Design and Analysis of Real-Time Systems
Bounding the Cache-Related Preemption Delay

Jan Reineke

July 25, 2013
Outline

1. Why Preemptive Scheduling?

2. CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3. Summary
Outline

1 Why Preemptive Scheduling?

2 CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3 Summary
Why use preemptive scheduling?

- Preemption often increases schedulability of task sets.
- Tasks with short deadlines are often unschedulable non-preemptively.

Example

Given: Two periodic tasks $T_1$ and $T_2$, with periods $P_1 = 2$, $P_2 = 8$, deadlines $D_1 = P_1$, $D_2 = P_2$, and execution times $C_1 = 1$, $C_2 = 3$. 

![Diagram showing periodic tasks $T_1$ and $T_2$ with preemption delays.]
Why use preemptive scheduling?

- Preemption often increases schedulability of task sets.
- Tasks with short deadlines are often unschedulable non-preemptively.

Example

Given: Two periodic tasks $T_1$ and $T_2$, with periods $P_1 = 2$, $P_2 = 8$, deadlines $D_1 = P_1$, $D_2 = P_2$, and execution times $C_1 = 1$, $C_2 = 3$. 
Preemption does not come for free!

- The preemtion task “disturbs” the state of performance-enhancing features like caches and pipelines.
- Once the preempted task resumes its execution, the disturbance may cause additional cache misses.
- The additional execution time due to additional cache misses is known as the cache-related preemption delay.

![Diagram showing task activation and preemption delay]

\[ T_1 \uparrow \quad \uparrow \quad \uparrow \quad \uparrow \]

\[ T_2 \uparrow \quad \quad \quad \quad \quad \quad \quad \quad \]

\[ = \text{CRPD} \]

\[ \uparrow = \text{Task Activation} \]
How to take preemption cost into account?

Where to account for preemption cost?

- Integrate into WCET Analysis: [Schneider, 2000]
  - Assume cache misses everywhere
  - Very pessimistic but easy to use with existing schedulability analyses

- WCET Analysis + CRPD Analysis: [Lee et al., 1996]
  - $\text{WCET}_{\text{bound}} + n \cdot \text{CRPD}_{\text{bound}} \geq$ execution time of task with up to $n$ preemptions
  - More precise but requires new schedulability analyses: except for some recent work by Davis, Altmeyer, and Maiza.
How to take preemption cost into account?

Where to account for preemption cost?

- Integrate into WCET Analysis: [Schneider, 2000]
  - Assume cache misses everywhere
  - Very pessimistic but easy to use with existing schedulability analyses

- WCET Analysis + CRPD Analysis: [Lee et al., 1996]
  - $WCET_{bound} + n \cdot CRPD_{bound} \geq$ execution time of task with up to $n$ preemptions
  - More precise but requires new schedulability analyses: except for some recent work by Davis, Altmeyer, and Maiza.

Focus of this lecture: approaches to bound the CRPD
Out-of-scope: using these bounds within schedulability analyses
Outline

1 Why Preemptive Scheduling?

2 CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3 Summary
CRPD Analyses

- Preempted Task:
  How many additional cache misses can a single preemption by *any* preemitting task cause in a *given* preempted task?

- Preempting Task:
  How many additional cache misses can a single preemption by a *given* preemitting task cause in *any* preempted task?

- Preempted + Preemting Task
  How many additional cache misses can a single preemption by a *given* preemitting task cause in a *given* preempted task?
Outline

1. Why Preemptive Scheduling?

2. CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3. Summary
Analysis of the Preempted Task: Useful Cache Blocks (UCB)

Definition (Useful Cache Block, [Lee et al., 1996])

A memory block $m$ at program point $P$ is called a useful cache block, if

a) $m$ may be cached at $P$

b) $m$ may be reused at program point $Q$ that may be reached from $P$ with no eviction of $m$ on this path.

$\times$ = hit
$\circ$ = miss

UCB $\supseteq \{A, B, C, D\}$
Definition (Useful Cache Block, [Lee et al., 1996])

A memory block $m$ at program point $P$ is called a useful cache block, if

a) $m$ may be cached at $P$

b) $m$ may be reused at program point $Q$ that may be reached from $P$ with no eviction of $m$ on this path.

$\times =$ hit  
$\circ =$ miss

UCB = \{A, B, C\}
UCB Analysis

What may be cached?
What may be reused?
Forward May-Analysis!
Backward May-Analysis!

minimal distance \leq \text{associativity}?

Program point P

minimal distance

minimal distance ≤ associativity?
UCB Analysis

Combination of two LRU-may-analyses:

What may be cached?
Forward May-Analysis!

What may be reused?
Backward May-Analysis!

Minimal age + Minimal distance to reuse \( \leq \) associativity

\[ \implies \text{Memory block may be useful} \]
Improvement: Path Analysis

Some blocks are never useful at the same time:

\[ \text{associativity} \leq y \leq \text{associativity} \]

Literature:
[Tomiyama and Dutt, 2000, Negi et al., 2003, Staschulat and Ernst, 2007]
Improvement: Avoid Accumulating Overestimations

Schedulability analyses rely on:

\[ WCET_{\text{bound}} + n \cdot CRPD_{\text{bound}} \geq \text{exec. time with up to } n \text{ preemptions} \]

# of possible preemptions

Yet, we usually have:

\[ WCET_{\text{bound}} \geq \text{execution time without preemptions} \]
\[ CRPD_{\text{bound}} \geq \text{additional execution time due to one preemption} \]

\[ \implies \text{Overestimation in both analyses adds up:} \]
\[ \text{Some cache misses are counted twice!} \]
Bounding the CRPD using UCBs for Fully-Associative Caches

- CRPD bound at program point $P$:
  \[
  \text{CRPD}_{\text{UCB}}^{\text{LRU}}(P) = \text{BRT} \cdot \min(|\text{UCB}(P)|, k),
  \]
  where $k =$ associativity and $\text{BRT} =$ Block Reload Time

- CRPD bound independent of program point:
  \[
  \text{CRPD}_{\text{UCB}}^{\text{LRU}} = \max_P \text{CRPD}_{\text{UCB}}^{\text{LRU}}(P)
  \]

Slightly more complicated for set-associative caches: sum up bounds of all cache sets.
Outline

1 Why Preemptive Scheduling?

2 CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3 Summary
Analysis of the Preempting Task: Evicting Cache Blocks

Definition (Evicting Cache Blocks (ECB), [Tomiyama and Dutt, 2000])

A memory block of the preempting task is called an *evicting cache block*, if it may be accessed during the execution of the preempting task.

Cache State: \([A, B, C, D]\)  
\[\rightarrow\]  
Cache State: \([X, Y, Z, D]\)

- \(X = \text{hit}\)
- \(Y, Z = \text{miss}\)
- \(\bullet = \text{additional miss due to preemption (CRPD)}\)

Bounding the Preemption Delay

July 25, 2013 17 / 36
Analysis of the Preempting Task: Evicting Cache Blocks

Definition (Evicting Cache Blocks (ECB), [Tomiyama and Dutt, 2000])

A memory block of the preempting task is called an *evicting cache block*, if it may be accessed during the execution of the preempting task.

Cache State: \([A, B, C, D]\)

Cache State: \([X, Y, Z, D]\)

\(\times = \text{hit}\)

\(\bigcirc = \text{miss}\)

\(\bullet = \text{additional miss due to preemption (CRPD)}\)

\[\text{CRPD}_{ECB}^{LRU} \leq \text{BRT} \cdot \min(|\text{ECB}|, k)\]

\(k = \text{associativity}\)

\(\text{BRT} = \text{Block Reload Time}\)
CRPD Computation for LRU using ECBs: Pitfall

\[ [b, a, 9, 8] \xrightarrow{8} [8, b, a, 9] \xrightarrow{9} [9, 8, b, a] \xrightarrow{a} [a, 9, 8, b] \xrightarrow{b} [b, a, 9, 8] \quad 0 \text{ misses} \]
CRPD Computation for LRU using ECBs: Pitfall

ECBs = \{ \text{e} \}

\[ [b, a, 9, 8] \xrightarrow{8} [8, b, a, 9] \xrightarrow{9} [9, 8, b, a] \xrightarrow{a} [a, 9, 8, b] \xrightarrow{b} [b, a, 9, 8] \quad 0 \text{ misses} \]

\[ [e, b, a, 9] \xrightarrow{8^*} [8, e, b, a] \xrightarrow{9^*} [9, 8, e, b] \xrightarrow{a^*} [a, 9, 8, e] \xrightarrow{b^*} [b, a, 9, 8] \quad 4 \text{ misses} \]

- \(|UCB| = 4\)
- \(|ECB| = 1\)
- \(k = \text{associativity} = 4\)
- number of additional misses = 4
CRPD Computation for LRU using ECB: Sound but Imprecise

- ECB analysis only to determine whether the set is used at all by the preemting task or not:

\[
\text{CRPD}_{\text{ECB}}^{\text{LRU}} = \begin{cases} 
0 & \text{if } \text{ECB} = \emptyset \\
\text{BRT} \cdot k & \text{otherwise}
\end{cases}
\]

- Cannot do better than that without knowledge of preempted task.
Outline

1 Why Preemptive Scheduling?

2 CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3 Summary
Analysis of Preempted and Preempting Task: “Shallow” Combination

Take the minimum of the $UCB$- and $ECB$-based estimations.

- CRPD bound for entire program:

$$\text{CRPD}_{\text{LRU}}^{\text{UCB+ECB}} = \min(\text{CRPD}_{\text{LRU}}^{\text{ECB}}, \text{CRPD}_{\text{LRU}}^{\text{UCB}})$$

Becomes slightly more complicated for set-associative caches:
For a program point: sum of point-wise minima of all cache sets.

Literature:

- For direct-mapped caches: [Negi et al., 2003]
- For set-associative caches:
  [Tan and Mooney, 2004, Burguière et al., 2009]
Analysis of Preempted and Preempting Task: “Deeper” Combination

Without preemption:
\[[a, 9, 8, 7] \xrightarrow{8} [8, a, 9, 7] \xrightarrow{9} [9, 8, a, 7] \xrightarrow{a} [a, 9, 8, 7]\]

With preemption:
\[[e, a, 9, 8] \xrightarrow{8} [8, e, b, a] \xrightarrow{9} [9, 8, e, b] \xrightarrow{a} [a, 9, 8, e]\]

Some of the UCBs are guaranteed to remain useful under preemption!

ECBs = \{e\}
Analysis of Preempted and Preempting Task: “Deeper” Combination

Without preemption:

\[
\begin{align*}
[a, 9, 8, 7] & \xrightarrow{8} [8, a, 9, 7] \xrightarrow{9} [9, 8, a, 7] \xrightarrow{a} [a, 9, 8, 7]
\end{align*}
\]

With preemption:

\[
\begin{align*}
[e, a, 9, 8] & \xrightarrow{8} [8, e, b, a] \xrightarrow{9} [9, 8, e, b] \xrightarrow{a} [a, 9, 8, e]
\end{align*}
\]

Some of the UCBs are guaranteed to remain useful under preemption!

- \( \text{CRPD}_{\text{UCB}&\text{ECB}} = \min(\text{CRPD}_{\text{UCB}}, \text{CRPD}_{\text{ECB}}) = \min(3, 4) = 3 \)
- Yet: actual number of additional misses: 0
Analysis of Preempted and Preempting Task: “Deeper” Combination

Without preemption:

\[
[a, 9, 8, 7] \xrightarrow{8} [8, a, 9, 7] \xrightarrow{9} [9, 8, a, 7] \xrightarrow{a} [a, 9, 8, 7]
\]

With preemption:

\[
[e, a, 9, 8] \xrightarrow{8} [8, e, b, a] \xrightarrow{9} [9, 8, e, b] \xrightarrow{a} [a, 9, 8, e]
\]

Some of the UCBs are guaranteed to remain useful under preemption!

- \( \text{CRPD}_{\text{UCB}\&\text{ECB}} = \min(\text{CRPD}_{\text{UCB}}, \text{CRPD}_{\text{ECB}}) = \min(3, 4) = 3 \)
- Yet: actual number of additional misses: 0

Why?
Analysis of Preempted and Preempting Task: “Deeper” Combination

Without preemption:

\[ [a, 9, 8, 7] \xrightarrow{8} [8, a, 9, 7] \xrightarrow{9} [9, 8, a, 7] \xrightarrow{a} [a, 9, 8, 7] \]

With preemption:

\[ [e, a, 9, 8] \xrightarrow{8} [8, e, b, a] \xrightarrow{9} [9, 8, e, b] \xrightarrow{a} [a, 9, 8, e] \]

Some of the UCBs are guaranteed to remain useful under preemption!

- \( \text{CRPD}_{UCB\&ECB} = \min(\text{CRPD}_{UCB}, \text{CRPD}_{ECB}) = \min(3, 4) = 3 \)
- Yet: actual number of additional misses: 0

Why?

- Minimal number of ECBs to evict a UCB is 2, but \(|ECB| = 1\)
- A single ECB is not sufficient to evict any of the UCBs.
Combining UCB and ECB: Notion of Resilience

Determining the maximal number of ECBs, such that no additional cache miss may occur:

\[ m \in UCB \]

\[
\begin{align*}
&\uparrow m \\
&\downarrow a_3 \\
&\quad \downarrow a_2 \\
&\quad \quad \downarrow a_1 \\
\end{align*}
\]

\[
\begin{align*}
&m [m, -, -, -, -, -, -, -, -] \\
a_3 [a_3, a_2, a_1, m, -, -, -, -] \\
a_2 [m, a_3, a_2, a_1, -, -, -, -] \\
a_1 [m, a_3, a_2, a_1, -, -, -, -] \\
\end{align*}
\]
Combining UCB and ECB: Notion of Resilience

Determining the maximal number of ECBs, such that no additional cache miss may occur:

\[ m \in UCB \]

\[ m \text{ is } 4\text{-resilient} \]

\[
\begin{align*}
\text{m} & \quad [m, -,-,-,-,-,-,-,-] \\
\text{a_1} & \quad [a_3, a_2, a_1, m, -,-,-,-] \\
\text{a_2} & \quad [m, a_3, a_2, a_1, -,-,-,-] \\
\text{a_3} & \quad [m, a_3, a_2, a_1, -,-,-,-] \\
\end{align*}
\]
Definition (Resilience)

A memory block \( m \) is called \( l \)-resilient at program point \( P \), if all possible next accesses to \( m \)
- that would be hits without preemption,
- would still be hits in case of a preemption at \( P \) with \( l \) accesses.
A memory block $m$ is called $l$-resilient at program point $P$, if all possible next accesses to $m$

- that would be hits without preemption,
- would still be hits in case of a preemption at $P$ with $l$ accesses.

- No UCB is $k$-resilient, i.e., no UCB remains useful after a preemption with $k$ (= associativity) many ECBs.
- Each $(l + 1)$-resilient UCB is also $l$-resilient.
- Each UCB is at least 0-resilient.
Resilience Analysis

Definition (Resilience)

A memory block \( m \) is called \( l \)-resilient at program point \( P \), if all possible next accesses to \( m \)
- that would be hits without preemption,
- would still be hits in case of a preemption at \( P \) with \( l \) accesses.

\( m \in UCB \)
m is 4-resilient

\( ECB = \{e_1, e_2, e_3, e_4\} \)

Jan Reineke
Bounding the Preemption Delay
July 25, 2013 24 / 36
A memory block $m$ is called $l$-resilient at program point $P$, if all possible next accesses to $m$
- that would be hits without preemption,
- would still be hits in case of a preemption at $P$ with $l$ accesses.

In general: if $|ECB| \leq l$ then the UCB is not evicted.
Resilience Analysis

Definition (Resilience)

A memory block $m$ is called $l$-resilient at program point $P$, if all possible next accesses to $m$

- that would be hits without preemption,
- would still be hits in case of a preemption at $P$ with $l$ accesses.

Jan Reineke
Bounding the Preemption Delay
July 25, 2013
Bounding the CRPD using Resilience

CRPD (Combining UCB and ECB by using Resilience)

\[ \text{blocks contributing to CRPD} \]
\[
\begin{align*}
\text{UCB} \setminus \left\{ m \mid m \text{ is ECB}-\text{resilient} \right\} \\
\text{useful} \quad \text{remain useful}
\end{align*}
\]
Bounding the CRPD using Resilience

**CRPD (Combining UCB and ECB by using Resilience)**

\[
\text{CRPD} \leq BRT \times \left| \text{UCB} \setminus \left\{ \left| \text{m is ECB}-\text{resilient} \right| \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \text{UCB} \setminus \left\{ \left| \text{m is ECB}-\text{resilient} \right| \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \text{UCB} \setminus \left\{ \left| \text{m is ECB}-\text{resilient} \right| \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]

\[
\left| \left\{ \text{m is ECB}-\text{resilient} \right\} \right| \text{blocks contributing to CRPD}
\]
Bounding the CRPD using Resilience: Example

ECBs = \{e\}

\[ (c, b, a, x) \xrightarrow{a} [a, c, b, x] \xrightarrow{b} [b, a, c, x] \xrightarrow{c} [c, b, a, x] \] no misses

\[ (e, c, b, a) \xrightarrow{a} [a, e, c, b] \xrightarrow{b} [b, a, e, c] \xrightarrow{c} [c, b, a, e] \] no misses
Bounding the CRPD using Resilience: Example

- $|\text{ECB}| = 1$
- $a$, $b$, and $c$ are 1-resilient
- $\text{CRPD}_{\text{UCB}&\text{ECB}}^{\text{res}} = BRT \times |\text{UCB} \setminus \{m \mid m \text{ is } |\text{ECB}|\text{-resilient}\}| = 0$
Bounding the CRPD using Resilience:
Example

ECBs
= \{e\}

\[
\begin{align*}
[a, c, b, x] & \xrightarrow{a} [a, c, b, x] \xrightarrow{b} [b, a, c, x] \xrightarrow{c} [c, b, a, x] & \text{no misses} \\
[e, c, b, a] & \xrightarrow{a} [a, e, c, b] \xrightarrow{b} [b, a, e, c] \xrightarrow{c} [c, b, a, e] & \text{no misses}
\end{align*}
\]

- $|\text{ECB}| = 1$
- $a$, $b$, and $c$ are 1-resilient
- $CRPD^\text{res}_{\text{UCB}\&\text{ECB}} = BRT \times |\text{UCB} \setminus \{m \mid m \text{ is } |\text{ECB}|\text{-resilient}\}| = 0$
- Instead of: $CRPD_{\text{UCB}\&\text{ECB}} = \min(CRPD_{\text{UCB}}, CRPD_{\text{ECB}}) = 3 \times BRT$
Bounding the CRPD using Resilience: Example

- $\text{ECBs} = \{e\}$
- $[c, b, a, x] \xrightarrow{a} [a, c, b, x] \xrightarrow{b} [b, a, c, x] \xrightarrow{c} [c, b, a, x]$ no misses
- $[e, c, b, a] \xrightarrow{a} [a, e, c, b] \xrightarrow{b} [b, a, e, c] \xrightarrow{c} [c, b, a, e]$ no misses

- $|\text{ECB}| = 1$
- $a, b, \text{ and } c$ are 1-resilient
- $\text{CRPD}_{\text{res}} = \text{BRT} \times |\text{UCB} \setminus \{m \mid m \text{ is } |\text{ECB}|\text{-resilient}\}| = 0$
- Instead of: $\text{CRPD}_{\text{UCB} \& \text{ECB}} = \min(\text{CRPD}_{\text{UCB}}, \text{CRPD}_{\text{ECB}}) = 3 \times \text{BRT}$
Outline

1. Why Preemptive Scheduling?

2. CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3. Summary
Deferred Preemption

- Restrict preemptions to a set of predefined preemption points.
- Introduces new problem: blocking time, i.e., time until next preemption point is reached.

Intervals between preemption points ≡ critical sections.

\[
T_1 \quad \uparrow \quad BT \quad \uparrow \quad BT \\
\quad \uparrow \quad T_2\\n\]

Context Switch Costs
↑ Task Activation
* Preemption Point

Where to place preemptions points, s.t.
- CRPD is minimized, and
- Maximum Blocking Time is minimized?

Analysis to determine maximum blocking time for given set of preemption points: [Lee et al., 1998, Altmeyer et al., 2009]
Cache Partitioning

Additional cache misses are due to interference on the cache.

⇒ Cache Partitioning eliminates this interference.

![Diagram of cache partitioning](image)

- **Software-based Cache Partitioning** [Wolfe, 1994, Mueller, 1995]:
  - Change layout of instructions and data such that tasks map to disjoint cache sets
  - Particularly difficult for large arrays

- **Hardware-based Cache Partitioning** [Kirk and Strosnider, 1990, Chiou, 1999]:
  - Partition cache by cache sets and/or cache ways
  - Increases hardware cost
  - Renewed interest in multi-cores with shared caches
Outline

1 Why Preemptive Scheduling?

2 CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3 Summary
Do existing approaches work for FIFO, PLRU, etc.?

Plain answer: No!
Do existing approaches work for FIFO, PLRU, etc.?

Plain answer: No!

Counterexample for FIFO [Burguière et al., 2009]:

$$\begin{align*}
\text{ECBs} = \{x\} \\
[b, a] &\xrightarrow{a} [b, a] \xrightarrow{e^*} [e, b] \xrightarrow{b} [e, b] \xrightarrow{c^*} [c, e] \xrightarrow{e} [c, e] \quad 2 \text{ misses} \\
[x, b] &\xrightarrow{a^*} [a, x] \xrightarrow{e^*} [e, a] \xrightarrow{b^*} [b, e] \xrightarrow{c^*} [c, b] \xrightarrow{e^*} [e, c] \quad 5 \text{ misses}
\end{align*}$$

- $|\text{UCB}(s)| = 2$
- $|\text{ECB}(s)| = 1$
- associativity $k = 2$
- But: number of additional misses $= 3$

Same result for PLRU.
Idea [Burguìère et al., 2009]:
Use Relative Competitiveness Results

Some relative competitiveness results:

- PLRU(n) is (1, 0)-miss-competitive relative to LRU(1 + log₂n).
- FIFO(n) is \(\left(\frac{n}{n-r+1}, r\right)\)-miss-competitive relative to LRU(r).

⇒ Performing WCET and CRPD analyses assuming LRU(1 + log₂n) replacement should give correct bounds for PLRU(n).

Can we also make use of non-(1, 0)-competitiveness?
Applying Relative Competitiveness: A sequence of memory accesses

- Notation:
  - $m =$ number of misses
  - $\bar{m} =$ number of misses in the case of preemption
Applying Relative Competitiveness: A sequence of memory accesses

- Notation:
  - $m = \text{number of misses}$
  - $\bar{m} = \text{number of misses in the case of preemption}$

\[ m_{\text{pre}} = 4 \quad m_{\text{post}} = 2 \]

\[ \bar{m}_{\text{pre}} = m_{\text{pre}} = 4 \quad \bar{m}_{\text{post}} = m_{\text{post}} + m_{\text{CRPD}} = 5 \]

- Assume $P(t)$ is $(k, c)$-miss-competitive rel. to LRU$(s)$. Then:

\[ \bar{m}^{P(t)} = \bar{m}^{P(t)}_{\text{pre}} + \bar{m}^{P(t)}_{\text{post}} \]
Applying Relative Competitiveness: A sequence of memory accesses

- Notation:
  - $m = \text{number of misses}$
  - $\bar{m} = \text{number of misses in the case of preemption}$

![Diagram showing memory accesses and preemption]

- Assume $P(t)$ is $(k, c)$-miss-competitive rel. to LRU(s). Then:

$$
\bar{m}^{P(t)} = \bar{m}_{pre}^{P(t)} + \bar{m}_{post}^{P(t)} \\
\leq [k \cdot m_{pre}^{LRU(s)} + c] + [k \cdot (m_{post}^{LRU(s)} + m_{CRPD}^{LRU(s)}) + c]
$$
Applying Relative Competitiveness: A sequence of memory accesses

- Notation:
  - $m = \text{number of misses}$
  - $\overline{m} = \text{number of misses in the case of preemption}$

\[
\begin{align*}
\overline{m}_{\text{pre}} &= 4 \\
\overline{m}_{\text{post}} &= 2 \\
\overline{m}_{\text{pre}} &= \overline{m}_{\text{pre}} = 4 \\
\overline{m}_{\text{post}} &= \overline{m}_{\text{post}} + m_{\text{CRPD}} = 5
\end{align*}
\]

- Assume $P(t)$ is $(k, c)$-miss-competitive rel. to LRU($s$). Then:

\[
\begin{align*}
\overline{m}^{P(t)} &= \overline{m}_{\text{pre}}^{P(t)} + \overline{m}_{\text{post}}^{P(t)} \\
&\leq [k \cdot m_{\text{pre}}^{\text{LRU}(s)} + c] + [k \cdot (m_{\text{post}}^{\text{LRU}(s)} + m_{\text{CRPD}}^{\text{LRU}(s)}) + c] \\
&= [k \cdot m^{\text{LRU}(s)} + c] + [k \cdot m_{\text{CRPD}}^{\text{LRU}(s)} + c]
\end{align*}
\]
Applying Relative Competitiveness

- Assume \( P(t) \) is \((k, c)\)-miss-competitive rel. to LRU(s). Then:

\[
\overline{m}^{P(t)} \leq [k \cdot m^{LRU(s)} + c] + [k \cdot m^{LRU(s)} + c]
\]

- In WCET analysis:
  Take into account \( k \cdot m^{LRU(s)} + c \) misses

- In CRPD analysis:
  Take into account \( k \cdot m^{CRPD} + c \) misses
Outline

1. Why Preemptive Scheduling?

2. CRPD Computation
   - Analysis of the Preempted Task
   - Analysis of the Preempting Task
   - Analysis of the Preempted and the Preempting Task
   - Simplifying or Eliminating the Problem
   - Policies other than LRU

3. Summary
Preemptive Scheduling desirable, but not for free:
  \[\Rightarrow\text{Need to bound CRPD}\]

For LRU, the CRPD can be bounded by analyzing
  \[\Rightarrow\text{the preempted task: UCB analysis}\]
  \[\Rightarrow\text{the preempting task: ECB analysis}\]
    \[\Rightarrow\text{Sound approach rather imprecise}\]
    \[\Rightarrow\text{Need to couple more tightly with analysis of preempted task}\]
  \[\Rightarrow\text{both, the preempted and the preempting task}\]
    \[\Rightarrow\text{“Shallow” combination}\]
    \[\Rightarrow\text{“Deeper” combination: Resilience analysis}\]

Approaches do not carry over to FIFO, PLRU, etc. immediately
  \[\Rightarrow\text{First approach: relative competitiveness}\]


Smart (strategic memory allocation for real-time) cache design using the mips r3000.


Integrated intra- and inter-task cache analysis for preemptive multi-tasking real-time systems.


In *Proceedings of the 8th ACM international workshop on Hardware/software codesign (CODES’00)*, pages 67–71, New York, NY, USA. ACM.