Dynamic Window-Constrained Scheduling for Multimedia Applications

Richard West and Karsten Schwan
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.
  - DWCS can be both fair and unfair when necessary - Performs Fair-Queueing, SP and EDF.
  - We demonstrate DWCS using a streaming video application.
DWCS Packet Scheduling

- Two attributes per packet:
  - Deadline (max inter-packet gap).
  - Loss-tolerance, \( \frac{x}{y} \).
    - \( x \) 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.

Rich West (1999)
DWCS - Conceptual View

Higher Priority = Lower Loss-Tolerance

Network Pipe

EDF-ordered queues
Heterogeneous Scheduling

- Higher Priority = Lower Loss-Tolerance
  (Time-constrained traffic)

- Lowest loss-tolerance first
  (Non-time-constrained traffic)

EDF-ordered queues → 0 loss-tolerance queue

Network Pipe

Compare first packet with packet at head of 0 loss-tolerance queue to see if schedulable.

Rich West (1999)
# 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-denominator 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

\[
\begin{array}{cccccccccccccccc}
& s_1 & s_2 & s_1 & s_3 & s_1 & s_2 & s_1 & s_3 & s_1 & s_2 & s_1 & s_3 & s_1 & s_2 & s_1 & s_3 \\
0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 14 & 15 & 16 \\
\end{array}
\]

\begin{align*}
& s_1 & 1/2(0), 1/1(1),1/2(2),1/1(3),1/2(4)...
\end{align*}

\begin{align*}
\end{align*}

\begin{align*}
& 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)...
\end{align*}

Rich West (1999)
Example: L1=1/2, L2=1/2, C1=5, C2=3, D1=5, D2=3

\[ s_1 \quad s_2 \quad s_2 \quad s_1 \]

\[ s_2 \quad s_2 \quad s_1 \quad s_2 \]

\[ s_1 \]: 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)...

Rich West (1999)
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 $(x_j' > 0)$ then
  - $x_j' = x_j' - 1$; $y_j' = y_j' - 1$;
- if $(x_j' = y_j' = 0)$ then $x_j' = x_j$; $y_j' = y_j$;
- else if $(x_j' = 0)$ and $(y_j > 0)$ then
  - $x_j' = 2x_j - 1$; $y_j' = 2y_j + (y_j' - 1)$; (method 1)
  - $x_j' = x_j$; $y_j' = y_j$; (method 2)
  - if $(x_j > 0)$ then $y_j' = y_j' + \left\lceil \frac{(y_j - x_j)}{x_j} \right\rceil$; (method 3)
  - if $(x_j = 0)$ then $y_j' = y_j' + y_j$.
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)
  \[ \text{Deadline}(i) = \text{Deadline}(i) + \text{Inter-Pkt Gap}(i) \]
- 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
    - \[ \text{Deadline}(j) = \text{Deadline}(j) + \text{Inter-Pkt Gap}(j) \]
Video Server

MPEG-1 active stream

Exponential idle time

VIDEO SERVER

Rich West (1999)
Fair Scheduling: $W=1,2$
$L=2/3,1/3$

(b)

Bandwidth (bps)

Time (mS)

DWCS (s1) & SFQ (s2)

DWCS (s1) & SFQ (s1)
Fair Scheduling: $W=1,1,2,4$
$L=7/8,14/16,6/8,4/8$

Rich West (1999)
Mixed Traffic: $L_1=1/3, L_2=2/3, L_3=0/100, D_1=1, D_2=1, D_3=\infty$
Mixed Traffic: $L_1=\frac{1}{3}, L_2=\frac{2}{3}, L_3=\frac{0}{1500}, D_1=1, D_2=1, D_3=\infty$

(b)
Mixed Traffic: L1=1/3, L2=2/3, L3=0/100, D1=1, D2=1, D3=∞
Mixed Traffic: L1=1/3, L2=2/3, L3=0/1500, D1=1, D2=1, D3=∞
Missed Deadlines: $W=1,2 \ L=2/3,1/3$

![Graph showing missed deadlines over time for different systems](image-url)
Mean Packet Delay: $W=1,2$
$L=2/3,1/3$

(b)

Delay (mS) vs. Number of Packets Sent

- DWCS (s1)
- DWCS (s2)
- SFQ (s1)
- SFQ (s2)

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.
- DWCS can perform fair scheduling, SP, and EDF scheduling. (*It can be unfair when necessary.*)
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).