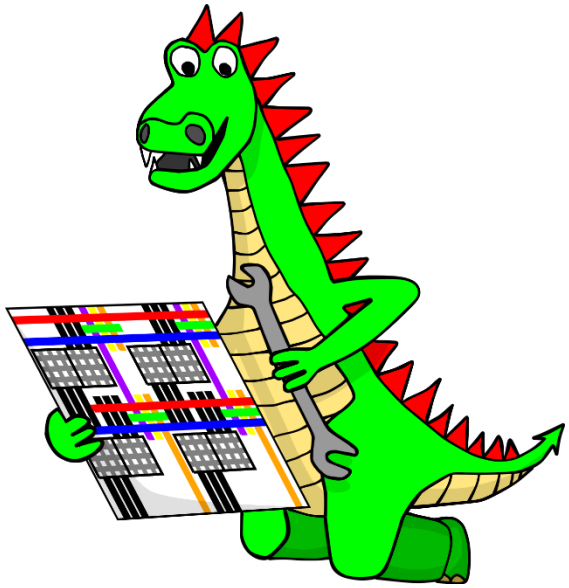
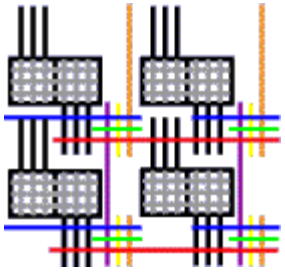
Penn



Quality-Time Tradeoffs in Component-Specific Mapping

Or: How to Train Your Dynamically Reconfigurable Array of Gates with Outrageous Network-delays



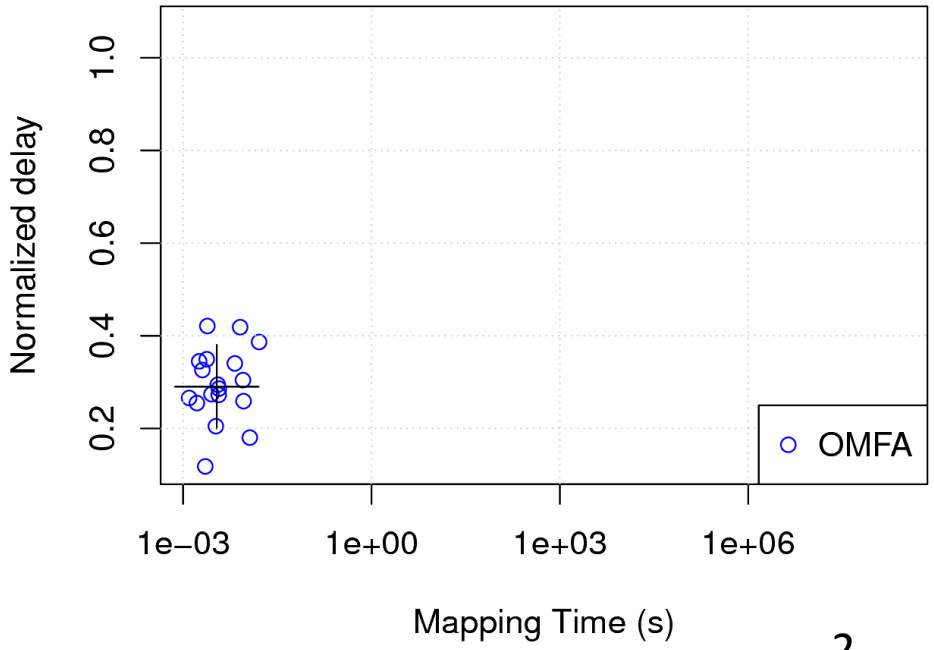
Hans Giesen, Raphael Rubin,
Benjamin Gojman, André DeHon

University of Pennsylvania

February 23, 2017

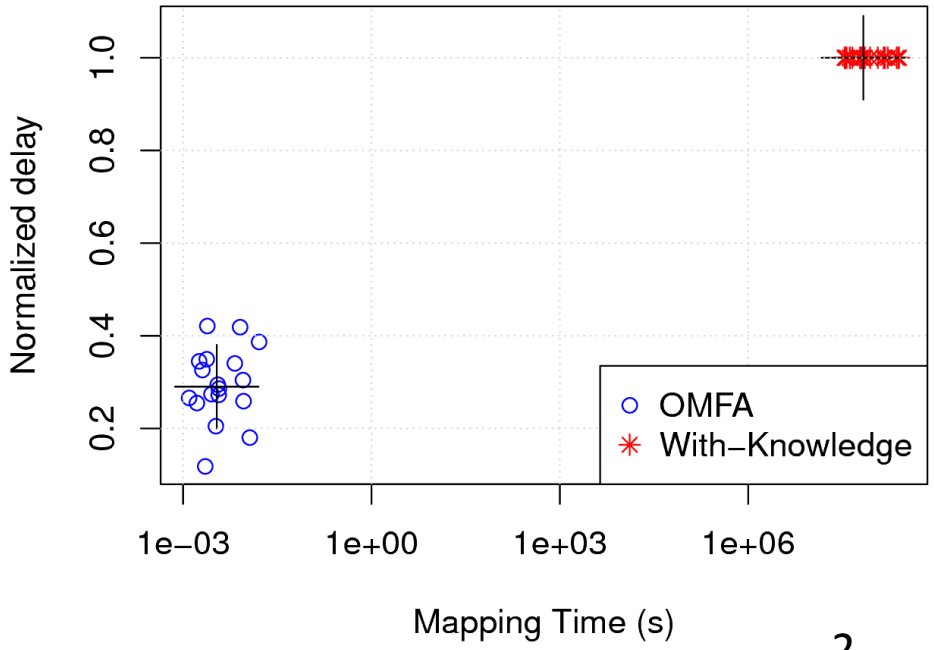
Component-Specific Mapping

- Traditionally: Same bitstream for every chip.
  - Poor performance and energy



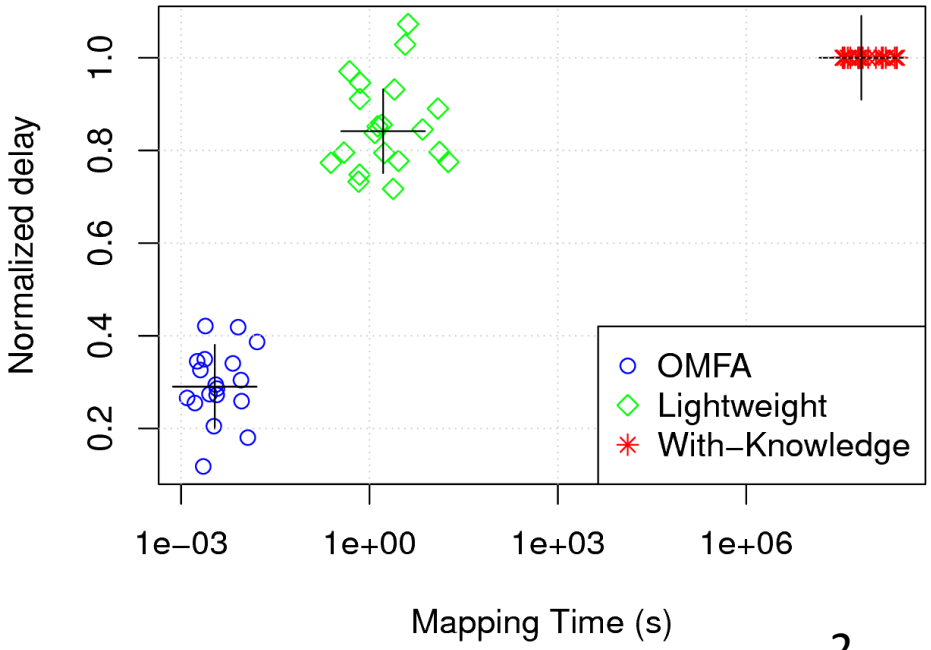
Component-Specific Mapping

- Traditionally: Same bitstream for every chip.
  - Poor performance and energy
- Mapping using actual delays reduces delay and energy.
  - Mapping can take days.



Component-Specific Mapping

- Traditionally: Same bitstream for every chip.
  - Poor performance and energy
- Mapping using actual delays reduces delay and energy.
  - Mapping can take days.
- Lightweight mapping combines best of both.
  - 77% of maximum delay benefit is achievable in 18 seconds.



Our Contributions

- We achieved a lightweight mapping.
  - Using online measurement
  - Adapting precomputed alternatives

Our Contributions

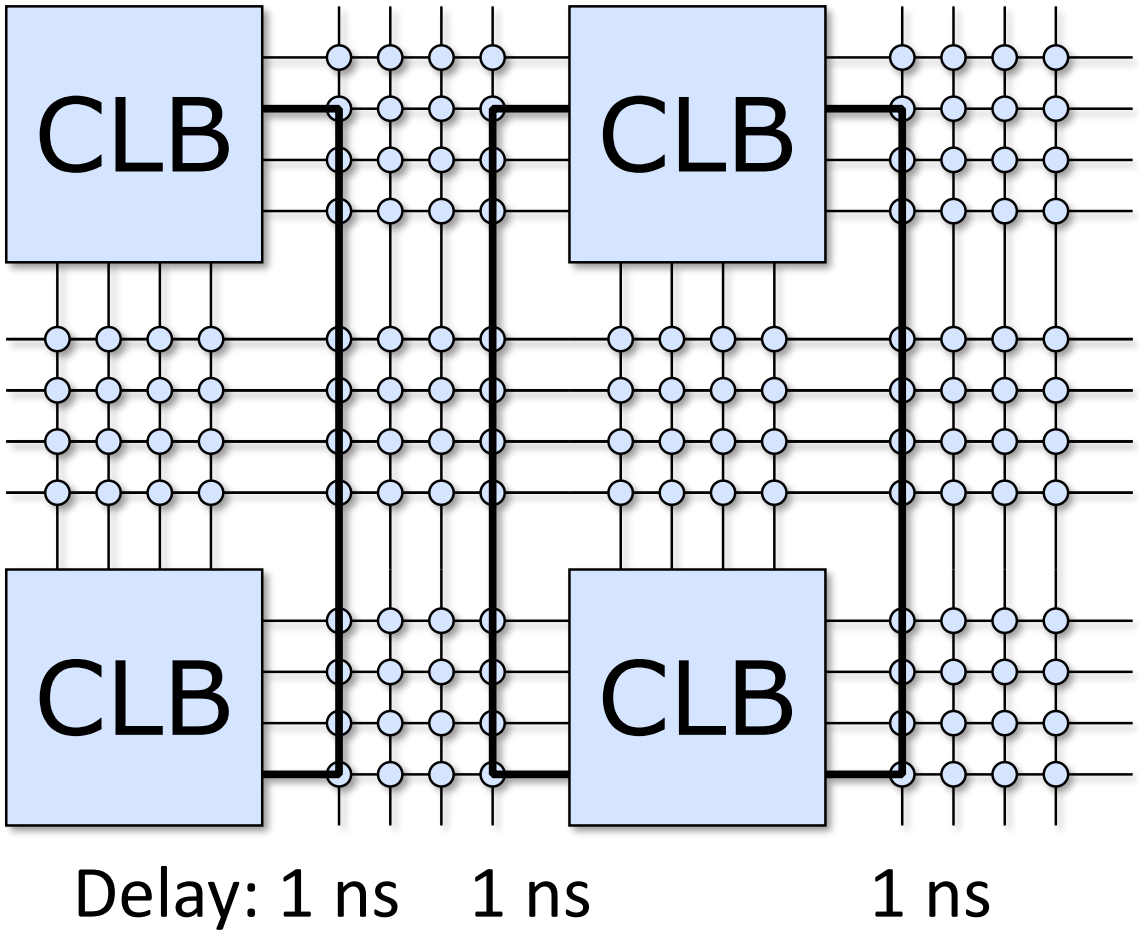
- We achieved a lightweight mapping.
  - Using online measurement
  - Adapting precomputed alternatives
- We developed 5 algorithms at intermediate points.
  - Characterized time and quality for all algorithms.

Overview

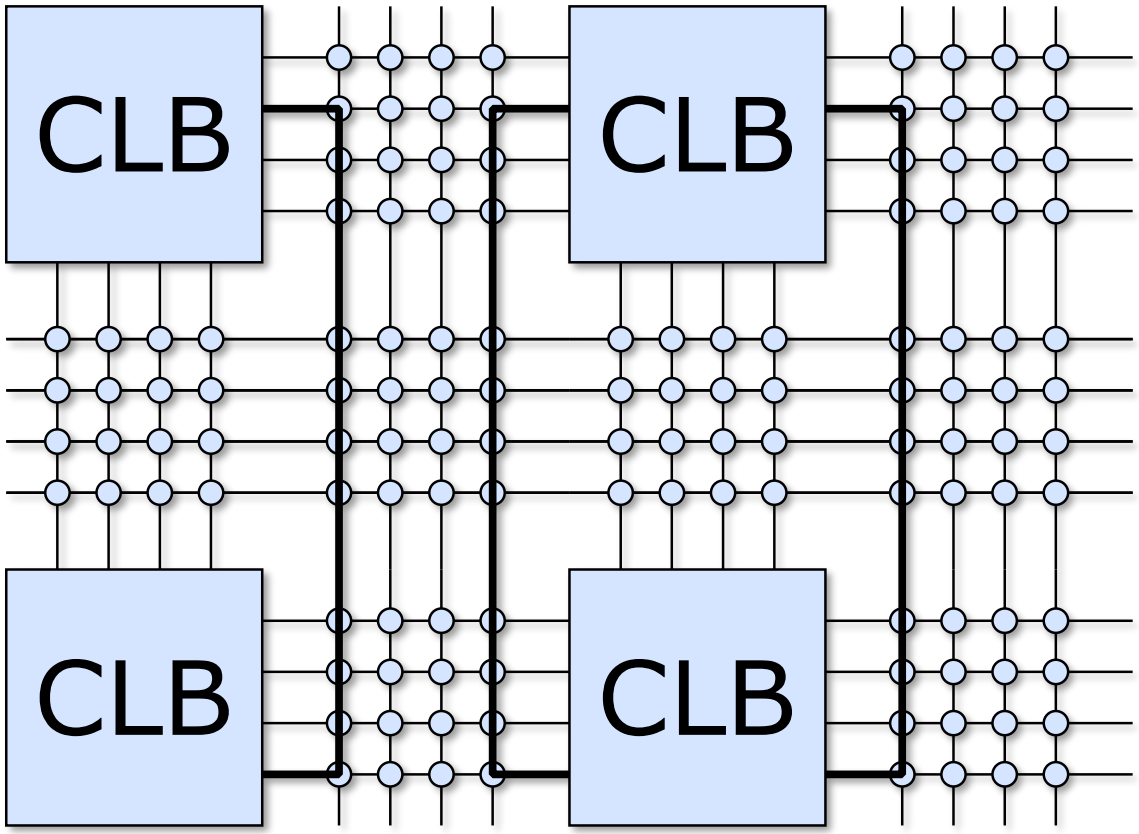
- Introduction
- Lightweight solutions
- Experiments
- Conclusions

Introduction

Mapping – CAD View



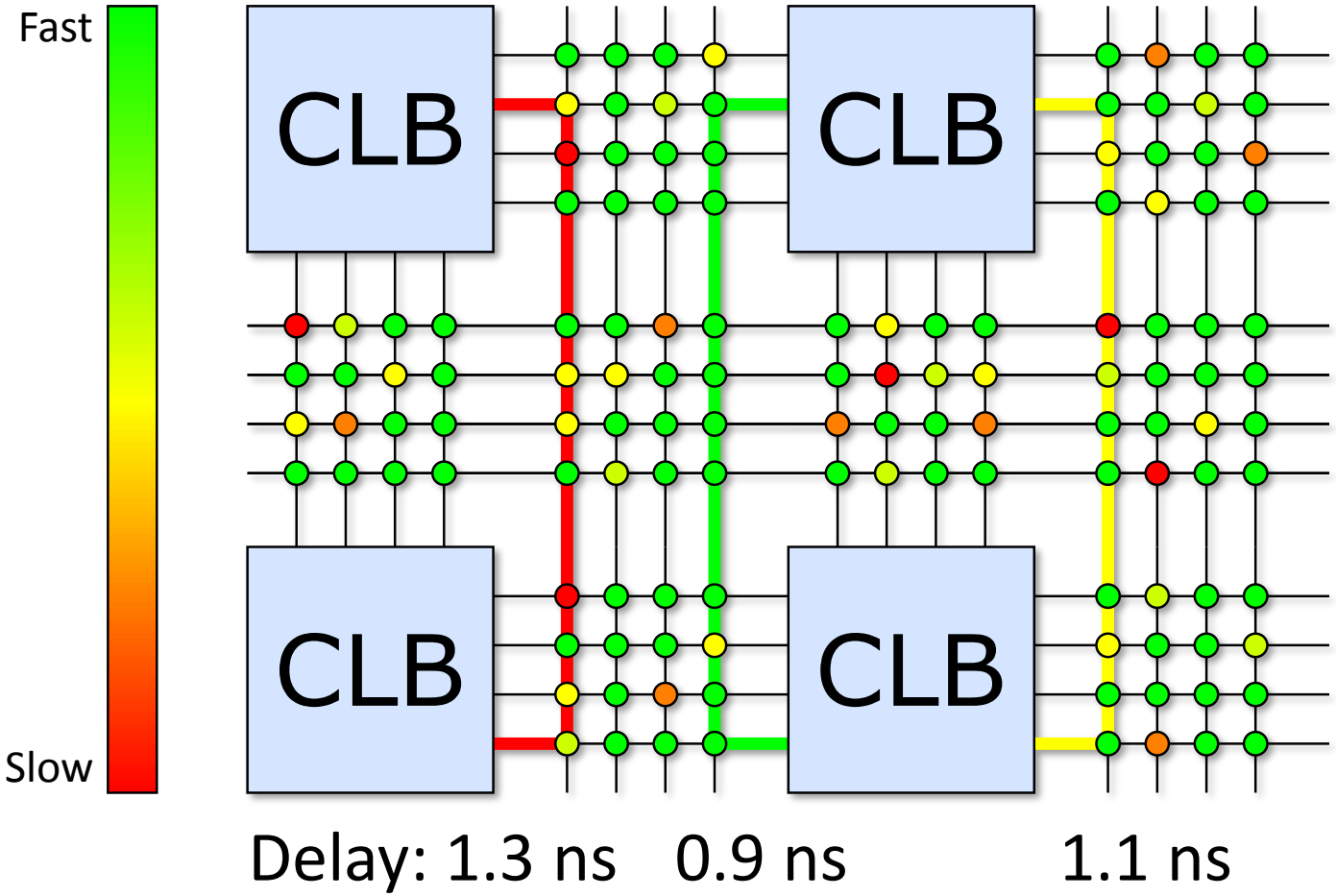
Mapping – CAD View



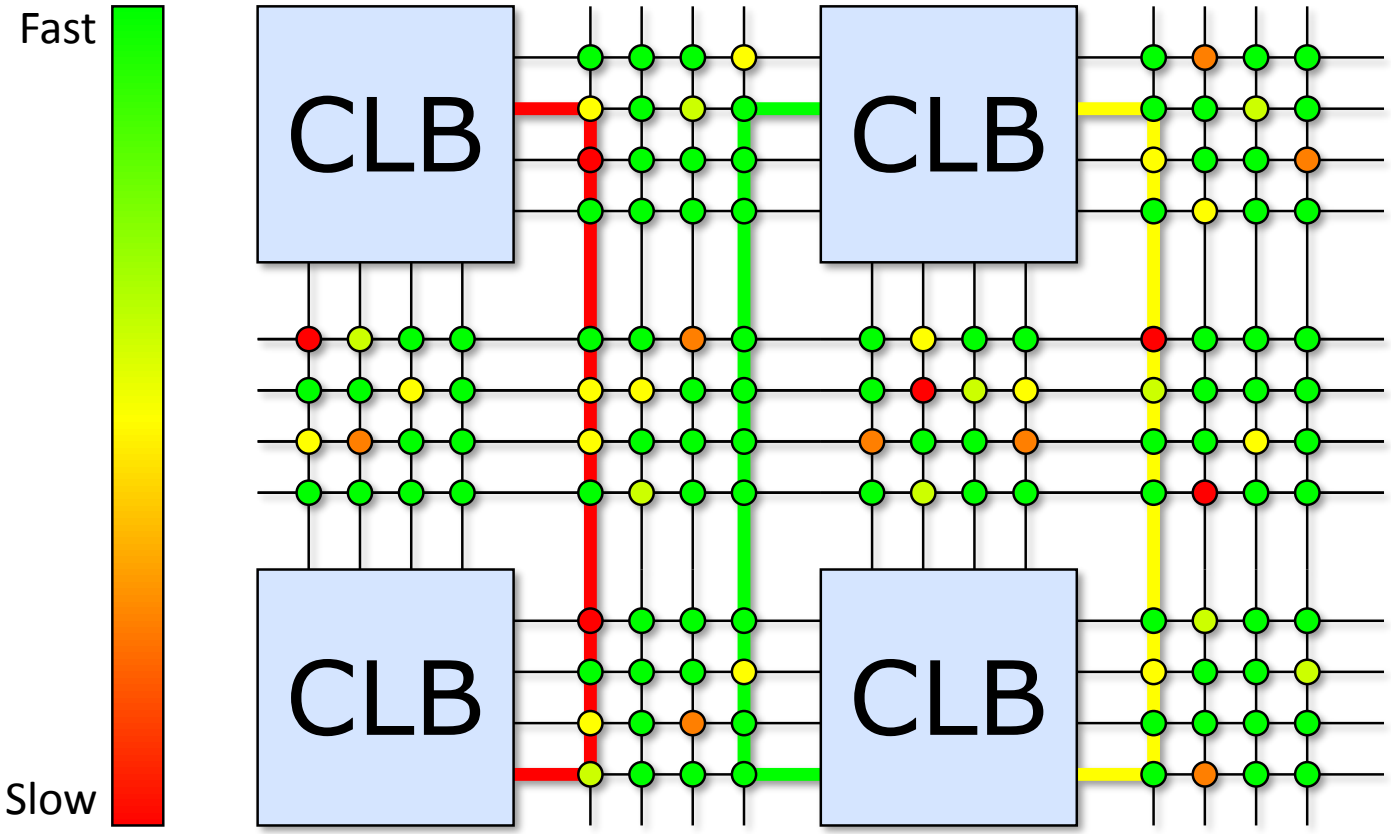
Delay: 1 ns 1 ns 1 ns

Clock period: 1 ns

Process Variation



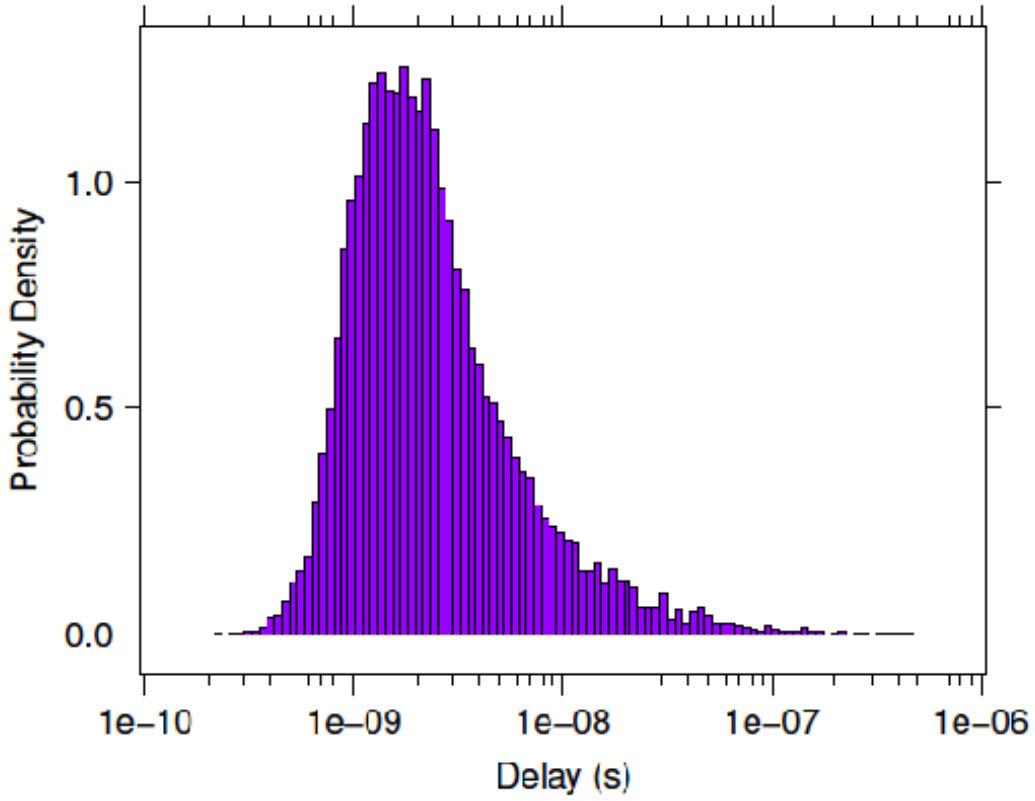
Process Variation



Delay: 1.3 ns 0.9 ns 1.1 ns

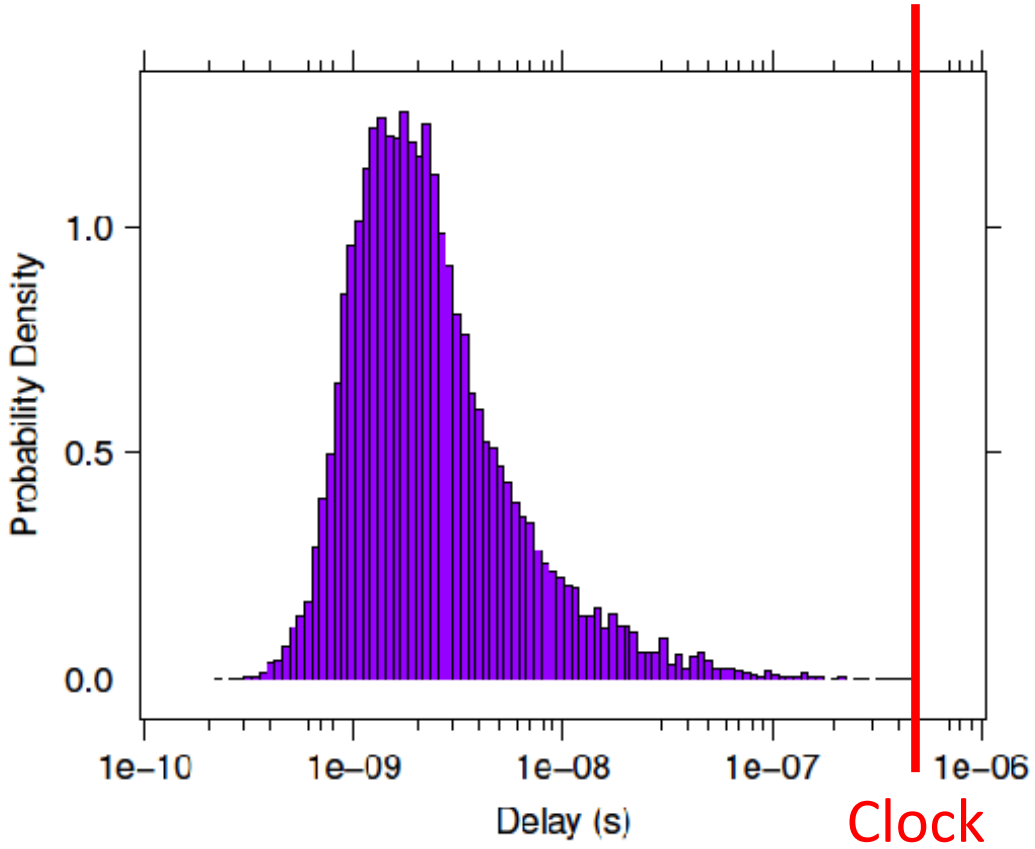
Clock period: 1.3 ns

Delay Distribution



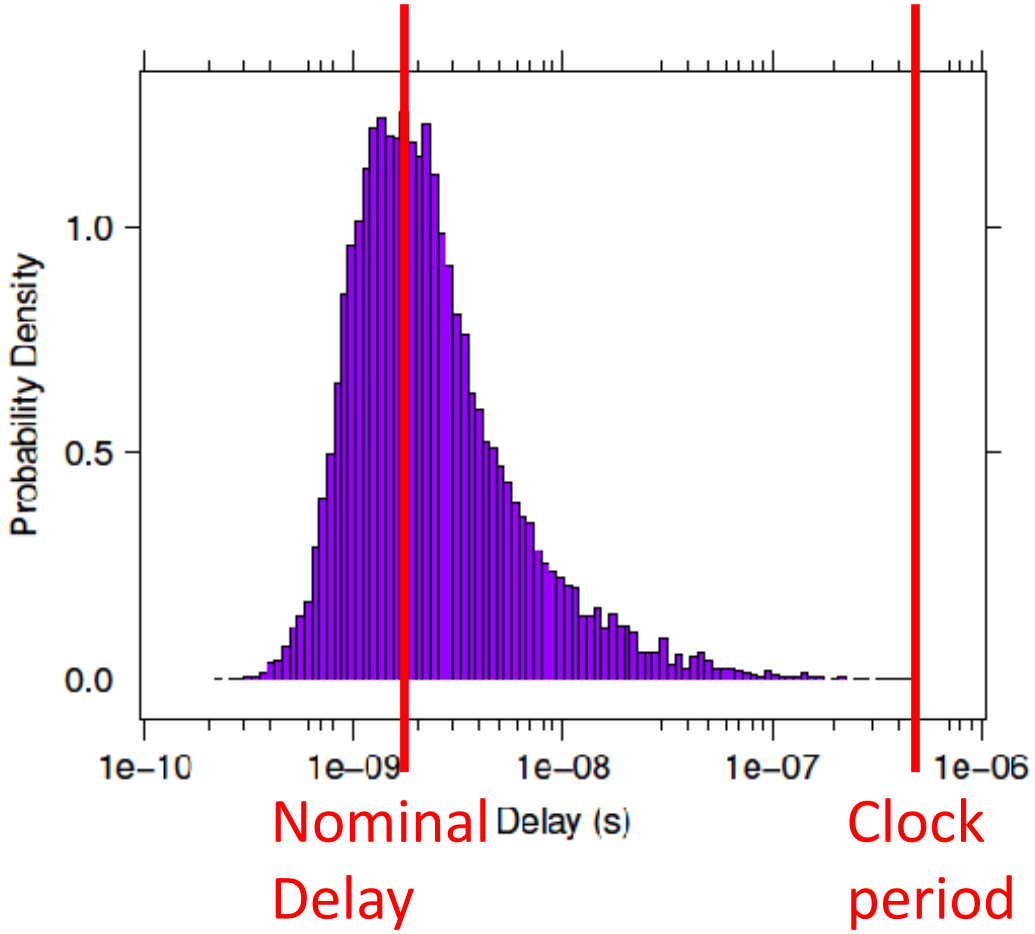
Delay Distribution

- One-Mapping-Fits-All (OMFA)



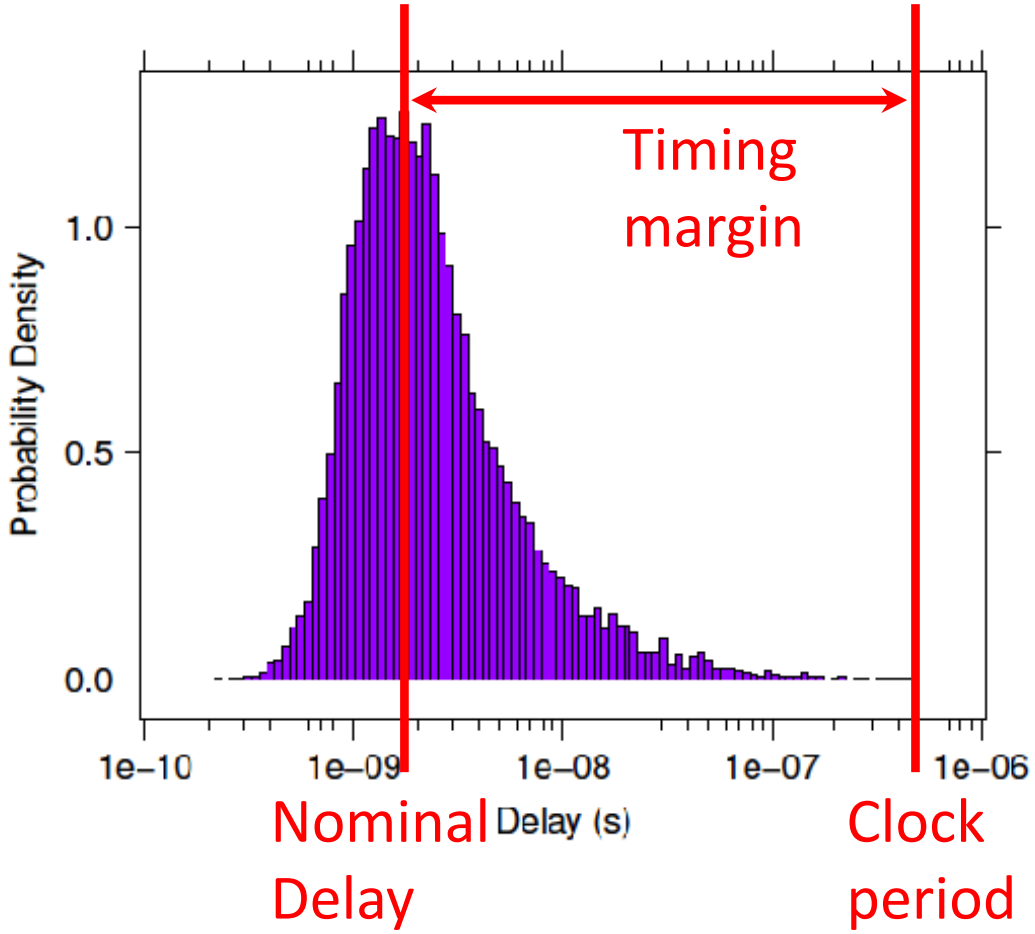
Delay Distribution

- One-Mapping-Fits-All (OMFA)



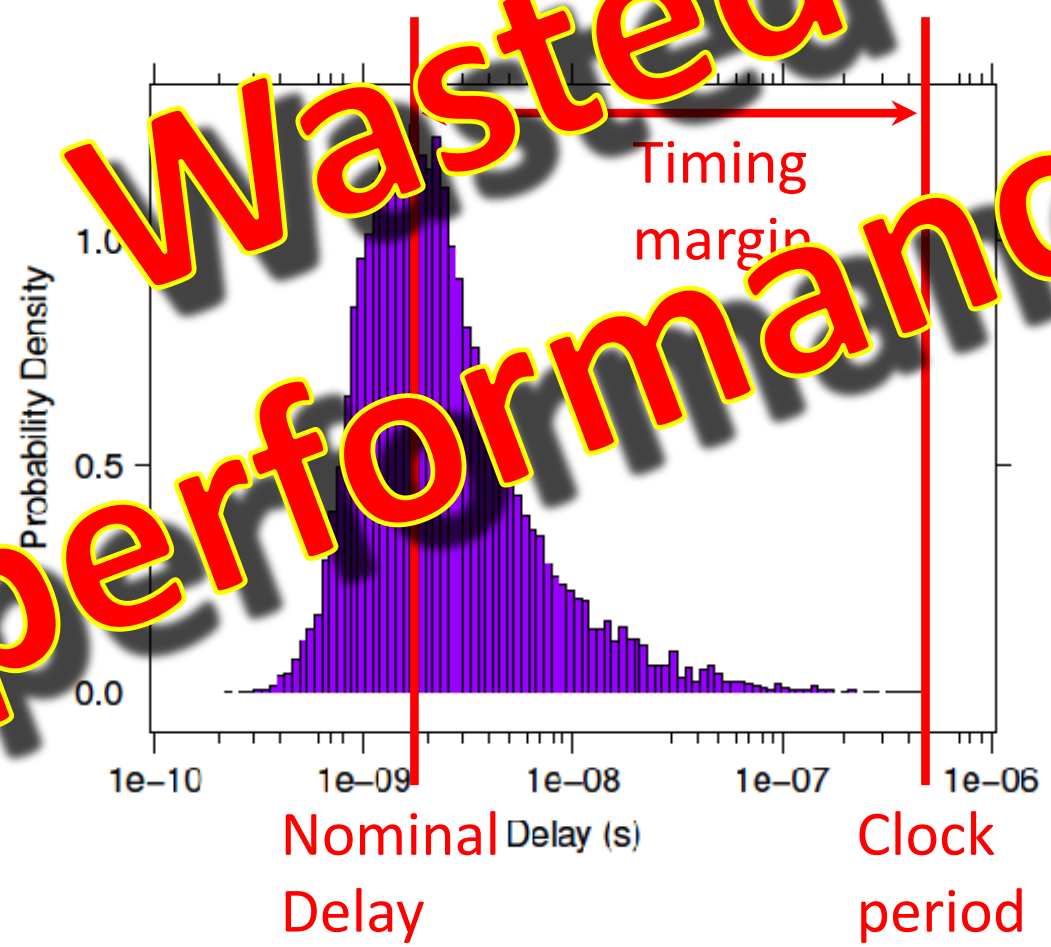
Delay Distribution

- One-Mapping-Fits-All (OMFA)

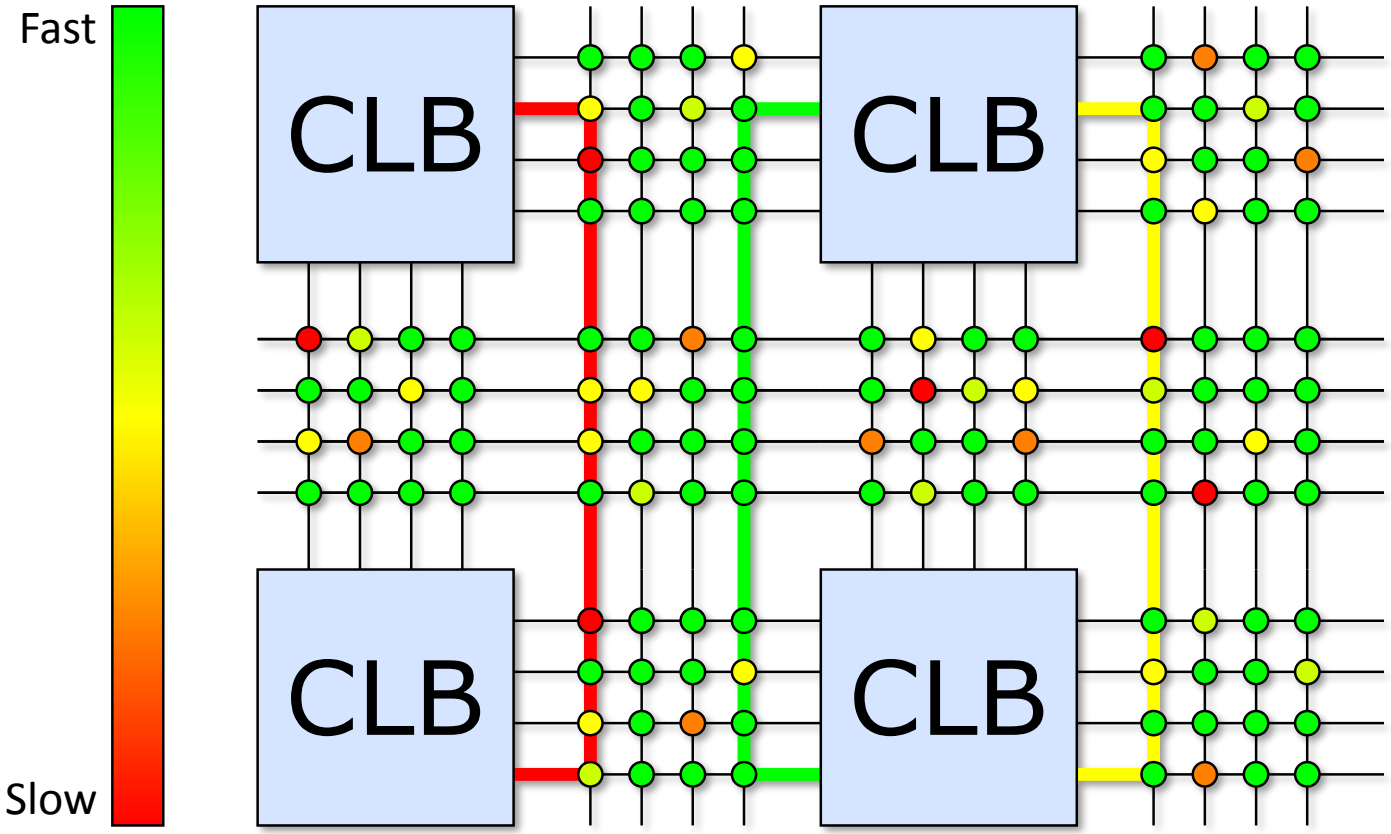


Delay Distribution

- One-Mapping-Fits-All (OMFA)



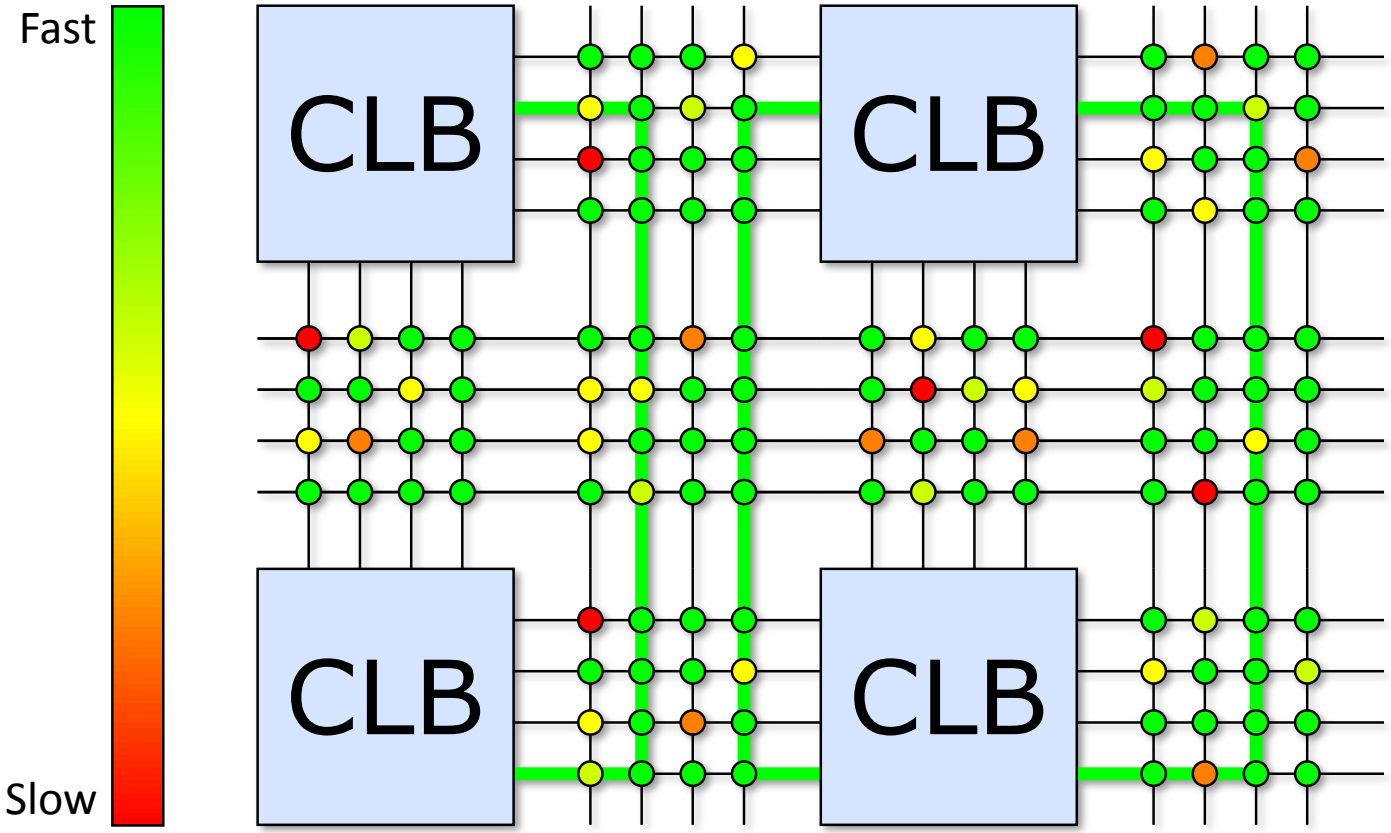
Full-Knowledge (FK) Mapping



Delay: 1.3 ns 0.9 ns 1.1 ns

Clock period: 1.3 ns

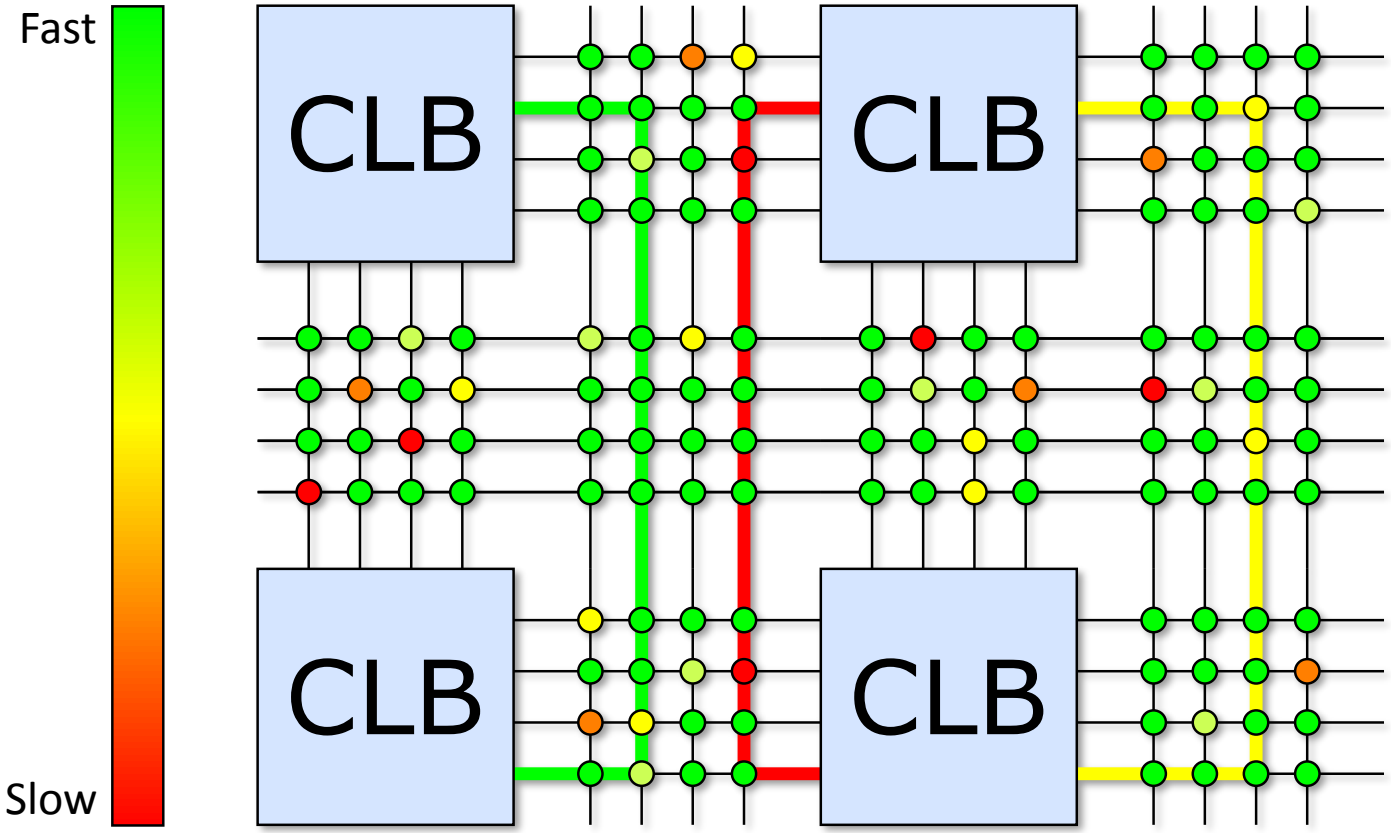
Full-Knowledge (FK) Mapping - Repaired



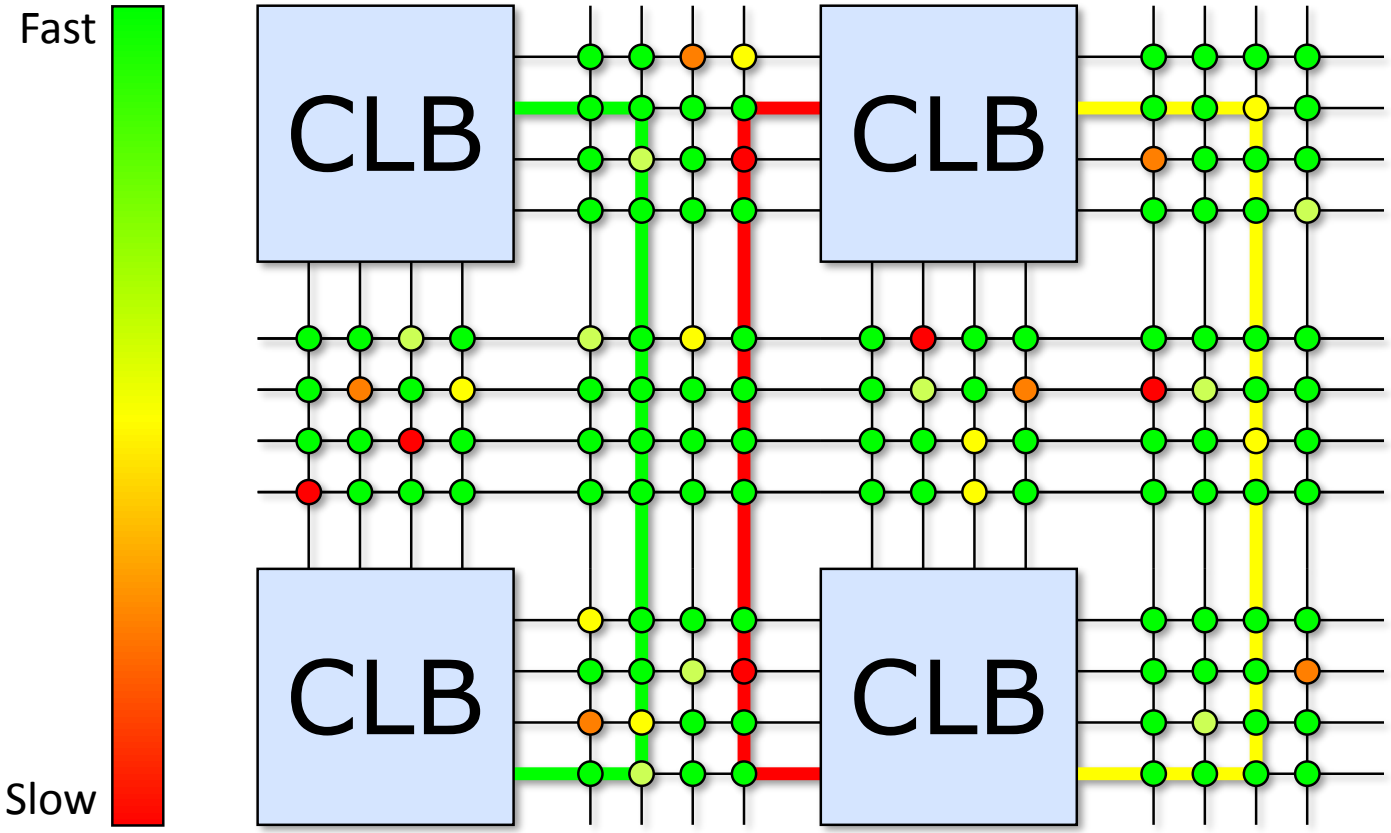
Delay: 1 ns 0.9 ns 0.9 ns

Clock period: 1 ns

Different Chip



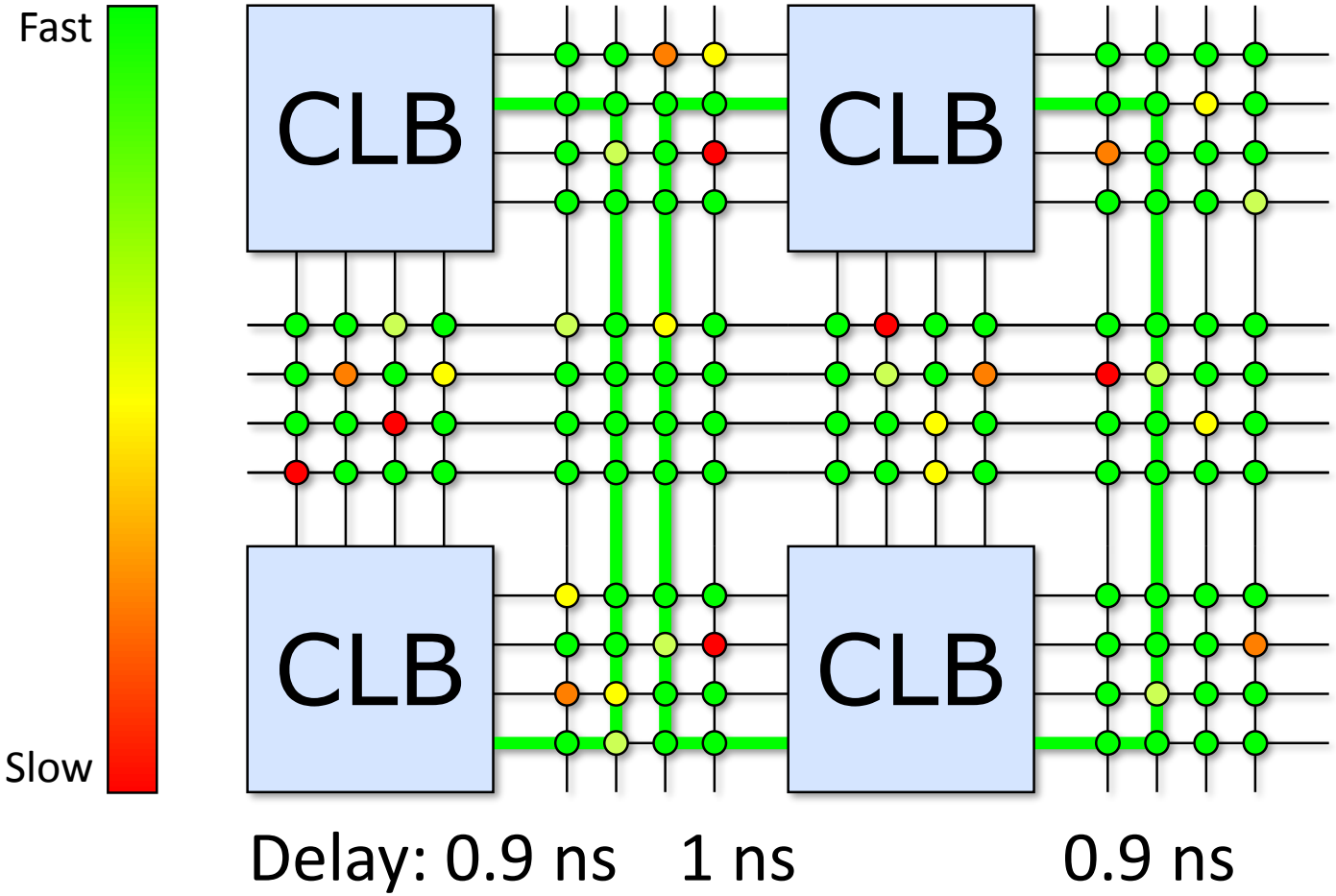
Different Chip



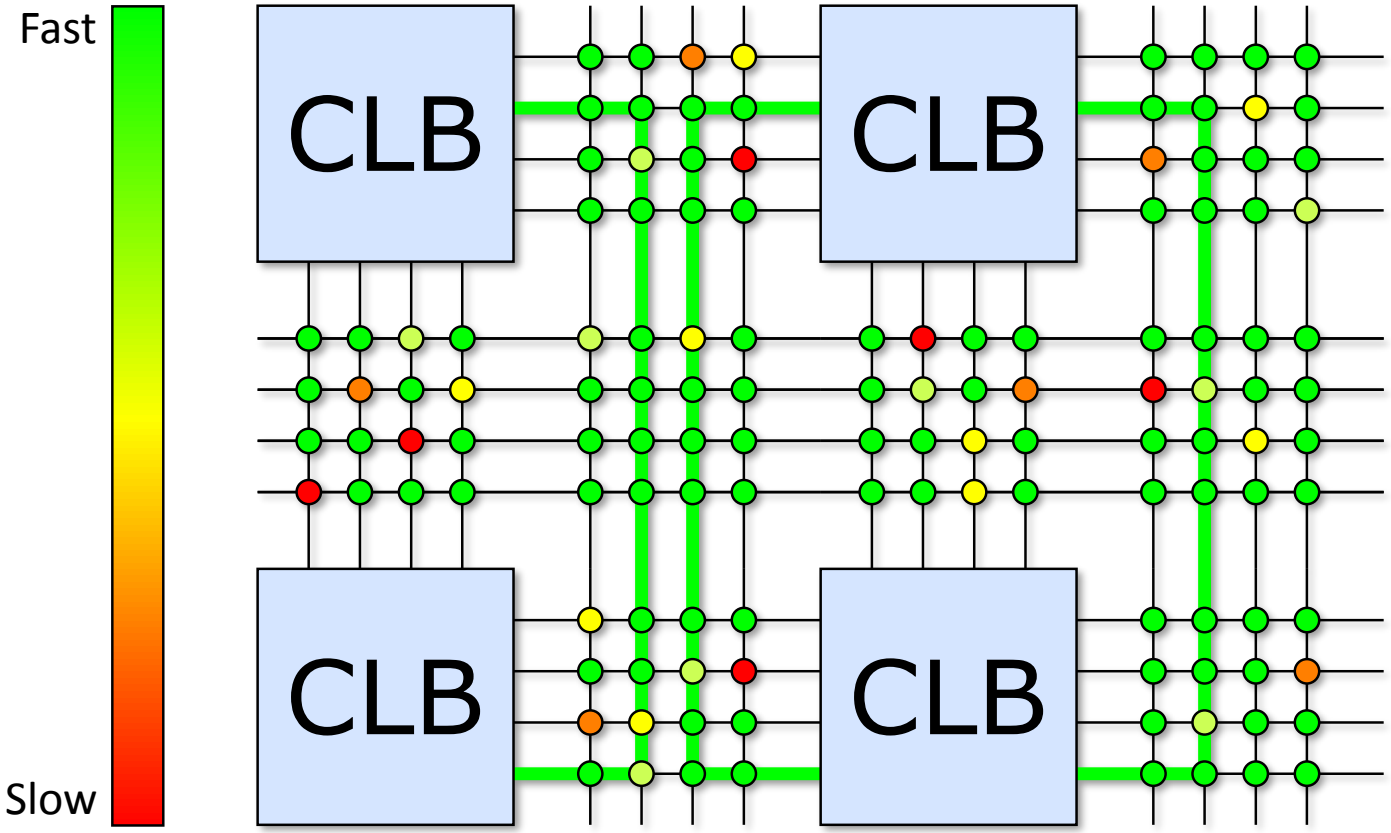
Delay: 0.9 ns 1.3 ns 1 ns

Clock period: 1.3 ns

Different Chip - Repaired



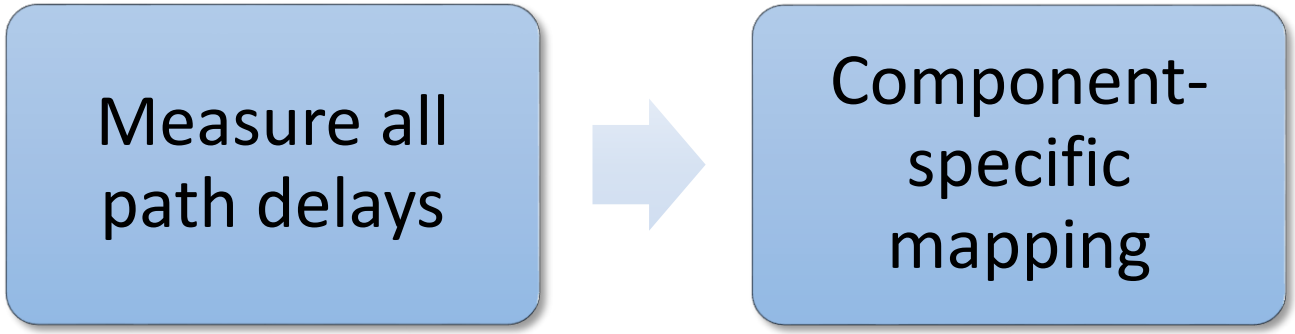
Different Chip - Repaired



Delay: 0.9 ns 1 ns 0.9 ns

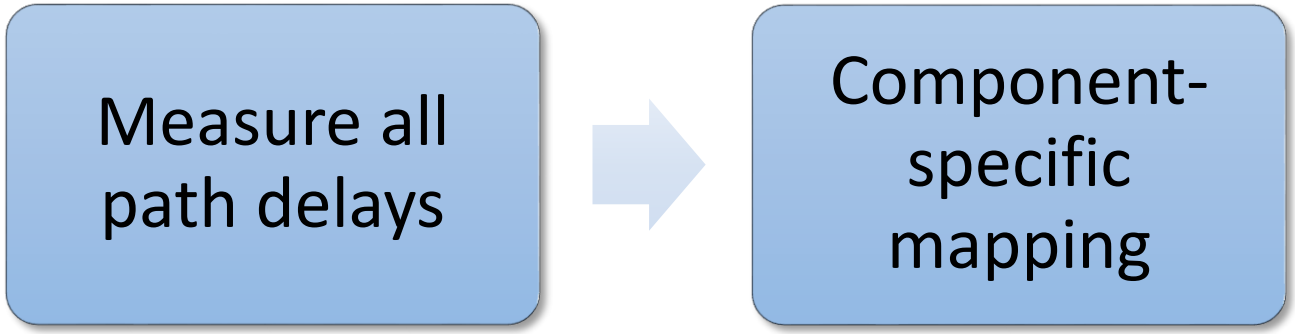
Clock period: 1 ns

Full-Knowledge Algorithm



Full-Knowledge Algorithm

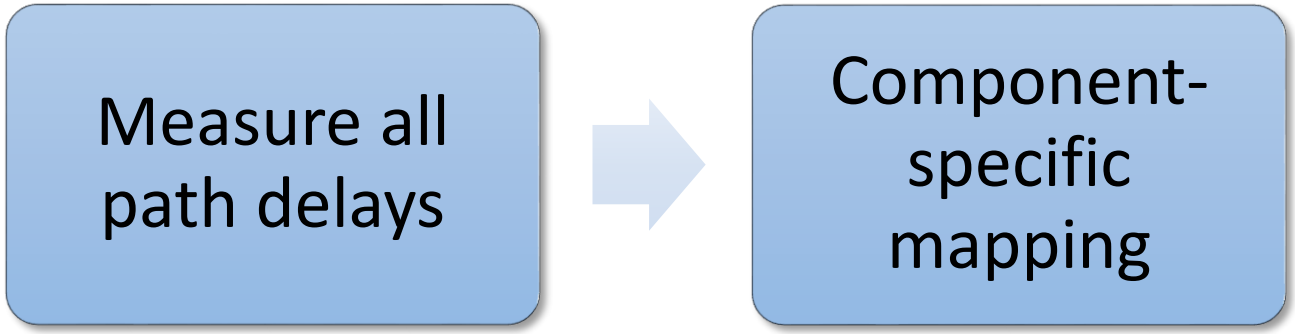
using GROK-LAB
or GROK-INT [Gojman, 2014]



Full-Knowledge Algorithm

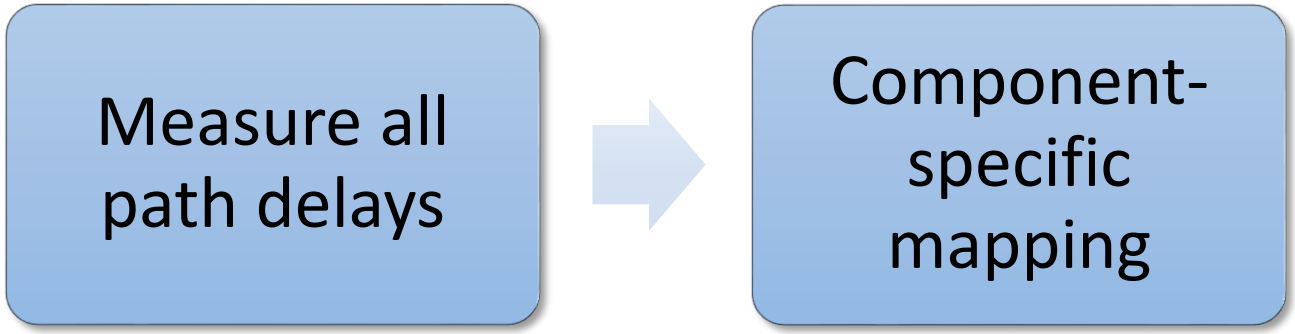
using GROK-LAB
or GROK-INT [Gojman, 2014]

using PathFinder



Full-Knowledge Algorithm

using GROK-LAB
or GROK-INT [Gojman, 2014]



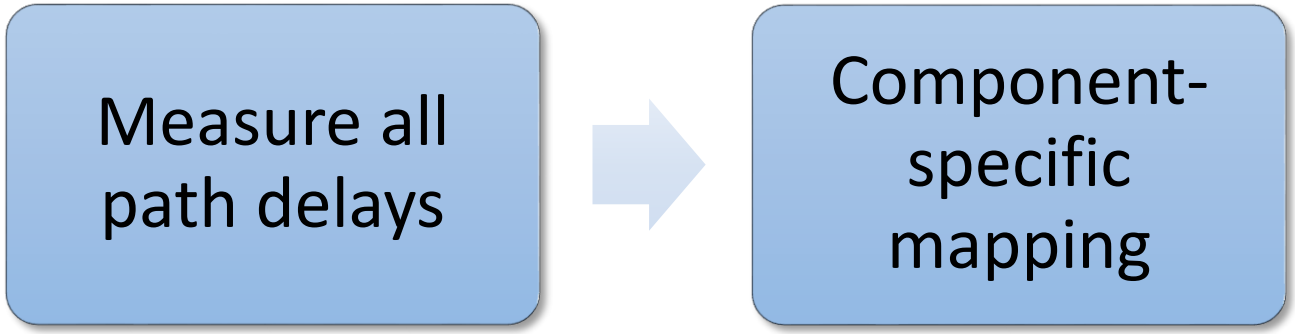
using PathFinder

Can take days

Full-Knowledge Algorithm

using GROK-LAB
or GROK-INT [Gojman, 2014]

using PathFinder



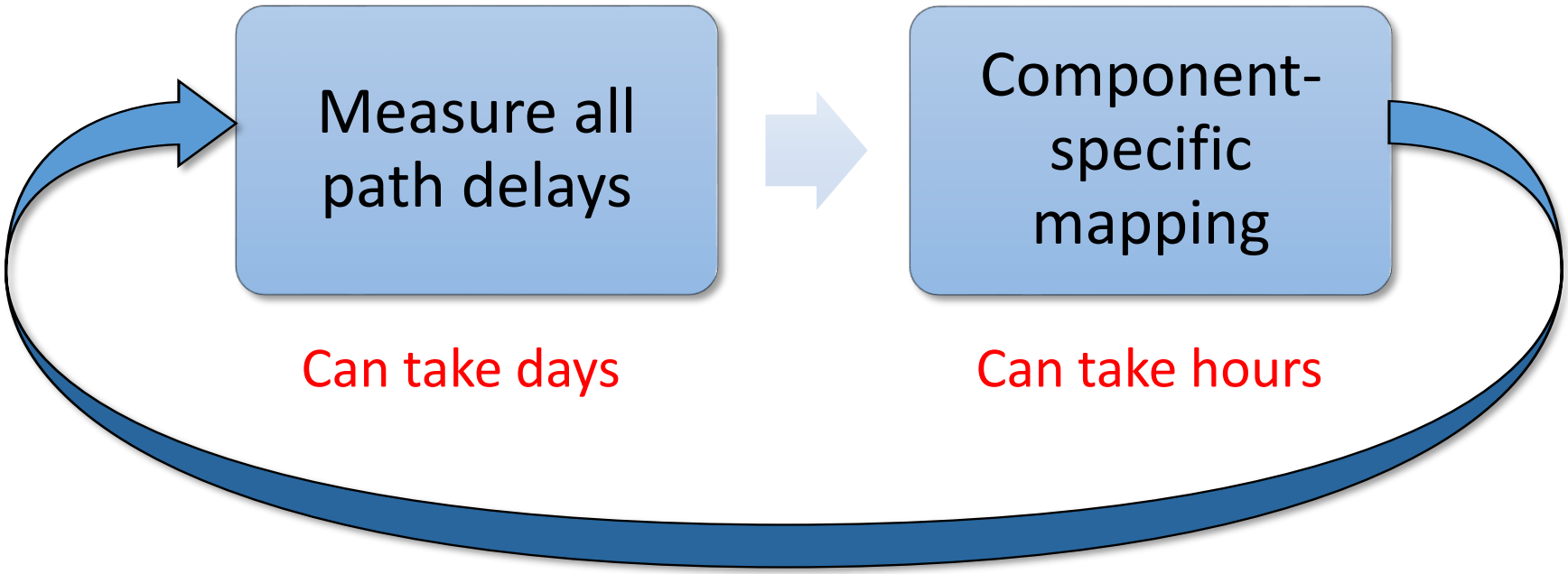
Can take days

Can take hours

Full-Knowledge Algorithm

using GROK-LAB
or GROK-INT [Gojman, 2014]

using PathFinder

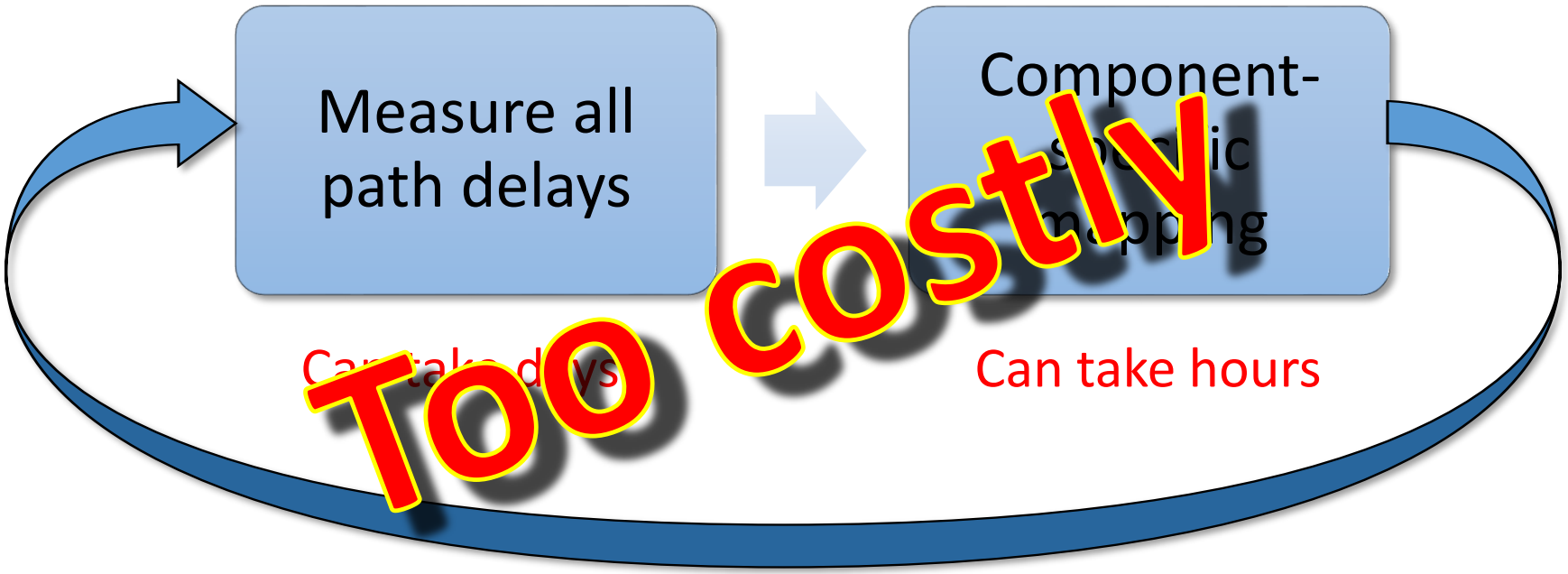


For *every* chip

Full-Knowledge Algorithm

using GROK-LAB
or GROK-INT [Gojman, 2014]

using PathFinder

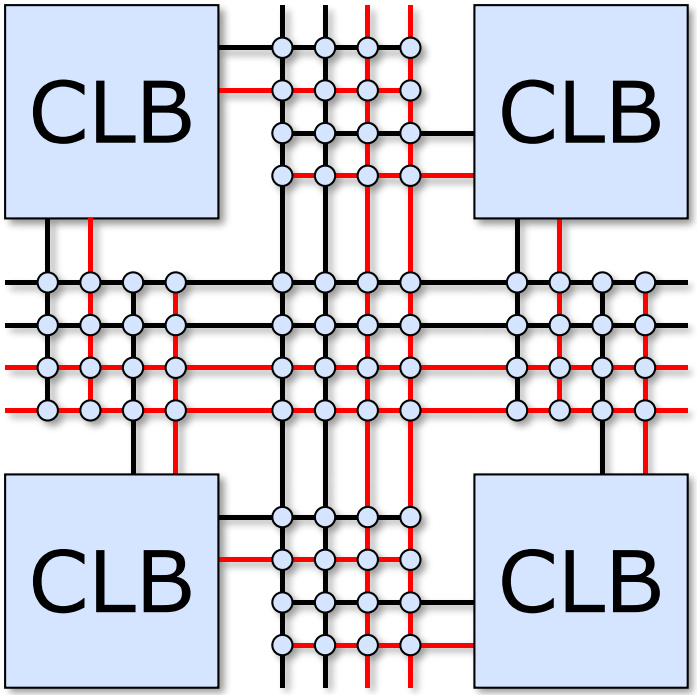


For *every* chip

Lightweight solutions

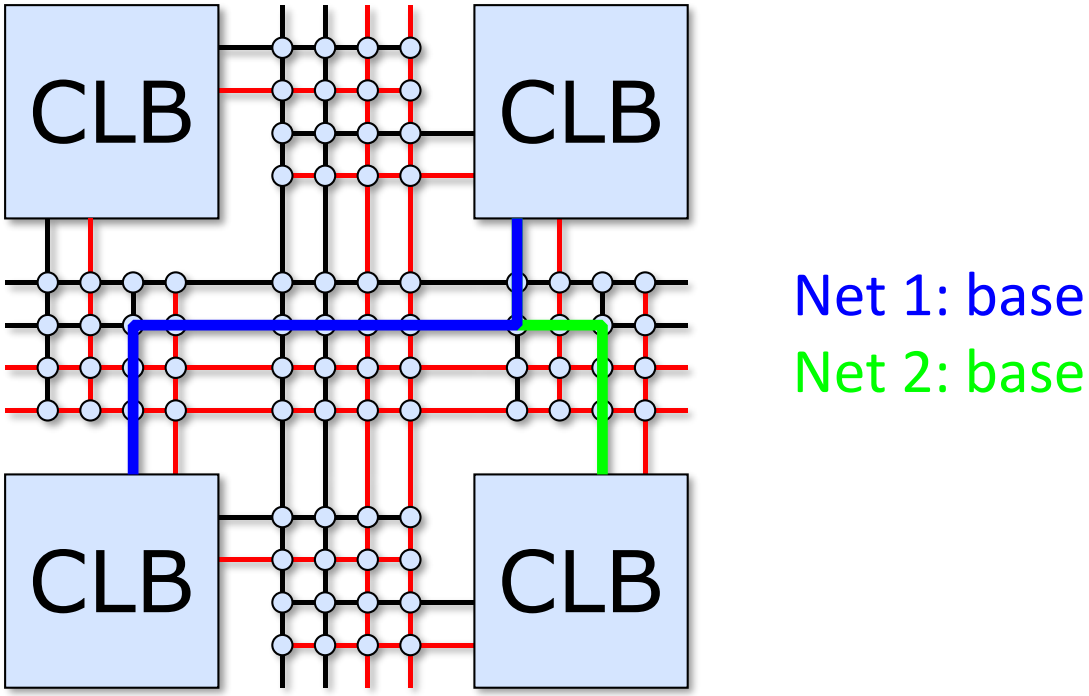
Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.



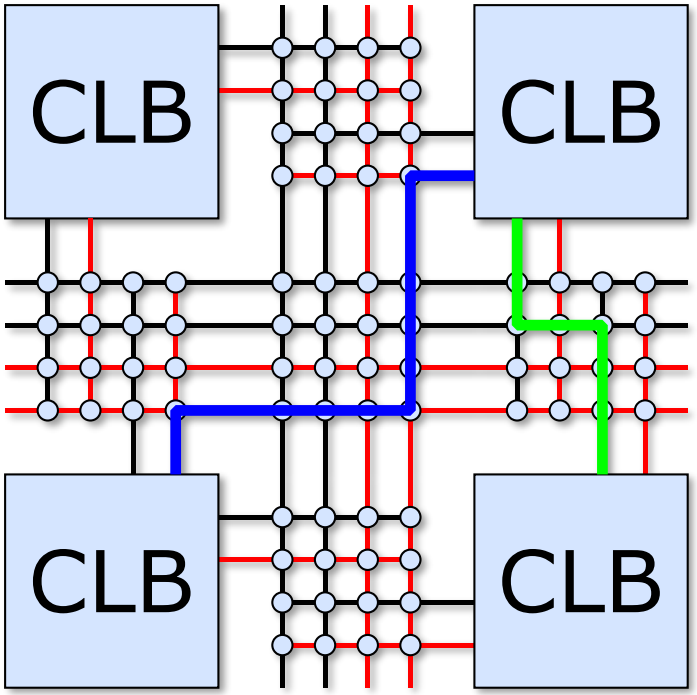
Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.



Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.

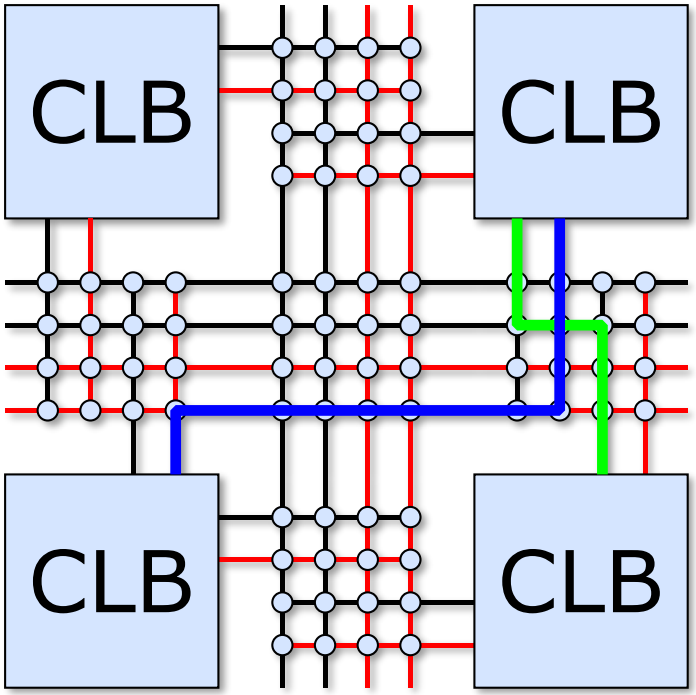


Net 1: alternative 1

Net 2: base

Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.

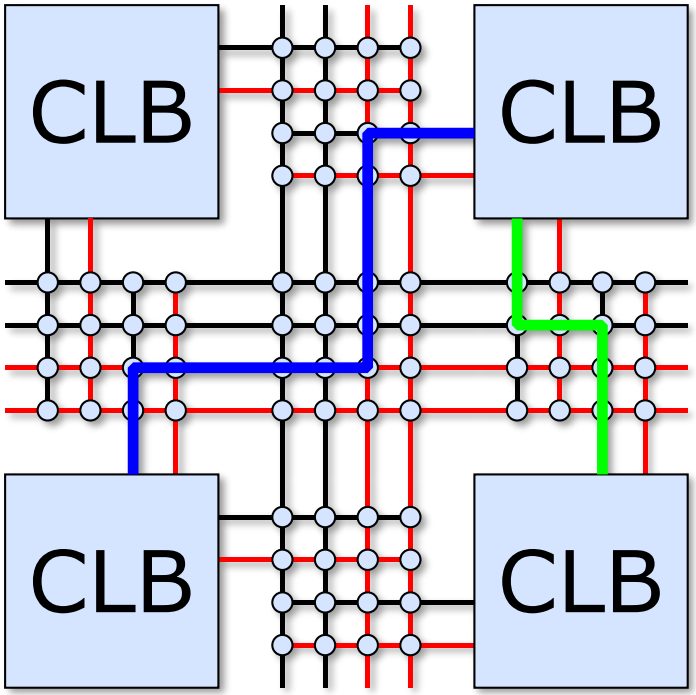


Net 1: alternative 2

Net 2: base

Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.

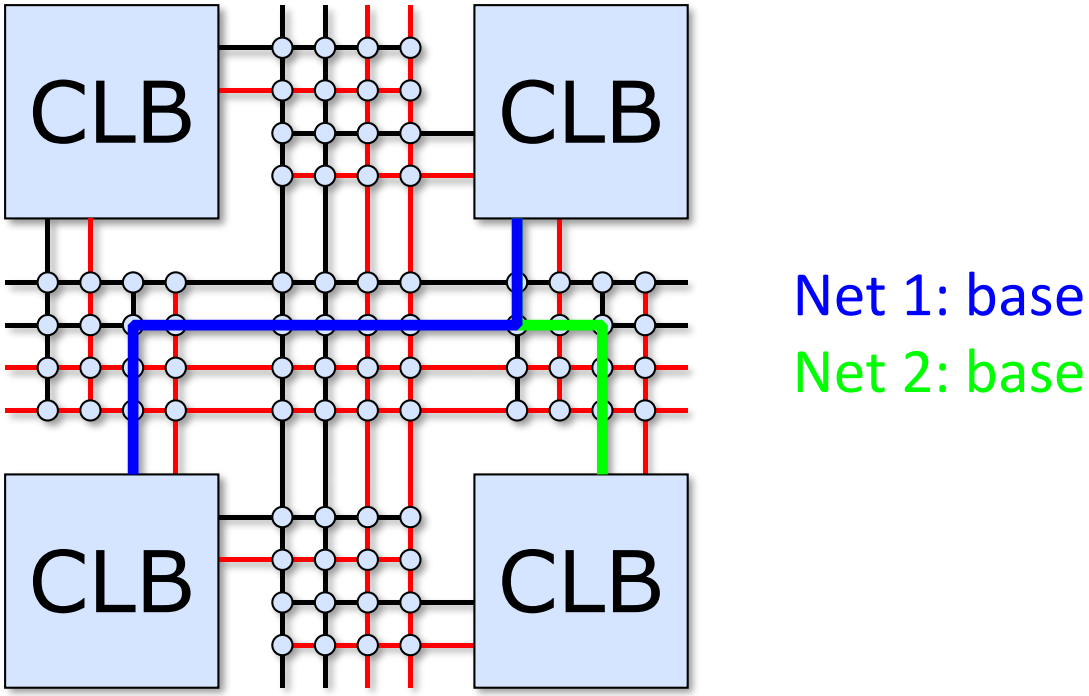


Net 1: alternative 3

Net 2: base

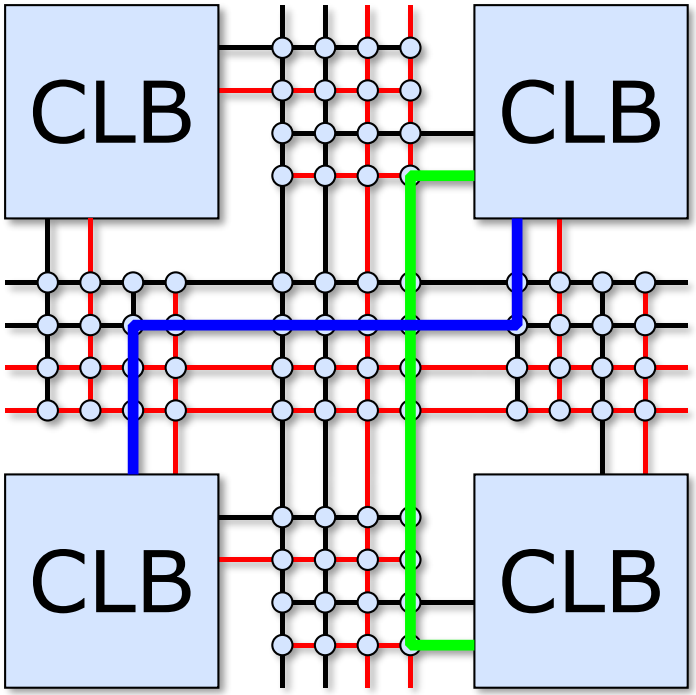
Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.



Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.

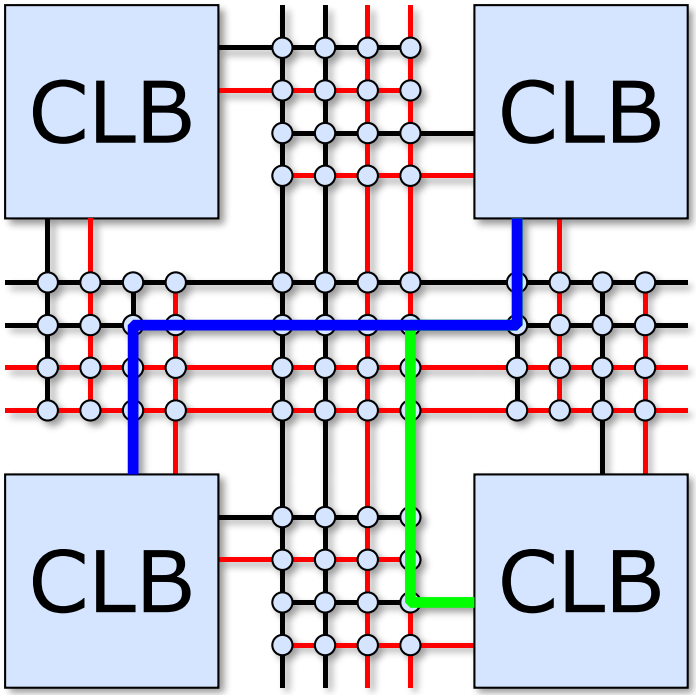


Net 1: base

Net 2: alternative 1

Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.

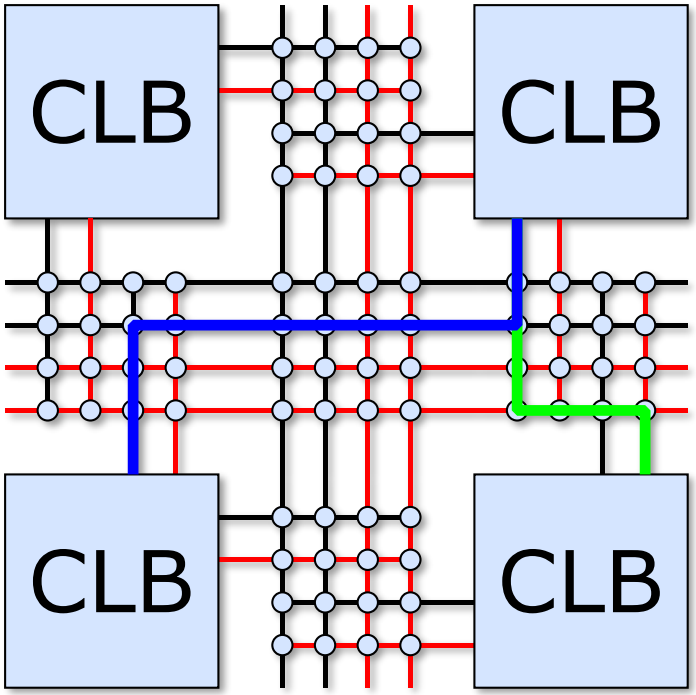


Net 1: base

Net 2: alternative 2

Choose-Your-own-Adventure (CYA) Mapping

- Precompute alternatives for 2-point nets.



Net 1: base

Net 2: alternative 3

CYA for Variation

- Original CYA
  - Developed for defect-tolerance.
  - Functionality check at load-time: 1 or 0

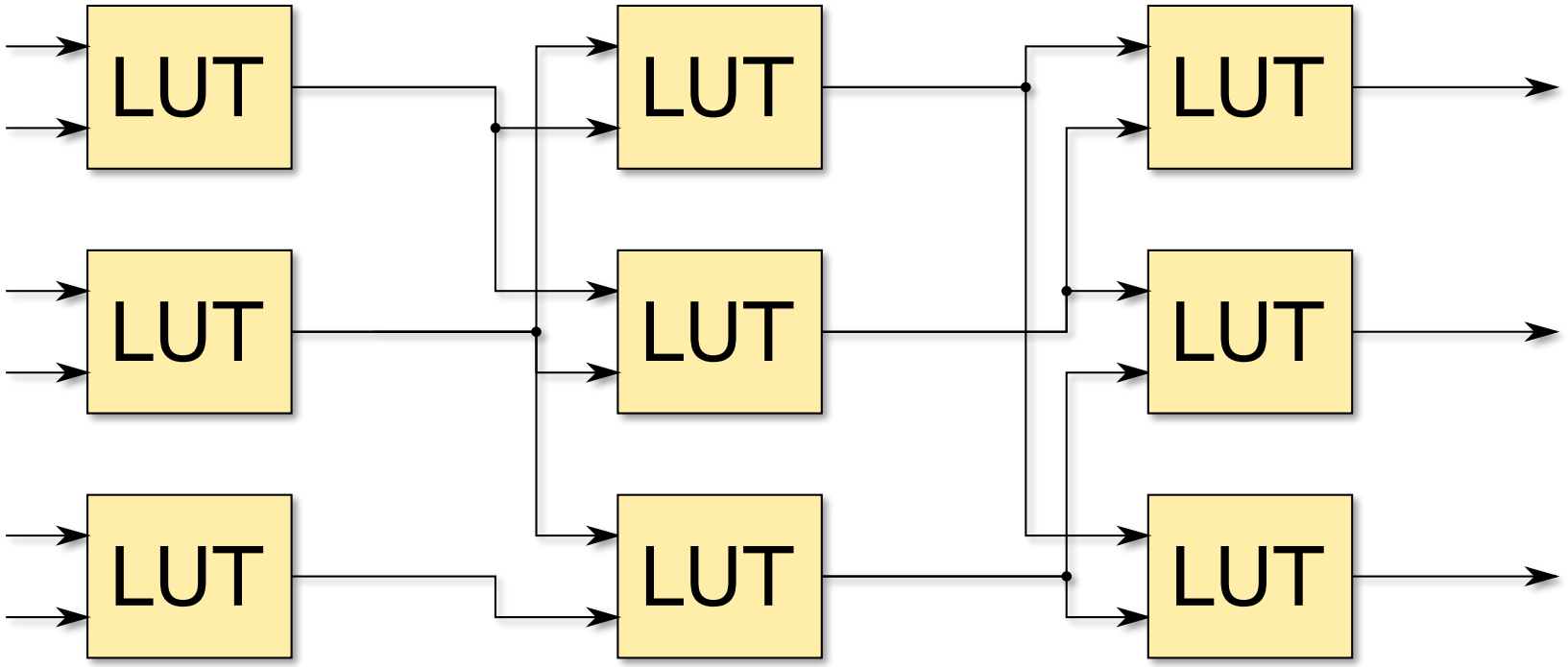
CYA for Variation

- Original CYA
  - Developed for defect-tolerance.
  - Functionality check at load-time: 1 or 0
- CYA for variation
  - Binary timing circuits: late or not

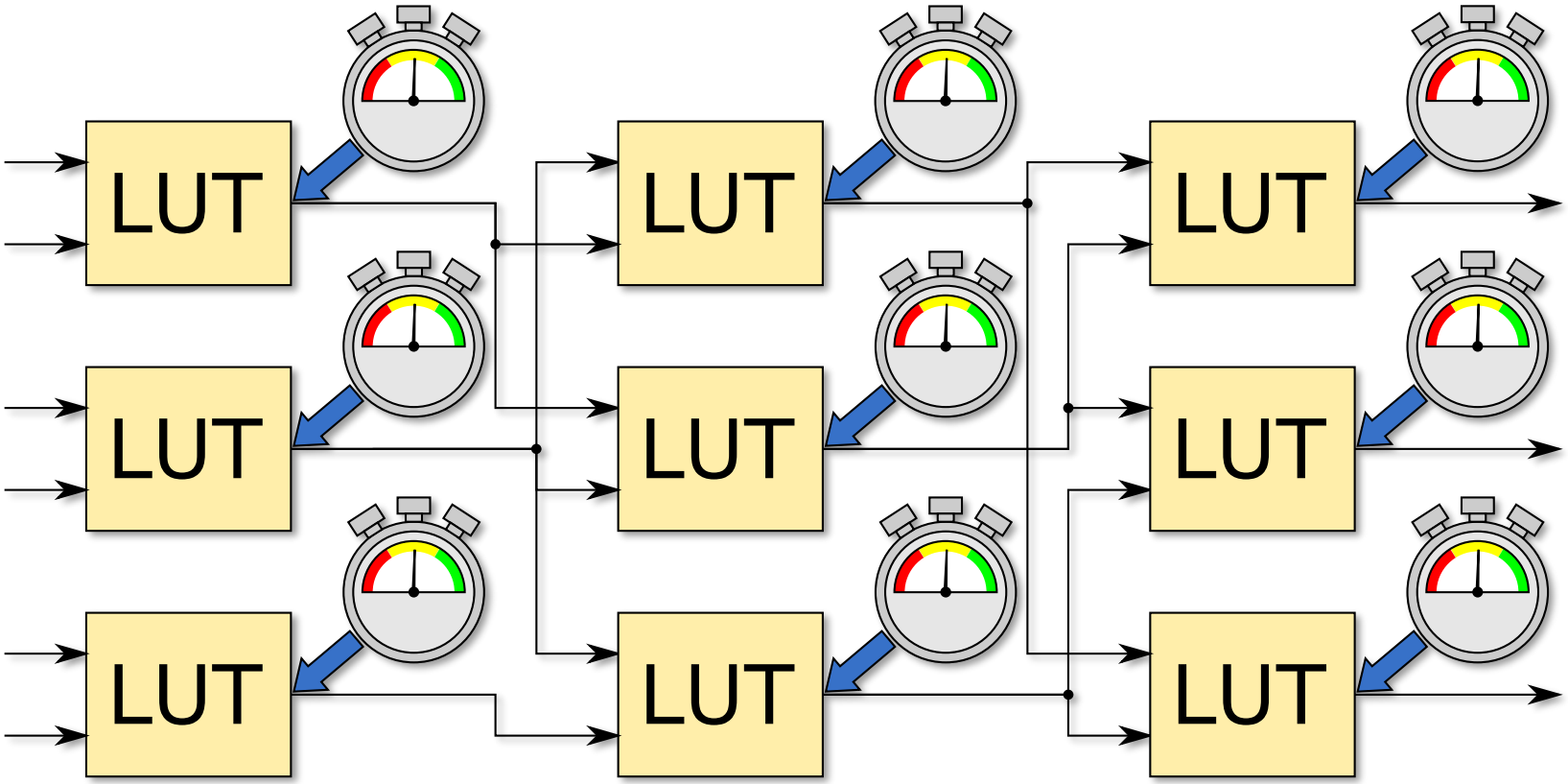
CYA for Variation

- Original CYA
  - Developed for defect-tolerance.
  - Functionality check at load-time: 1 or 0
- CYA for variation
  - Binary timing circuits: late or not
- But: alternatives may conflict.
  - Can run out of good alternatives.
  - Delay unknown, so cannot prioritize nets.

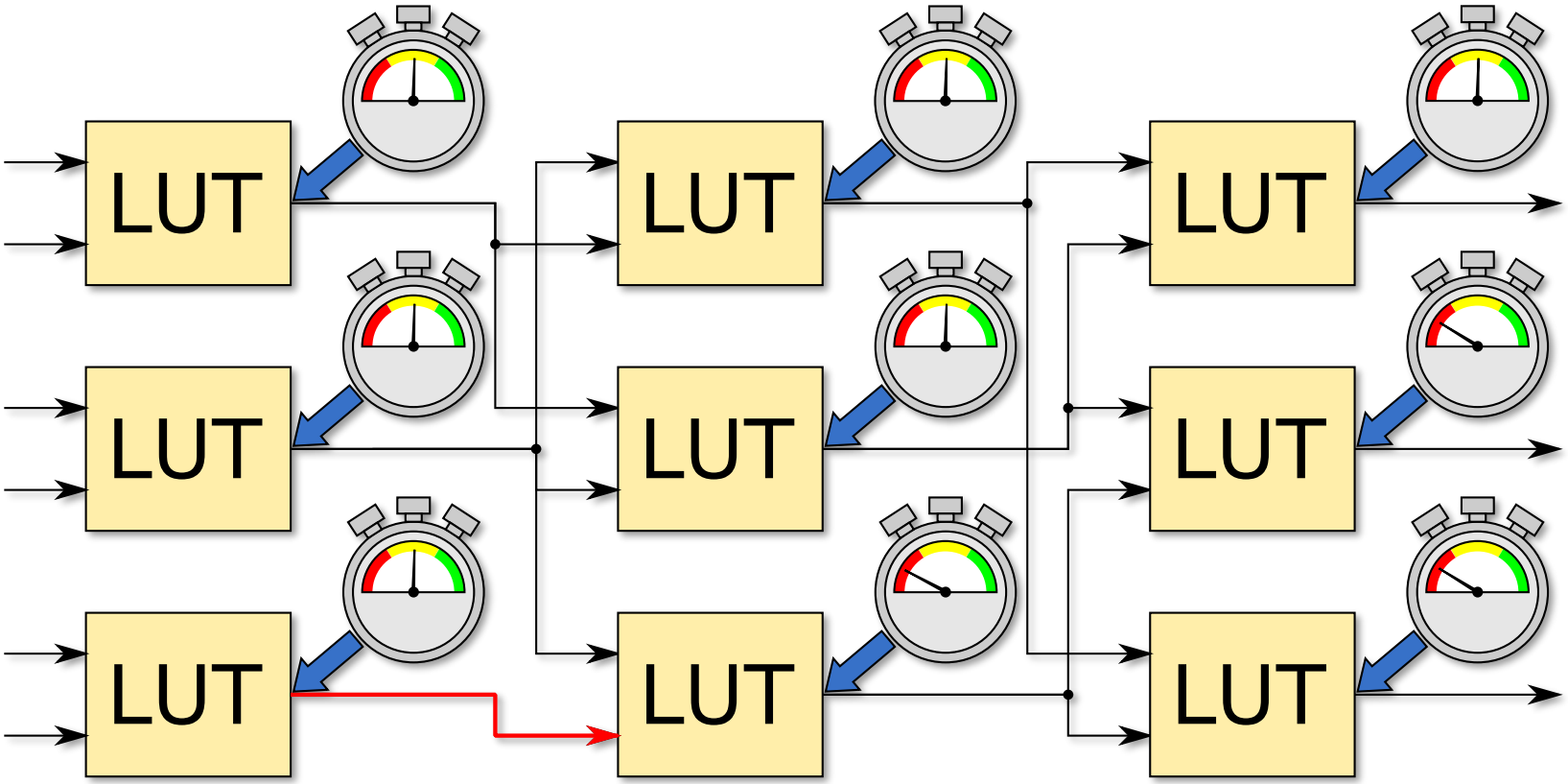
Locating Slow Resource



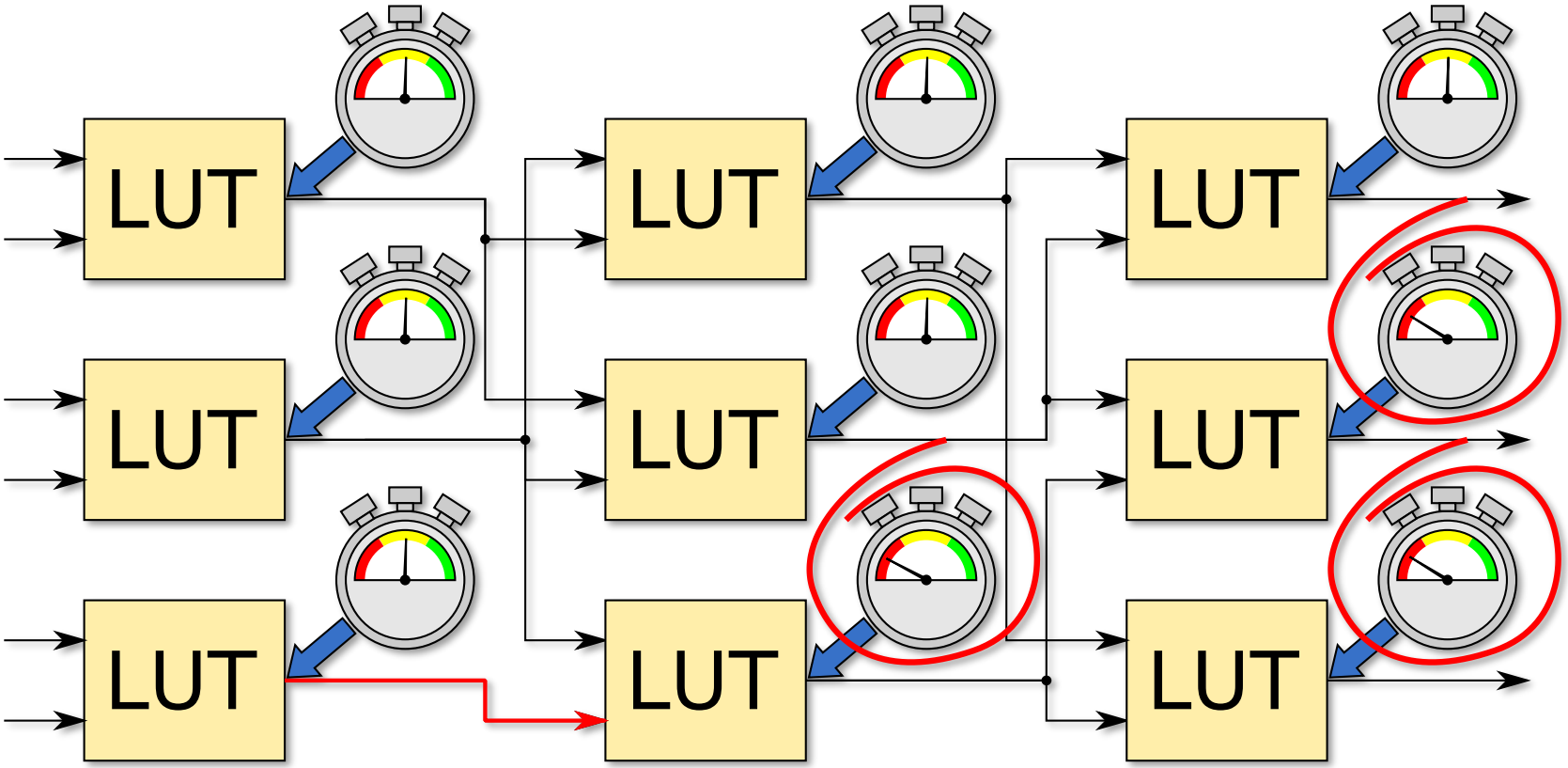
Locating Slow Resource



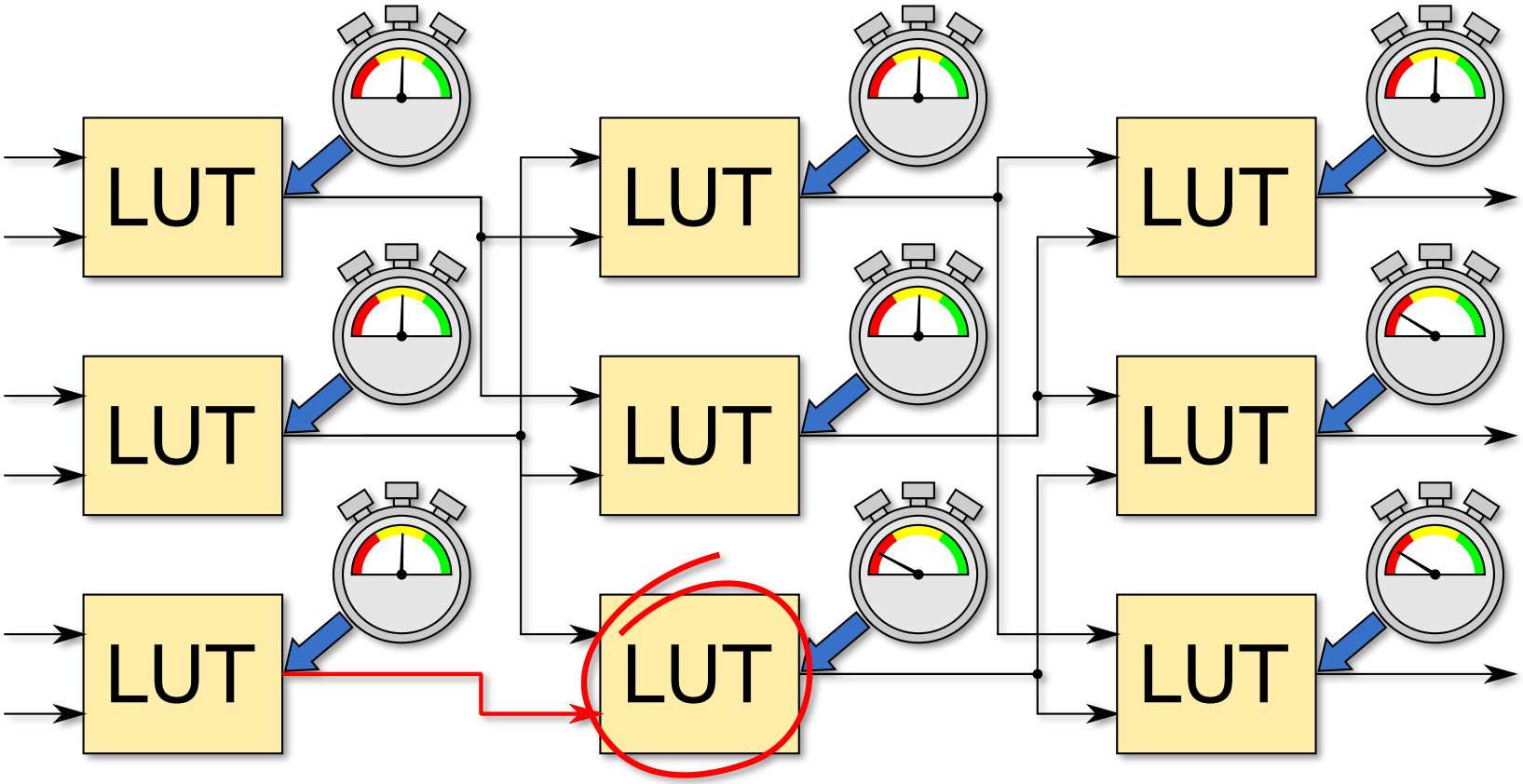
Locating Slow Resource



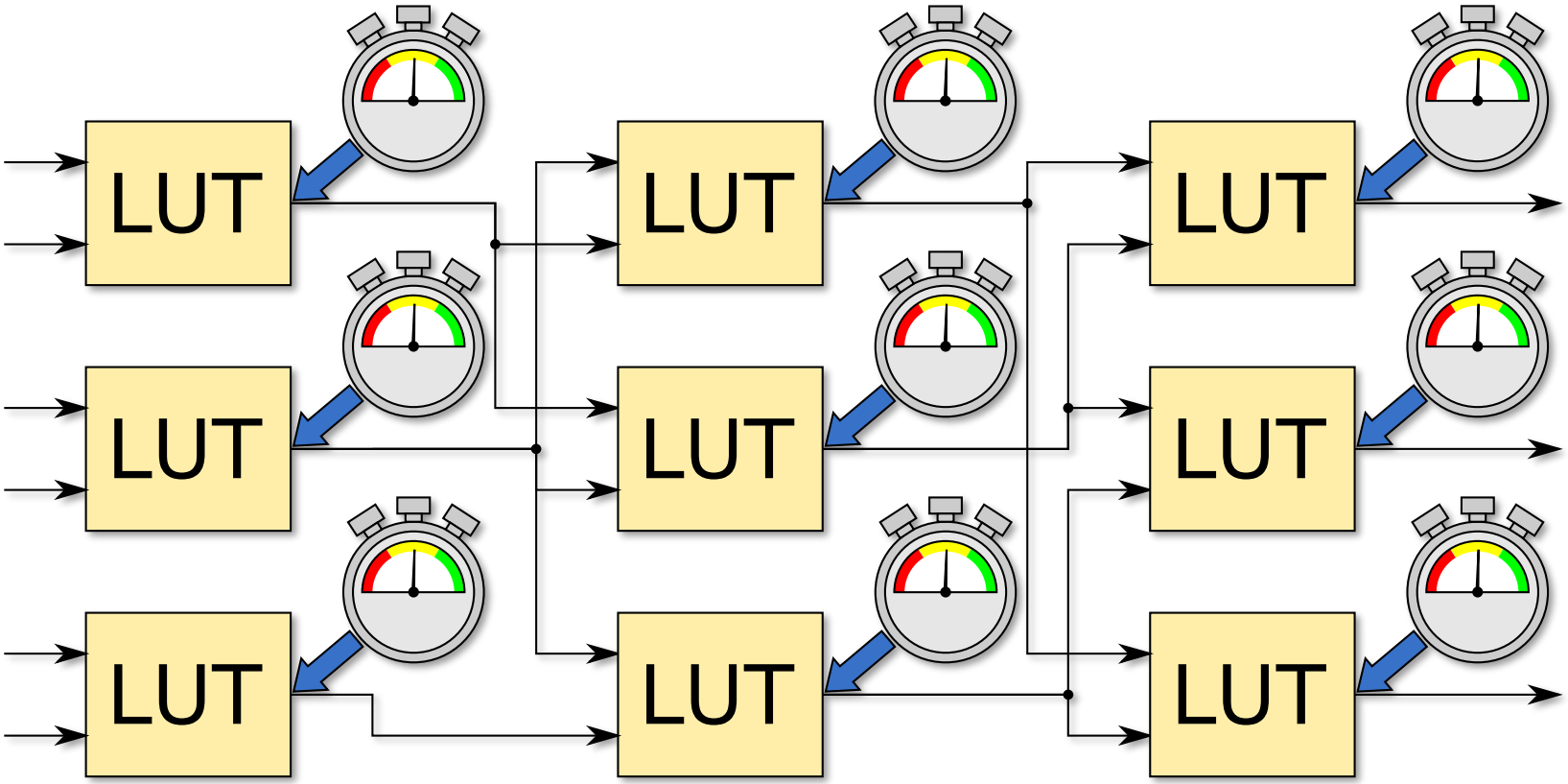
Locating Slow Resource



Locating Slow Resource

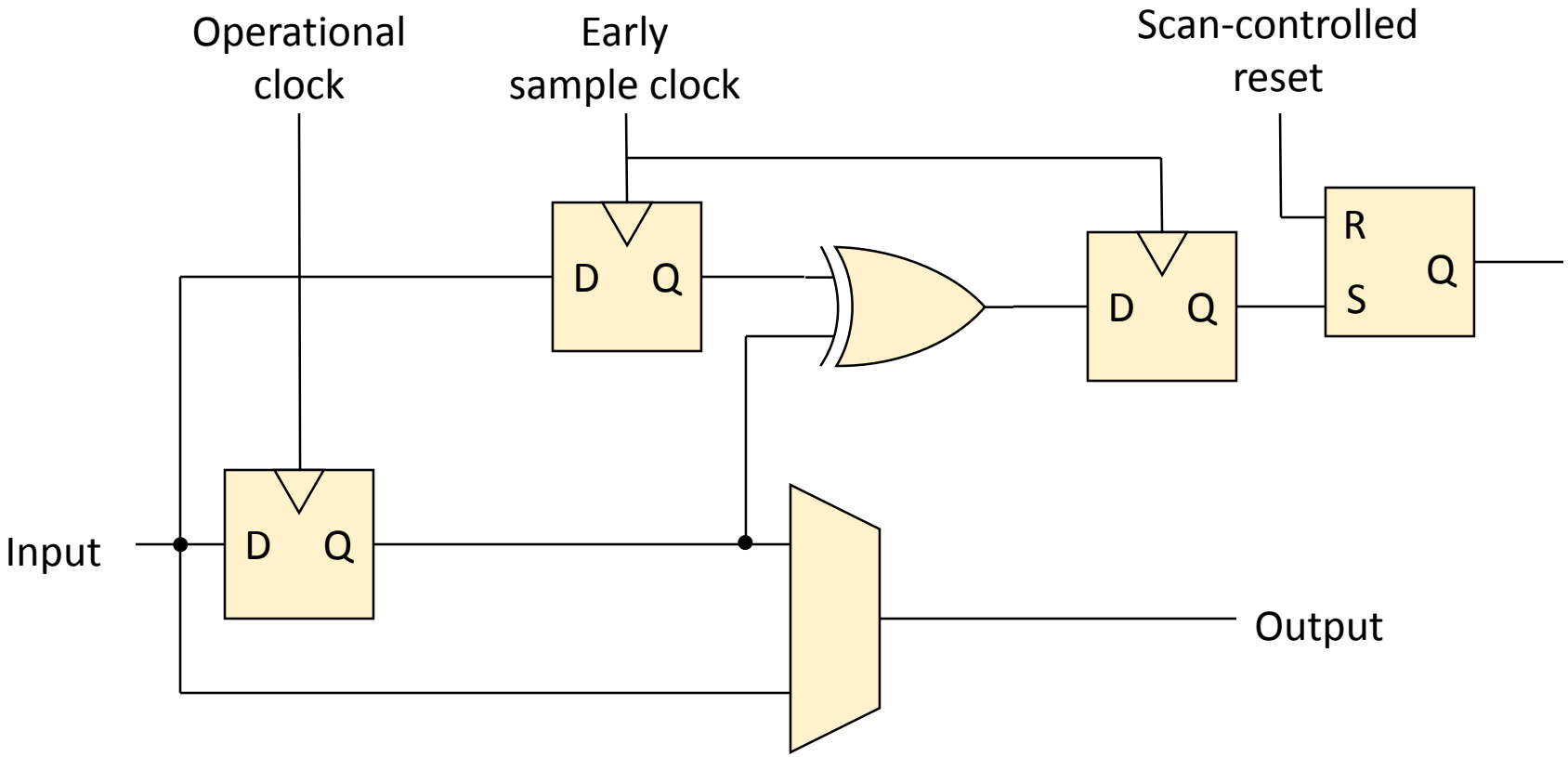


Locating Slow Resource



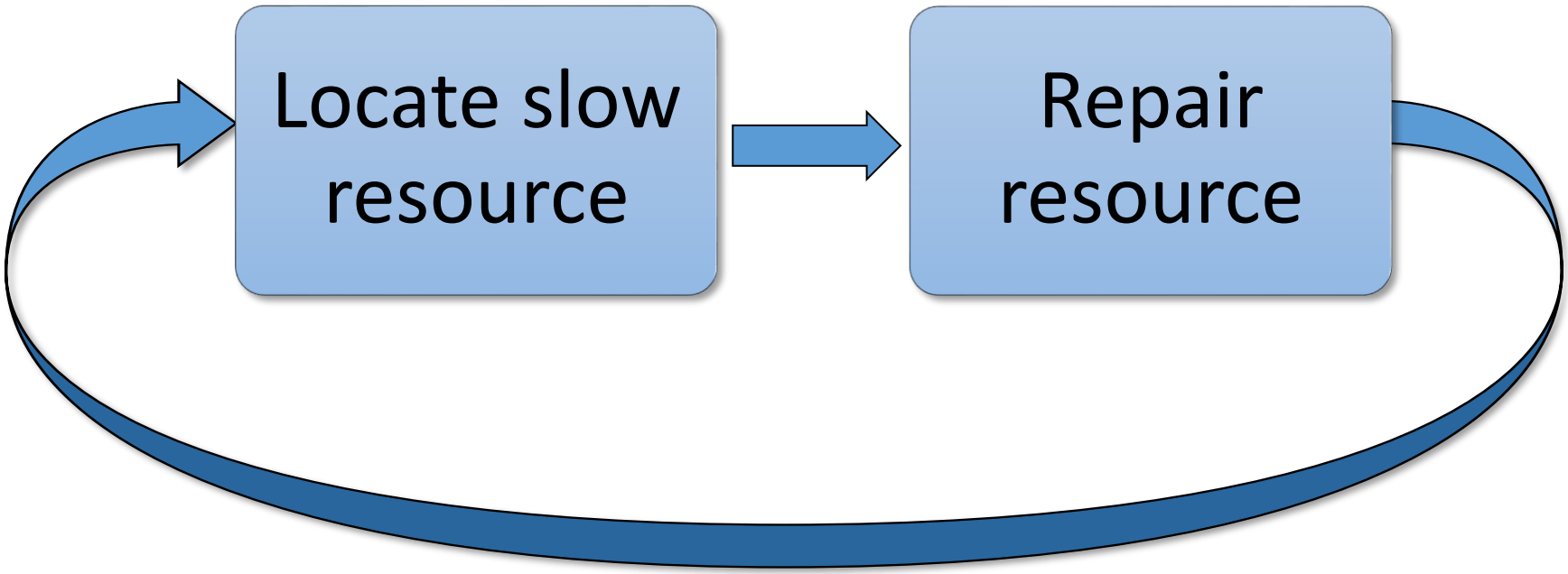
Difference Detector with First-Fail Latch

- Adds 4% area overhead to Stratix-IV CLB.



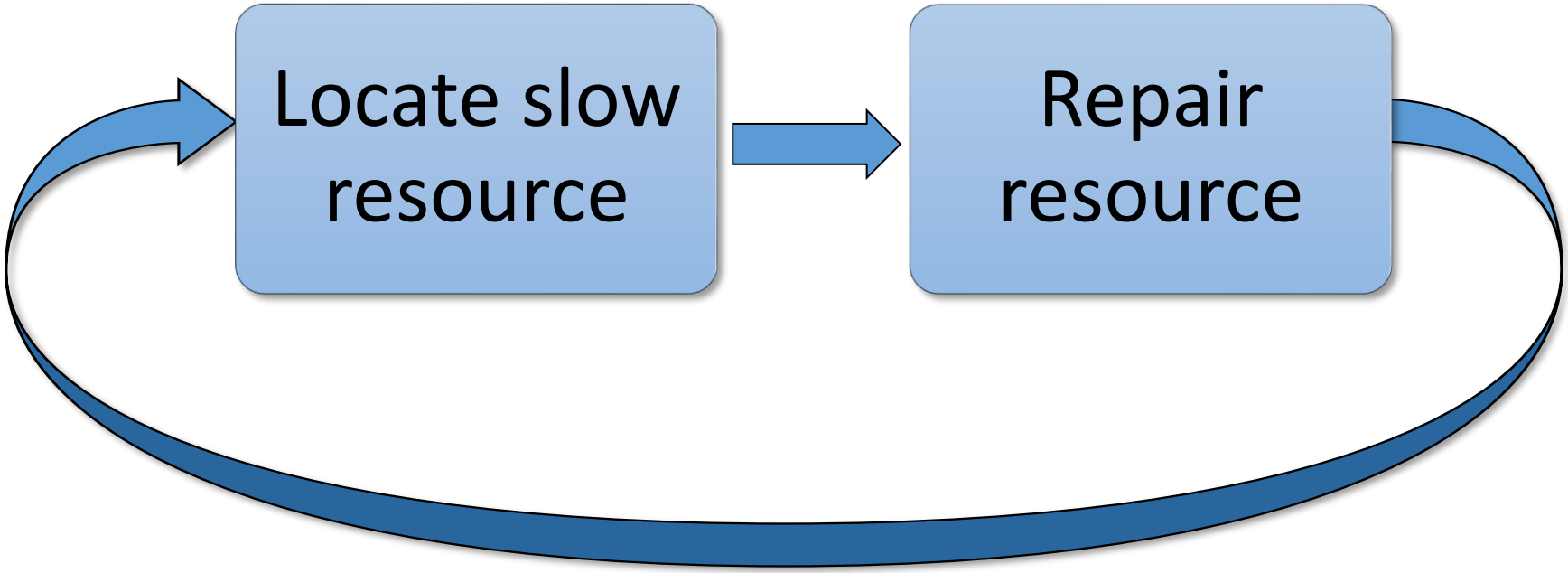
[Levine, FCCM 2012]

Incremental CYA



Incremental CYA

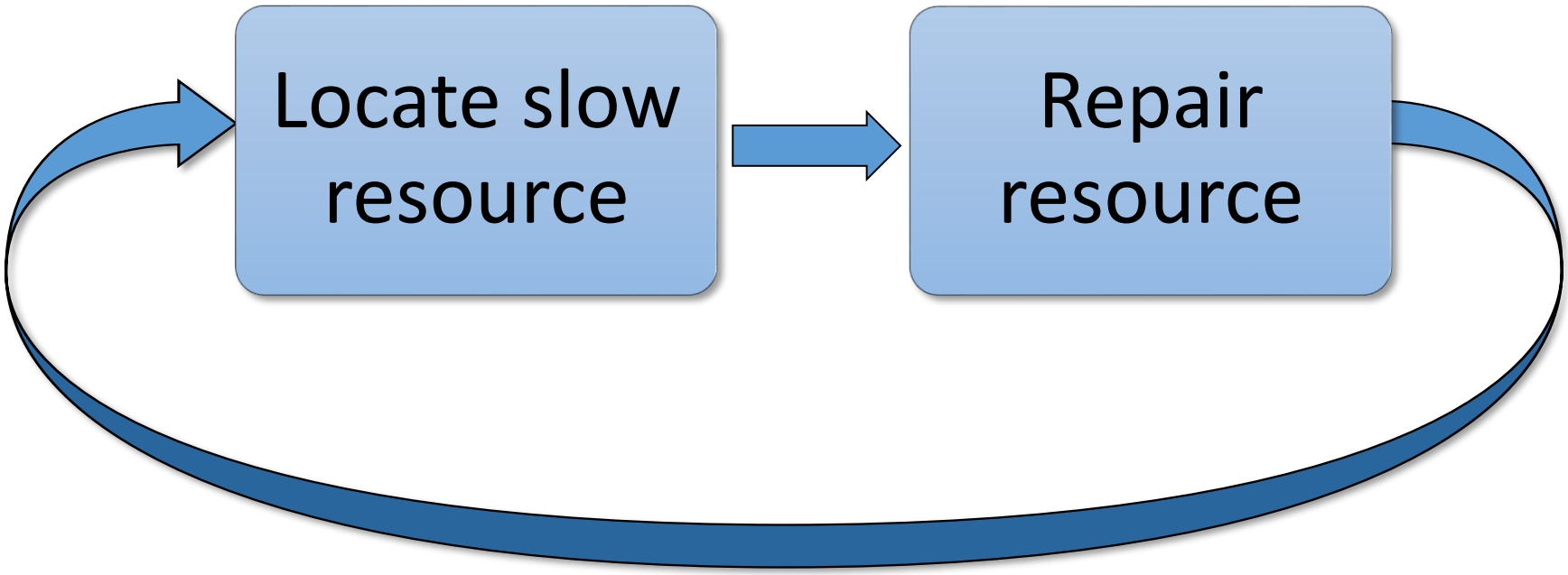
Based on COSMIC TRIP
[Giesen, FCCM 2016]



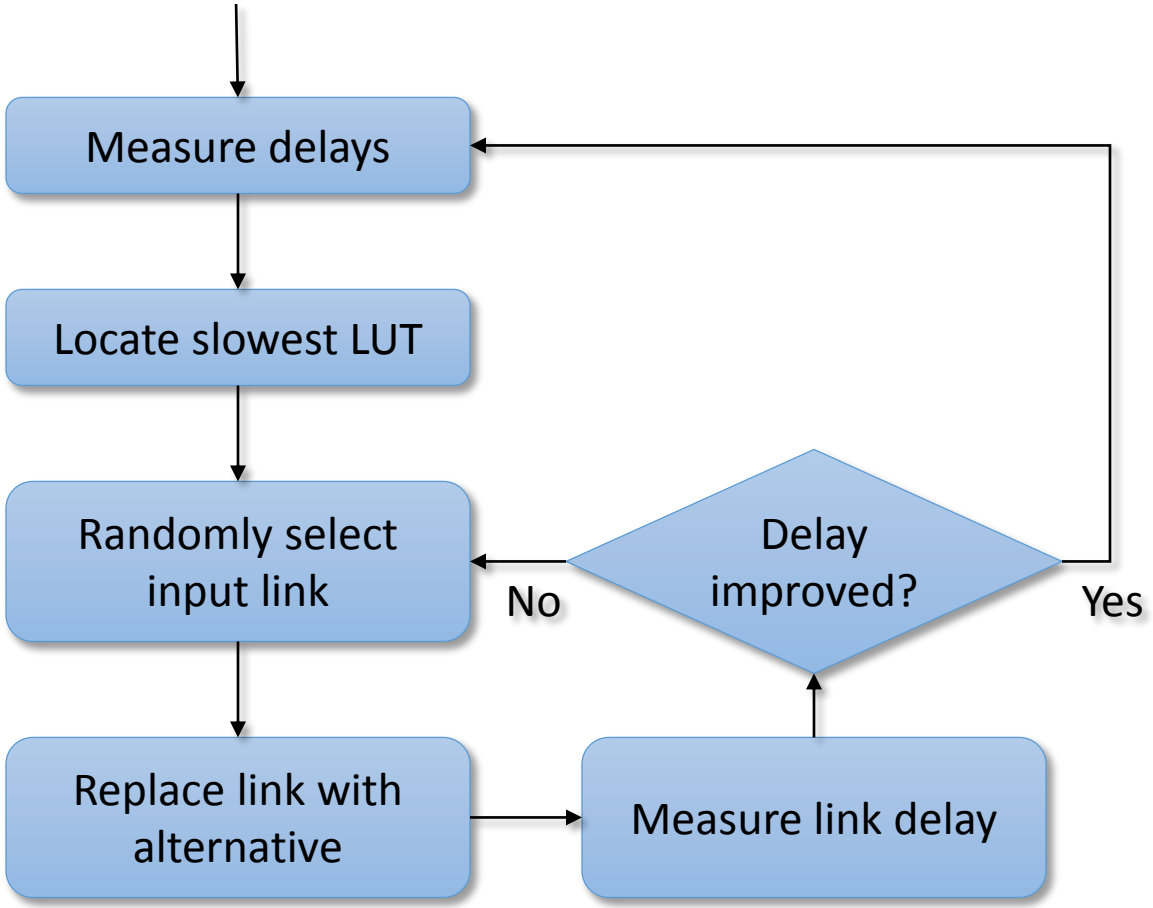
Incremental CYA

Based on COSMIC TRIP
[Giesen, FCCM 2016]

using CYA

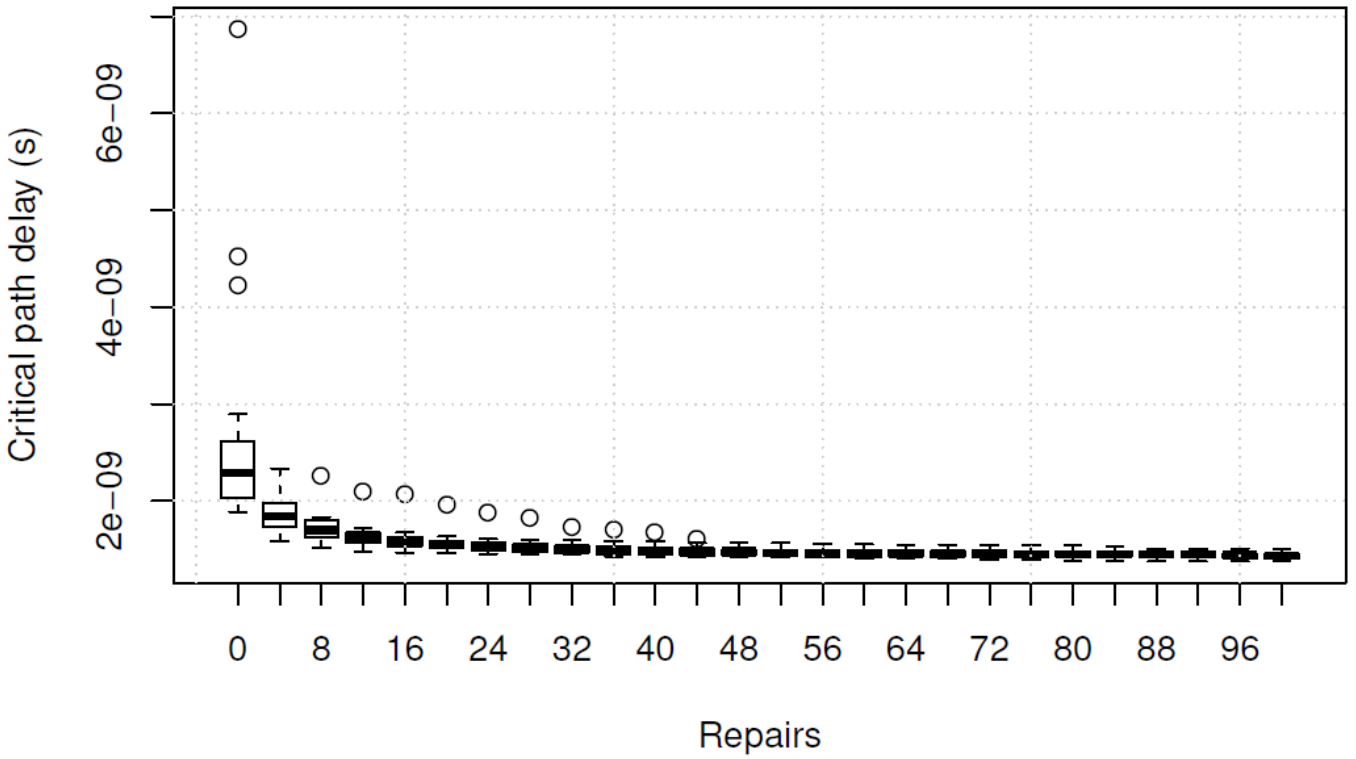


Incremental CYA



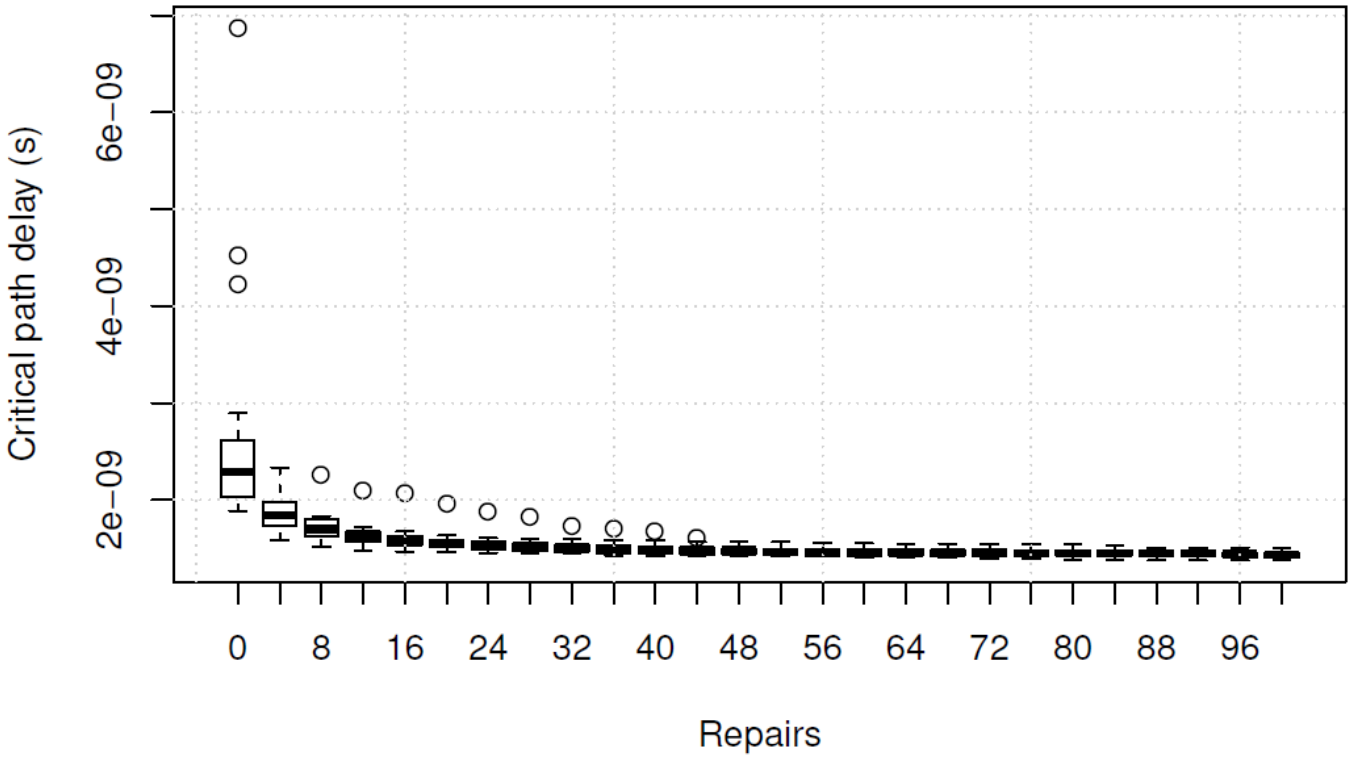
Incremental CYA - Example

- 20 chips, random data input



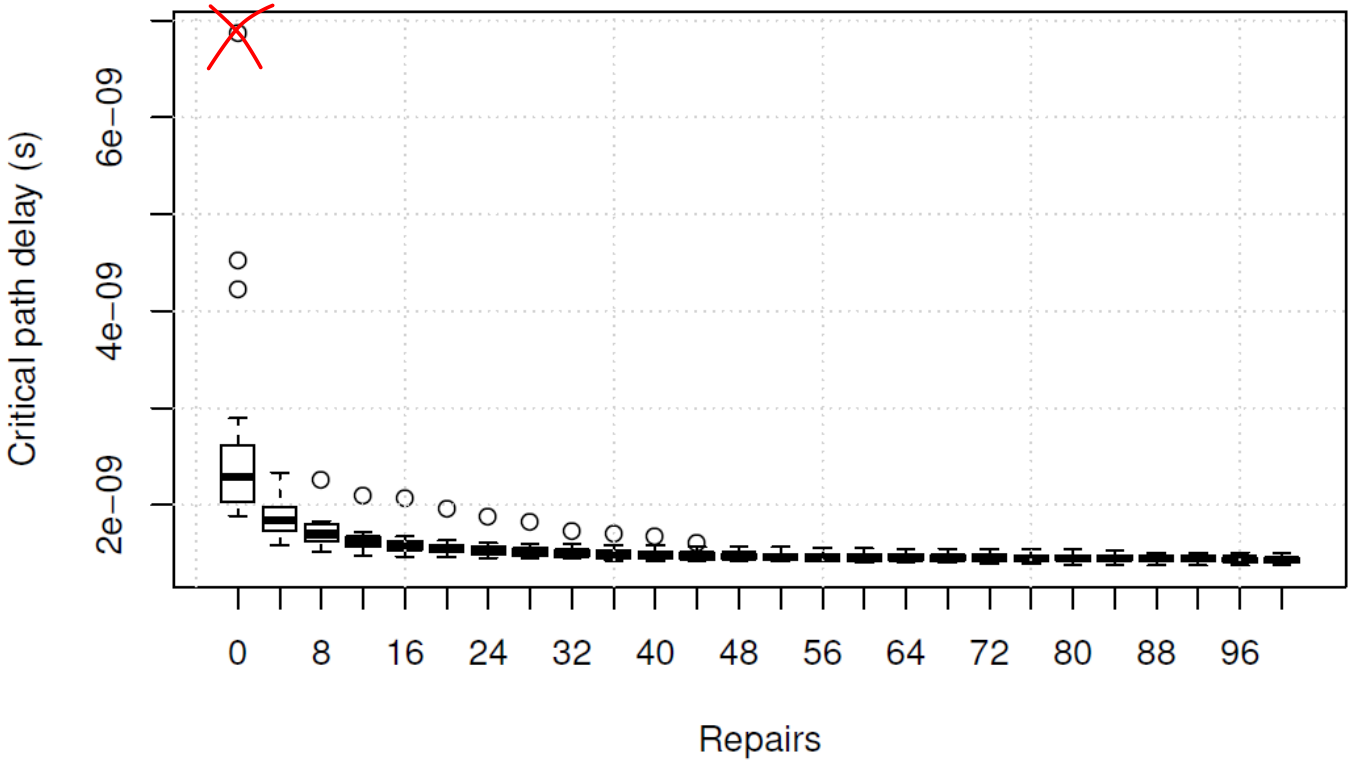
Incremental CYA - Example

- 20 chips, random data input
- Upcoming results reported at 95% yield point.



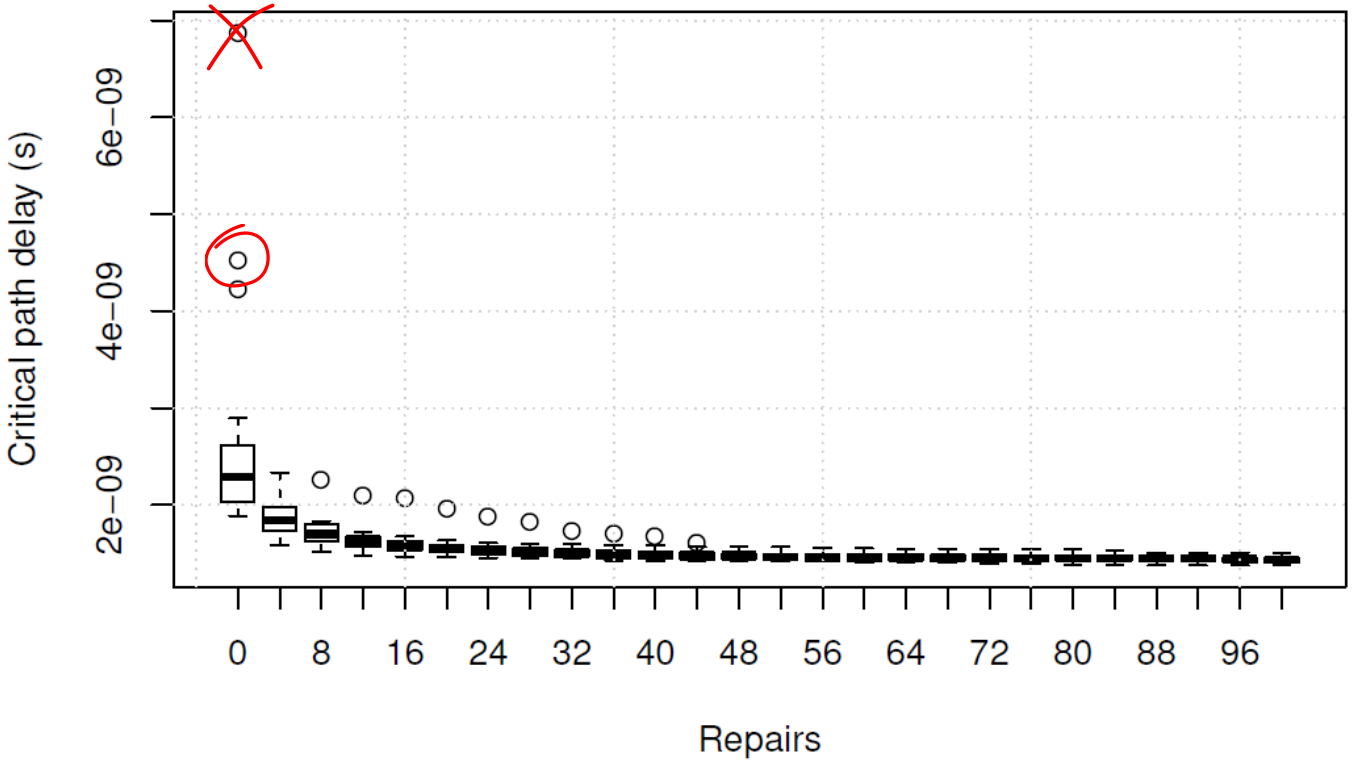
Incremental CYA - Example

- 20 chips, random data input
- Upcoming results reported at 95% yield point.



Incremental CYA - Example

- 20 chips, random data input
- Upcoming results reported at 95% yield point.

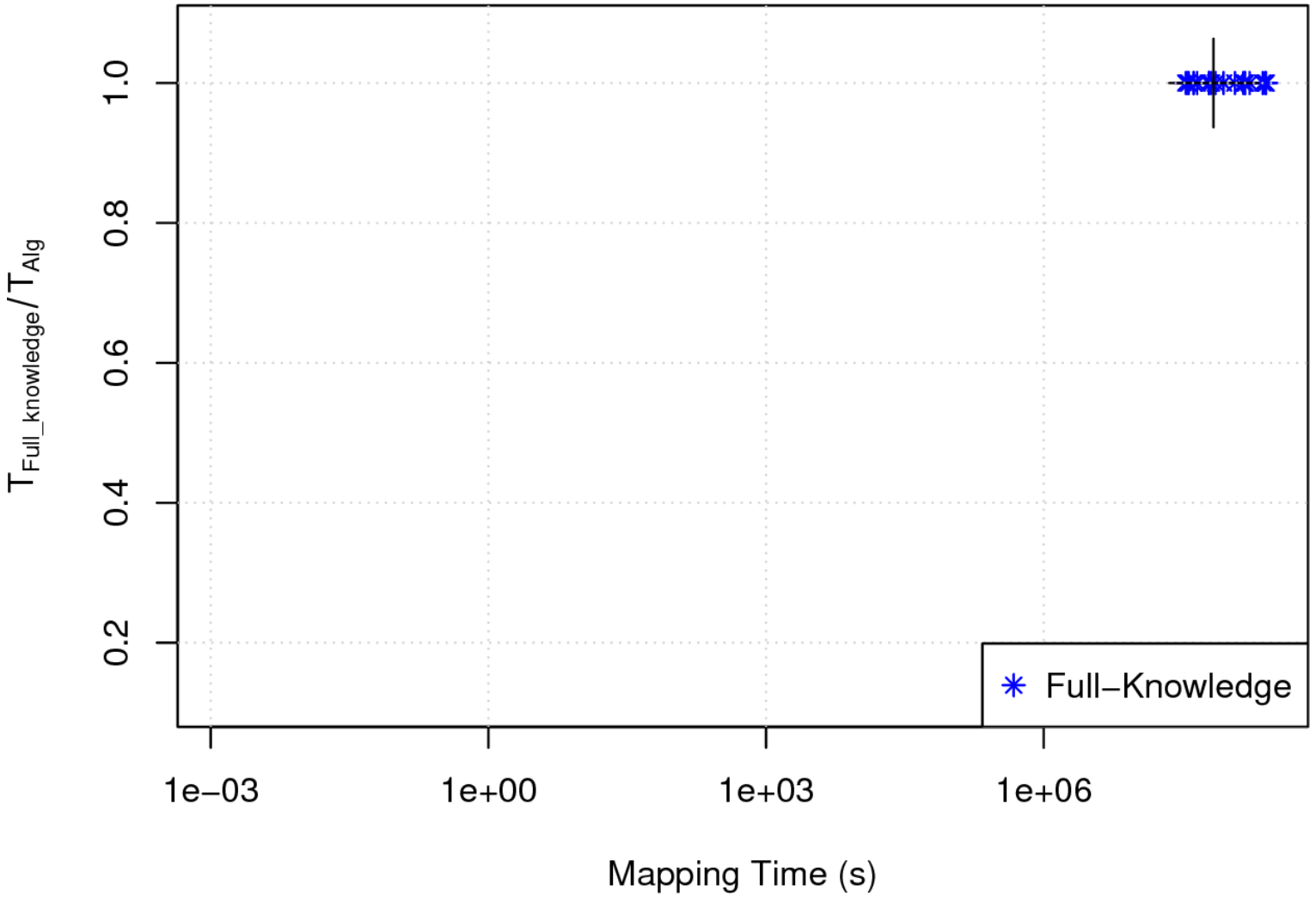


Experiments

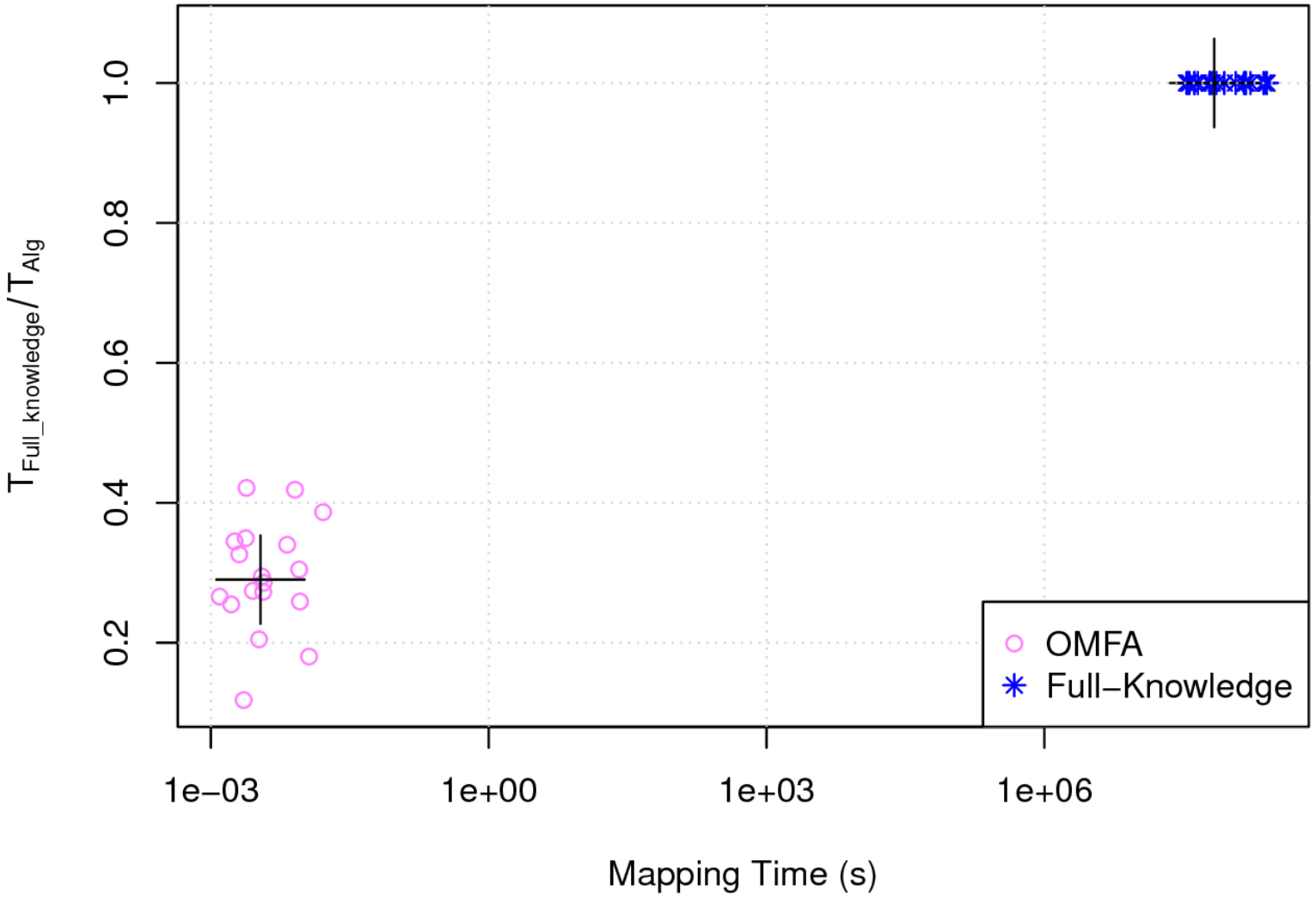
Methodology

- VPR 5.0.2 with CYA extension
  - Custom timing simulator for Incremental CYA
- Toronto20 benchmarks
- 22-nm CMOS, 0.8 V nominal
- Architecture similar to Stratix-IV
  - 4 LUTs, 16 CLB inputs, and 16 tracks extra
- 64 alternatives per 2-point net

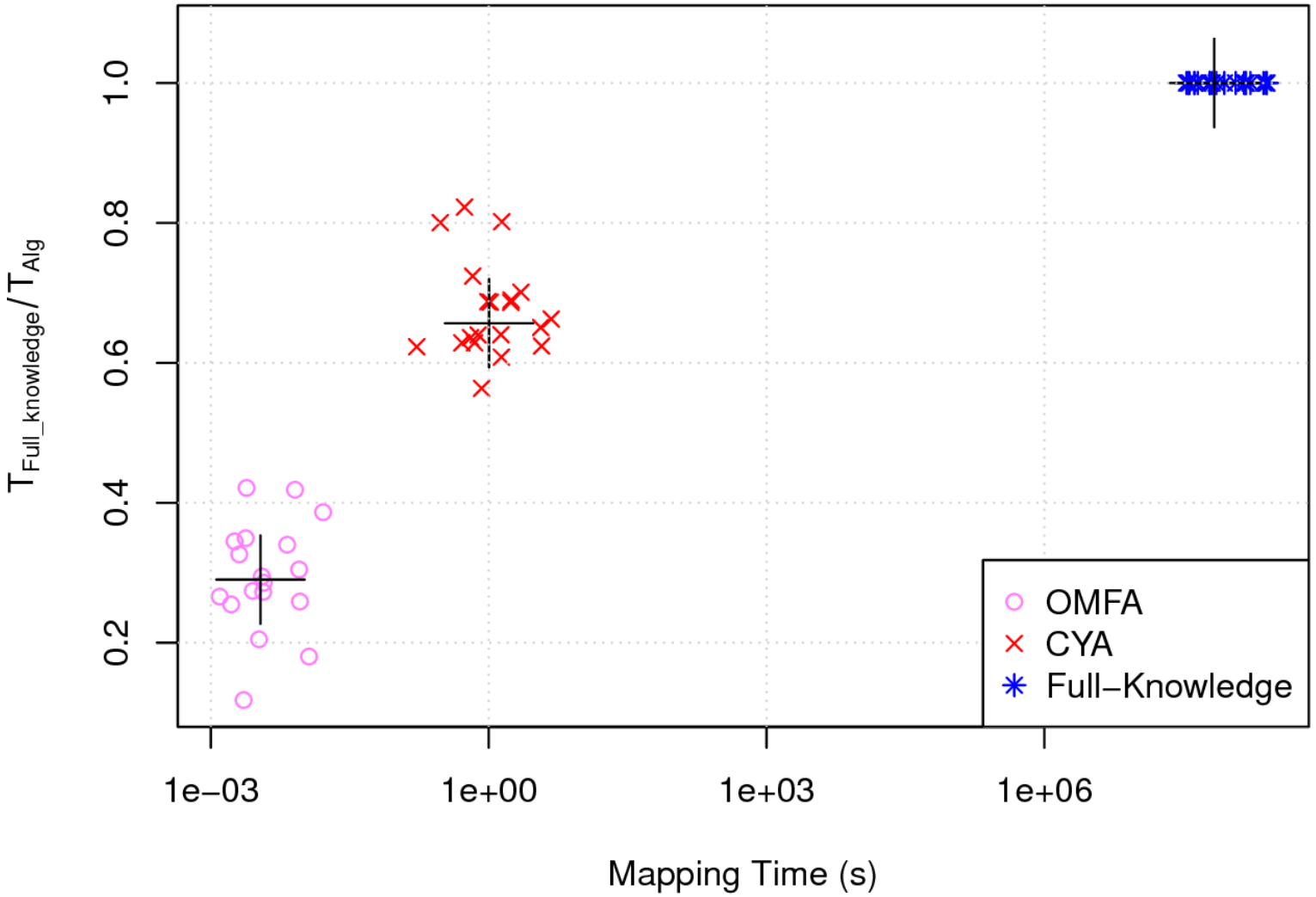
Comparison - Delay



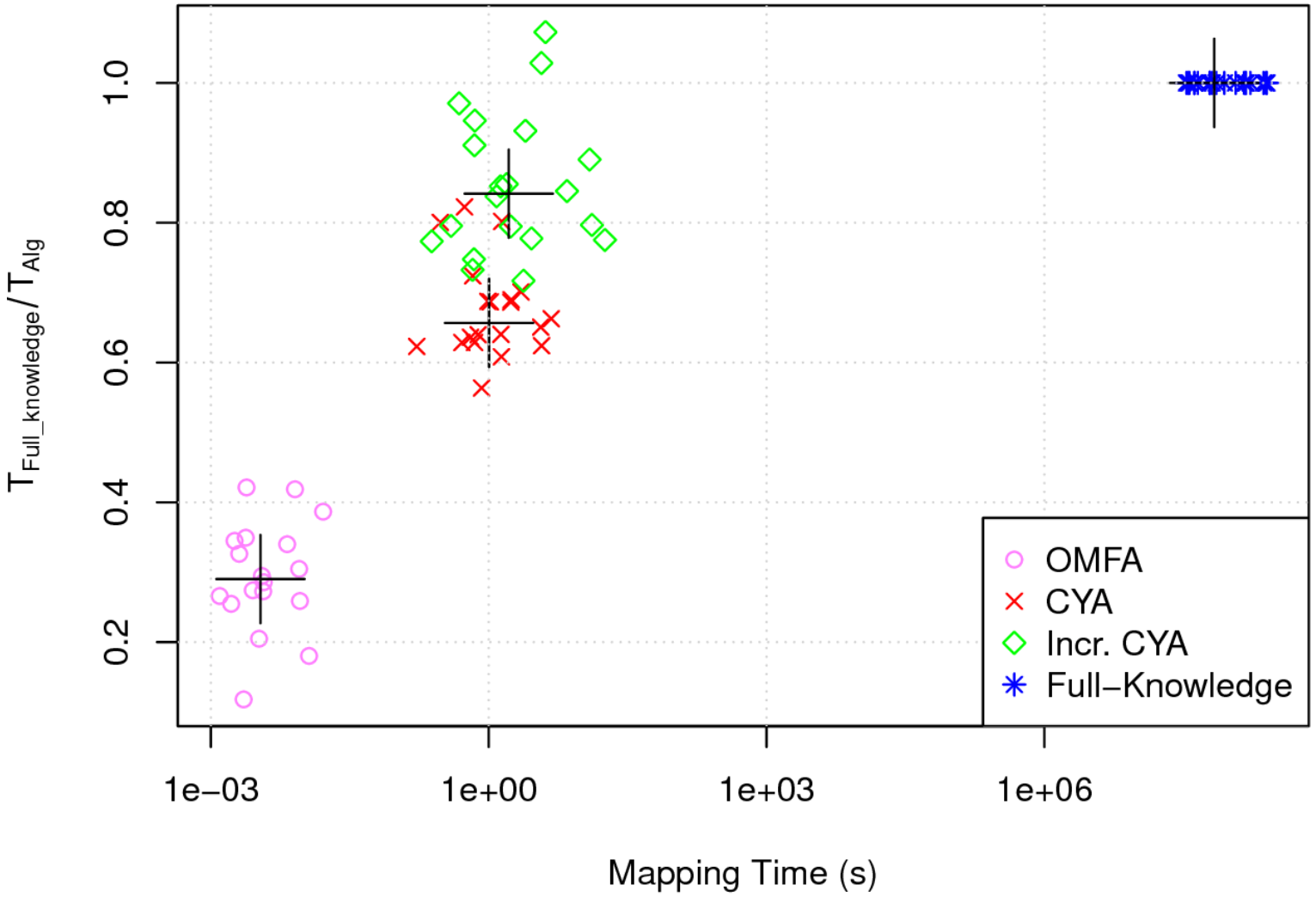
Comparison - Delay



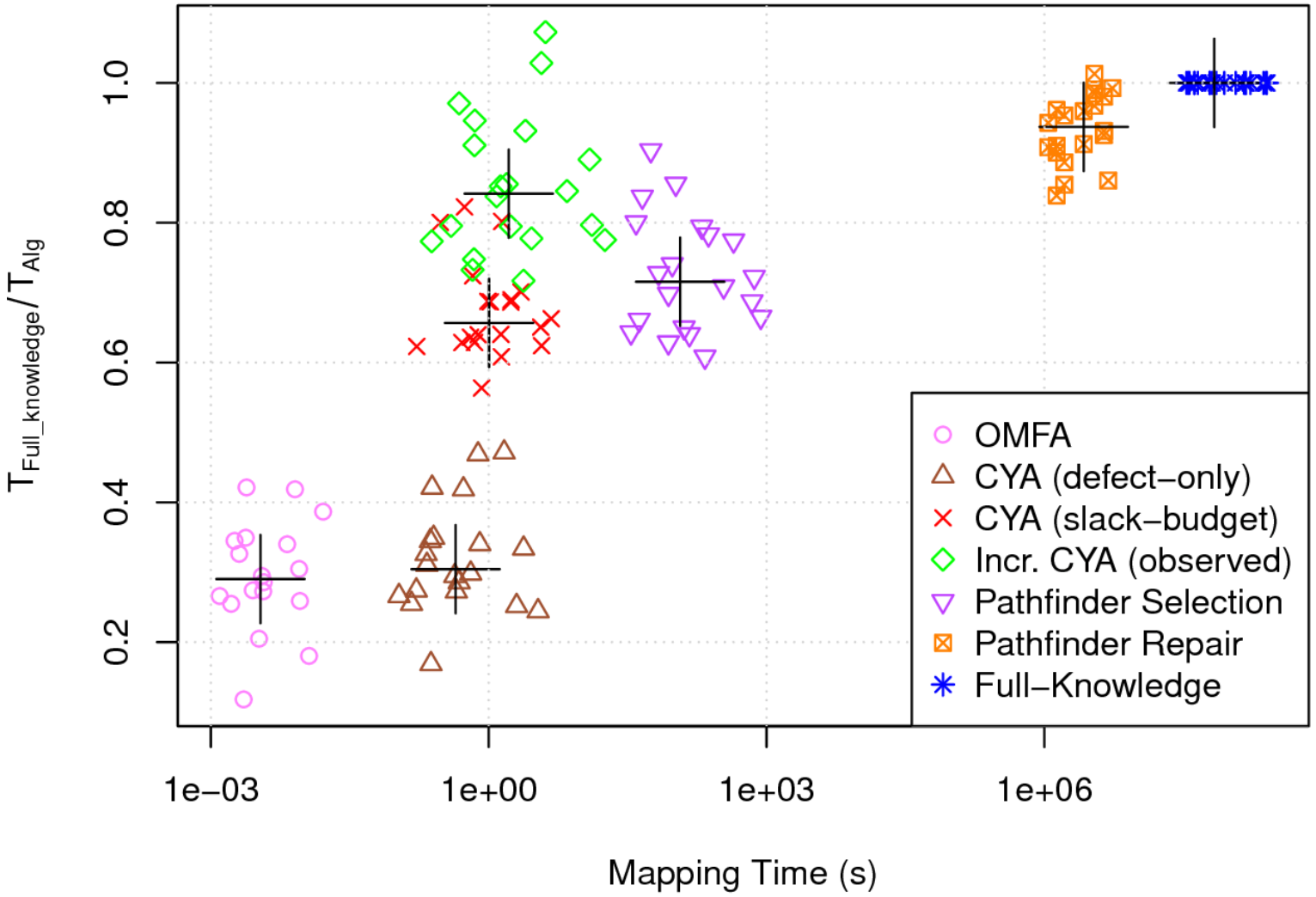
Comparison - Delay



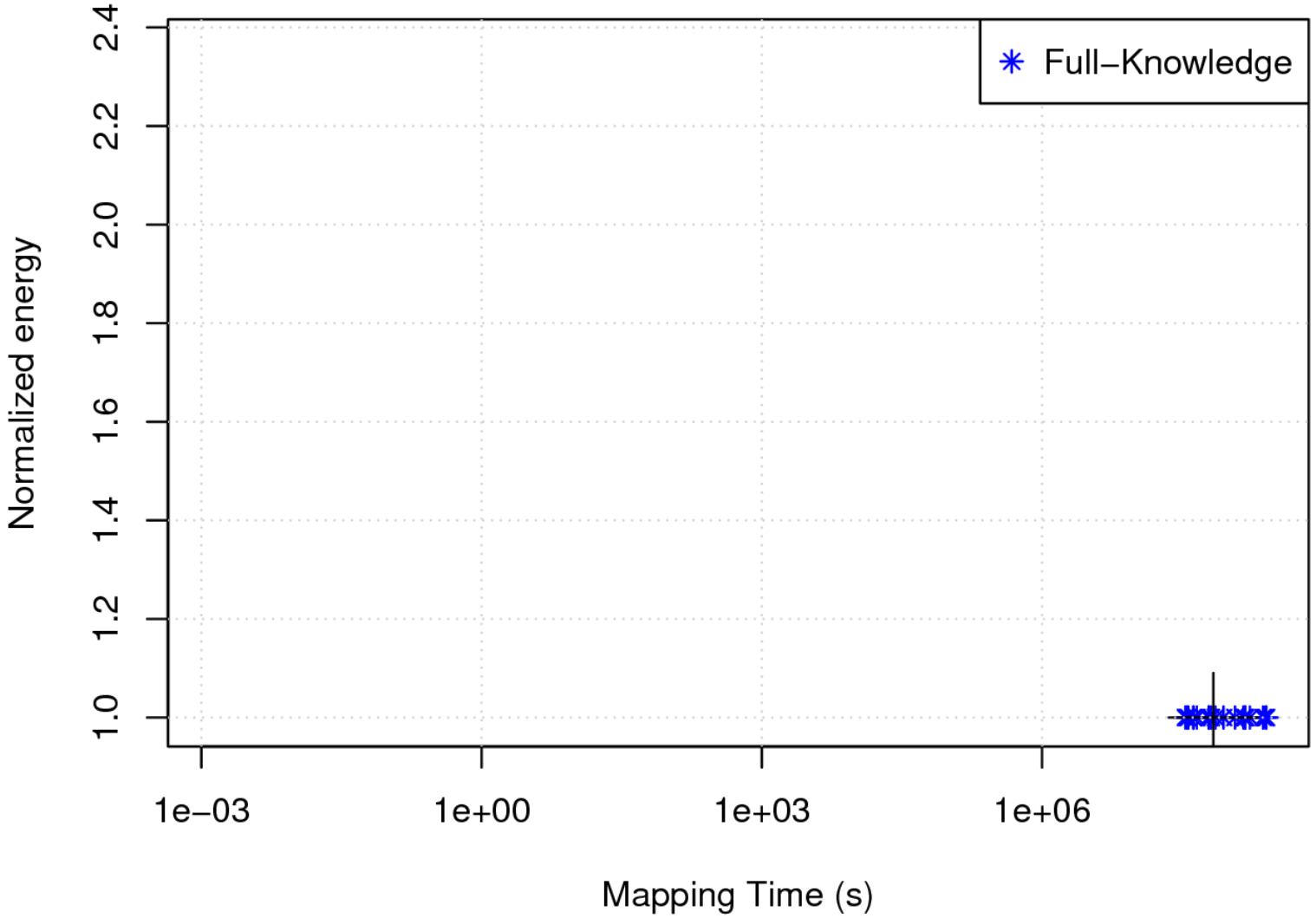
Comparison - Delay



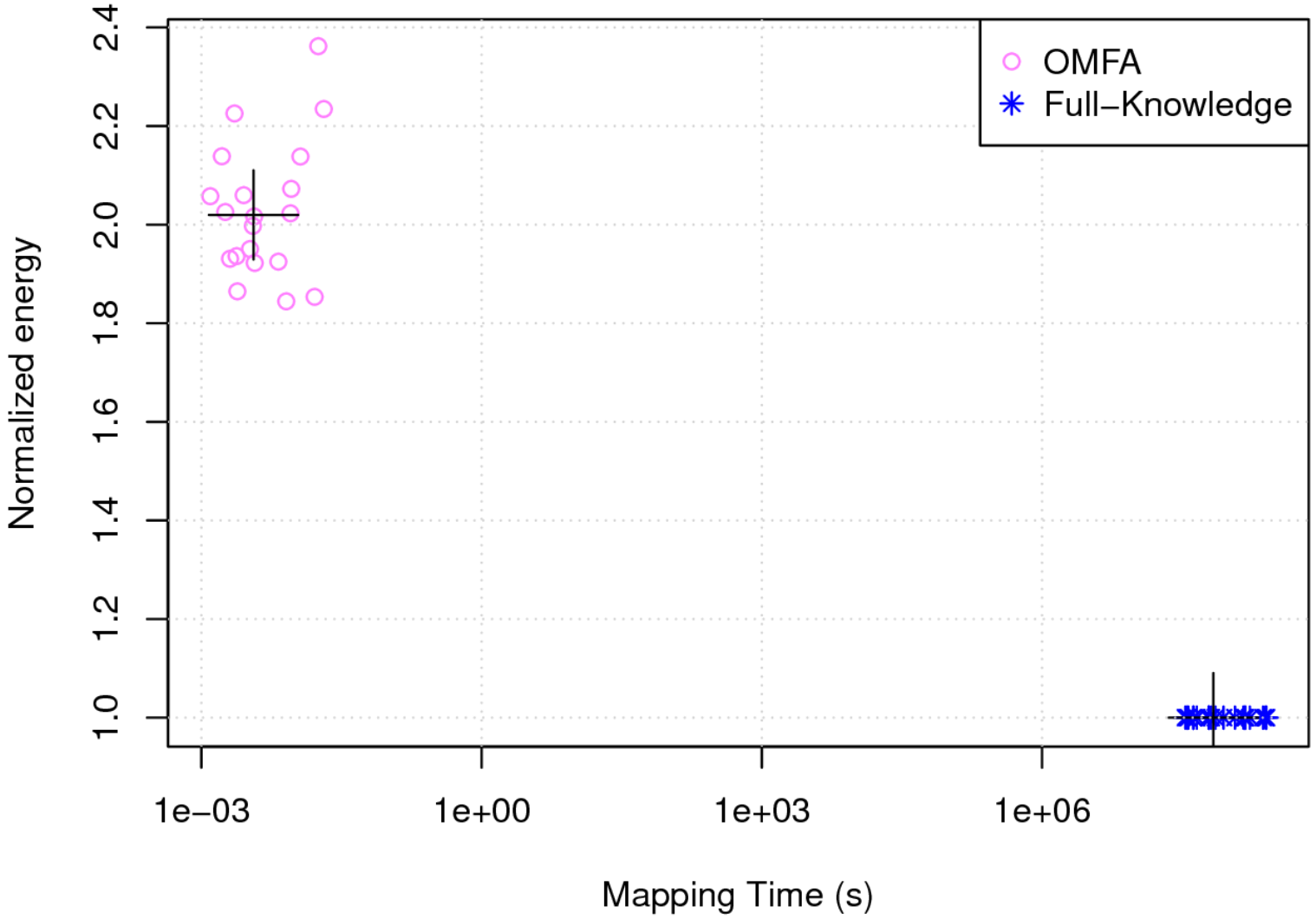
Comparison - Delay



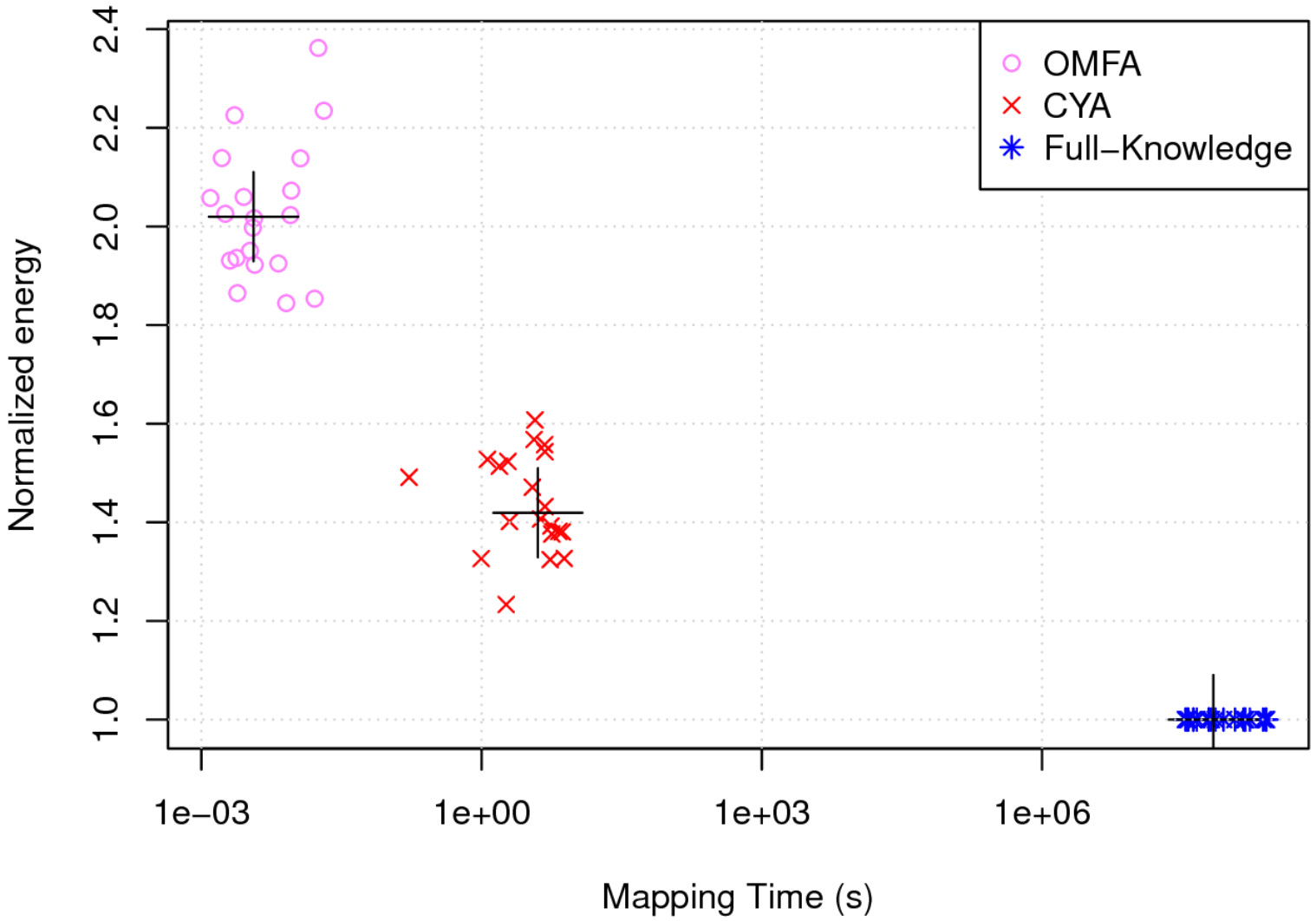
Comparison - Energy



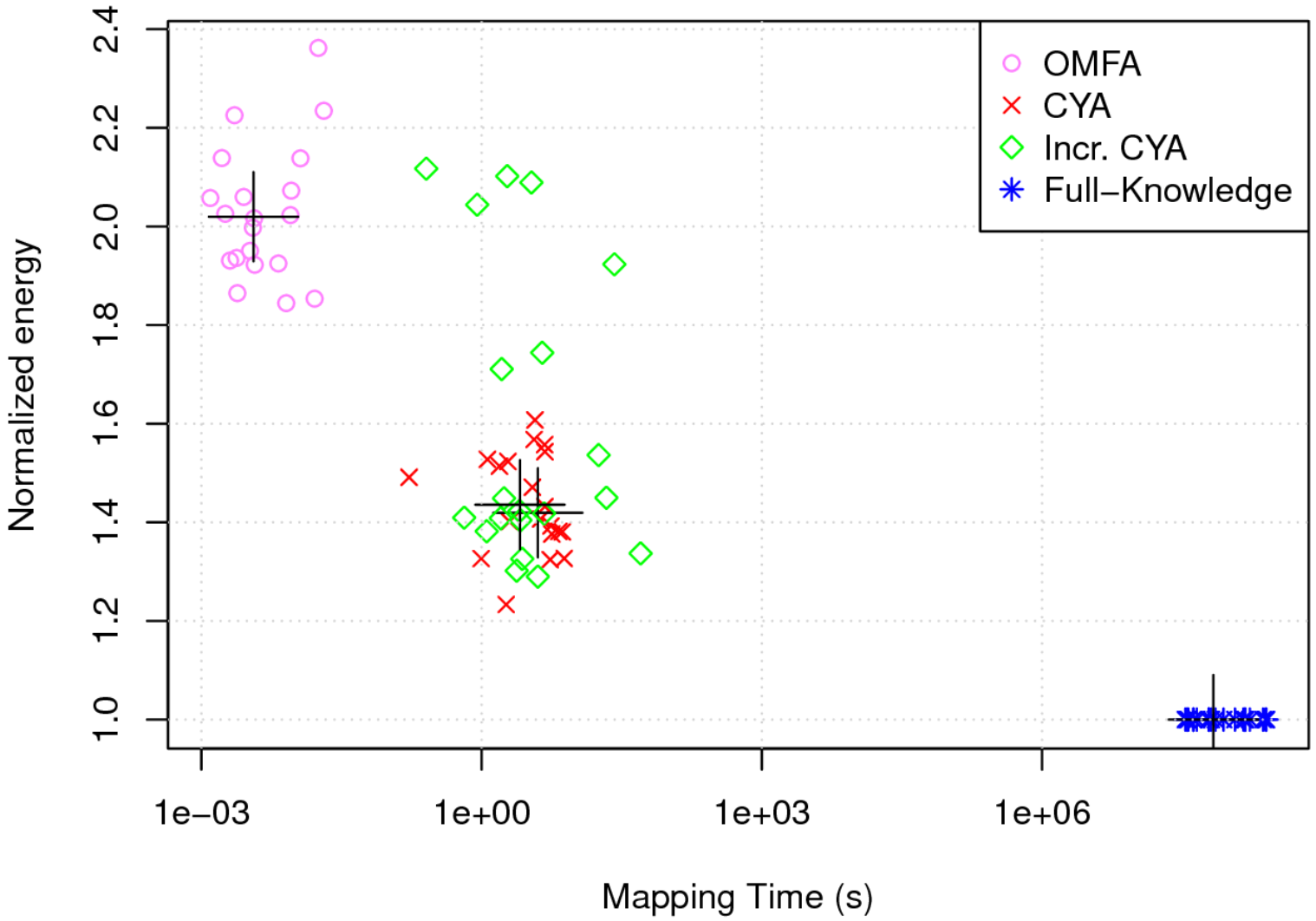
Comparison - Energy



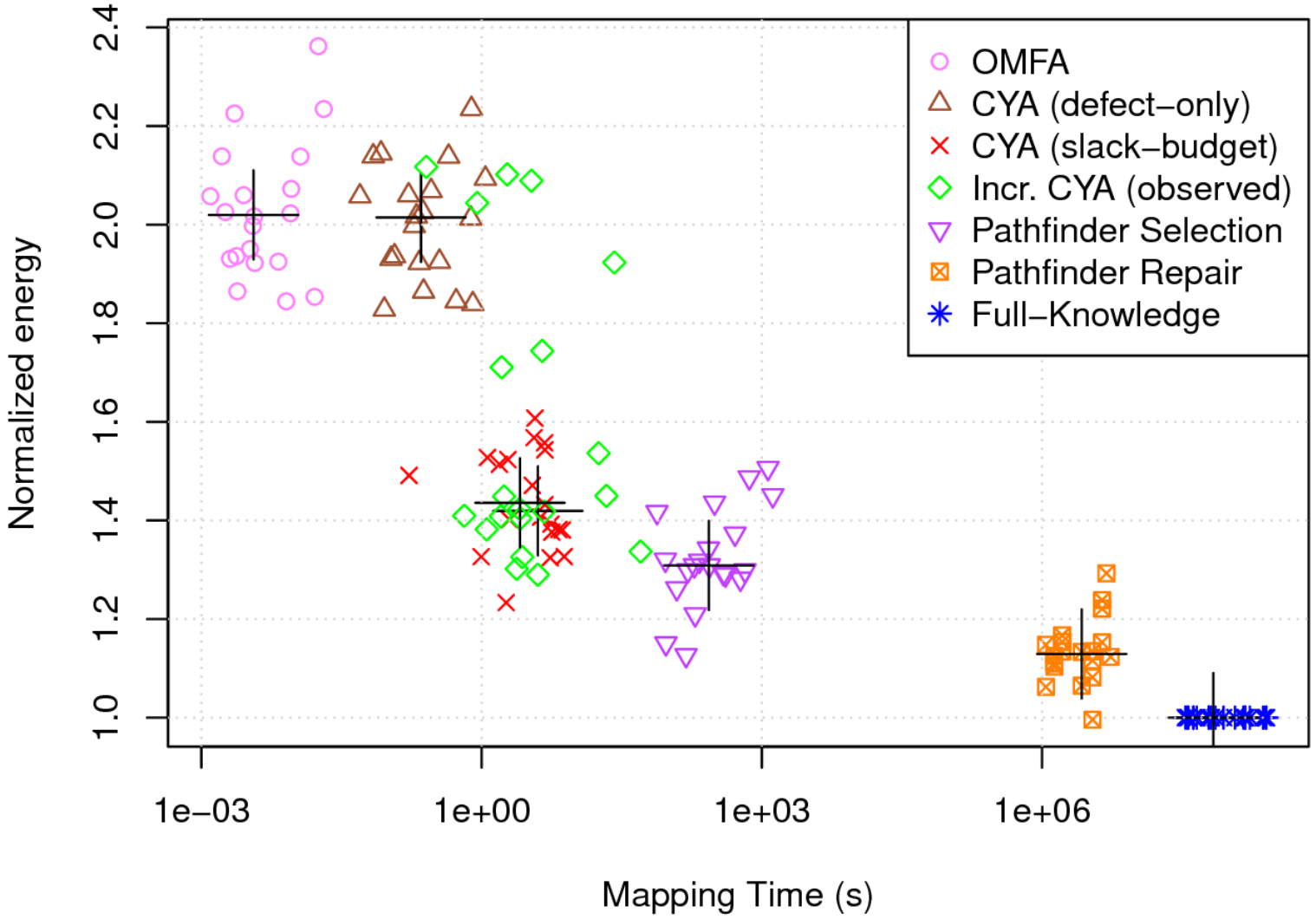
Comparison - Energy



Comparison - Energy

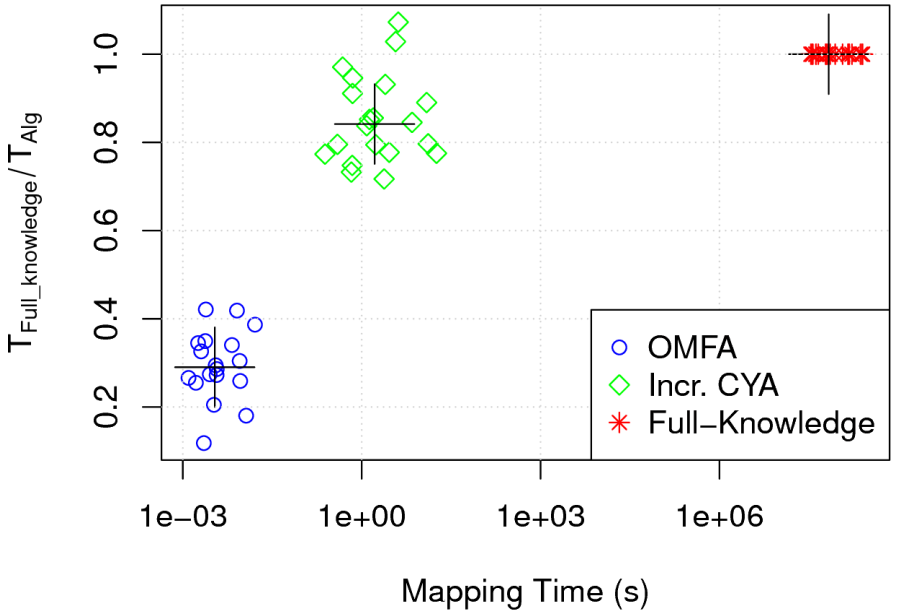


Comparison - Energy



Conclusions

Conclusions



- OMFA mapping has outrageous timing margin.
- FK mapping typically lasts days.
  - Must be repeated for each chip.
- Fast algorithms that eliminate most of timing margin are feasible.
  - CYA achieves >50% of delay and energy gain.
  - Incremental CYA eliminates >50% of energy and >70% delay gain.

Questions?

Thank you