Analysis of a Window-Constrained Scheduler for Real-Time and Best-Effort Packet Streams

Richard West & Christian Poellabauer

Boston University & Georgia Institute of Technology

Introduction

- Certain distributed, RT applications can tolerate lost / late info transferred across a network.
  - e.g., streaming multimedia applications.
- Restrictions on:
  - numbers of **consecutive** late / lost packets.
- Need:
  - real-time scheduling of packets (info carriers).
  - guarantees that no more than **x** out of **y** packets are late / lost.

Contributions

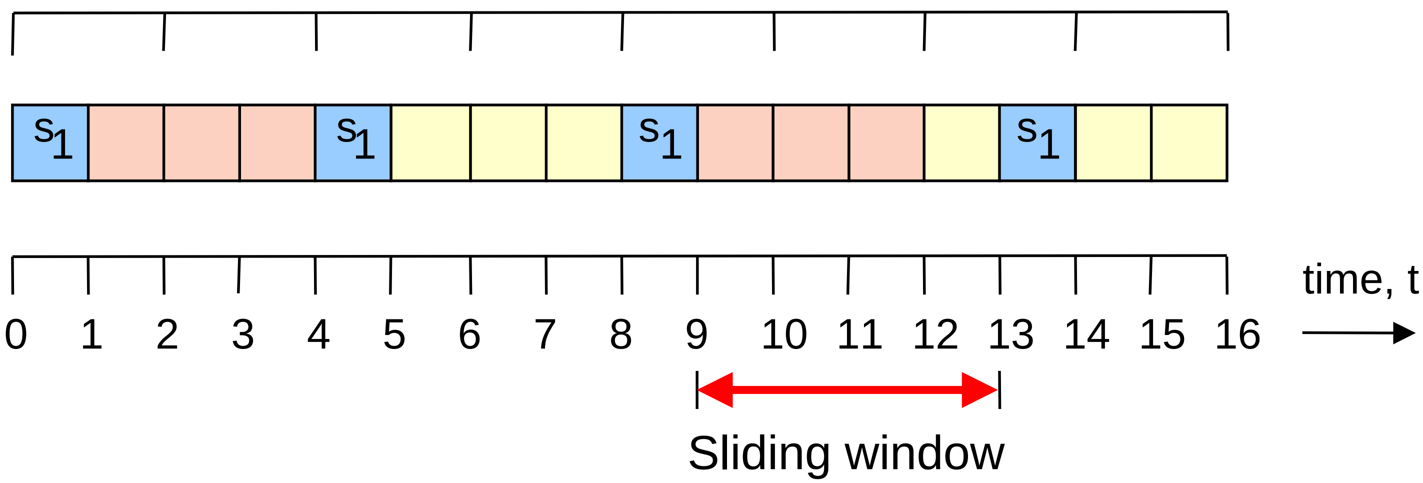
- Dynamic Window-Constrained Scheduling (DWCS):
  - Can guarantee at most **x** late / lost packets every fixed window of **y** packets.
    - **(x,y)-hard** as opposed to **(x,y)-firm** deadlines!
  - Bounded service delay, even in overload.
  - 100% utilization bound for fixed-length packets.
- Fast response & low jitter for best-effort packet streams.

DWCS Packet Scheduling

- Two attributes per packet stream, $\mathbf{S_i}$:
  - Request period, $\mathbf{T_i}$.
    - Defines interval between deadlines of consecutive pairs of packets in $\mathbf{S_i}$.
  - Window-constraint, $\mathbf{W_i} = \mathbf{x_i/y_i}$.
    - Essentially, a “loss-tolerance”.

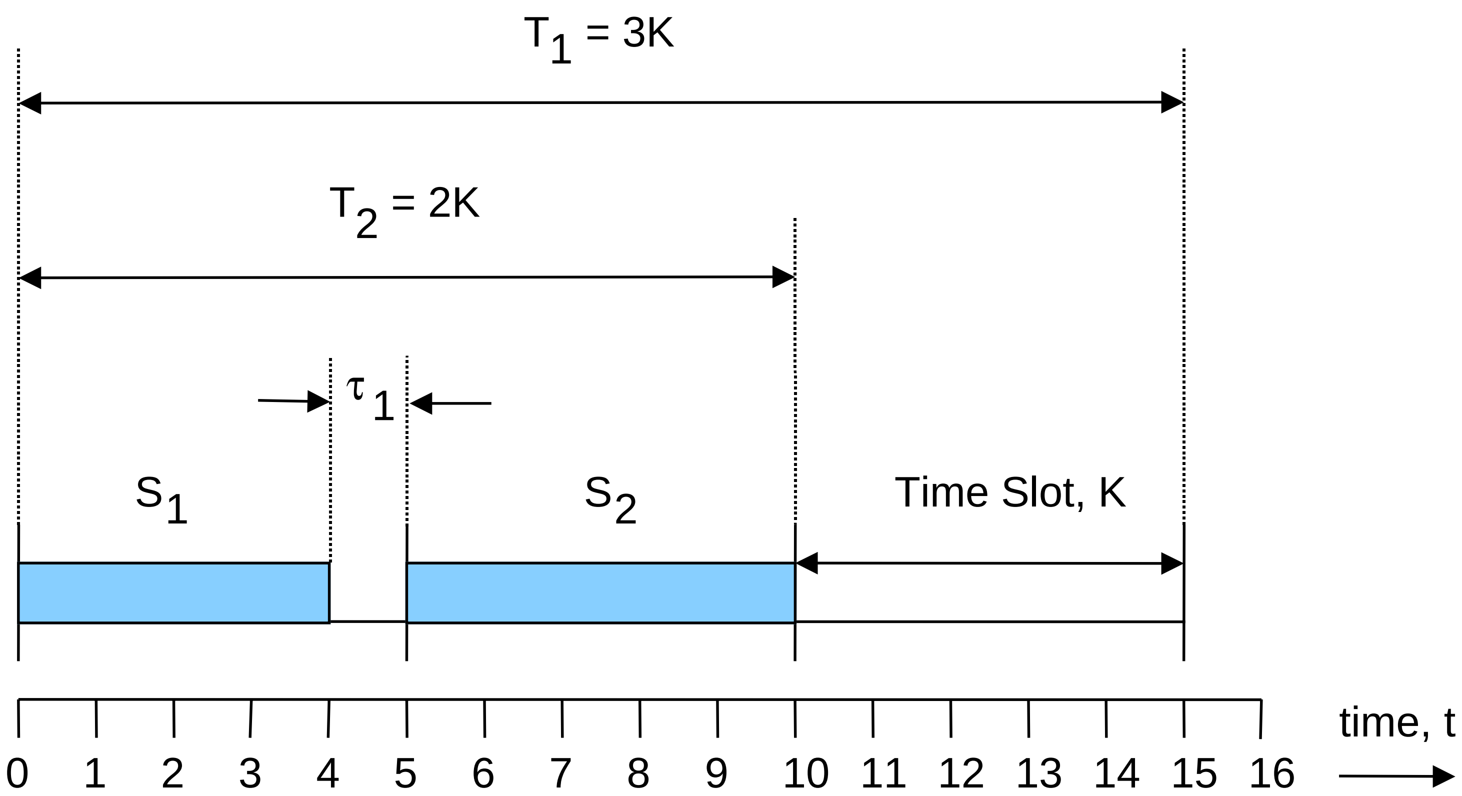
“x out of y” Guarantees

- e.g., Stream S_1 with C_1=1, T_1=2 and W_1=1/2



- Feasible schedule if “x out of y” guarantees are met.

Scheduling Granularity



Rich West (2000)

Pairwise Packet Ordering Table

Precedence amongst pairs of packets
• Earliest deadline first (EDF)
• Same deadlines, order lowest window-constraint first
• Equal deadlines and zero window-constraints, order highest window-denominator first
• Equal deadlines and equal non-zero window-constraints, order lowest window-numerator first
• All other cases: first-come-first-serve

Original Pairwise Packet Ordering Table

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

Window-Constraint Adjustment (A)

- For stream $\mathbf{S_i}$ whose head packet is serviced **before** its deadline:
  - if ($\mathbf{y_i'} > \mathbf{x_i'}$) then $\mathbf{y_i'}=\mathbf{y_i'}-1$;
  - else if ($\mathbf{y_i'} = \mathbf{x_i'}$) **and** ($\mathbf{x_i'} > 0$) then
    - $\mathbf{x_i'}=\mathbf{x_i'}-1$; $\mathbf{y_i'}=\mathbf{y_i'}-1$;
  - if ($\mathbf{x_i'}=\mathbf{y_i'}=0$) **or** ($\mathbf{S_i}$ **is tagged**) then
    - $\mathbf{x_i'}=\mathbf{x_i}$; $\mathbf{y_i'}=\mathbf{y_i}$;
  - if ($\mathbf{S_i}$ **is tagged**) then **reset tag**;

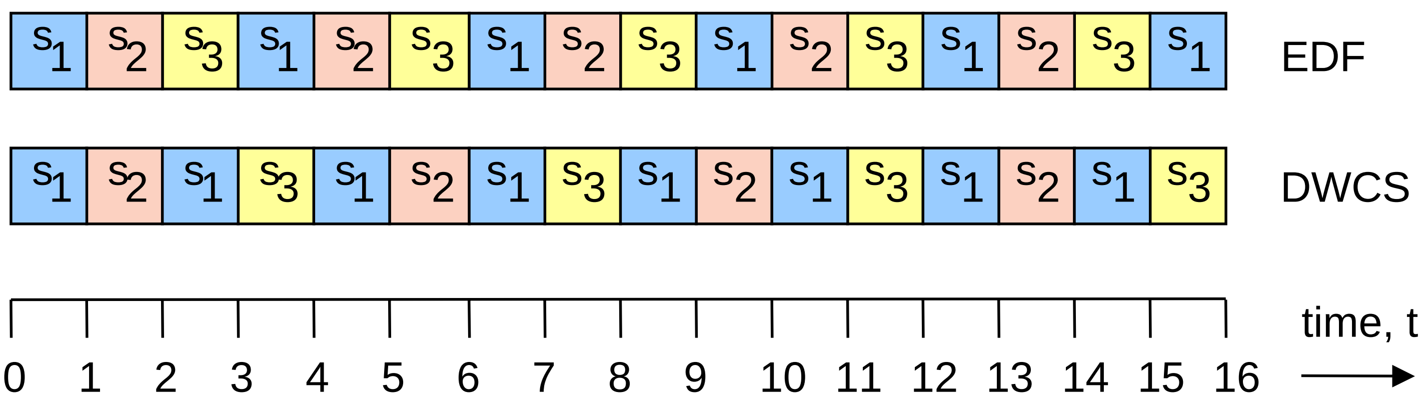
Window-Constraint Adjustment (B)

- For stream $\mathbf{S_j}$ 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
    - $\mathbf{y_j'}=\mathbf{y_j'}+\varepsilon$;
    - **Tag $\mathbf{S_j}$** with a violation;

DWCS Algorithm Outline

- Find stream $\mathbf{S_i}$ with highest priority **(see Table)**
- Service head packet of stream $\mathbf{S_i}$
- Adjust $\mathbf{W_i'}$ according to **(A)**
- $\mathbf{Deadline_i}$ = $\mathbf{Deadline_i}$ + $\mathbf{T_i}$
- For each stream $\mathbf{S_j}$ missing its deadline:
  - While deadline is missed:
    - Adjust $\mathbf{W_j'}$ according to **(B)**
    - Drop head packet of stream $\mathbf{S_j}$ if droppable
    - $\mathbf{Deadline_j}$ = $\mathbf{Deadline_j}$ + $\mathbf{T_j}$

EDF versus DWCS



s_1 1/2(1),1/1(2),1/2(3),1/1(4),1/2(5)...

s_2 3/4(1),2/3(2),2/2(3),1/1(4),3/4(5),2/3(6),2/2(7),1/1(8),3/4(9)...

s_3 6/8(1),5/7(2),4/6(3),3/5(4),3/4(5),2/3(6),1/2(7),0/1(8),6/8(9)...

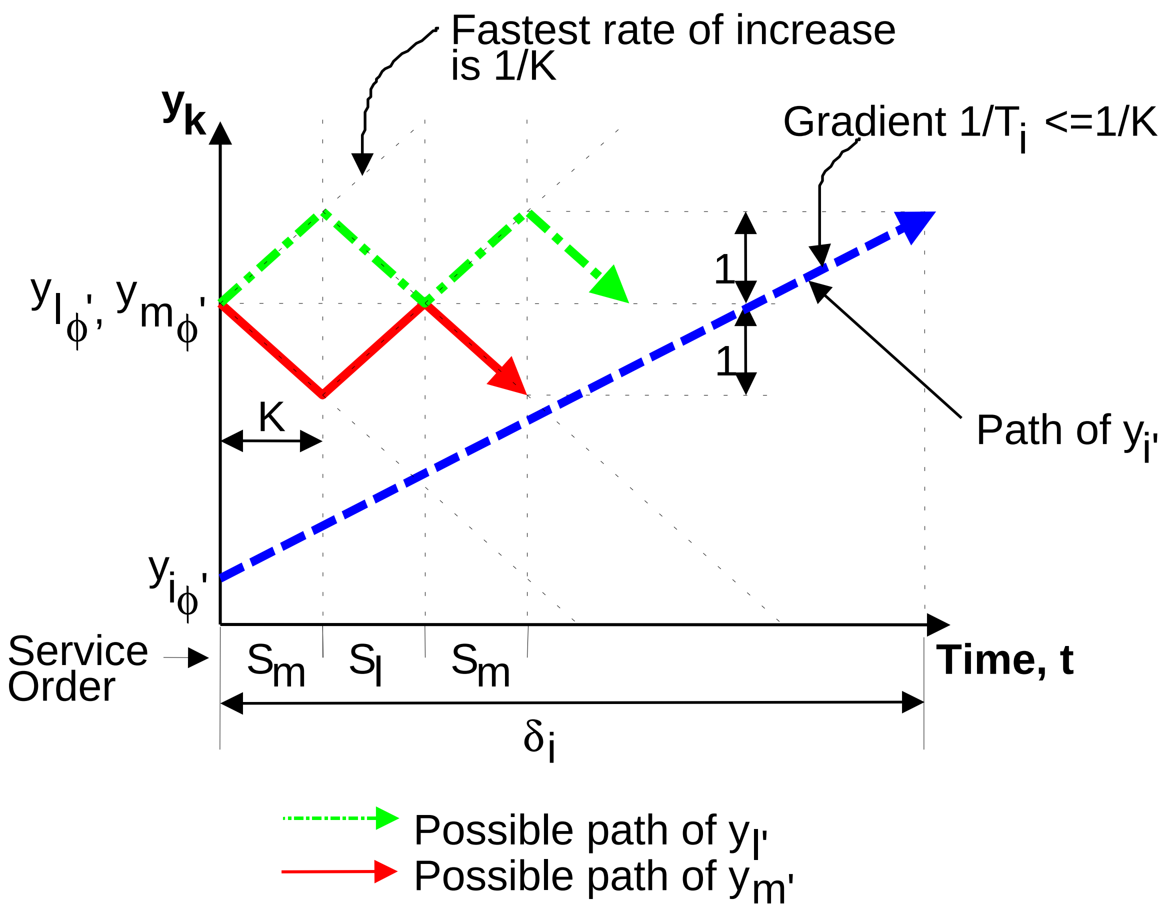
Rich West (2000)

DWCS Delay Characteristics

- If feasible schedule, max delay of service to $\mathbf{S_i}$ is:
  - $(\mathbf{x_i} + 1)\mathbf{T_i} - \mathbf{C_i}$
  - Note: Every time $\mathbf{S_i}$ is not serviced for $\mathbf{T_i}$ time units $\mathbf{x_i'}$ is decremented by 1 until it reaches 0.

- If no feasible schedule, max delay of service to $\mathbf{S_i}$ is still bounded.
- Function of time to have:
  - Earliest deadline, lowest window-constraint, highest window-denominator.

Possible Change in Window-Denominators



Rich West (2000)

Bandwidth Utilization

- Minimum utilization factor of stream $\mathbf{S_i}$ is:

$$U_i = \frac{(y_i - x_i)C_i}{y_i T_i}$$

  - i.e., min req’rd fraction of bandwidth.
- **Least upper bound** on utilization is min of utilization factors for all streams that fully utilize bandwidth.
  - i.e., guarantees a feasible schedule.
  - L.U.B. is 100% in a slotted-time system.

Least Upper Bound on Utilization

- Why 100%?
- If all **W_i**'s are 0, all deadlines must be met. DWCS schedules packets in EDF order - optimal.
- If all **W_i**'s > 0, DWCS schedules EDF then lowest **W_i** first.
  - If all **W_i**'s are normalized to same denominator, intuitively worst-case tolerable delay of a stream with lowest **W_i** is less than one with higher **W_i**.
  - This is like scheduling in EDF order, which is optimal.

Scheduling Test

- If:

$$\sum_{i=1}^{n} \frac{(1-\frac{x_i}{y_i}).C_i}{T_i} \leq 1.0$$

  and C_i=**K**, T_i=**qK** for all **i**, where **q** is 1,2,…etc, then a feasible schedule exists.
- For variable length packets:
  - let C_i<=**K** for all **i** or fragment/combine packets & translate service constraints.
    - e.g., ATM SAR layer.

Simulation Scenario

- 8 classes of packet streams:
  - (W_i, T_i) = {1/10,400}, {1/20,400}, {1/30,480}, {1/40,480}, {1/50, 560}, {1/60, 560}, {1/70,640}, {1/80,640}
- Varied number of streams **n**, uniformly distributed amongst traffic classes.
- Total of a million packets serviced.

Simulation Scenario

- 8 classes of packet streams:

W_i	1/10	1/20	1/30	1/40	1/50	1/60	1/70	1/80
T_i	400	400	480	480	560	560	640	640

- Varied number of streams **n**, uniformly distributed amongst traffic classes.
- Total of a million packets serviced.

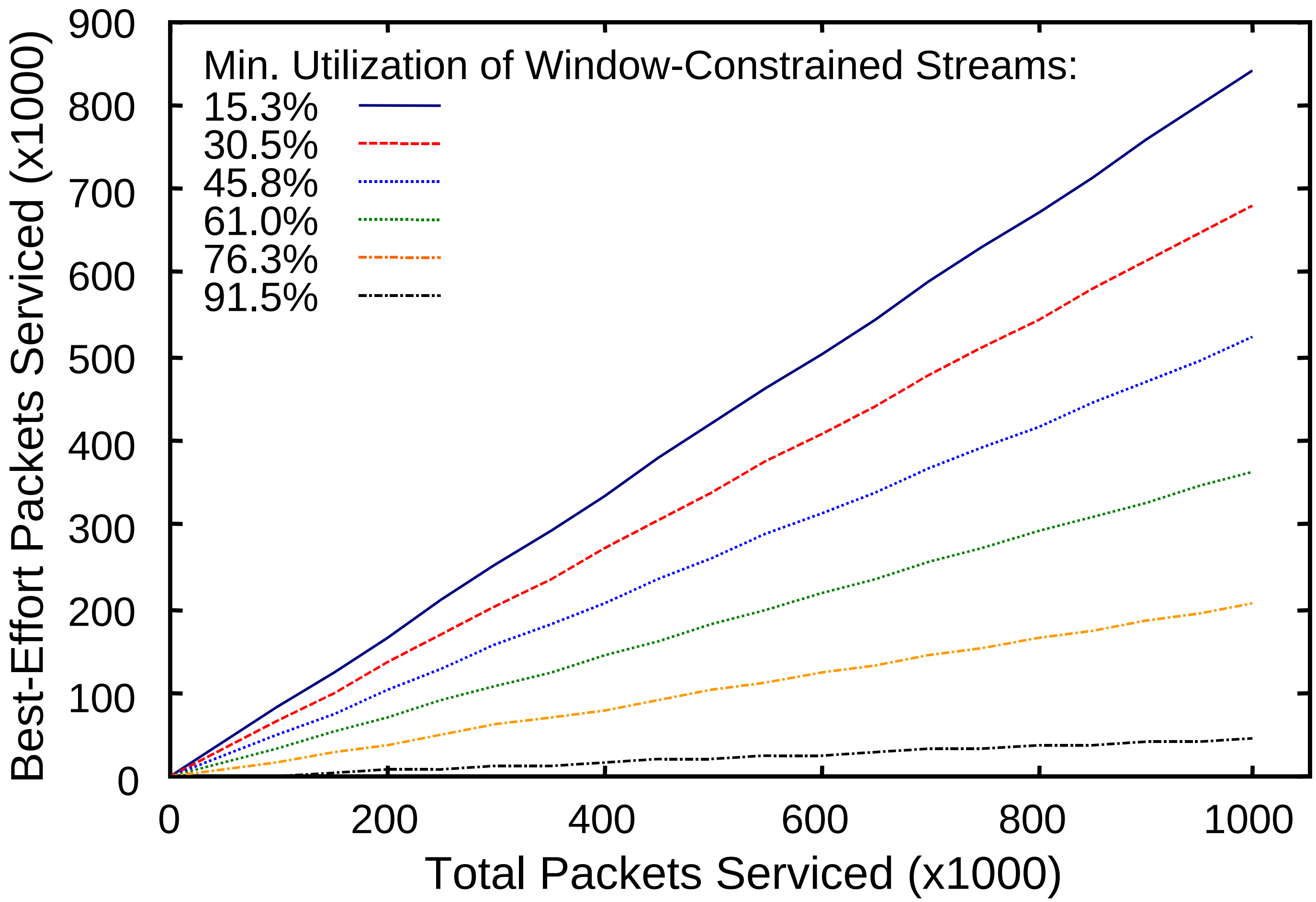
Bandwidth Utilization Results

n	D	V	U	$\frac{n}{8}\cdot\sum_{i=1}^{8}\frac{C_i}{T_i}$
480	0	0	0.9156	0.9518
496	0	0	0.9461	0.9835
504	0	0	0.9613	0.9994
512	15152	0	0.9766	1.0152
520	30990	0	0.9919	1.0311
528	46828	7038	1.0071	1.047
544	78528	31873	1.0376	1.0787
560	110240	53455	1.0681	1.1104
640	268800	148143	1.2207	1.269

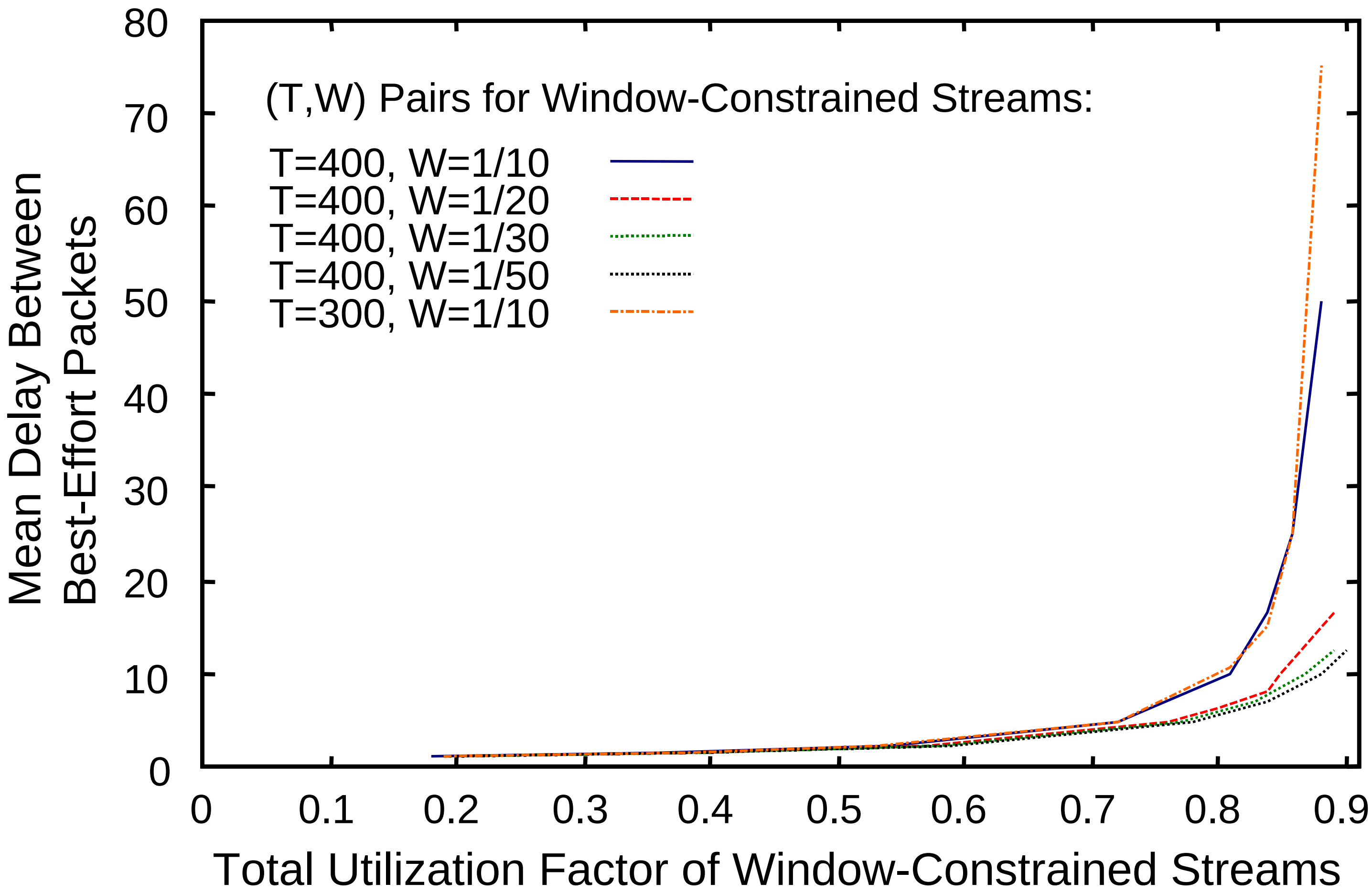
Heterogeneous Packet Streams

- Minimize mean delay (or jitter) to best-effort packets.
- Maintain service guarantees to real-time packets.
- For best-effort packets:
  - calculate pseudo values, $\mathbf{W}_{BE}$ and $\mathbf{T}_{BE}$ and treat like RT packets, or
  - service best-effort packets when all RT packets serviced in current request periods.

Heterogeneous Packet Streams - Simulated Results (1)



Heterogeneous Packet Streams - Simulated Results (2)



Conclusions

- Presented a modified version of DWCS from that in RTAS'99:
  - Support for **(x,y)-hard** deadlines as opposed to **(x,y)-firm** deadlines.
  - Bounded service delay, even in overload.
  - 100% utilization bound for fixed-length packets.
  - Fast response for best-effort packet streams.
- DWCS aimed at servicing packets with delay and loss-constraints.

Current and Future Work

- Switch / co-processor implementation of DWCS.
- Scheduling variable-length packets.
- Replacement CPU scheduler in Linux kernel.
  - **www.cc.gatech.edu/~west/dwcs.html**
  - “Guarantee” **minimum** quantum of service every fixed window of service time to competing threads.

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), Skip-Over (Koren & Shasha).
- **Pinwheel Scheduling**: Holte, Baruah etc.
- **Other multimedia scheduling**: SMART (Nieh and Lam).

Related Research Papers

- **Experimentation with Event-Based Methods of Adaptive QoS Management**, *GIT-CC-99-25.*
- **Analysis of a Window-Constrained Scheduler for Real-Time and Best-Effort Traffic Streams**, *RTSS'2000.*
- **Dynamic Window-Constrained Scheduling for Multimedia Applications**, *ICMCS'99*.
- **Scalable Scheduling Support for Loss and Delay-Constrained Media Streams**, *RTAS'99*.
- **Exploiting Temporal and Spatial Constraints on Distributed Shared Objects**, *ICDCS'97*.