dist-gem5: Distributed Simulation of Compute Clusters

Mohammad Alian, Umur Darbaz, Gabor Dozsa, Stephan Diestelhorst, Daehoon Kim, Nam Sung Kim

University of Illinois Urbana-Champaign

ARM Ltd., Cambridge, UK

Outline

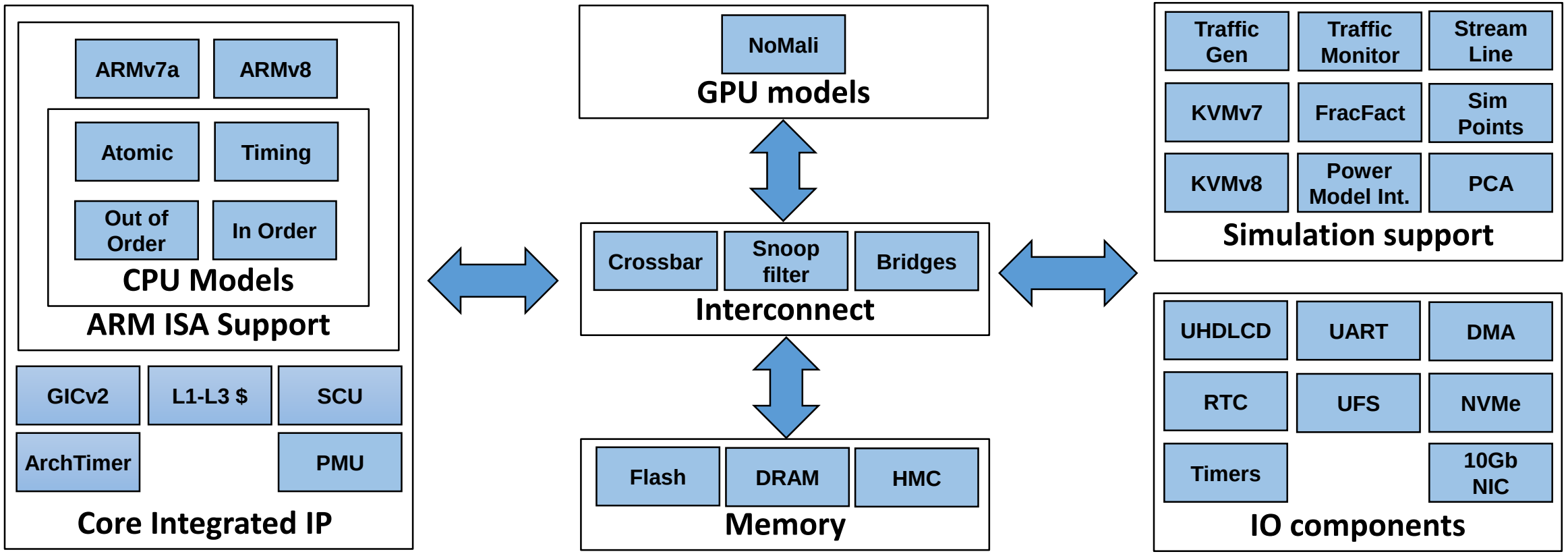
- **motivation**
  - ✓accelerating large-scale simulation
- **dist-gem5 architecture**
  - ✓packet forwarding
  - ✓synchronization
  - ✓checkpointing
  - ✓network model
- **evaluation**
  - ✓validation, speedup, synchronization overhead
- **conclusion**

Outline

- **motivation**
  - ✓accelerating large-scale simulation
- **dist-gem5 architecture**
  - ✓packet forwarding
  - ✓synchronization
  - ✓checkpointing
  - ✓network model
- **evaluation**
  - ✓validation, speedup, synchronization overhead
- **conclusion**

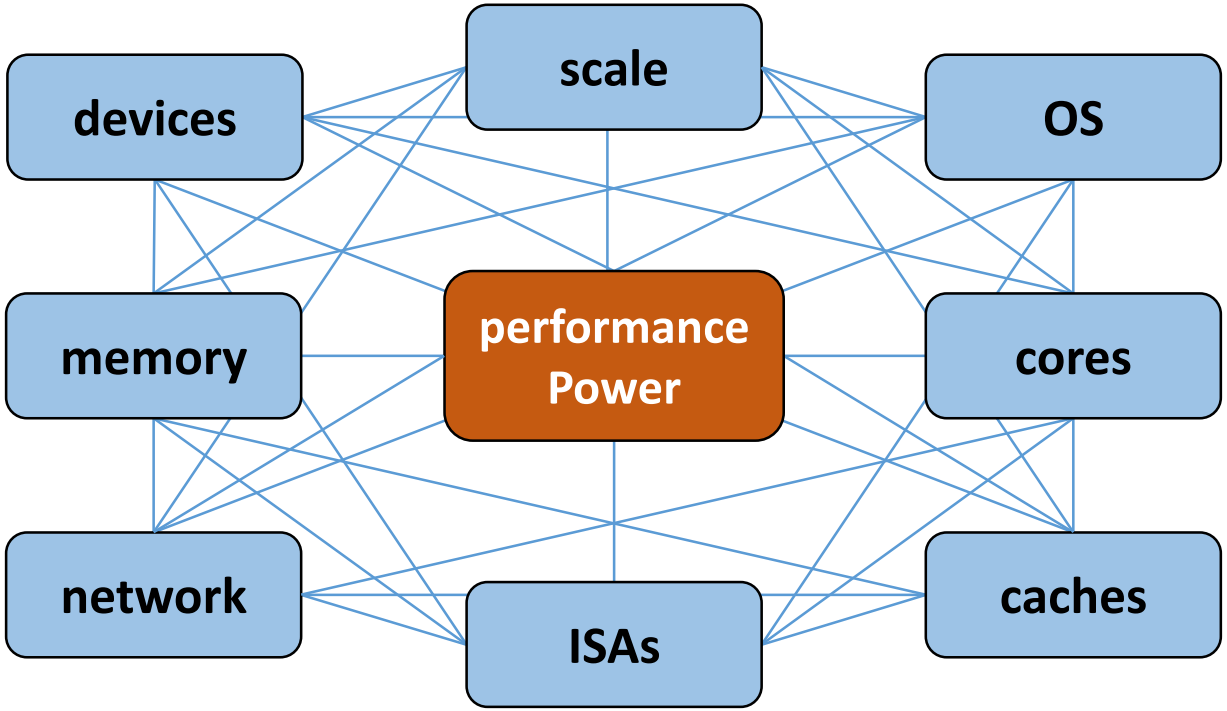
What is gem5 – overview

- full-system, cycle-level, event-driven simulator
- used/maintained at universities and industry



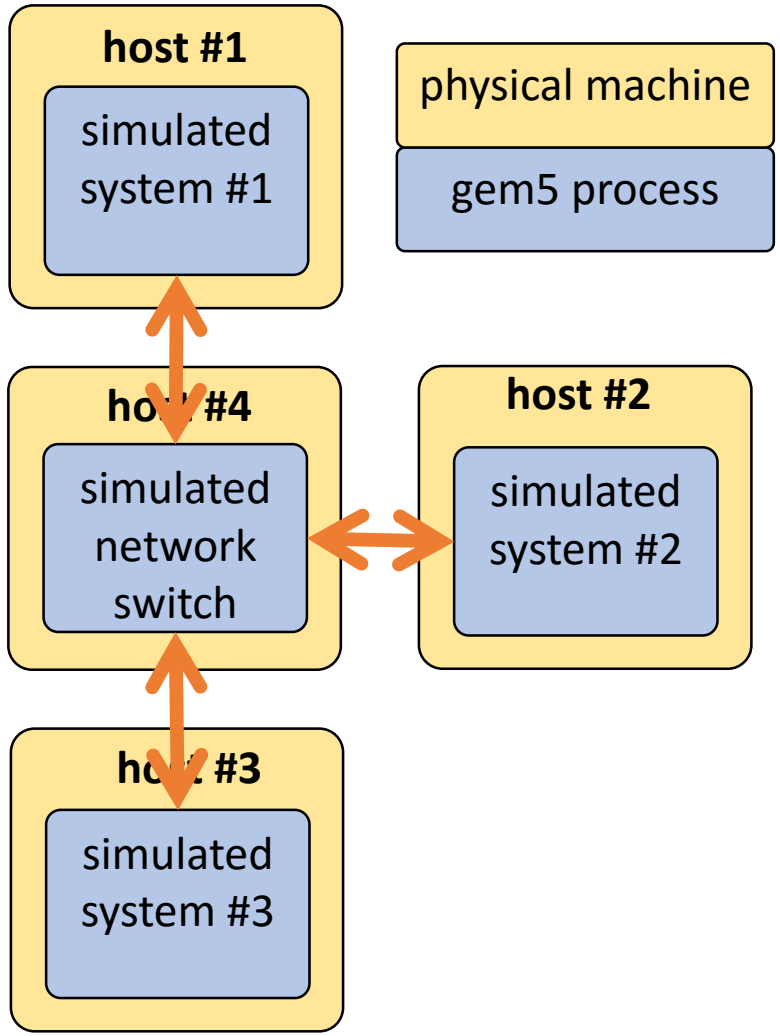
Why dist-gem5?

- performance and power dissipation of a distributed system
  - ✓complex interplay among system components at scale
- need a full-system, cycle-level simulator which is fast enough to simulate a large-scale computer system
- distributed simulation:
  - ✓simulate a distributed system w/ many simulation hosts



dist-gem5 architecture – high level view

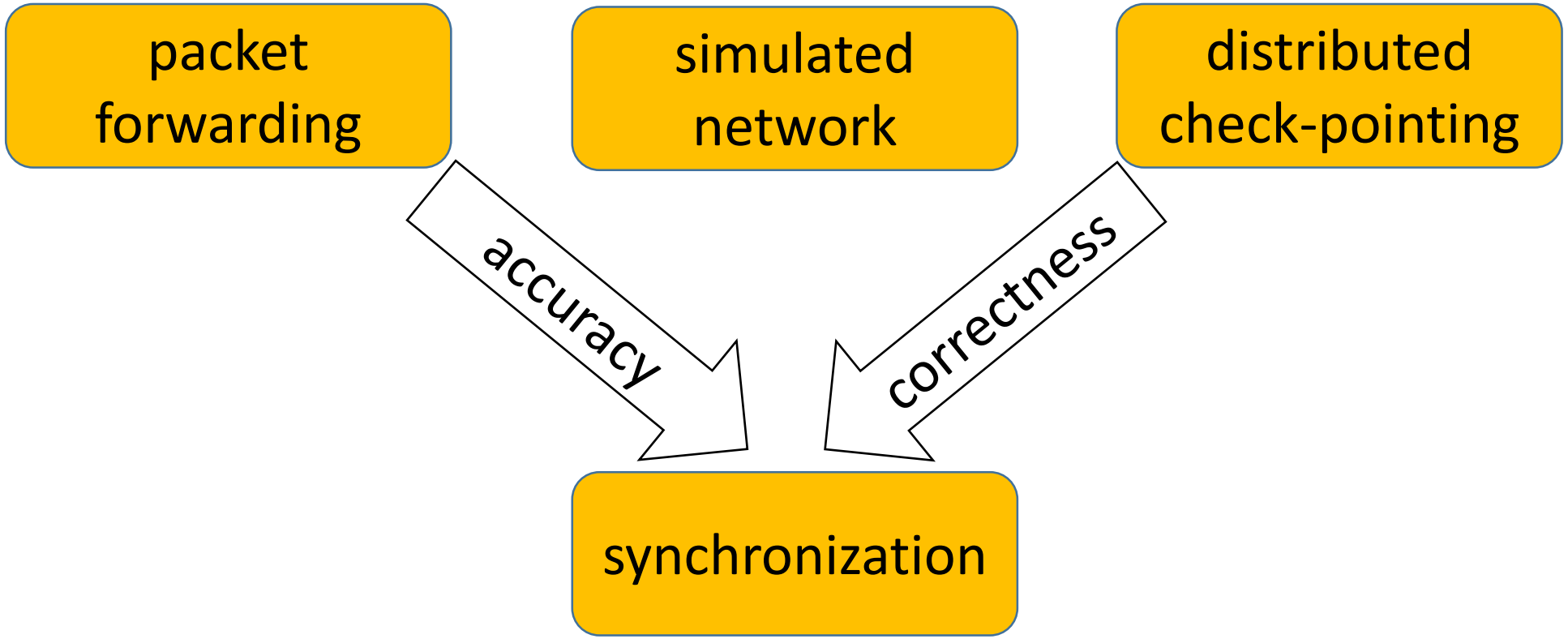
- gem5 processes modeling full systems run in parallel on a cluster of physical machines
- simulated network switch
  - ✓forward packets among the simulated systems
  - ✓synchronize the distributed simulation
  - ✓simulate network topology



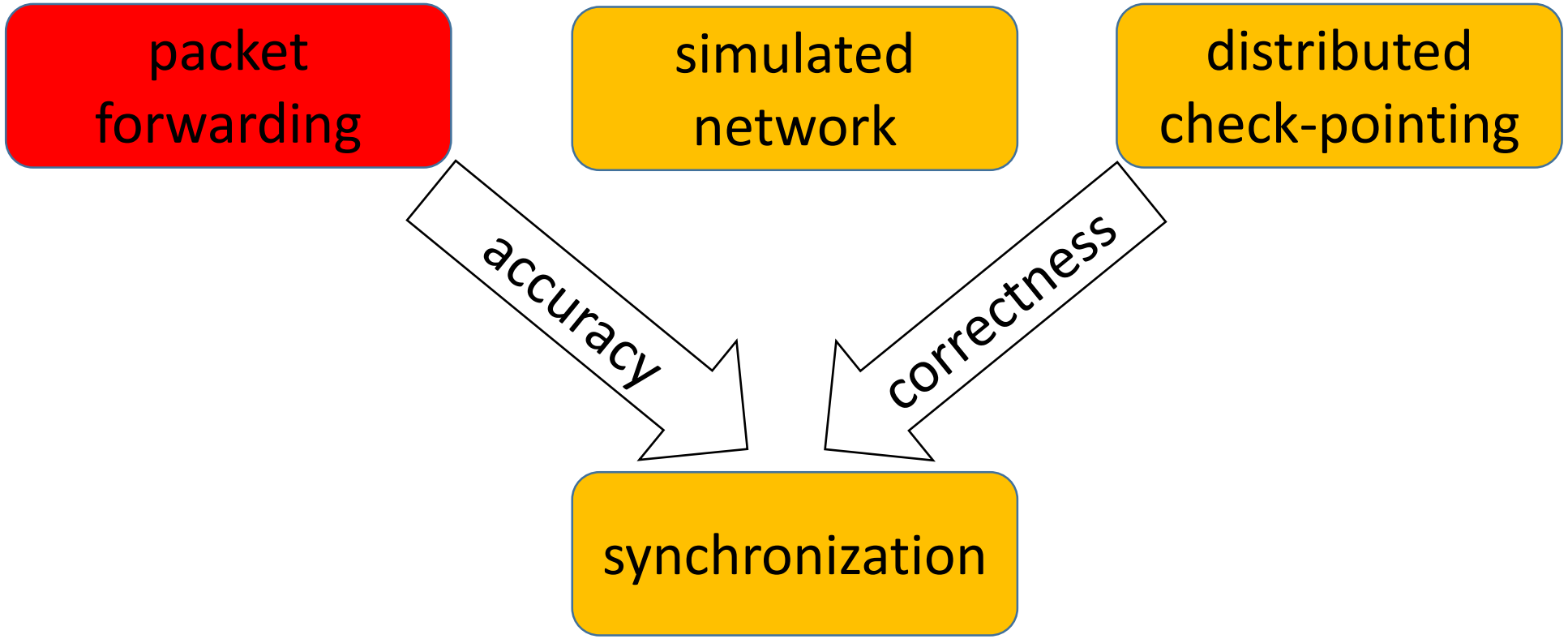
Outline

- motivation
  - ✓accelerating large-scale simulation
- **dist-gem5 architecture**
  - ✓packet forwarding
  - ✓synchronization
  - ✓checkpointing
  - ✓network model
- evaluation
  - ✓validation, speedup, synchronization overhead
- conclusion

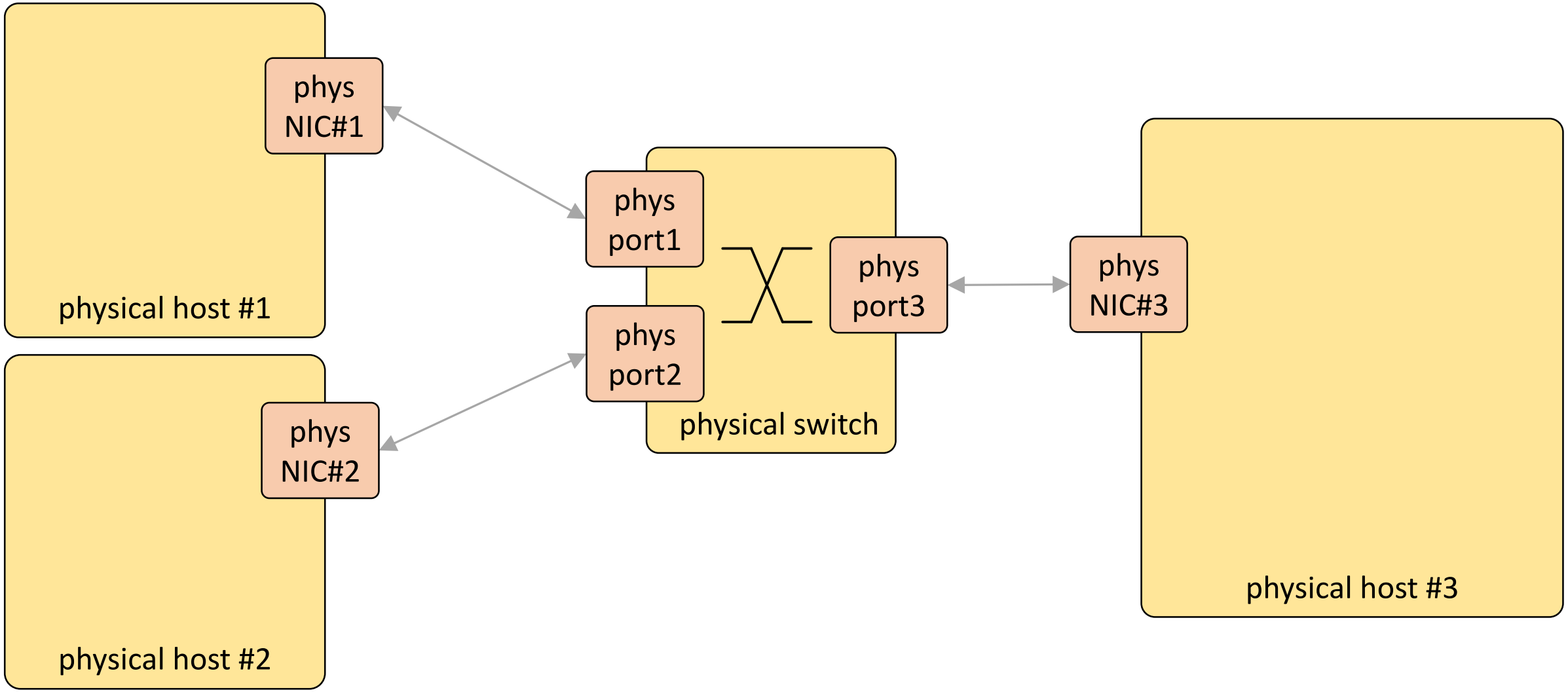
dist-gem5 architecture – core components



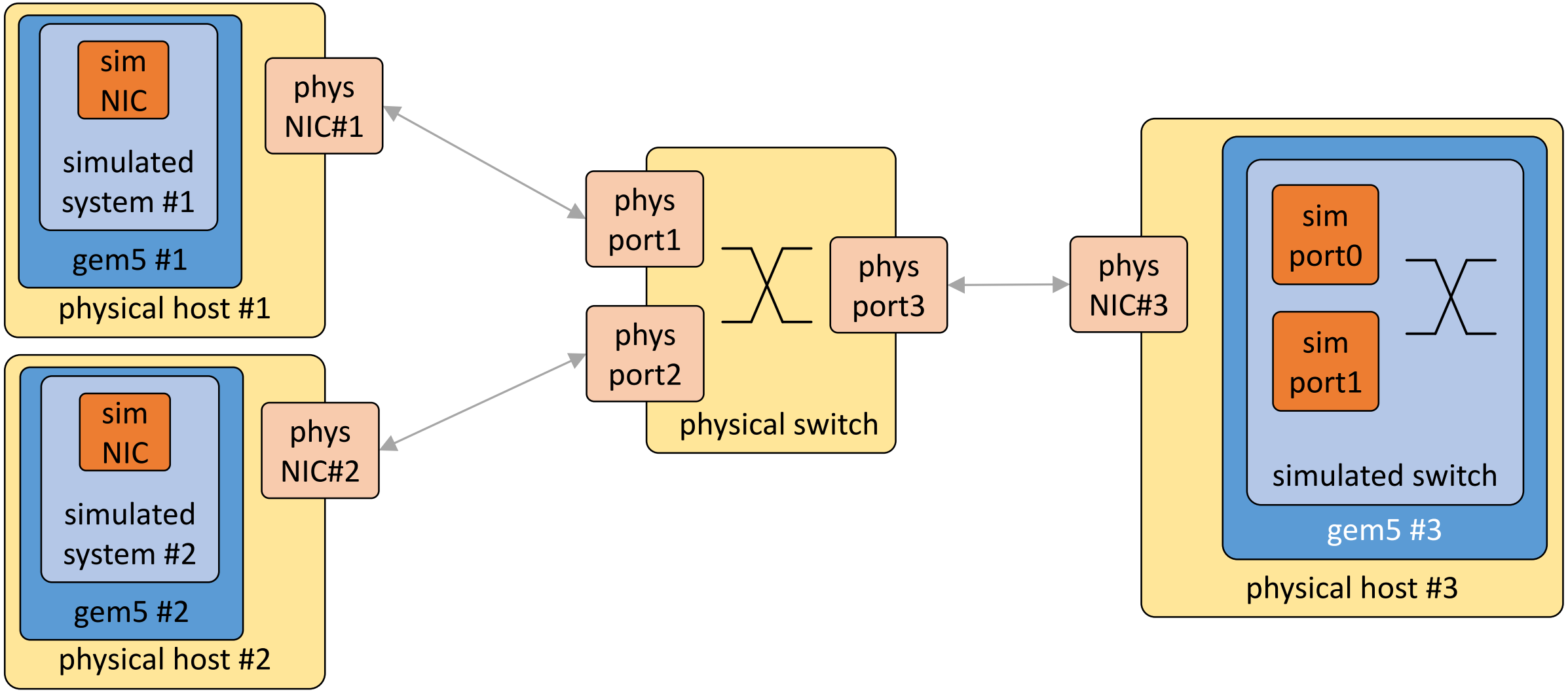
dist-gem5 architecture – core components



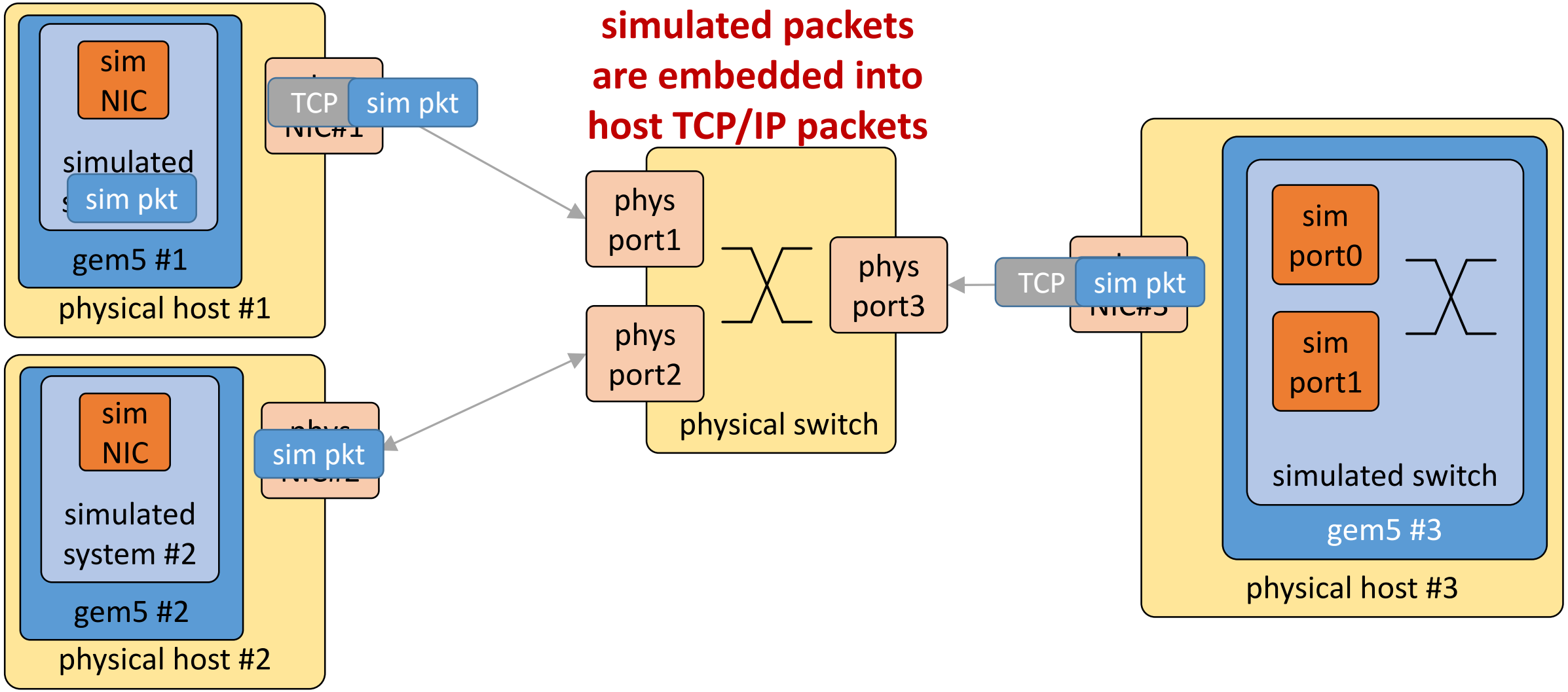
dist-gem5 architecture – packet forwarding



dist-gem5 architecture – packet forwarding

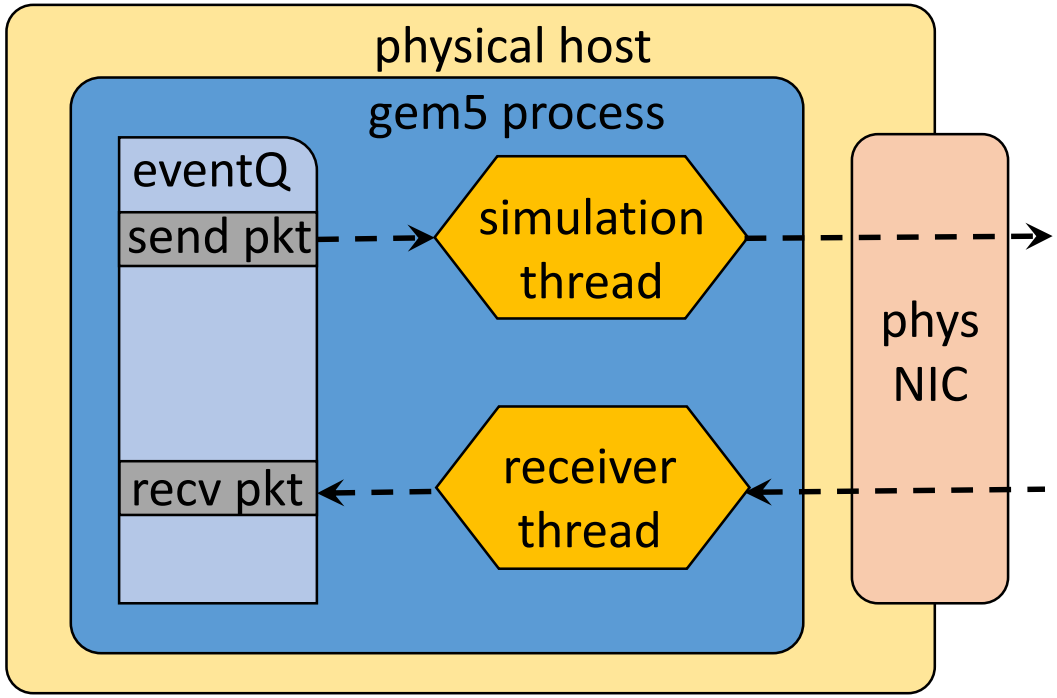


dist-gem5 architecture – packet forwarding

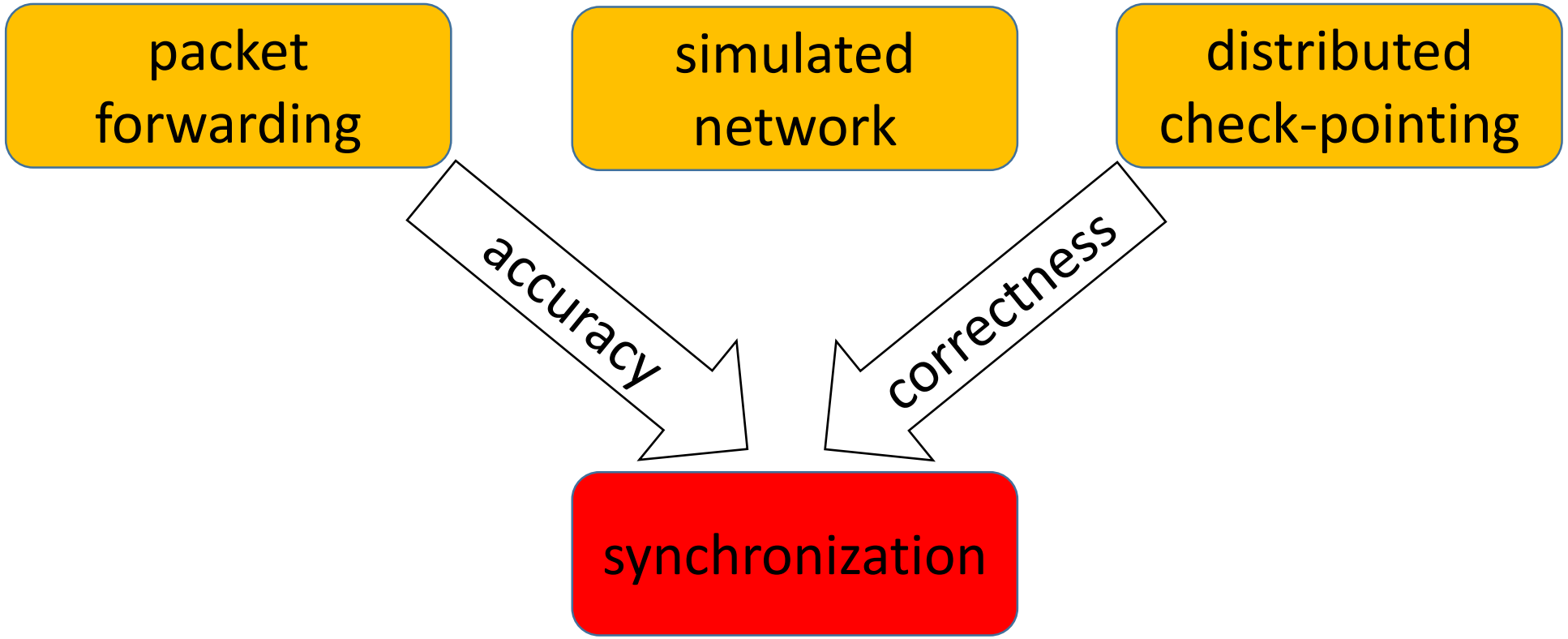


Asynchronous processing of incoming messages

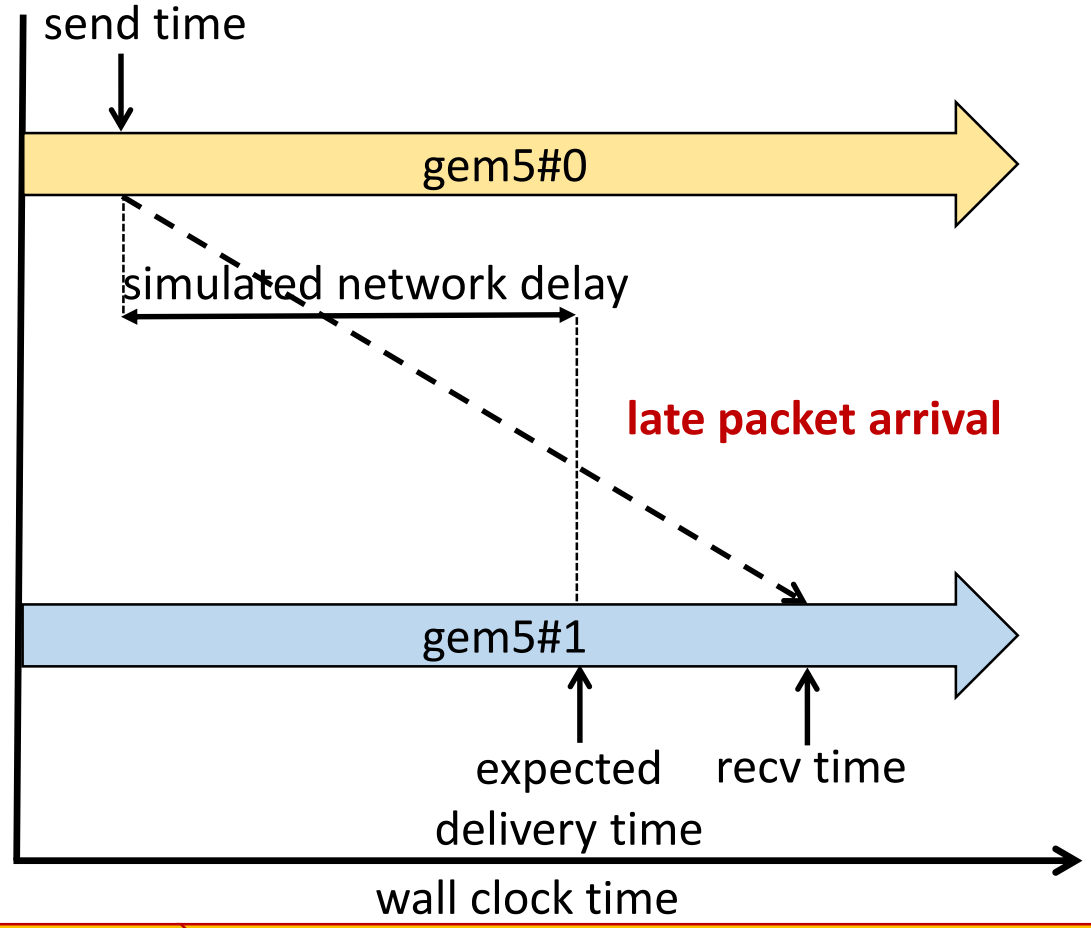
- simulation thread (main thread)
  - ✓process/insert events in the event queue
  - ✓in case of send pkt event, encapsulate the simulated Ethernet packet in a message and send it out
- receiver thread
  - ✓create for each gem5 process
  - ✓waits for incoming packets
  - ✓creates a recv pkt event and insert it to the event queue



dist-gem5 architecture – core components

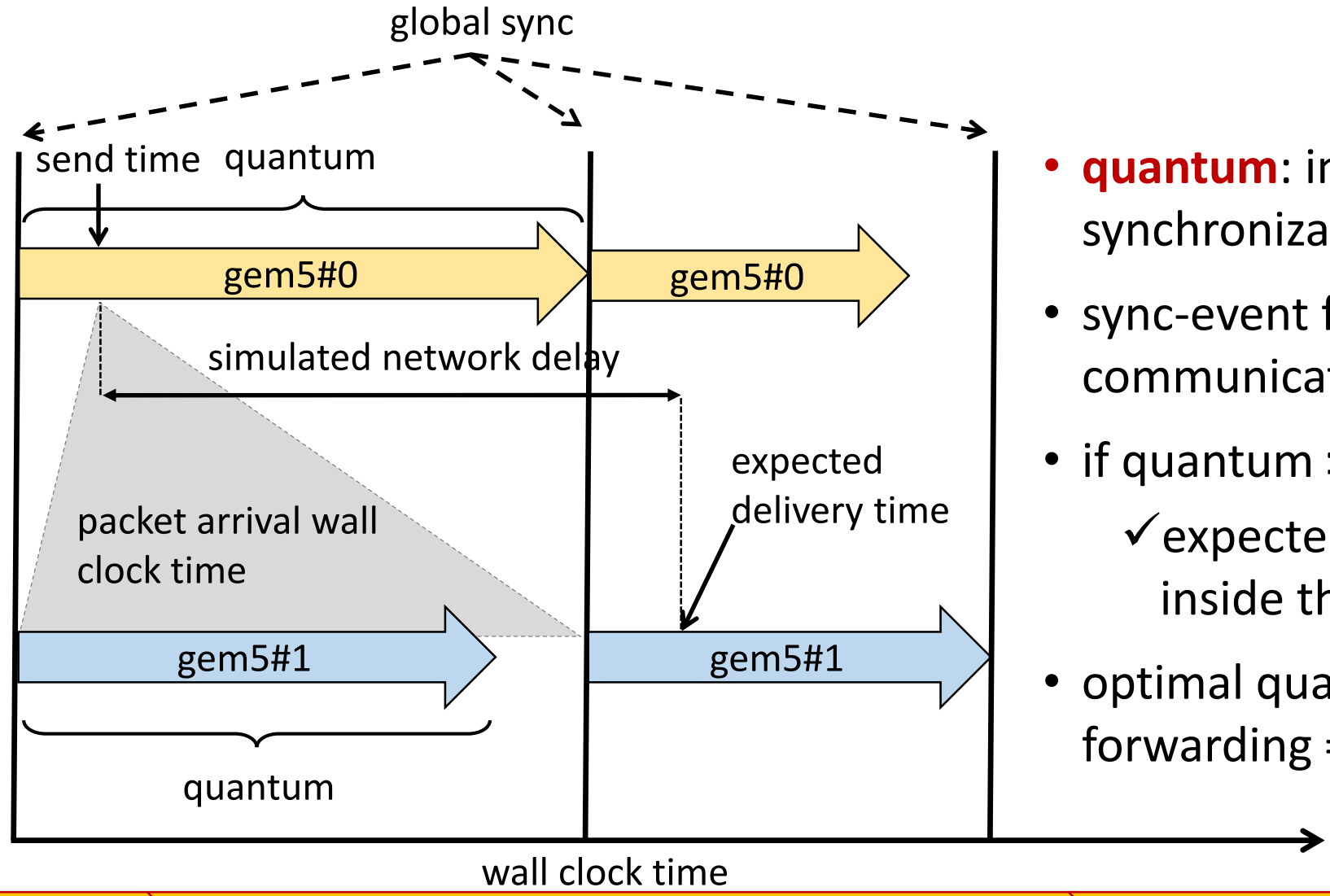


Need for synchronization



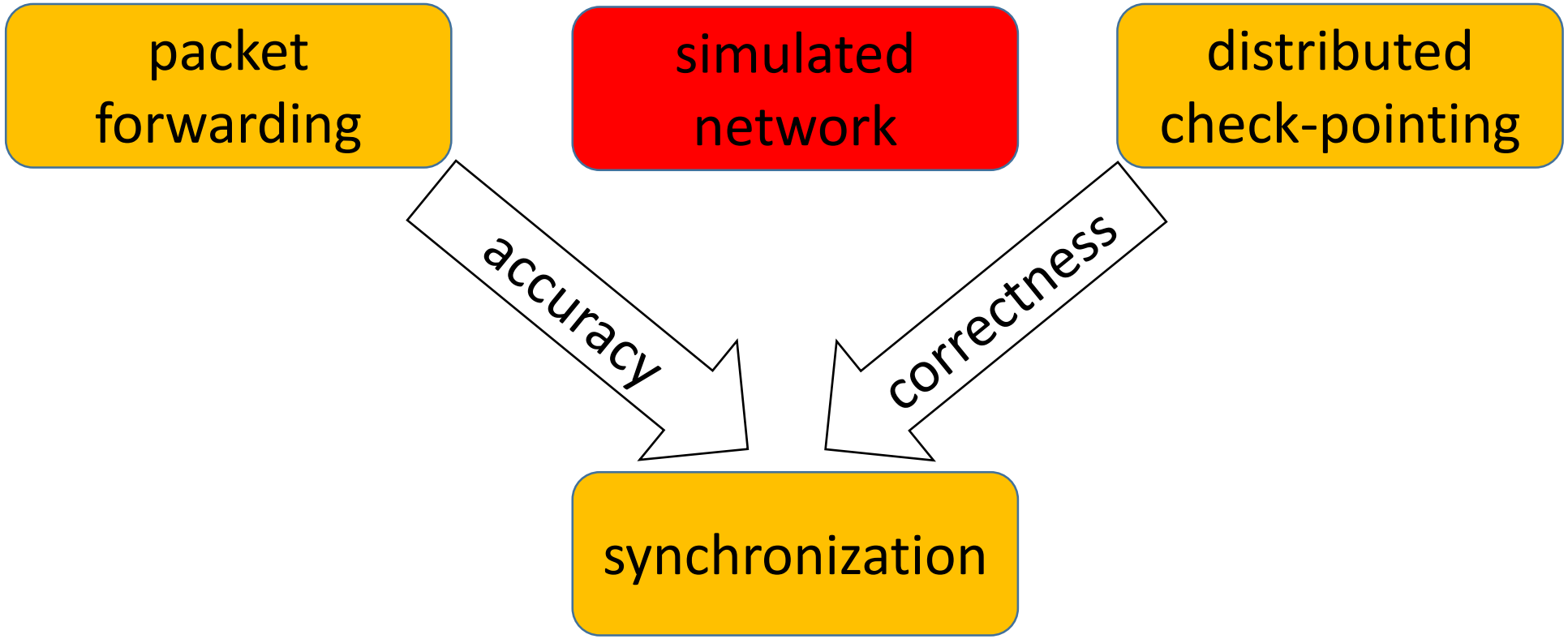
- receiver gem5 can run ahead of sender gem5
  - ✓physical host mismatch
  - ✓different events to be processed
- slowed down receiver gem5 to ensure simulation accuracy
- quantum-based synchronization

Accurate packet forwarding

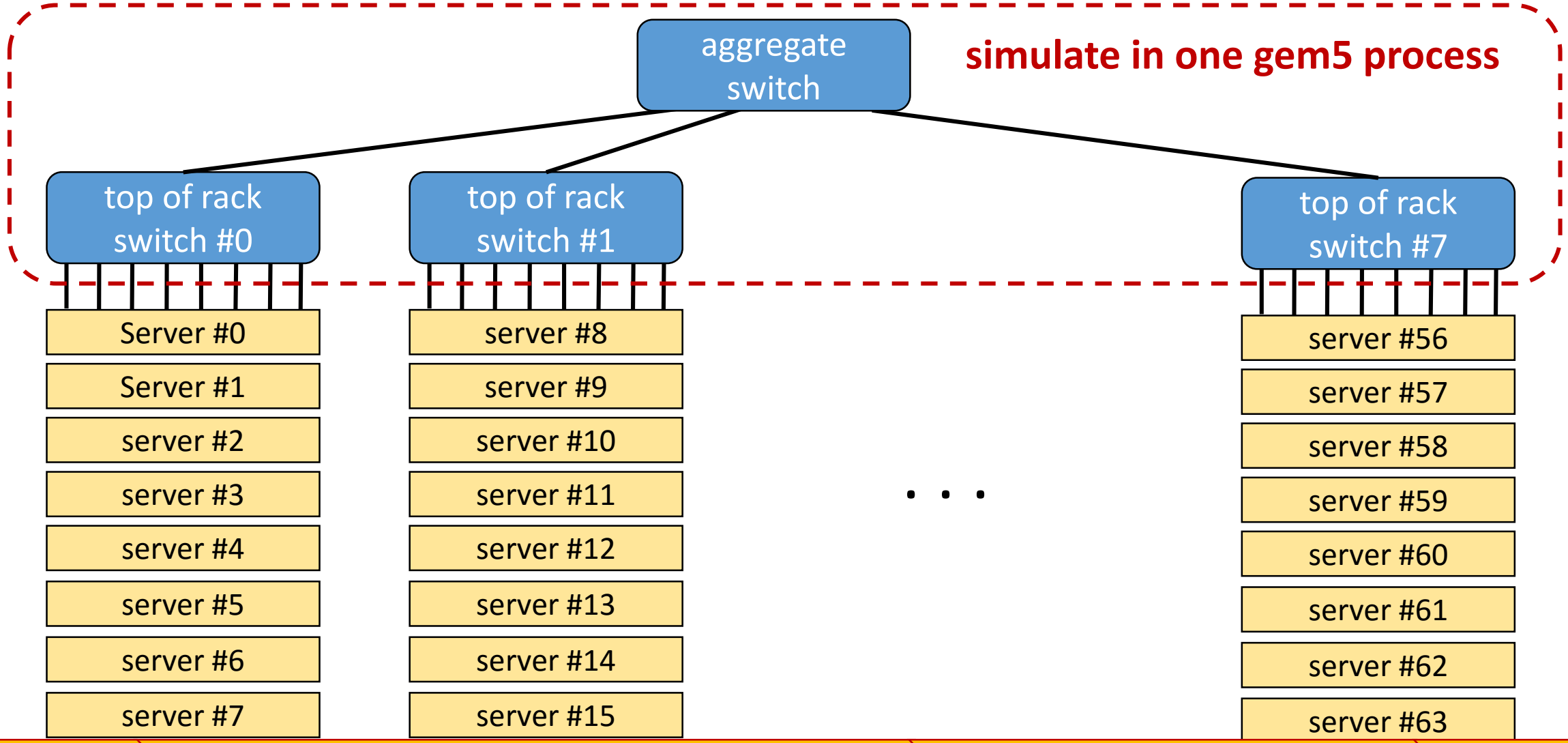


- **quantum**: interval for periodic synchronization in simulated time
- sync-event flushes inter gem5 communication channels
- if quantum ≤ simulated link delay:
  - ✓expected delivery tick falls inside the next quantum
- optimal quantum size for accurate forwarding == **simulated link delay**

dist-gem5 architecture – core components

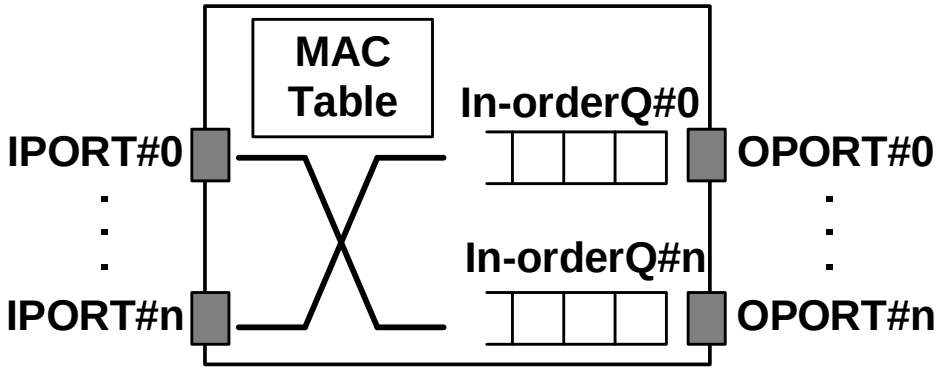


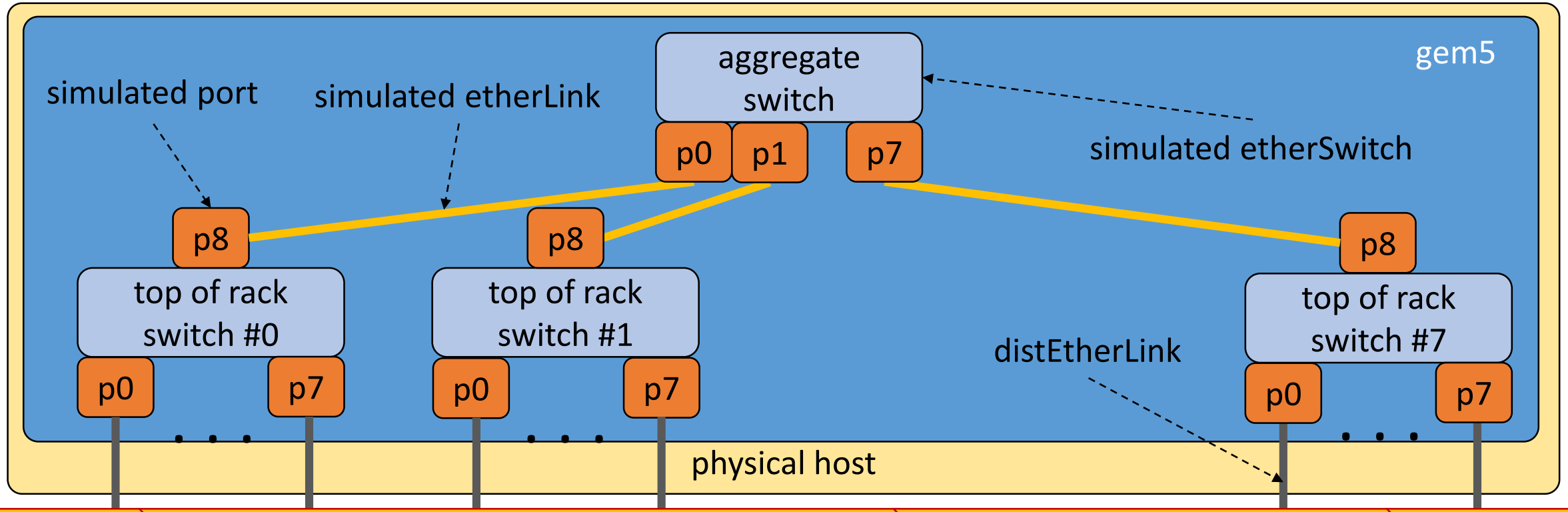
dist-gem5 architecture – network modeling



Configurable network model

- configurable baseline Ethernet switch model
  - ✓port number, delay, bandwidth, buffer size





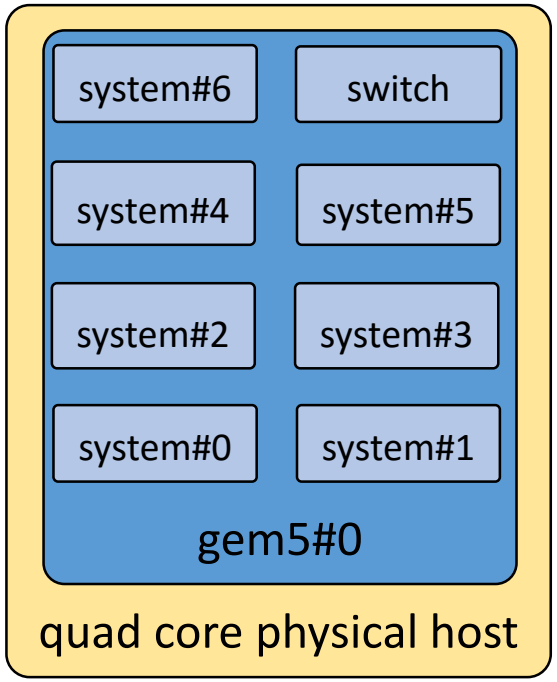
Outline

- **motivation**
  - ✓accelerating large-scale simulation
- **dist-gem5 architecture**
  - ✓packet forwarding
  - ✓synchronization
  - ✓checkpointing
  - ✓network model
- **evaluation**
  - ✓validation, speedup, synchronization overhead
- **conclusion**

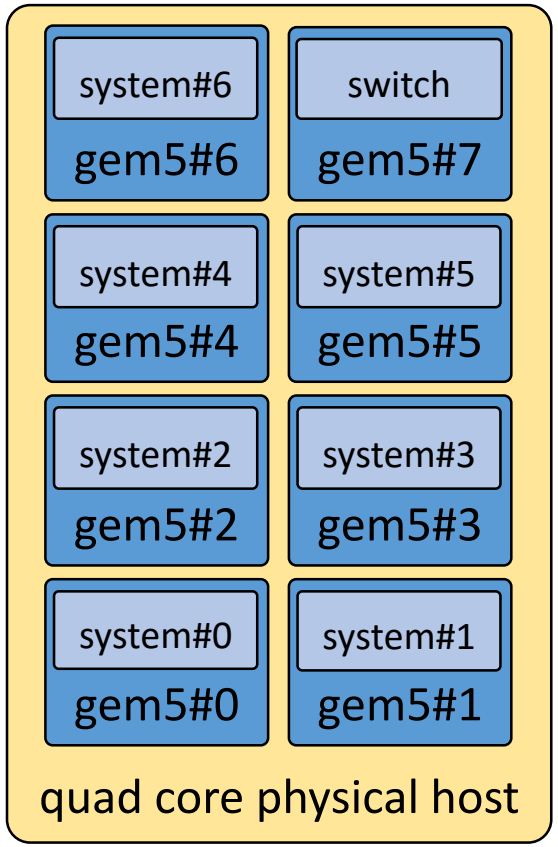
Methodology – simulation techniques

- For example, simulating a cluster w/ 7 nodes and 1 network switch:

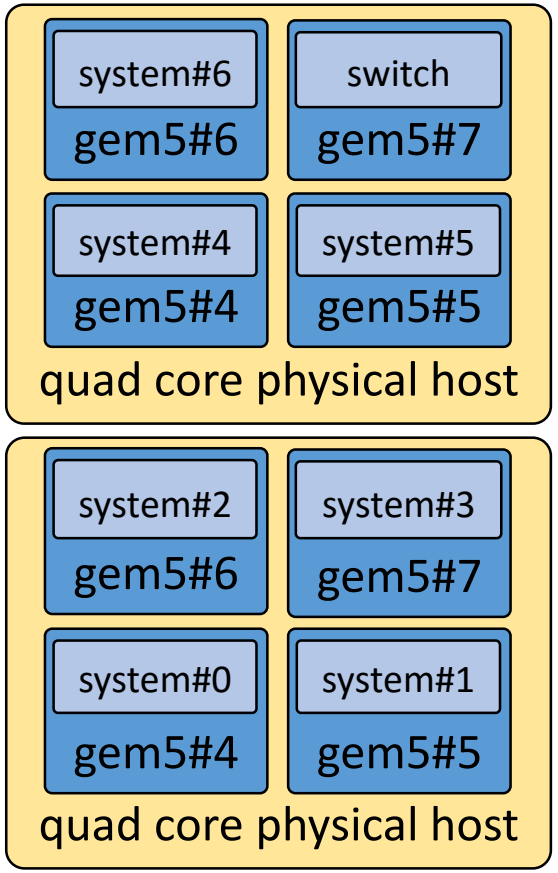
single-threaded-gem5



parallel-gem5



dist-gem5



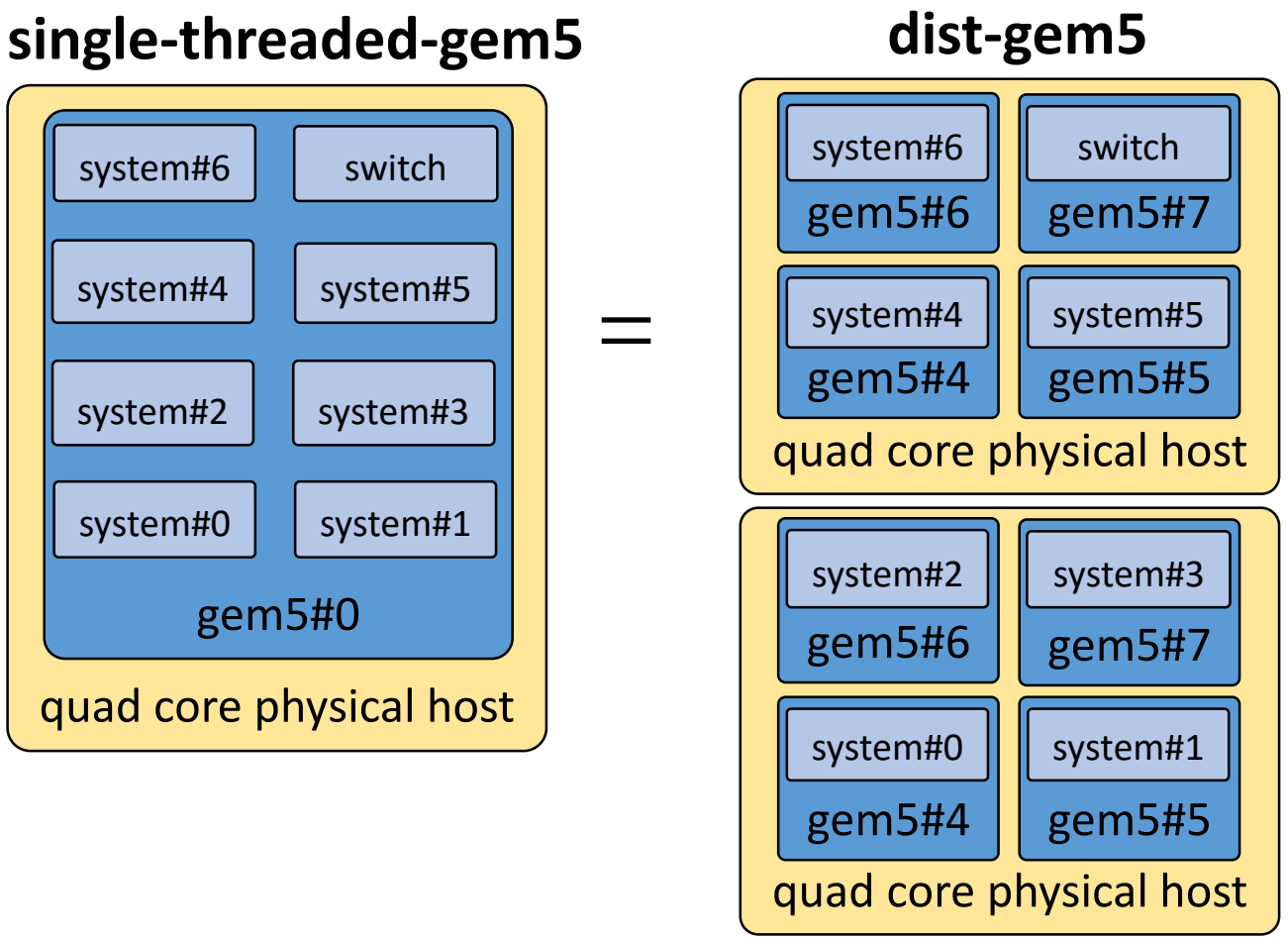
Methodology – experimental setup

- focus on off-chip network performance using network intensive applications
  - ✓iperf, memcached, httperf, tcptest, netperf, NAS parallel benchmark
- verification/validation against:
  - ✓single-threaded-gem5
  - ✓physical cluster
    - 4 node cluster w/ AMD A10-5800K
- speedup comparison against:
  - ✓single-threaded-gem5
  - ✓parallel-gem5

category	gem5 configuration
O3 core	4 cores; 4 way superscalar
memory	8GB DDR3 1600 MHz
network	Intel GbE NIC; 1 µs Link latency
OS	Linux Ubuntu 14.04 (Kernel 4.3)

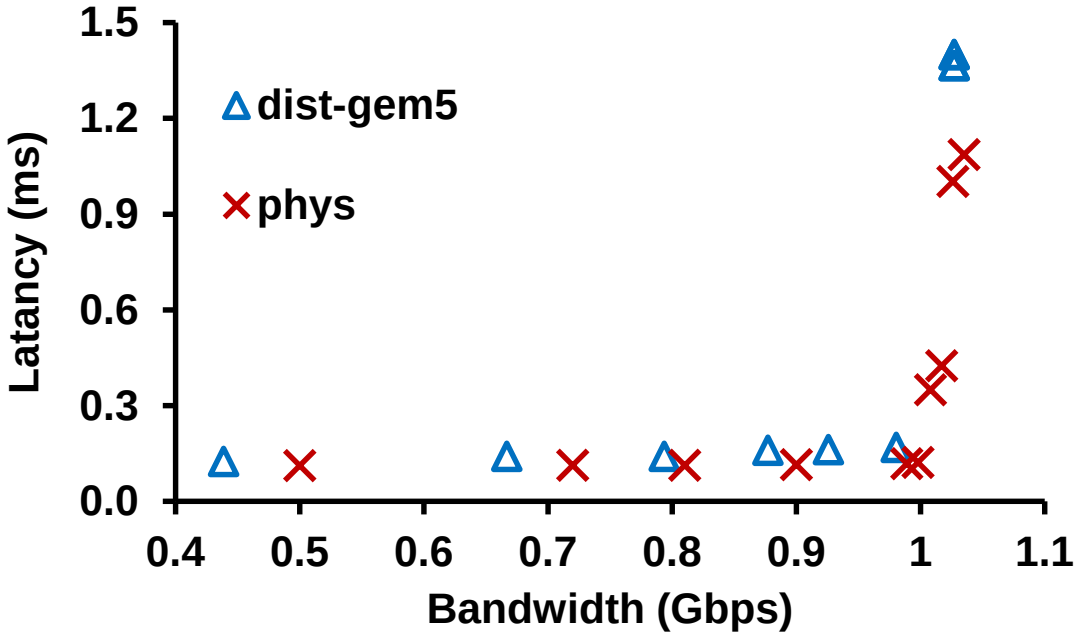
Verification

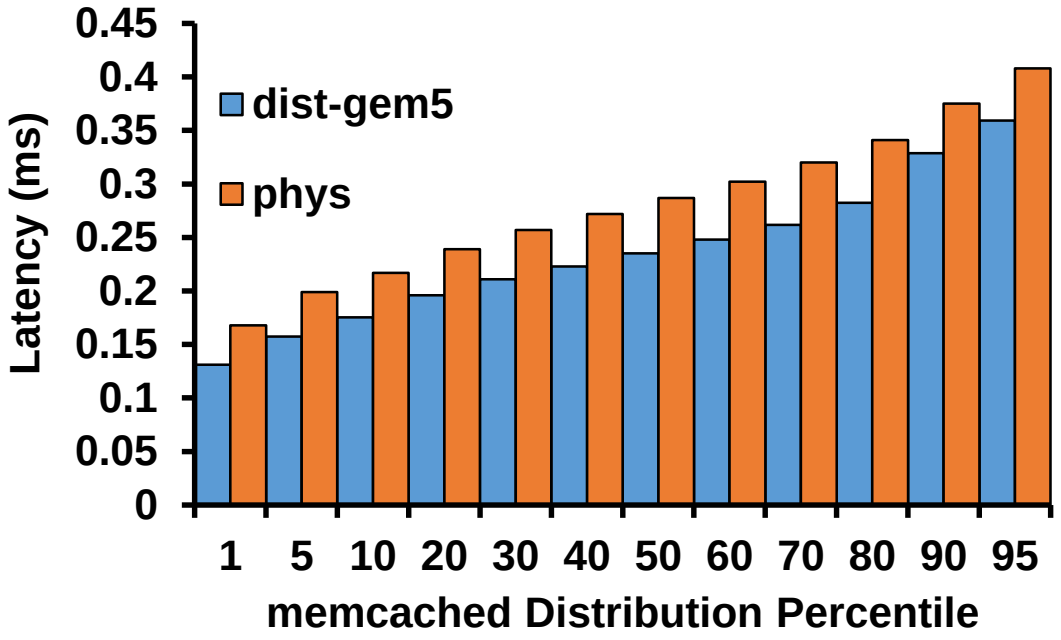
- same node/network config
  - ✓dist-gem5 generates identical simulation statistics compared to single-threaded-gem5
  - ✓different cluster sizes



Validation – network latency and bandwidth

- iperf (left) and memcahed (right)
- follows the behavior of physical setup
- 17.5% lower response time for memcached

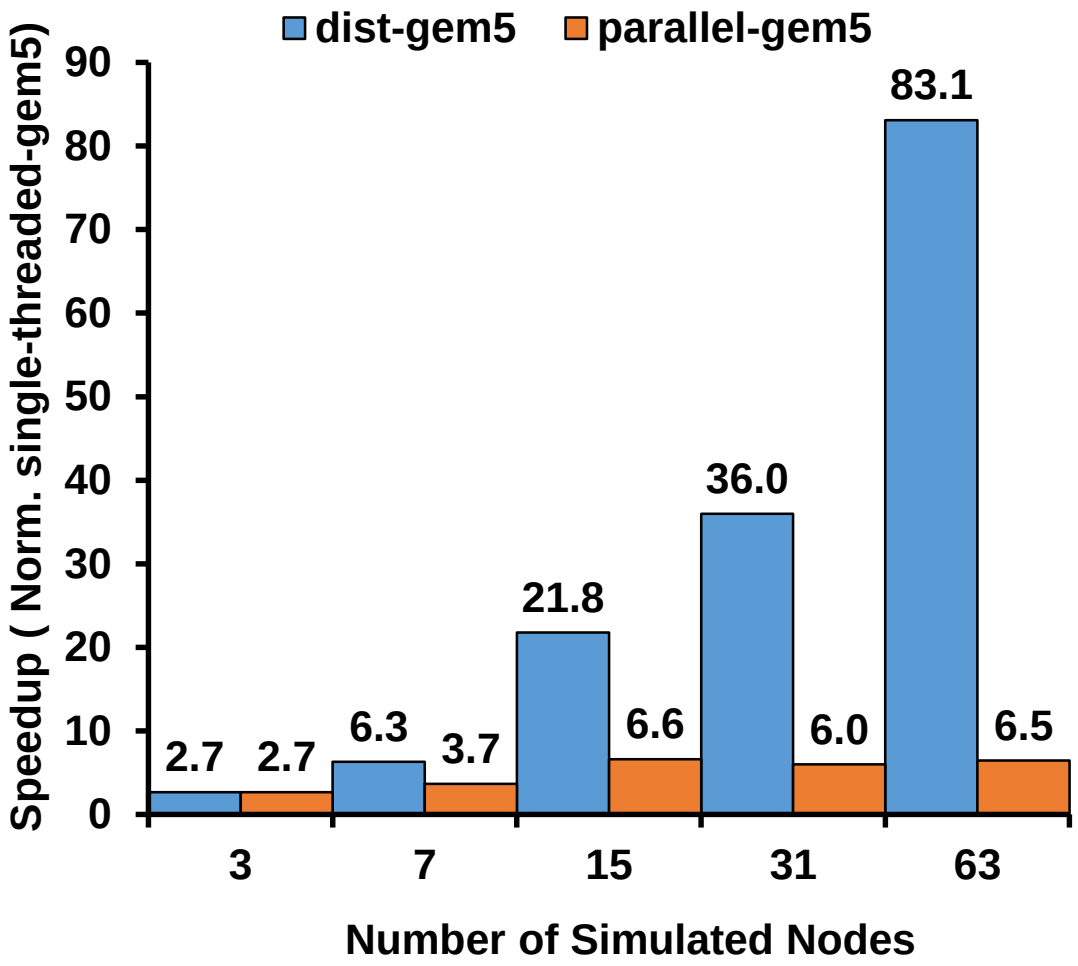




Speedup – simulation time reduction

- running httperf on each simulated node sending fixed number of requests to a unique simulated node (apache server)
- compared with single-threaded-gem5
- dist-gem5 simulating 63 nodes on 16 physical hosts is
  - ✓83.1× faster than single-threaded-gem5
  - ✓12.8× faster than parallel-gem5

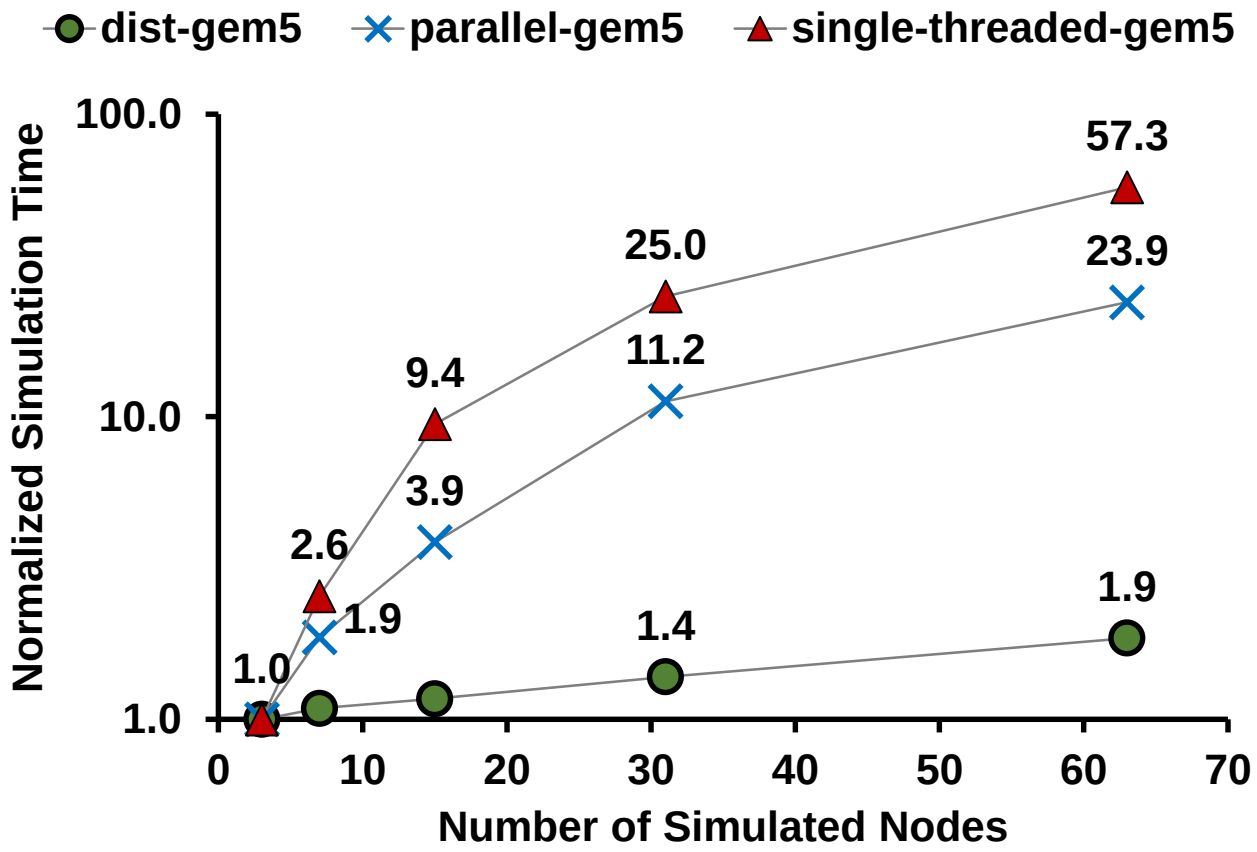
speedup of parallel-gem5 saturates!



Scalability – simulation time vs. simulated cluster size

- simulation time increase for simulating 64 vs. 3 nodes:
  - ✓57.3× for Single-threaded-gem5
  - ✓23.9× for parallel-gem5
  - ✓1.9× for dist-gem5

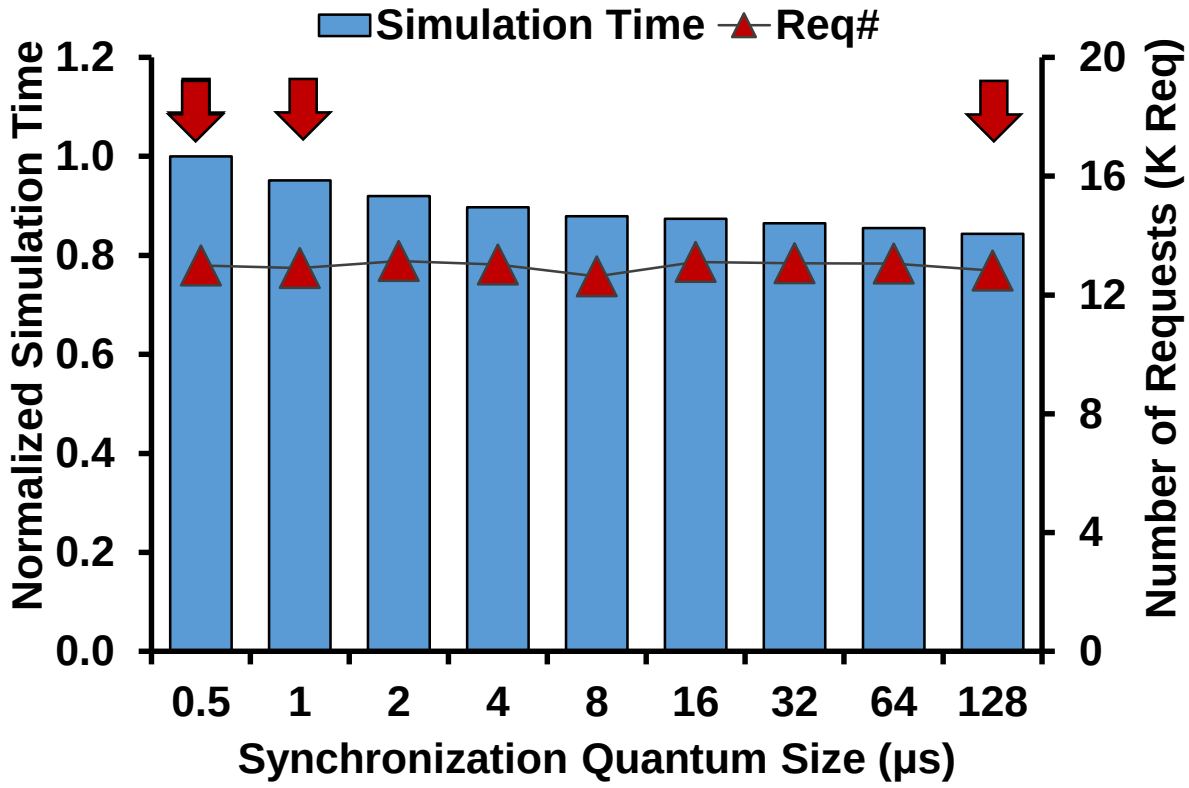
dist-gem5 scales well!



Synchronization overhead

- sweep synchronization quantum size
- # of http req remains near constants
  - ✓maximum 2.6% variance
  - ✓almost the same amount of work done at each quantum size
- simulation time improvement
  - ✓4.9% from 0.5 µs to 1 µs
  - ✓15.7% from 0.5 µs to 128 µs

dist-gem5 synchronization is efficient!



Conclusion

- dist-gem5 is a distributed version of gem5 for modeling computer clusters
  - ✓validated against a physical cluster
  - ✓accurate/deterministic
  - ✓rich off-chip network modeling
  - ✓83.1x speedup over single-threaded-gem5 simulating a 63 node cluster
- integrated to mainstream gem5
  - ✓available at gem5.org
  - ✓enabled via “--dist” command line option
- developed/maintained by university and industry

Thank You