ExaScience Lab
Intel Labs Europe
EXASCALE COMPUTING

Sampled Simulation of Multi-threaded Applications

Trevor E. Carlson, Wim Heirman,
Lieven Eeckhout

http://www.snipersim.org
Monday, April 22nd, 2013

Universiteit Gent
intel

Overview

- How can we help the hero save the princess?
- How can we create a representative sample of a multi-threaded application?

- Prior Work
- Key Contributions of this Work
- Results

DEMANDS ON SIMULATION ARE INCREASING

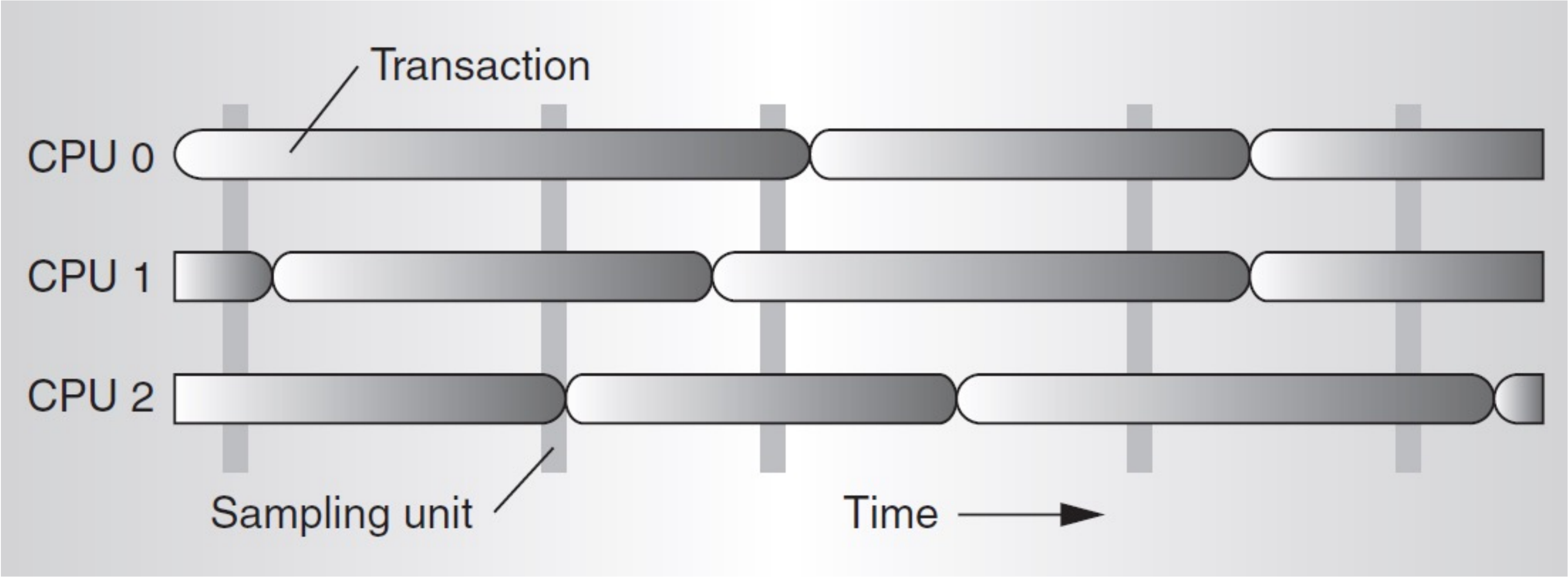
- Increasing cache sizes
  - Need a large working set to fully exercise a large cache
  - Scaled-down applications do not exhibit the same behavior
- Increasing core counts
  - Linear increase in simulator workload
  - Single-threaded simulator sees a rising gap
    - workload: increasing target cores
    - available processing power: near-constant single-thread performance of host machine
- Multi-threaded workloads
  - Not reproducible with traces requiring a number of simulation runs
- New solutions are needed

Workload Reduction is the Key

- Many workload reduction techniques exist today
  - Sampling
    - SimPoint
    - SMARTS
    - FlexPoints
  - Reduction
    - Smaller input sizes
    - Reduced numbers of iterations
- Current sampling techniques are not sufficient
  - Using CPI as a proxy for runtime does not hold for multi-threaded applications
    - Invalidates assumptions of previous work
    - Waiting for locks and barriers and other synchronization primitives

FlexPoints

- Overview
  - Supports sampling multi-threaded throughput (server) applications
  - Creates a sample based on a number of sampling units to minimize CPI variation
  - Not applicable to applications where threads synchronize or communicate



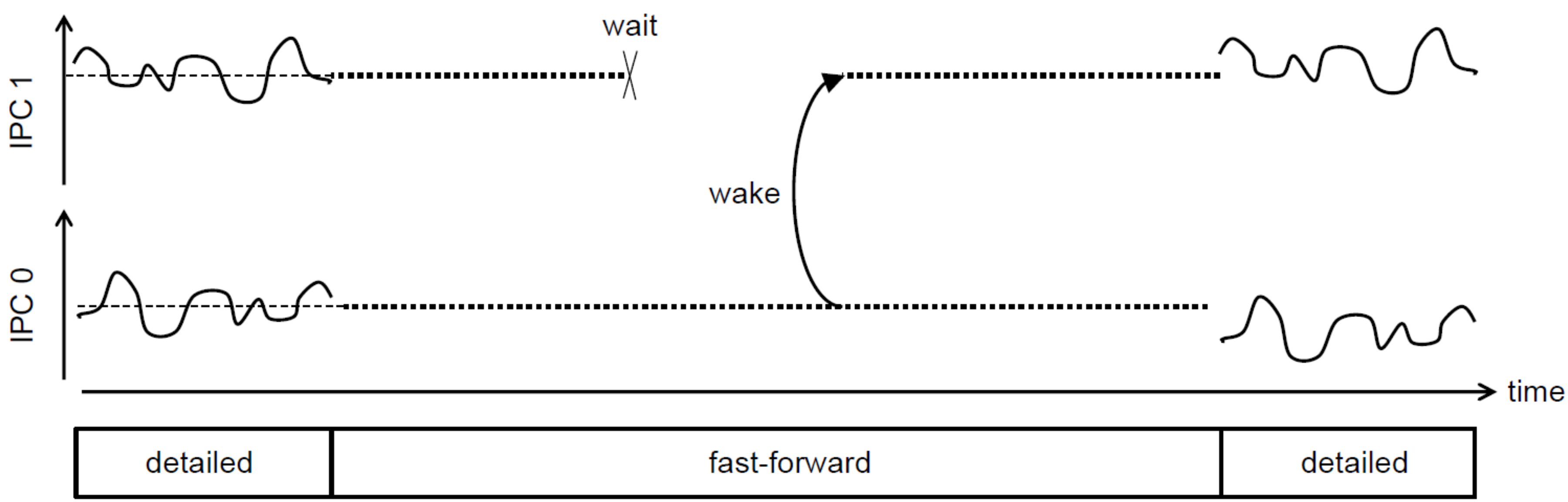
Wenisch, et al., IEEE MICRO 2006

Multi-Threaded Sampling

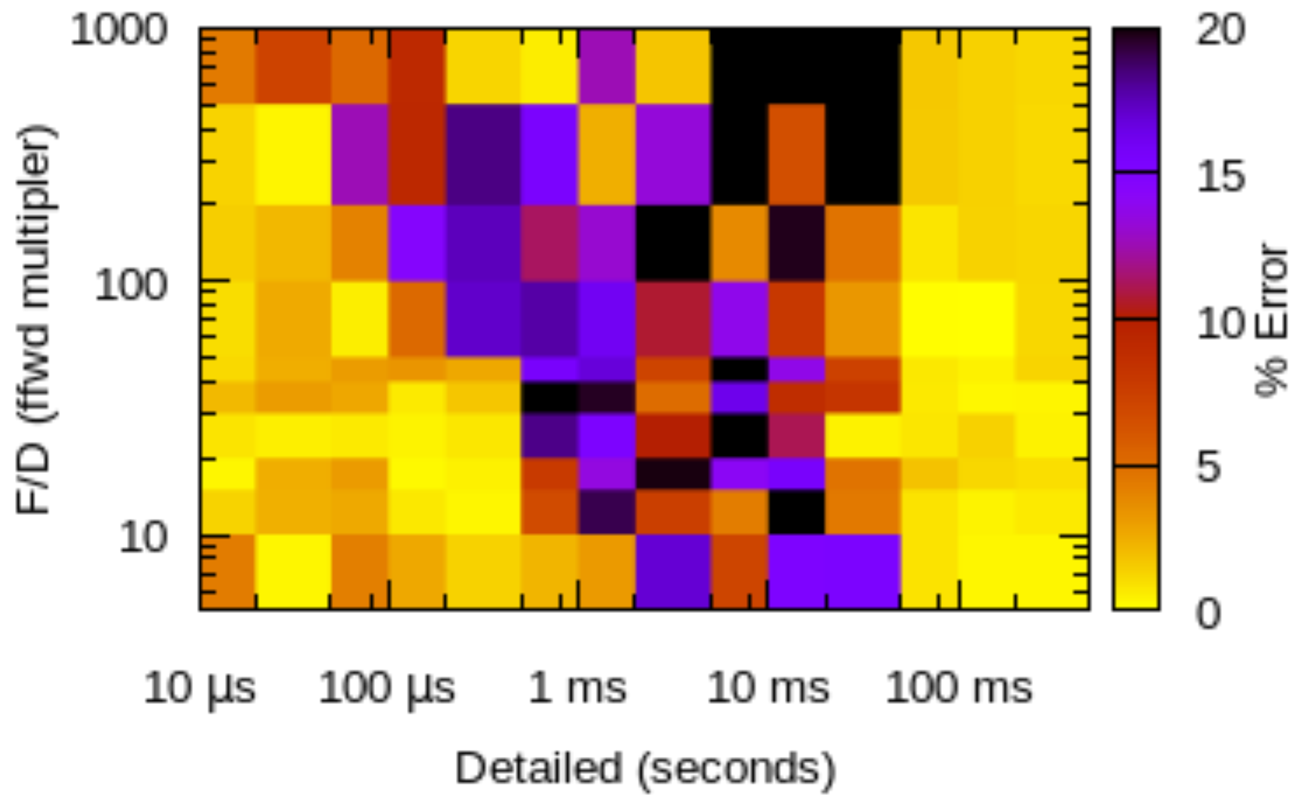
- Goals:
  - Accurately predict application runtime of synchronizing multi-threaded applications
    - (not just average CPI)
  - Periodically sample a multi-threaded application to reduce amount of detailed simulation time
- Examples of synchronizing mechanisms
  - Barriers, mutexes
    - OMP-style parallelism
    - Pipelined parallelism
  - LOCKed instructions, compare-and-swap

Initial Sampling Process

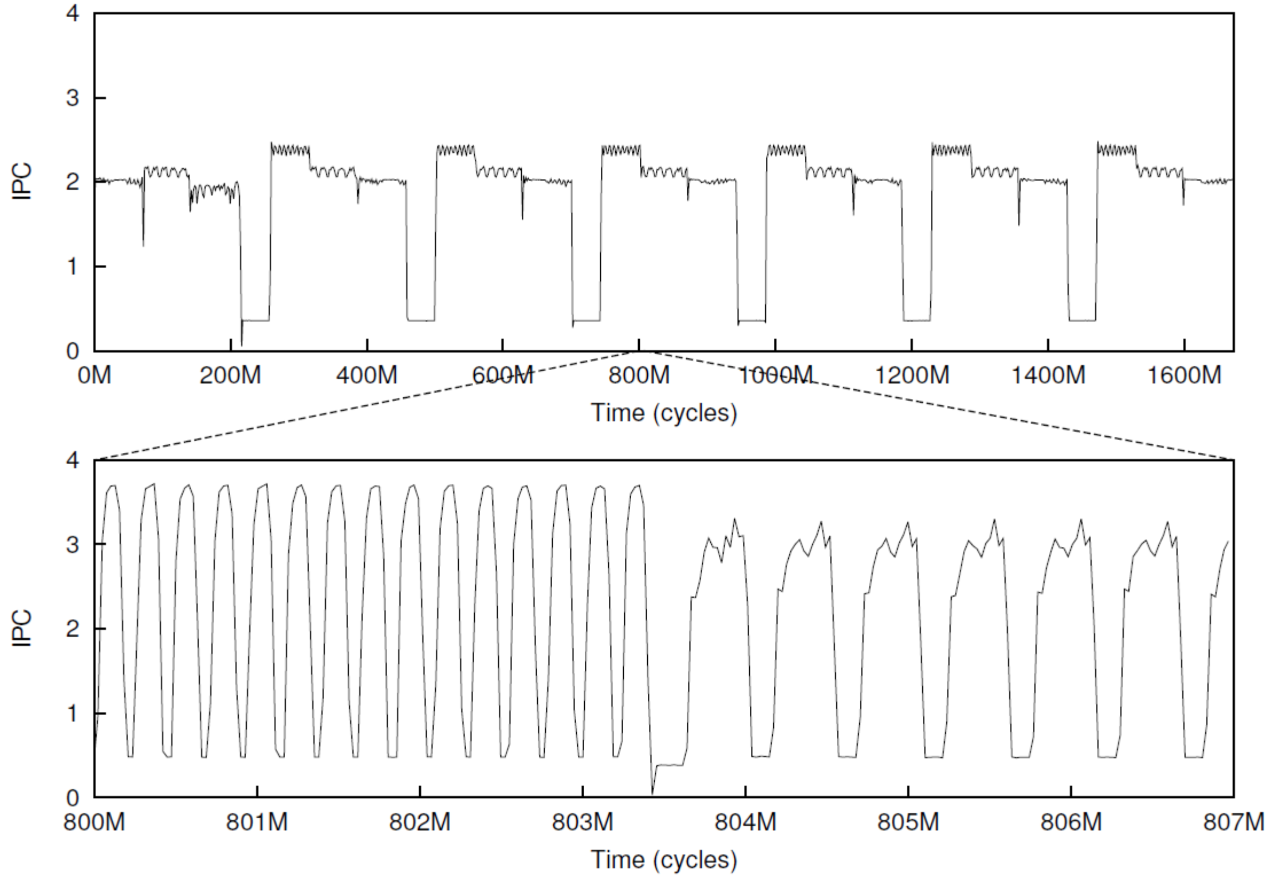
- Sampling Overview
  - Detailed = all components enabled (warmup+simulation)
  - Fast-forward = memory-hierarchy enabled
- Key Insights
  - Independent IPCs for each individual thread
  - Keeping track of wait/wake during fast-forwarding



Sample Selection

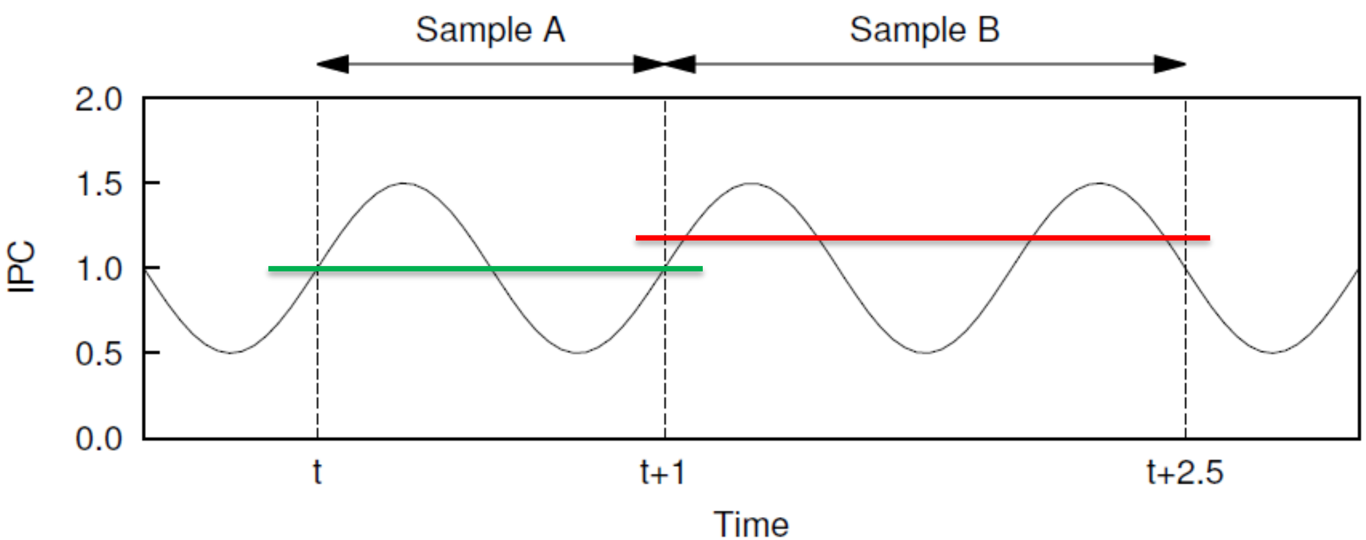


Applications are Periodic



npb-ft, class A, 8 threads

Application periodicity Affects Accuracy

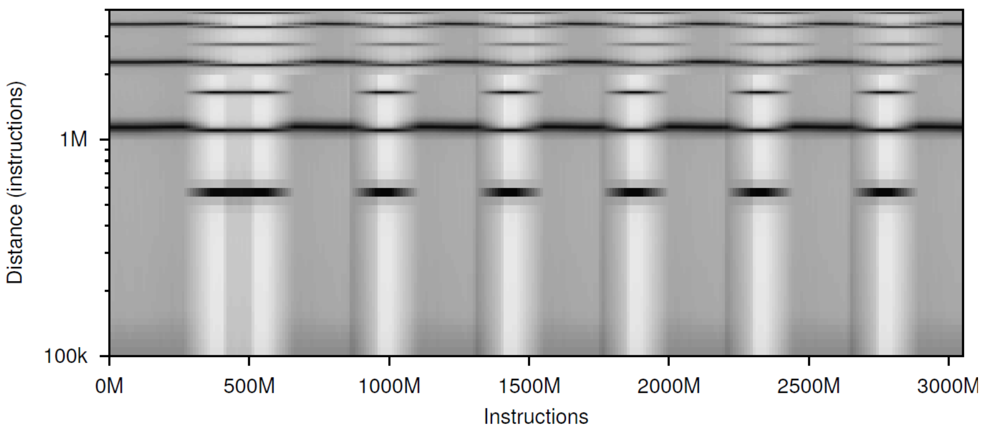


Sampling at exactly one period would produce excellent results

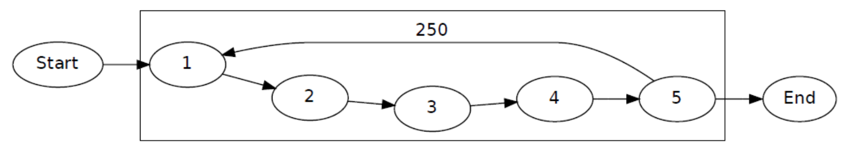
Sampling at more than one period can produce a sampling error

Identify Periodicities

- We wanted to identify application periodicities in a micro-architectural independent manner



BBV Autocorrelation
npb-ft, class A, 8 threads, with 550k and 1.14M insn periodicities

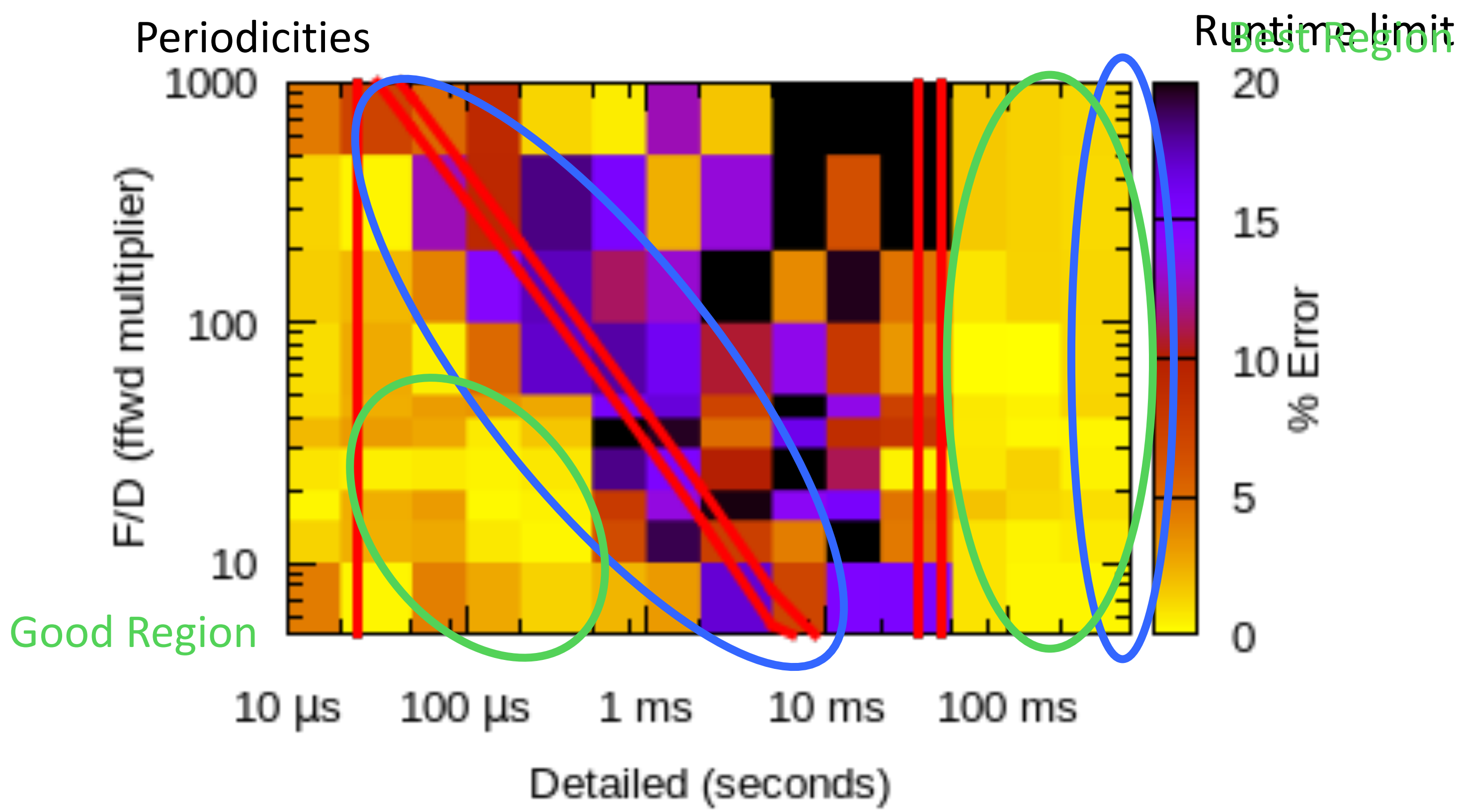


Edge	Avg	Δ/μ
$1 \rightarrow 2$	37.14 M	12.0%
$2 \rightarrow 3$	38.97 M	16.1%
$3 \rightarrow 4$	1.96 M	36.6%
$4 \rightarrow 5$	17.45 M	<1%
$5 \rightarrow 1$	9.83 M	<1%

OMP Call Structure
npb-lu, class A, 8 threads
with high variability (not used)

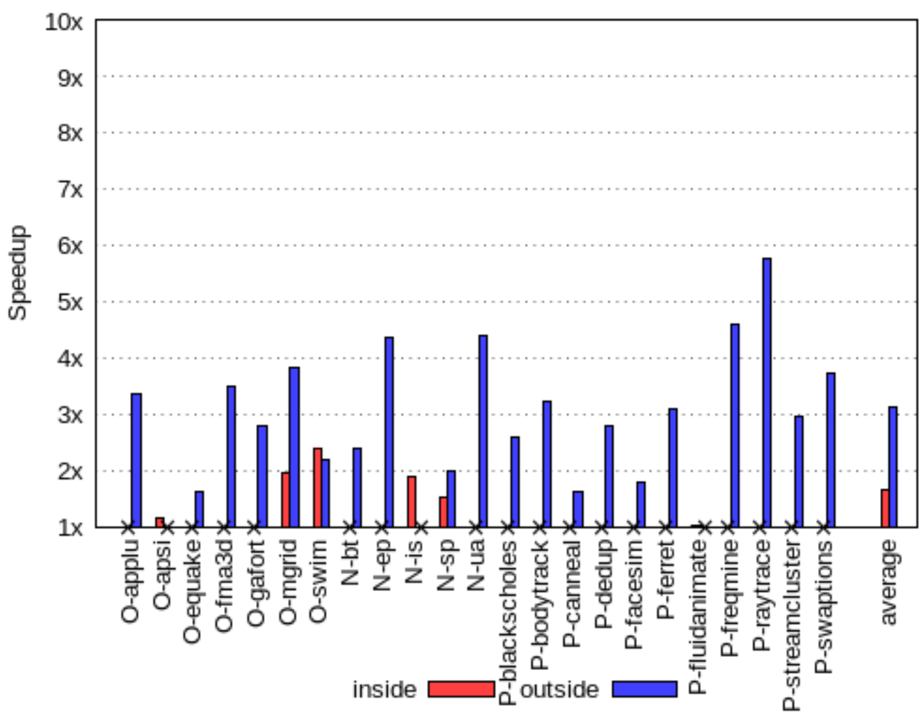
Sampling Process

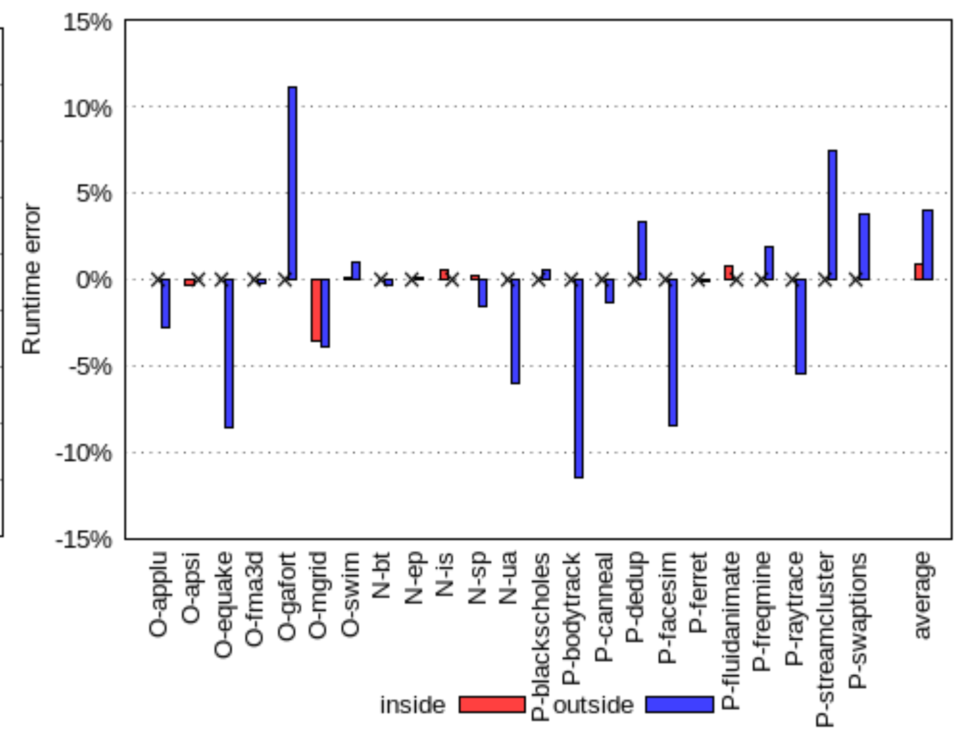
- Sampling sufficiently above or below the period will minimize error



RESULTS

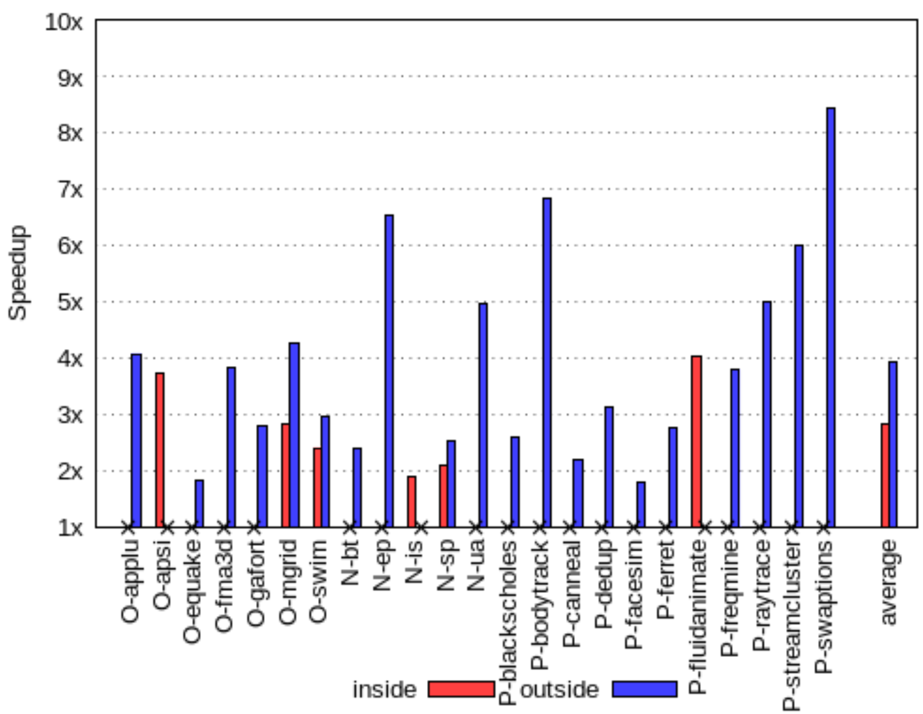
- Predicted Most-Accurate Results
  - Average speedup of 2.9x, maximum of 5.8x
  - Average absolute error of 3.5%

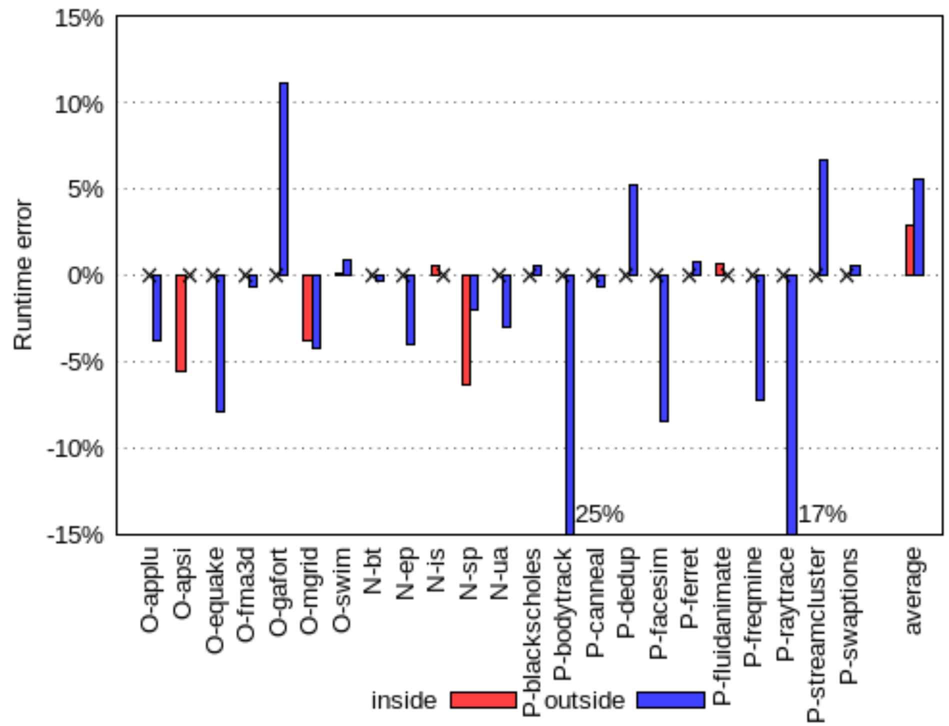




RESULTS

- Predicted Fastest Results
  - Average speedup of 3.8x, maximum of 8.4x
  - Average absolute error of 5.1%





Multi-Threaded Sampling

- Key Contributions
  - Understanding application phase behavior is key to effective sampling
  - Modeling inter-thread interactions during fast-forwarding is important for multi-threaded sampling accuracy
- Predicted Most-Accurate Results
  - Average speedup of 2.9x, maximum of 5.8x
  - Average absolute error of 3.5% across applications
- Predicted Fastest Results
  - Average speedup of 3.8x, maximum of 8.4x
  - Average absolute error of 5.1% across applications

ExaScience Lab
Intel Labs Europe
EXASCALE COMPUTING

Sampled Simulation of Multi-threaded Applications

Trevor E. Carlson, Wim Heirman,
Lieven Eeckhout

http://www.snipersim.org
Monday, April 22nd, 2013

Universiteit Gent
intel