Rate Control and Medium priority traffic behaviour.

Lessons Learned from Java Simulations

Stein Gjessing
Simula Research Laboratory
and University of Oslo

Contents

- Part 1
  - Granularity of rate control
    - Fine grained vs bursty rate control
      1. Simple version
      2. Choke point version
- Part 2
  - Performance of medium priority traffic

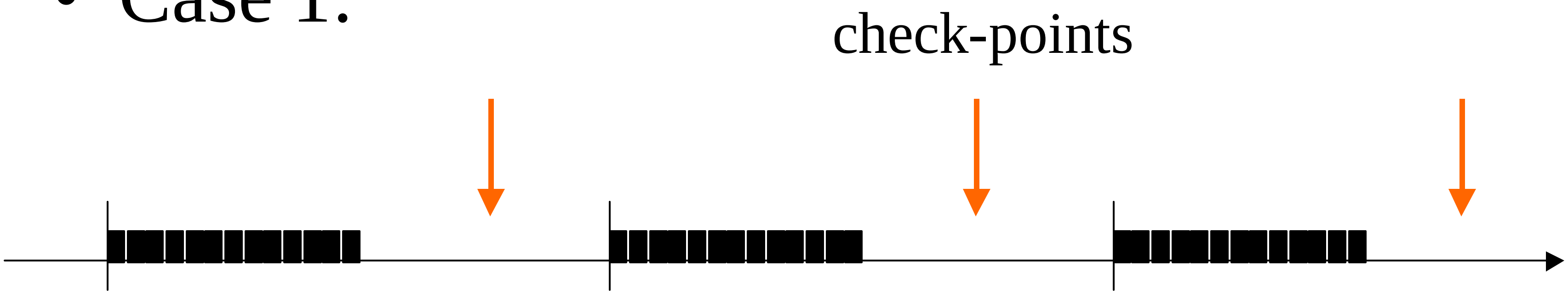
Bursty rate control

- Gandalf (and Darwin ?) today:
  - OK to send if (**myUsage < allowUsage**)
  - myUsage is increased when packet sent
- Result:

e.g. 100 microsec intervals:

This is bad because:

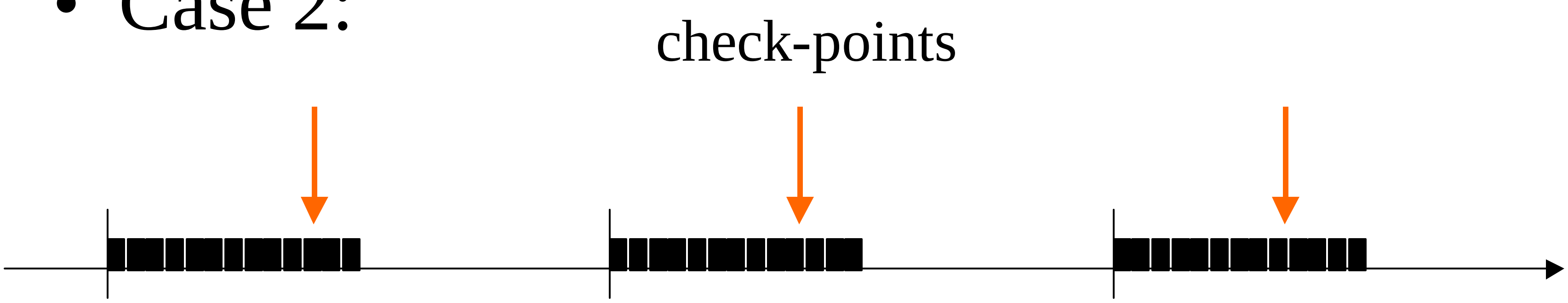
- Downstream node receives the packets and checks if it is congested with the same 100 us intervals:
- Case 1:



Then the downstream node will find that it **is not** congested

This is bad because:

- Downstream node receives the packets and checks if it is congested with the same 100 us intervals:
- Case 2:



Then the downstream node will find that it **is** congested

This is bad because:

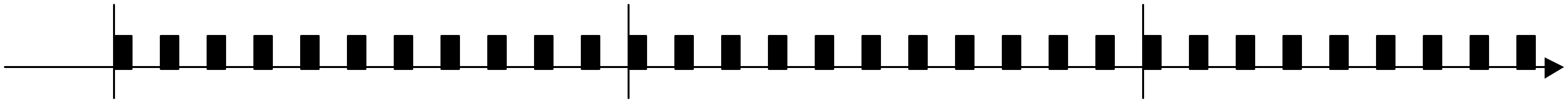
- Even if the downstream node receives the packets and is congested almost all the time, it might seem not to be:
- Case 3:

check-points

Then the downstream node will find that it **is not** congested

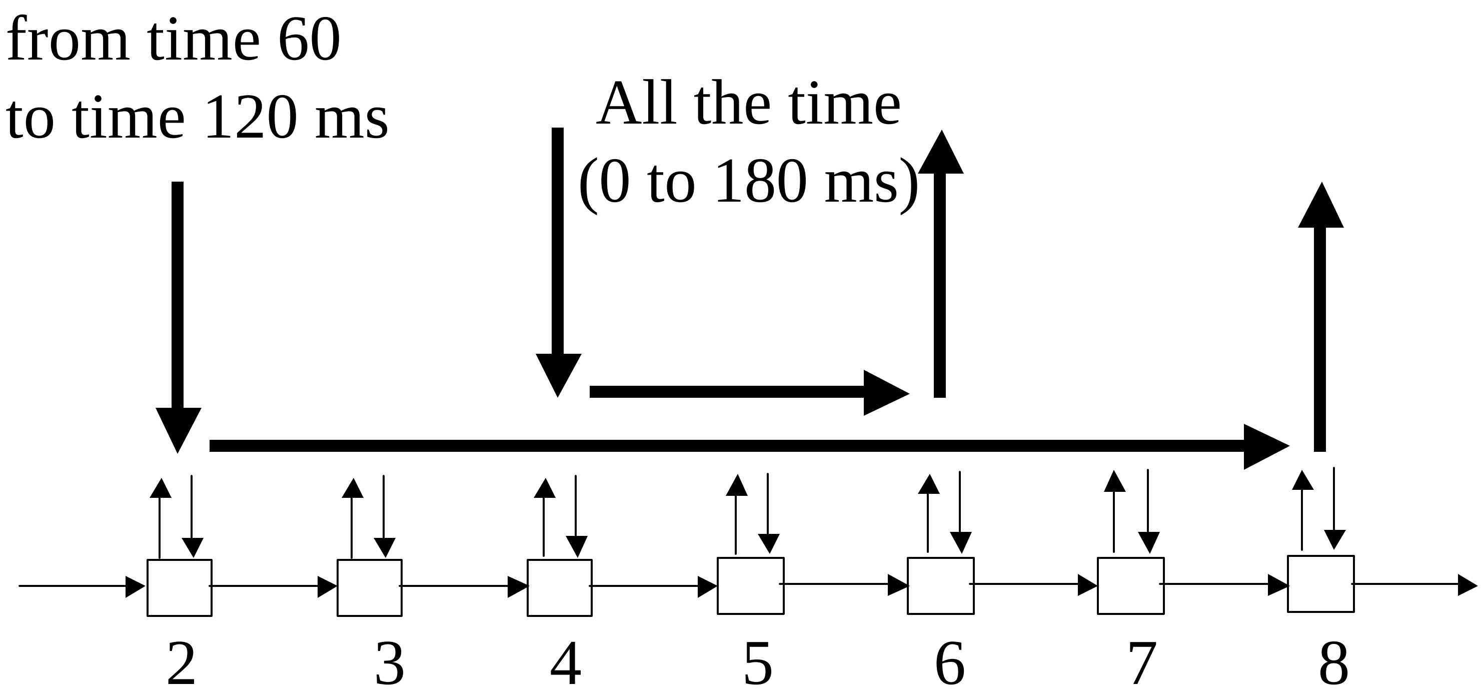
Fine grained rate control

- Send at all packets at **allowUsage / maxRate** rate
- Implemented in my Java simulator
- Result:



Thesis: This is much better for everyone ”downstream”

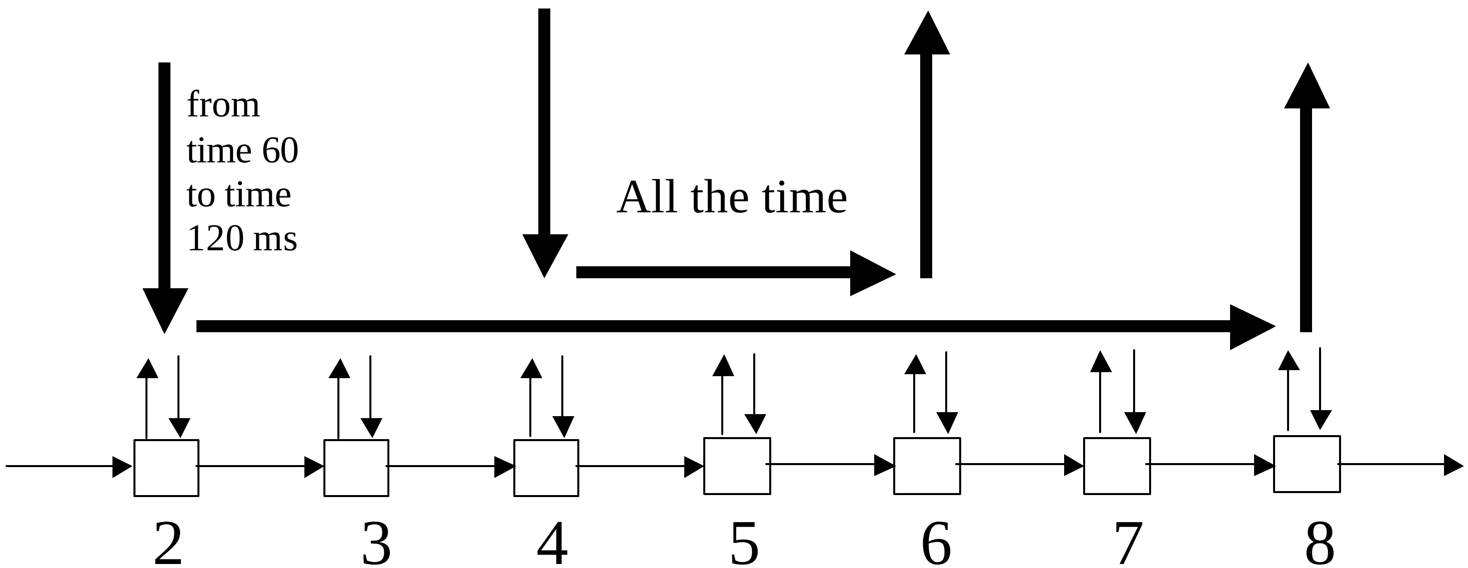
Experiment – new upstream flow



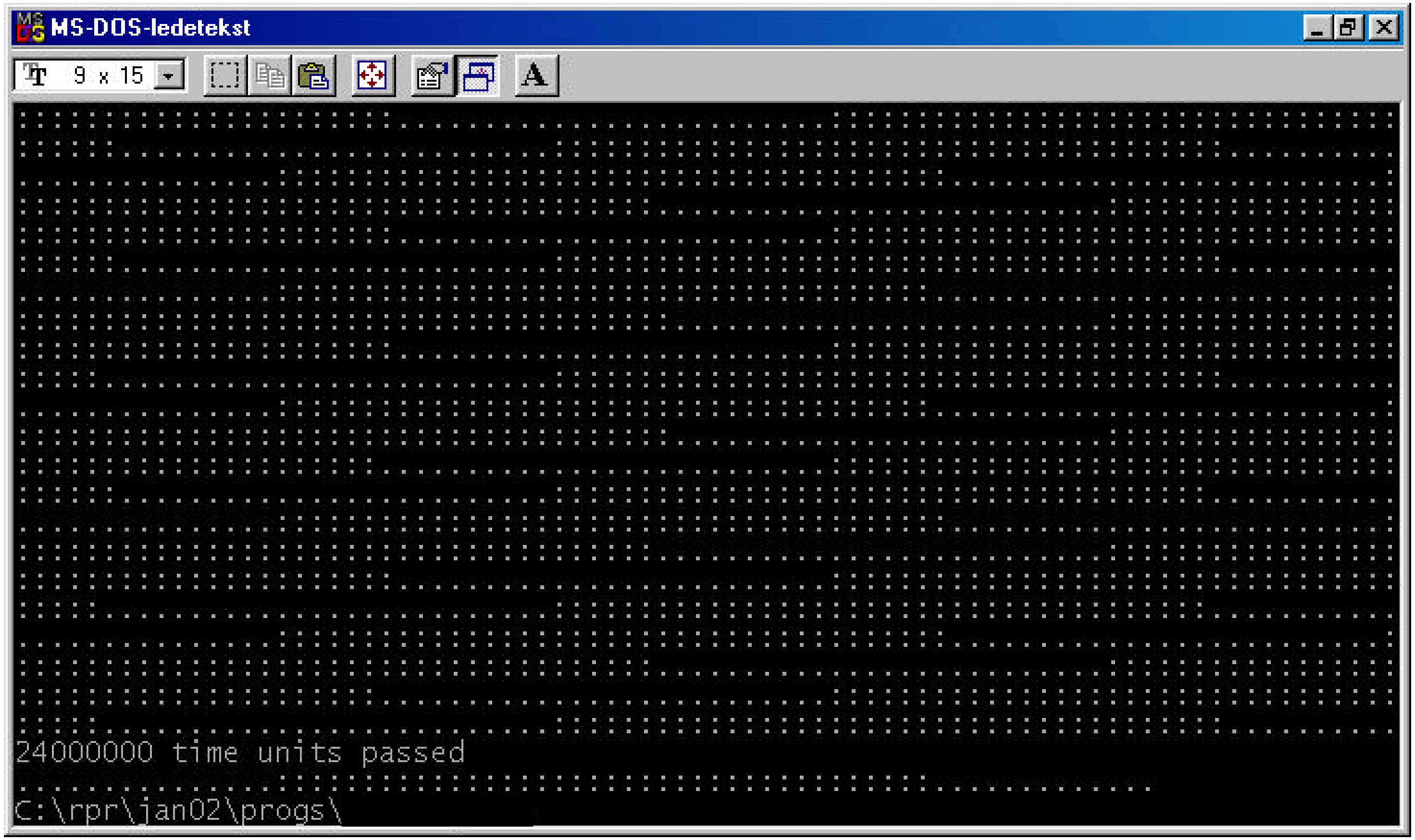
When the flow from 2 starts, it will congest station 4.
Station 4 sends upstream congestion notification to station 2.

Simulation results

- Sequence of packets passing station 5
  - 500 bytes packets
- Number of packets received per unit time (100 microsec) at stations 6 and 8

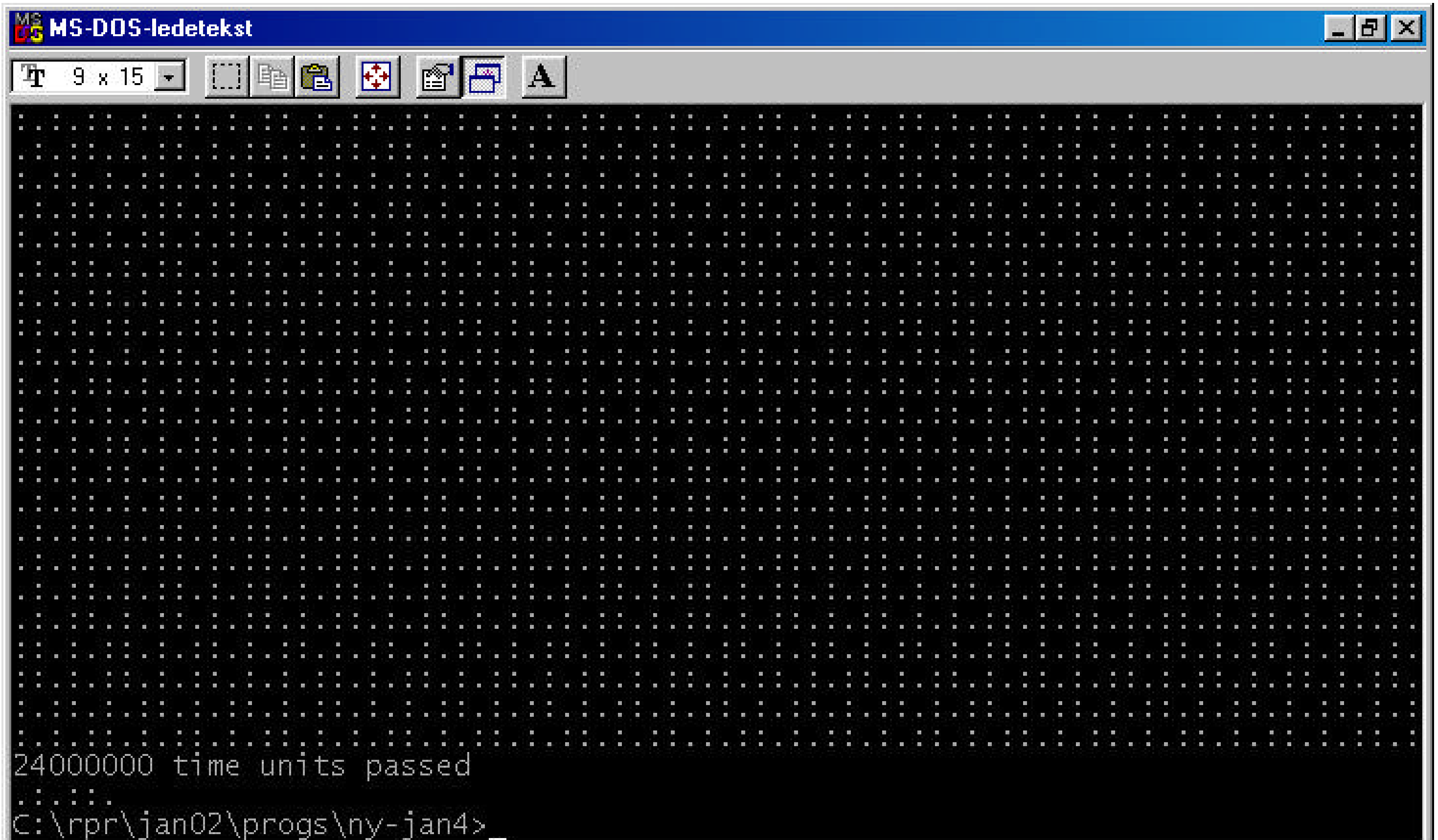


Packets passing station 5 – bursty rate control



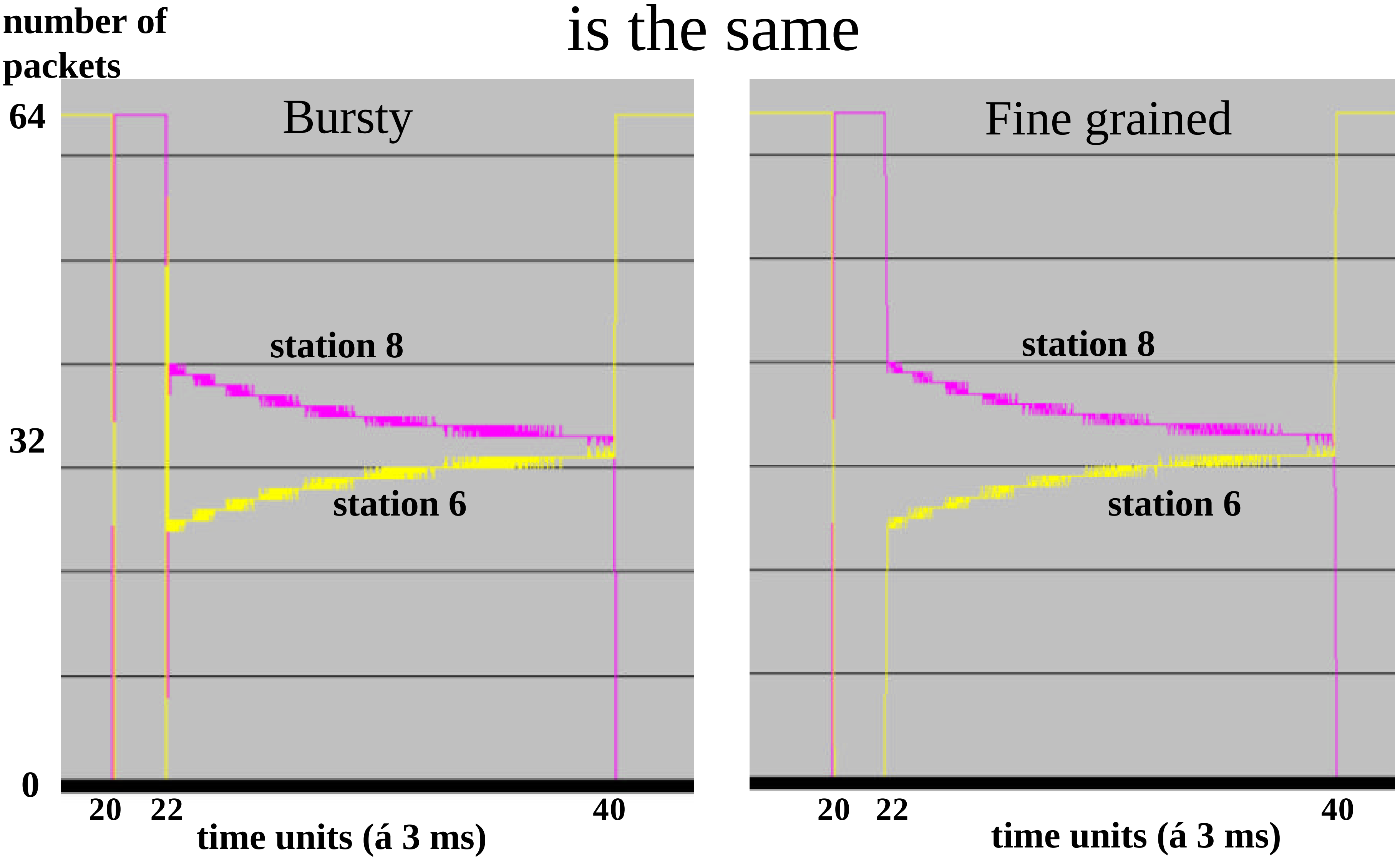
: packet from station 2 . packet from station 4

Packets passing station 5 – with fine grained rate control

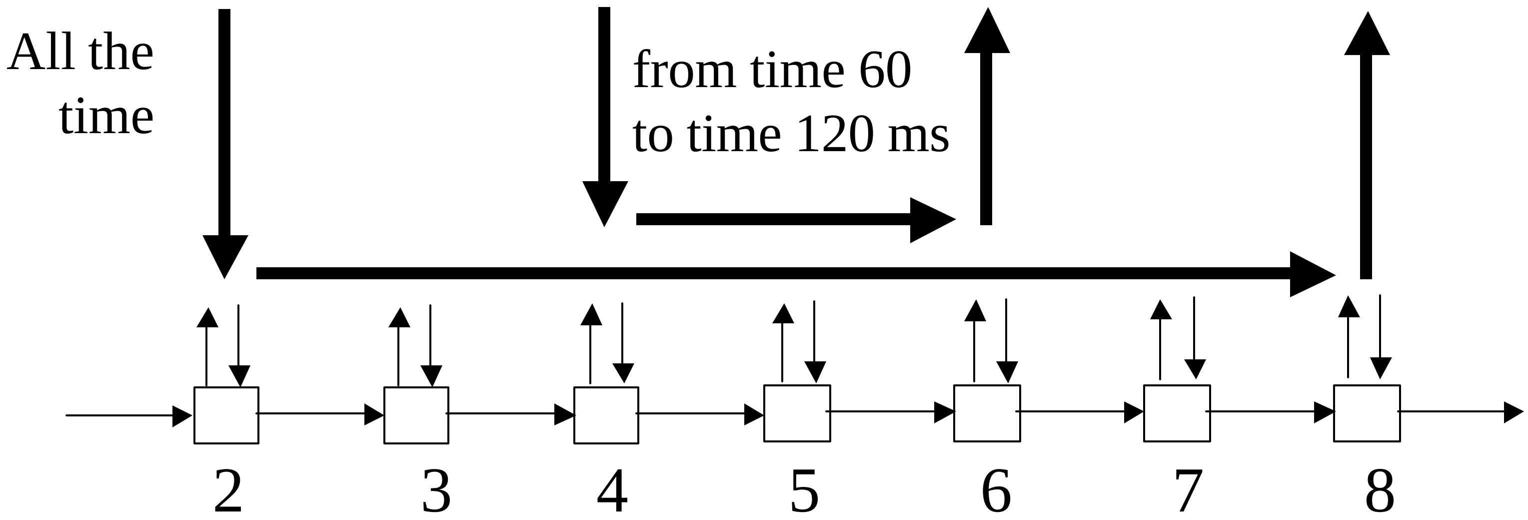


: packet from station 2 . packet from station 4

Number of packets received per 100 microsec. is the same



New experiment -
new downstream flow

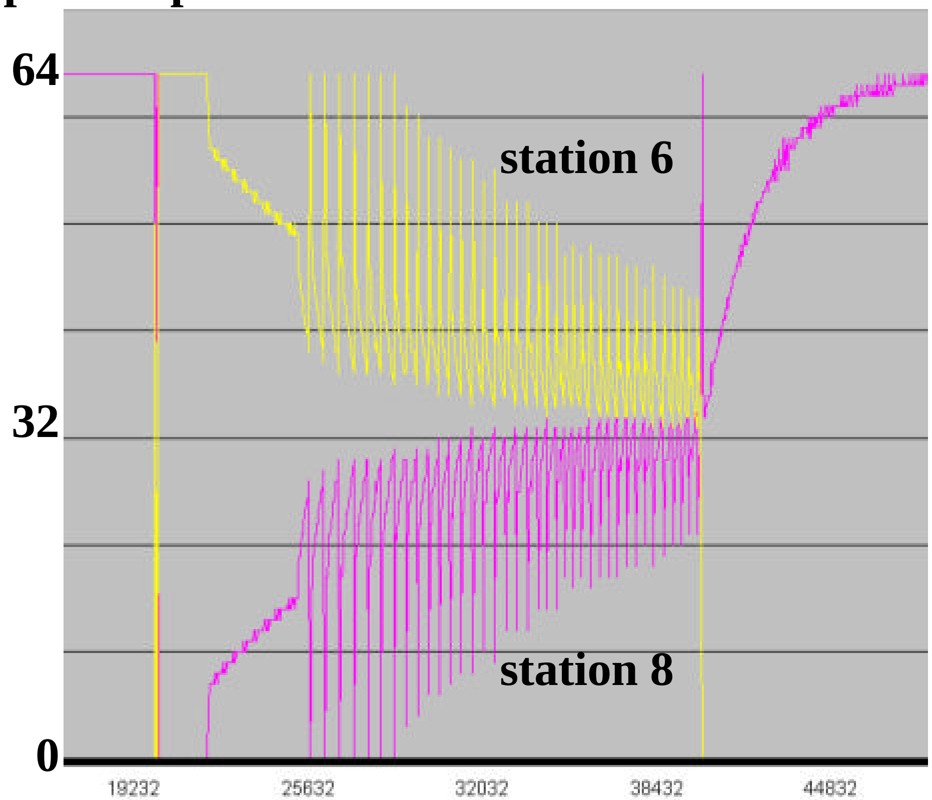


When the flow from 4 starts, station 4 will be congested.
Station 4 sends upstream congestion notification to station 2.

- Same pattern as before for packets passing by staion 5 (not shown)

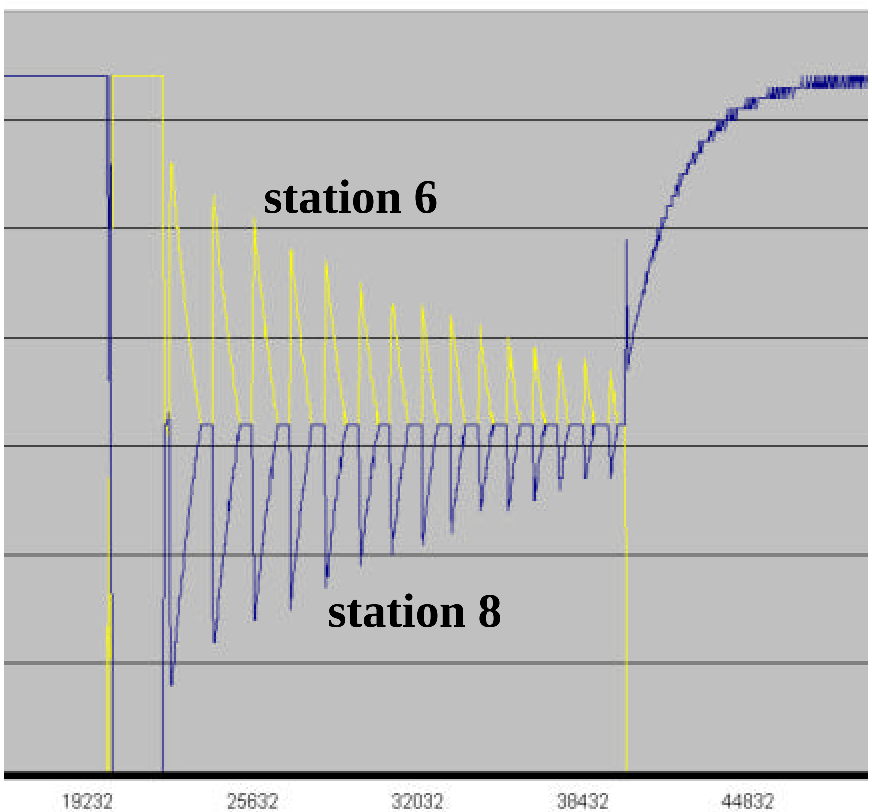
New downstream flow

number of
packets per 100 microsec



time units (3 us)

Bursty rate control



time units (3 us)

Fine grained rate control

Fine grained rate control makes congestion discovery more precise!

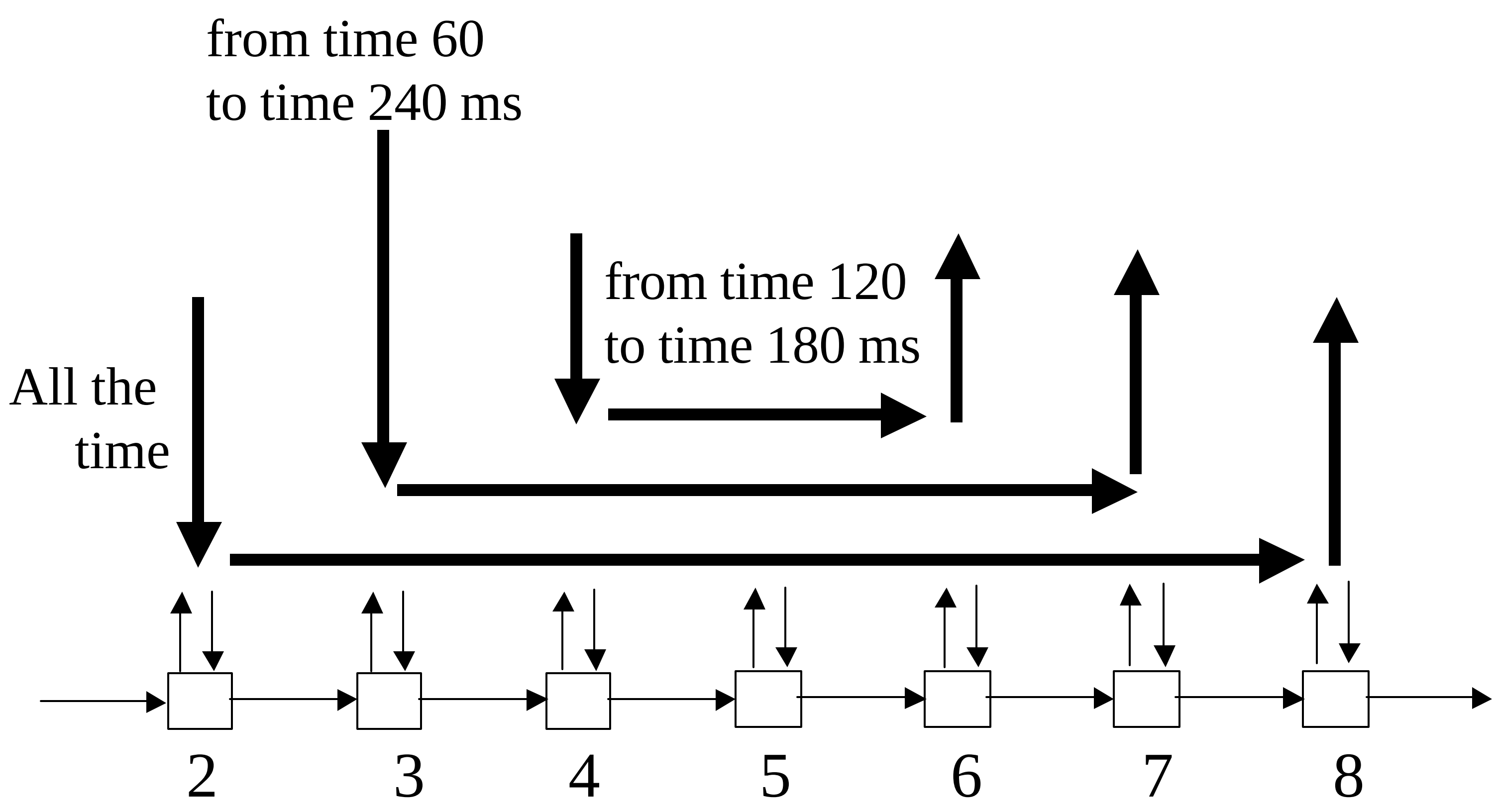
Choke points and fine grained rate control

- Java simulator with VOQ's, up to N choke points and fine grained rate control
  - (The full ring is N stations)
- The allowed rate at each choke point i, is
  allowUsage[i] / maxRate (or maxRate[i])

New experiment

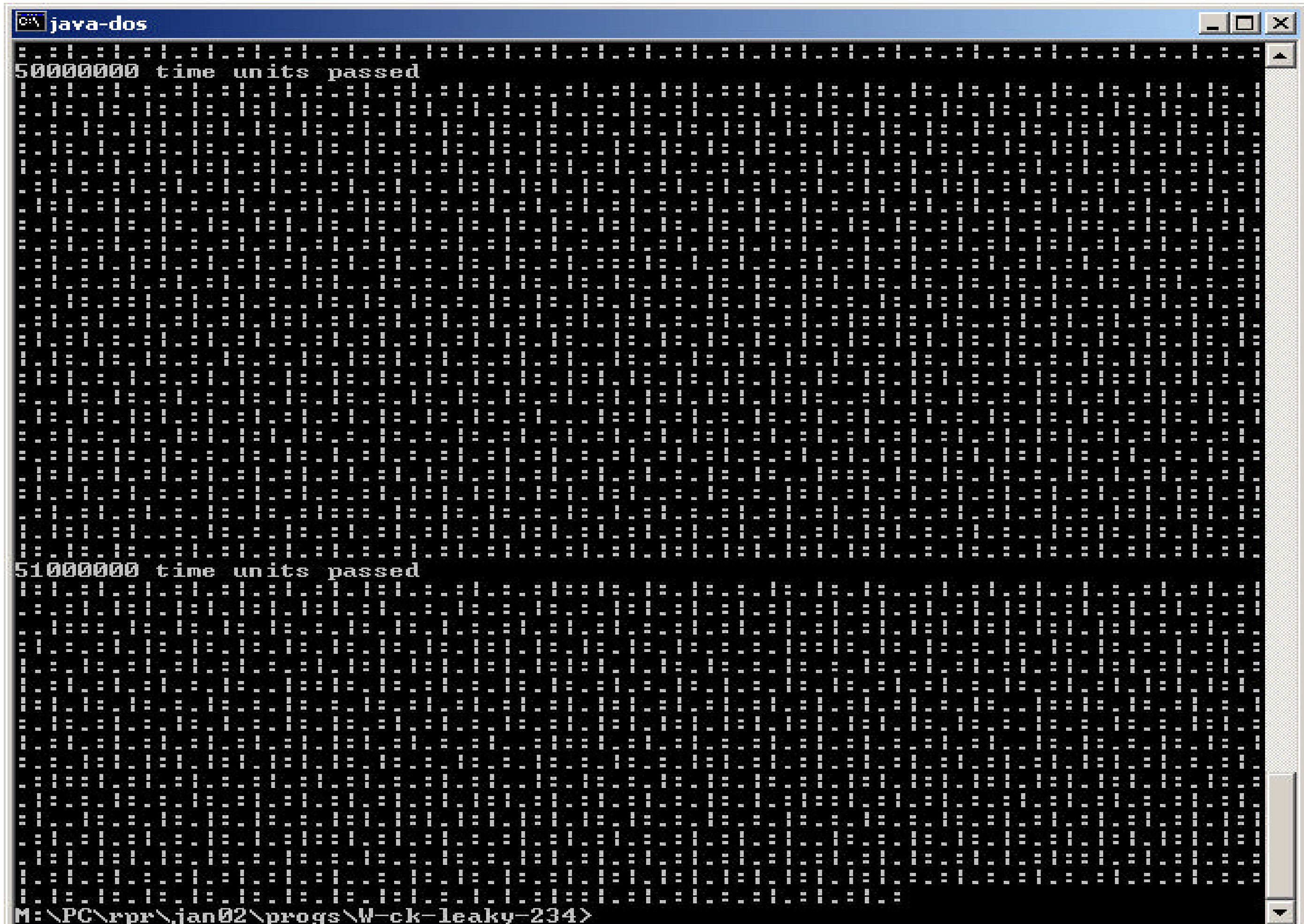
- Three flows (500 byte packets)
  - 2 to 8 sends all the time
  - 3 to 7 sends from time 60 to 240 ms.
  - 4 to 6 sends from time 120 to 180 ms.
- Station 2 will experience a change of choke point and its rate
- However in this example VOQ's are not used (only choke points)

The new experiment – three flows



Packet passings station 5

Fine grained rate control does not give bursts



. from 2

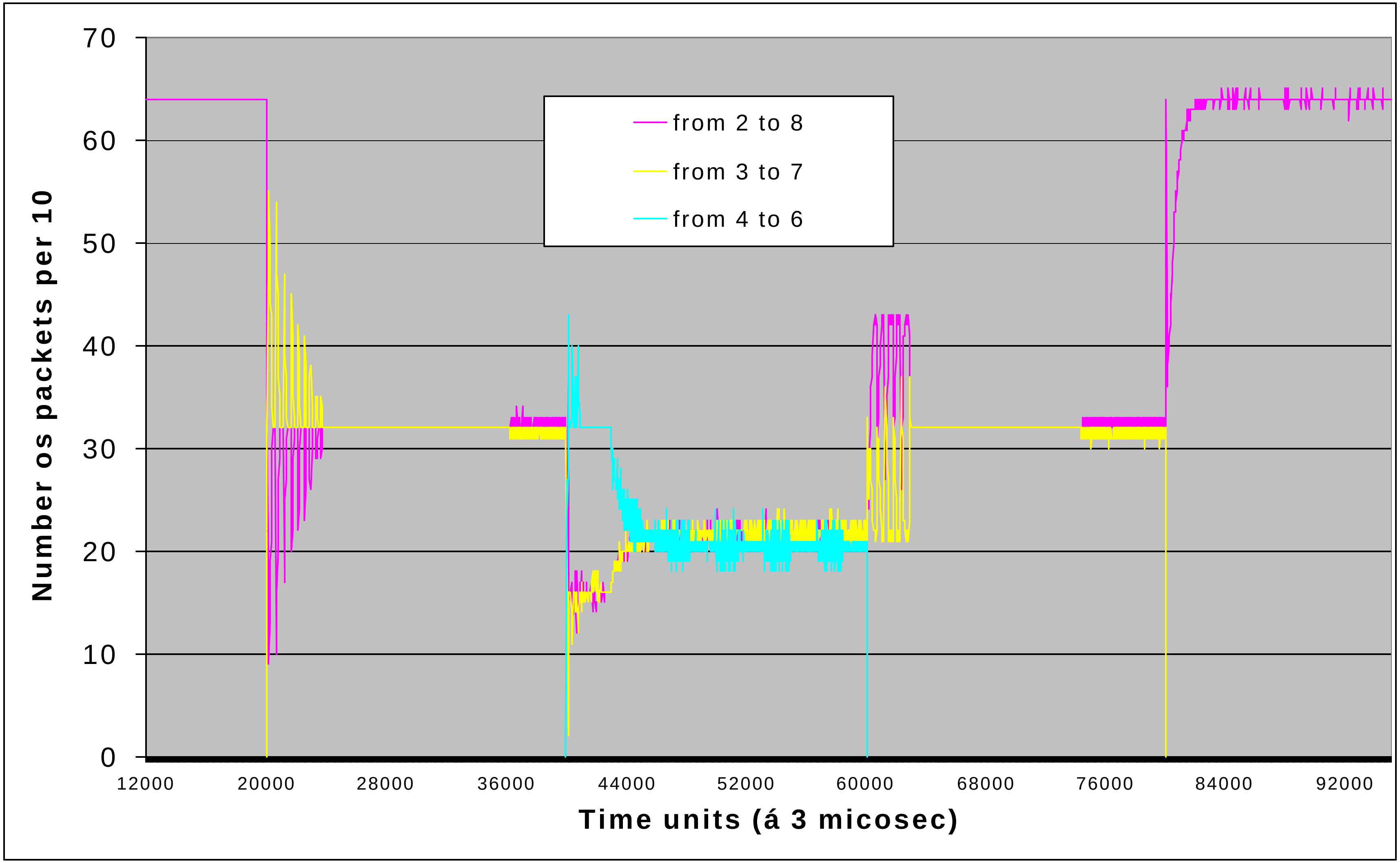
: from 3

| from 4

2.5 Gbit/s

One time unit is three ns.

Packets received



Part 2
Performance of
Medium priority traffic

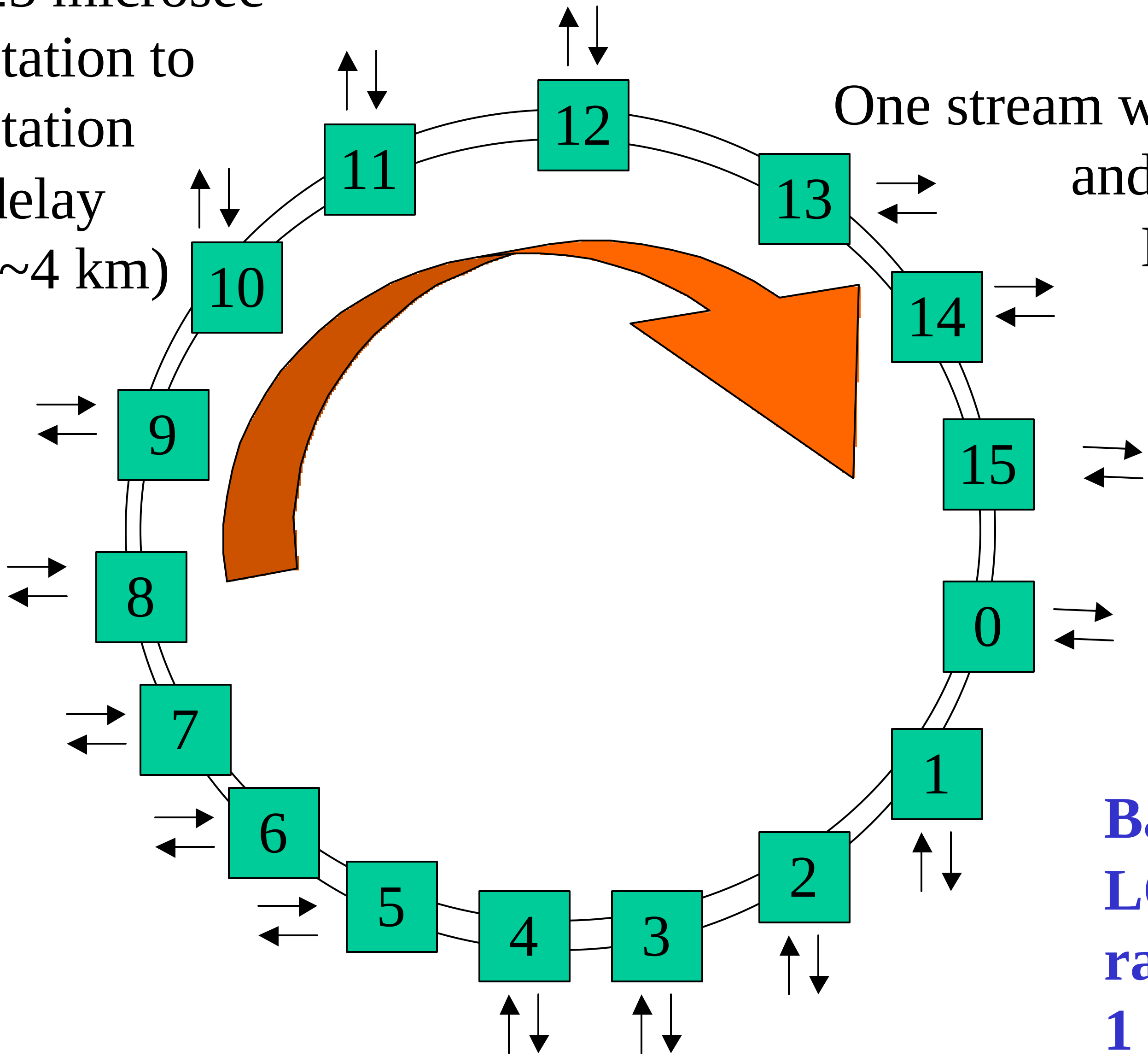
Latency of
medium priority traffic vs
high priority traffic vs
low priority traffic

Streaming TDM traffic from 8 to 15

25 microsec station to station delay (~4 km)

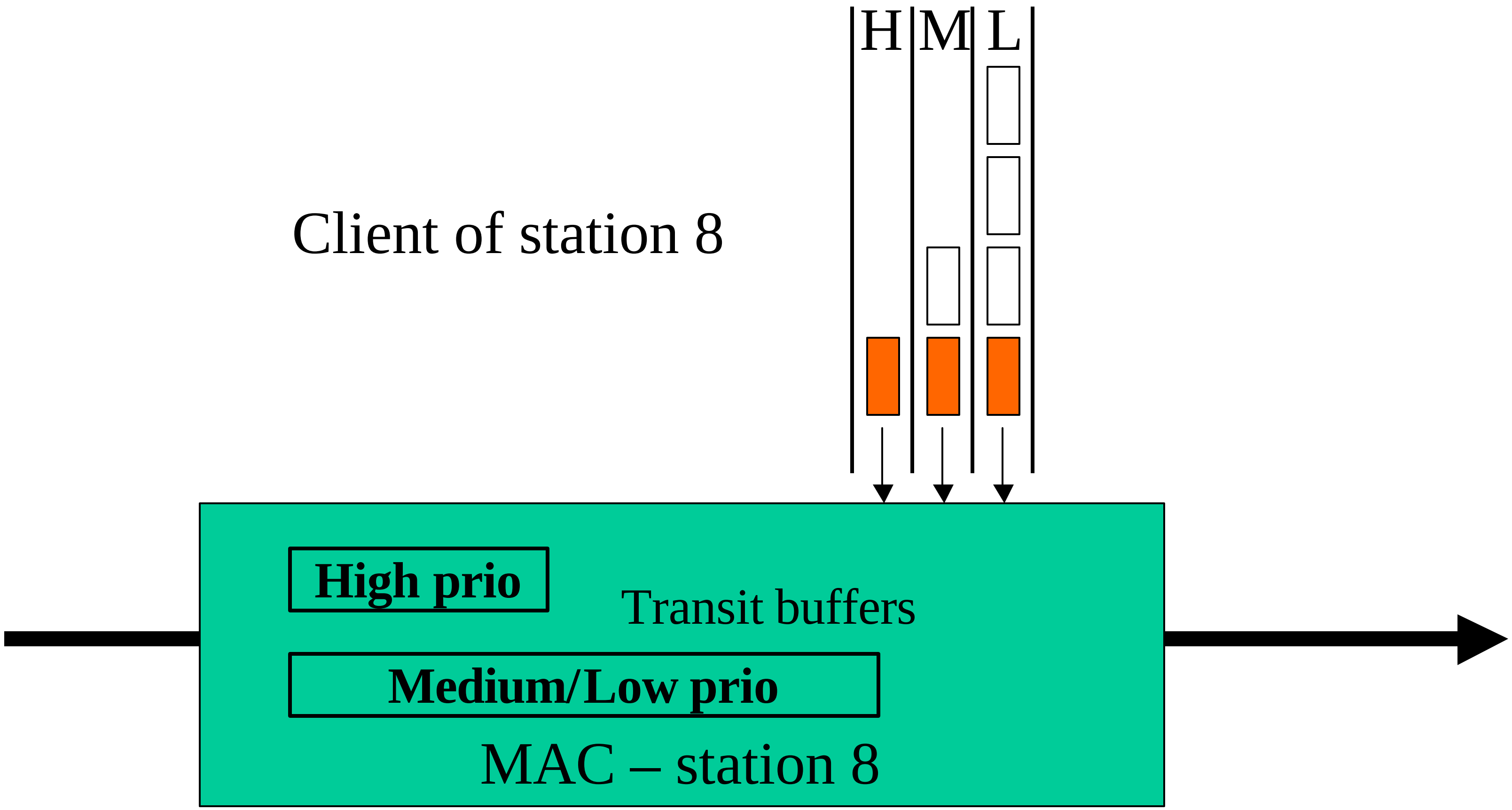
One stream with High priority and one stream with Medium priority

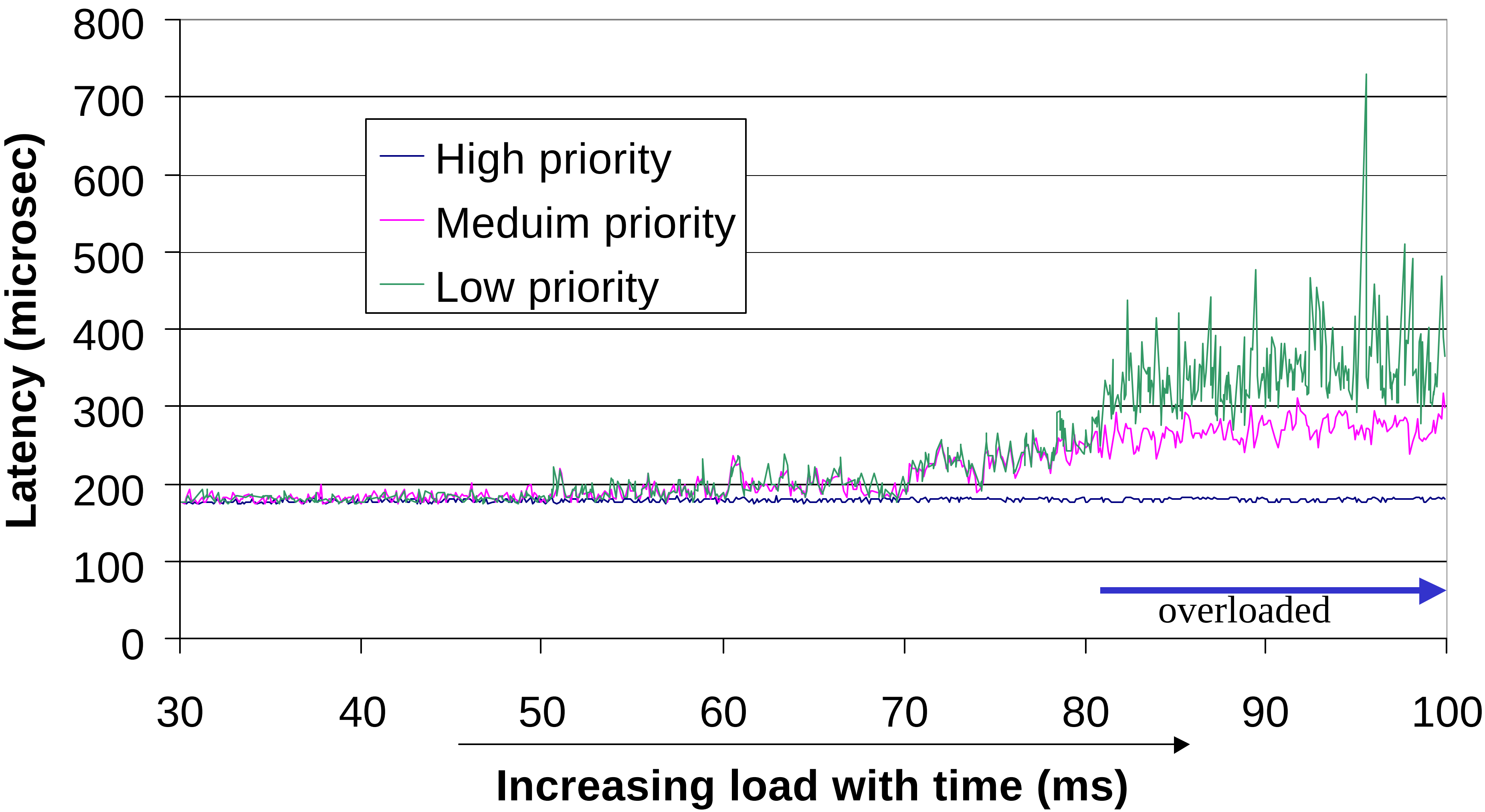
Also measuring Low priority latency from station 8 to 15



Background traffic: LOW PRIORITY random (all to all) 1 G byte / sec links

Starts to measure latency when first in queue





Low priority packet load increases every 10 ms

Latency of medium priority packets

- Congestion threshold in the low/medium transit buffers are 25 000 bytes in my simulations
- At 1Gbyte/sec this is 25 microsec delay
- From station 8 to station 15 there are 6 transit buffers: 6 * 25 microsec = **150** microsec
- In my experiment (previous slide), with overloaded ring, medium priority traffic have ~270 microsec mean (which is ~**100** microsec more than in an empty ring)

Conclusion 1 (rate control)

- Fine grained rate control is easy to implement
  - Also together with choke points and VOQs
  - Rate beyond choke point is **allowedUsage[i] / maxRate**
    where i is the choke point
- Fine grained rate control makes congestion detection more precise and hence improves fairness
- Thesis: Fine grained rate control smooth traffic and decreases overall buffer needs.

Conclusion 2 (medium priority traffic)

- Medium priority traffic behaves as expected:
  - Medium priority traffic enters the ring immediately
  - The maximum delay is the number of stations times the delay in each transit buffer
    - High priority traffic may delay medium priority traffic even more