Scalable Scheduling Support for Loss and Delay Constrained Media Streams

Richard West, Karsten Schwan &
Christian Poellabauer

Georgia Institute of Technology

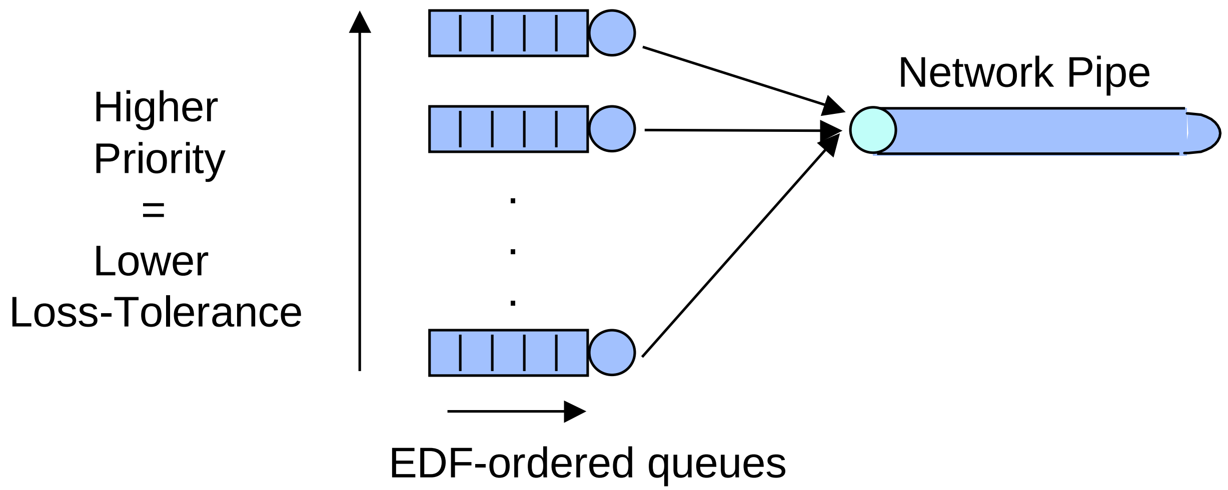
Introduction

- Real-Time media servers need to support 100s (even 1000s) of clients with individual RT (QoS) constraints.
- Need fast/efficient scheduling on such servers.
- We describe Dynamic Window-Constrained Scheduling (DWCS):
  - DWCS limits the number of late packets over finite windows of arrivals requiring service.
  - Focus on a scalable implementation of DWCS.
  - Approximating DWCS trades execution speed for service quality.

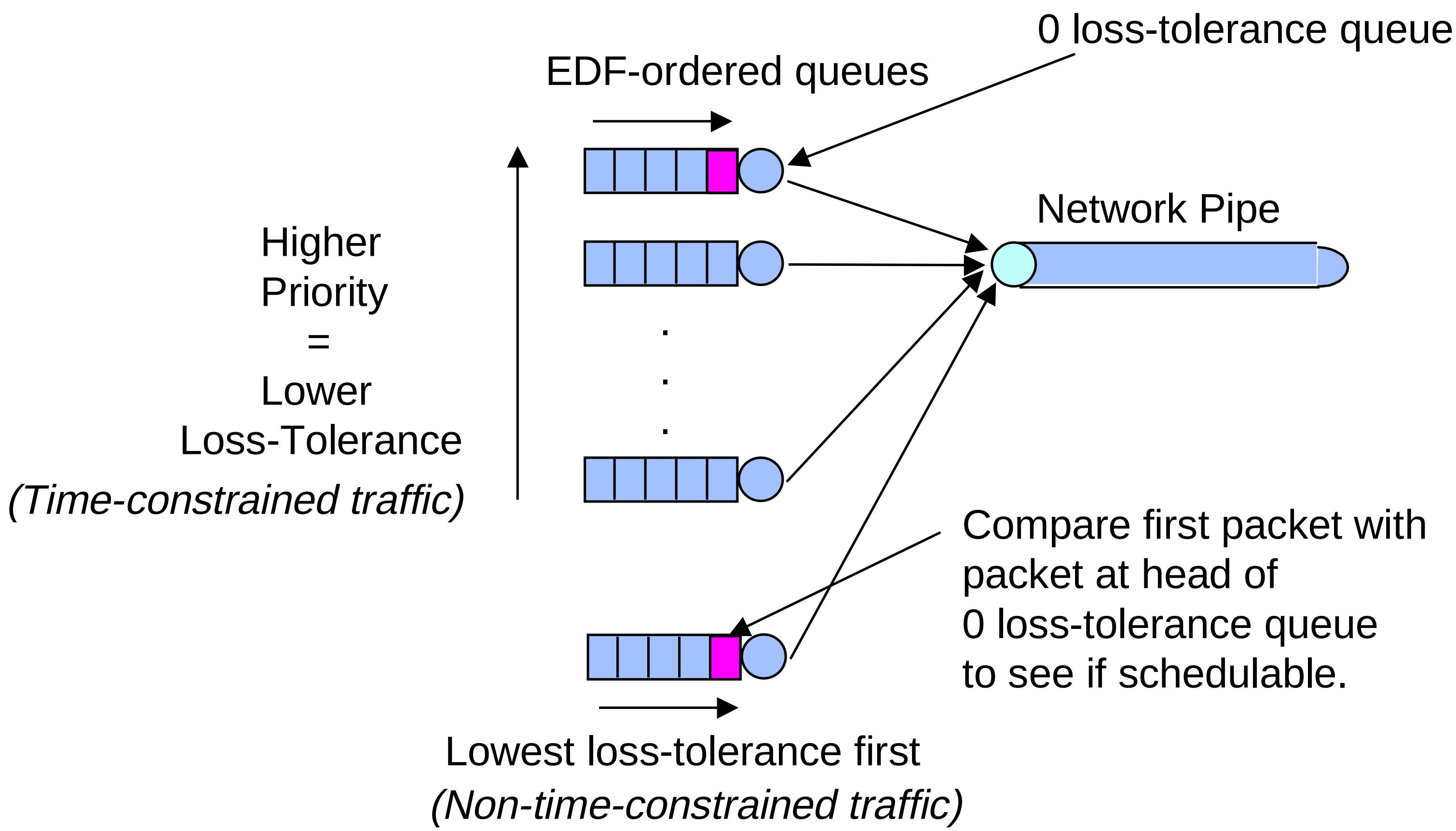
DWCS Packet Scheduling

- Two attributes per packet:
  - Deadline (max inter-packet gap).
  - Loss-tolerance, **x/y**.
    - **x** late/lost packets every **y** arrivals for service from same stream.
- At any time, all packets in the same stream:
  - Have the same current loss-tolerance.
  - Have deadlines offset by a fixed amount from predecessors.

DWCS - Conceptual View



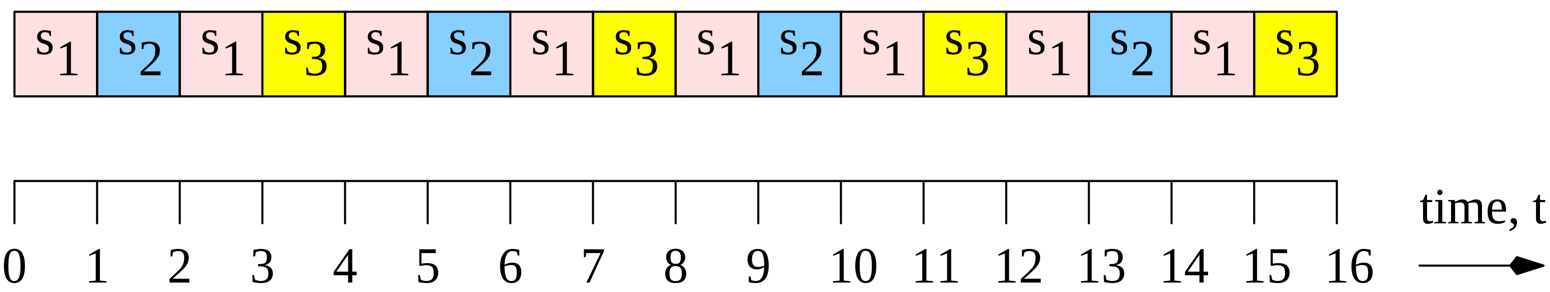
Heterogeneous Scheduling



Pairwise Packet Ordering Table

Precedence amongst pairs of packets
• Lowest loss-tolerance first
• Same non-zero loss-tolerance, order EDF
• Same non-zero loss-tolerance & deadlines, order lowest loss-numerator first
• Zero loss-tolerance and denominators, order EDF
• Zero loss-tolerance, order highest loss-denominator first
• All other cases: first-come-first-serve

Example: L1=1/2, L2=3/4, L3=6/8 D=1, Service Time (C)=1

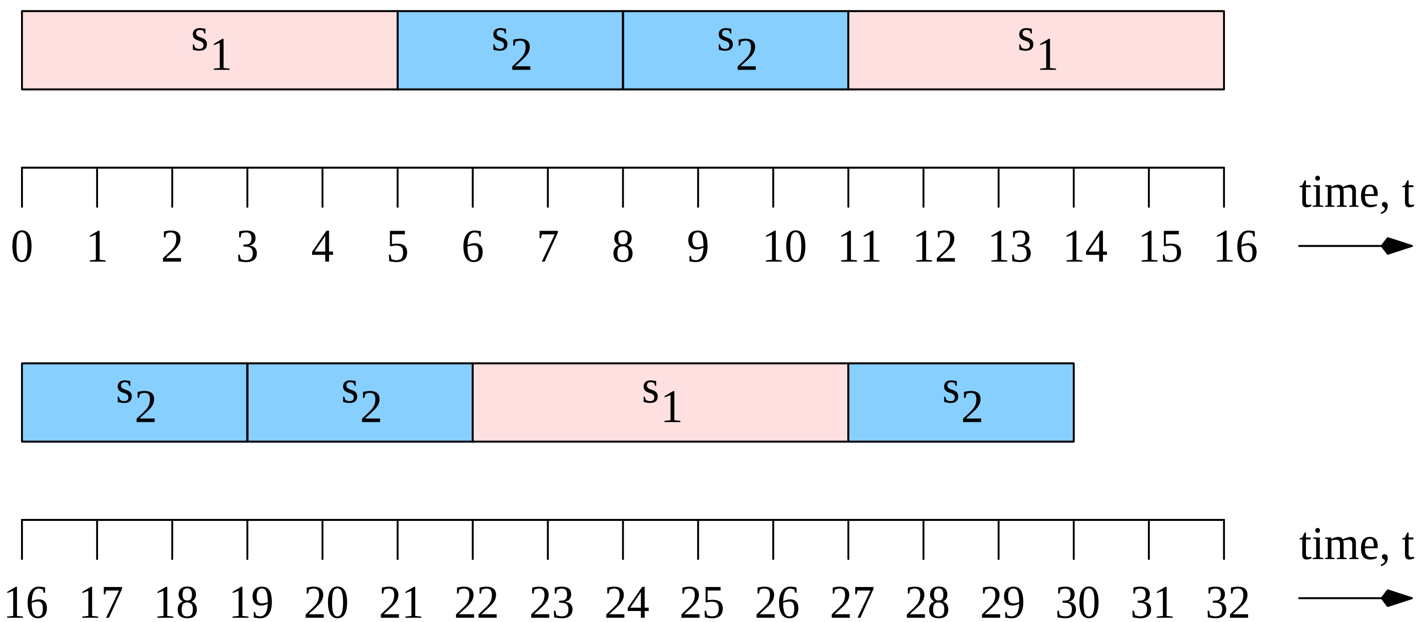


s_1 1/2(0), 1/1(1),1/2(2),1/1(3),1/2(4)...

s_2 3/4(0),2/3(1),2/2(2),1/1(3),3/4(4),2/3(5),2/2(6),1/1(7),3/4(8)...

s_3 6/8(0),5/7(1),4/6(2),3/5(3),3/4(4),2/3(5),1/2(6),0/1(7),6/8(8)...

Example: L1=1/2, L2=1/2, C1=5, C2=3, D1=5, D2=3



s_1 1/2(0),1/1(5),1/2(10),0/1(15),1/2(20),0/1(25),1/2(30)...

s_2 1/2(0),0/1(3),1/4(6),1/3(9),1/2(12),0/1(15),1/4(18),

1/3(21),1/2(24),0/1(27),1/2(30)...

Loss-Tolerance Adjustment (A)

- For stream **i** whose head packet is serviced before its deadline:
  - if ($y_i' > x_i'$) then $y_i' = y_i' - 1$;
  - if ($x_i' = y_i' = 0$) then $x_i' = x_i$; $y_i' = y_i$;
- Where:
  - x_i=Original loss-numerator for stream **i**
  - y_i=Original loss-denominator for stream **i**
  - x_i'=Current loss-numerator for stream **i**
  - y_i'=Current loss-denominator for stream **i**

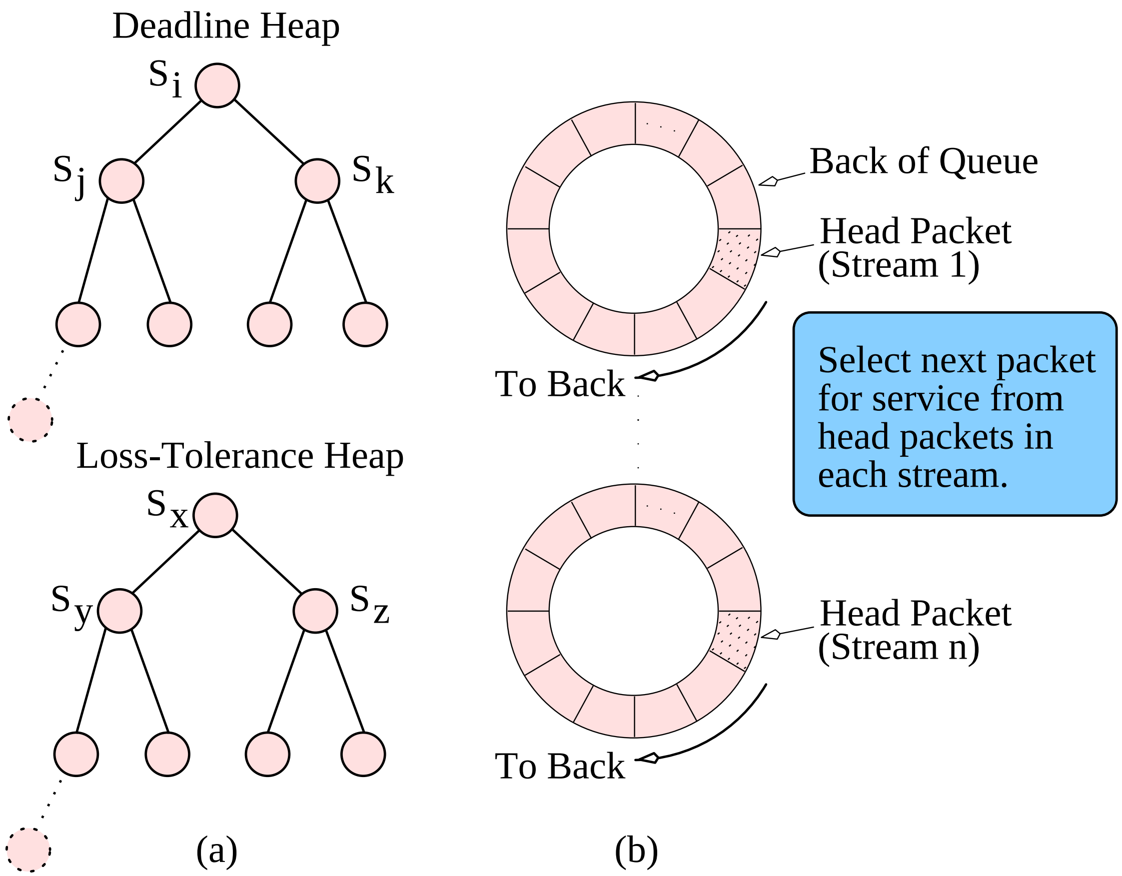
Loss-Tolerance Adjustment (B)

- For stream **j** whose head packet misses its deadline:
  - if ($\mathbf{x_j'} > 0$) then
    - $\mathbf{x_j'}=\mathbf{x_j'}-1$; $\mathbf{y_j'}=\mathbf{y_j'}-1$;
    - if ($\mathbf{x_j'}=\mathbf{y_j'}=0$) then $\mathbf{x_j'}=\mathbf{x_j}$; $\mathbf{y_j'}=\mathbf{y_j}$;
  - else if ($\mathbf{x_j'}=0$) and ($\mathbf{y_j} > 0$) then
    - $\mathbf{x_j'}=2\mathbf{x_j}-1$; $\mathbf{y_j'}=2\mathbf{y_j}+(\mathbf{y_j'}-1)$; (**method 1**)
    - $\mathbf{x_j'}=\mathbf{x_j}$; $\mathbf{y_j'}=\mathbf{y_j}$; (**method 2**)
    - if ($\mathbf{x_j} > 0$) then $\mathbf{y_j'}=\mathbf{y_j'}+\lceil(\mathbf{y_j}-\mathbf{x_j})/\mathbf{x_j}\rceil$; (**method 3**)
    - if ($\mathbf{x_j}=0$) then $\mathbf{y_j'}=\mathbf{y_j'}+\mathbf{y_j}$;

DWCS Algorithm Outline

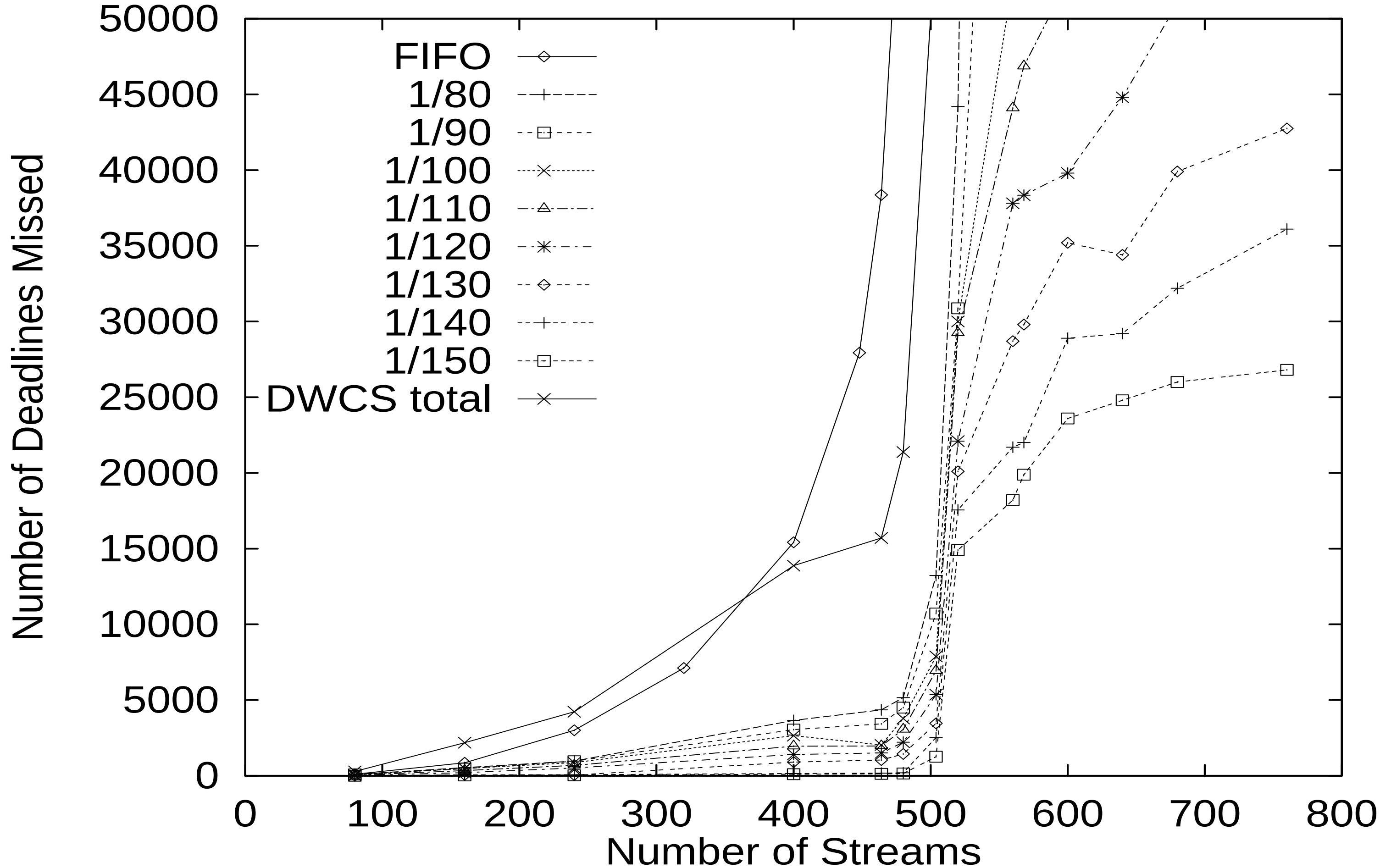
- While **TRUE**:
  - Find stream **i** with highest priority **(see Table)**
  - Service packet at head of stream **i**
  - Adjust loss-tolerance for **i** according to **(A)**
  - Deadline(**i**) = Deadline(**i**) + Inter-Pkt Gap(**i**)
  - For each stream **j** missing its deadline:
    - While deadline is missed:
      - Adjust loss-tolerance for **j** according to **(B)**
      - Drop head packet of stream **j** if droppable
      - Deadline(**j**) = Deadline(**j**) + Inter-Pkt Gap(**j**)

DWCS Implementation



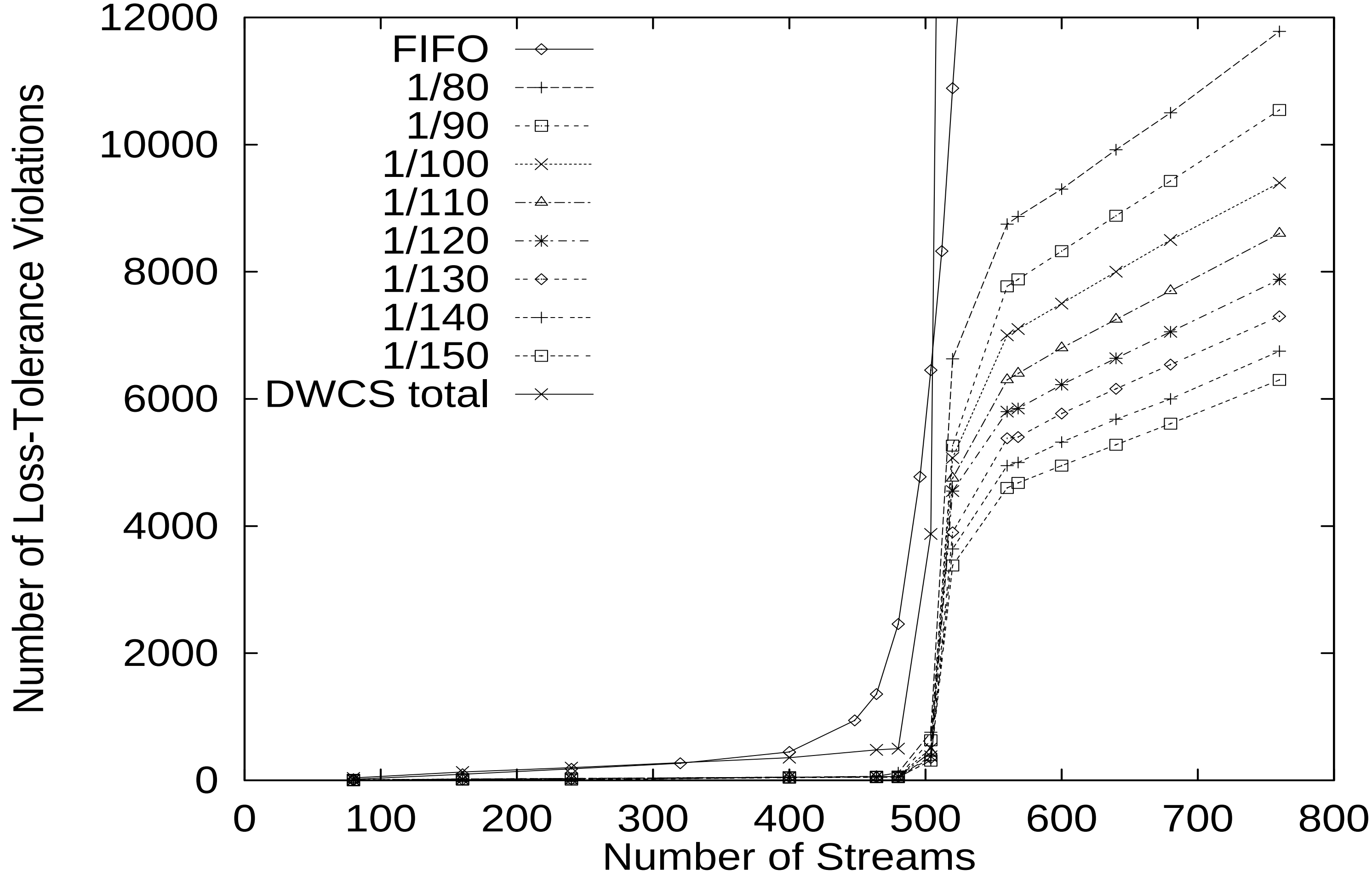
Rich West (1999)

Missed Deadlines (D=500, C=1)



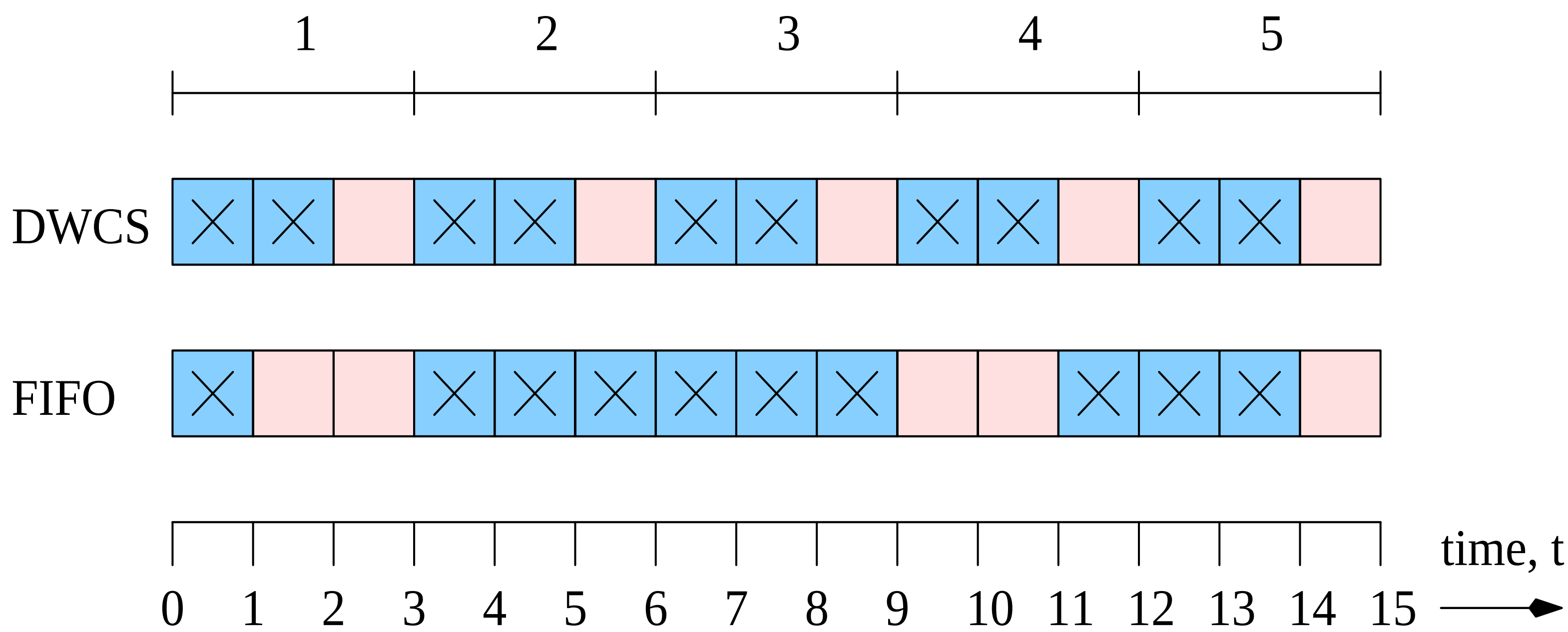
Rich West (1999)

Loss-Tolerance Violations (D=500, C=1)



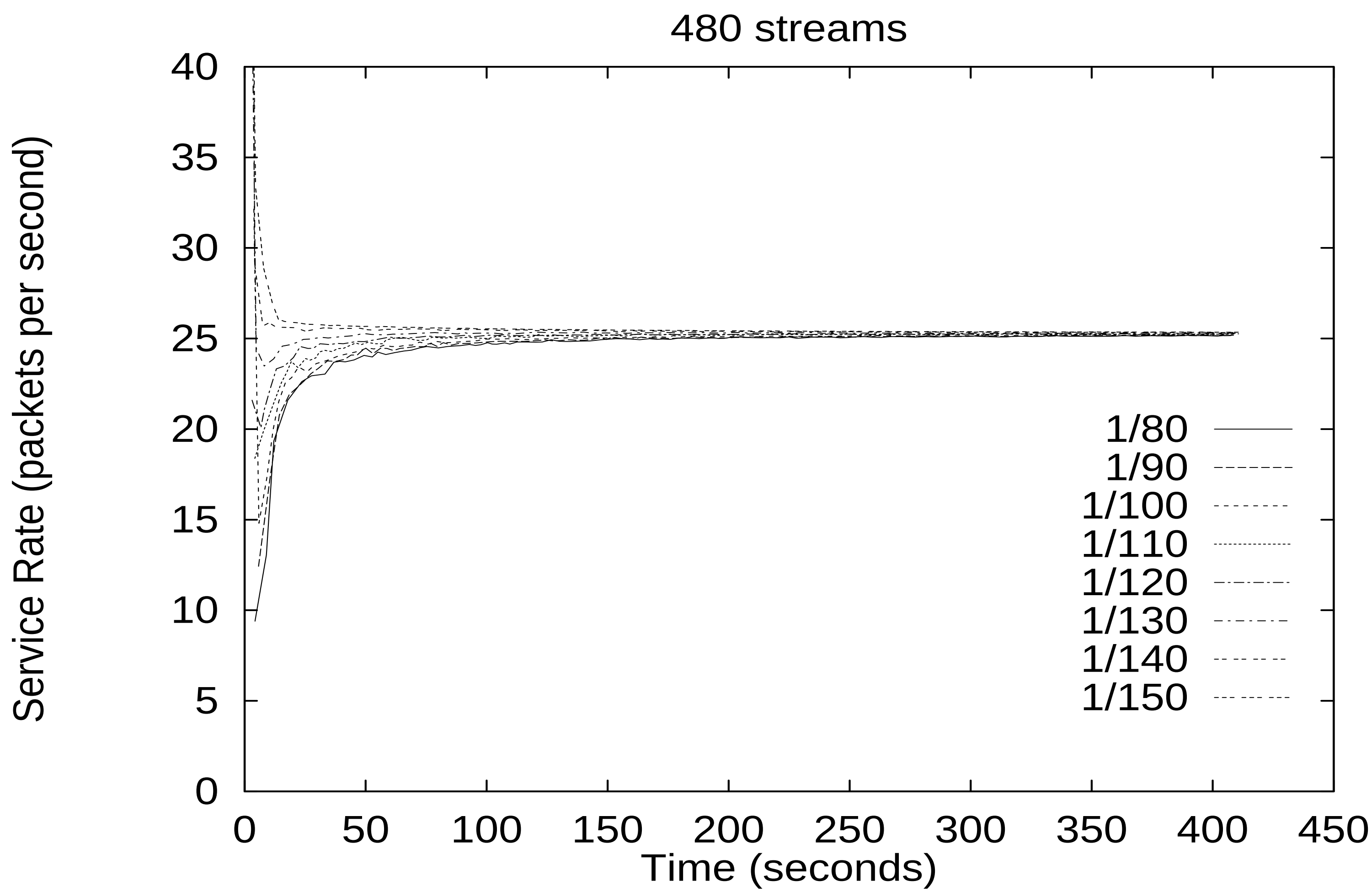
Rich West (1999)

DWCS Spreads Losses

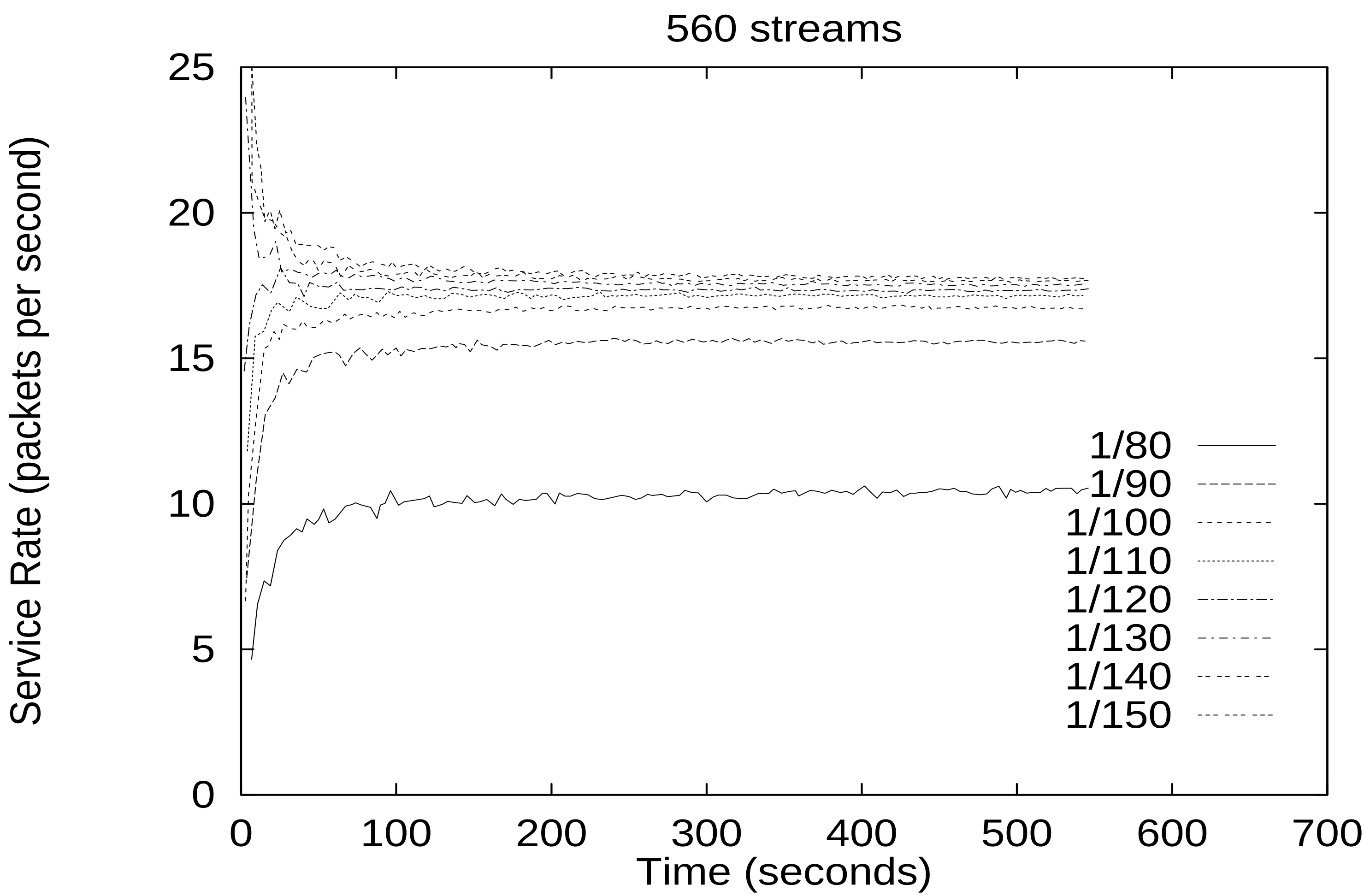


Rich West (1999)

Packets Serviced Per Second

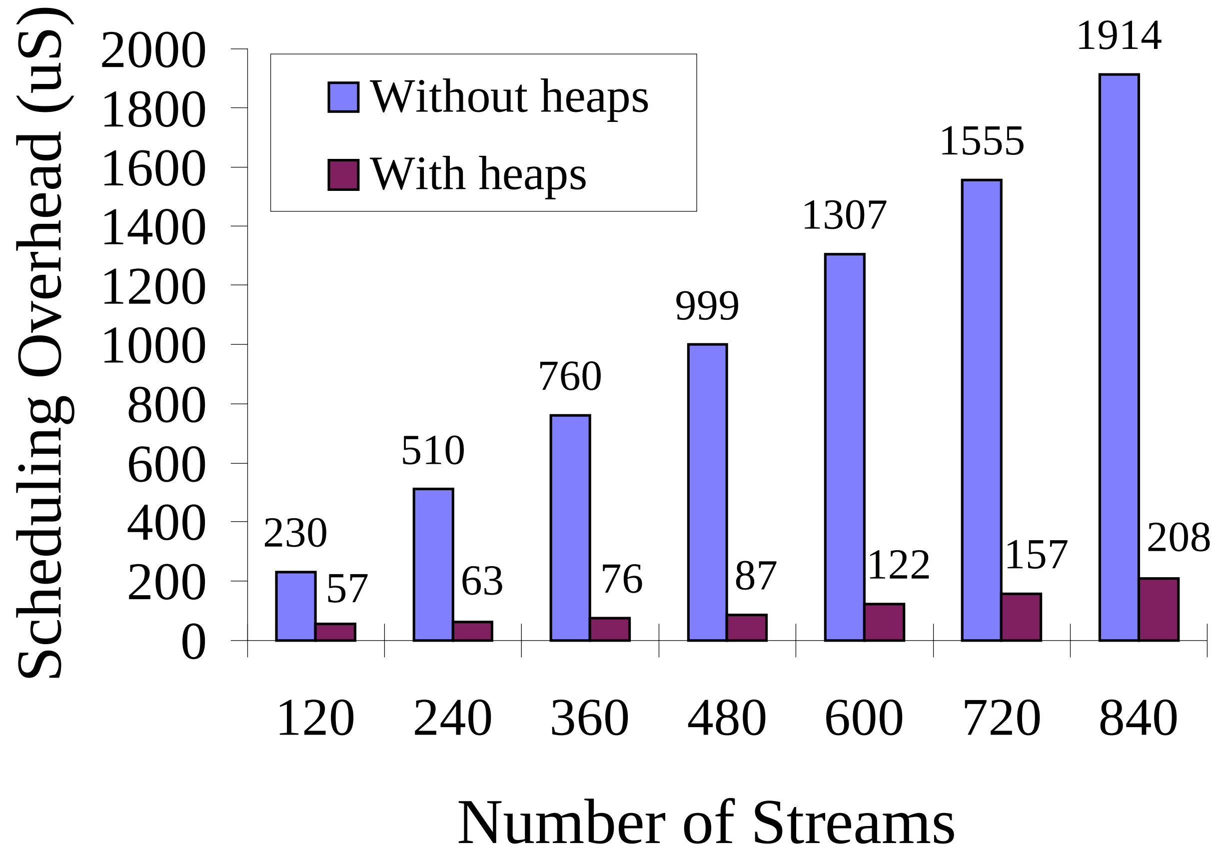


Packets Serviced Per Second



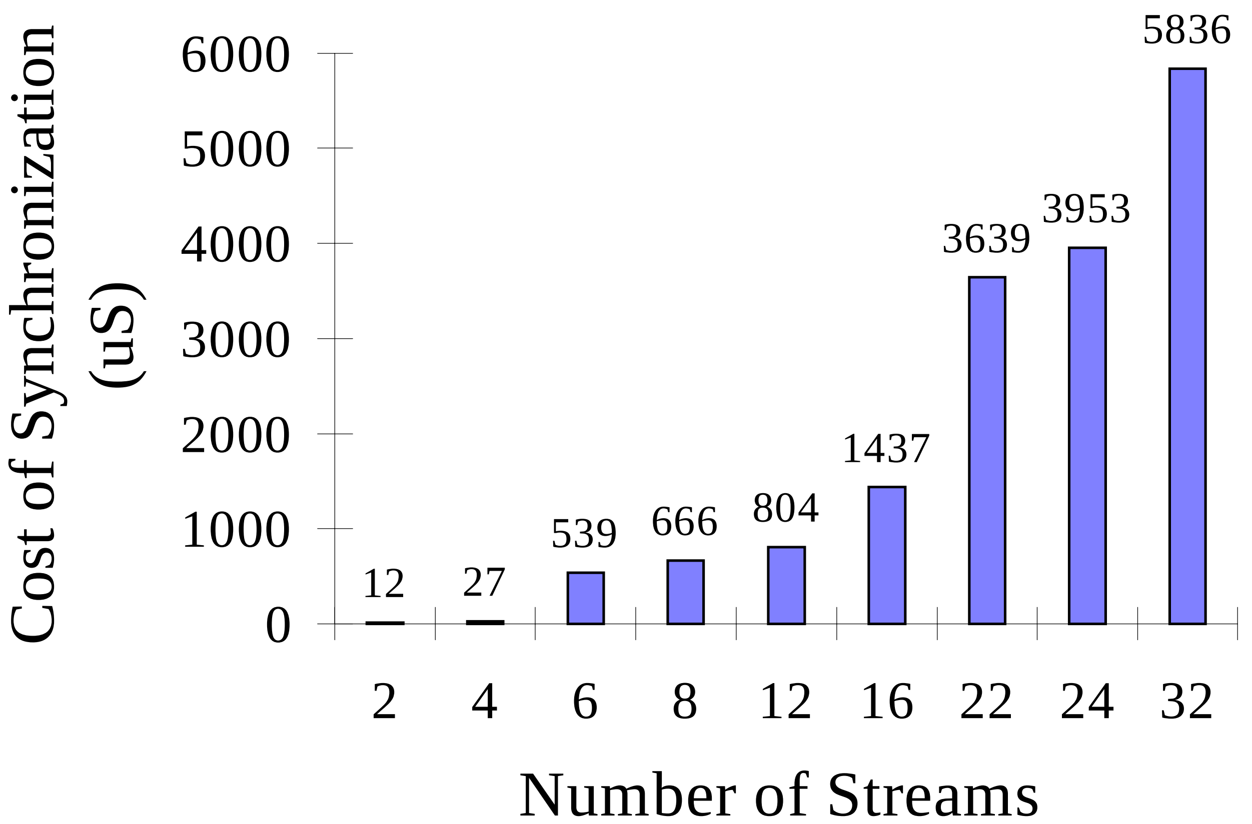
Rich West (1999)

Scheduling Overhead



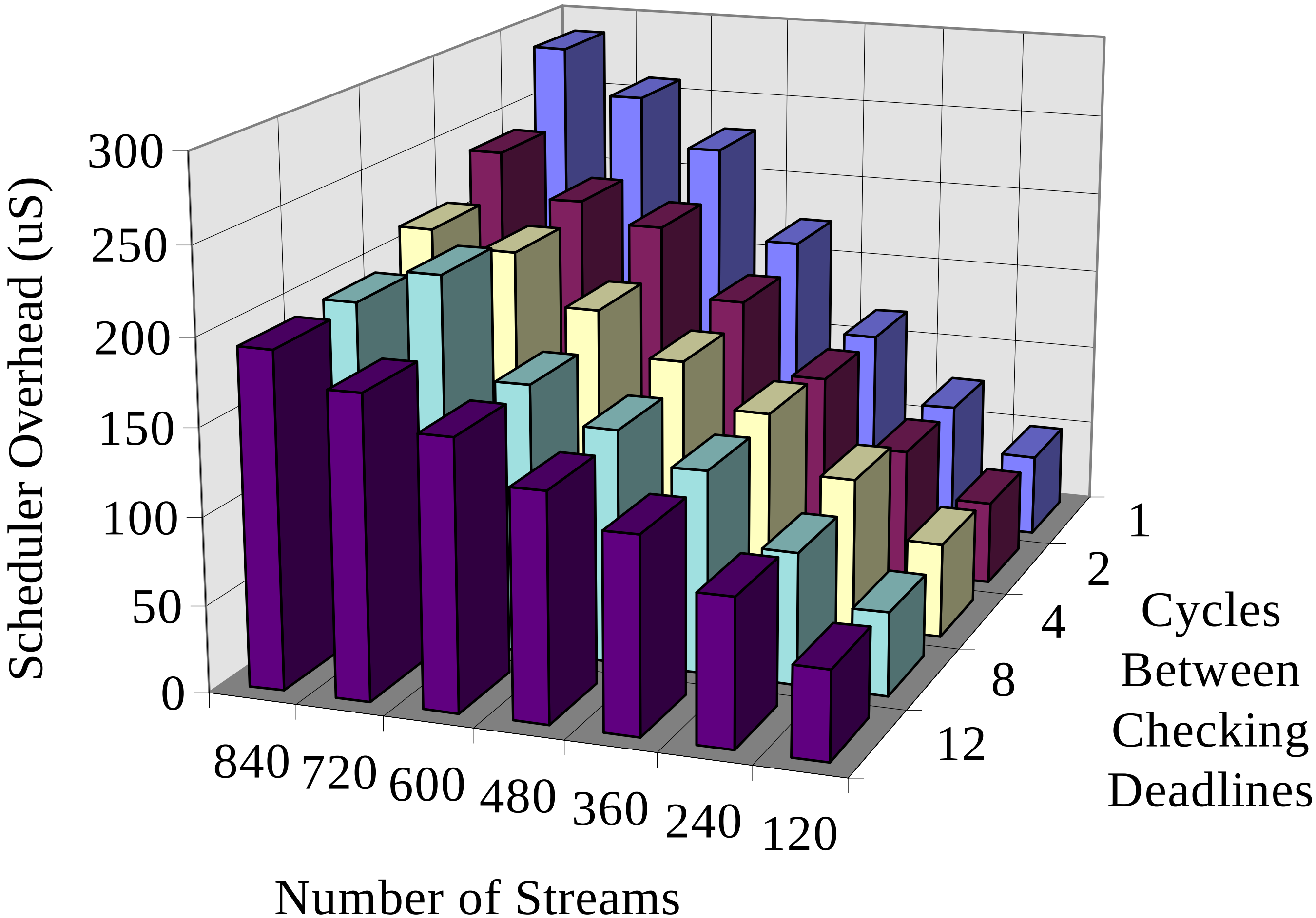
Rich West (1999)

Synchronization Costs



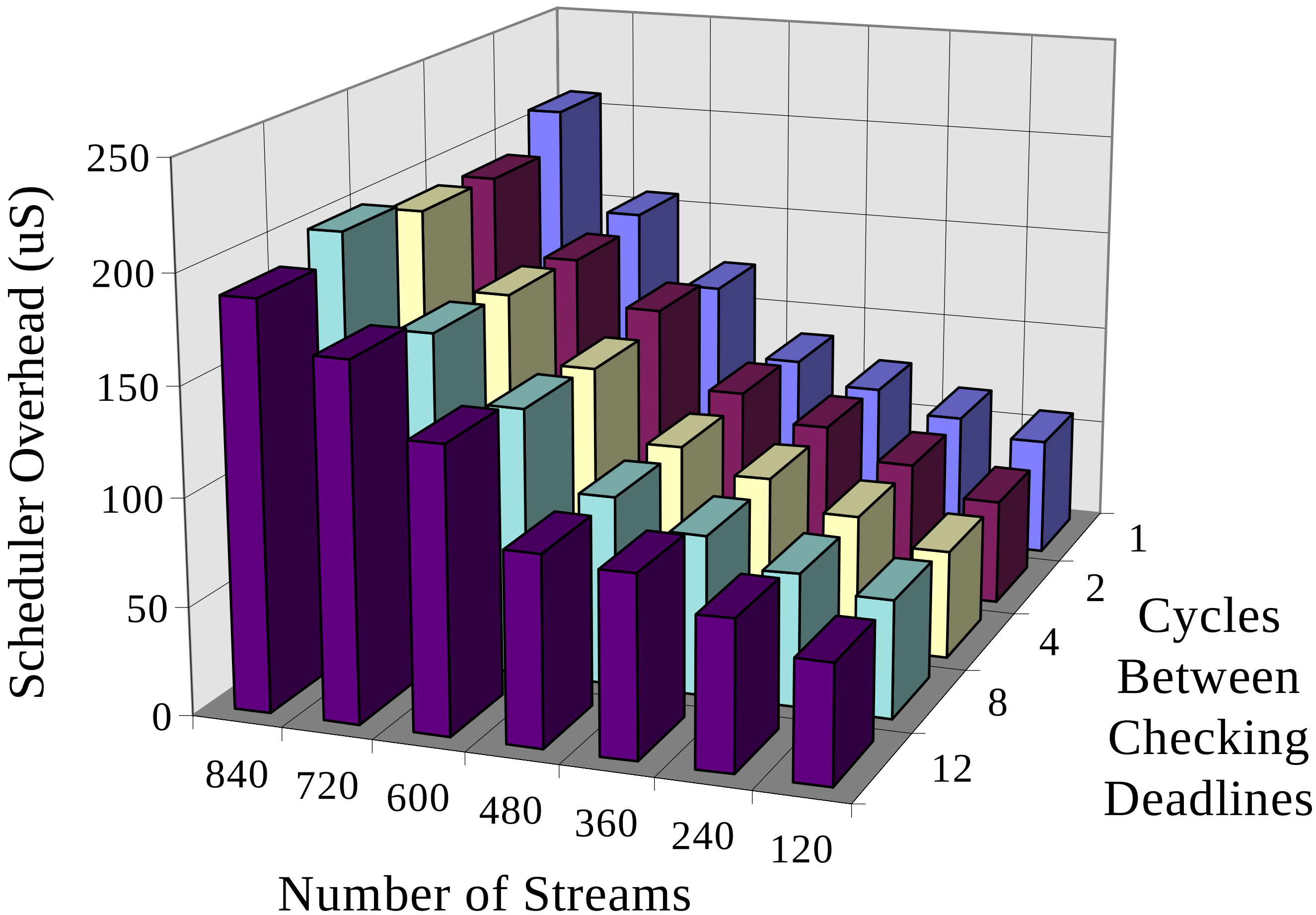
Rich West (1999)

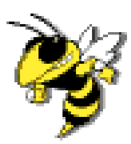
Approximation Overheads (D=200)



Rich West (1999)

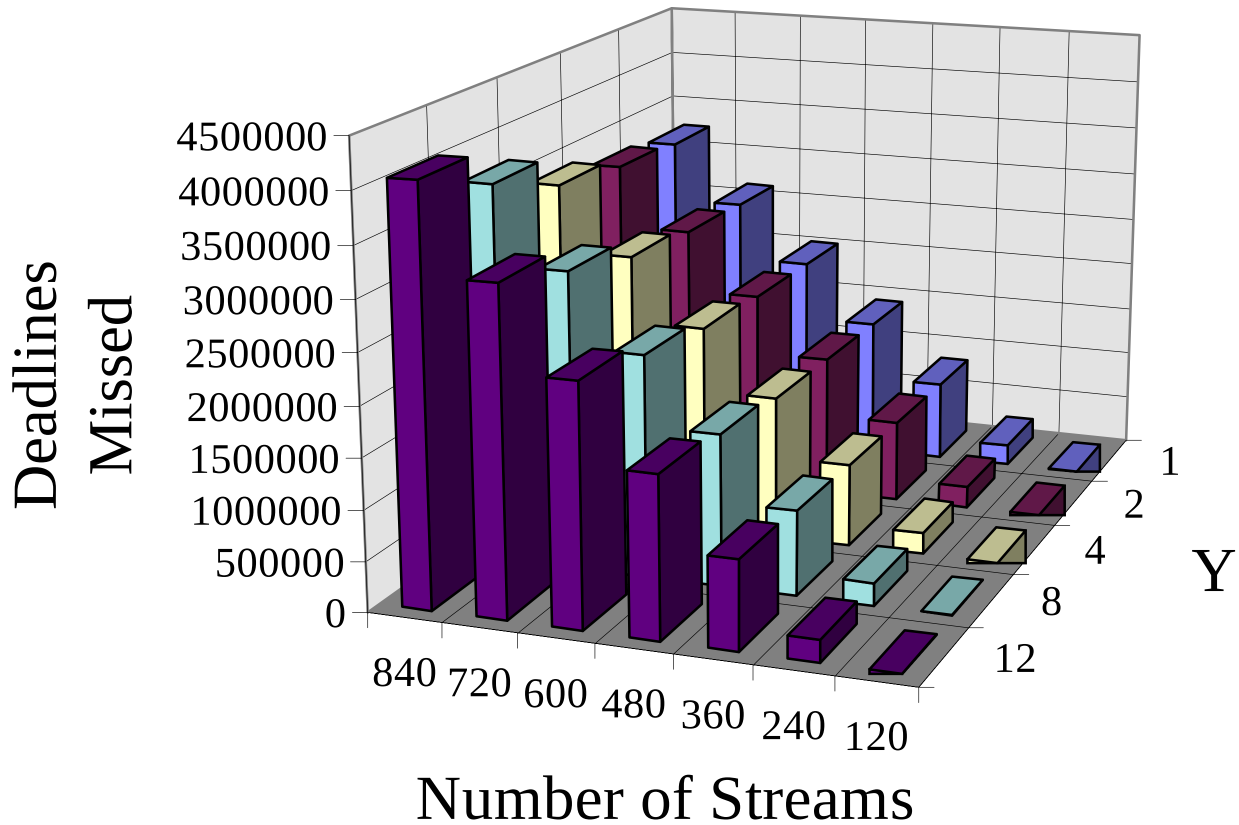
Approximation Overheads (D=500)





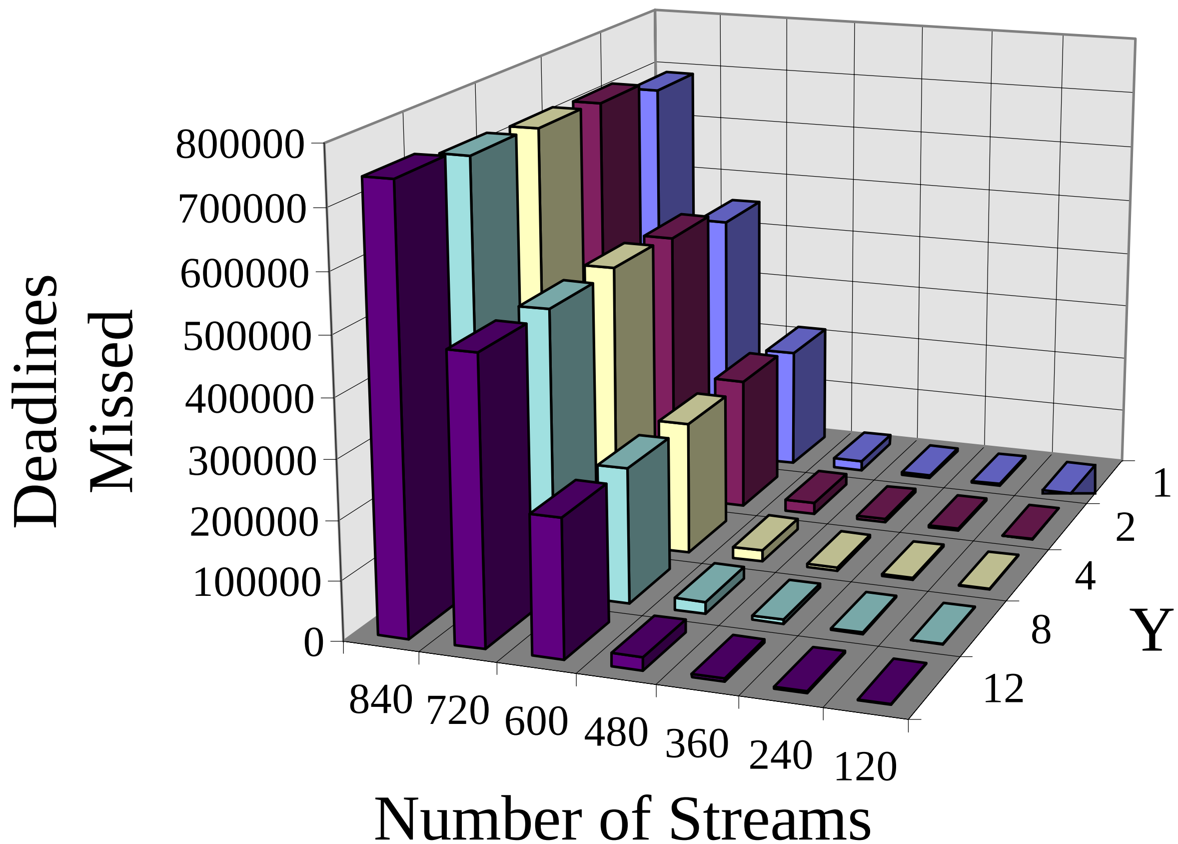
Rich West (1999)

Deadlines Missed (D=200)



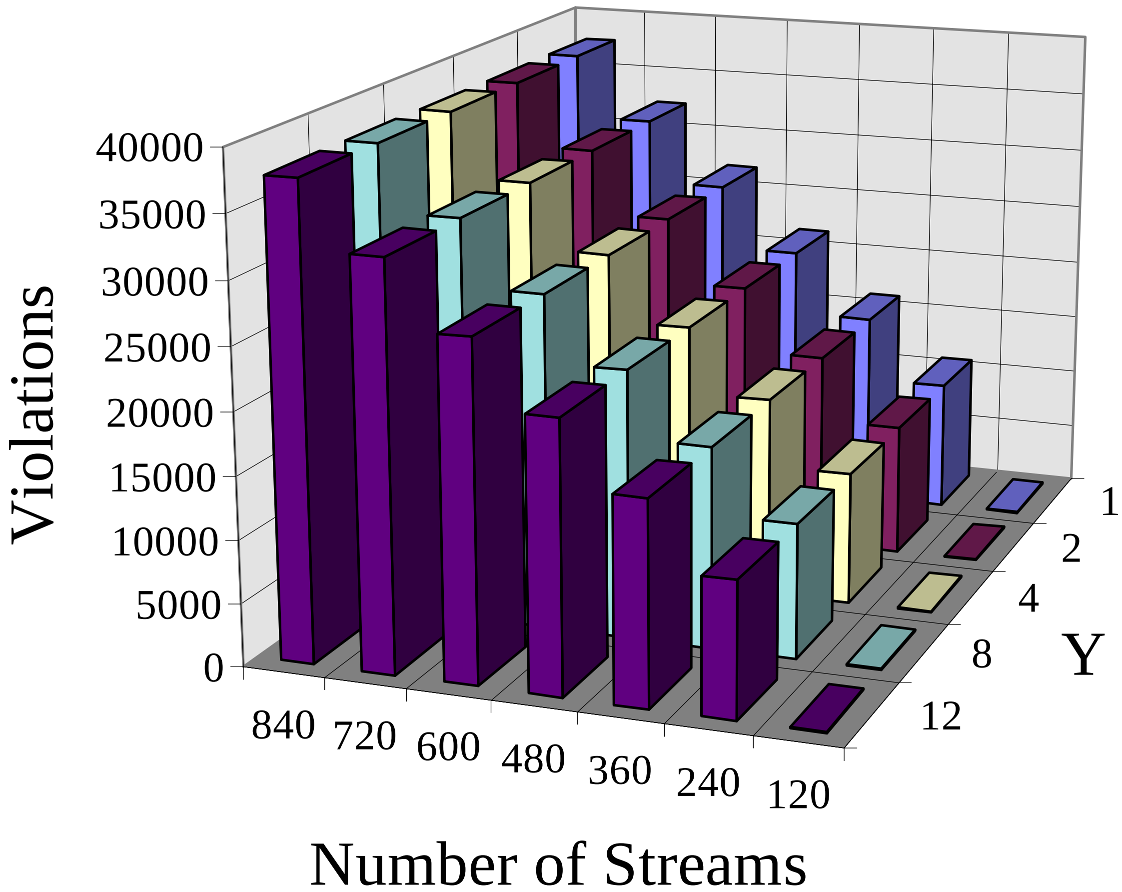
Rich West (1999)

Deadlines Missed (D=500)



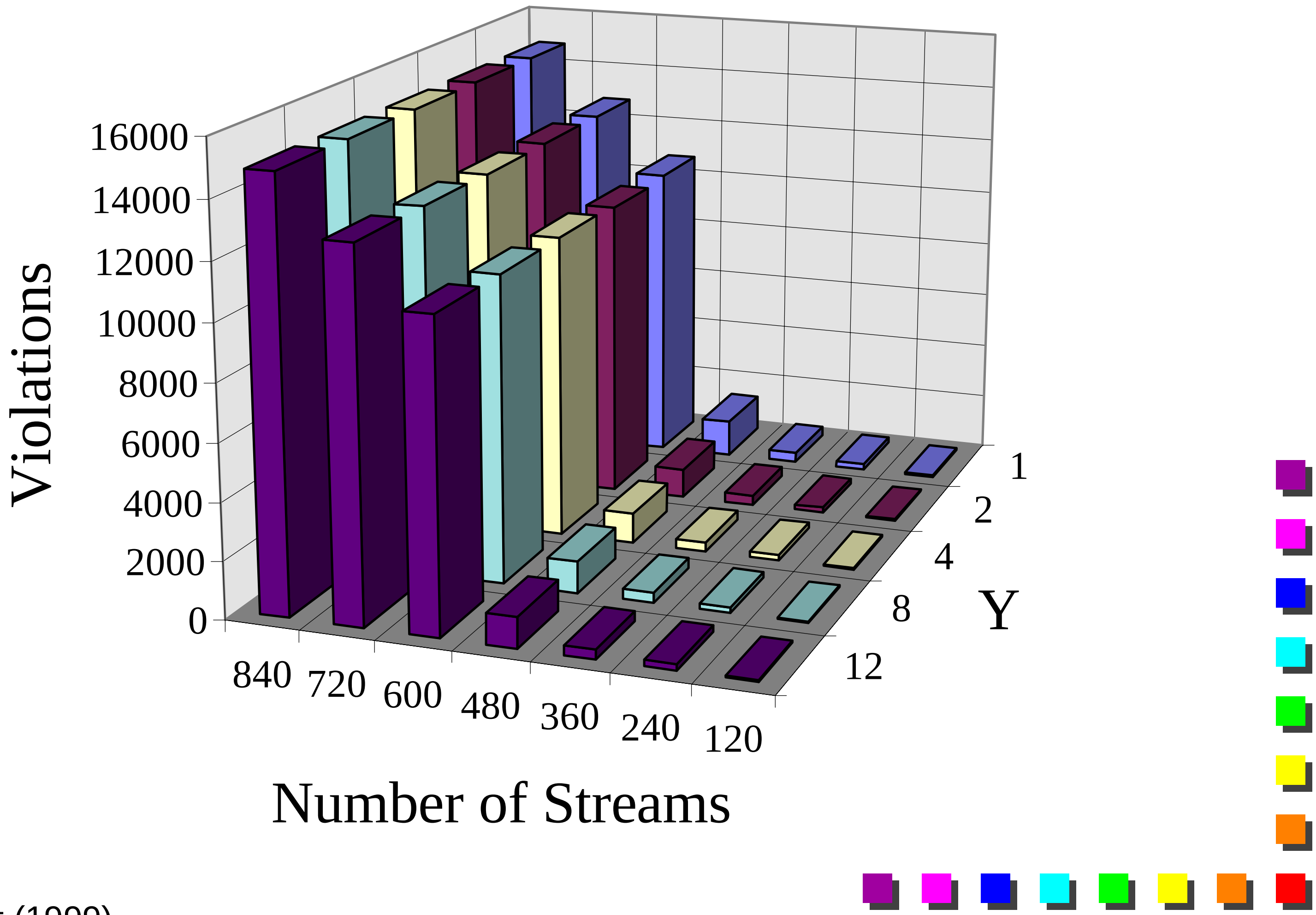
Rich West (1999)

Loss-Tolerance Violations (D=200)



Rich West (1999)

Loss-Tolerance Violations (D=500)



Rich West (1999)

DWCS Summary

- Aimed at servicing packets with delay and loss-constraints.
- Attempts to service each stream so that at most **x** packets are lost/late for every **y** packets requiring service.
  - DWCS minimizes the number of consecutive late packets over any finite window of packets in a given stream.
- Scheduling overhead can be reduced (and scalability increased) by using appropriate data structures (heaps, circular queues) and approximation methods.

DWCS - Current Work

- DWCS is currently being adapted for use as a CPU scheduler (using Linux), for **hard** real-time threads, so that **(y-x)** out of **y** deadlines can be met.
  - Leads to bounded service delay, and guaranteed service in any finite window of service time.
- Aim is to support **coordinated** thread/packet scheduling.

Scheduling Related Work

- **Fair Scheduling**: WFQ/WF^{2}Q (Shenker, Keshav, Bennett, Zhang etc), SFQ (Goyal et al), EEVDF/Proportional Share (Stoica, Jeffay et al).
- **(m,k) Deadline Scheduling**: Distance-Based Priority (Hamdaoui & Ramanathan), Dual-Priority Scheduling (Bernat & Burns).