Embedded System Programming

WCET Analysis (2)
(Module 39)

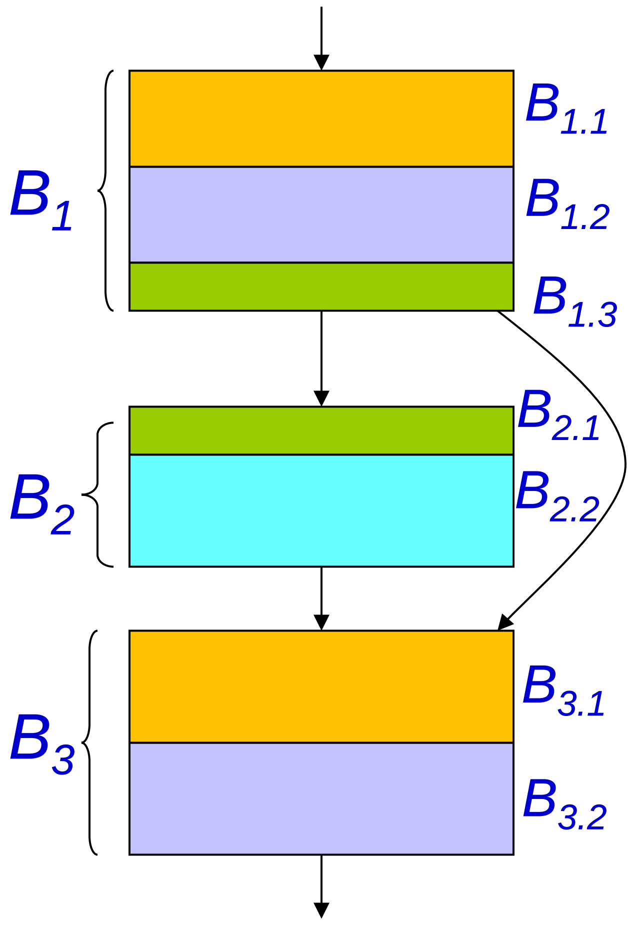
Yann-Hang Lee
Arizona State University
yhlee@asu.edu
(480) 727-7507

Summer 2014

Micro-architectural Modeling -- Cache

- **Modify cost function (cache hit and miss have different costs)**
- **Add linear constraints to describe relationship between cache hits and misses**
- **Basic idea**
  - Basic blocks assumed to be smaller than entire cache
  - Subdivide instruction counts (x_i) into counts of cache hits (x_i^{hit}) and misses (x_i^{miss})
  - Line-block (or l-block) is a contiguous sequence of code within the same basic block that is mapped to the same cache line in the instruction cache
  - Either all hit or all miss in a l-block

Basic Blocks to Line Blocks (Direct-mapped)



Color	Cache Set
🟧	0
🟪	1
🟩	2
🟦	3

Cache Constraints:

No conflicting l-blocks: $x_{2.2}^{miss} \leq 1$

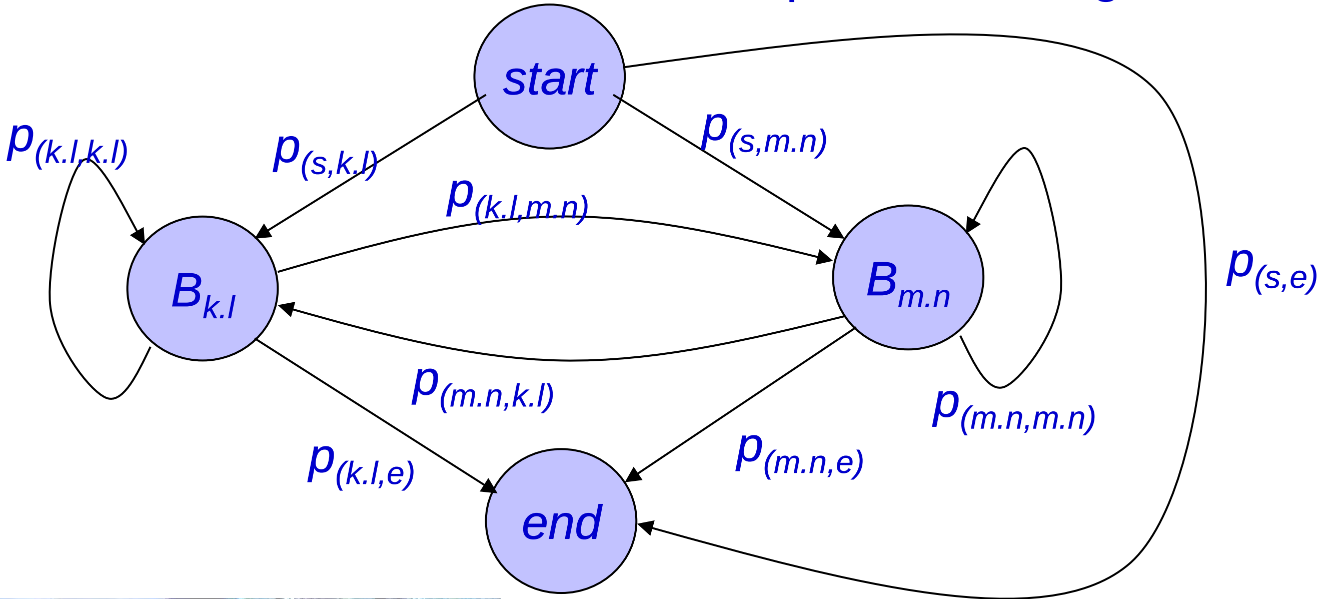
(only the first execution has a miss)

Two nonconflicting l-blocks are mapped to same cache line $x_{1.3}^{miss} + x_{2.1}^{miss} \leq 1$

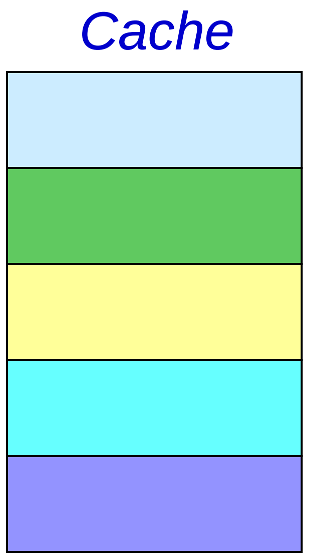
Conflicting blocks: affected by the sequence

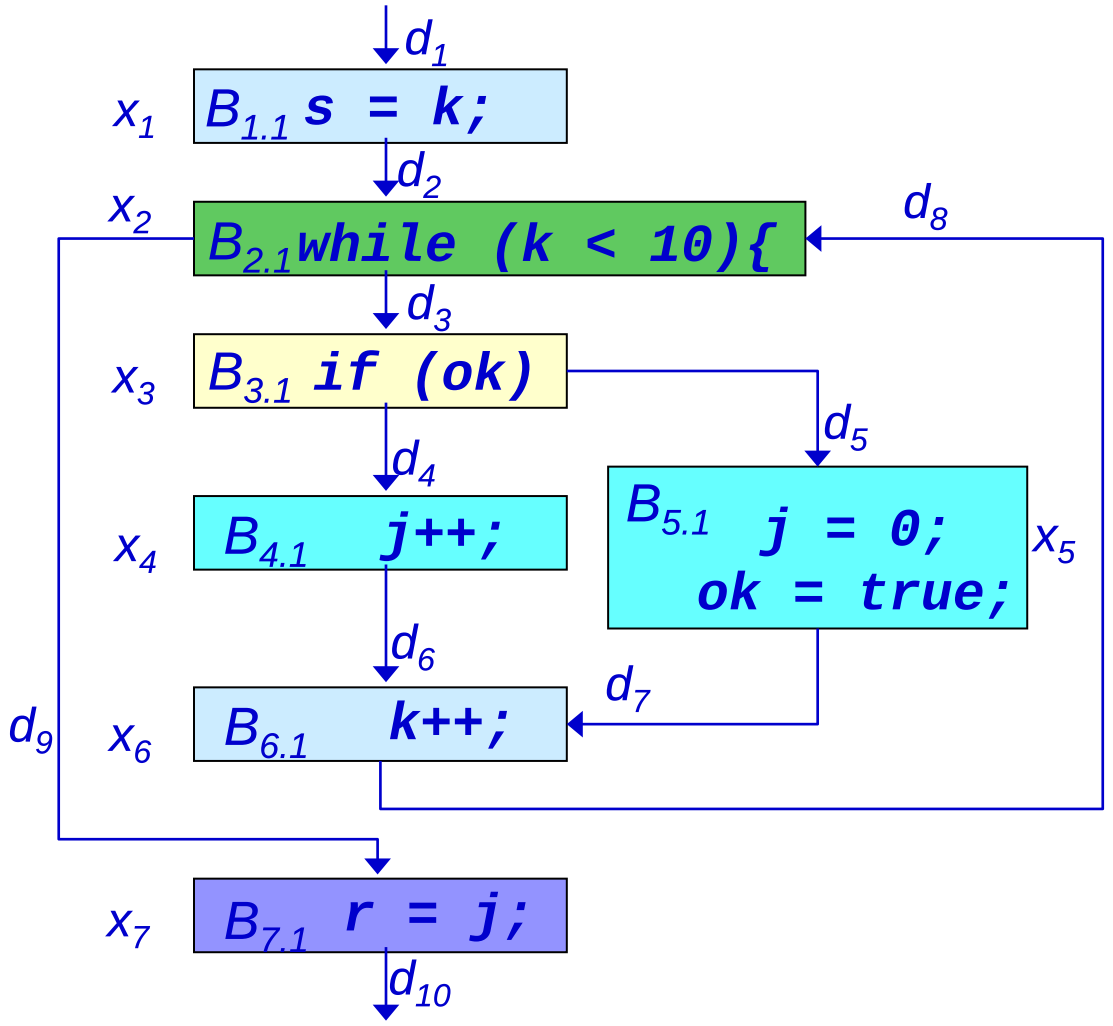
Cache Conflict Graph

- **For every cache set containing two or more conflicting l-blocks**
  - start node, end node, and node $B_{k.l}$ for every l-block in the cache set
- **Edge from $B_{k.l}$ to $B_{m.n}$: control can pass between them without passing through any other l-blocks of the same cache set.**
  - $p_{(i.j,u.v)}$: the number of times that the control passes through that edge.

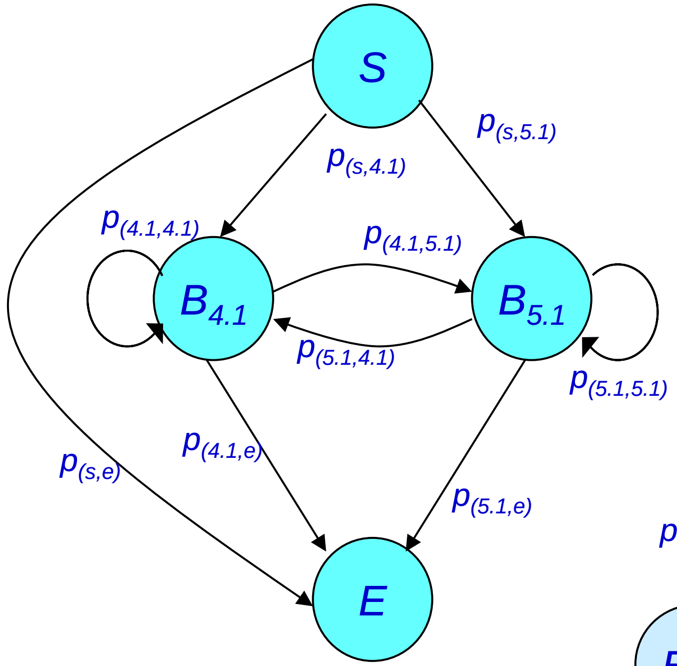


Cache Constraints Example (1)





Cache Constraints Example (2)



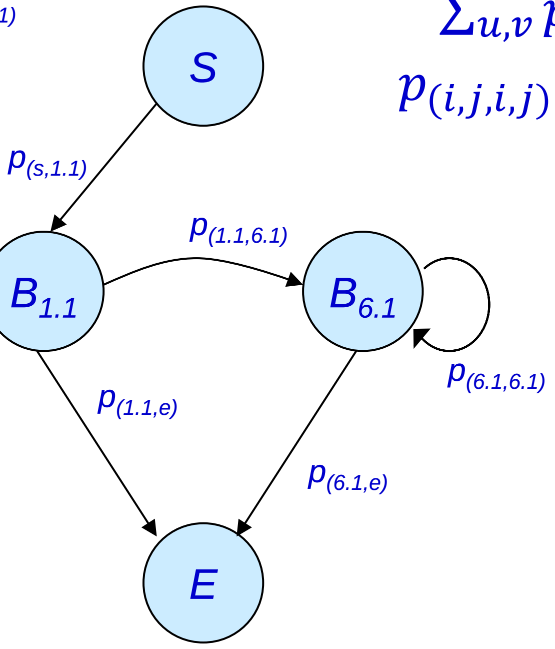
$$Total\ execution\ time = \sum_i^N \sum_j^{n_i} (c_{i,j}^{hit} x_{i,j}^{hit} + c_{i,j}^{miss} x_{i,j}^{miss})$$

$$x_i = x_{i,j}^{hit} + x_{i,j}^{miss} \qquad j = 1,2,\ldots,n_i$$

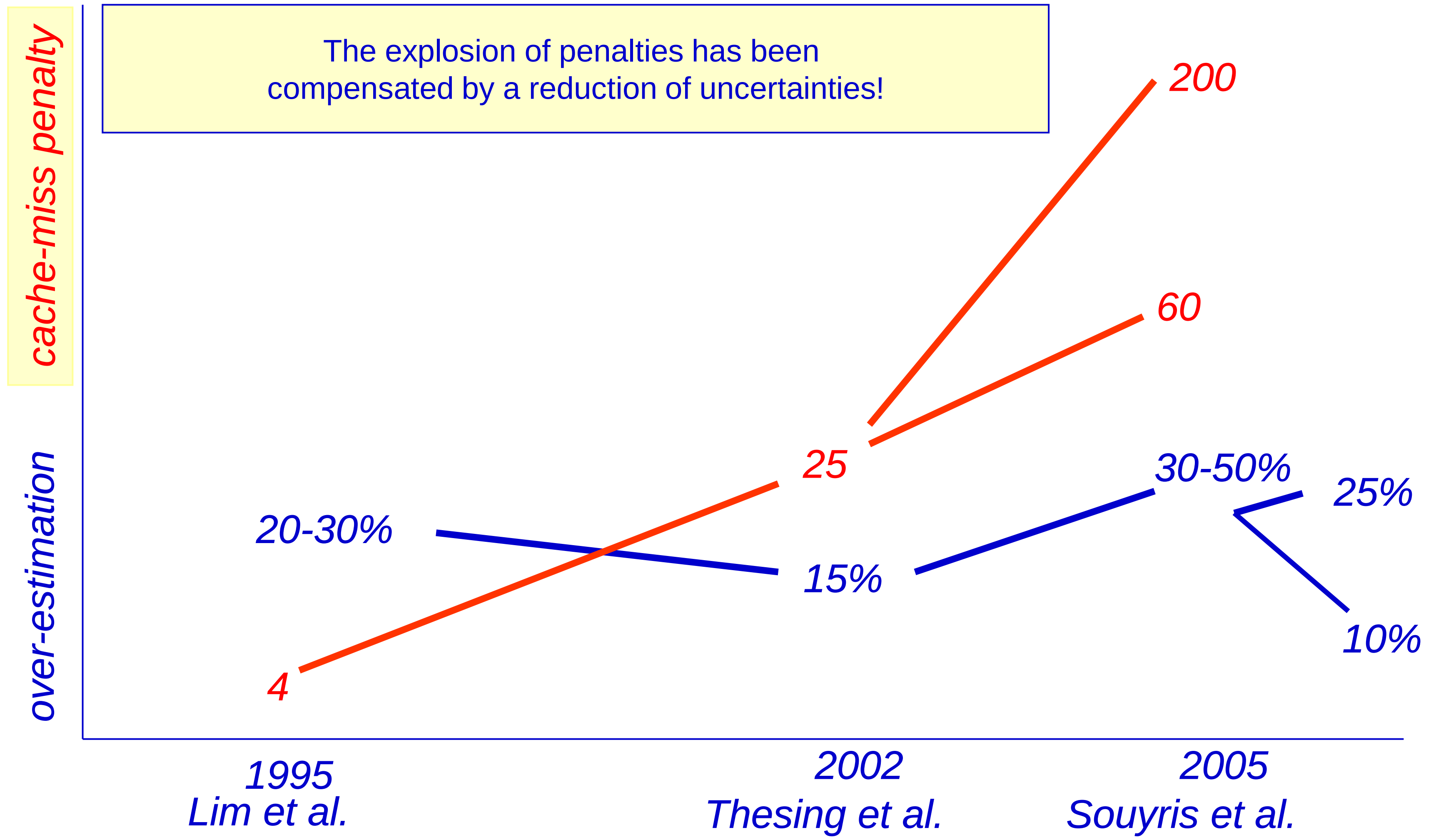
$$x_i = \sum_{u,v} p_{(u,v,i,j)} = \sum_{u,v} p_{(i,j,u,v)}$$

$$\sum_{u,v} p_{(s,u,v)} = \sum_{u,v} p_{(u,v,e)} = 1$$

$$p_{(i,j,i,j)} \le x_{i,j}^{hit} \le p_{(s,i,j)} + p_{(i,j,i,j)}$$



Progress During the Past 10 Years



Open Problems

- **Architectures are getting much more complex.**
  - Can we create processor behavior models without the pain?
  - Can we change the architecture to make timing analysis easier?
- **Small changes to code and/or architecture require completely re-doing the WCET computation**
  - Use robust techniques that learn about processor/platform behavior
- **Need more reliable ways to measure execution time**
- **References:**
  - Li, Malik, and Wolfe, “Cache Modeling for Real-Time Software: Beyond Direct Mapped Instruction Caches”
  - Wilhelm, “Determining bounds on execution times,” Handbook on Embedded Systems, CRC Press, 2005

Supplementary Slides