Sound and Efficient WCET Analysis in the Presence of Timing Anomalies

Jan Reineke[1], Rathijit Sen[2]

[1]Saarland University, Saarbrücken

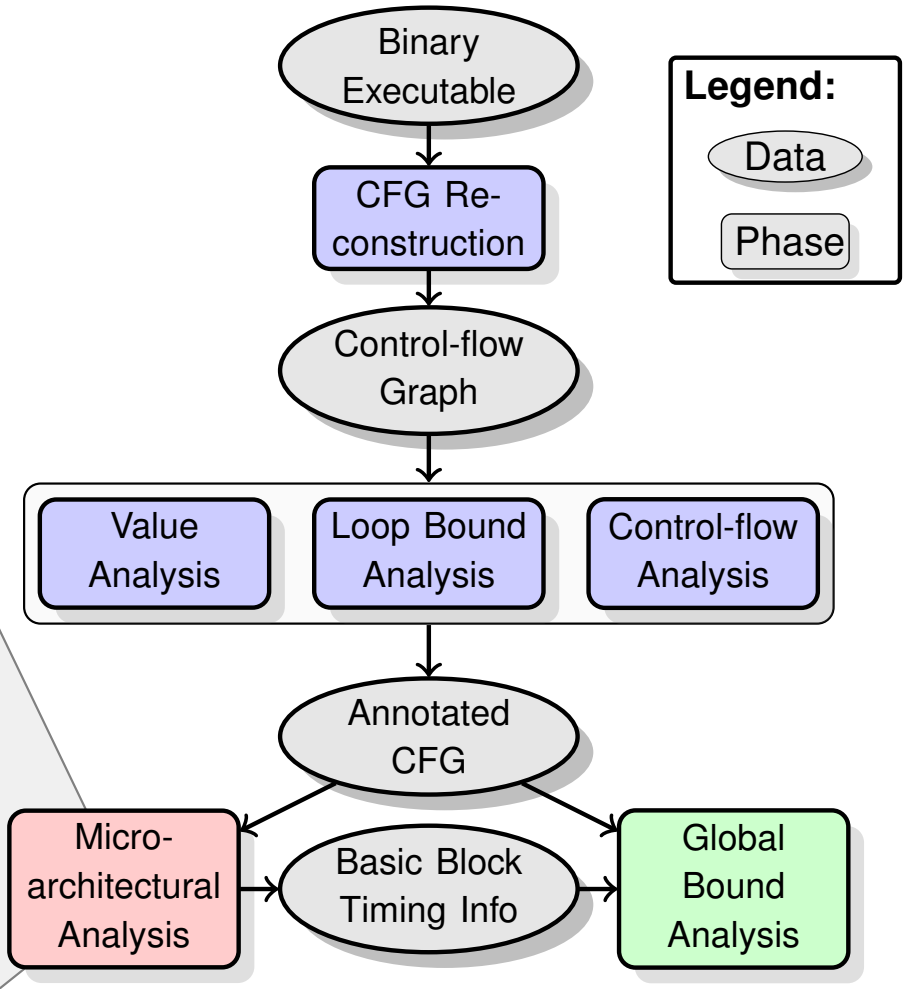
[2]University of Wisconsin, Madison

Workshop on WCET Analysis, Dublin 2009

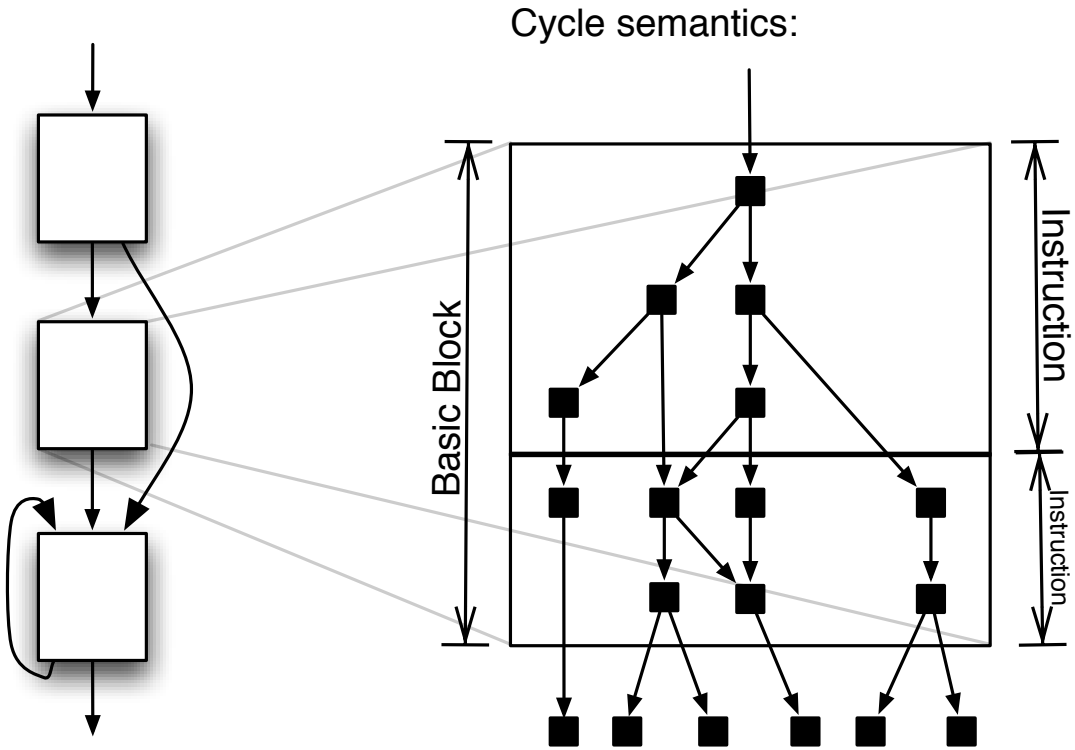
Timing Analysis Framework

SAARLAND UNIVERSITY
COMPUTER SCIENCE

- ▷ Determines bounds on execution times of basic blocks
- ▷ Based on an abstract model of the hardware
- ▷ Either *sound* or *efficient* due to *timing anomalies*
- ▷ Usually most expensive part of WCET analysis

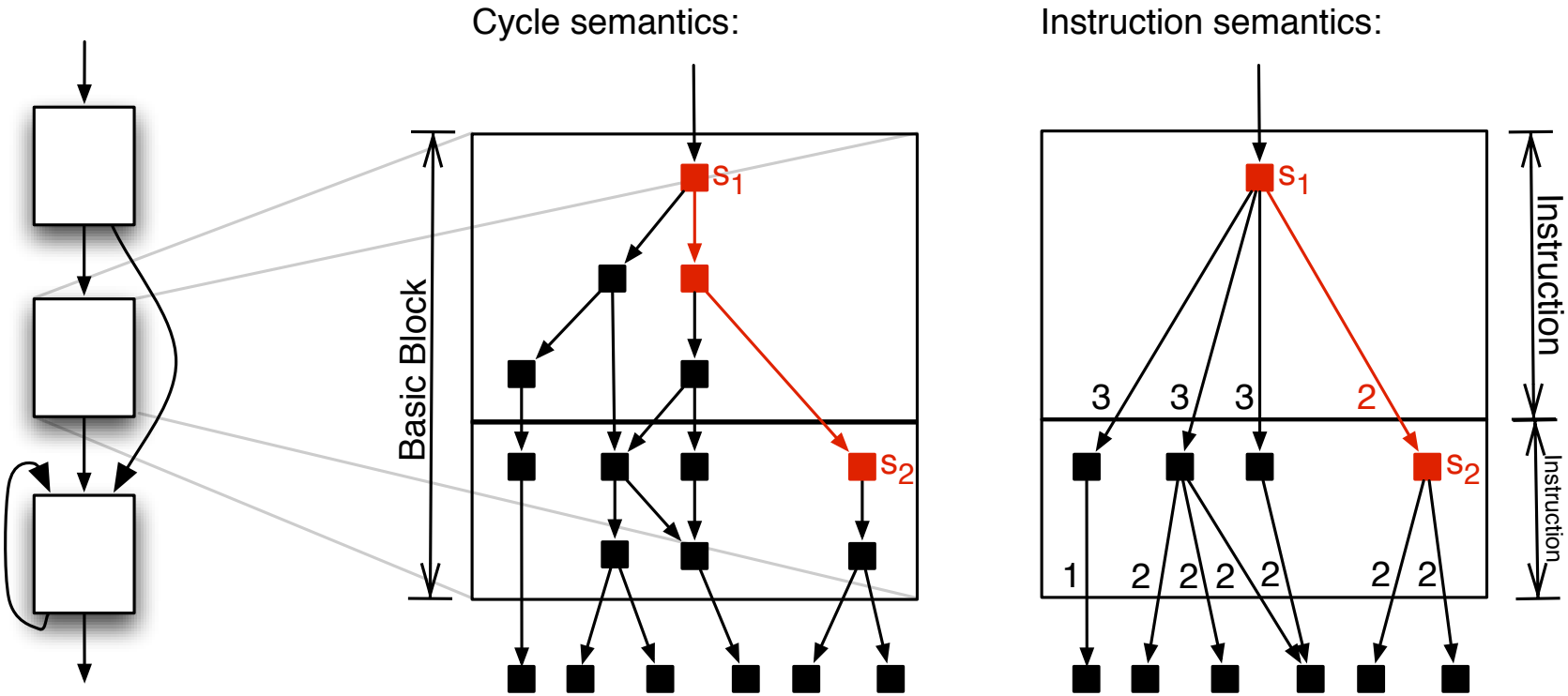


Model of Micro-Architectural Analysis



Model of Micro-Architectural Analysis

SAARLAND UNIVERSITY
COMPUTER SCIENCE



Notation:

$s \xrightarrow[\iota_0 \ldots \iota_n]{t} s'$

$\max(s, \iota_0 \ldots \iota_n) := \max\{t \mid s \xrightarrow[\iota_0 \ldots \iota_n]{t} s'\}$

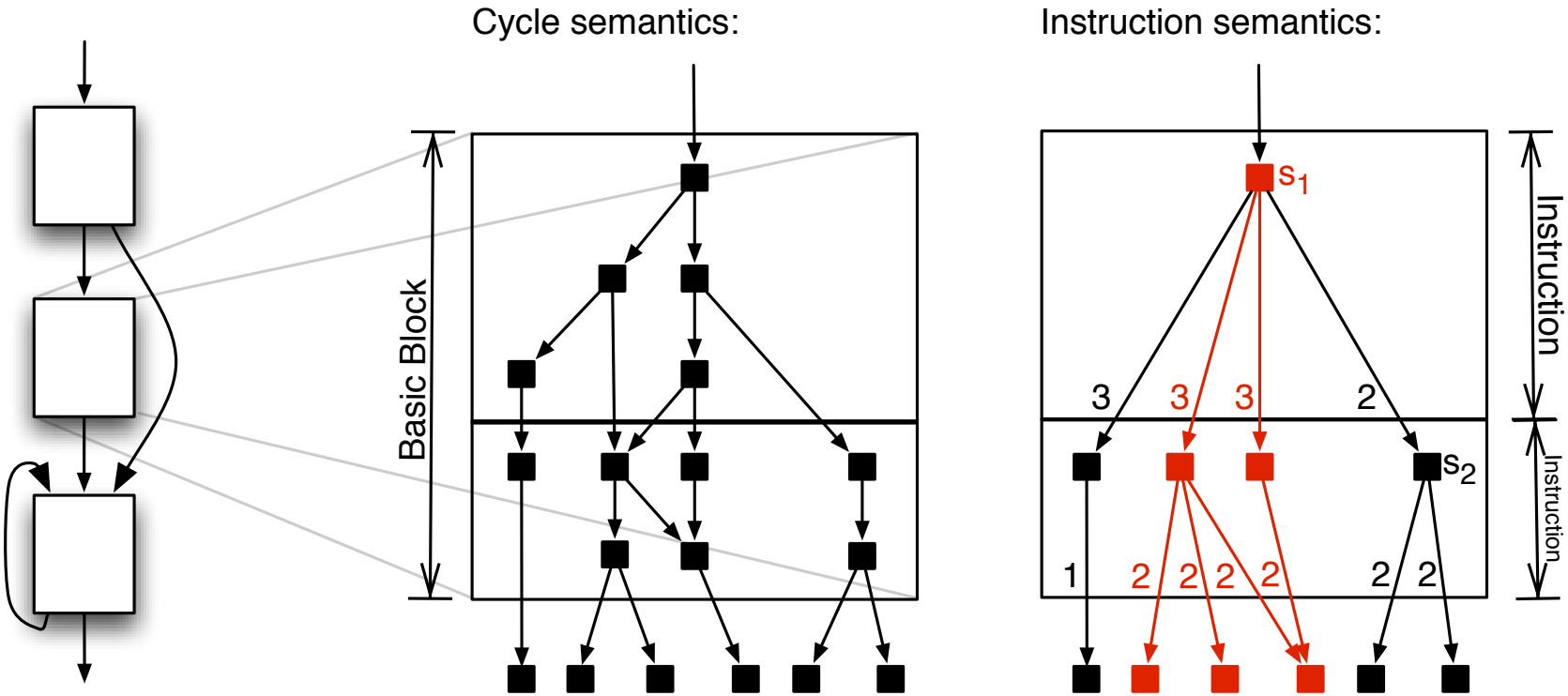
Example:

$s_1 \xrightarrow[\iota_1]{2} s_2$

$\max(s_1, \iota_1 \iota_2) = 5$

Model of Micro-Architectural Analysis

SAARLAND UNIVERSITY
COMPUTER SCIENCE



Notation:

$s \xrightarrow[\iota_0 \ldots \iota_n]{t} s'$

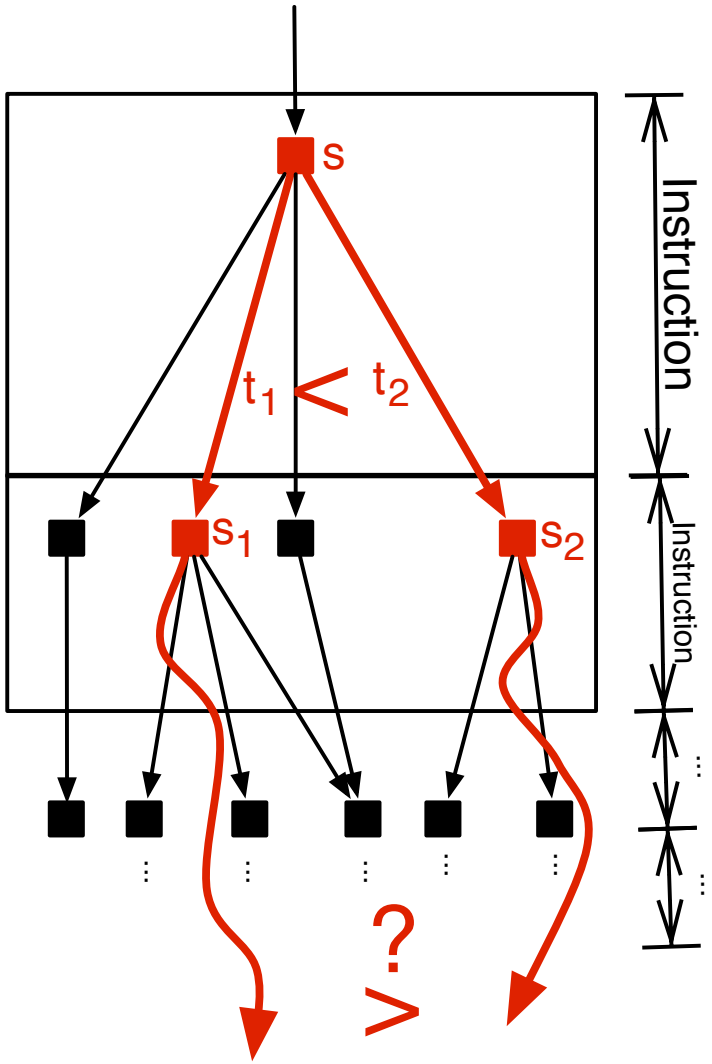
$\max(s, \iota_0 \ldots \iota_n) := \max\{t \mid s \xrightarrow[\iota_0 \ldots \iota_n]{t} s'\}$

Example:

$s_1 \xrightarrow[\iota_1]{2} s_2$

$\max(s_1, \iota_1 \iota_2) = 5$

Timing Anomalies



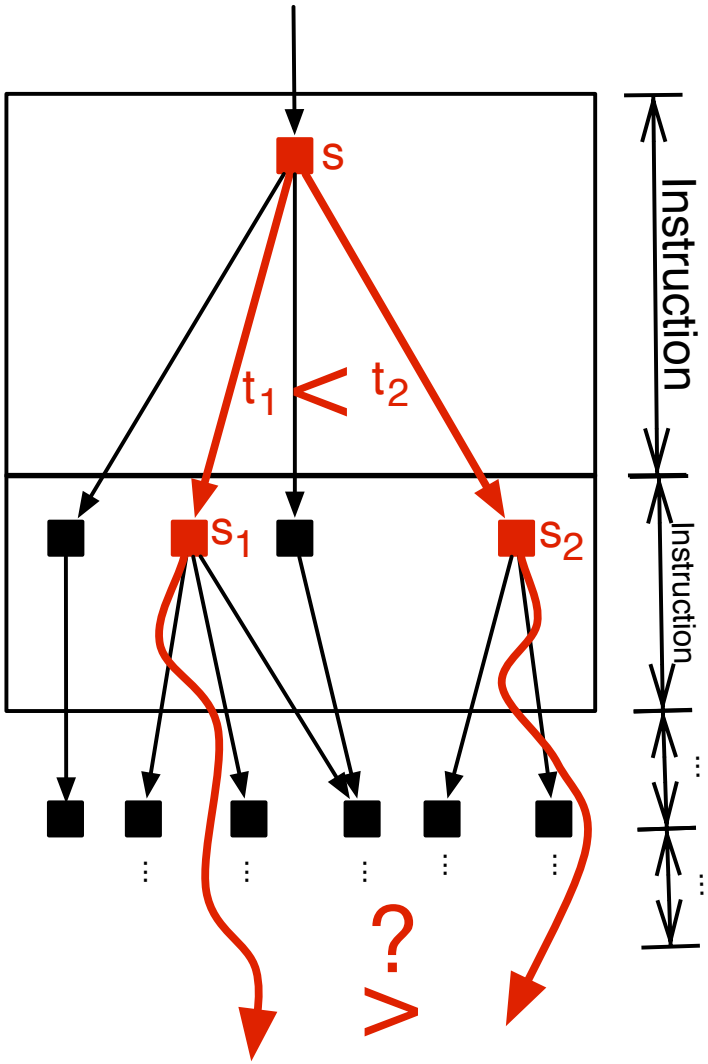
Definition (Timing anomaly)

An instruction semantics has a *timing anomaly* if there exists a sequence of instructions $\iota_0 \iota_1 \ldots \iota_n$, and an abstract state s, such that

- there are states s_1, s_2, with $s \xrightarrow[\iota_0]{t_1} s_1$ and $s \xrightarrow[\iota_0]{t_2} s_2$, and $t_1 < t_2$, such that
- $t_1 + \max(s_1, \iota_1 \ldots \iota_n) > t_2 + \max(s_2, \iota_1 \ldots \iota_n)$.

Safely Discarding Analysis States

SAARLAND UNIVERSITY
COMPUTER SCIENCE

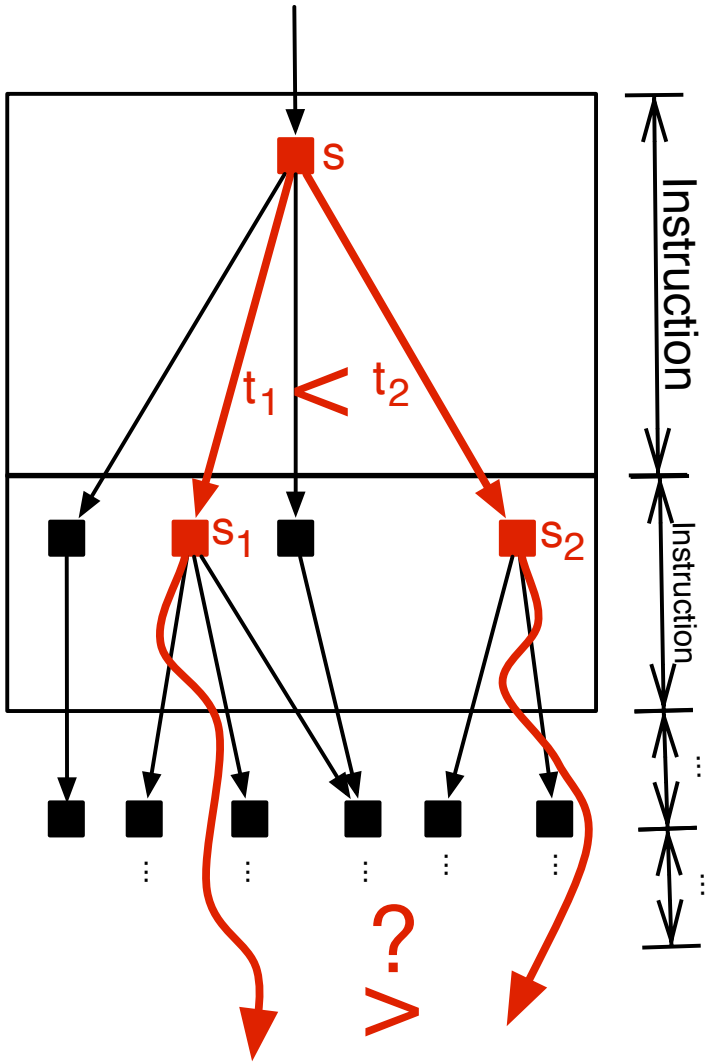


Idea: Precompute maximal difference in timing for pairs of states.

Discard states that cannot "overtake" others anymore.

"Locally exclude timing anomalies."

Valid Δ Function



Definition (Valid Δ)

A Δ function is *valid*, if for all pairs of states s_1, s_2 and for all instruction sequences $\iota_0 \ldots \iota_n$:

$$\Delta(s_1, s_2) \geq \max(s_1, \iota_0 \ldots \iota_n) - \max(s_2, \iota_0 \ldots \iota_n)$$

Discard s_1 if $\Delta(s_1, s_2) + t_1 \leq t_2$.

Discard s_2 if $\Delta(s_2, s_1) + t_2 \leq t_1$.

Computing Δ Functions

System of *difference* constraints:

For empty sequence of instructions:

$$\Delta(s_1, s_2) \geq 0$$

Recursive constraints:

$$\Delta(s_1, s_2) \geq t_1' - t_2' + \Delta(s_1', s_2') \quad \text{if } s_1 \xrightarrow[\iota]{t_1'} s_1' \wedge s_2 \xrightarrow[\iota]{t_2'} s_2' \text{ for some } \iota.$$

$\longrightarrow$ Can be solved by a shortest paths computation.

Domino Effects

Least $\Delta(s_1, s_2)$ not always finite:

Definition (Domino effect)

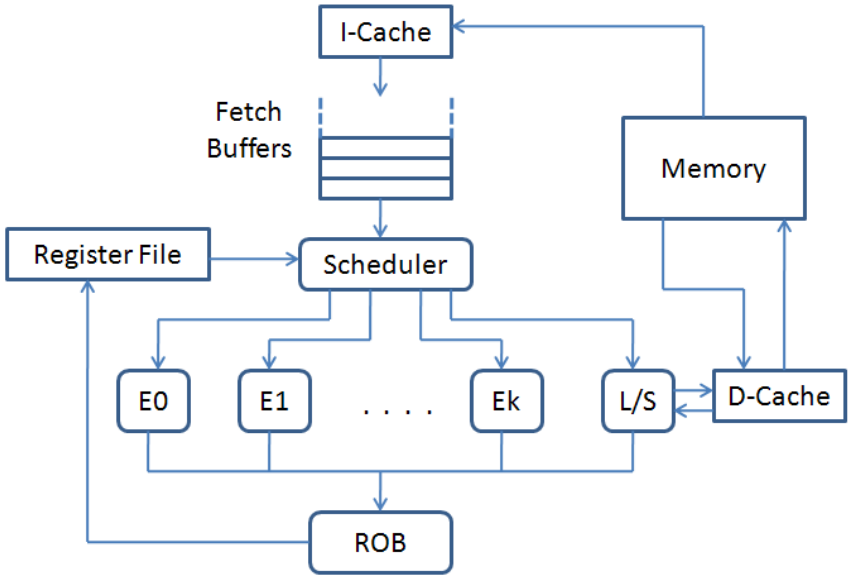
An instruction semantics has a *domino effect* if there are two states s_1, s_2, such that for each $\Delta \in \mathbb{N}$ there is a sequence of instructions $\iota_0 \ldots \iota_n$, such that

$$\max(s_1, \iota_0 \ldots \iota_n) - \max(s_2, \iota_0 \ldots \iota_n) \geq \Delta.$$

But: Ratio $\frac{\max(s_1, \iota_0 \ldots \iota_n)}{\max(s_2, \iota_0 \ldots \iota_n)}$ always bounded.

Case Study

- Computed Δ function for simple processor with:
  - 2 instruction types
  - 2 functional units
  - execution times between 2 and 6 cycles
  - a 4 instruction fetch buffer
- Results:
  - 555 states
  - 97340 constraints
  - Δ function ranges from 0 through 7



Conclusions & Future Work

- Sound and efficient WCET analysis in the presence of timing anomalies, by
  - locally excluding timing anomalies, using
  - precomputed Δ functions.
- Computed Δ functions for relatively simple architectures.

- Future work:
  - Compute Δ functions for real-world architectures.
  - Perform WCET analysis based on that basis.
  - Explore further trade-offs between efficiency and precision.