Simulators

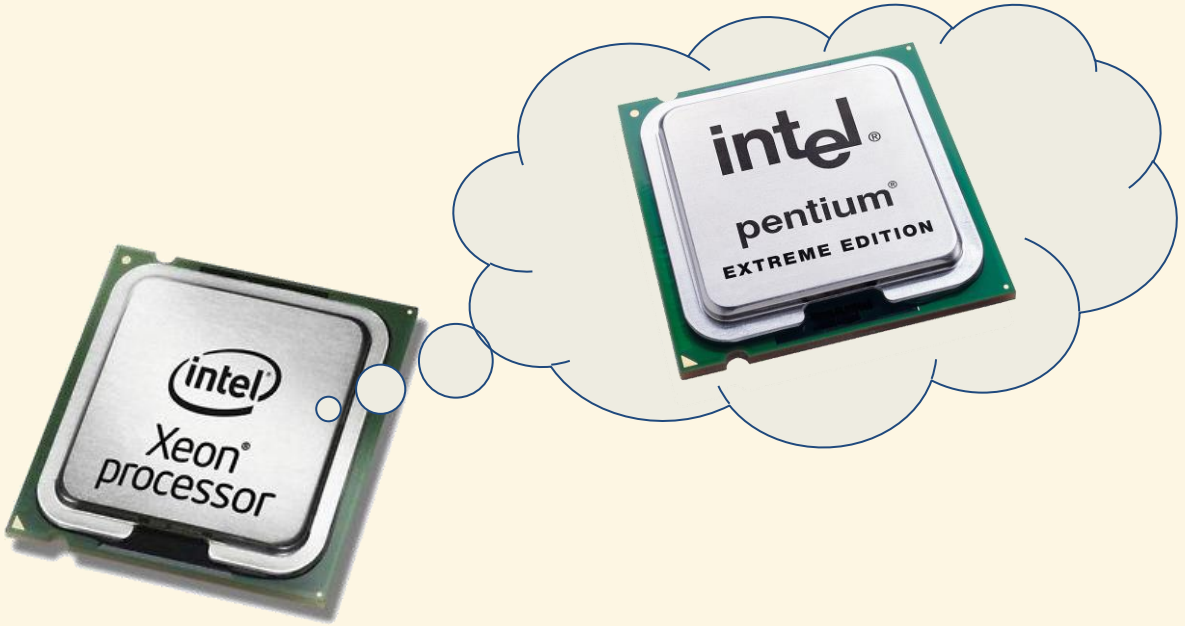
5SIA0

Patrick Wijnings

(Original slides by Luc Waaijen)

Processors Processing Processors

The meta-lecture



Why Simulators?

Your Friend Harm

Why Simulators?

Harm Loves Tractors

Harm

Why Simulators?

Unfortunately for Harm you need to go outside to drive tractors

Harm

The outside world

Why Simulators?

And the outside world is filled with dangers

Harm

The outside world

Why Simulators?

And the outside world is filled with dangers

Harm

The outside world

Why Simulators?

And the outside world is filled with dangers

Harm

The outside world

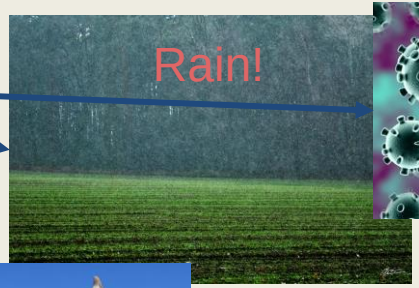
Scary Animals!

Why Simulators?

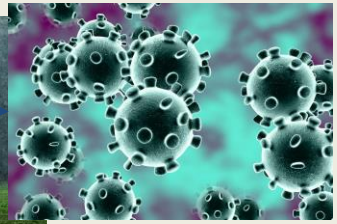
And the outside world is filled with dangers

Harm

The outside world



Rain!



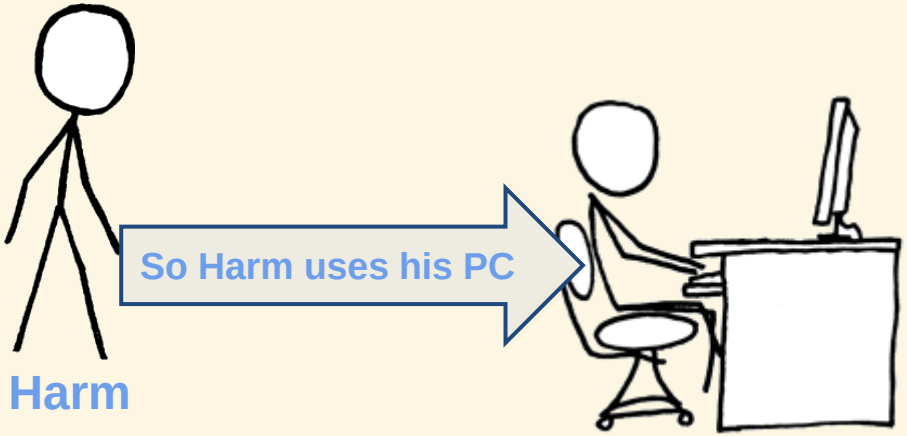
Virusses!

Scary Animals!

Why Simulators?

Harm

Why Simulators?



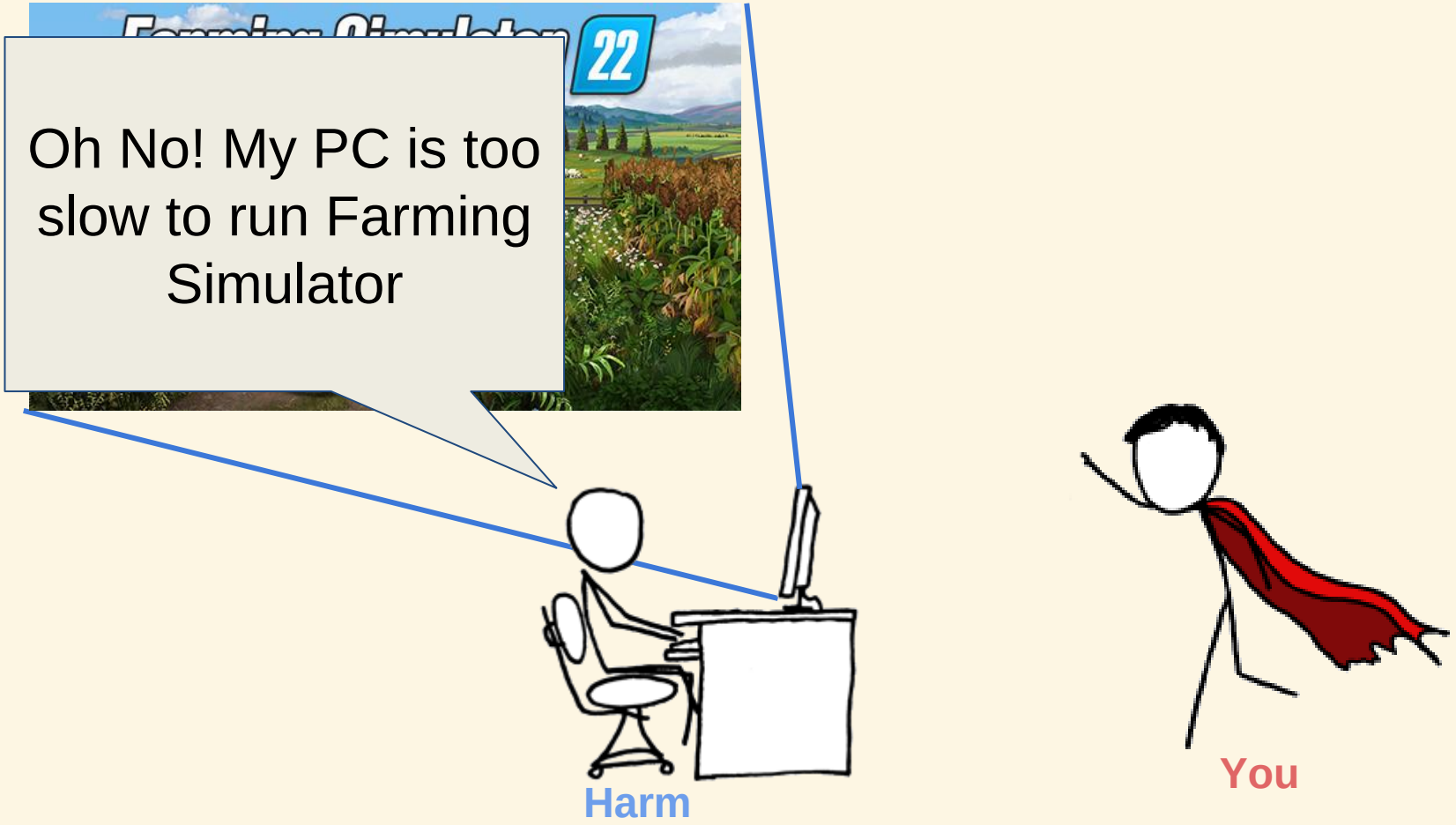
Why Simulators?

Harm

Why Simulators?

Harm

Why Simulators?



Why Simulators?

Oh No! My PC is too slow to run Farming Simulator

Harm

Stand back!
I'm a **computer architect**!

You

How to help Harm?

Of course you have many ideas on how to speed-up Harms computer.

But which ones should you apply?

You

Design Space Exploration Options

Design Space Exploration Options

Buy (or build) all hardware options

Design Space Exploration Options

Buy (or build) all hardware options

Design Space Exploration Options

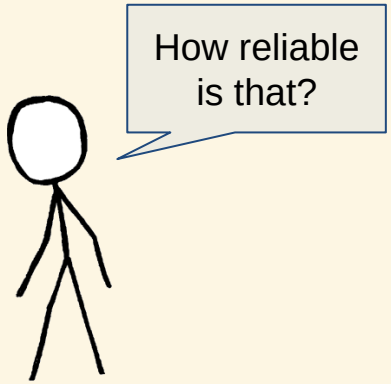
Buy (or build) all hardware options

Use analytical models

Design Space Exploration Options

Buy (or build) all hardware options

Use analytical models



Design Space Exploration Options

Buy (or build) all hardware options

Use analytical models

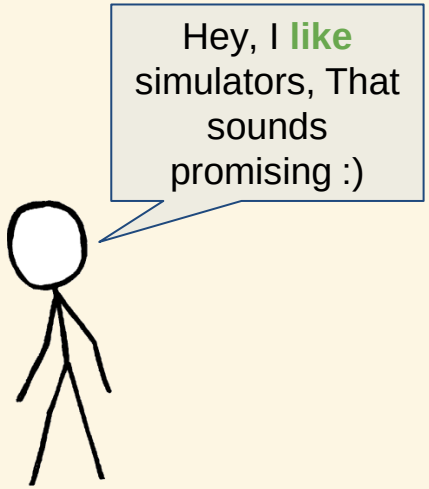
Simulate the design points!

Design Space Exploration Options

Buy (or build) all hardware options

Use analytical models

Simulate the design points!



What to simulate for?

What to simulate for?

- Performance
- Energy
- Power (**!=Energy**)
- Thermal

What to simulate for?

- Performance
- Energy
- Power (**!=Energy**)
- Thermal

What details to simulate?

What to simulate for?

- Performance
- Energy
- Power (**!=Energy**)
- Thermal

What details to simulate?

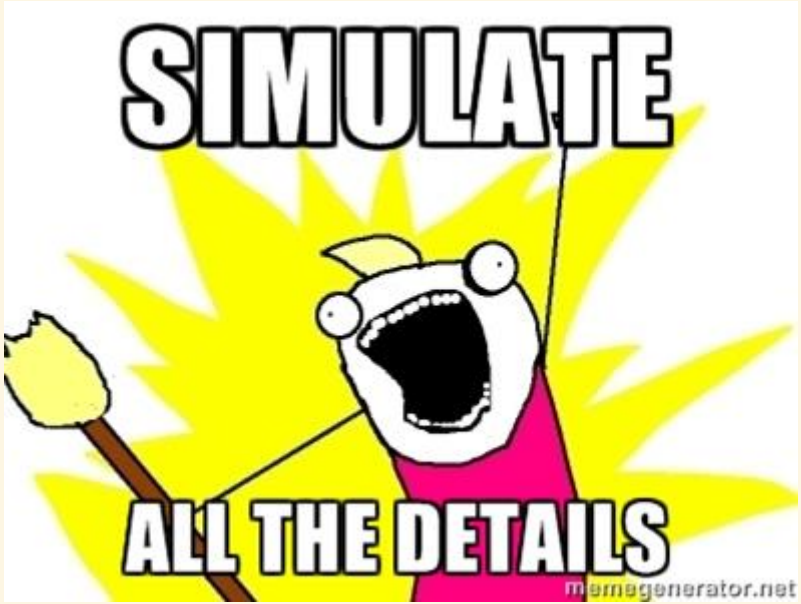
Cycle accurate vs Functionality
Caches
Full operating system
Disk accesses
Background tasks
...

What to simulate for?

- Performance
- Energy
- Power (**!=Energy**)
- Thermal

What details to simulate?

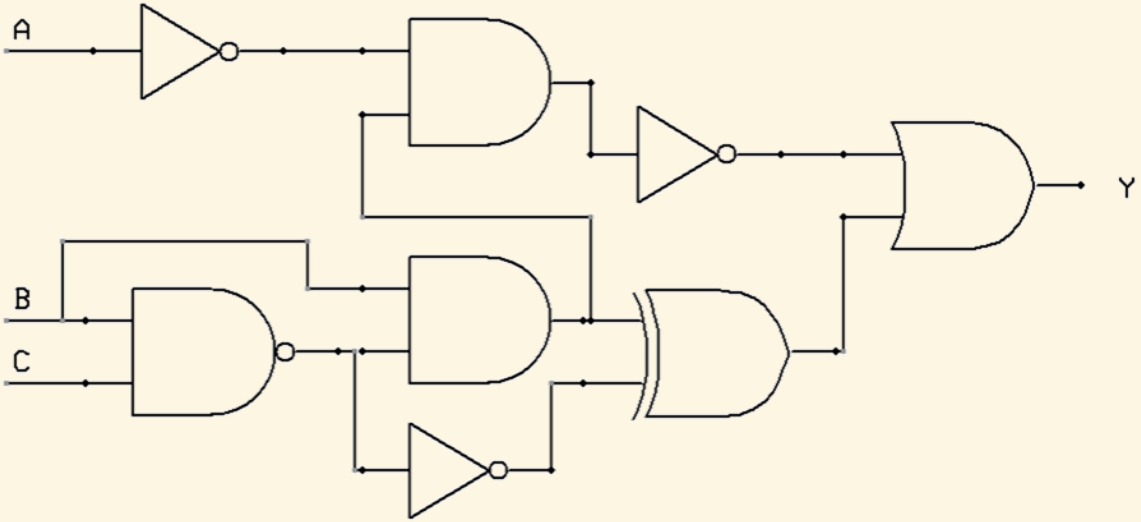
Cycle accurate vs Functionality
Caches
Full operating system
Disk accesses
Background tasks
...



All the details: RTL Simulation

All the details: RTL Simulation

Simulate at **gate level**:



All the details: RTL Simulation

Simulate at **gate level**:

- modelsim/questasim (Mentor)
- ncsim (Cadence)
- VCS (Synopsys)
- Icarus Verilog (Open Source!)
- ...

All the details: RTL Simulation

Simulate at **gate level**:

- modelsim/questasim (Mentor)
- ncsim (Cadence)
- VCS (Synopsys)
- Icarus Verilog (Open Source!)
- ...

Advantages:

- No need to build a custom simulator if you need RTL to build hardware anyway
- Highest level of precision and detail

All the details: RTL Simulation

Simulate at **gate level**:

- modelsim/questasim (Mentor)
- ncsim (Cadence)
- VCS (Synopsys)
- Icarus Verilog (Open Source!)
- ...

Advantages:

- No need to build a custom simulator if you need RTL to build hardware anyway
- Highest level of precision and detail

Disadvantage:

- Horribly slow for realistic designs

All the details: RTL Simulation

Simulate at **gate level**:

- modelsim/questasim (Mentor)
- ncsim (Cadence)
- VCS (Synopsys)
- Icarus Verilog (Open Source!)
- ...

Advantages:

- No need to build a custom simulator if you need RTL to build hardware anyway
- Highest level of precision and detail

Disadvantage:

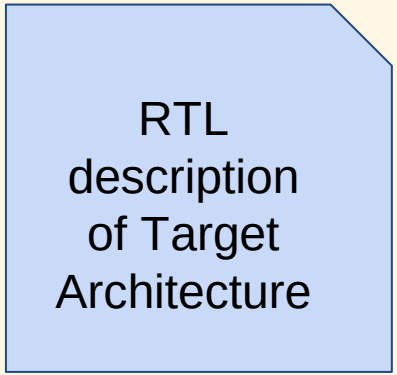
- Horribly slow for realistic designs

Nvidia GPU with > 1 Billion transistors
Small tests take over 8 hours! [1]

[1] http://www.deepchip.com/items/0523-04.html

modified from http://xkcd.com/303/

Slightly less horribly slow: Hardware Emulation



Slightly less horribly slow: Hardware Emulation

RTL description of Target Architecture

Synthesize for FPGA (slow)

Slightly less horribly slow: Hardware Emulation

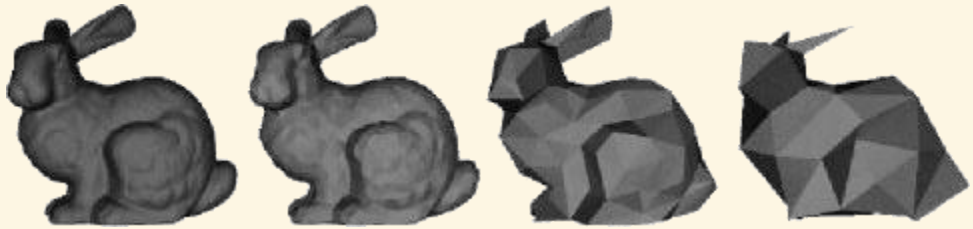
RTL description of Target Architecture

Synthesize for FPGA (slow)

Emulate on FPGA (fast!)
Note: instrumentation required to get detailed information out!

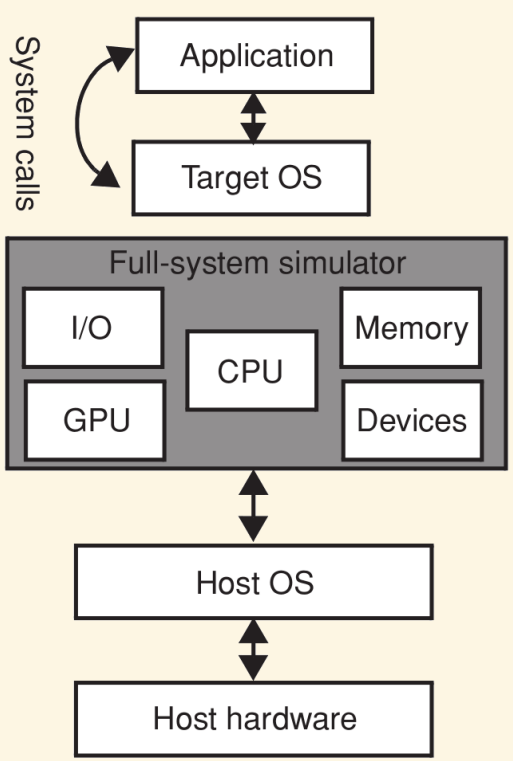
Levels of detail in Simulation

- Full-System versus User-level
- Cycle Accurate versus Functional
- Execution- versus Trace-driven



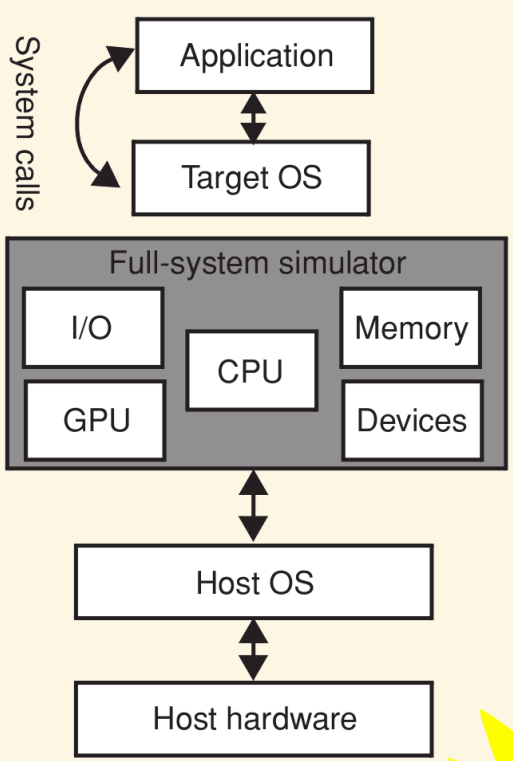
To OS or not to OS?

To OS or not to OS?



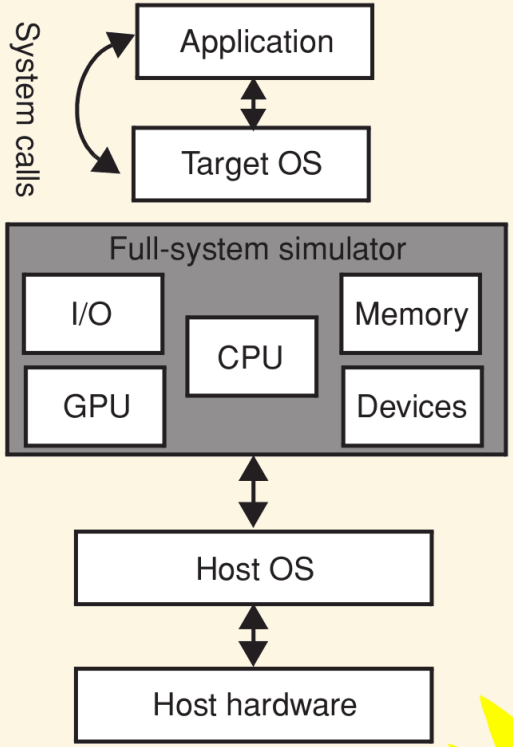
Full-System

To OS or not to OS?

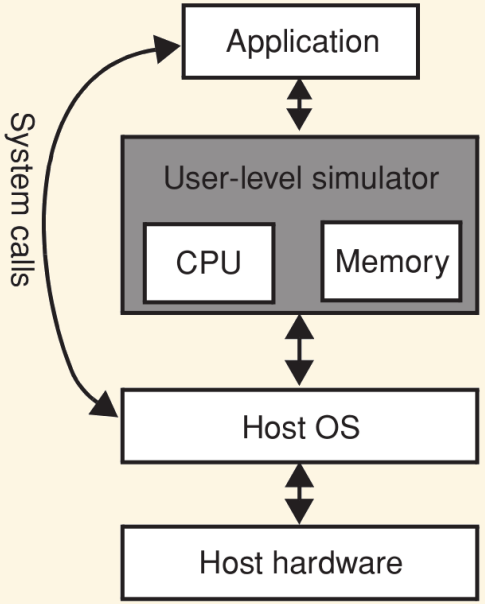


Full-System

To OS or not to OS?



Full-System



User-Level

User-Level

Famous example: **Simple Scalar [1]**

Advantages

- Fast to develop and update to new architectures
- Usually ‘accurate enough’

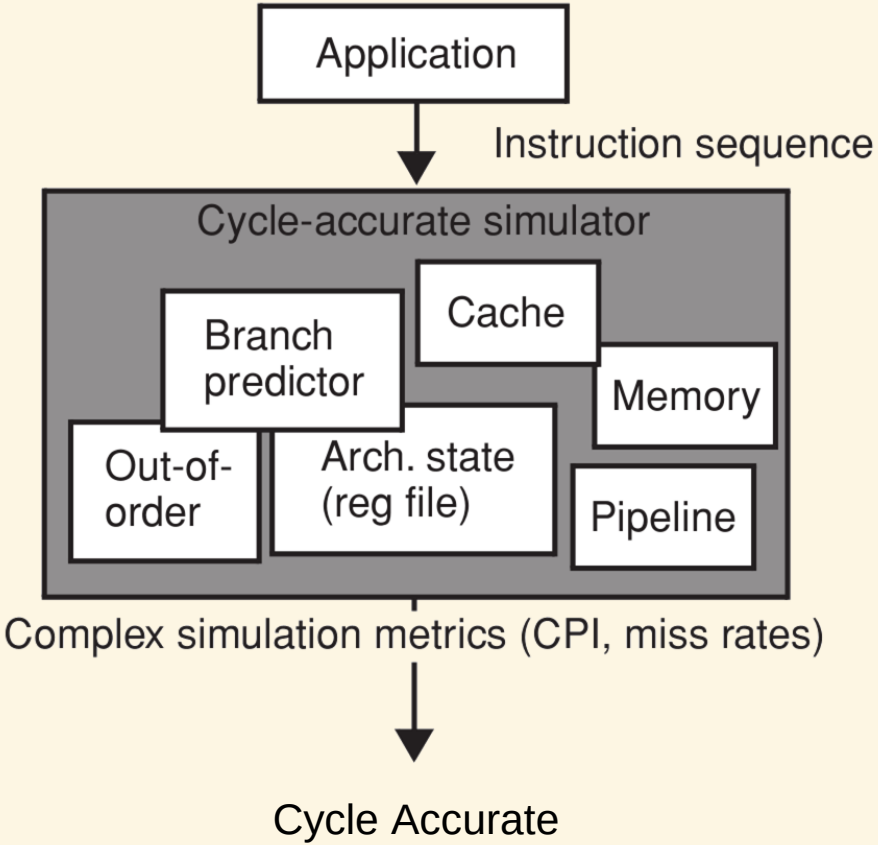
Disadvantages

- Any time spent in the OS is not modelled accurately. Can have severe impact, database applications spent 20-30% of their time in OS mode.

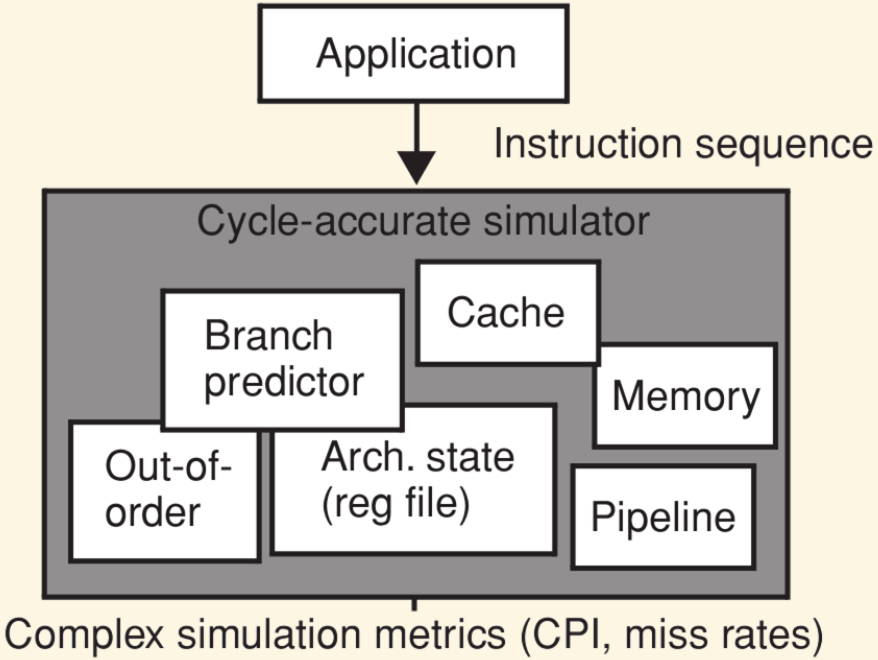
[1] http://www.simplescalar.com/

Cycle Accurate versus Functional

Cycle Accurate versus Functional

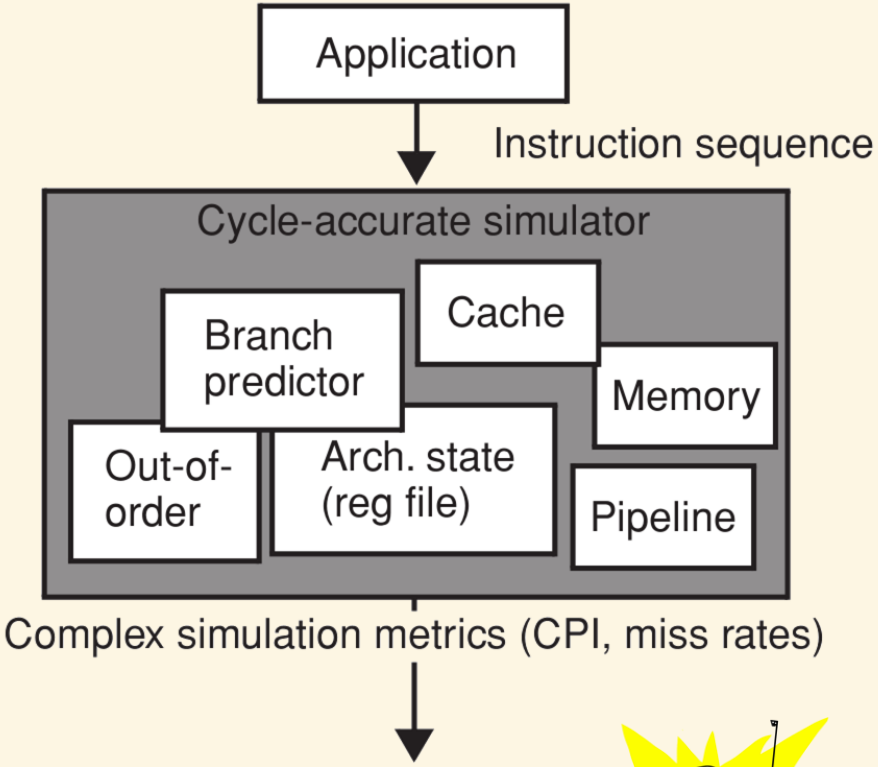


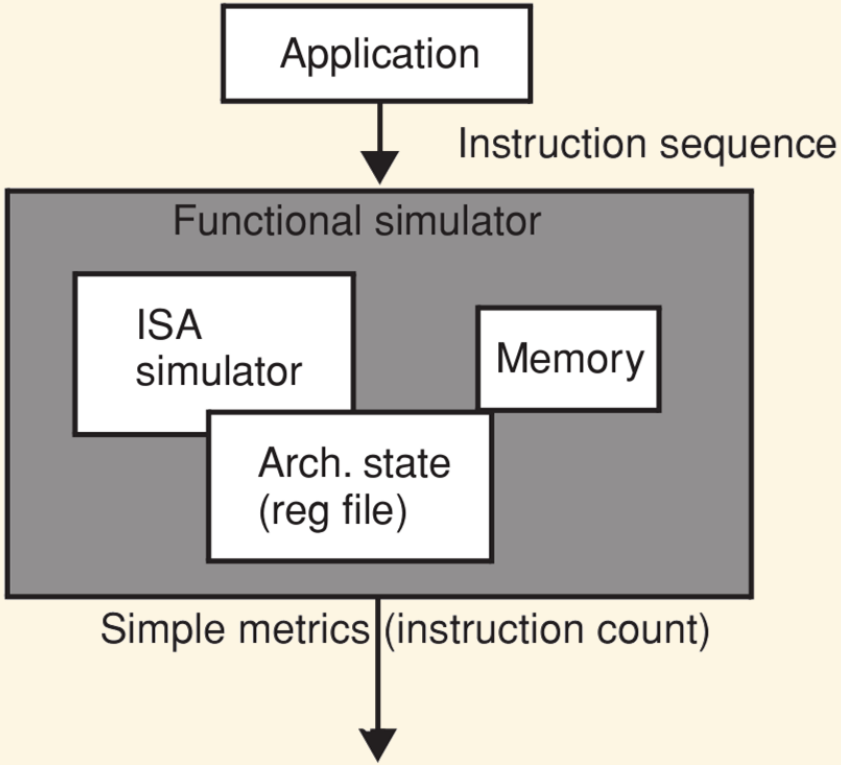
Cycle Accurate versus Functional



Cycle Accurate

Cycle Accurate versus Functional





Cycle Accurate versus Functional

Functional - no/limited model of the micro architecture

An (add) instruction of the target can be translated to an (add) instruction on the host, and be simulated that way.

Example 1: Simple Scalar sim-fast

Example 2: QEMU, Full-system emulator using dynamic translation

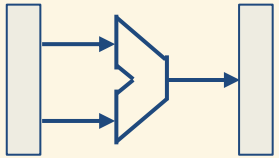
Cycle Accurate - includes model of the micro architecture

Block resources in the pipeline when instruction executes

Use target branch predictor scheme

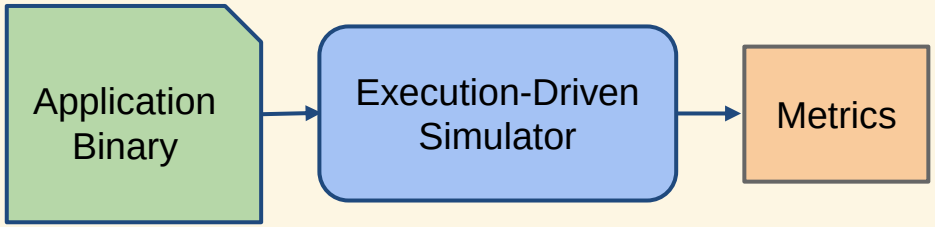
Out-of-order execution

Example: Simple Scalar sim-outorder



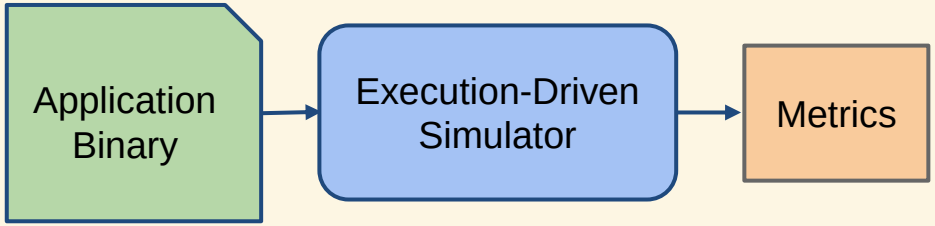
Execution- versus Trace-driven

Execution- versus Trace-driven

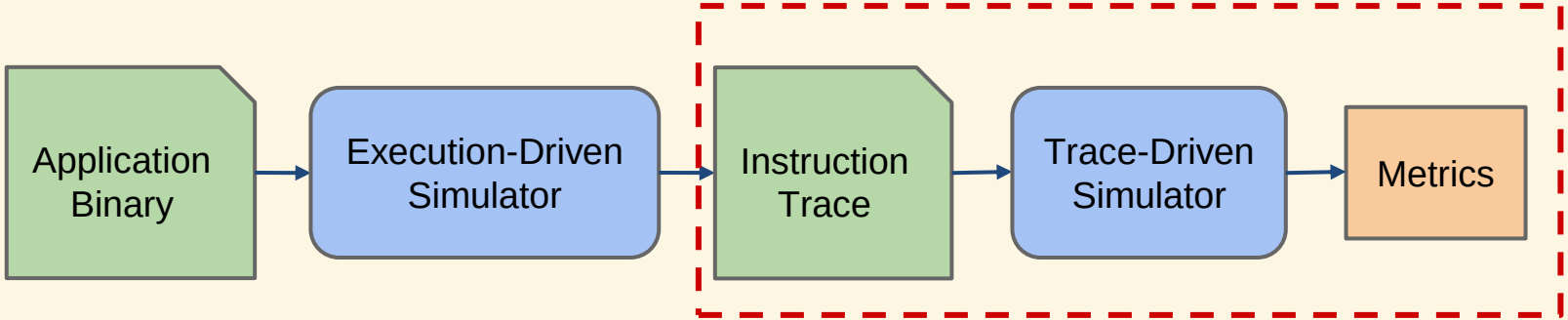


Execution Driven: application executes on simulator

Execution- versus Trace-driven

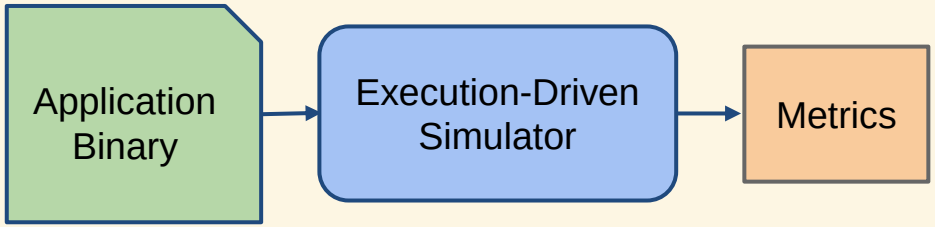


Execution Driven: application executes on simulator



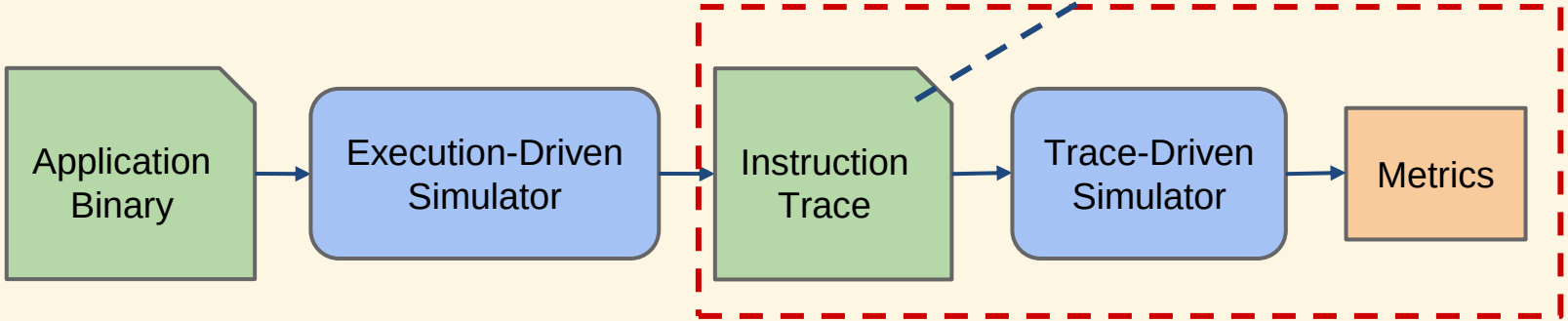
Trace Driven: simulator uses trace as input

Execution- versus Trace-driven



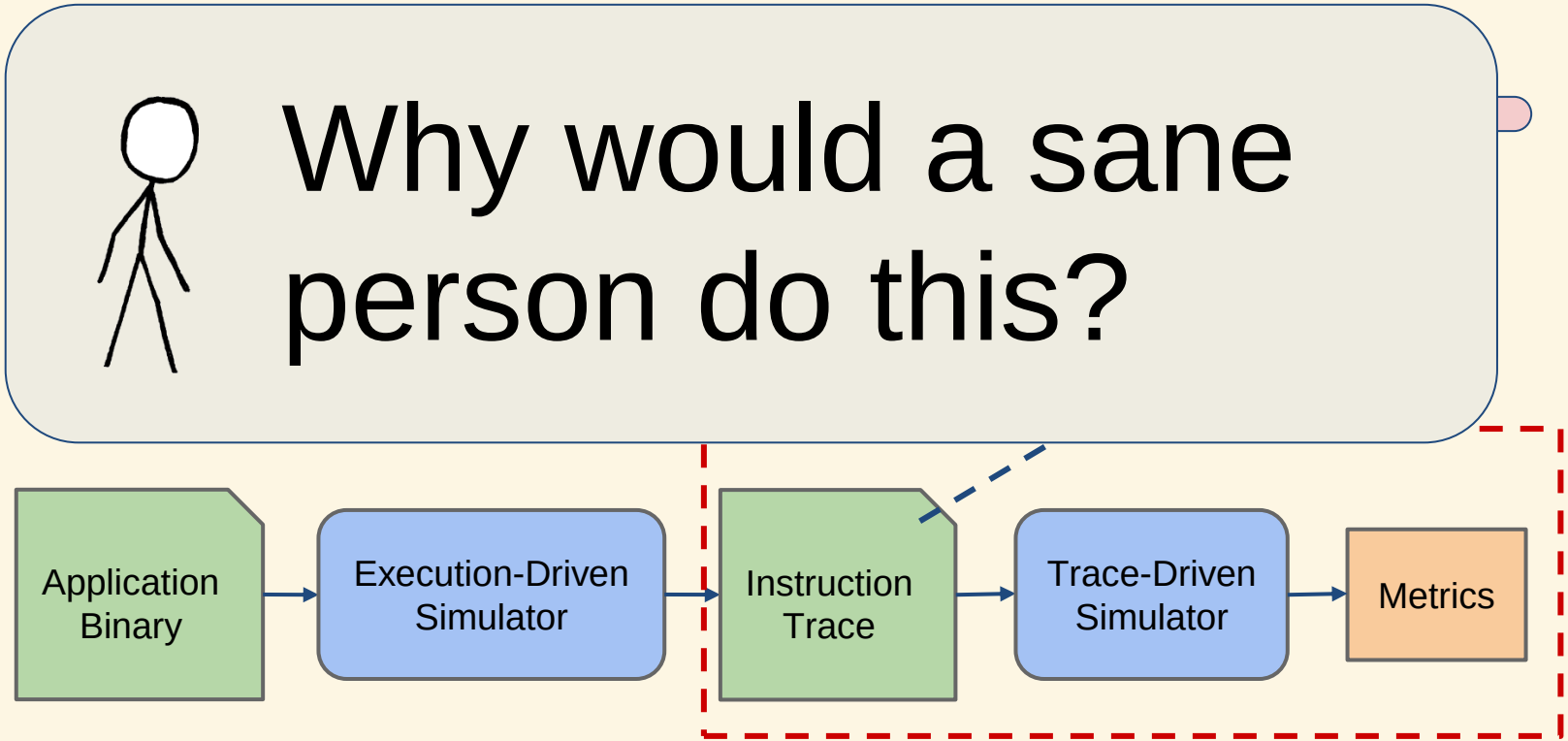
Execution Driven: application executes on simulator

```
mov edx,len
mov ecx,msg
mov ebx,1
mov eax,4
int 0x80
```



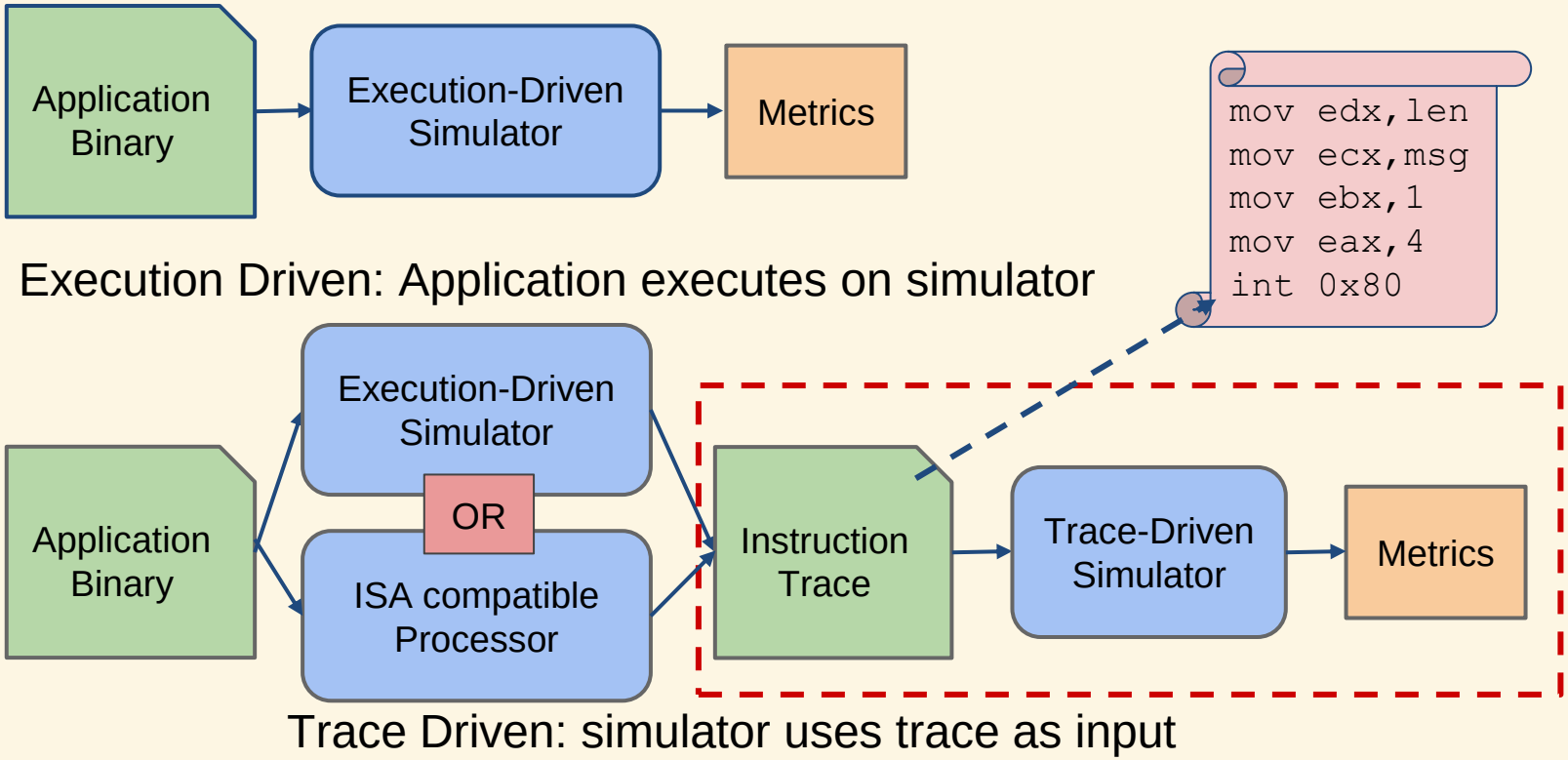
Trace Driven: simulator uses trace as input

Execution- versus Trace-driven



Trace Driven: simulator uses trace as input

Execution- versus Trace-driven



Trace-driven Simulation

Advantages

- Trace collection only required once
- Trace collection can be done with ISA compatible processor
- Trace simulator does not need to simulate all instructions, can skip ahead in trace if not implemented

Trace-driven Simulation

Advantages

- Trace collection only required once
- Trace collection can be done with ISA compatible processor
- Trace simulator does not need to simulate all instructions, can skip ahead in trace if not implemented

Disadvantages

- Cannot speculatively execute code (trace is fixed)
- Trace file can become huge for large applications (hundreds of GBs)

Mixing Simulation Strategies

Direct-execution

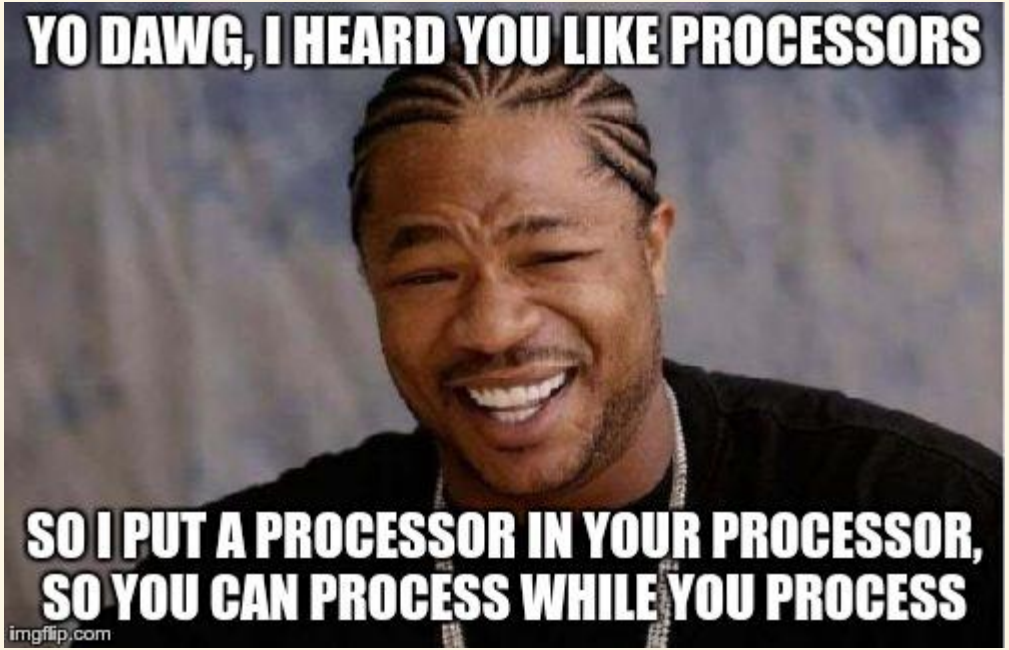
Parts execute directly on the host (e.g. using dynamic translation such as QEMU)

Other parts are executed on cycle accurate simulation

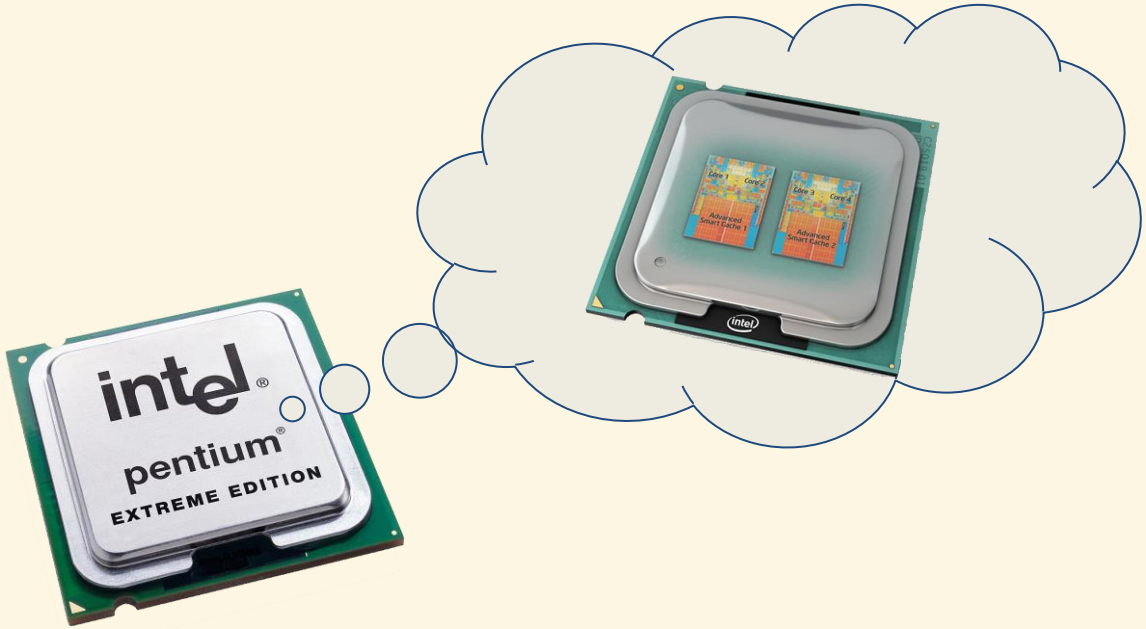
Use case:

Interested in memory accesses and memory behavior. Execute only loads and stores on the simulator, emulate the rest directly on the host machine

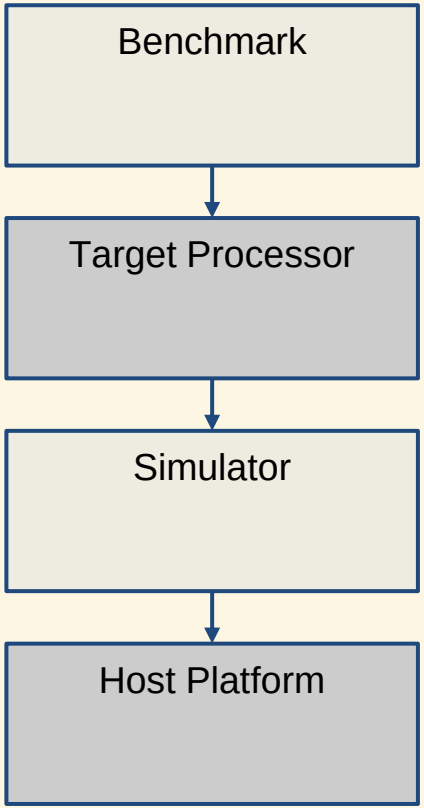
Simulation in the Multiprocessor Era



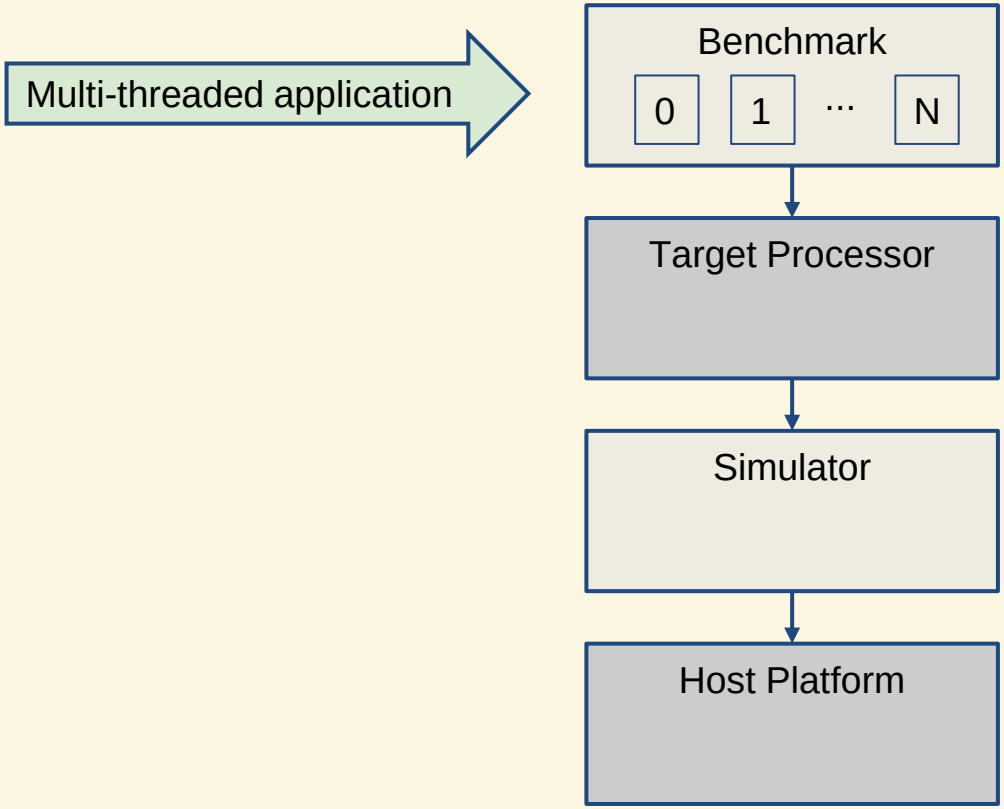
Simulation in the Multiprocessor Era



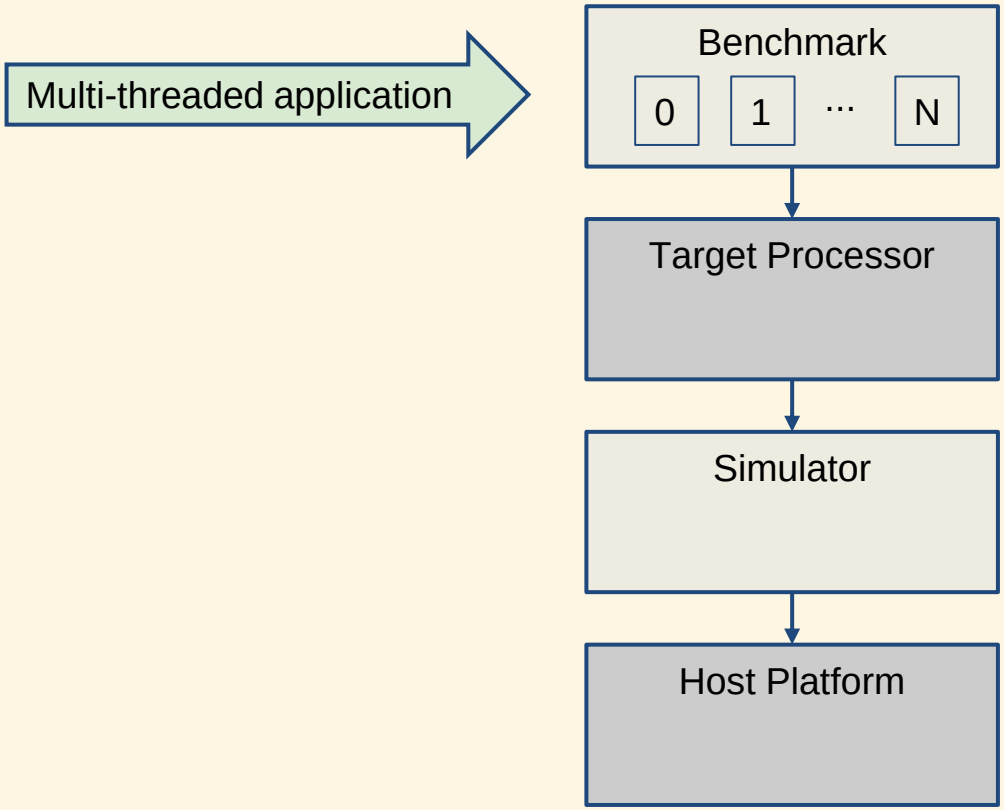
Parallelisation in all levels of the simulation stack



Parallelisation in all levels of the simulation stack



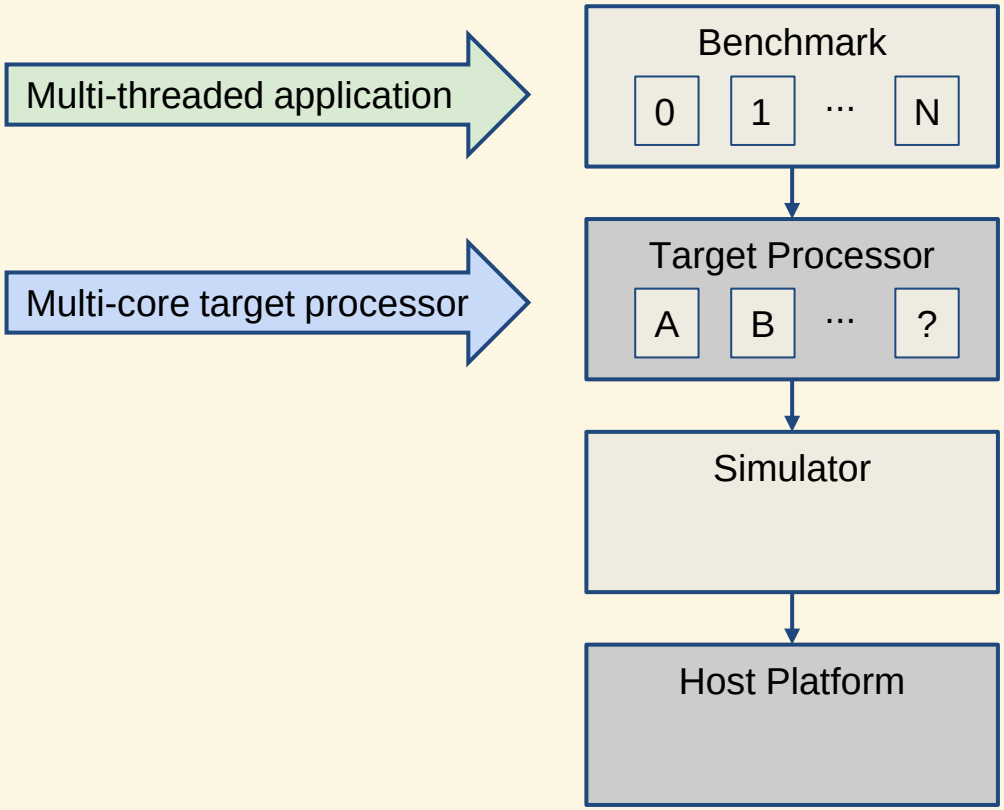
Parallelisation in all levels of the simulation stack



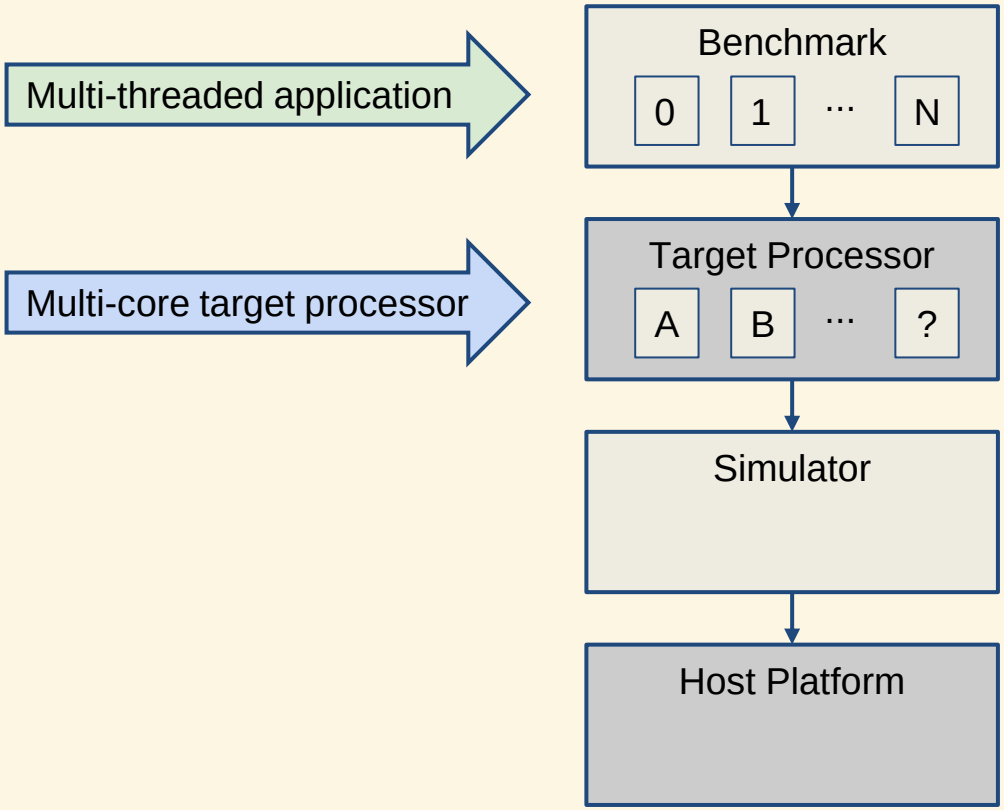
A multi-threaded application running on a single core target processor.

Question: Does this make sense?

Parallelisation in all levels of the simulation stack



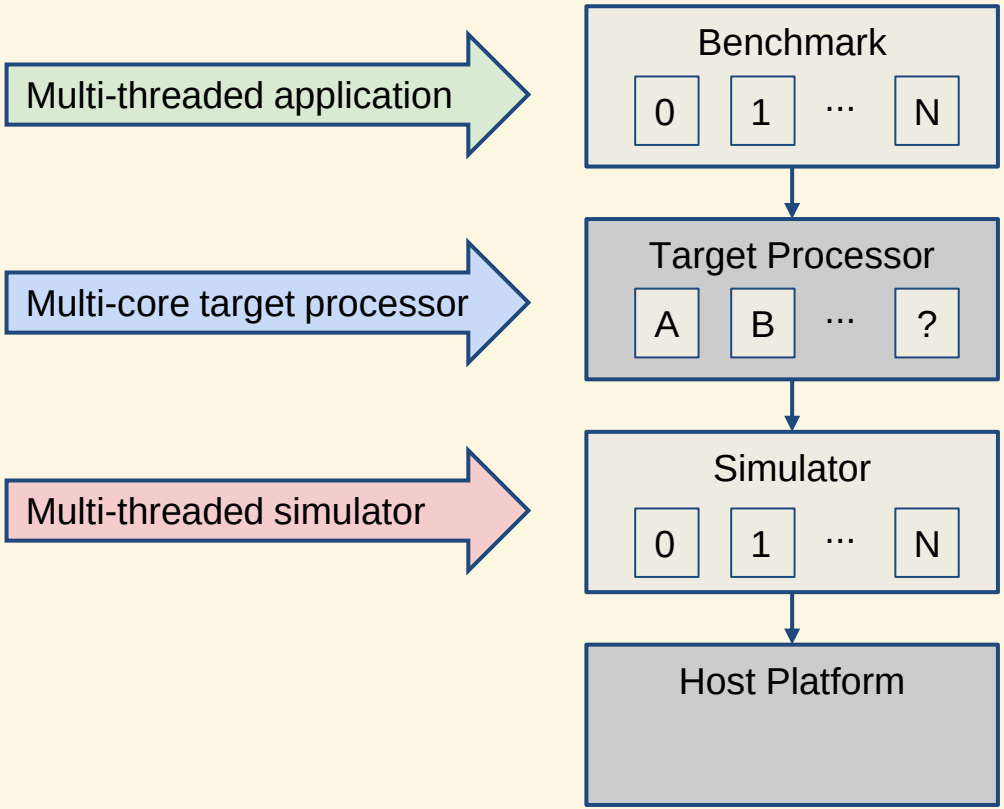
Parallelisation in all levels of the simulation stack



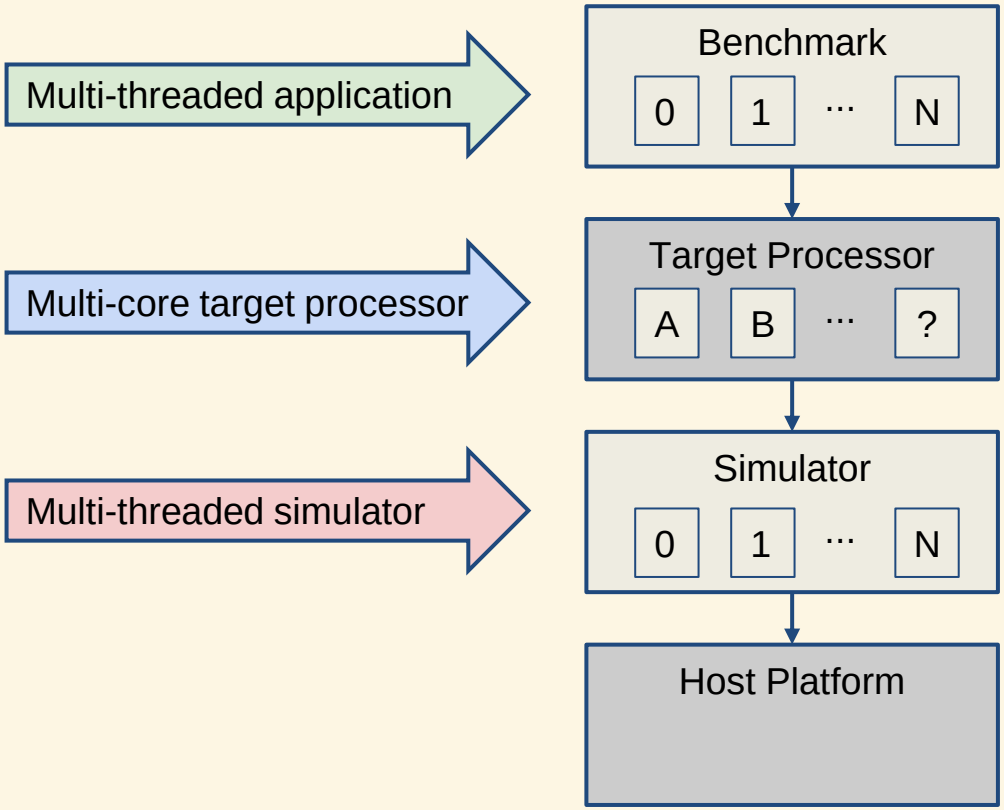
A multi-core processor running on a single threaded simulator.

Question: Does this make sense?

Parallelisation in all levels of the simulation stack



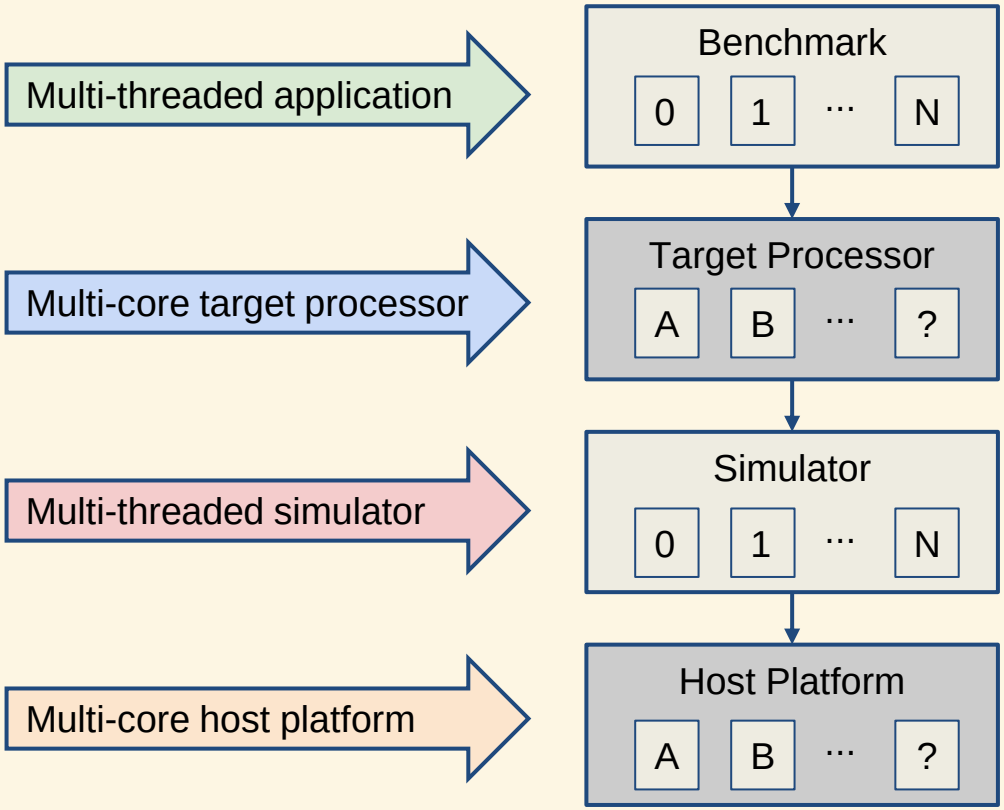
Parallelisation in all levels of the simulation stack



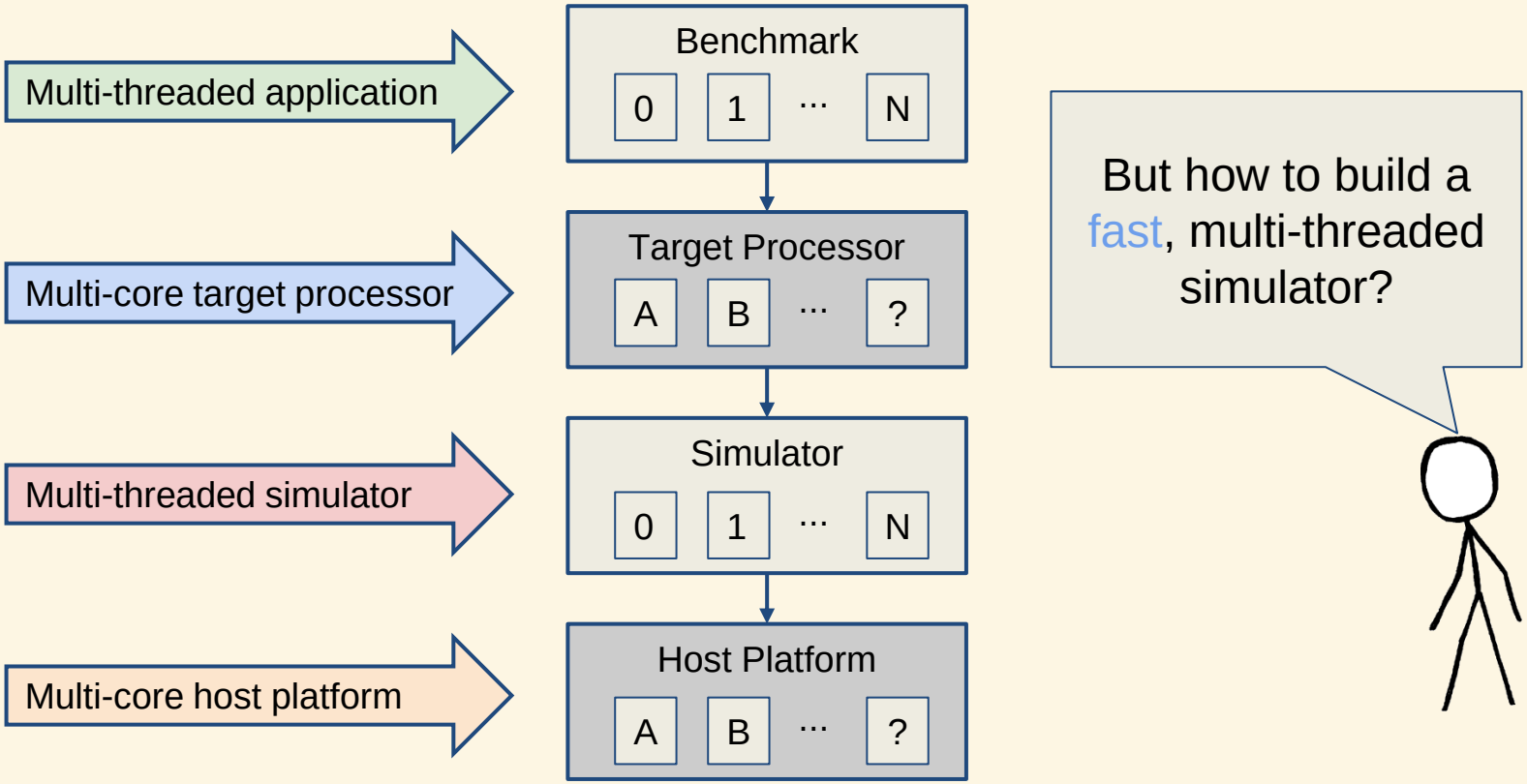
A multi-threaded simulator running on a single-core host.

Question: Does this make sense?

Parallelisation in all levels of the simulation stack

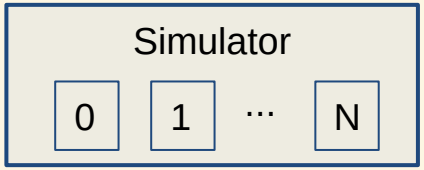


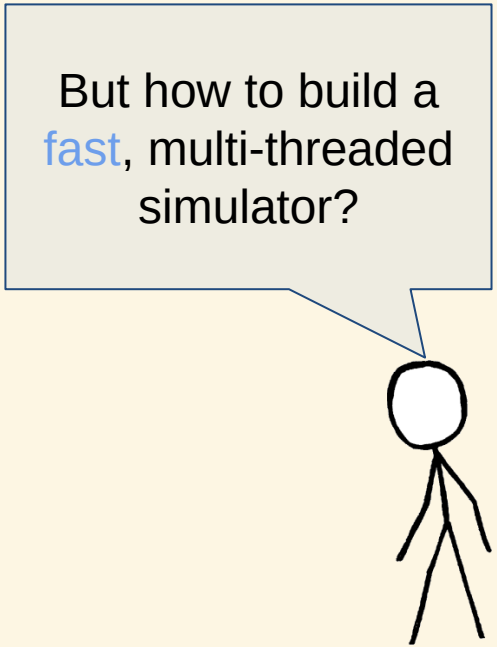
Parallelisation in all levels of the simulation stack



Parallelisation in all levels of the simulation stack

Multi-threaded simulator





Parallel Simulation Techniques

- Discrete event simulation
- Quantum simulation
- Slack simulation

Parallel Simulation Techniques

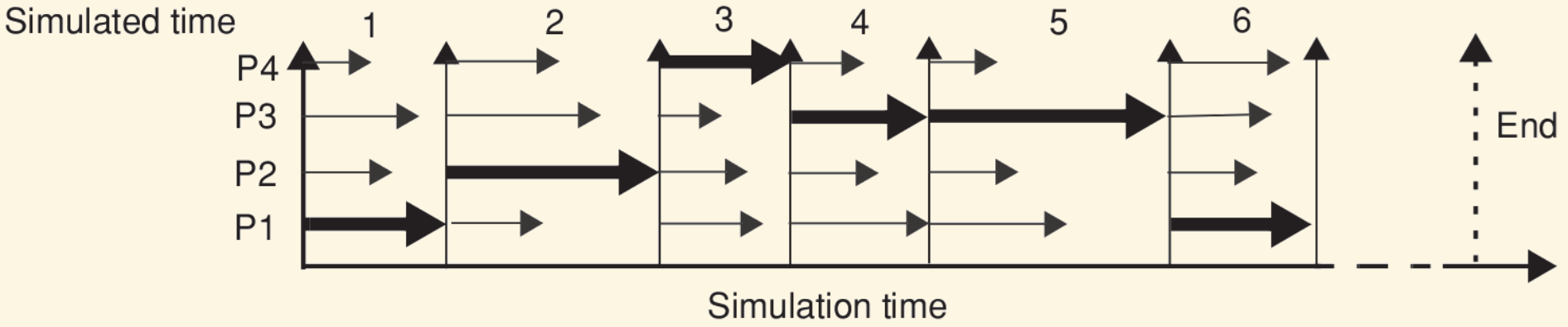
Discrete event simulation

Quantum simulation

Slack simulation

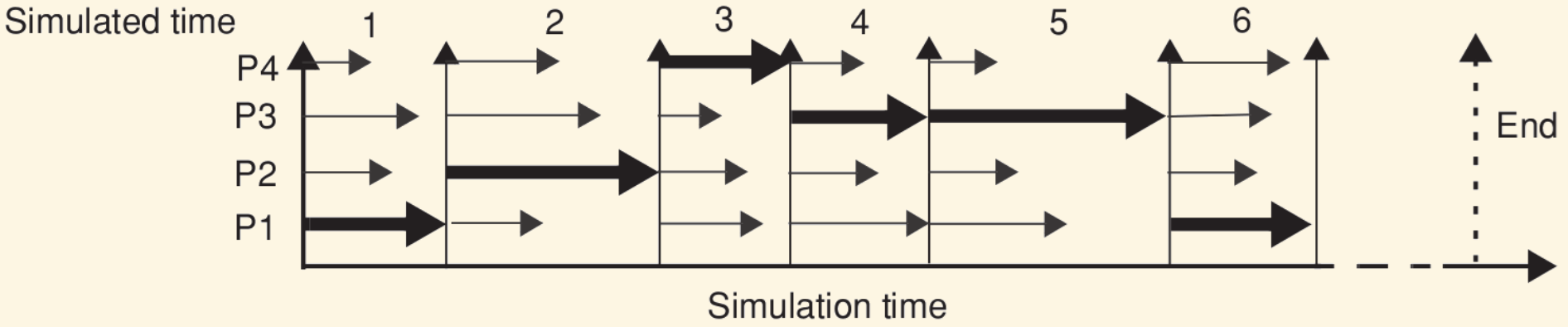
Not schrödinger's cat
‘quantum’ though

Discrete-Event Simulation



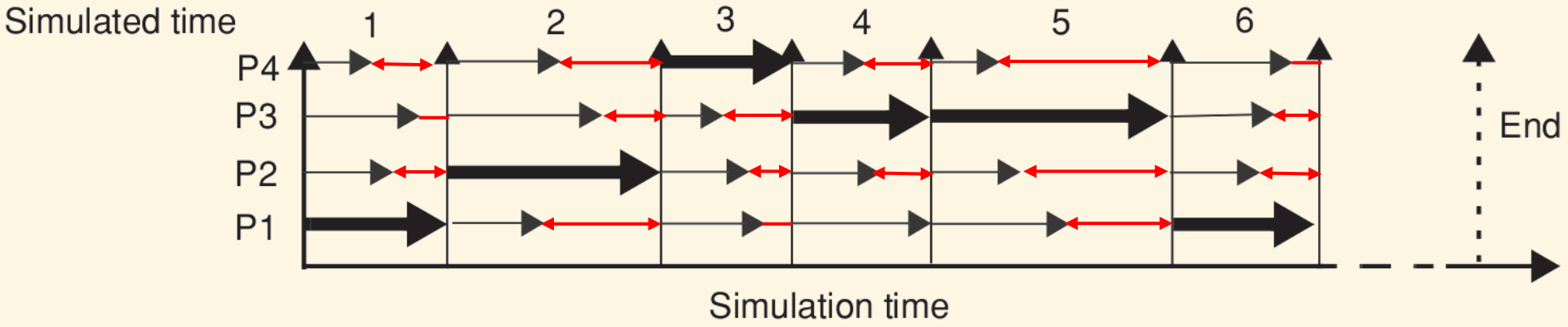
A logical choice for a simulator “time step” is one cycle for the fastest core.

Discrete-Event Simulation



Disadvantage

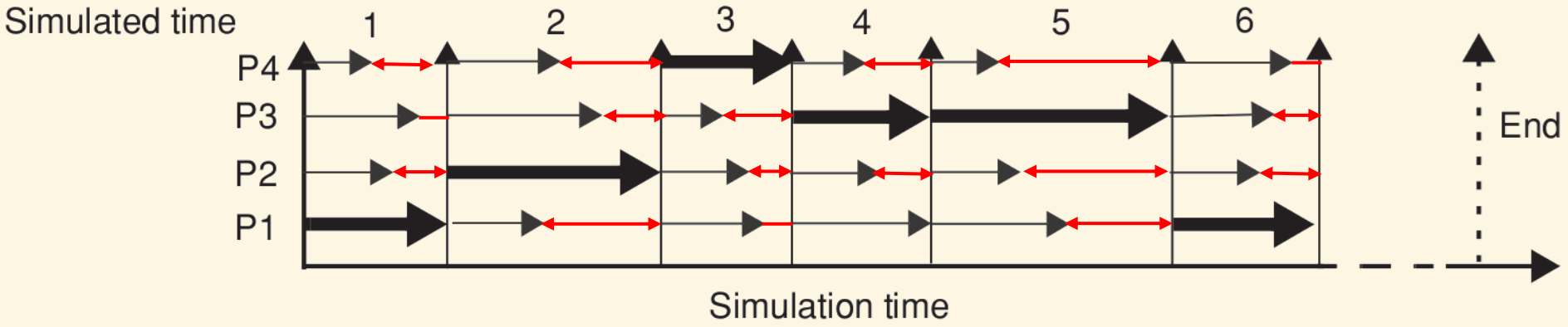
Discrete-Event Simulation



Disadvantage

Under utilisation of the host platform if threads are idle for synchronisation

Discrete-Event Simulation



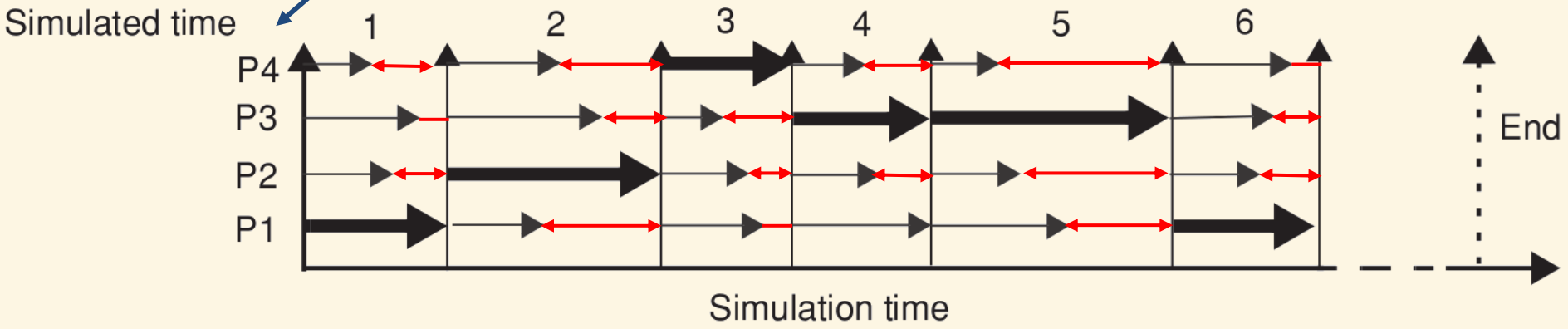
Is it really this bad? What assumption was made here?

Disadvantage

Under utilisation of the host platform if threads are idle for synchronisation

Discrete-Event Simulation

Every target processor Pn is mapped to a separate host core



Is it really this bad? What assumption was made here?

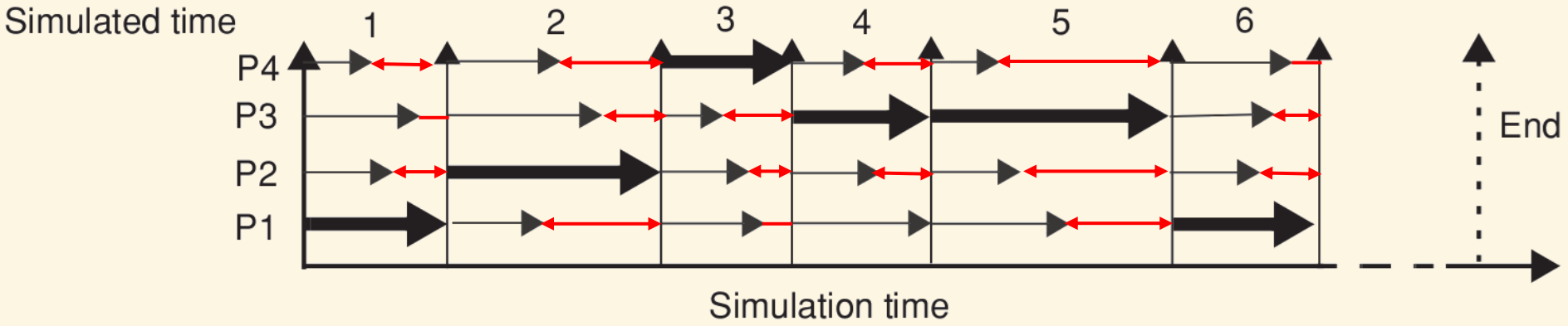
Disadvantage

Under utilisation of the host platform if threads are idle for synchronisation

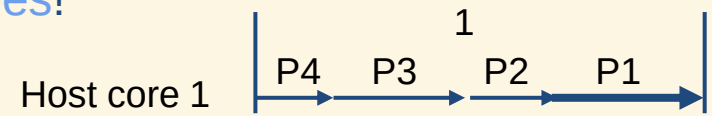
Target vs Host Cores

There is **no relation** between the number of target cores and the number of host cores!!!

Discrete-Event Simulation

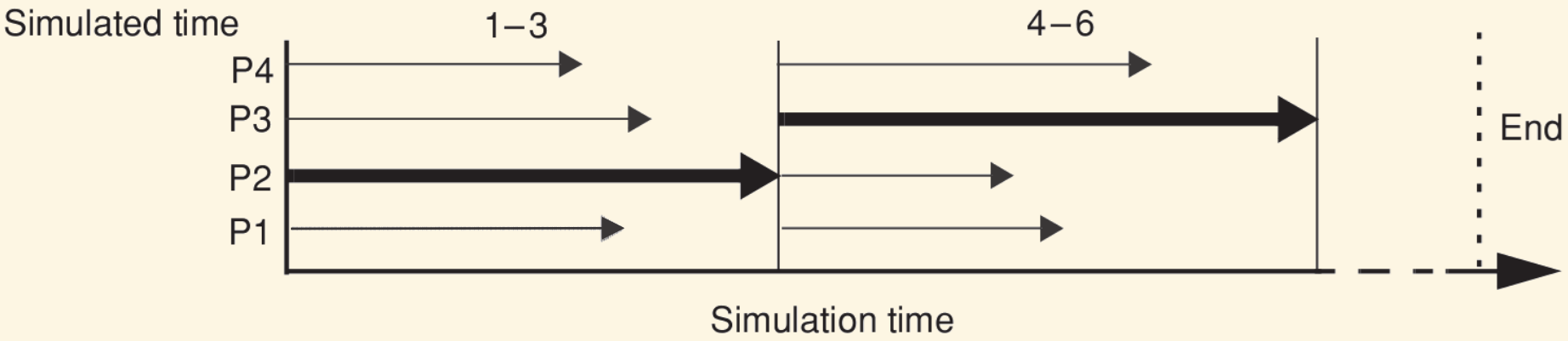


Utilisation of host depends on variation in processing time of a cycle, but also on the amount of host cores!



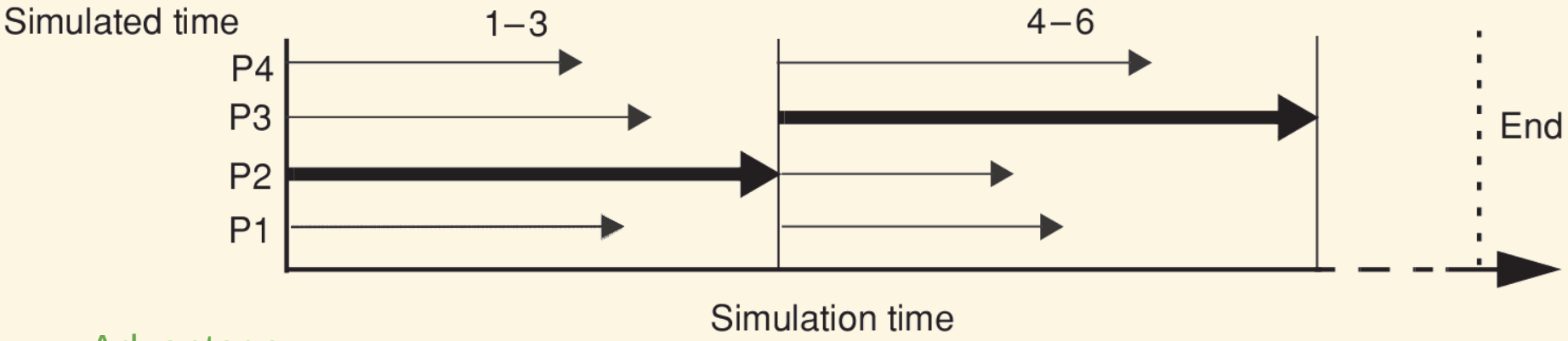
Quantum Simulation

Synchronize threads at larger time-steps, e.g. 3 cycles



Quantum Simulation

Synchronize threads at larger time-steps, e.g. 3 cycles



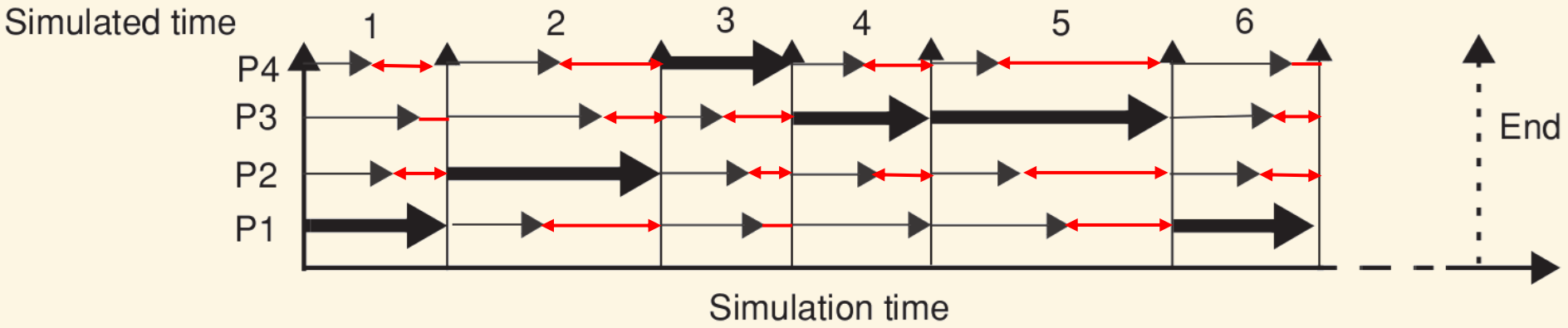
Advantage

Utilisation improves, because the variation of processing is amortized over longer sections of simulation

Disadvantage

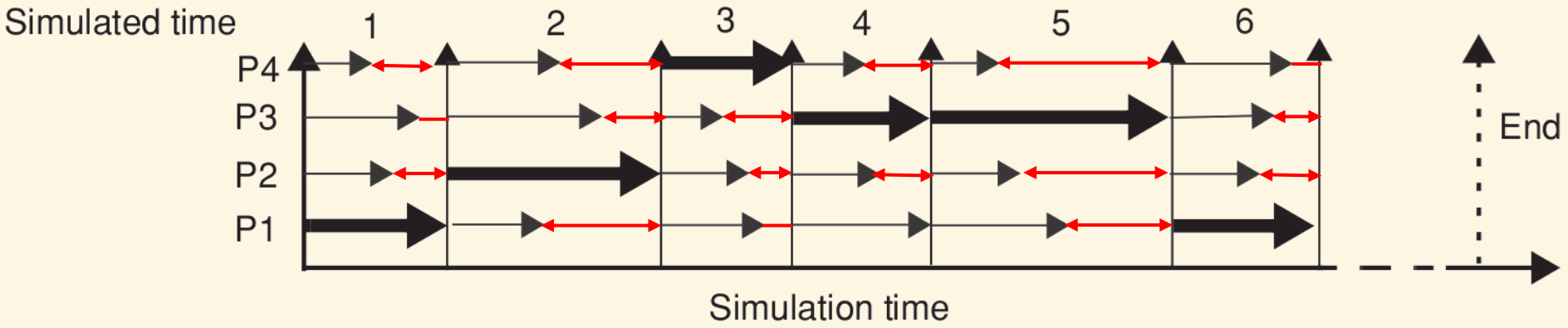
No longer cycle accurate

Slack Simulation



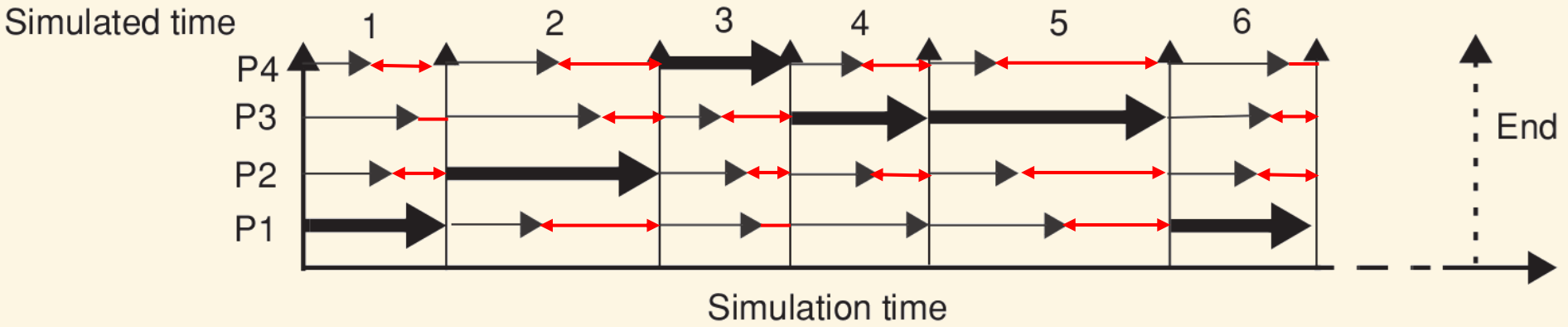
Start with discrete-event simulation schedule

Slack Simulation

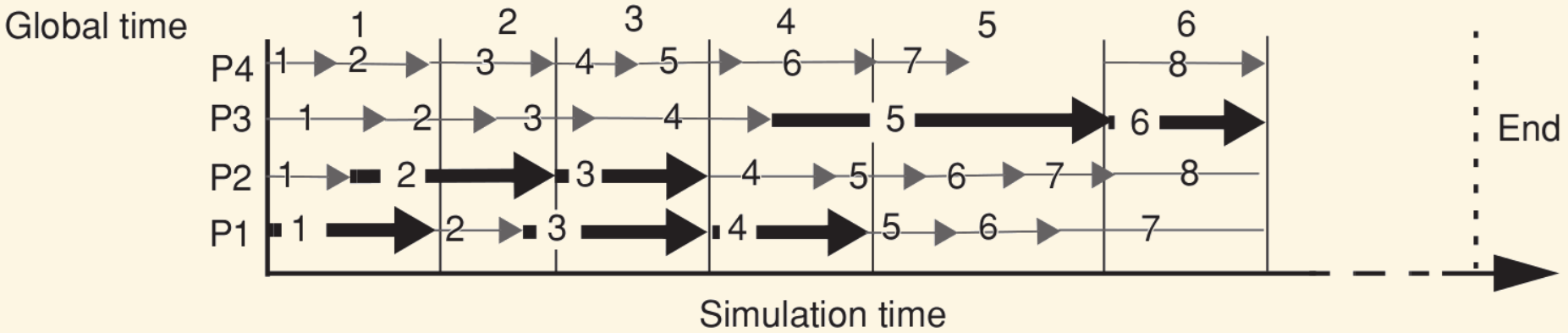


Instead of waiting in the red areas, use slack to process ahead

Slack Simulation



Instead of waiting in the red areas, use slack to process ahead



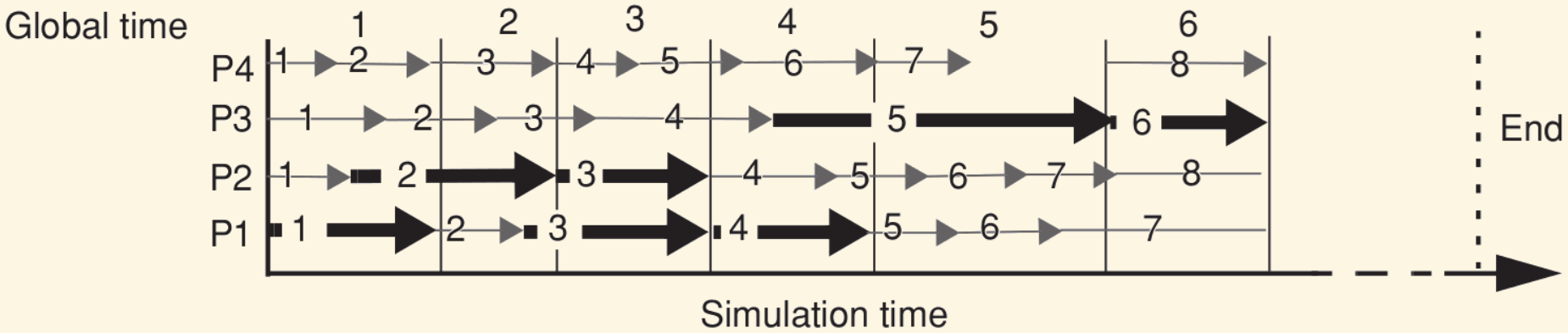
Slack Simulation

Side-effect: Drift

The cores might be simulating different points in time, and could drift apart

Mitigation

Allow a maximum drift (or slack), and synchronize when this value is exceeded



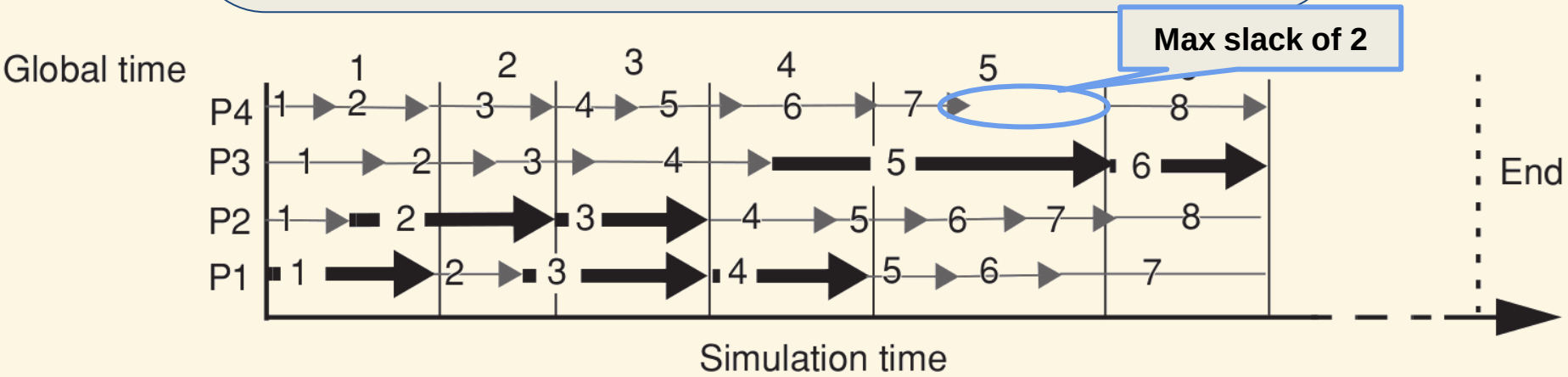
Slack Simulation

Side-effect: Drift

The cores might be simulating different points in time, and could drift apart

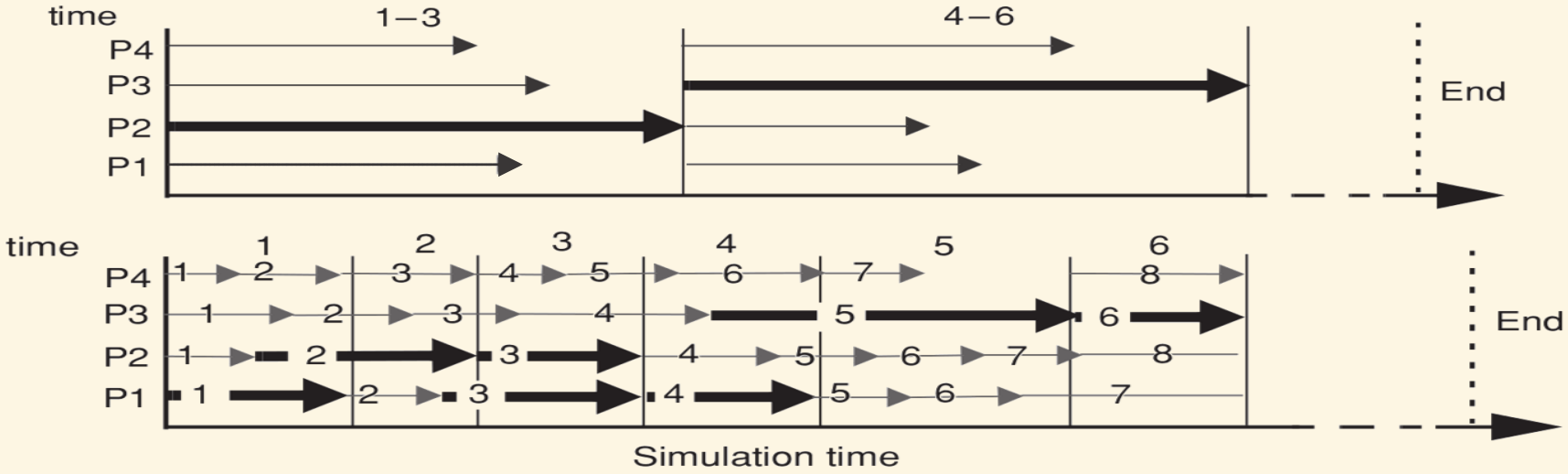
Mitigation

Allow a maximum drift (or slack), and synchronize when this value is exceeded



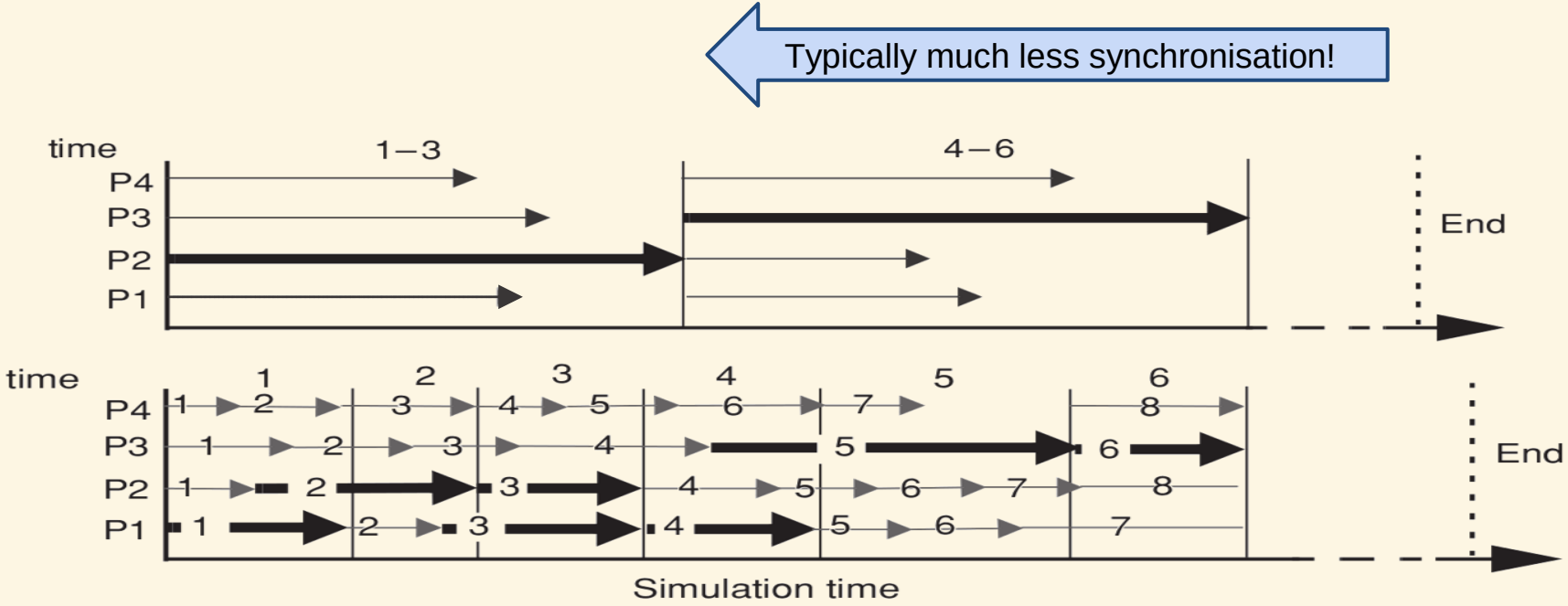
Slack versus Quantum simulation

In quantum simulation, the core simulation times always stay within a cycle window, which is fixed in global time.
Also in slack simulation the simulation times stay within a window, but with the key difference that this is a sliding window.



Slack versus Quantum simulation

In quantum simulation, the core simulation times always stay within a cycle window, which is fixed in global time.
Also in slack simulation the simulation times stay within a window, but with the key difference that this is a sliding window.



Still not good enough

From the paper ***Graphite: a Distributed Parallel Simulator for Multicores***

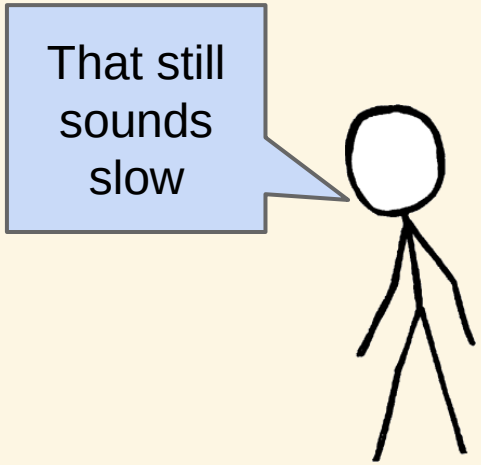
"Simulation slowdown is as low as 41× versus native execution"

[1] *Graphite: A Distributed Parallel Simulator for Multicores* - Jason E. Miller et al.

Still not good enough

From the paper ***Graphite: a Distributed Parallel Simulator for Multicores***

"Simulation slowdown is as low as 41× versus native execution"

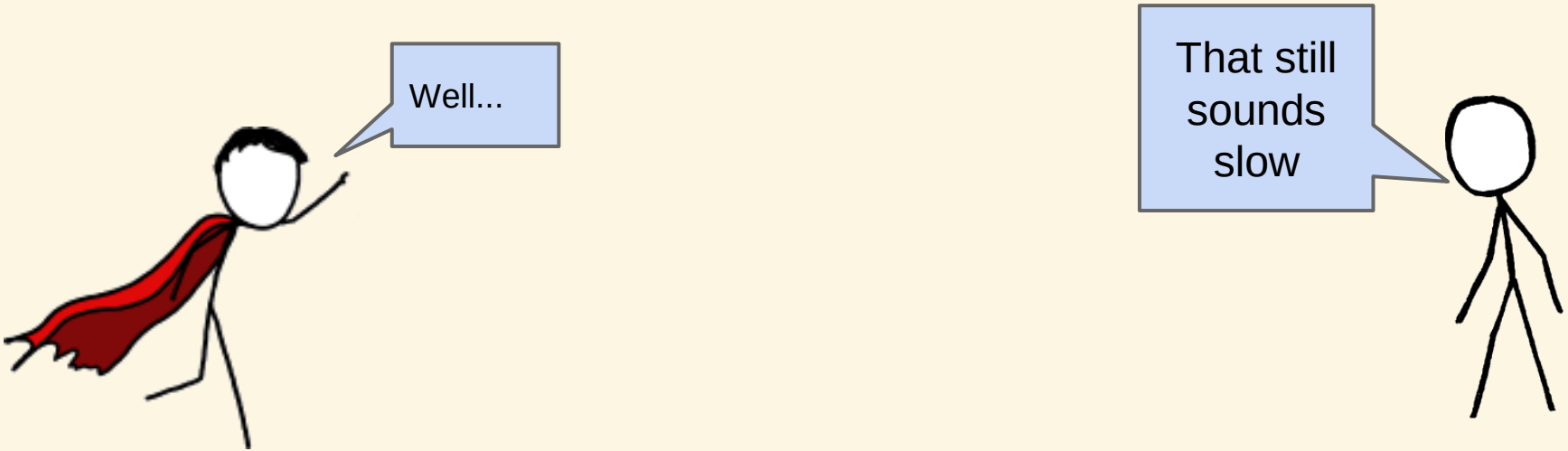


[1] *Graphite: A Distributed Parallel Simulator for Multicores* - Jason E. Miller et al.

Still not good enough

From the paper ***Graphite: a Distributed Parallel Simulator for Multicores***

"Simulation slowdown is as low as 41× versus native execution"

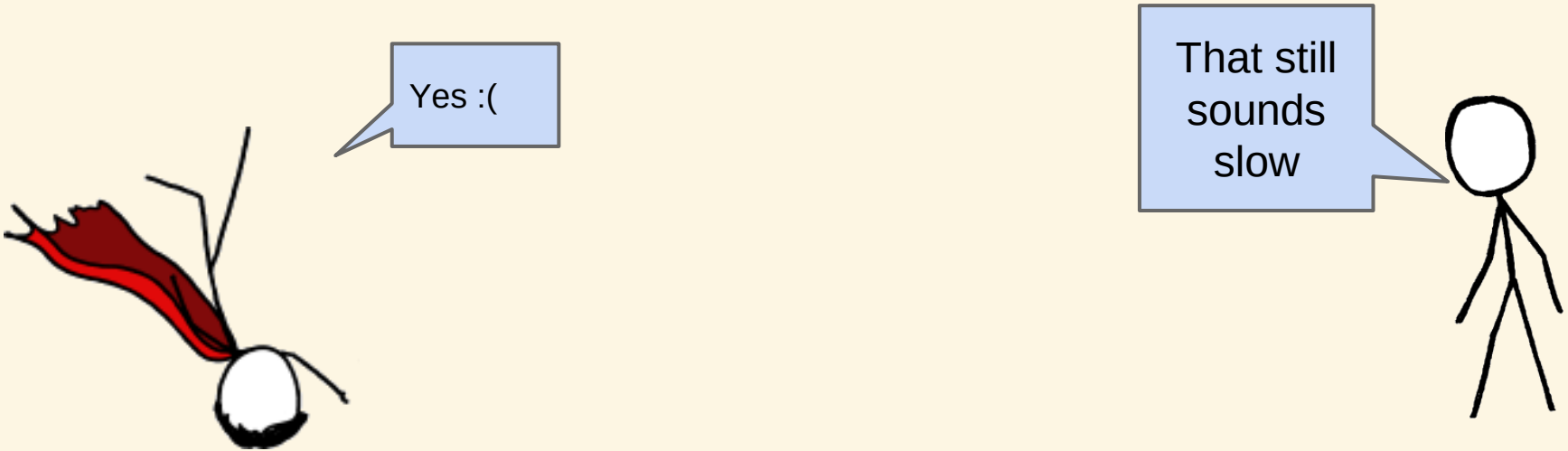


[1] *Graphite: A Distributed Parallel Simulator for Multicores* - Jason E. Miller et al.

Still not good enough

From the paper ***Graphite: a Distributed Parallel Simulator for Multicores***

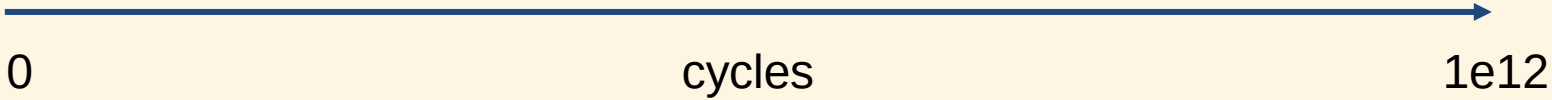
"Simulation slowdown is as low as 41× versus native execution"



[1] *Graphite: A Distributed Parallel Simulator for Multicores* - Jason E. Miller et al.

Question

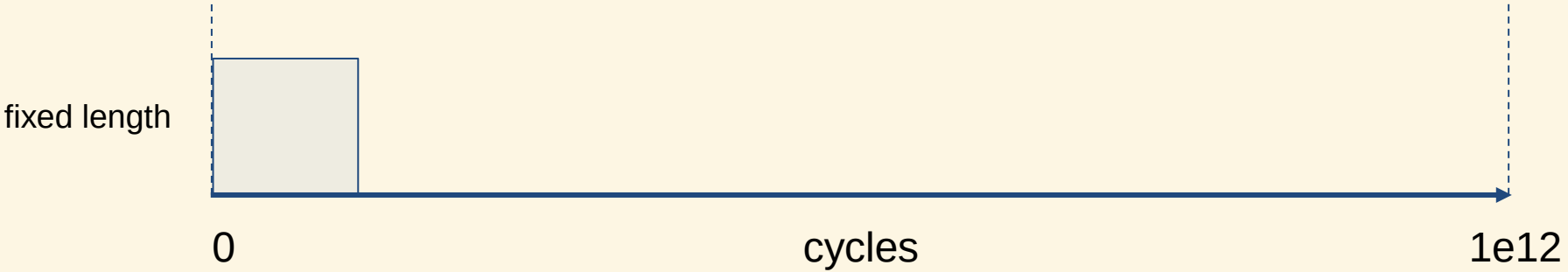
What can we do if it still takes weeks or months to simulate a full benchmark?



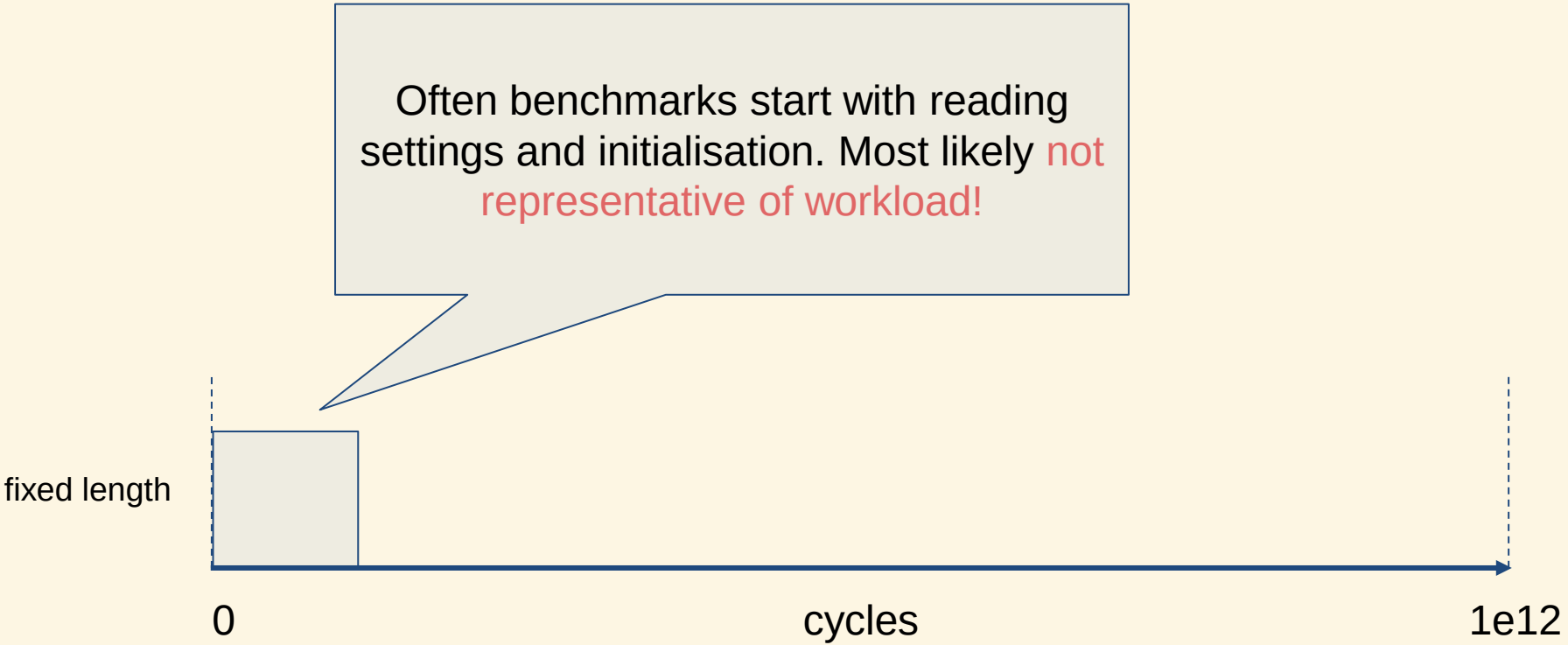
Workload Sampling

Naive Approach

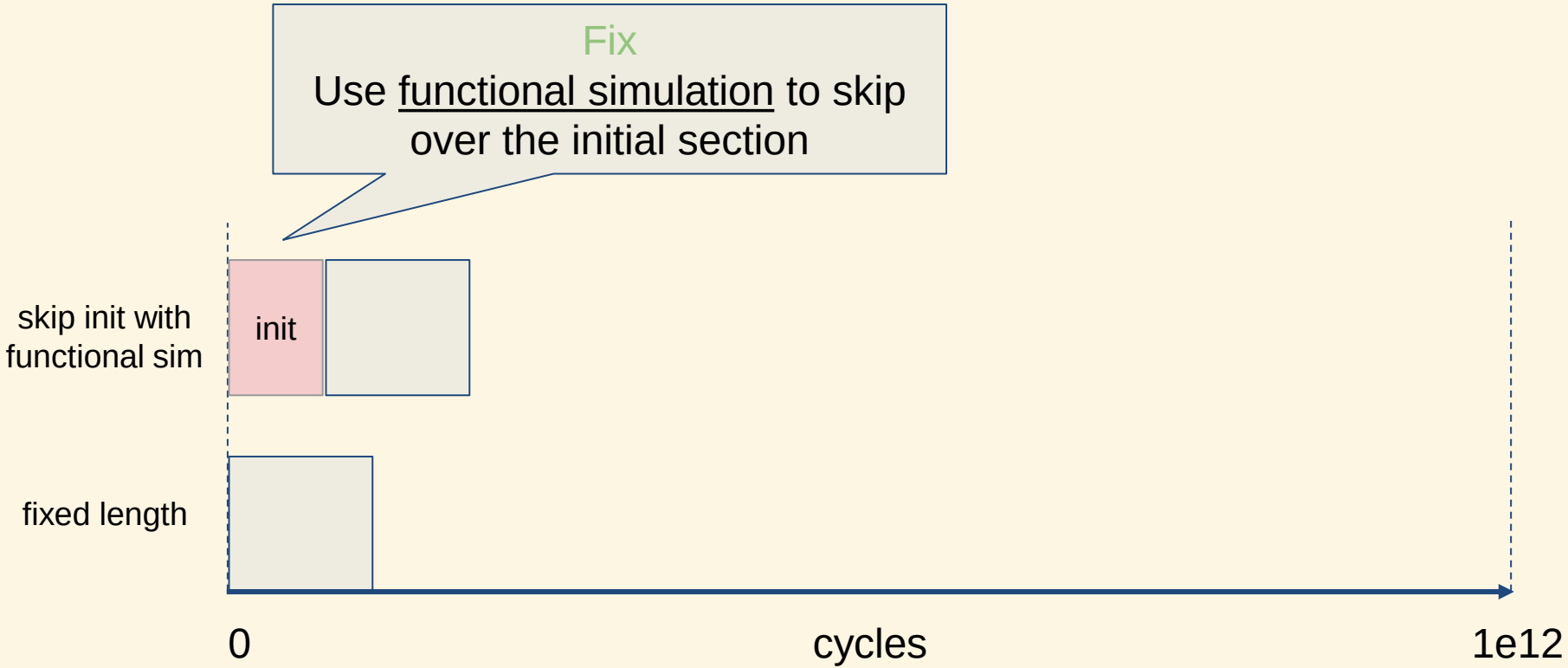
Only simulate first X cycles



Workload Sampling



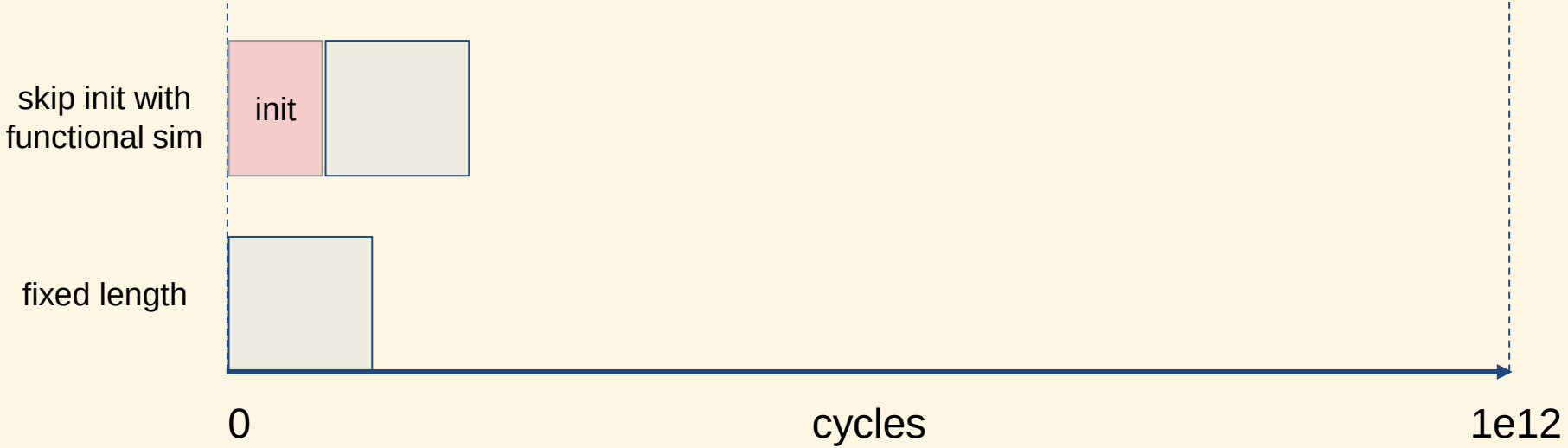
Workload Sampling



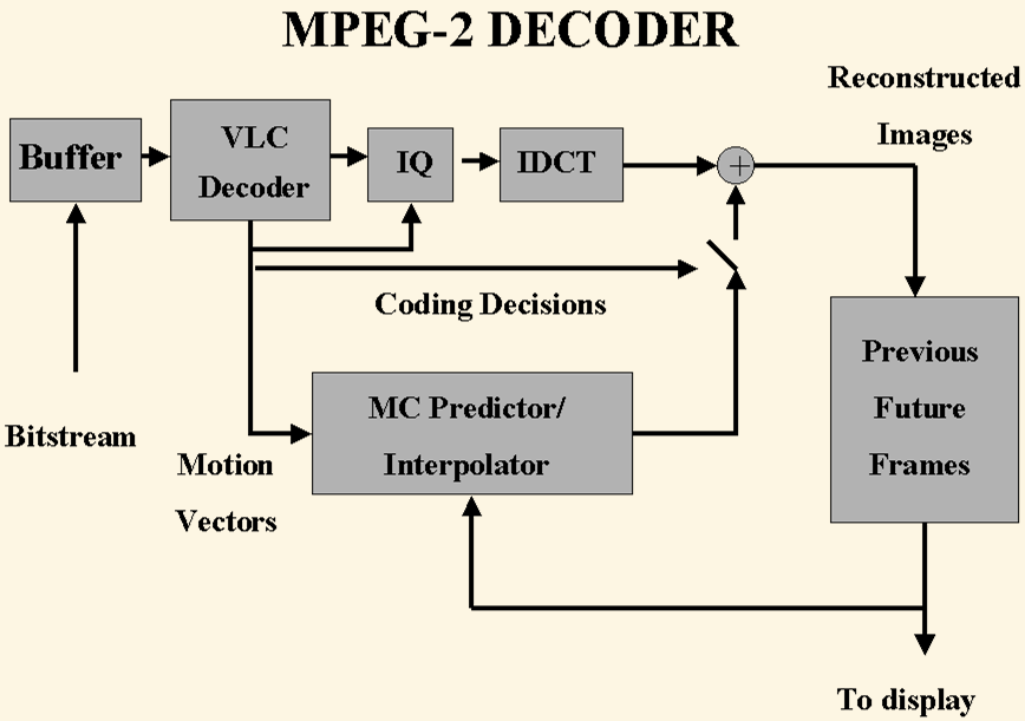
Workload Sampling

Question

Is the window always a good representation of the benchmark? Why/why not?



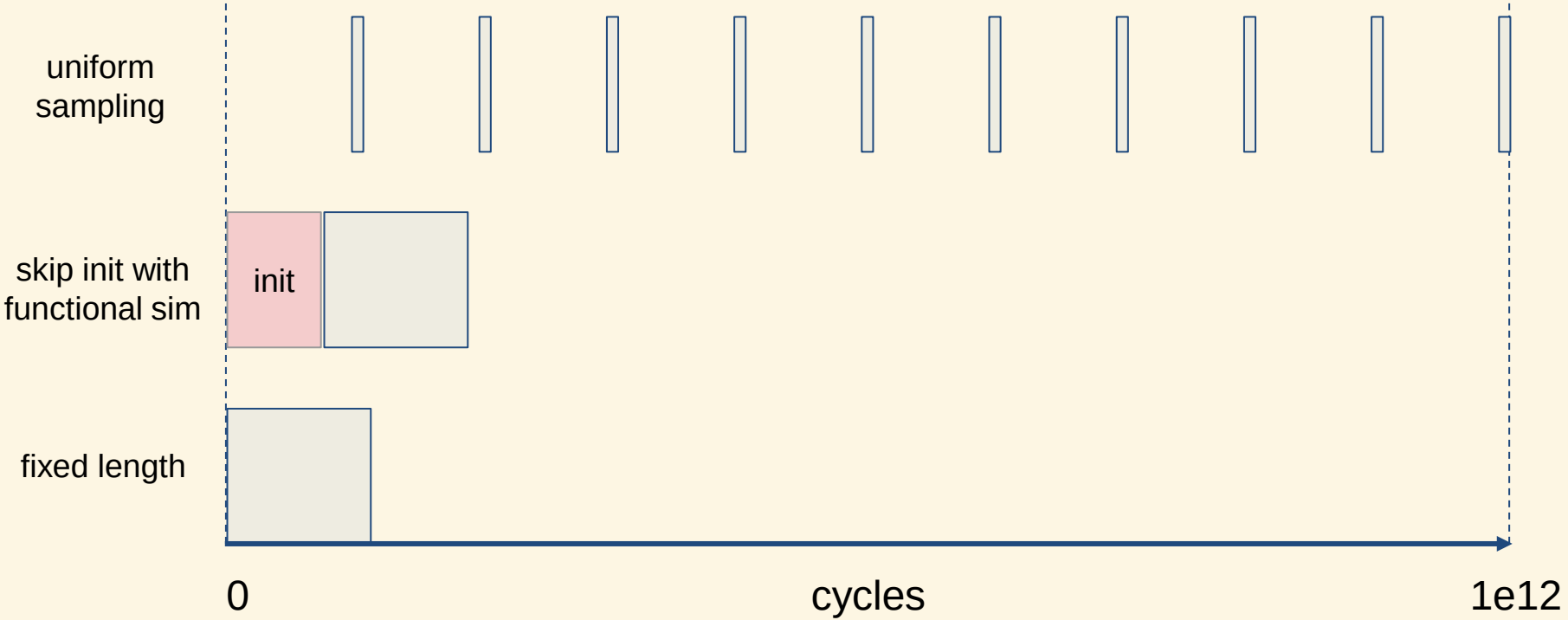
Program Modes



Real world programs spend time in different modes, which can have very different characteristics

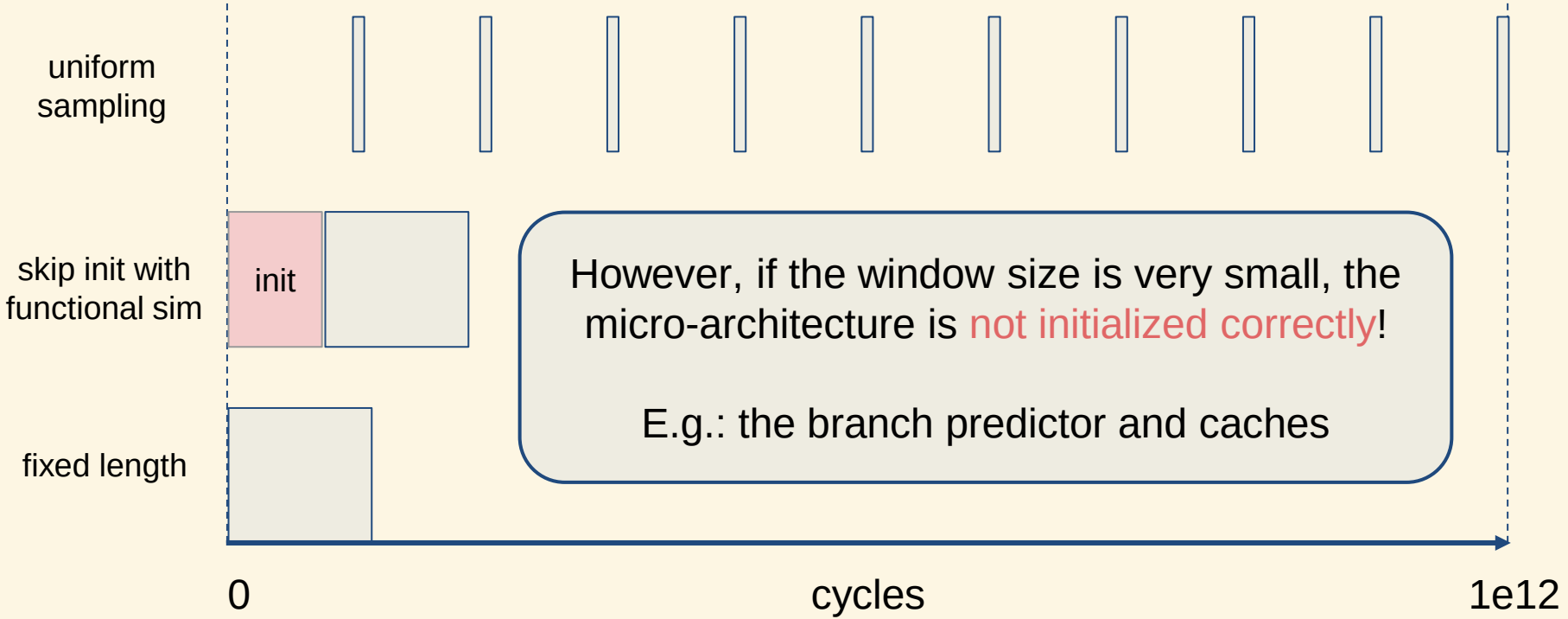
Workload Sampling

Sample uniformly over the program, hopefully capturing the dominant modes



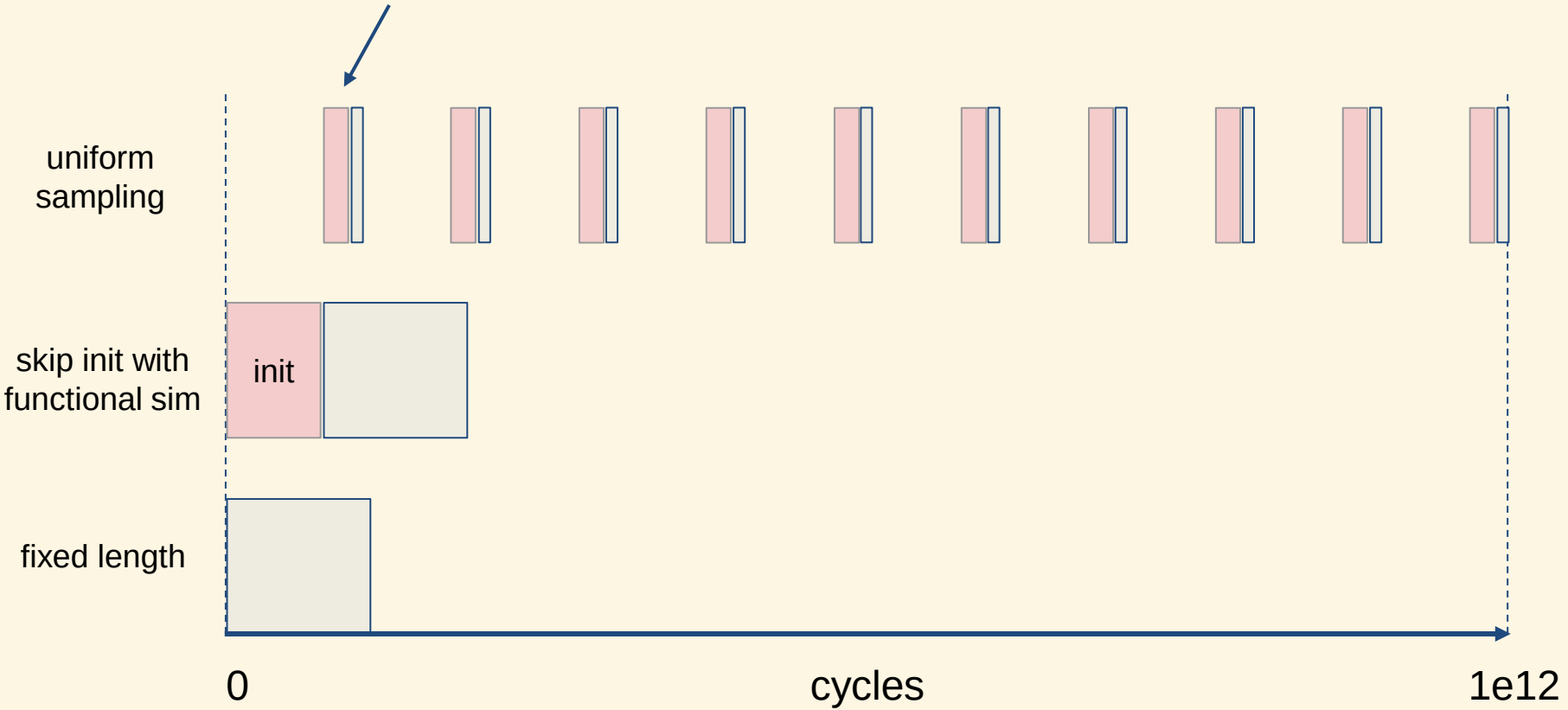
Workload Sampling

Sample uniformly over the program, hopefully capturing the dominant modes

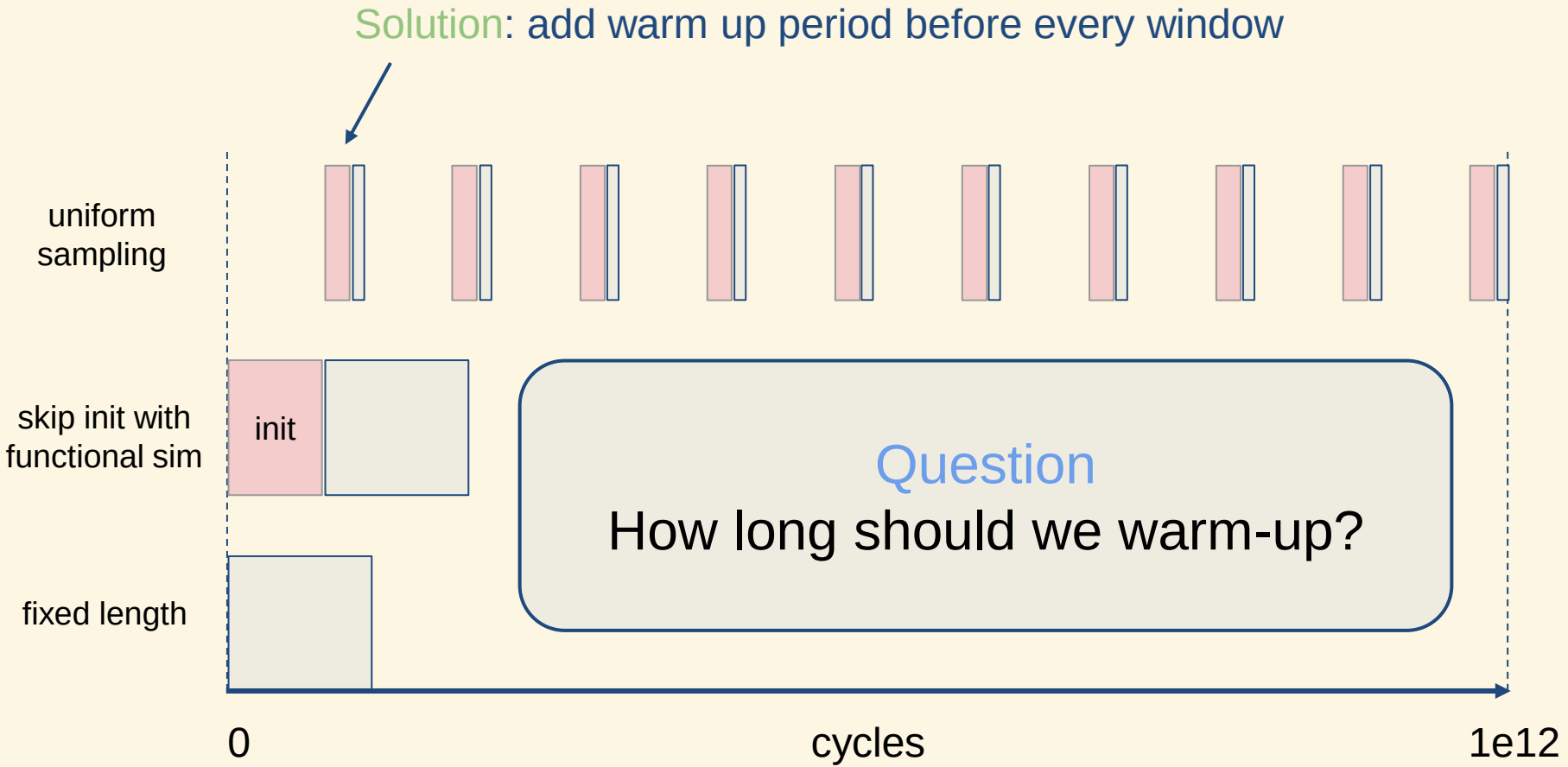


Workload Sampling

Solution: add warm up period before every window



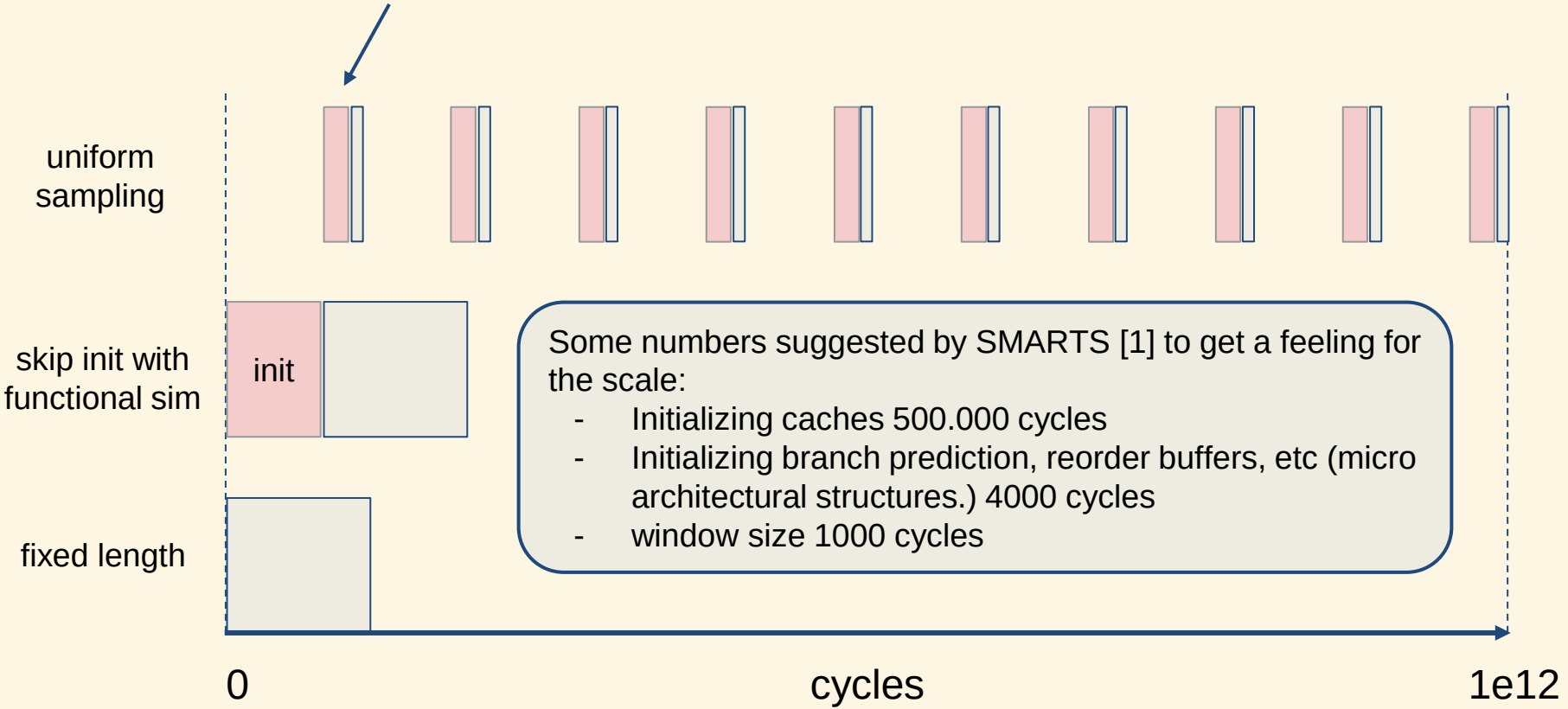
Workload Sampling



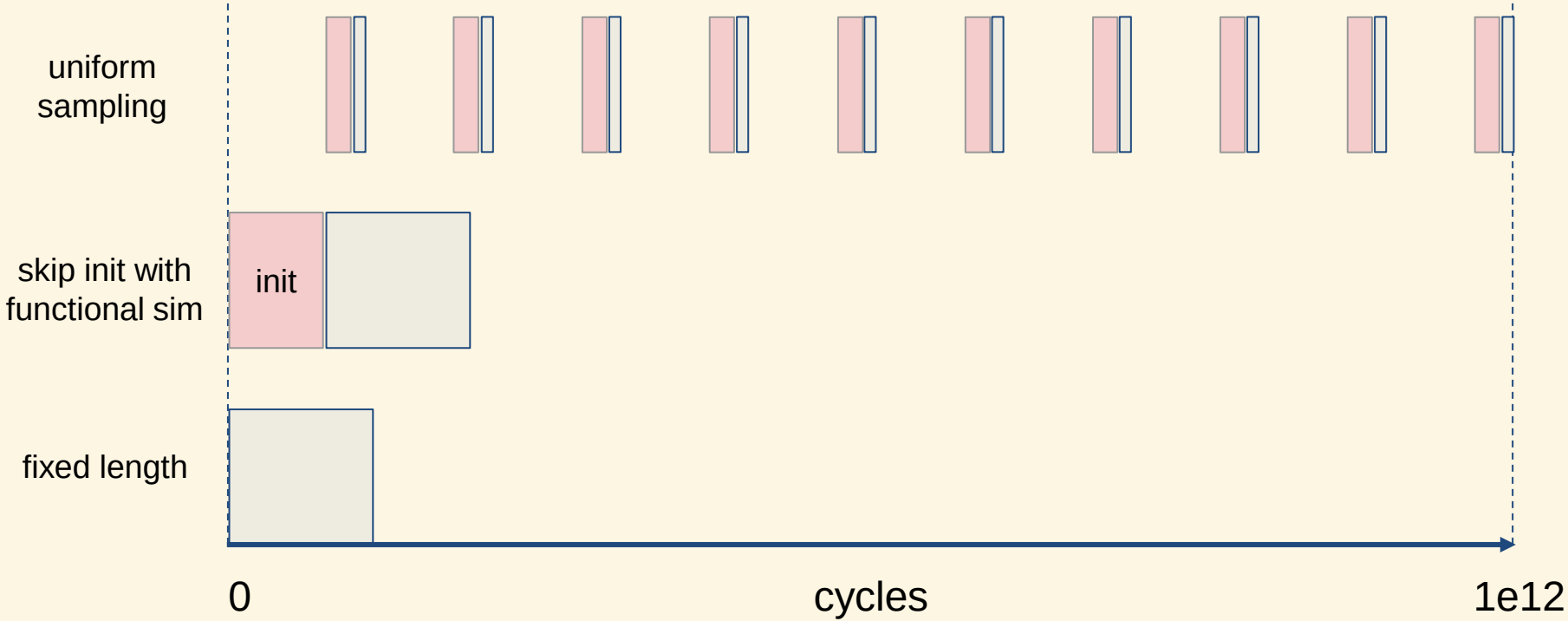
Workload Sampling

[1] *SMARTS: accelerating microarchitecture simulation via rigorous statistical sampling* - Roland E. Wunderlich et al.

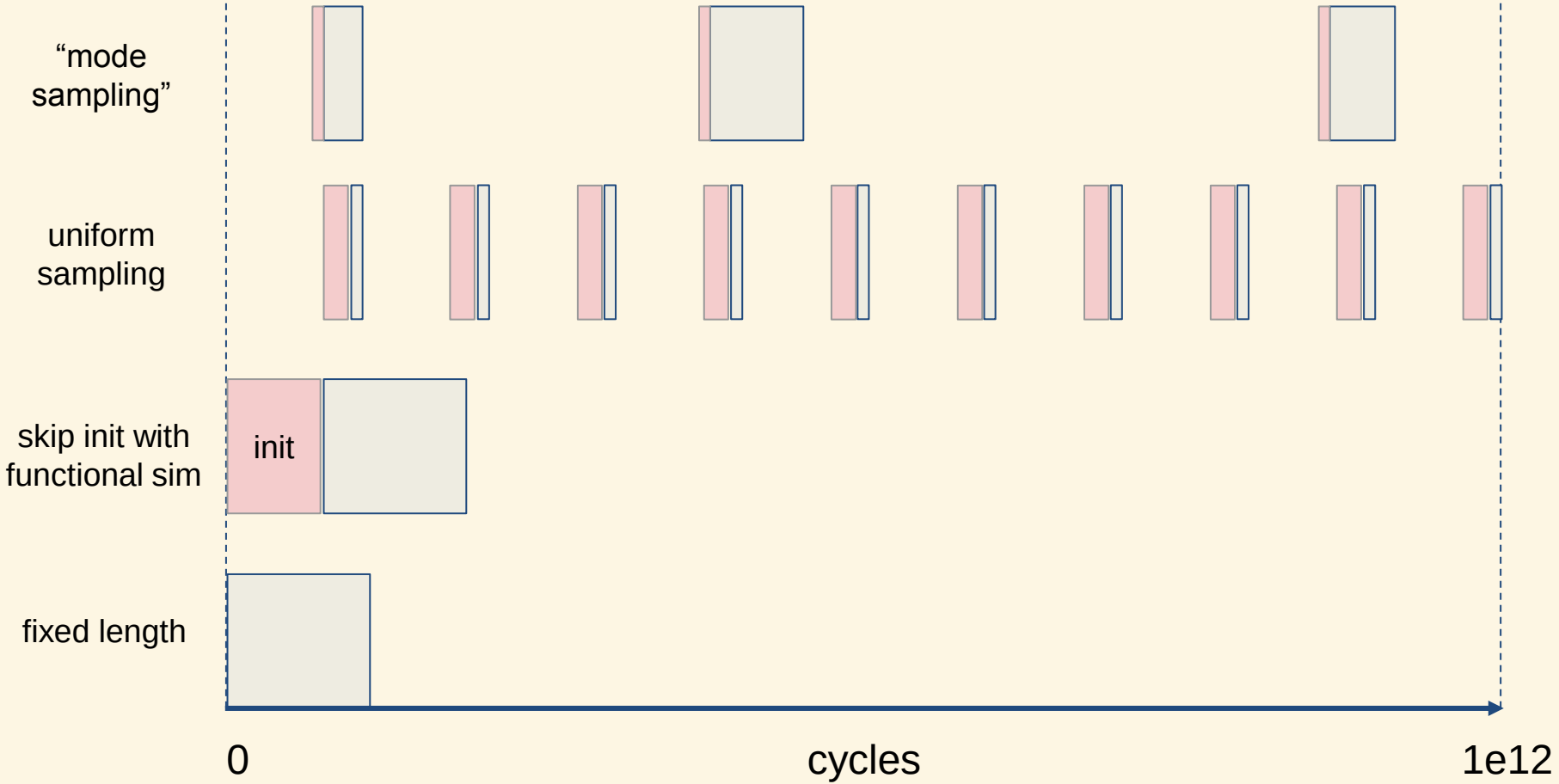
Solution: add warm up period before every window



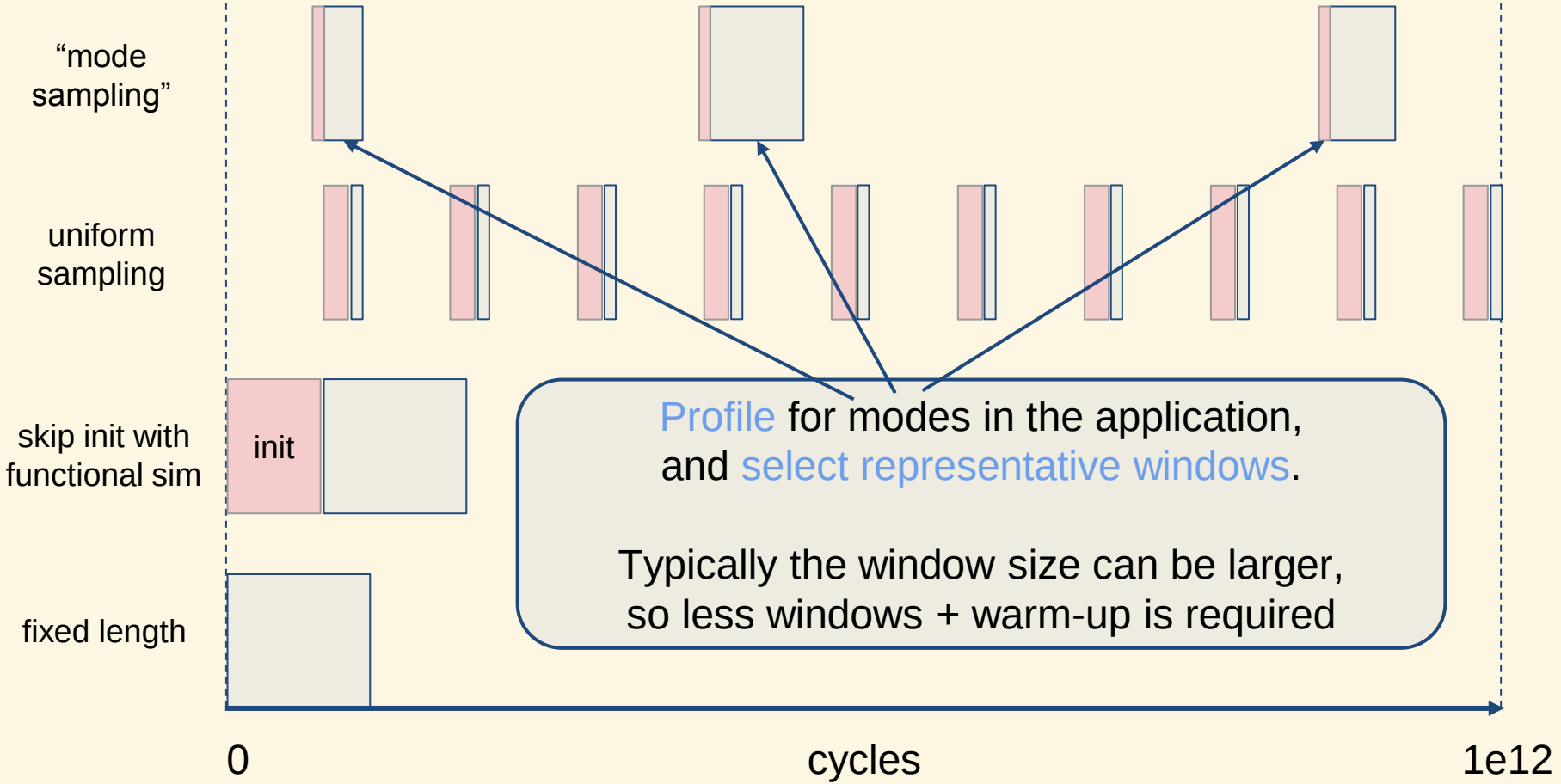
Workload Sampling



Workload Sampling



Workload Sampling



Summary

- Why Simulators
  - More accurate than models
  - Cheaper than building hardware
- Simulation detail
  - Full-System vs User-level
  - Functional vs Cycle Accurate (micro-arch.) vs Gate-Level
  - Execution- vs Trace-driven
- (Fast) Multiprocessor Simulation
  - Discrete event
  - Quantum
  - slack
- Workload Sampling
- Summary (the meta lecture)

Summary

- Why Simulators
  - More accurate than models
  - Cheaper than building hardware
- Simulation detail
  - Full-System vs User-level
  - Functional vs Cycle Accurate (micro-arch.) vs Gate-Level
  - Execution- vs Trace-driven
- (Fast) Multiprocessor Simulation
  - Discrete event
  - Quantum
  - slack
- Workload Sampling
- Summary (the meta lecture)

You can find more (optional) background information in chapter 9 of the book by Dubois et al.