Timing Analysis

Introduction

Reinhard Wilhelm

Saarland University, Saarbrücken, Germany

January 18, 2011

Hard Real-Time Systems

Safety critical applications:

- Avionics, automotive, train industries, manufacturing control

Sideairbag in car, Reaction in <10 mSec

Crankcraft-synchronous tasks,

Reaction in <45 μSec

Embedded controllers must finish their tasks *within given time bounds*. Developers would like to know the *Worst-Case Execution Time (WCET)* to give a guarantee

Static Timing Analysis

producing the input to schedulability analysis

Schedulability analysis has assumed the knowledge of the execution time of tasks.
So, the problem to solve:

- Given
  1. a software task to produce some reaction,
  2. a hardware platform, on which to execute the software,
  3. a required reaction time, e.g. the period of the task.
- Derive:
  - a reliable (and precise) upper bound on the execution times.

What does the execution time depends on?

SAARLAND UNIVERSITY
COMPUTER SCIENCE

- the input—this has always been so and will remain so,
- the initial state of the platform—this is (relatively new)

- interferences from the environment—this depends on whether the system design admits it (preemptive scheduling, interrupts, multi-core)

What does the execution time depends on?

SAARLAND UNIVERSITY
COMPUTER SCIENCE

- the input—this has always been so and will remain so,
- the initial state of the platform—this is (relatively new)
  - Caused by caches, pipelines, speculation, etc.

- interferences from the environment—this depends on whether the system design admits it (preemptive scheduling, interrupts, multi-core)

What does the execution time depends on?

SAARLAND UNIVERSITY
COMPUTER SCIENCE

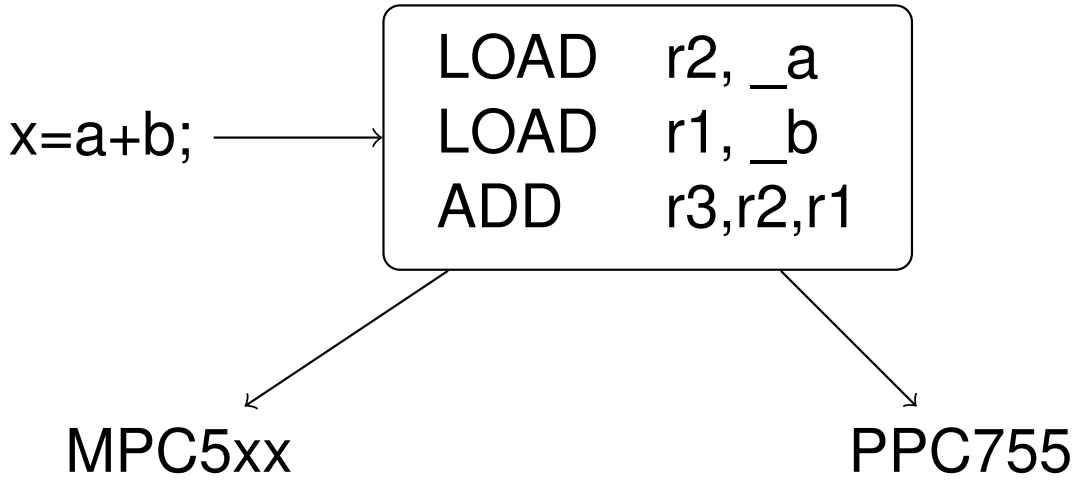
- the input—this has always been so and will remain so,
- the initial state of the platform—this is (relatively new)
  - Caused by caches, pipelines, speculation, etc.
  - Explosion of the space of inputs **and** initial states
    $\Rightarrow$ all exhaustive approaches infeasible
- interferences from the environment—this depends on whether the system design admits it (preemptive scheduling, interrupts, multi-core)

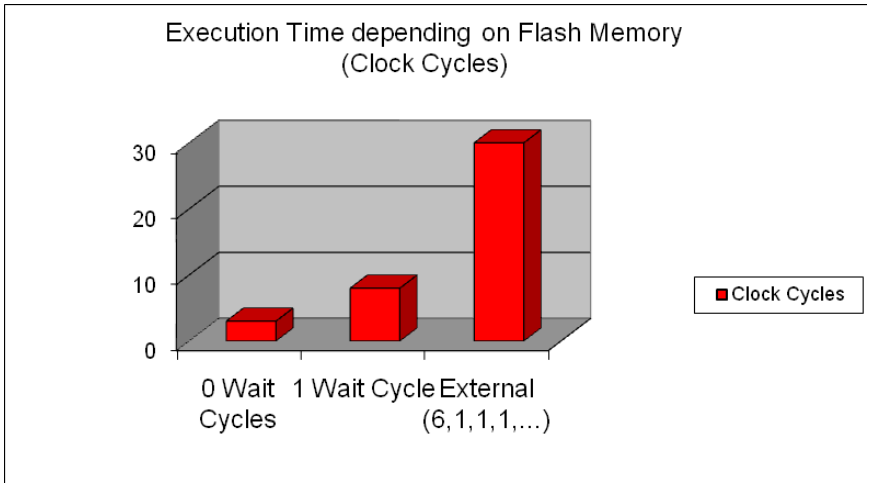
What does the execution time depends on?

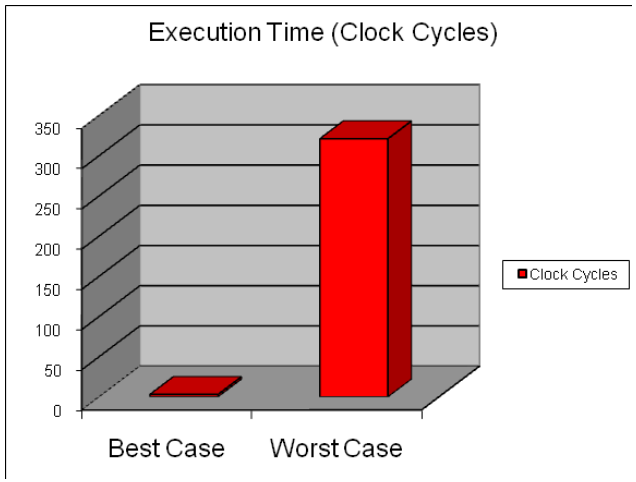
SAARLAND UNIVERSITY
COMPUTER SCIENCE

- the input—this has always been so and will remain so,
- the initial state of the platform—this is (relatively new)
  - Caused by caches, pipelines, speculation, etc.
  - Explosion of the space of inputs **and** initial states
    $\Rightarrow$ all exhaustive approaches infeasible
- interferences from the environment—this depends on whether the system design admits it (preemptive scheduling, interrupts, multi-core)
  - External interferences as seen from analyzed task

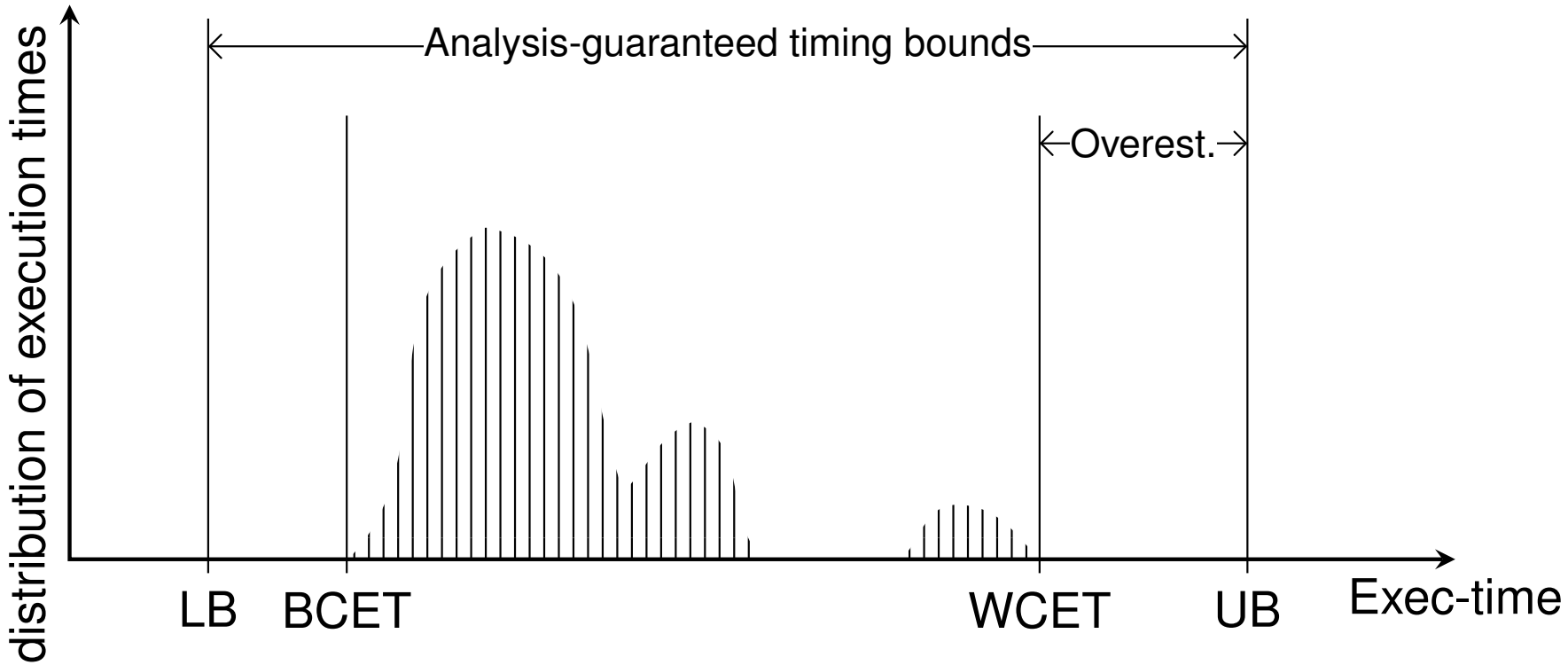
Access Time


x=a+b;
LOAD r2, _a
LOAD r1, _b
ADD r3,r2,r1
MPC5xx
PPC755


Execution Time depending on Flash Memory
(Clock Cycles)
30
20
10
0
0 Wait Cycles
1 Wait Cycle
External (6,1,1,1,...)
Clock Cycles

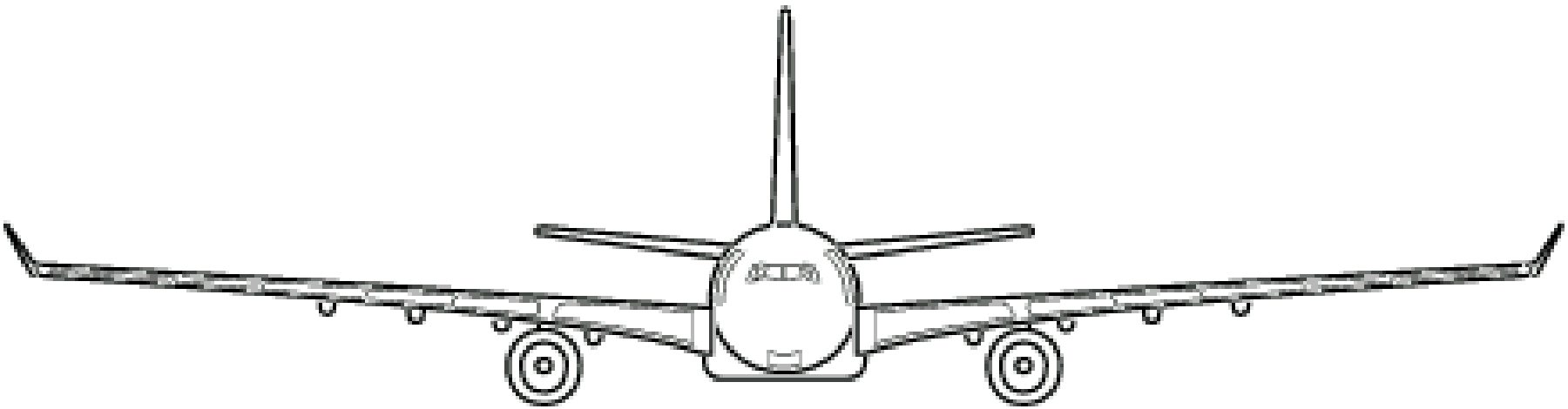

Execution Time (Clock Cycles)
350
300
250
200
150
100
50
0
Best Case
Worst Case
Clock Cycles

Timing Analysis



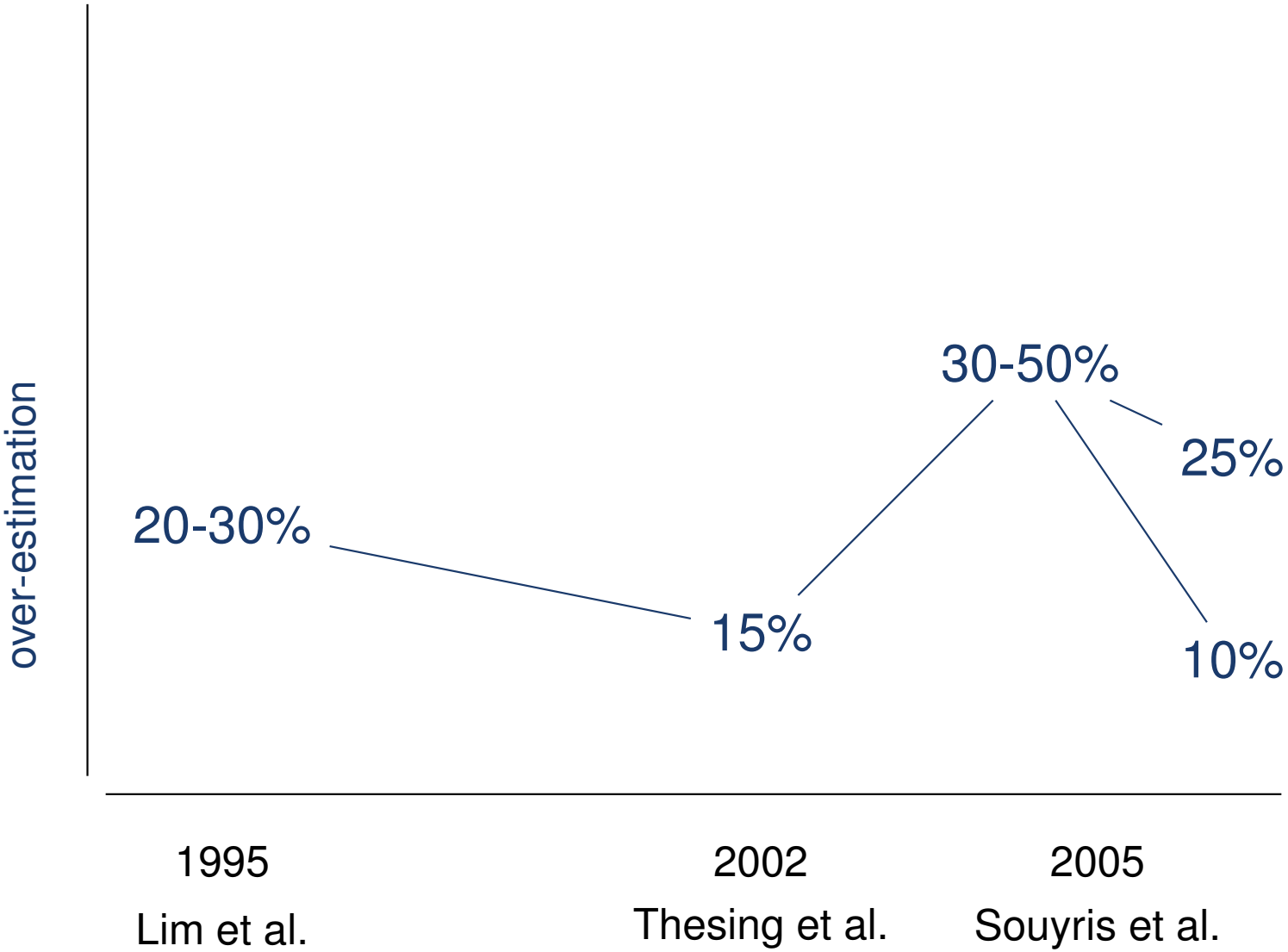
aiT WCET Analyzer

IST Project DAEDALUS final review report:
"*The AbsInt tool is probably the best kind in the world and it is justified to consider this result as a breakthrough.*"

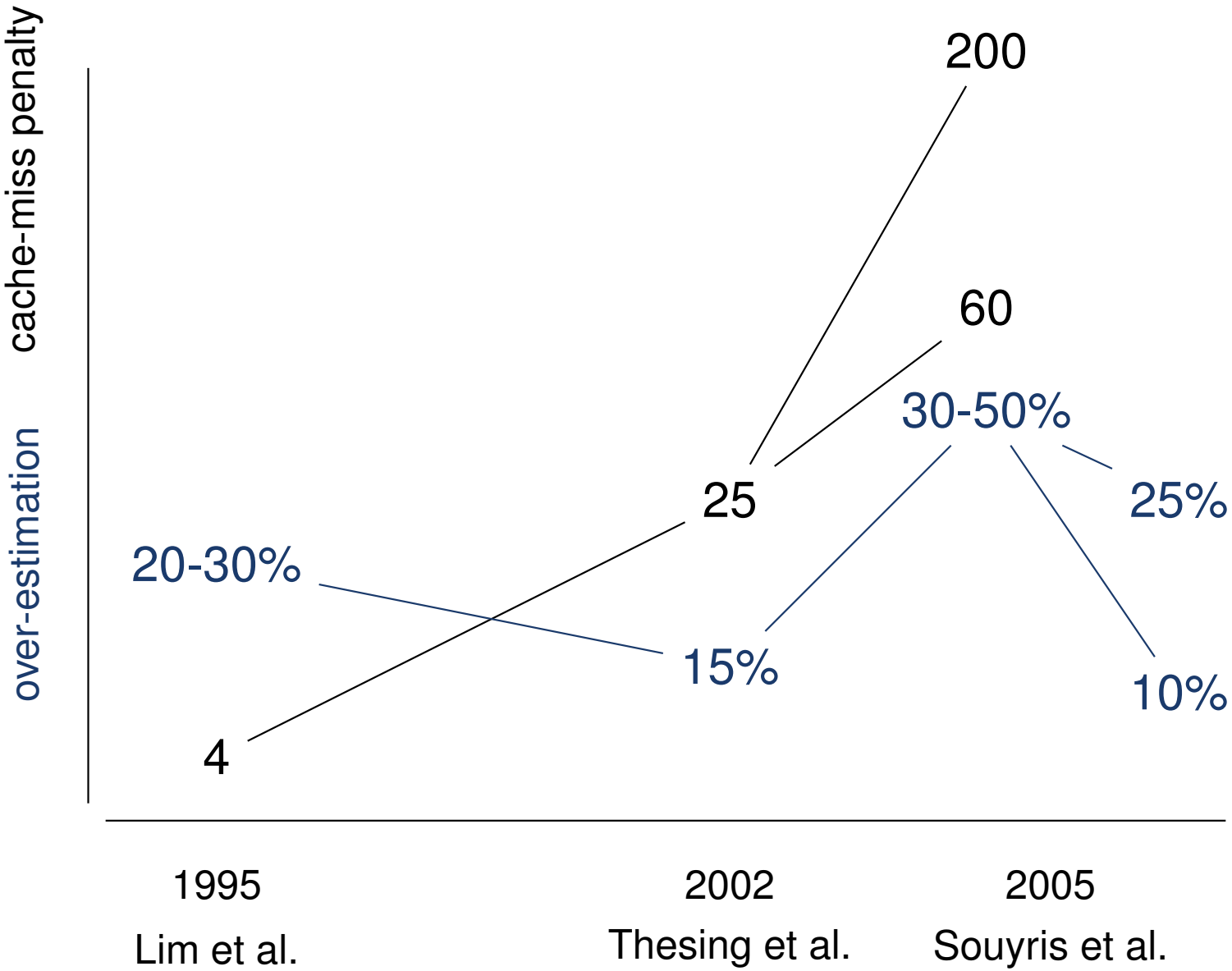


Several time-critical subsystems of the airbus A380 have been certified using aiT; aiT is the only validated tool for these applications.

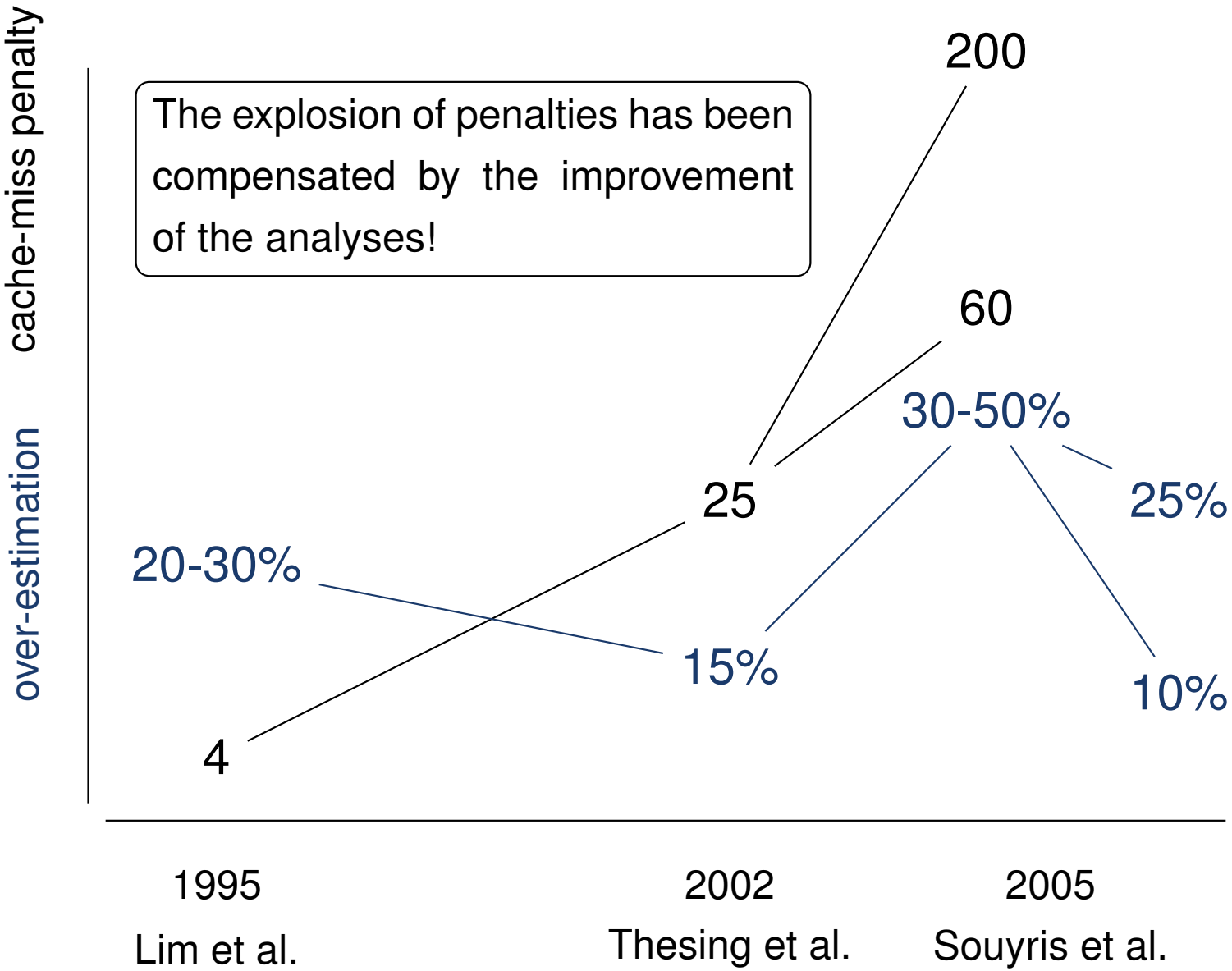
Tremendous Progress during the 15 past Years



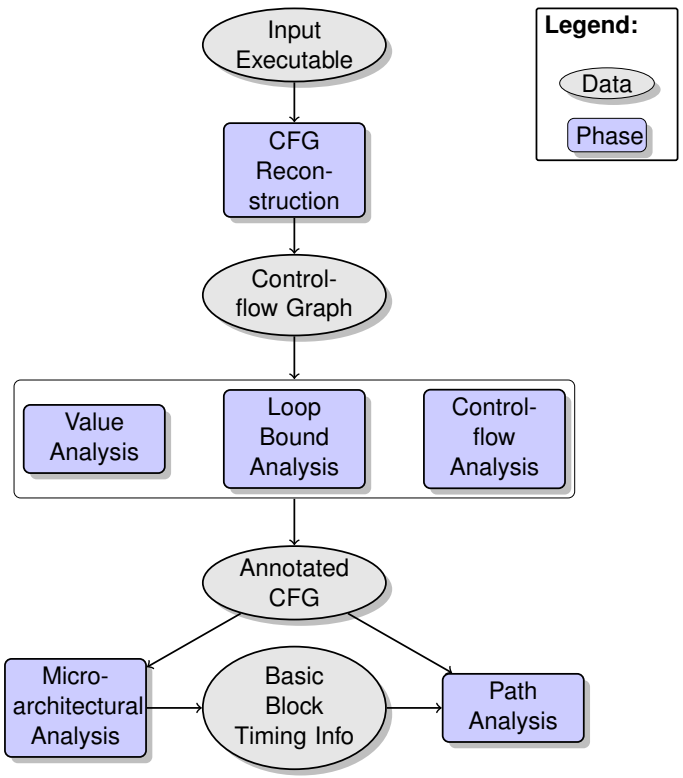
Tremendous Progress during the 15 past Years



Tremendous Progress during the 15 past Years

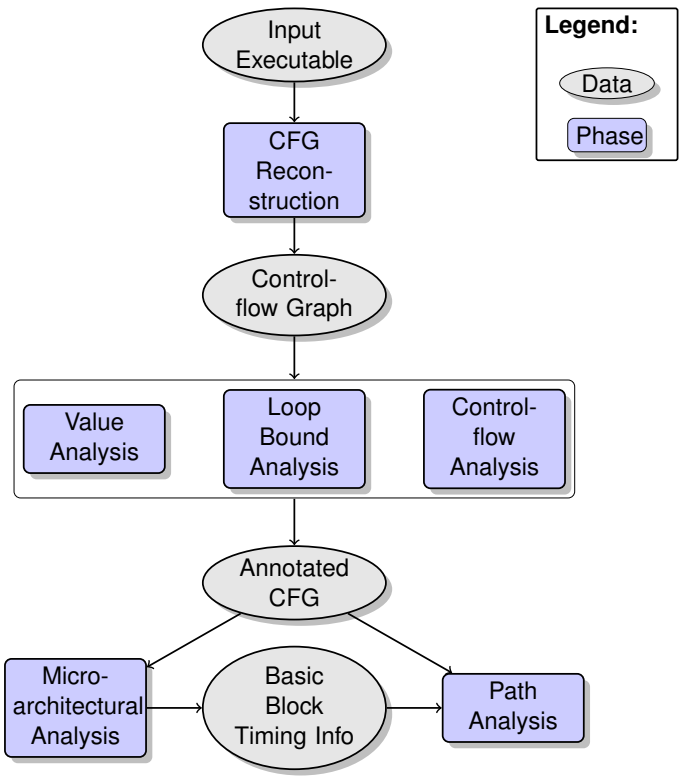


Static Timing Analysis



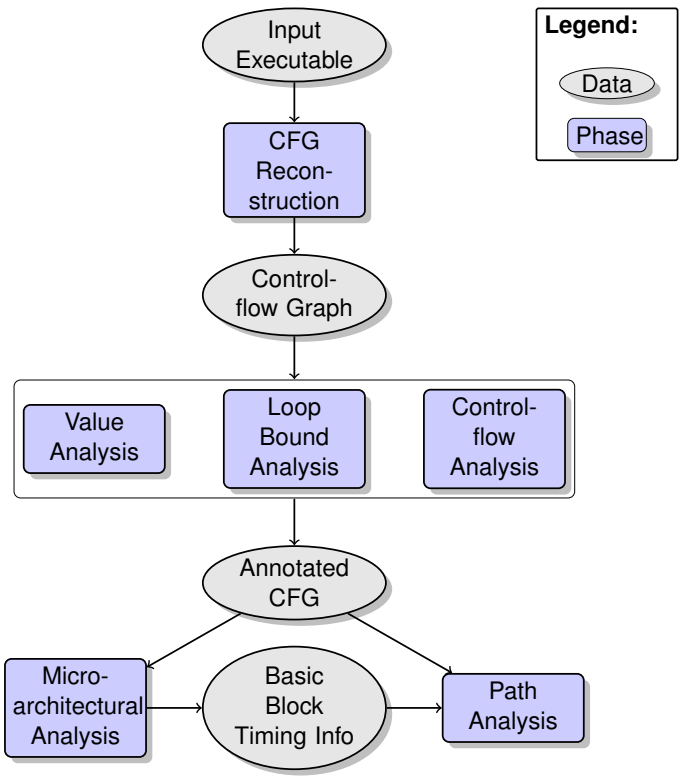
- Value Analysis:
  - determines enclosing intervals for the values in registers and local variables
- Loop Bound analysis:
  - determines loop bounds
- Control Flow Analysis:
  - determines infeasible paths

Static Timing Analysis



- Value Analysis:
  - determines enclosing intervals for the values in registers and local variables
- Loop Bound analysis:
  - determines loop bounds
- Control Flow Analysis:
  - determines infeasible paths
- Micro-architectural Analysis:
  - combined cache and pipeline analysis
  - derives invariants about architectural execution states, computes bounds on execution times of basic blocks

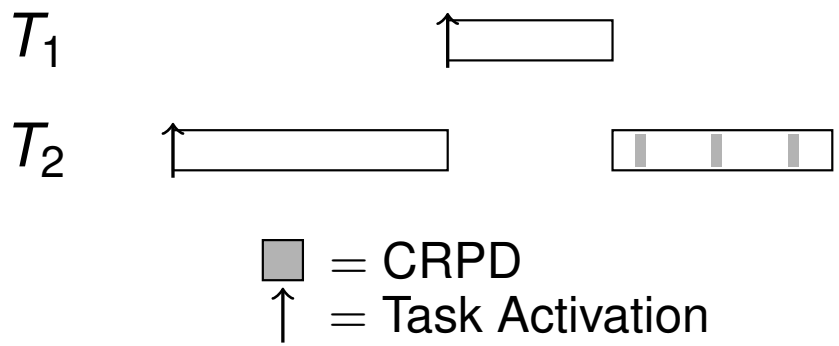
Static Timing Analysis



- Value Analysis:
  - determines enclosing intervals for the values in registers and local variables
- Loop Bound analysis:
  - determines loop bounds
- Control Flow Analysis:
  - determines infeasible paths
- Micro-architectural Analysis:
  - combined cache and pipeline analysis
  - derives invariants about architectural execution states, computes bounds on execution times of basic blocks
- Global Bound Analysis:
  - determines a worst-case path and an upper bound

Timing analysis - Preemptive Systems

- Timing analysis:
  - upper-bound on the execution time (WCET)
  - uninterrupted execution
- Preemptive scheduling
  - upper bound on execution times
  - context-switch costs – composed of task-management costs and architecture-related costs



Predictability

- Predictability: not a boolean property
- Some performance-enhancing features, like certain
  - Caches
  - Pipelines

  are analyzable, others are not...
- Explore trade-offs between (worst-case)predictability, (average-case) performance, and costs
- Goal:
  Design architectures with high worst-case performance that can be precisely and efficiently determined

Predictable Multi-Core Architecture

SAARLAND UNIVERSITY
COMPUTER SCIENCE

- Predictable cores are a prerequisite for predictable multi-cores
- The main culprit: *Sharing* of resources
  - Main memory, caches
  - Busses
  - I/O
  - Flash memory
- introduces variability of execution times and, thus, imprecision,
- increases the state space to analyze and, thus, the complexity.

Timing analysis and timing predictability

- Caches
- Bounding the preemption delay
- Pipeline analysis and Predictability of architectures