TDDD07 – Real-time Systems

Lecture 6: Distributed Systems III

Simin Nadjm-Tehrani

Real-time Systems Laboratory
Department of Computer and information Science

LINKÖPING UNIVERSITY

Overview: Next three lectures

From one CPU to networked CPUs:

- First, from one CPU to multiple CPUs
  - Allocating VMs on multiple CPUs: Cloud
- Next, fully distributed systems
  - fundamental issues with timing and order of events
- Next, hard real-time communication
  - Guaranteed message delivery within a deadline, bandwidth as a resource
- Finally: QoS guarantees instead of timing guarantees, focus on soft RT

Reading Material

- CAN: Davis et al. (2007) with a focus on section 3 or Ch. 4.5 in Carlsson et al
- QoS: El-Gendy et al. (2003)

Recall: Two approaches

- We will look at two well-known methods for bus scheduling
  - Time triggered (TTP)
  - Event triggered (CAN)

- Used extensively in aerospace and automotive applications respectively

Event-triggered (CAN) protocol

Response time analysis

- Scheduling analysis: Is *every* message delivered before its deadline?

Worst case response

According to [Tindell & Burns 94]:

Message response time =

J_i: Jitter (from event to placement in queue)+

w_i: Queuing time (response time of first bit)+

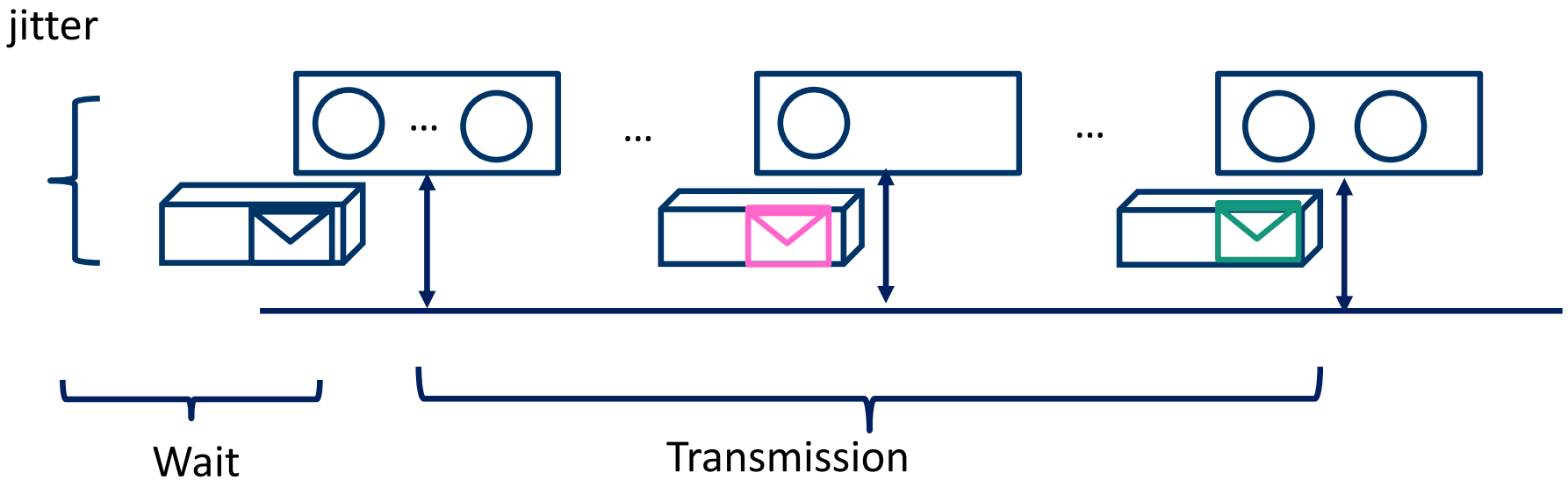
C_i: Transmission time for whole message

$$R_i = J_i + w_i + C_i - t_{bit}$$

$$w_i = t_{bit} + B_i + I_i$$

$B_i + I_i$: Blocking and Interference time (as RMS)

Jitter+wait+transmission



Interference and Blocking

- I_i: waiting due to higher priority messages, bounded if messages are sent periodically
- B_i: waiting due to lower priority messages, only one can start before i
- J_i: jitter, has to be assumed bounded (by assumptions on the node CPU scheduling policy!)

Solving recurrent equations

- Blocking is fixed: max C_j of all lower priority messages
- $w_i = B_i + \sum_{k \in hp(i)} \lceil (w_i + J_k + t_{bit}) / T_k \rceil C_k$
- $w^o_i = B_i$
- $w^{n+1}_i = B_i + \sum_{k \in hp(i)} \lceil (w_i^n + J_k + t_{bit}) / T_k \rceil C_k$
- After fixed point is reached: $B_i + w^{n+1}_i + C_i \leq D_i$?

Solving recurrent equations

- Blocking is fixed: max C_j of all lower priority messages
- $w_i = B_i + \sum_{k \in hp(i)} \lceil (w_i + J_k + t_{bit}) / T_k \rceil C_k$
- $w^0_i = B_i$
- $w^{n+1}_i = B_i + \sum_{k \in hp(i)} \lceil (w^n_i + J_k + t_{bit}) / T_k \rceil C_k$
- After fixed point is reached: $B_i + w^{n+1}_i + C_i \leq D_i$?

- From [Davis et al. 2007]:
  - To show how a w-term for each message was computed based on original method from 1994
  - Assume $J_i = 0$

Message	Priority	Period (Ti)	Deadline (Di)	TX time (Ci)
A	high	2.5 ms	2.5 ms	1 ms
B	med	3.5 ms	3.25 ms	1 ms
C	low	3.5 ms	3.25 ms	1 ms

Exercise

- Check that computed response times according to Tindell and Burns for the three messages meets every deadline!

The original analysis

- … was Optimistic!

- Constructed a case where (old) analysis shows schedulability but in fact deadlines can be missed!

[Davis, Burns, Bril, Lukkien 2007]

The correct analysis

- Takes account of the fact that different instances of the *same* message may affect the length of a busy period

and

- All instances should be shown to meet their deadlines!

[Reading: Sec. 3.1 & 3.2, Davis et al. 07]

Revised computation

- $R_m(q) = J_m + w_m(q) - qT_m + C_m$
- q=i, w(q) computes busy period for i^{th} instance of message m
- To know range of q, i.e. how many instances of message m are relevant, we need to find the longest busy period for message m, denoted t_m

Exercise

- $Q_m = \lceil (t_m + J_m) / T_m \rceil$

- Redo the same exercise with the correct variant of the busy period, where q stands for the q^{th} instance of the same message ($q \in \{0,..., Q_m - 1\}$)

$$w^{n+1}_m(q) =$$

$$B_m + q.C_m +$$

$$\sum_{k \in hp(m)} \lceil (w^n_m + J_k + t_{bit}) / T_k \rceil C_k$$

Example revisited

- Now with the new formula where busy period term is according to [Davis et al. 2007]

Message	Priority	Period (Ti)	Deadline (Di)	TX time (Ci)
A	high	2.5 ms	2.5 ms	1 ms
B	med	3.5 ms	3.25 ms	1 ms
C	low	3.5 ms	3.25 ms	1 ms

Solution

- To know how many instances of message m are relevant we need to find the longest busy period for m, denoted t_m. We focus on message C here.

Longest busy period for message C

- $t^0_C = C_C = 1$
- $t^1_C = \lceil t^0_C/T_C \rceil C_C + \lceil t^0_C/T_B \rceil C_B + \lceil t^0_C/T_A \rceil C_A = 1+1+1 = 3$
- $t^2_C = \lceil t^1_C/T_C \rceil C_C + \lceil t^1_C/T_B \rceil C_B + \lceil t^1_C/T_A \rceil C_A = 1+1+2 = 4$
- $t^3_C = \lceil t^2_C/T_C \rceil C_C + \lceil t^2_C/T_B \rceil C_B + \lceil t^2_C/T_A \rceil C_A = 2+2+2 = 6$
- $t^4_C = \lceil t^3_c/T_C \rceil C_C + \lceil t^3_C/T_B \rceil C_B + \lceil t^3_C/T_A \rceil C_A = 2+2+3 = 7$
- $t^5_C = \lceil t^4_C/T_C \rceil C_C + \lceil t^4_C/T_B \rceil C_B + \lceil t^4_C/T_A \rceil C_A = 2+2+3 = 7$
- $t_C = 7$

Using $Q_m = \lceil (t_m + J_m)/ T_m \rceil$

- means 2 instances of message C are relevant! $Q_C = 2$ and q: $0..Q_C-1$

Computing the queuing time

- $w^0_C(0) = B_C + 0.C_C = 0$
- $w^1_C(0) = \lceil (w^0_C(0) + t_{bit})/T_B \rceil C_B + \lceil (w^0_C(0) + t_{bit})/T_A \rceil C_A = 1+1 = 2$
- $w^2_C(0) = 1+1 = 2$

$\Rightarrow w_C(0) = 2$

$\Rightarrow R_C(0) = w_C(0) - qT_C + C_C = 3$

- $w^0_C(1) = w_C(0) + C_C = 2+1 = 3$
- $w^1_C(1) = C_C + \lceil (w^0_C(1) + t_{bit})/T_B \rceil C_B + \lceil (w^0_C(1) + t_{bit})/T_A \rceil C_A = 1+1+2 = 4$
- $w^2_C(1) = C_C + \lceil (w^1_C(1) + t_{bit})/T_B \rceil C_B + \lceil (w^1_C(1) + t_{bit})/T_A \rceil C_A = 1+2+2 = 5$
- $w^3_C(1) = C_C + \lceil (w^2_C(1) + t_{bit})/T_B \rceil C_B + \lceil (w^2_C(1) + t_{bit})/T_A \rceil C_A = 1+2+3 = 6$
- $w^4_C(1) = C_C + \lceil (w^3_C(1) + t_{bit})/T_B \rceil C_B + \lceil (w^3_C(1) + t_{bit})/T_A \rceil C_A = 1+2+3 = 6$

$\Rightarrow w_C(1) = 6$

$\Rightarrow R_C(1) = w_C(1) - qT_C + C_C = 3.5$

Maximum response time

$R_C = \max_{\{q:0..Q_C-1\}} R_C(q) = 3.5$

- Recall deadline for message C= 3.25

CAN error detection

- If a transmitted message is corrupted the Cyclic Redundancy Check (CRC) field will be wrong
- The first receiver that notes this sends oooooo
- Note that corruption at source and corruption in transit cannot be distinguished

- This works as long as a *node* is not erroneous – Babbling idiot!

Further developments

- New solutions to combine event-triggered and time-triggered messages have appeared:
  - Simulating CAN over TTP, or TT-CAN
  - FlexRay
  - RT/TT-Ethernet
- In the past ten years there are many standardisation efforts ongoing for industrial IoT to make the link layer more reliable, e.g. Time-Sensitive Networking (TSN) for 5G

DOI: 10.1109/COMST.2023.3275038

Overview: Next three lectures

From one CPU to networked CPUs:

- First, from one CPU to multiple CPUs
  - Allocating VMs on multiple CPUs: Cloud
- Next, fully distributed systems
  - fundamental issues with timing and order of events
- Next, hard real-time communication
  - Guaranteed message delivery within a deadline, bandwidth as a resource
- Finally: QoS guarantees instead of timing guarantees, focus on soft RT

QoS Guarantees

From messages to flows

When there is overload:

- Need to *allocate* available resources
  - To some applications/flows (which ones?)

- Applications may need to *adapt* as load mix and resource dynamics changes
  - *Same* flow can get different treatments at different nodes

2014: >50% of Internet traffic

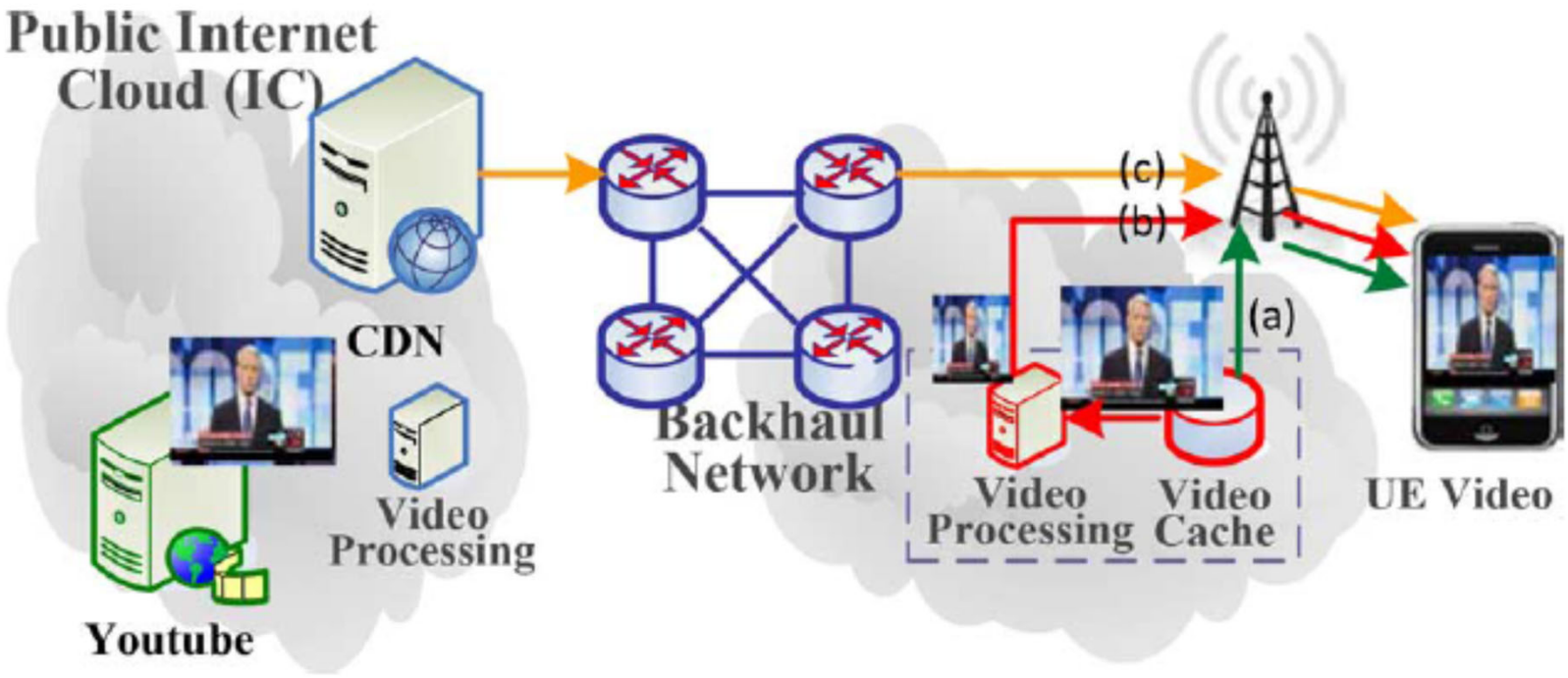


Image from Pedersen and Dey 2016
DOI:10.1109/TNET.2015.2410298

And it keeps growing…

QoS Overview

- Some basic notions: QoS parameters, requirement vs. provision
- We focus on allocation (not adaptation)
- Quality of service in networked (wired) applications
  - QoS mechanisms at nodes
  - Network-wide: Intserv, Diffserv

Which resources?

- Application nodes (edge nodes)
  - CPU
  - Memory (buffer space)
  - Power
- Links
  - Bandwidth
- Forwarding nodes: buffer space

What is Quality of service?

- Providing QoS: ability to provide resource assurance and service differentiation in a network
- Why is it important? See various actors' (Netflix, Verizon,…) stands (2014-2019)

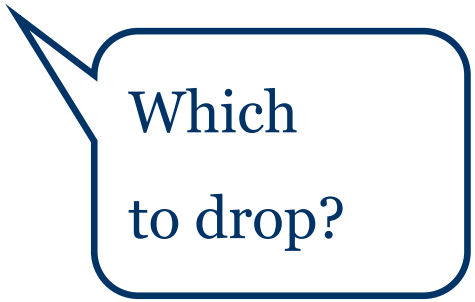
  https://www.technologyreview.com/2014/05/07/172935/talk-of-an-internet-fast-lane-is-already-hurting-some-startups/

  https://www.forbes.com/sites/stevensalzberg/2017/11/26/when-the-fcc-kills-net-neutrality-heres-what-your-internet-will-look-like/#6280dad4c687

  https://www.theverge.com/2019/10/4/20898779/fcc-net-neutrality-court-of-appeals-decision-ruling

Philosophies

- Service differentiation
  - When there are overloads some connections/packets/applications are preferred to others
- Fairness
  - All should get something



Opinions on both sides

The Hill, 2016-11-27:

https://thehill.com/policy/technology/307460-trump-picks-strike-fear-into-net-neutrality-backers

Image by Getty

FCC and ability to make decisions…

https://www.sdxcentral.com/articles/analysis/fcc-states-u-s-5g-global-leadership-tied-to-its-spectrum-authority/2023/04/

https://www.fcc.gov/about/leadership/anna-gomez

https://www.reuters.com/world/us/trump-taps-brendan-carr-chairman-federal-communications-commission-2024-11-18/

By Getty images

Adaptation

- Orthogonal to both:
  - Adaptive flows may adapt to make room for non-adaptive ones

- Back to basics…

How do we characterise QoS?

- Application-level requirements
  - Image quality (resolution/sharpness), viewing size, voice quality
- Enforcement (provision) level indicators
  - Bandwidth guarantee (measured as throughput)
  - delay
  - jitter
  - loss ratio
  - reliability (lack of erroneous messages and duplications)

QoS guarantees

- Need description of required/provided service
  - service commitment: e.g. % of dropped packets, average end-to-end delay
- In presence of a traffic model
  - Traffic profile: definition of the flow entitled to the service e.g. by arrival rates, burstiness, packet size,…

Application categories

- Elastic or inelastic
  - Mail vs. video conference
- Interactive or non-interactive
  - Voice communication vs. emergency warning at accidents
- Tolerant or non-tolerant
  - MPEG video-on-demand vs. automated control
- Adaptive or non-adaptive
  - Audio/video streaming vs. electronic trading
- Real-time or non-real-time
  - IP-telephony vs. A/V on demand (streaming)

QoS Overview

- Some basic notions: QoS parameters, requirement vs. provision

- We focus on allocation (not adaptation)

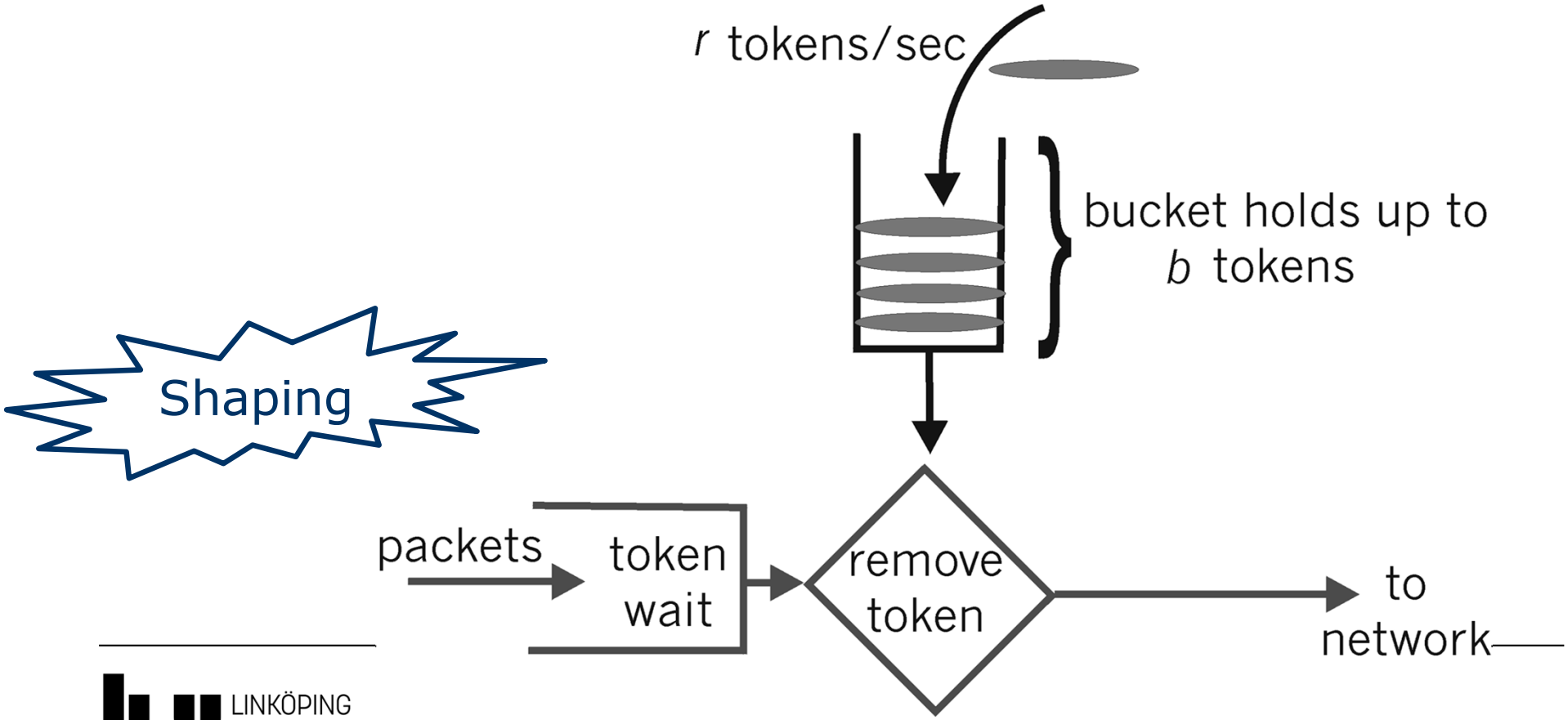
- Quality of service in networked (wired) applications
  - QoS mechanisms at nodes
  - Network-wide: Intserv, Diffserv

QoS mechanisms

- Admission control
  - To manage the limited resource in presence of oversubscriptions
  - Examples:
    - Policing (does the application ask for the same level of resources that was assumed as a traffic profile?)
    - Shaping (influencing the rate of packets fed into the network to adapt to current resource picture)
- Scheduling
- Buffer management

Leaky bucket

- Arrival profile can be described in terms of a pair (r, b) where r is the average bit rate, and b is an indication of burst size



Scheduling

Which packet should be forwarded at the network layer (to serve which QoS parameters)?

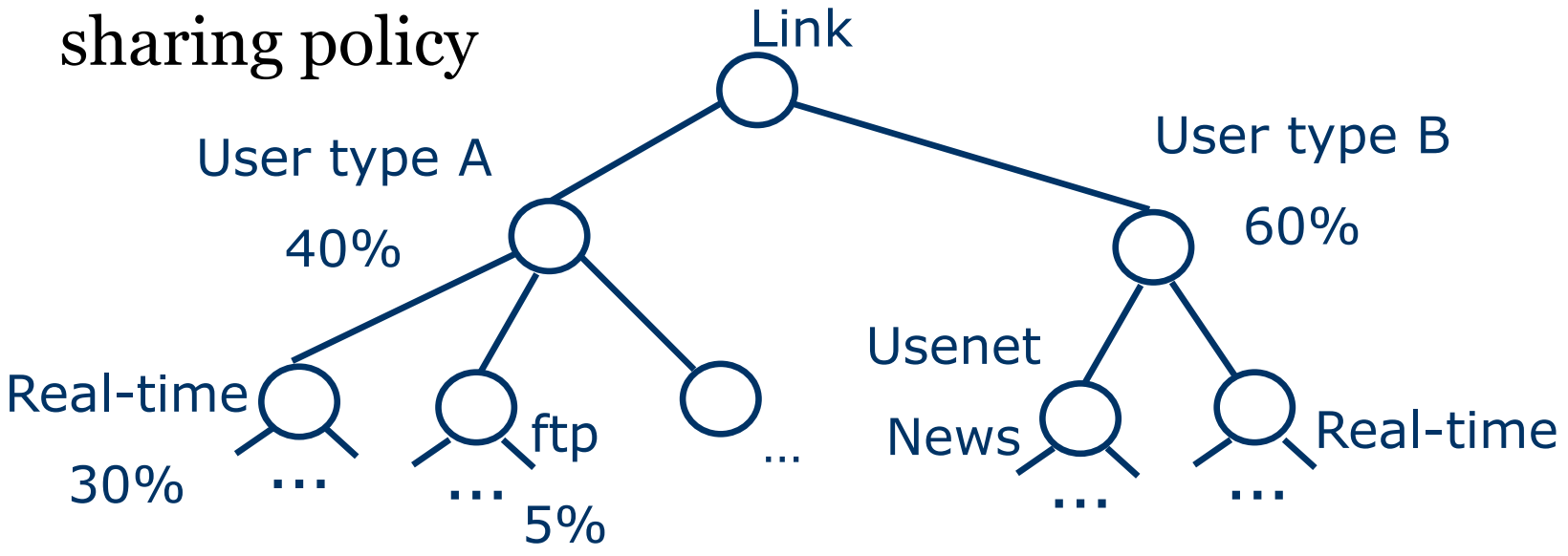
- No QoS: FIFO
- Fixed priority scheduling (similar to CAN when selecting from a queue)
  - With no guarantees on per packet delay, some can starve
- Weighted Fair Queuing (WFQ)
- Class based queuing

WFQ rough description

- Instead of allocating to all packets from one flow at a time, imagine an approximation to an ideally fair scheduler: one packet from each flow in a given time interval
- Allocate the outgoing bandwidth according to a weight for each flow
- For flows that are described as a leaky bucket, the max delay per packet is computable

Class-based link sharing

- Hierarchical allocation of the bandwidth according to traffic classes
- Each class allocated a max share under a given interval, and the excess shared according to some sharing policy



Buffer Management

- Scheduling is enough as long as buffers are infinite
  - In reality buffers (queues) get full during overloads
  - Shall we drop *all* the packets arriving *after* the overload starts?
- Buffer management is about determining which stored packets to drop in preference to incoming ones
  - Can adopt differentiated drop policies

QoS Overview

- Some basic notions: QoS parameters, requirement vs. provision
- We focus on allocation (not adaptation)
- Quality of service in networked (wired) applications
  - QoS mechanisms at nodes
  - Network-wide: Intserv, Diffserv

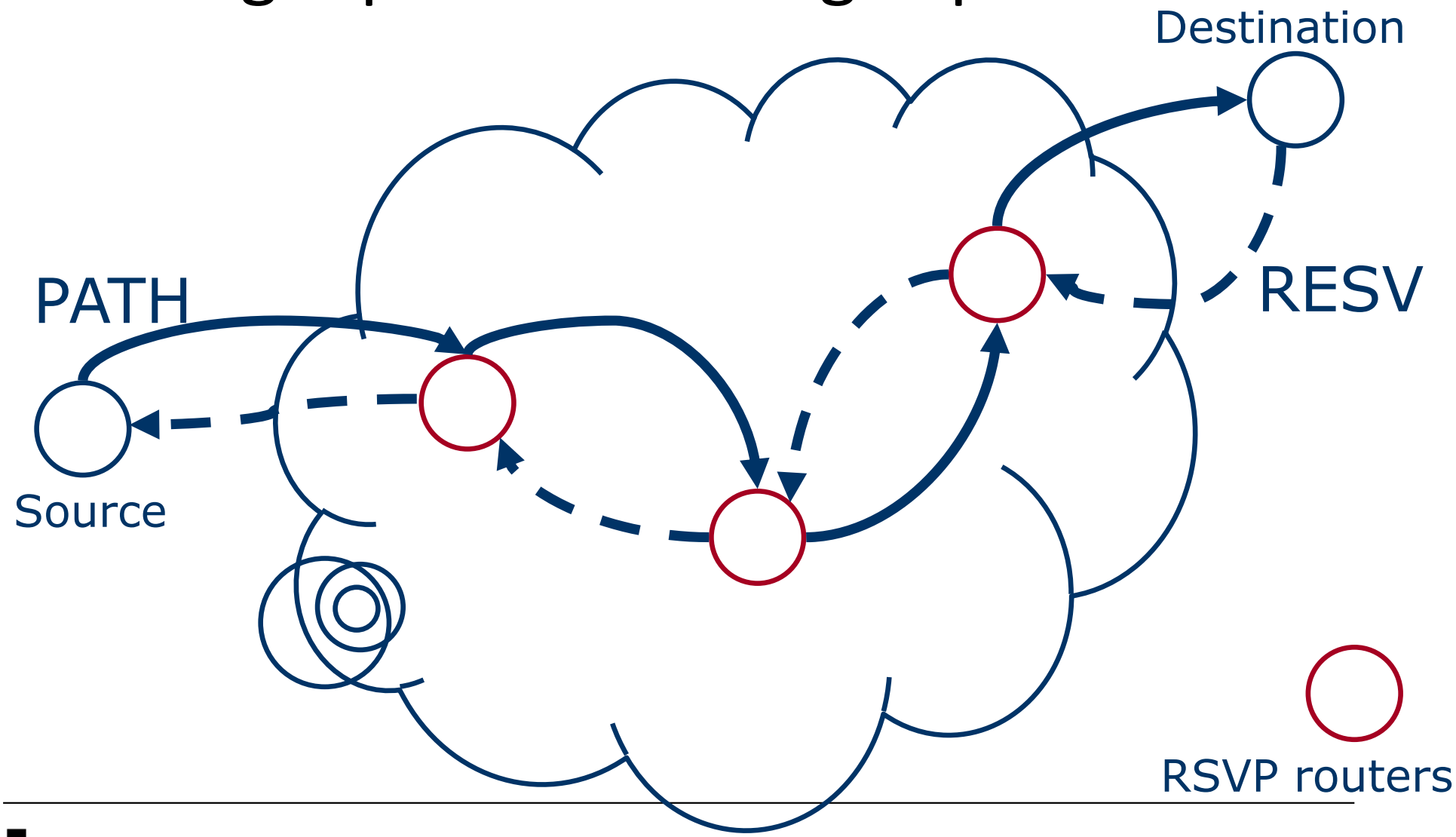
Across network nodes

- IP datagrams delivered with best effort
- **IntServ** was defined to deliver IP packets with differentiated treatment across multiple routers (1994)
- Introduced 3 service classes:
  - **BE:** Best effort
  - **CL**: Controlled Load (acceptable service when no overload)
  - **GS**: Guaranteed Service (strict bounds on e-to-e delay)

Intserv

- Each router keeps a “soft state” for each flow (a session) currently passing through it
  - GS: the leaky-bucket-based requirements from a flow induce a max local delay in each router
- The soft state is created with a reservation scheme RSVP, and refreshed while the session is in progress

Sending Tspec & receiving Rspec



Intserv QoS specifications

- T-spec (traffic specification)
  - A token bucket specification
    - token rate - r
    - bucket size - b
    - peak rate - p
    - maximum packet size - M
    - minimum policed unit - m
- R-spec (reservation specification)
  - Service Rate – R
    - The bandwidth requirement
  - Slack Term – S
    - The delay requirement

Not deployed successfully!

- IntServ met resistance for several reasons, including:
  - Not all routers RSVP enabled
  - Set up time can be proportionately long compared to session time
  - Interactive sessions need to set up a path at both ends
  - Dynamic and major changes in traffic pattern

Diffserv (1998)

- Based on resource provisioning (for a given SLA) as opposed to reservation
- Applied to traffic aggregates as opposed to single flows
- Forwarding treatment as opposed to end-to-end guarantees
- Edge routers labelling packets/flows in forwarding to next *domain*, and accepting only in-profile packets when accepting from other domains

Diffserv Service classes

Marked with two bits:

- (P) Premium class: intended for preferential treatment to which policing is applied with a small bucket size
- (A) Assured class: pass through policing with a bucket size equal to the given burst
- Packets with A-bit compete with best effort packets when buffers get full

Scalability of Diffserv

- Admission control is now at edge nodes not every path on a route
- No set-up time and per-flow state in each router
- At the cost of *no end-to-end guarantees*

- *Current research (2020) https://ieeexplore.ieee.org/document/9110430*

Differentiation revisited

- Differentiated connectivity made possible with 5G SA (see page 13)
  - Four classes identified in Figure 10
  - Automotive, Gaming, Video

https://www.ericsson.com/49ed78/assets/local/reports-papers/mobility-report/documents/2024/ericsson-mobility-report-june-2024.pdf

Questions?

www.ida.liu.se/~TDDD07

LINKÖPING UNIVERSITY