Quality of Service

1

1

Quality of Service (QoS)

- RFC 2386: "A set of service requirements to be met by the network while transporting a flow"
- Achieve certain, specified levels of service
- Parameters: Throughput, delay, jitter, packet loss, priority, …
- Particularly important for interactive and latency-sensitive applications (telephony, gaming, video streaming, …)

2

2

Why do we need QoS?

- Without QoS: Bufferbloat
  - Router buffers fill up
  - Packets get delayed or dropped
- With QoS: certain packets get treated differently

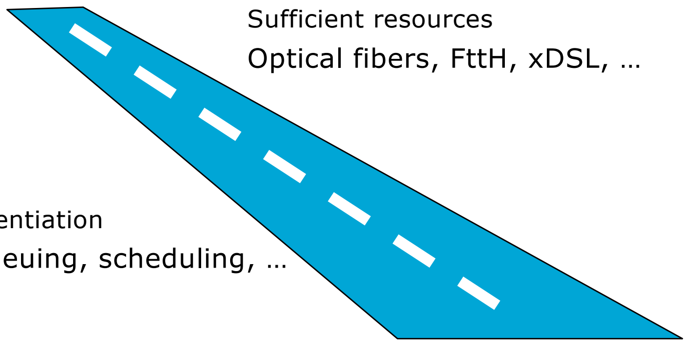
- Best-effort service: default, no QoS mechanisms applied
- Deterministic service guarantees: network guarantees a parameter is kept below/above some threshold for certain application data (e.g., VoIP traffic delay < 5 ms)
- Statistical service guarantees: network guarantees that some percentage of traffic is kept below/above some threshold (e.g., for 80% of data delay < 20 ms)

TUDelft

3

3

The road to QoS



Sufficient resources

Optical fibers, FttH, xDSL, …

Packet differentiation

DiffServ, queuing, scheduling, …

QoS routing (protocol & algorithm)

TUDelft

4

4

What is required for achieving QoS?

- Service Level Specifications
  - Agreements between economic actors on the level of service that should be achieved
- Classification of flows into service classes
- Input control: traffic meters must ensure traffic conforms to expectations (e.g., limit data rate)
- Scheduling of packets at routers

TU Delft

5

5

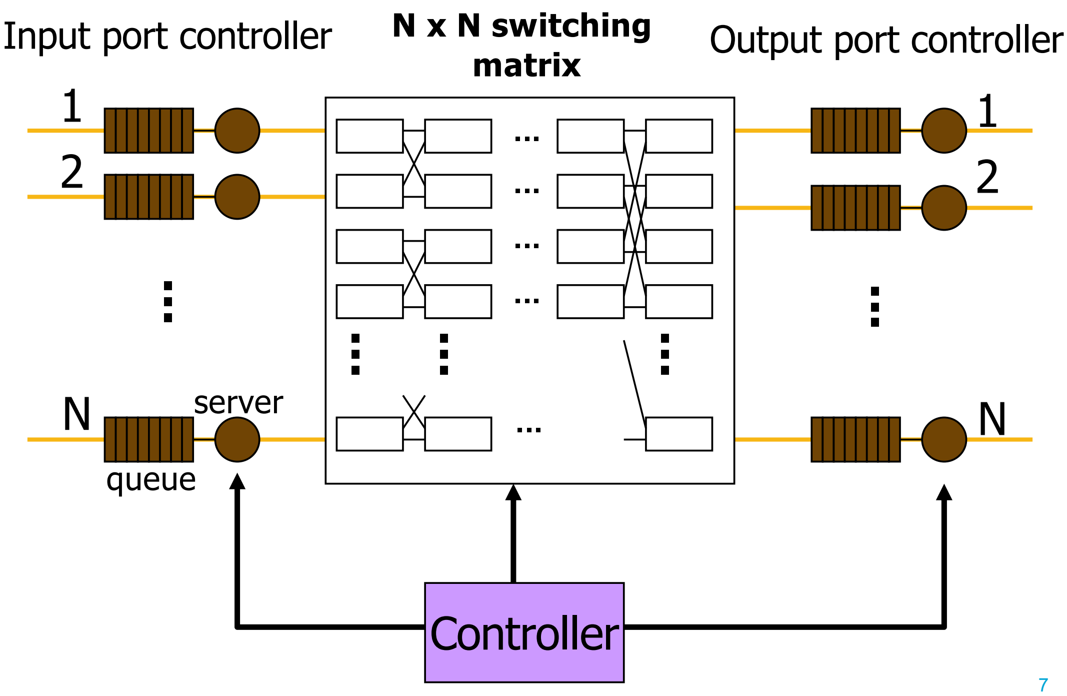
Scheduling

TU Delft

6

6

Router



TUDelft

Rule of thumb

- Short buffers, shorter delay, larger loss
- Large buffers, longer delay, smaller loss
- Loss and delay probabilities are increasing functions of the load

TUDelft

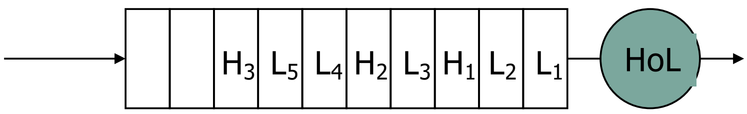
Loss sensitive scheduling

- Assume 2 priority classes: high and low
- Packets in same class are served in FIFO order
- Buffer has K positions
- Head-of-the-Line, Partial Buffer Sharing, Push-out buffer, Random Early Detect

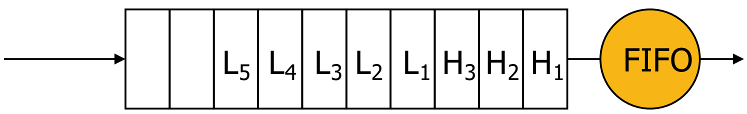
TUDelft

Head of the Line (HoL)

The head of the line loss sensitive scheduling rule always serves high priorities in the buffer before low priorities:



is equivalent to

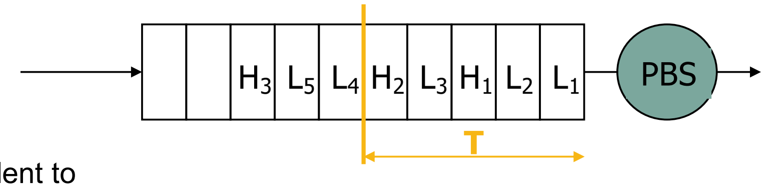


Preemptive: placing back of low priority packet in server to queue
(processing of a packet can get interrupted)
Non-preemptive: an arriving high priority packet has to wait until service of low priority packet has been terminated
(processing of a packet cannot be interrupted)
Priority queuing: multiple priority classes

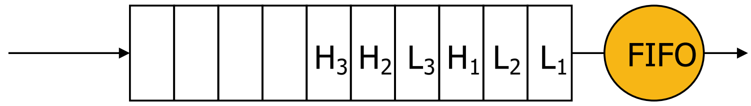
TUDelft

Partial Buffer Sharing (PBS)

Above the threshold T only arriving high priority packets are allowed to enter and arriving low priority packets are discarded.



is equivalent to



Below threshold: identical to FIFO and sequence order is preserved
Above threshold: only a high priority regime until buffer is full

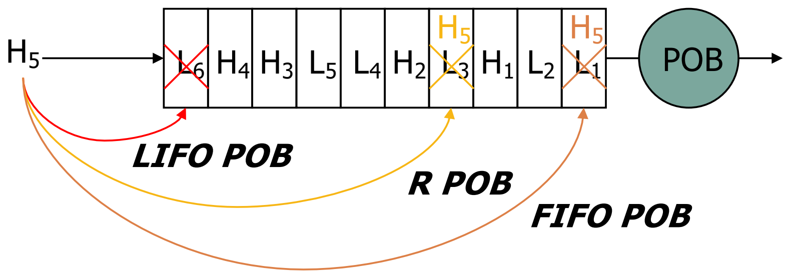
TUDelft

11

11

Push-Out Buffer (POB)

Only if the buffer is full: an arriving high priority packet is allowed to push-out a previously entered low priority packet



LIFO POB: the last entered low priority packet is discarded
FIFO POB: the first entered low priority packet is discarded
R POB: a randomly chosen low priority packet is pushed out

TUDelft

12

12

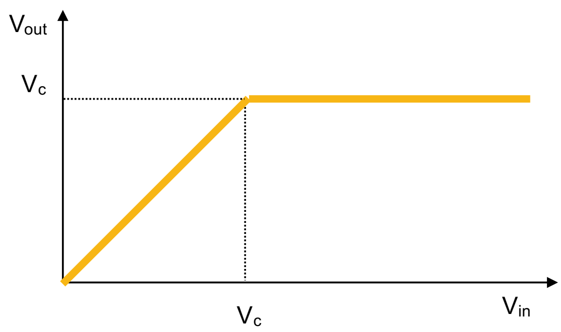
QoS guarantees

TUDelft

13

13

Input Control



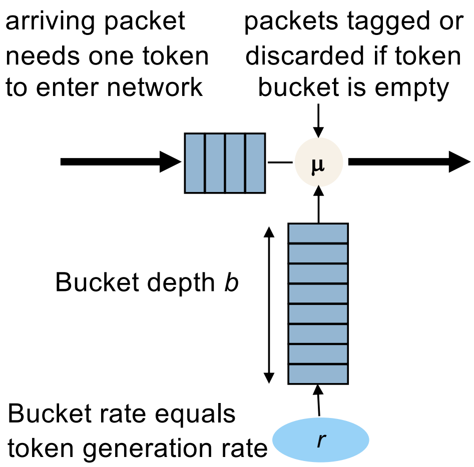
V_{in} : Received rate
V_{out} : Admitted rate
V_c : Contracted rate

TUDelft

14

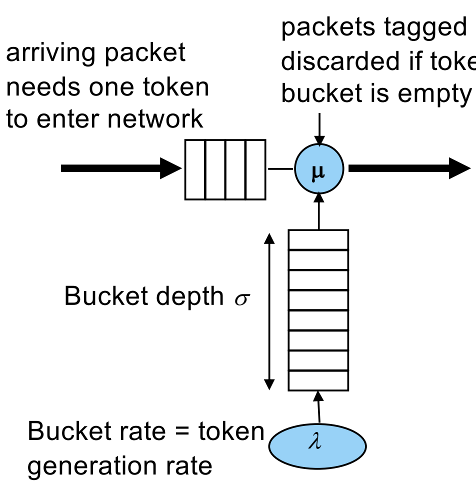
14

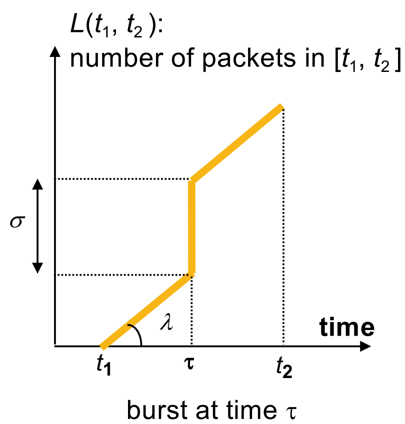
Token Bucket (b,r)



TU Delft

Burstiness Constraint: (σ,λ)





Burstiness Constraint:

$L(t_1, t_2) \leq \lambda\,(t_2 - t_1) + \sigma$

TU Delft

Upper Bound Method

- Each customer arrival process satisfies a certain burstiness constraint
- The service time of each packet is deterministic (= proportional to packet length)
- Each non-empty QoS class k has a minimal service rate
  - Fair: no class can prevent another class from getting served

TUDelft

17

17

Packets in the queue

- Burstiness constraint for class k
- Service times are deterministic
- Fair scheduling policy
- Then queue count $N_k(t)$ is bounded:

$$N_k(t) = \max_u \left[L_k(u,t) - M_k(u,t) : u \leq t\right]$$
$$N_k(t) = \max_u \left[\sigma_k + \lambda_k(t-u) - \mu_k(t-u) : u \leq t\right]$$
$$N_k(t) \leq \sigma_k : \mu_k \geq \lambda_k$$

L is the number of inflow packets (upper-bounded by the burstiness constraint) and M is derived from the minimal service rate.

TUDelft

18

18

Upper bound on delay & loss

- Assume single queue/multiplexer, H QoS classes.
- Zero packet loss if K is larger than sum of burstiness constraints σ_k
- Delays are bounded by σ_k/λ_k
- The deterministic approach shows that packet networks are able to guarantee loss and delay constraints.

TUDelft

19

19

Reserving resources

TUDelft

20

20

Resource reSerVation Protocol (RSVP)

- IETF's first signaling protocol (RFC-2205) based on multicast
- RSVP fundamental message types: *Path, Resv*
- *Path:*
  - previous hop,
  - sender template (describes the format of data packets that the sender will originate),
  - Tspec (specifies traffic characteristics of the sender's data),
  - Adspec (specifies the e2e QoS requirements)
- RSVP fundamental teardown messages: *PathTear, ResvTear*

TU Delft

21

21

Resource ReSerVation Protocol

- RSVP is a *signaling* (and not routing) protocol
- Reservation for unicast as well as many-to-many multicast
- RSVP operates on top of IP (v4 or v6)
- RSVP model is receiver-oriented: receiver initiates and maintains reservation
- Receiver sends reservation requests upstream and each node either accepts or rejects it:
  - Router checks local policy and admission control
  - State info is stored in scheduler and classifier

TU Delft

22

22

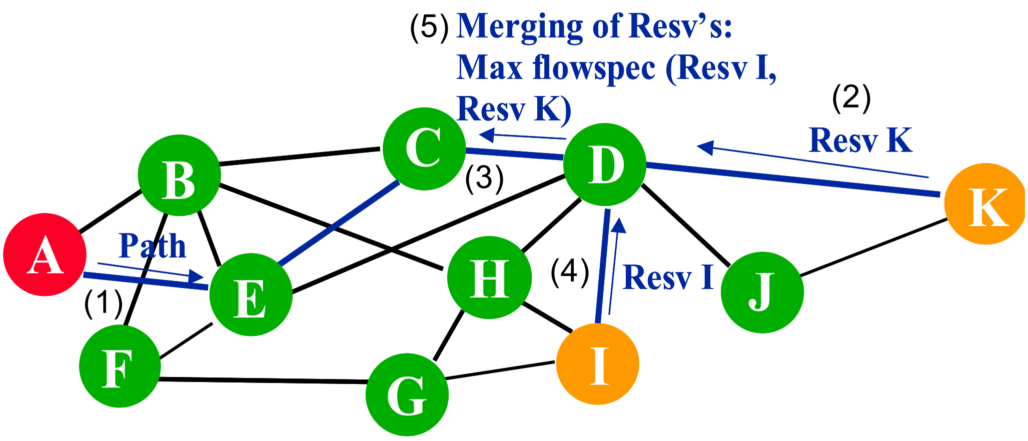
Resource reSerVation Protocol

- RSVP depends on routing protocol
- *Path and Resv* messages are sent periodically to maintain the reservation state along a particular traffic path

QoS state = SOFT STATE

TUDelft

RSVP



TUDelft

QoS protocols

IntServ & DiffServ

TU Delft

25

Integrated Services (IntServ)

- Goal: provide guaranteed QoS
- Additional components:
  - packet classifiers (to identify flows that are to receive a certain level of service)
  - schedulers (to handle the service of different packet flows)
  - admission control (to determine whether a router has the necessary resources to accept a new flow)

TU Delft

26

Integrated Services (IntServ)

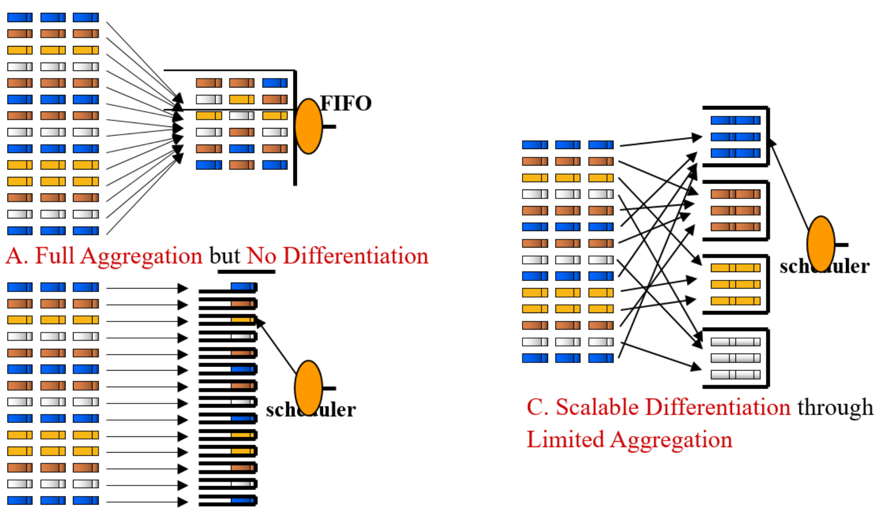
- In combination with RSVP via (Tspec,Rspec)
- Traffic Classes:
  - Best-effort
  - controlled load (RFC2211): probabilistic guarantee
  - guaranteed service (RFC2212): on maximum queuing delay (not average, nor jitter)
- Disadvantage: complex and not scalable!
  - requires routers to keep state for each flow

TUDelft

27

27

Differentiation / Aggregation



A. Full Aggregation but No Differentiation

B. Full Differentiation but No Aggregation

C. Scalable Differentiation through Limited Aggregation

A. BE / No QoS B. IntServ C. DiffServ

TUDelft

28

28

Differentiated Services

- **Problem**: give *scalable* "better" service to some traffic (at the expense of worse service to the rest)
- IETF Differentiated Services (Diffserv) standardizes the "per hop behavior (PHB)" and not the service
  - service consists of "packet forwarding" + "rules"
- Types (besides Best Effort (BE)) of PHB
  - Expedited Forwarding (EF): virtual leased line where users want *absolute* BW independent of other traffic
    - implementation: priority queuing, strict policing
  - Assured Forwarding (AF): 'better' best effort with better control of *relative* BW
    - 4 AF classes, each with 3 drop preferences (green, yellow, red)
    - implementation: drop preference, weighted round-robin, WFQ
  - Other types exist, not covered here

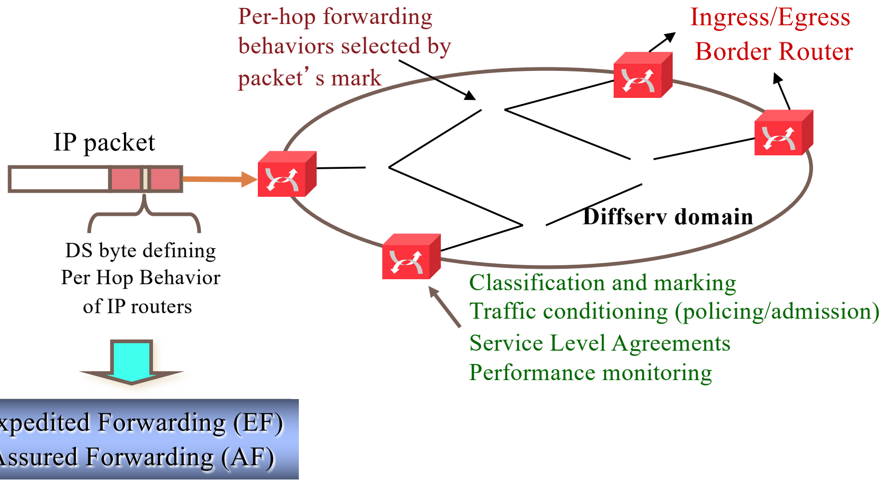
TUDelft

Basic difference between AF and EF

- Services built with EF PHB rely heavily on strict traffic conditioning at the network's boundary so that excess traffic and congestion within an EF Behavior Aggregate (BA) is kept out of the network
- Services built with AF PHB let excess traffic enter the network and use active buffer management at each DS network node to handle congestion within each AF class
- Active buffer management like Random Early Drop (RED) is recommended

TUDelft

Diffserv Architecture

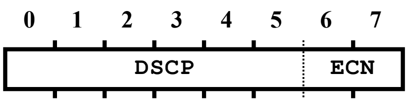


TU Delft

31

31

Diffserv Code Points



BE:	0 0 0 0 0 0
NCT:	1 1 x 0 0 0
EF:	1 0 1 1 1 0
AF:	all others

The semantics of the IPv4 TOS byte (as well as IPv6 Traffic Class octet) are redefined

- DSCP: *Differentiated Services Code Point* index to identify/select the particular Per- Hop Behavior (PHB) an IP datagram is aiming to receive at a given network node
- ECN: *Not used for DiffServ but for Explicit Congestion Notification*
- Unrecognized DSCP should not be changed and should be forwarded transparently and treated as Best Effort traffic.
- NCT : Network Control Traffic

TU Delft

32

32

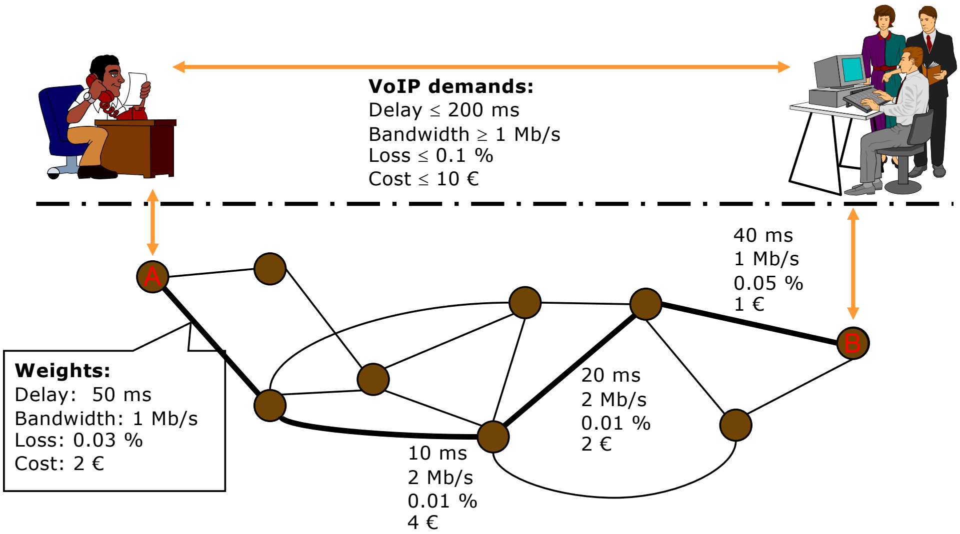
QoS algorithm

TUDelft

33

33

QoS routing problem (SAMCRA)



P. Van Mieghem and F.A. Kuipers, *Concepts of Exact Quality of Service Algorithms*, IEEE/ACM Transactions on Networking, vol. 12, no. 5, pp. 851-864, October 2004.

TUDelft

34

34

QoS concepts in SAMCRA

- Non-linear length for exactness:

$$\max\left(\frac{w_1(P)}{L_1}, \frac{w_2(P)}{L_2}, \ldots, \frac{w_m(P)}{L_m}\right)$$

Subsections of the shortest path are not always shortest paths

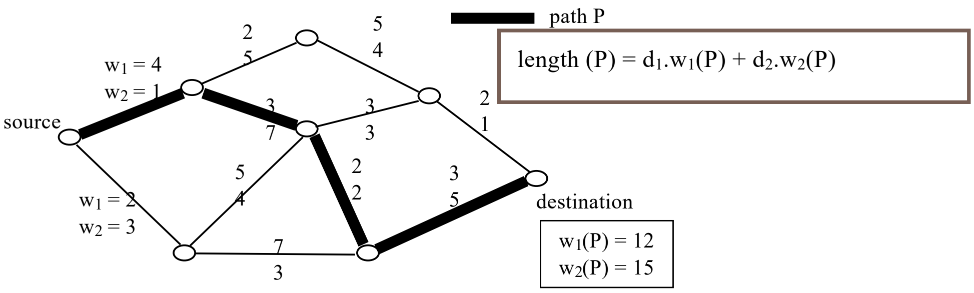
- We must compute *k*-shortest paths
- Reduce search space:
  - Non-dominance
  - Look-ahead

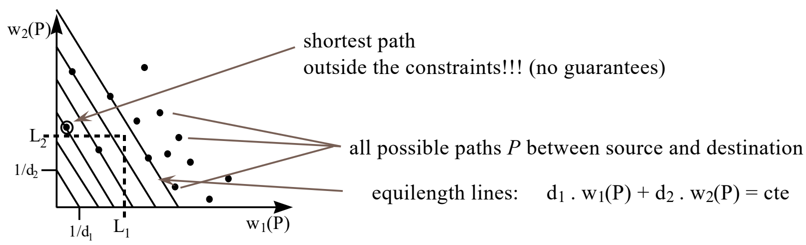
TUDelft

35

35

Linear definition of path length



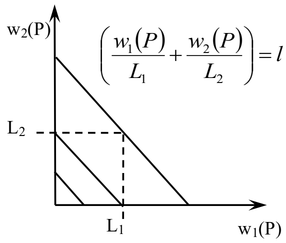


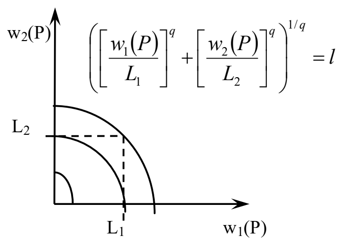
TUDelft

36

36

Non-linear path length



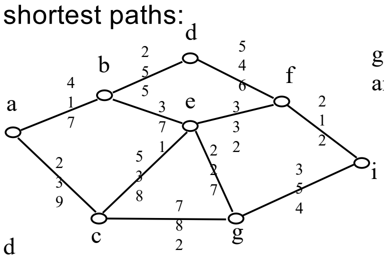


$$\boxed{\text{length (P)} = \left(\sum_{i=1,\ldots,m}\left(\frac{w_i(P)}{L_i}\right)^q\right)^{1/q} \quad m = \text{number of constraints}}$$

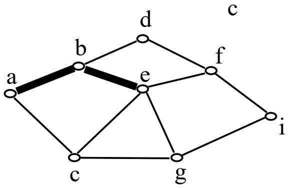
$$\lim_{q\to\infty}\left(\sum_{i=1,\ldots,m}\left(\frac{w_i(P)}{L_i}\right)^q\right)^{1/q} = \max\left(\frac{w_1(P)}{L_1},\frac{w_2(P)}{L_2},\ldots,\frac{w_m(P)}{L_m}\right)$$

k-shortest paths

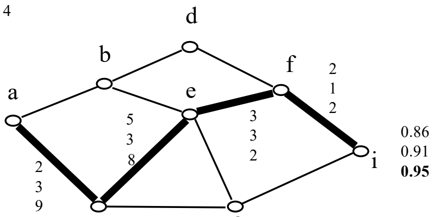
Example that subsections of the shortest path are not always shortest paths:



graph with three measures and three constraints (14, 11 and 22)



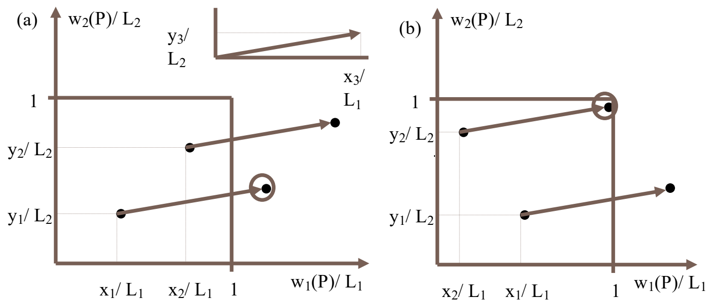
shortest path from a to node e



shortest path from a to node i
2nd shortest path from a to node e

Path Dominance

if $w_i(P_1) \leq w_i(P_2)$ for all i, then P_1 dominates P_2



Reducing search space reduces complexity

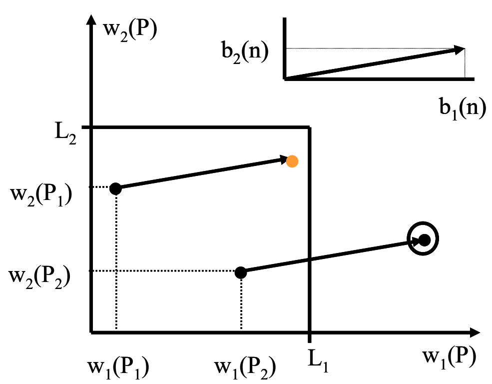
TUDelft

39

39

Look-Ahead

Compute for each of the m link weights the shortest paths tree rooted at B to any node n in network:
lower bounds $b(n) = \{w_i(P^*_{B\to n})\}$ for $1 \leq i \leq m$
Check: $w_i(P_{A\to n}) + b_i(n) \leq L_i$



TUDelft

40

40

Meta-code of SAMCRA

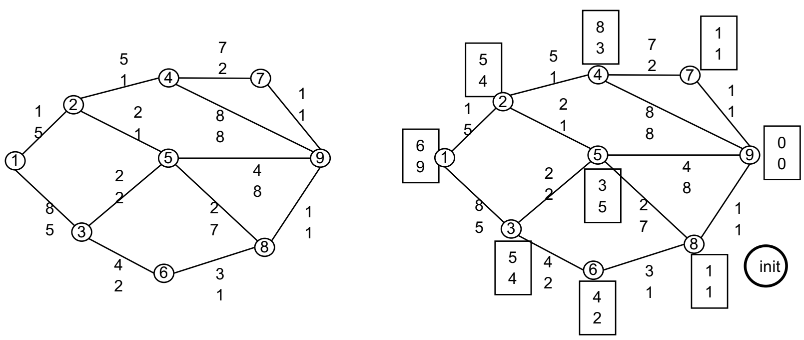
```
SAMCRA(G,m,A,B,L)
Initialize and find lower bounds
While(Q not empty)
  Extract-min(Q) → u[i]
  if u = B, return path
  else for each neighbor v of u
     check for path dominance and length
     if length < maxlength and not dominated
        Insert the path in Q
        if v = B, update maxlength
```

TU Delft

41

41

Operation of SAMCRA



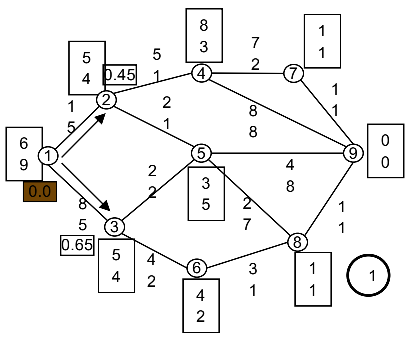
Source 1, destination 9, Constraints: (20,20)
Dijkstra lower bounds in rectangular boxes beside the node

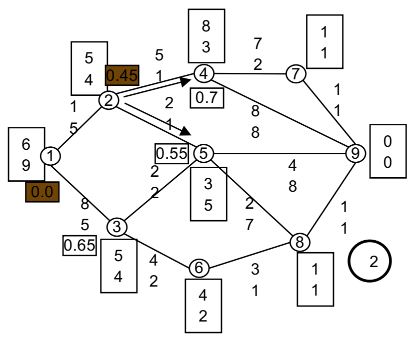
TU Delft

42

42

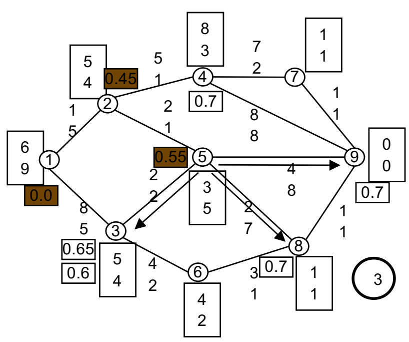
Operation of SAMCRA

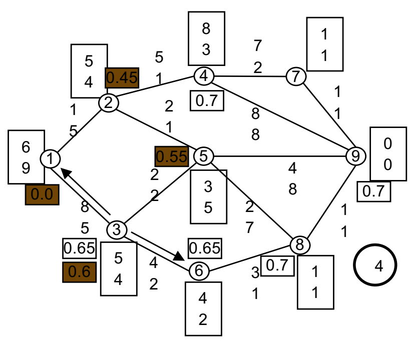




TU Delft

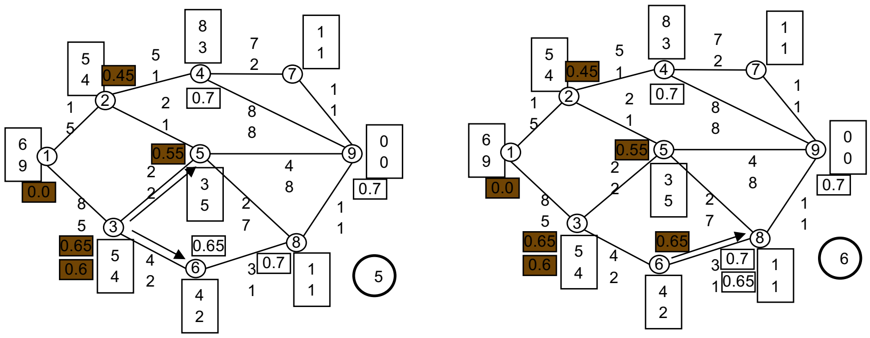
Operation of SAMCRA



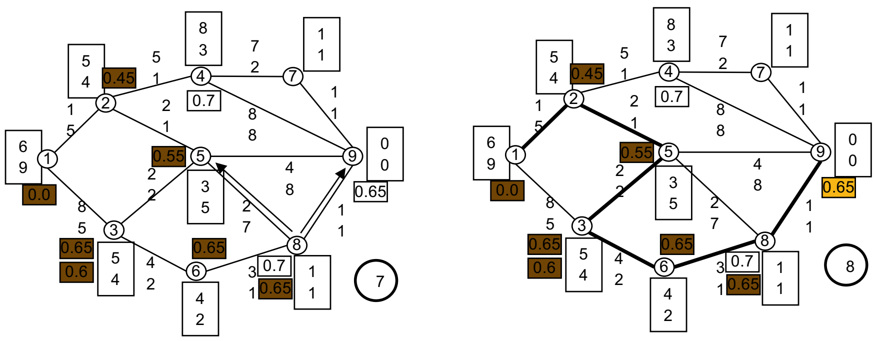


TU Delft

Operation of SAMCRA

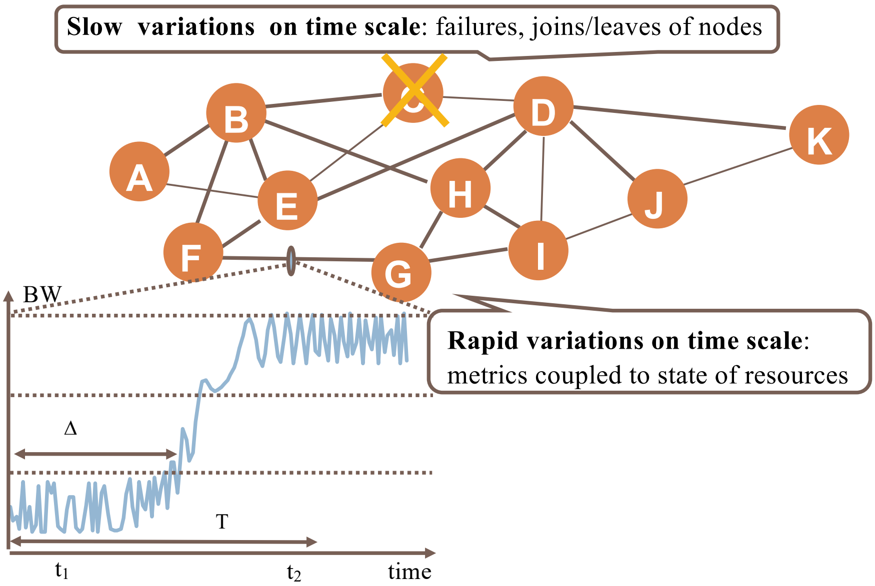


Operation of SAMCRA



QoS dynamics

Topology Changes



QoS dynamics

- Protocol:
  - How to distribute information on available resources?
  - When to distribute information on available resources?
- How to distribute:
  - Flooding -> may be costly if the frequency of updates is expected to be high
  - Tree-based protocols -> information is distributed over predetermined trees (less robust)

TUDelft

49

49

When to distribute?

- Link-state update policies:
  - Periodic LSU policy
  - Trigger-based policy:
    - Class-based (equal or exponential classes)
    - Threshold
  - Additional techniques:
    - Hold-down timer
    - Moving average

TUDelft

50

50

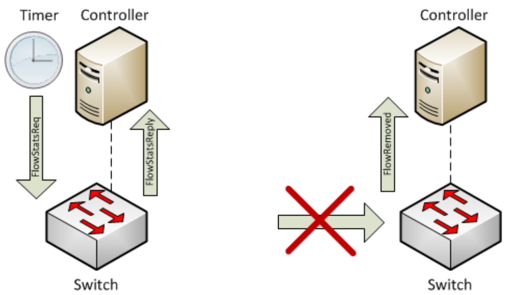
QoS with SDN

51

51

Measuring throughput

- Per-flow counters
  - Packet counter
  - Byte counter
  - Flow duration

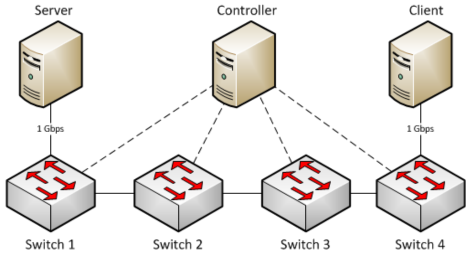


52

52

Measuring throughput

- Traffic Switch 1 -> Switch 2
- $Avg(Th) = \frac{ByteCounter}{FlowDuration}$
- $Th = \frac{\Delta BC}{\Delta FD}$



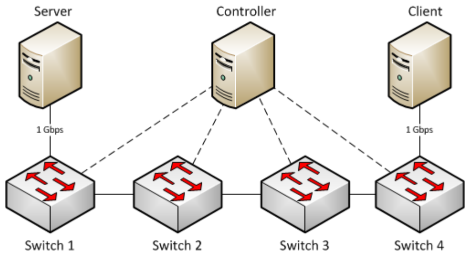
- $Th_{Sw1} \geq Th_{Sw2} \geq Th_{Sw3} \geq Th_{Sw4}$
  - Due to packet loss

TU Delft

53

53

Measuring packet loss



$$Avg(PackLoss) = PacketCounter_{Sw1} - PacketCounter_{Sw4}$$

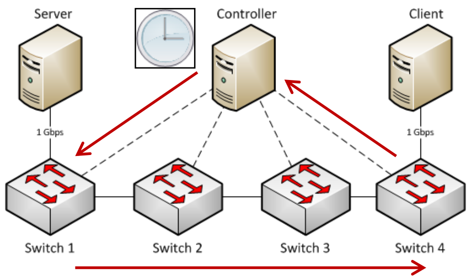
TU Delft

54

54

Measuring delay

- Insert time-stamped probe packets
- Compare time-stamps at arrival
- Deduct delay between controller and switches



TUDelft

55

55

Bandwidth guarantees with SDN

- OpenFlow supports 2 QoS mechanisms:
  - Queues
  - Meters

TUDelft

56

56

Queues

- (Zero or more) queues bound to specific output port
- Parameters:
  - Minimum rate
  - Maximum rate
  - Priority
- Provide bandwidth guarantees

TUDelft

57

57

Queues

- Cannot be configured with OpenFlow
- OpenFlow can only add flow entries placing packets in existing queues
- Configure queues directly with specific switch configuration protocol (e.g. OF-Config or Open vSwitch Database)

TUDelft

58

58

Meters

- Configured with OF protocol
- Attached to flow entries
  - 1 meter can be attached to multiple flow entries
  - Can only attach 1 meter per flow entry
- Not properly supported by Open vSwitch

TUDelft

59

59

Meters

- Measures packet rate of attached flow entries (via token bucket)
- Meter bands
  - Rate
  - Type
- Applies band with highest rate below the measured rate (if such a band exists)
- Types:
  - drop: drops packets
  - dscp remark: change DSCP field (differentiated services field)

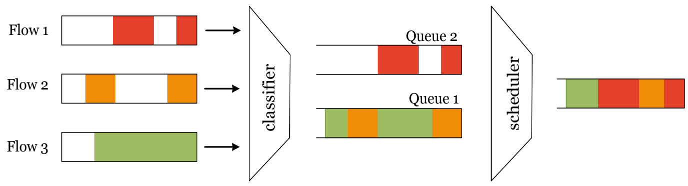
TUDelft

60

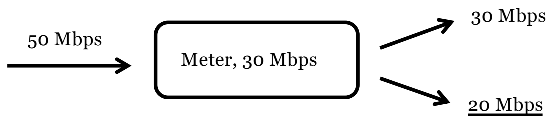
60

Illustration Queue & Meter

- Queue
  - Min_rate
  - Max_rate
  - Priority



- Meter
  - Drop
  - DSCP_remark



61

61

Possible design

- Admission control by the controller
- Traffic policing through *Meter table*
- Traffic prioritization at the switch through *Queue*
- Hard reservation

62

62

Admission control

- Objective: to prevent QoS traffic from competing with each other
- QoS traffic requests to reserve bandwidth
- Controller rejects the QoS traffic if no path can accommodate the requested rate
- Best-effort traffic bypasses the admission control

TUDelft

63

63

Traffic prioritization at the switch

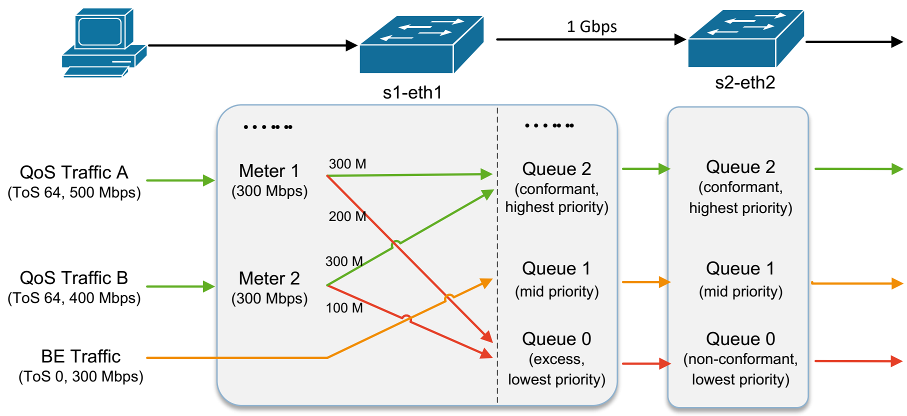
- In each switch port, 3 queues:
  - Queue 2: Highest priority for conforming QoS traffic
  - Queue 1: Best-effort traffic
  - Queue 0: Excess QoS traffic

TUDelft

64

64

Traffic aggregation



- QoS flow guaranteed rate: 300 Mbps
- Link capacity 1 Gbps

TU Delft

65

65

Hard reservation

- QoS request to controller via OFPT_PACKET_IN with appropriate DSCP bits
- Resources freed via OFPT_FLOW_REMOVED
- Both are standard OpenFlow messages: no extra signaling overhead

TU Delft

66

66