Timing analysis and predictability of architectures
Cache analysis

Claire Maiza
Verimag/INP

01/12/2010
Timing Analysis

Analysis-guaranteed timing bounds

Frequency

LB  BCET  WCET  UB  Exec-time

Overest.
Static Timing Analysis

Legend:
- Data
- Phase

1. Input Executable
2. CFG Reconstruction
3. Control-flow Graph
4. Value Analysis
5. Loop Bound Analysis
6. Control-flow Analysis
7. Annotated CFG
8. Micro-architectural Analysis
9. Basic Block Timing Info
10. Path Analysis
Timing anomalies

- When local worst-case does not lead to the global worst-case

Scheduling anomaly.

Speculation anomaly.
Classification of architectures

- **Timing compositional**
  - No timing anomalies
  - e.g., ARM7

- **Compositional with bounded effects**
  - Timing anomalies but no domino effects
  - e.g., TriCore (probably)

- **Non-compositional architectures**
  - Timing anomalies, domino effects
  - e.g., PPC 755

*from Wilhelm et al.: Memory Hierarchies, Pipelines, and Buses for Future Architectures in Time-critical Embedded Systems, IEEE TCAD, July 2009*
Why LRU is predictable?
LRU (Least Recently Used) “forgets” about past quickly:

FIFO (First In First Out):
Preemption does not come for free!

- The preempting 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*.

\[
T_1 \quad \uparrow \quad \uparrow \quad \uparrow \\
T_2 \quad \uparrow \\
\]

\[= CRPD\]

\[\uparrow = \text{Task Activation}\]
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 in schedulability analysis

- WCET Analysis + CRPD Analysis: [Lee et al., 1996]
  - \[ WCET_{\text{bound}} + n \cdot CRPD_{\text{bound}} \geq \]
    execution time of task with up to \( n \) preemptions
  - more precise but only supported by very few schedulability analyses
CRPD Analyses

- Preempted Task:
  How many *useful* memory blocks are in the cache?

- Preempting Task:
  How much damage can the preemption task do to the cache contents of *any* task?

- Preempted + Preempting Task
  How much damage can the preemption task do to the useful cache contents of the preempted task?
Useful Cache Block Analysis

Useful = may be cached and may be reused

Program point P

minimal distance $\leq$ associativity?

Claire Maiza
Synchron 2010

01/12/2010 10 / 18
Useful Cache Block 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

$\implies$ Memory block may be useful
Improvement: Path Analysis

Some blocks are never useful at the same time:

\[ \leq \text{associativity} \]

\[ \geq y \]

\[ \geq \text{associativity} \]

Literature:
[Tomiyama and Dutt, 2000, Negi et al., 2003, Staschulat et al., 2007]
Analysis of Preempted and Preempting Task: “Deeper” Combination [Altmeyer et al., 2010]

**Definition (Resilience)**

The resilience $\text{res}_P(m)$ of memory block $m$ at program point $P$ is the greatest $l$, such that all possible next accesses to $m$,

- **a)** that would be hits without preemption,
- **b)** would still be hits in case of a preemption with $l$ accesses at $P$. 
Analysis of Preempted and Preempting Task: “Deeper” Combination [Altmeyer et al., 2010]

Definition (Resilience)

The resilience $\text{res}_P(m)$ of memory block $m$ at program point $P$ is the greatest $l$, such that all possible next accesses to $m$,

a) that would be hits without preemption,

b) would still be hits in case of a preemption with $l$ accesses at $P$.

preempted task

preempting task

$m$ is useful
$\text{res}(m) = 4$

$m$

$[m, \_ , \_ , \_ , \_ , \_ , \_ ]$

$a_1$

$a_2$

$a_3$

$m$

$[a_3, e_4, e_3, e_2, e_1, a_2, a_1, m]$}

$[m, a_3, e_4, e_3, e_2, e_1, a_2, a_1]$

${e_1, e_2, e_3, e_4}$
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]:

\[
\text{ECBs} = \{x\} \quad [b, a] \xrightarrow{a} [b, a] \xrightarrow{e^*} [e, b] \xrightarrow{b} [e, b] \xrightarrow{c^*} [c, e] \xrightarrow{e} [c, e] \text{ 2 misses}
\]

\[
[b, a] \xrightarrow{a} [b, a] \xrightarrow{e^*} [e, b] \xrightarrow{b} [e, b] \xrightarrow{c^*} [c, e] \xrightarrow{e} [c, e] \text{ 5 misses}
\]

- 2 *useful* cache blocks
- 1 block loaded by the preemtting task
- associativity = 2
- But: number of additional misses = 3

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

Some relative competitiveness results:

- PLRU(n) is \((1, 0)\)-miss-competitive relative to LRU\((1 + \log_2 n)\).
- FIFO(n) is \(\left(\frac{n}{n-r+1}, r\right)\)-miss-competitive relative to LRU(r).

\[\implies\] Performing WCET and CRPD analyses assuming LRU\((1 + \log_2 n)\) 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
- \( \overline{m} \): number of misses in the case of preemption

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

\[
\begin{align*}
m_{\text{pre}} &= 4 \\
m_{\text{post}} &= 2
\end{align*}
\]
Applying Relative Competitiveness: A sequence of memory accesses

- **Notation:**
  - \( m \) = number of misses
  - \( \overline{m} \) = number of misses in the case of preemption

\[
\overline{m}_{pre} = 4 \quad \text{and} \quad m_{post} = 2
\]

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

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

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

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

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

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

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

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

- **Example:**
  - $m_{\text{pre}} = 4$
  - $m_{\text{post}} = 2$
  - $\overline{m}_{\text{pre}} = \overline{m}_{\text{pre}} = 4$
  - $\overline{m}_{\text{post}} = \overline{m}_{\text{post}} + m_{\text{CRPD}} = 5$

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

\[
\overline{m}^{P(t)} = \overline{m}^{P(t)}_{\text{pre}} + \overline{m}^{P(t)}_{\text{post}} \\
\leq [k \cdot m^{\text{LRU}(s)}_{\text{pre}} + c] + [k \cdot (m^{\text{LRU}(s)}_{\text{post}} + m_{\text{CRPD}}^{\text{LRU}(s)}) + c] \\
= [k \cdot m^{\text{LRU}(s)} + c] + [k \cdot m_{\text{CRPD}}^{\text{LRU}(s)} + c]
\]
Assume $P(t)$ is $(k, c)$-miss-competitive rel. to LRU$(s)$. Then:

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

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

- In CRPD analysis:
  Take into account $k \cdot m^{LRU(s)} + c$ misses
CRPD bounded using a number of reloads:
- a miss is the worst-case
- the reload cost is bounded

For LRU, the CRPD can be bounded by analyzing
- the preempted task: useful cache block analysis
- the preemption task
- both, the preempted and the preemption task
  * Resilience Analysis

Approaches do not carry over to FIFO, PLRU, etc. immediately
- First approach: relative competitiveness


Extending the reach of microprocessors: column and curious caching.
PhD thesis.
Supervisor-Arvind, and Supervisor-Rudolph, Larry.


Limited preemptible scheduling to embrace cache memory in real-time systems.


Staschulat, J. et al. (2007).

Program path analysis to bound cache-related preemption delay in preemptive 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.

Deferred Preemption - simplifying the problem

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

![Diagram showing preemption points and blocking time](image)

*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 - eliminating the problem
Additional cache misses are due to interference on the cache.
⇒ Cache Partitioning eliminates this interference.

![Diagram of cache partitioning]

**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
CRPD for PLRU: Pitfalls

- $|UCB(s)| = 4$
- $|ECB(s)| = 2$
- $n = 4$
- But: number of additional misses $= 5$
Resilience - Domain of the analysis

\[ D : \mathbb{D}_{ca} \times \mathbb{D}_{ua} \]  \hspace{1cm} (1)

with

\[ \mathbb{D}_{ca} : \mathbb{M} \rightarrow \{0, \ldots, k - 1\} \]  \hspace{1cm} (2)

and

\[ \mathbb{D}_{ua} : \mathbb{M} \rightarrow \{0, \ldots, k - 1, \infty\} \]  \hspace{1cm} (3)
Resilience - Transfer functions

\[ t_{ua} : D_{ua} \times M \rightarrow D_{ua} \]

\[
t_{ua}(ua, m) := \lambda m'. \begin{cases} 
0 & m' = m \\
ua(m') & ua(m') \geq ua(m) \\
ua(m') + 1 & ua(m') < ua(m) \land ua(m') < k - 1 \\
\infty & \text{otherwise}
\end{cases}
\] (4)

\[ t_{ca} : D_{ca} \times D_{ua} \times 2^M \times M \rightarrow D_{ca} \]

\[
t_{ca}(ca, ua, ucb, m) := \lambda m'. \begin{cases} 
0 & m' = m \lor m' \notin ucb \\
ca(m') & ca(m') \geq ua(m) \lor ca(m') = k - 1 \\
ca(m') + 1 & ca(m') < ua(m)
\end{cases}
\] (5)