Runtime variability in scientific parallel applications

W. Heirman, J. Dambre, D. Stroobandt, J. Van Campenhout
ELIS Department, Ghent University, Belgium

Sponsored by IAP-V *PHOTON* & IAP-VI *photonics@be*,
Belgian Science Policy Office

Outline

- Variability in simulation: what & why?
- Measurements in SPLASH-2
- Effects on fidelity and relative accuracy
- Overview of possible solutions
- Conclusions

UNIVERSITEIT GENT

Variability in simulation

- Simulations heavily used to predict performance of architectural or other improvements
- Variation in performance measurements (program runtime, transactions per second)
- What if variation > performance delta?
  → wrong conclusions!

Variability: causes

- Non-determinism in Operating System
  - (background tasks)
  - (external: network, user input)
  - scheduler
  - VM management
- Non-determinism in Parallel application
  - locks: synchronization races
  - magnified by: task queues (changes in load balance), convergence (#iterations can depend on sequence)

Variability: existing work

- Alameldeen & Wood (HPCA'03, MICRO'06)
  - *"Variability is a well-known phenomenon in real systems, but is nearly universally ignored in simulation experiments"*
  - Commercial applications: request-driven (task queues!)
  - Large variations in IPC, transactions/sec
  - Solution: use multiple simulations & statistics (non-overlapping confidence intervals to compare architectures)

Variability: existing work

- Wenisch et. al. (MICRO‘06)
  - Measure User-IPC:
    - Good (near-linear) correlation between U-IPC (measurement) and transactions/sec (performance)
    - Less variation in U-IPC, thus allows higher accuracy
  - Simflex: sampling methodology using stored state
- Lepak et. al. (PACT‘03)
  - ‘Redeeming IPC’: insert delay to remove variability

UNIVERSITEIT GENT

Variability: existing work – scientific applications

- Bienia et. al.: PARSEC benchmark suite
  - Characterization of variability of *architecture-independent* benchmark properties
  - Low for most applications, but:
    - What with architecture-*dependent* properties (=performance indicators)
    - Measured on long (~15 second) input sets, what with input sets suitable for microarchitectural simulations (i.e. Simics/GEMS: x100,000 slowdown, ~1s runtime)

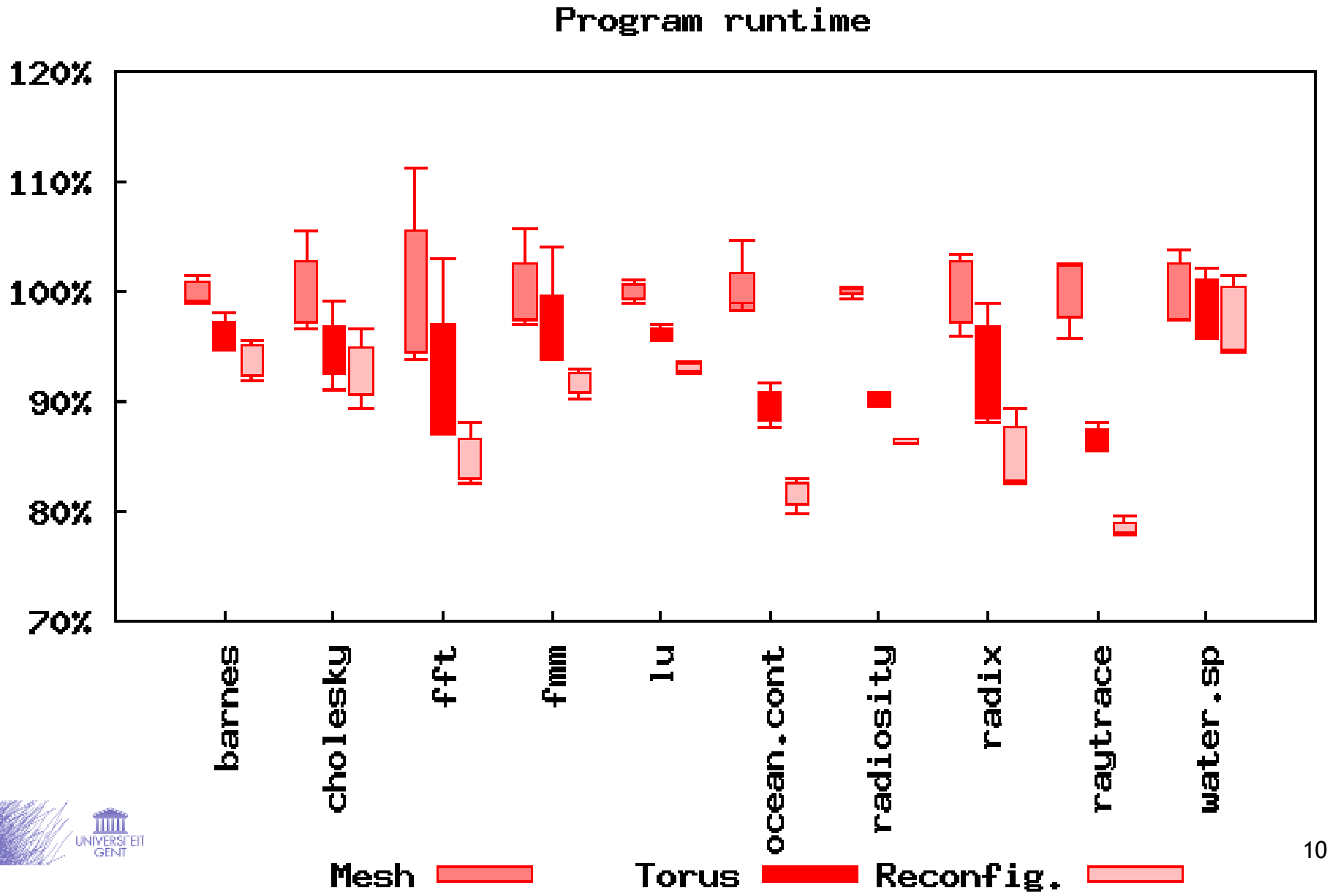
Outline

- Variability in simulation: what & why?
- Measurements in SPLASH-2
- Effects on fidelity and relative accuracy
- Overview of possible solutions
- Conclusions

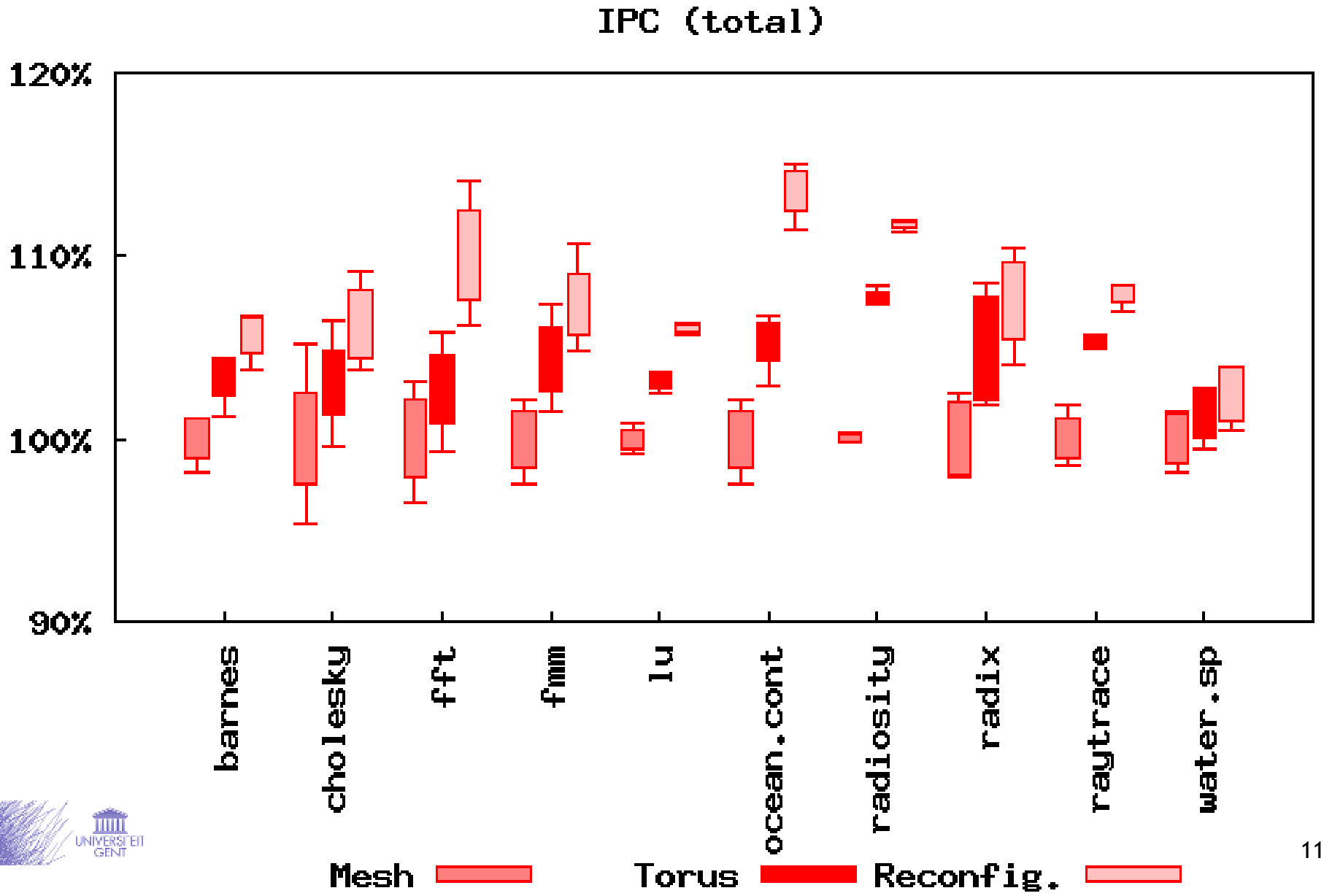
Simulation platform

- Simics + directory-based coherence, interconnect simulation
- 16 processors (UltraSPARC), Solaris 9
- SPLASH-2 benchmarks
- Introduce variability by waiting
  → difference in scheduler initial state
- Study performance while changing the interconnection network

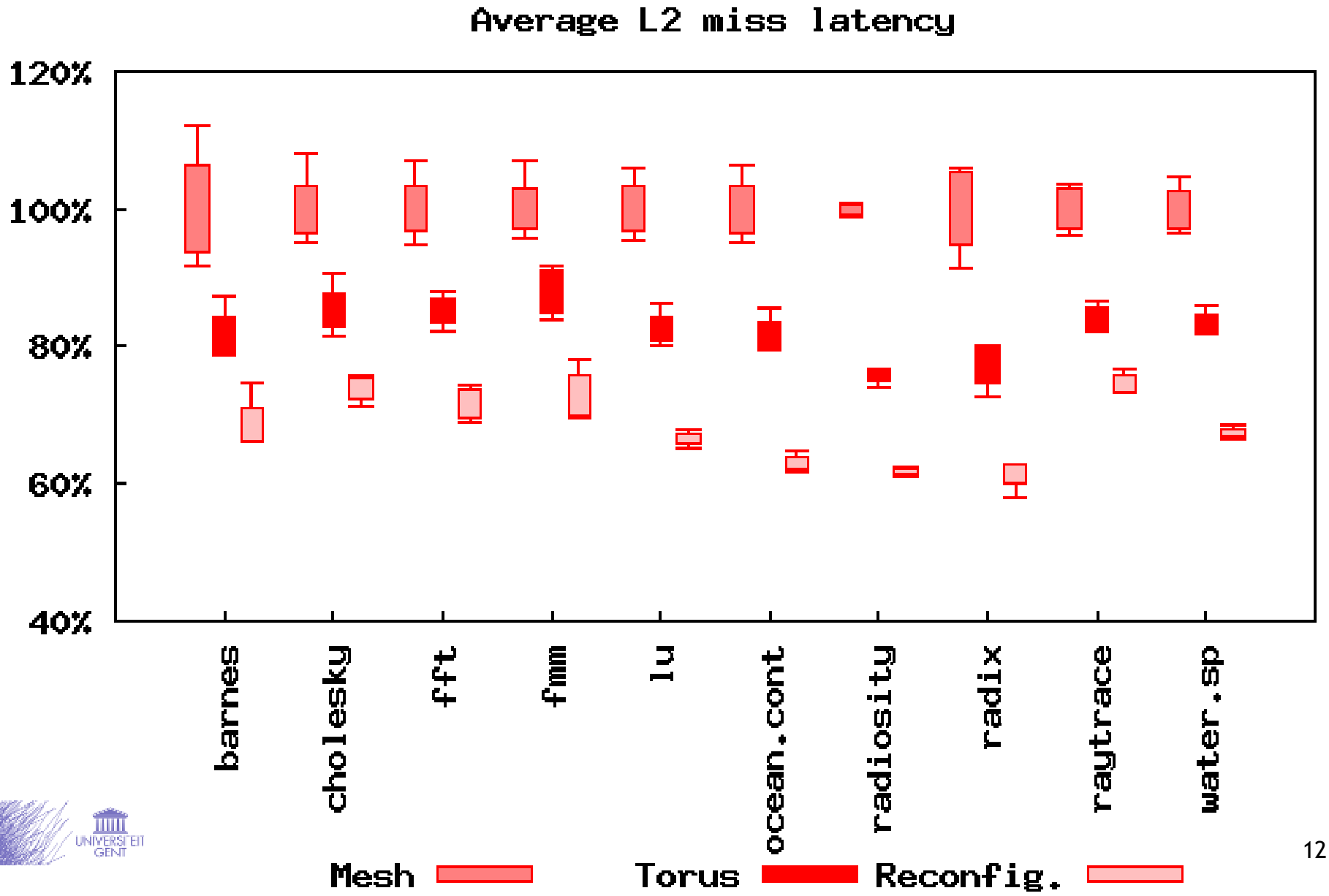
Variability: program runtime



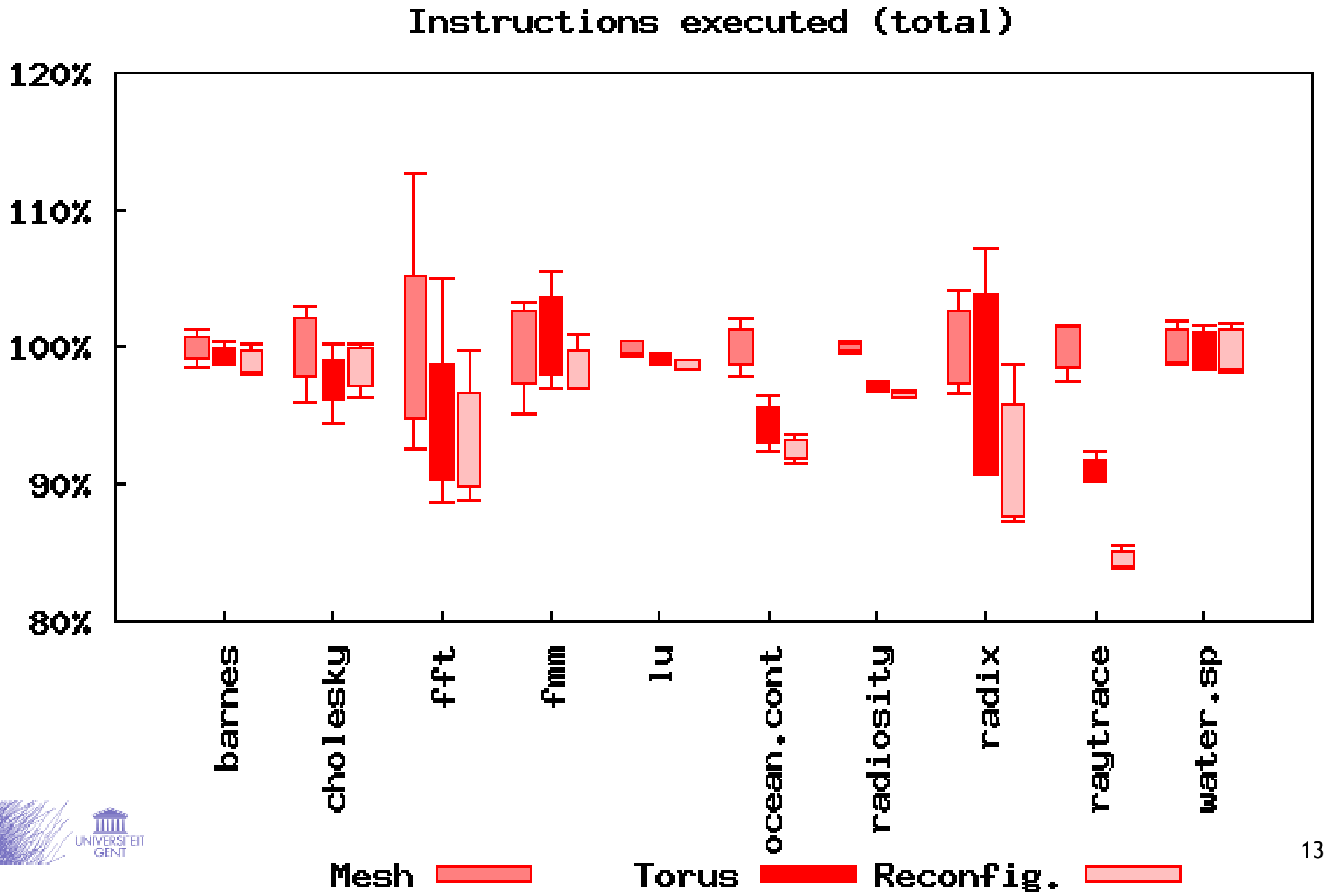
Variability: IPC



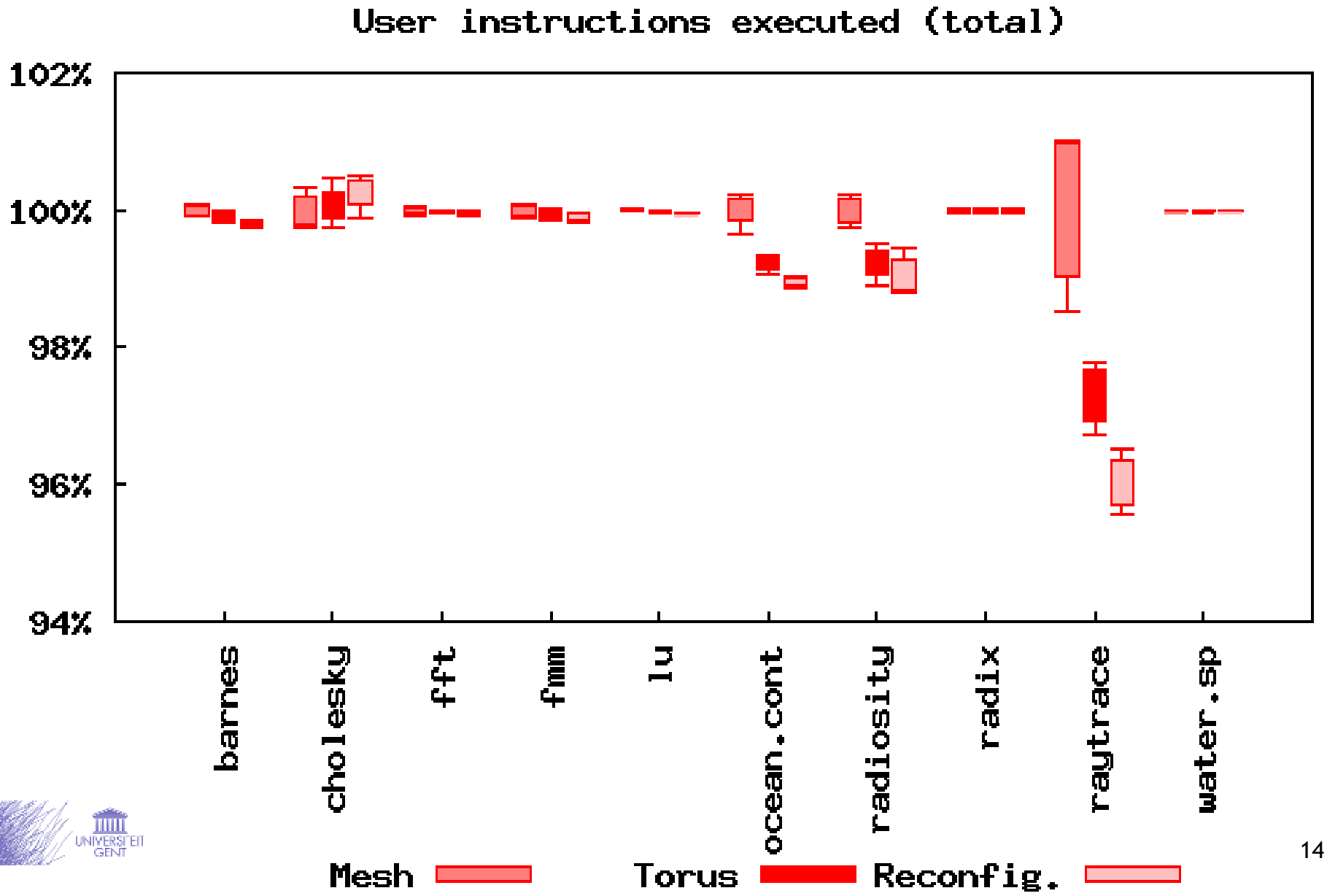
Variability: L2 miss latency



Instructions executed



User instructions executed



Observations

- Variability in runtime: 5 to 10%, can be as big as improvement of better network
- # instructions changes (spinlocks),
  # user instructions constant for most *(but not all)* benchmarks
- Low-level metrics (L2-miss latency): larger difference, less variability, but not always effect on total performance

Outline

- Variability in simulation: what & why?
- Measurements in SPLASH-2
- Effects on fidelity and relative accuracy
- Overview of possible solutions
- Conclusions

UNIVERSITEIT GENT

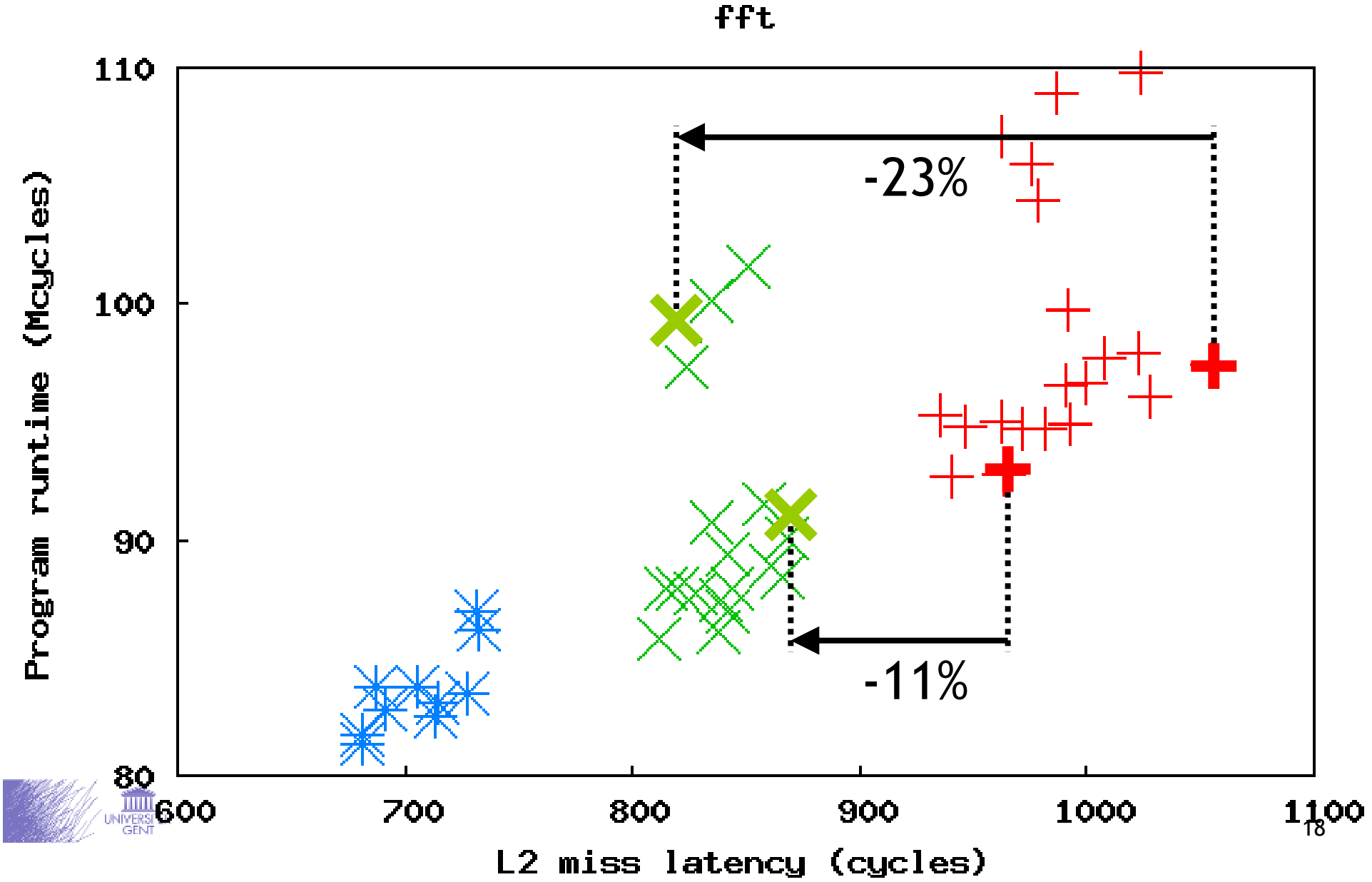
Fidelity and relative accuracy

- Experiment to compare performance of two architectures
- Fidelity:

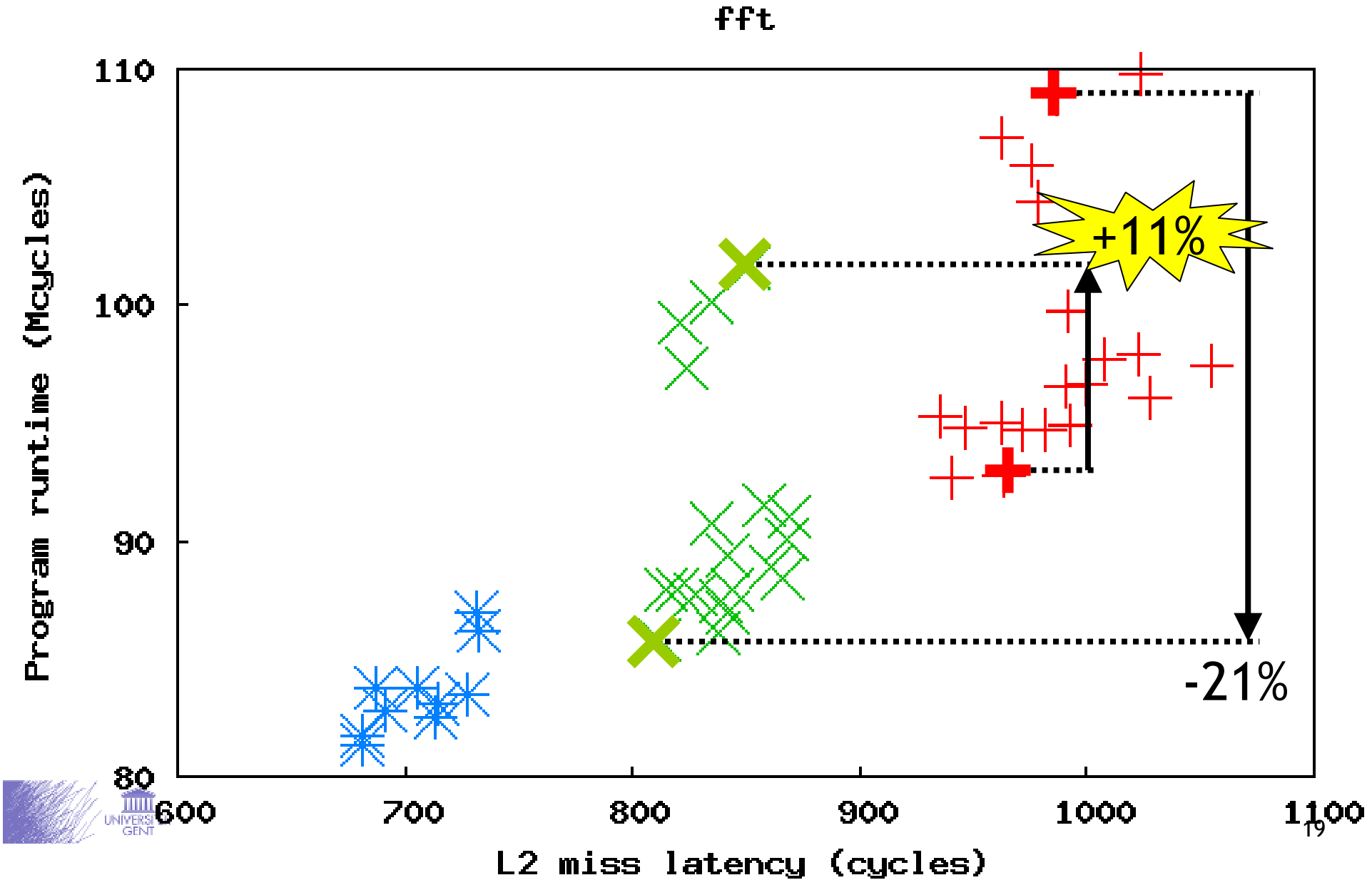
  "torus is better than mesh network"
- Relative accuracy:

  "torus is 10% better than mesh network"
- Variability:
  - Each measurement has confidence interval: $\mu \pm \sigma$
  - Improvement: $(\mu_2 - \mu_1) \pm \sqrt{\sigma_1^{\,2} + \sigma_2^{\,2}}$
  - Performance improvement: 10±5%, 5±5%, 2±5% ?

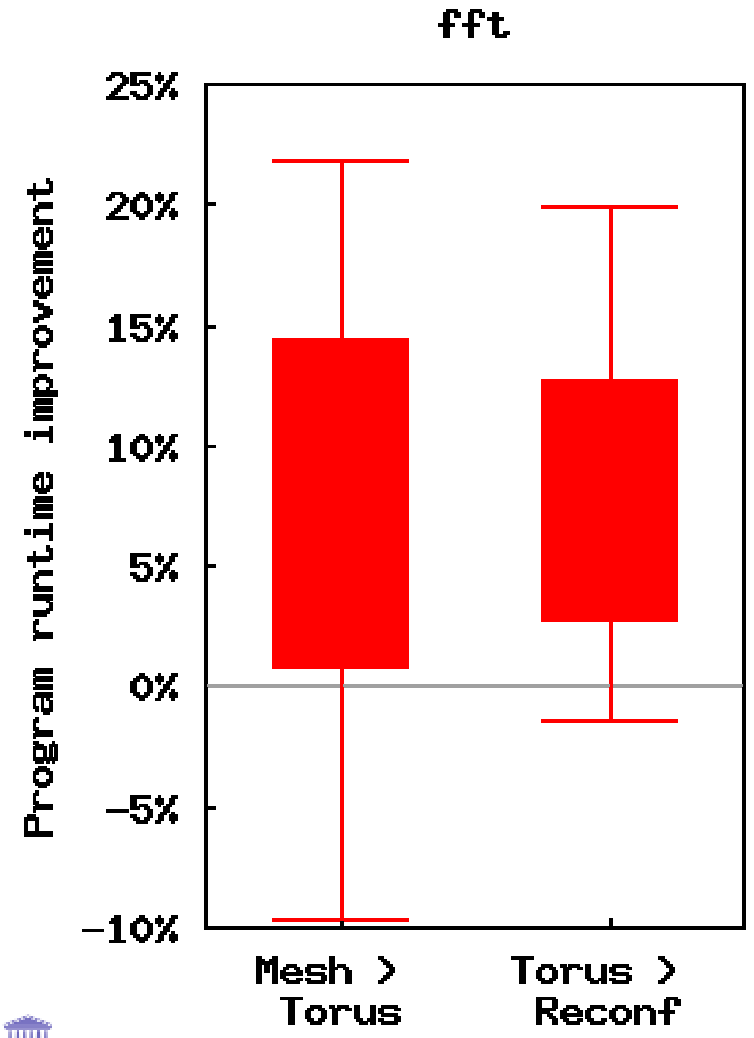
Variation of miss latency improvement

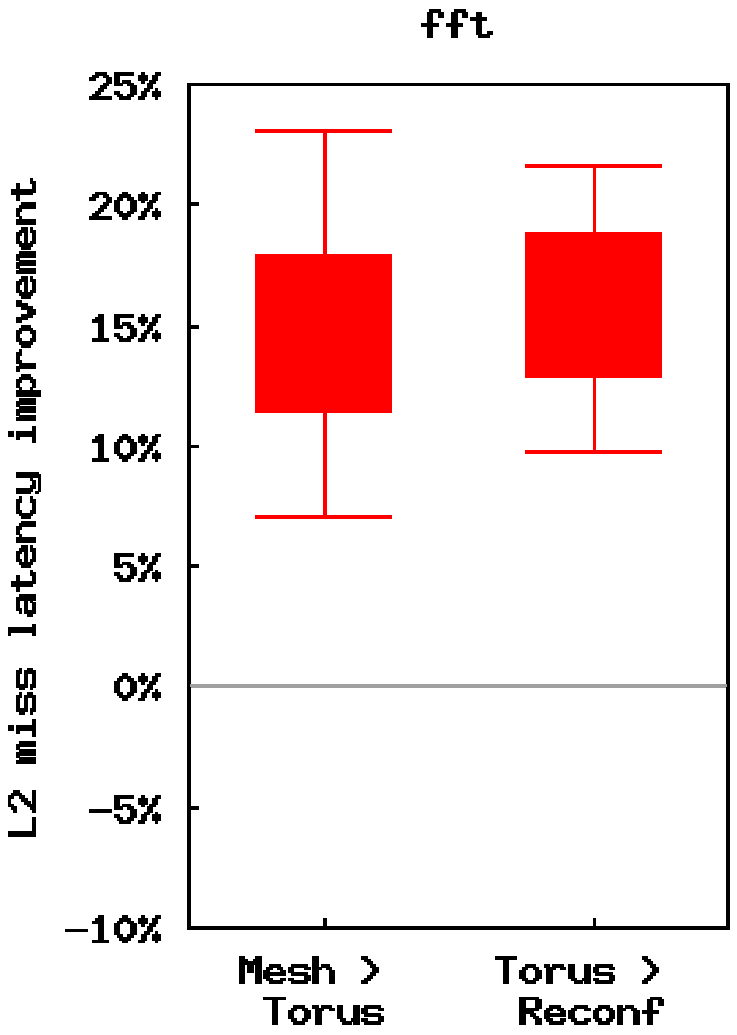


Variation of runtime improvement



Improvement: confidence intervals





Outline

- Variability in simulation: what & why?
- Measurements in SPLASH-2
- Effects on fidelity and relative accuracy
- Overview of possible solutions
- Conclusions

UNIVERSITEIT GENT

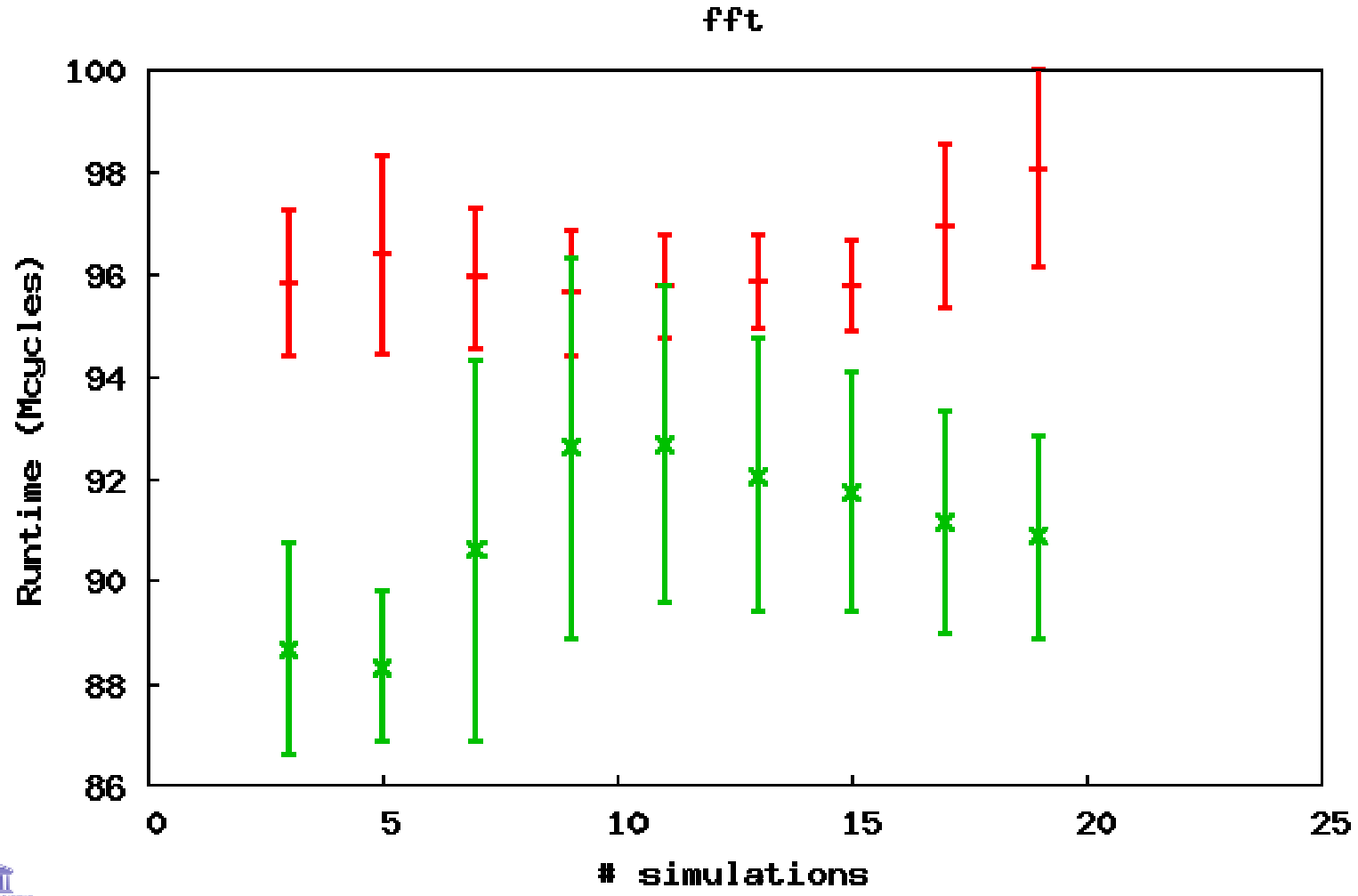
Solution – 'easy' way

- multiple measurements & statistics
- Alameldeen & Wood, HPCA 2003
- N measurements $x_1..x_N$ with average μ and standard deviation σ
- Using Student's t-distribution: t = f(N, p) for instance: N = 4, p = 95%: t = 2.13
- 'Actual' performance will be, with probability *p*, in the range

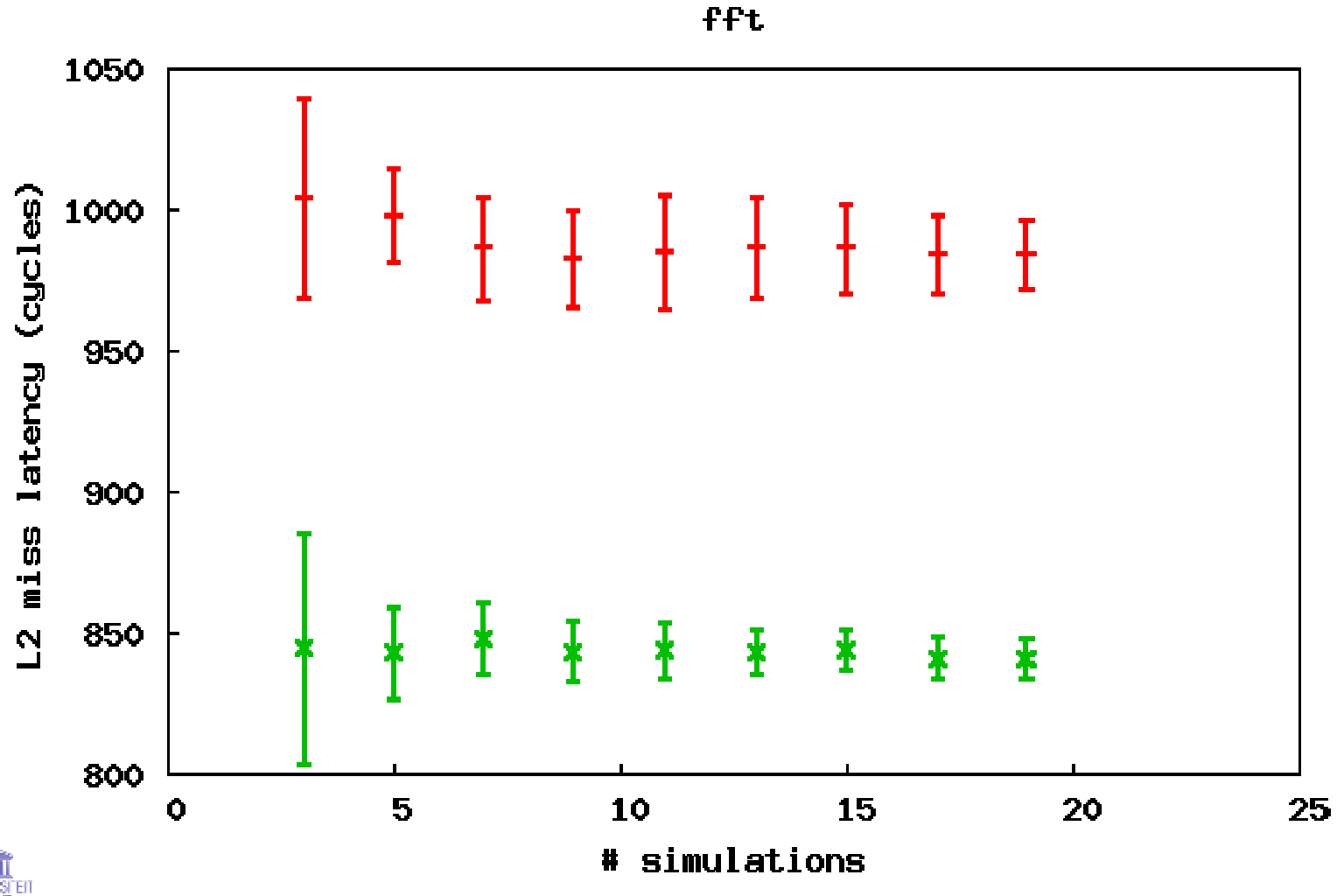
$$[\mu - t \cdot \sigma, \mu + t \cdot \sigma]$$

- Comparing two architectures: simulate until confidence intervals no longer overlap

Multiple simulations: runtime



Multiple simulations: L2 miss latency



Solutions

- Multiple simulations: guarantees (probable) correctness
- But in reality: only limited simulation time available

$$= \frac{\#\text{days to ISCA deadline} \times \#\text{machines available}}{\#\text{benchmarks} \times \#\text{architectures}}$$

Low-level vs. high-level metrics

- Not all metrics are created the same
- Difference in magnitude of
  - variation
  - differentiation between architectures
- Low-level metrics often perform better, but:

  Effect of low-level improvement on high-level performance should always be checked (i.e. communication improvement on non-communicating applications!)

Low-level vs. high-level metrics

Possible methodology:

- Choose a good low-level metric
  (e.g. miss or packet latency for network studies)
- Characterize its variability, and determine the required number of simulations = N
- Run performance measurements N times for each architecture under test and each benchmark

→ Reliable low-level performance per architecture

- Determine relationship between low-level metric and high-level performance *per benchmark*, again using sufficient (but possibly >>N) simulations, but probably only a few architectures are needed

→ Compute reliable high-level performance figures

UNIVERSITEIT GENT

Simulator detail versus input set size

- One reviewer commented:

"*use of in-order processor simulation influences results*"

- Emergence of a trade-off:

  Detailed simulator (high accuracy)

  vs.

  Large data set / multiple runs (low variability)

- Lowest combined error ??
  (for given simulation time)

Matched-pair sample comparison

SimFlex [Wenisch et. al., MICRO‘06]

- Sampling methodology
- For one architecture: full simulation, store some architectural state at fixed points
- Subsequent architectures: sample, each can load state to avoid long warm-up period
- E.g.: network simulation: store cache state (long-living), reconstruct network state (buffer contents, short-living)

Matched-pair sample comparison

- Avoids variability:
  - each sample is started from the same state
  - short simulations so no time for variability to build op
- Drawbacks:
  - uses sampling (which sample set? bias?)
  - state not always re-usable for all types of architectural studies

Outline

- Variability in simulation: what & why?
- Measurements in SPLASH-2
- Effects on fidelity and relative accuracy
- Overview of possible solutions
- Conclusions

UNIVERSITEIT GENT

Conclusions

- Variability in parallel programs is a problem, also in scientific benchmarks
- Up to 10% variation in program runtime, IPC, miss latency
- Influence on absolute accuracy
- Much higher influence on performance improvement measurements
- Take care when comparing architectures!
- No definitive solution; improvements through multiple simulations, use of other metrics, matched-pair sample comparison