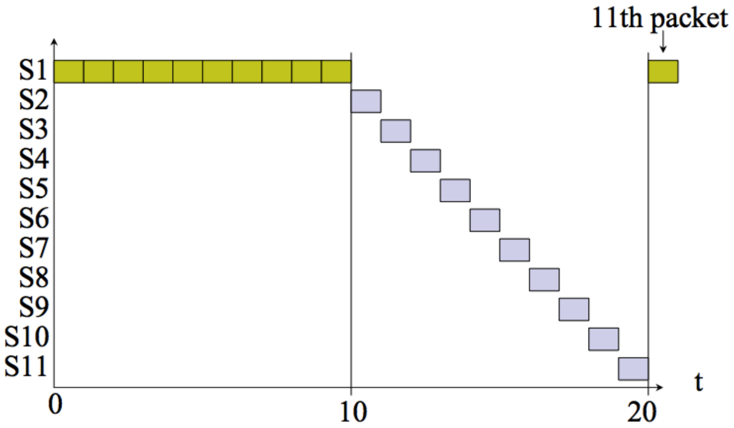
Type 4.2: WFQ vs WF^{2}Q

GPS service order



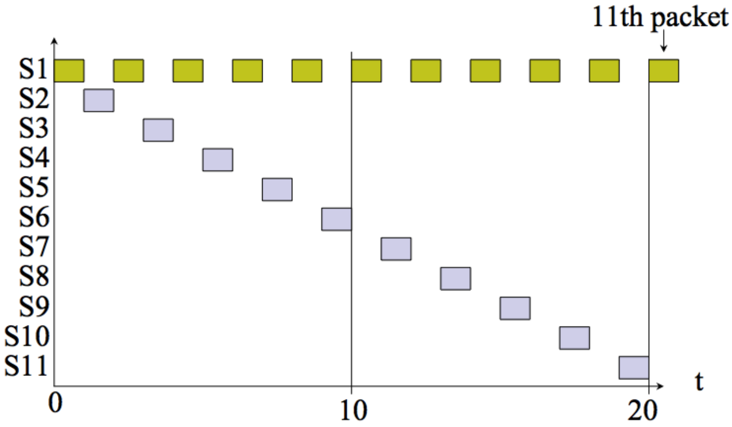
Type 4.2: WFQ vs WF^{2}Q

WFQ service order



Type 4.2: WFQ vs WF^{2}Q

WF^{2}Q service order



Type 4.2: WF^{2}Q

Pros:

- **optimal** service B-WFI (bit Worst-Case Fair Index) of 1MSS def. as:

$$\max_{i,\Delta t}\{\phi_i \cdot W(\Delta t) - W_i(\Delta t)\}$$

- gives Principle 1, 2 and 3

Cons:

- $\Omega(\log n)$ time complexity

Resources

- HFS details on my page: `http://www.dii.unimo.it/wiki/index.php/Carlo_Augusto_Grazia`
- Networks Simulation lesson and ns3 `http://www.dii.unimo.it/wiki/images/b/ba/LessonNetworksSilmulation.pdf`
- “GoogleTechTalks qfq”: `http://info.iet.unipi.it/~luigi/qfq/`
- P. Valente, “Providing Near-Optimal Fair-Queueing Guarantees at Round-Robin Amortized Cost” `http://algo.ing.unimo.it/people/paolo/agg-sched/agg-sched.pdf`
- GPS problem: `http://en.wikipedia.org/wiki/Generalized_processor_sharing`
- WFQ : `http://en.wikipedia.org/wiki/Weighted_fair_queuing`

Contacts

UNIVERSITÀ DEGLI STUDI
DI MODENA E REGGIO EMILIA

carloaugusto.grazia@unimore.it

extra slides

Type 4.3: QFQ+

Quick Fair Queueing Plus

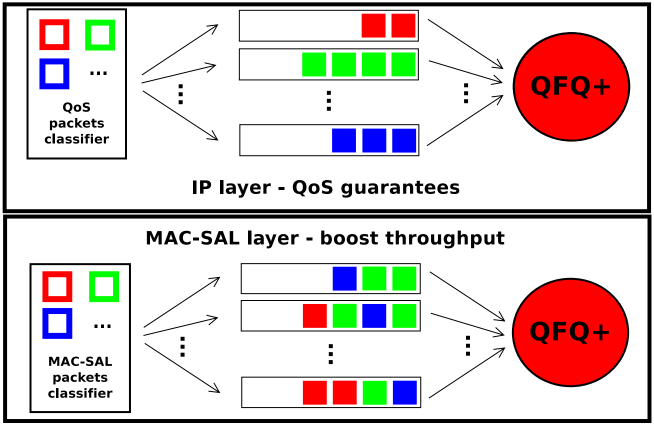
State-of-the-art packet scheduler, currently in Linux Kernel from 3.8

- O(1) time complexity
- near-optimal guarantees (B-WFI $\sim$5 MSS)
- it's real, not just a paper design
- also faster than RR (due to proper cache use)

Type 5: Packet Scheduler for Wireless environment

HFS: High-throughput twin Fair Scheduler

QoS layer: quasi-optimal service guarantees, cost close to DRR
MAC-SAL layer: high throughput, quasi-optimal service guarantees, cost close to DRR


QoS packets classifier
QFQ+
IP layer - QoS guarantees
MAC-SAL layer - boost throughput
MAC-SAL packets classifier
QFQ+

Type 5.1: HFS

HFS Architecture

feasible, flexible and modular architecture which decouples QoS guarantees and link issues tasks

High-throughput twin Fair Scheduler

flexible, efficient and green packet scheduler for wireless links

- throughput higher than W^2F^2Q
- T-WFI and B-WFI close to WF^2Q+
- O(1) time complexity
- low energy consumption due to:
  - increase throughput $\rightarrow$ more packets successfully transmitted per energy consumed $\rightarrow$ less retransmission $\rightarrow$ **less power consumption**
  - low execution time per packet processing $\rightarrow$ **less power consumption**

Active Queue Management (AQM)

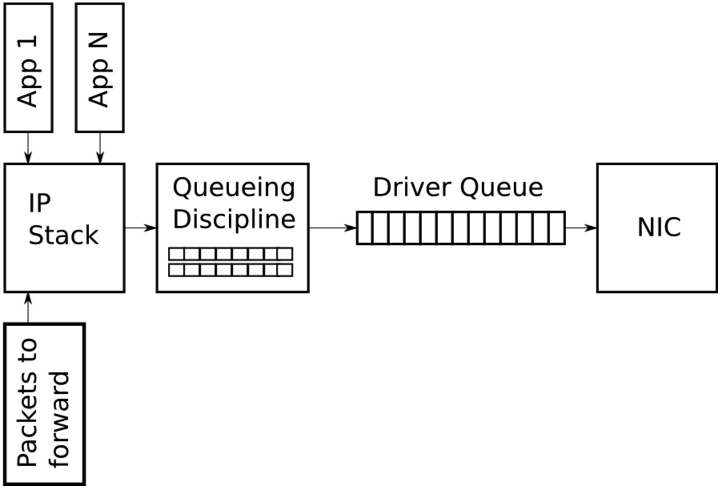
Is it a sort of Packet Scheduling?? —> NO

RFC 2309

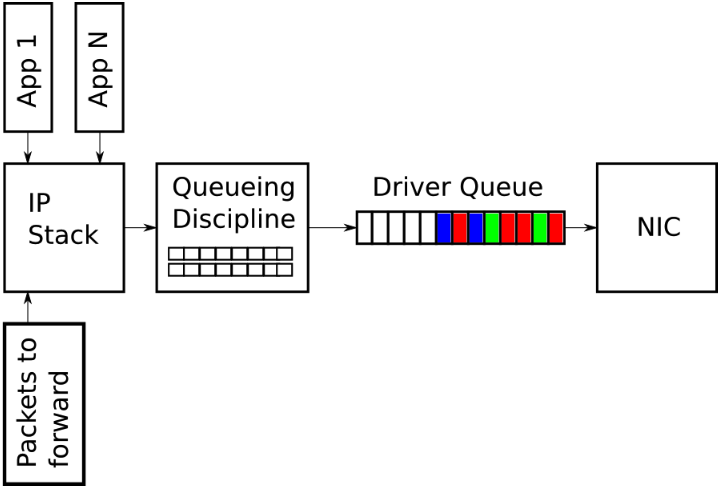
It is useful to distinguish between two classes of router algorithms related to congestion control: "queue management" versus "scheduling" algorithms. To a rough approximation, queue management algorithms manage the length of packet queues by dropping packets when necessary or appropriate, while scheduling algorithms determine which packet to send next and are used primarily to manage the allocation of bandwidth among flows. While these two router mechanisms are closely related, they address rather different performance issues.

- AQM is a smoochy and *colorful* window on networks algorithms - :)

AQM schema

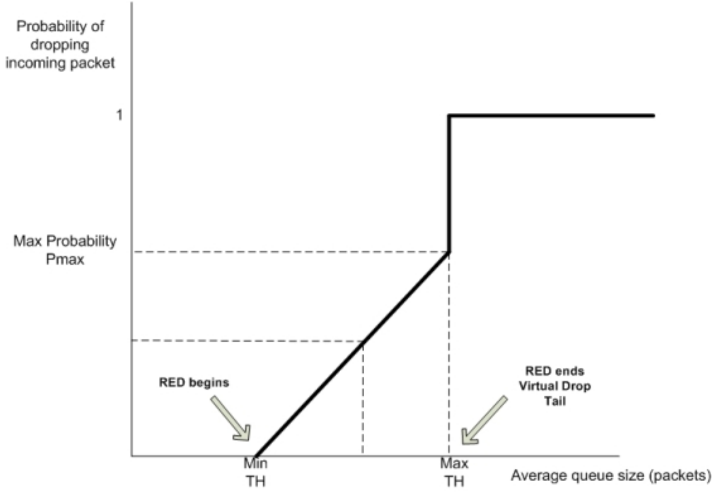

App 1
App N
IP Stack
Packets to forward
Queueing Discipline
Driver Queue
NIC

AQM schema

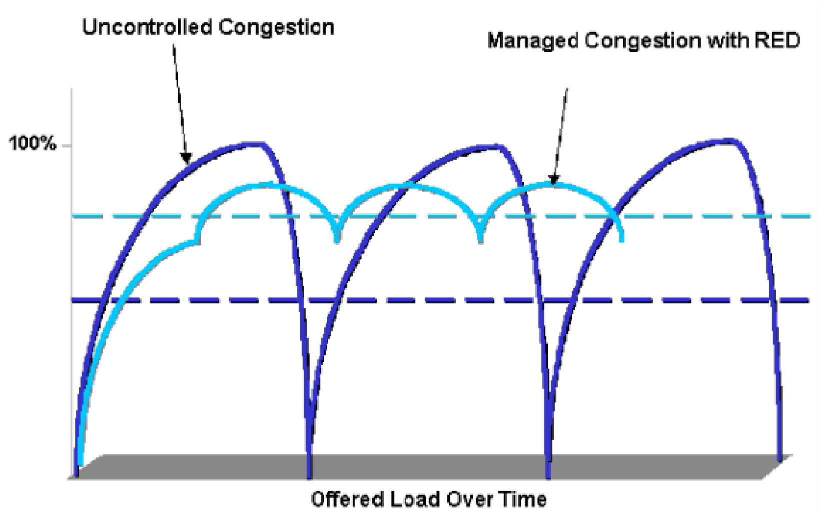

App 1
App N
IP Stack
Queueing Discipline
Driver Queue
NIC
Packets to forward

Random Early Detection (RED)

One of the first Congestion Avoidance queueing algorithm (AQM) produced. Tons of variants available! Most of the Switch/Router deploy it.



RED Benefits



Other AQM algos

The main competitor

- CoDel: Controlled Delay, a sort of RED designed to be "self-configurable" without human design. Based on the Bufferbloat paradigm.

The literature is full of specific AQM:

- BLUE
- BLACK
- GRAY
- YELLOW
- GREEN

A power-rangers style science!