Improving Disaggregated System Evaluation with Modular End-to-End Simulation

Bin Gao, Hejing Li, Jialin Li, Antoine Kaufmann

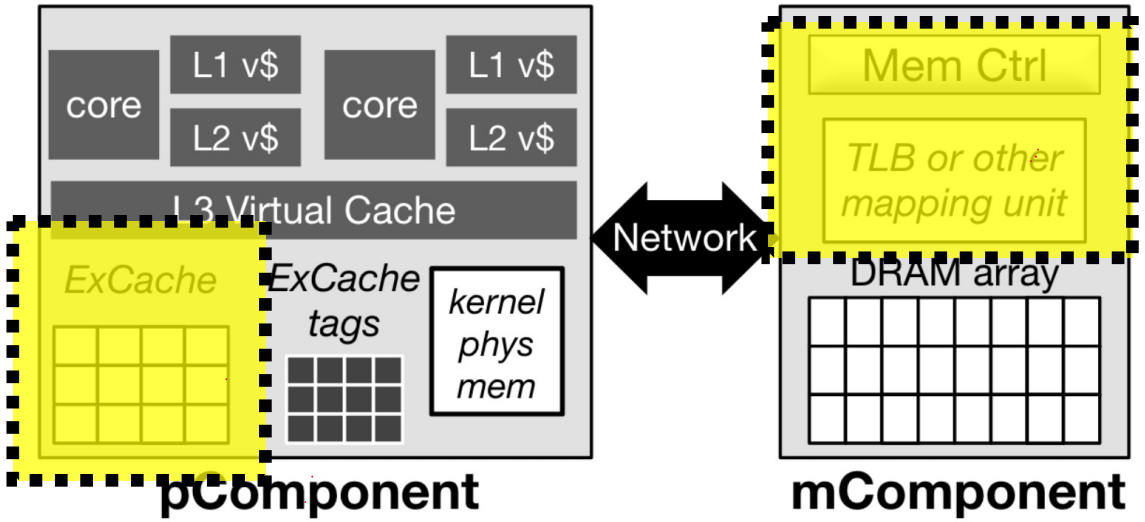
MAX PLANCK INSTITUTE
FOR SOFTWARE SYSTEMS

Why is disaggregated systems research often hard to evaluate?

Challenge 1: Radical Hardware Changes

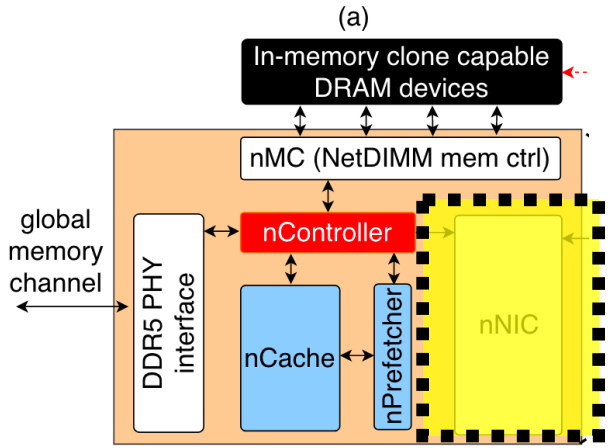
Hardware testbeds are often not available

- Memory Disaggregation
  - ❑LegoOS [OSDI'18], Clio [ASPLOS'22], MIND [SOSP'21], ...



LegoOS [OSDI'18]

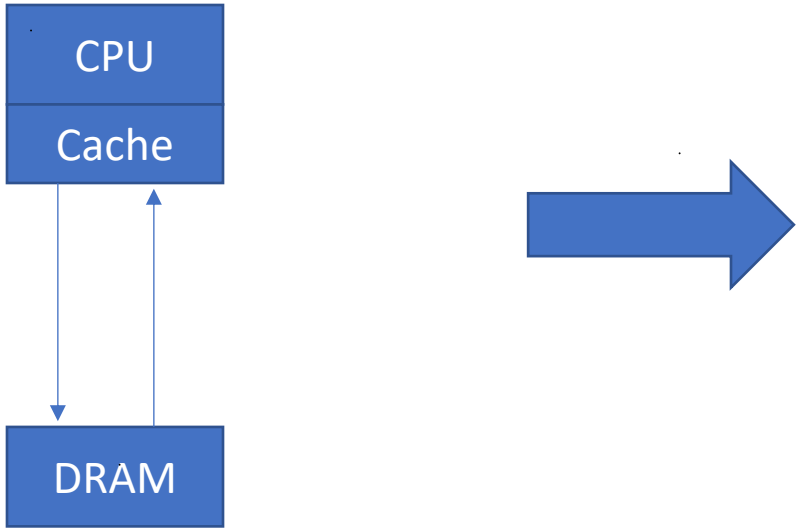
- I/O Devices
  - ❑NetDIMM [MICRO'19], NanoPU [OSDI'21], ...



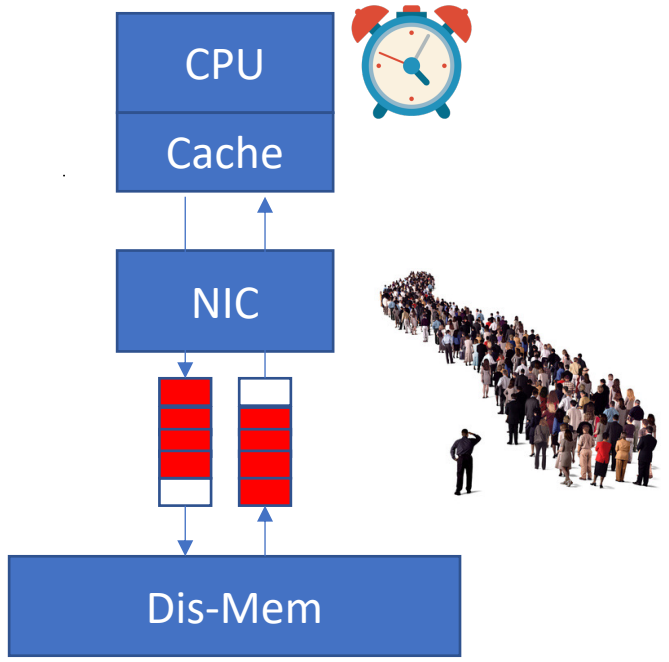
NetDIMM [MICRO'19]

Challenge 2: Component Interactions

Disaggregated systems add more components to critical path with complex interactions.



Local Memory



Disaggregated Memory

Full system / end-to-end measurements are essential

Challenge 3: Increased System Scale

Disaggregated systems by design require larger scale

Typical minimal disaggregated system size: 1 — N racks

How do we evaluate disaggregated systems research?

Approach 1: Emulation

	Accurate	Flexible	Scalable
Emulation	:-(	:-)	:-)

Pros:

- Flexible
  - Just need to implement an emulator
- Typically reasonably fast & scalable

Cons:

- Limited performance accuracy
  - Emulation matches functionality but not performance

Approach 2: Component Simulation

	Accurate	Flexible	Scalable
Emulation	:-(	:-)	:-)
Component Simulation	:-(	:-)	:-(

Pros:

- Flexible
  - Implement/modify an appropriate simulator
- Portable & reproducible
  - Many simulators are deterministic

Cons:

- No full system end-to-end performance results
- Simulations (often) take a long time to run

Approach 3: Full System Simulation

	Accurate	Flexible	Scalable
Emulation	:-(	:-)	:-)
Component Simulation	:-(	:-)	:-(
Full System Simulation	:-)	:-/	:-(

Pros:

- Accurate end-to-end results
  - As long as the simulator is good
- Portable & reproducible

Cons:

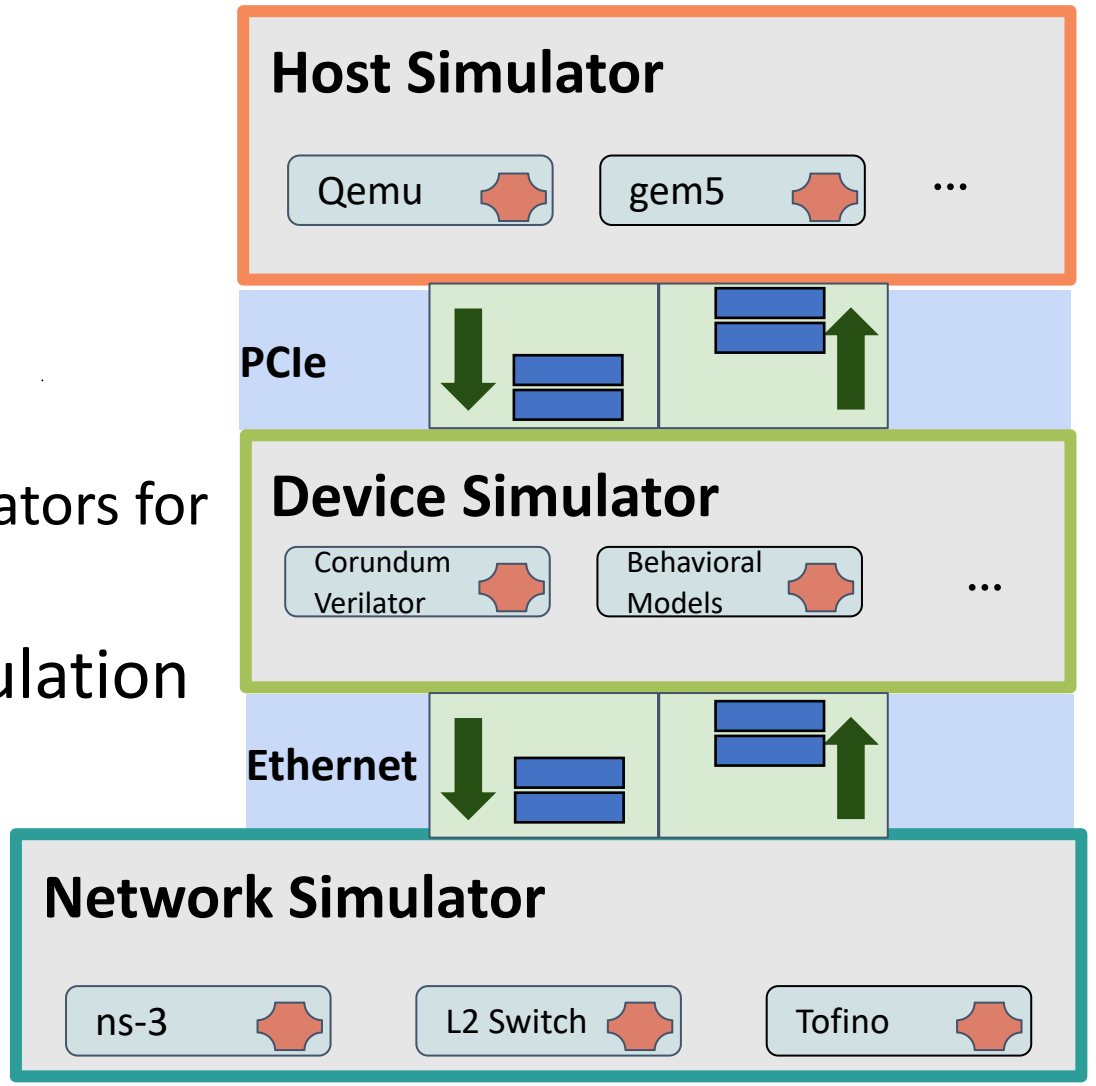
- Significant effort for development and validation
  - Typically one-off modifications of some existing simulator
- Not scalable and very slow

How can we make end-to-end simulations easier?

Modularity is Key

Starting point: **SimBricks**

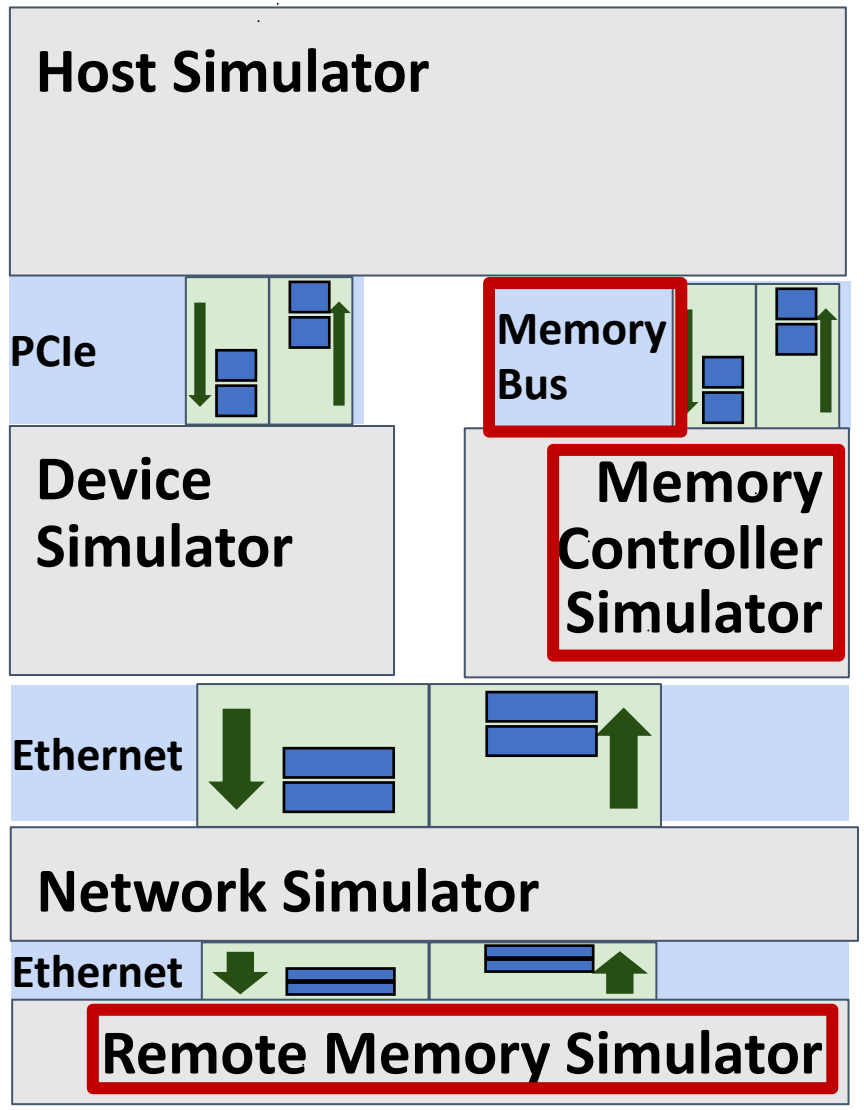
- Modular simulation framework
  - Combine & connect multiple different simulators for individual components
- Built for end-to-end network system simulation
- Scales to 1000s of components
- Reproduces performance results from prior research work



SimBricks: End-to-End Network System Evaluation with Modular Simulation
Hejing Li, Jialin Li, Antoine Kaufmann. In *ACM SIGCOMM 2022*.

Extending *SimBricks* for Disaggregated Systems

- Add missing components
  - Already supported: **Host, device, network**
  - Need: **Remote memory controller, network attached memory/accelerator/...**
- Add missing interconnect
  - Already supported: **PCIe, Ethernet**
  - Need: **Memory bus**
    + corresponding adapters in host simulators



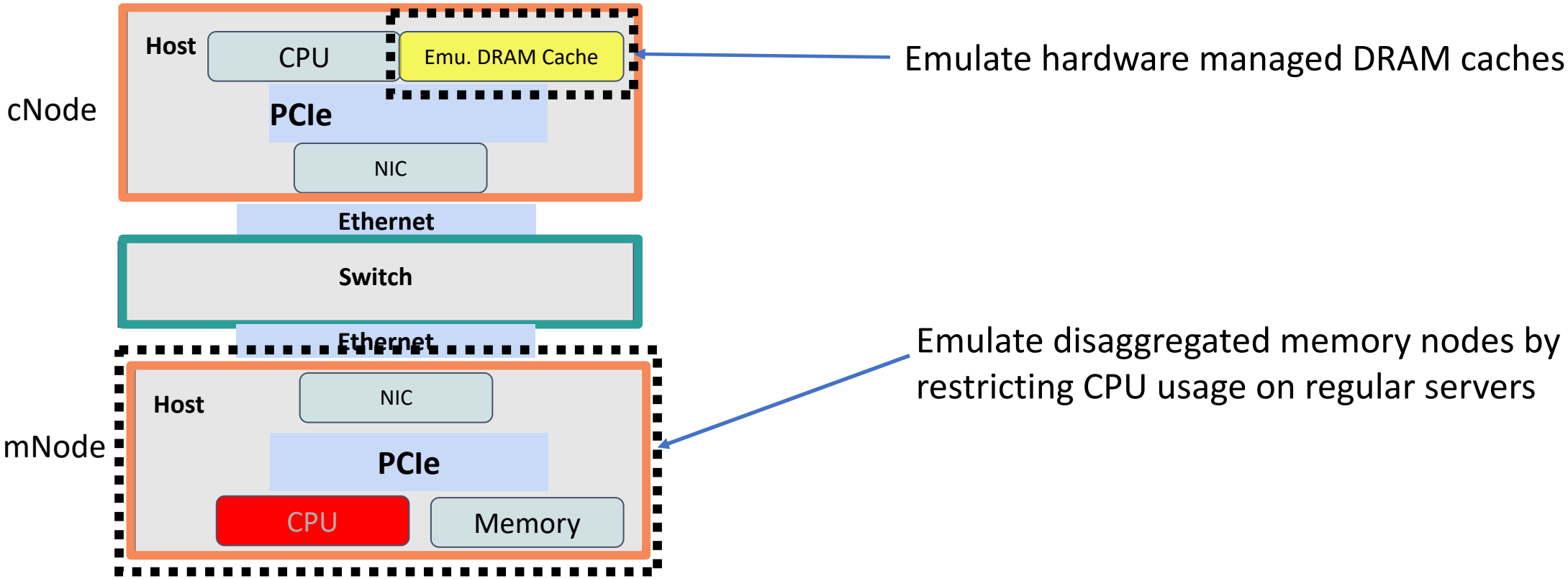
How to use our framework to simulate a disaggregated system?

- Choose the appropriate existing component simulators
  - Processor: gem5, Qemu, ...
  - Network: ns-3, Tofino, ...
  - Device and hardware: Verilator, NIC models, …
- *Optional:* for new simulators and hardware designs, implement *adapters* that connect to SimBricks interfaces
  - PCIe, Ethernet, memory bus, ...
- Write (python) configuration script that
  - Instantiates and connects all components
  - Sets simulation parameters
- Run simulation

Case Studies

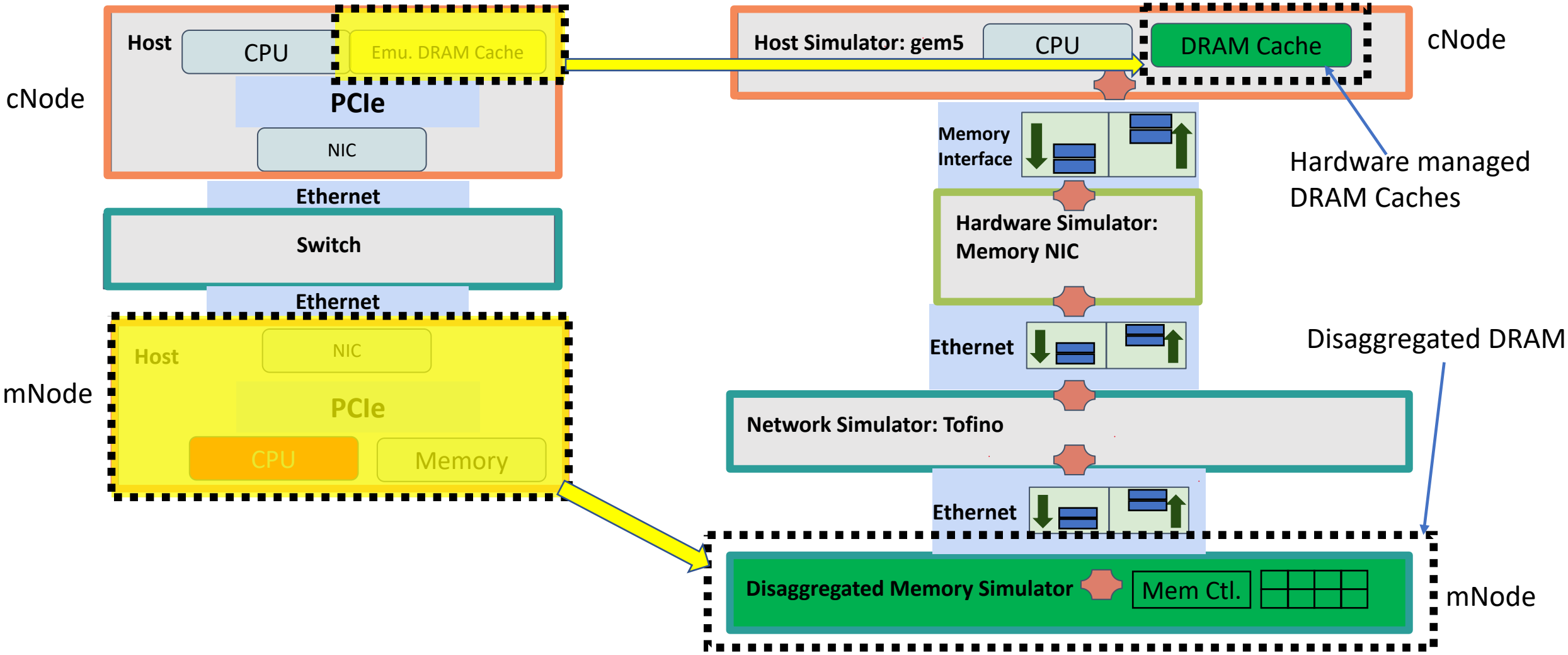
- Network-attached Disaggregated Memory
  - LegoOS [OSDI '18]
- In-network Disaggregated Memory Management
  - MIND [OSDI '21]

Example 1: Network-attached Disaggregated Memory



Emulated nodes in LegoOS [OSDI'18]

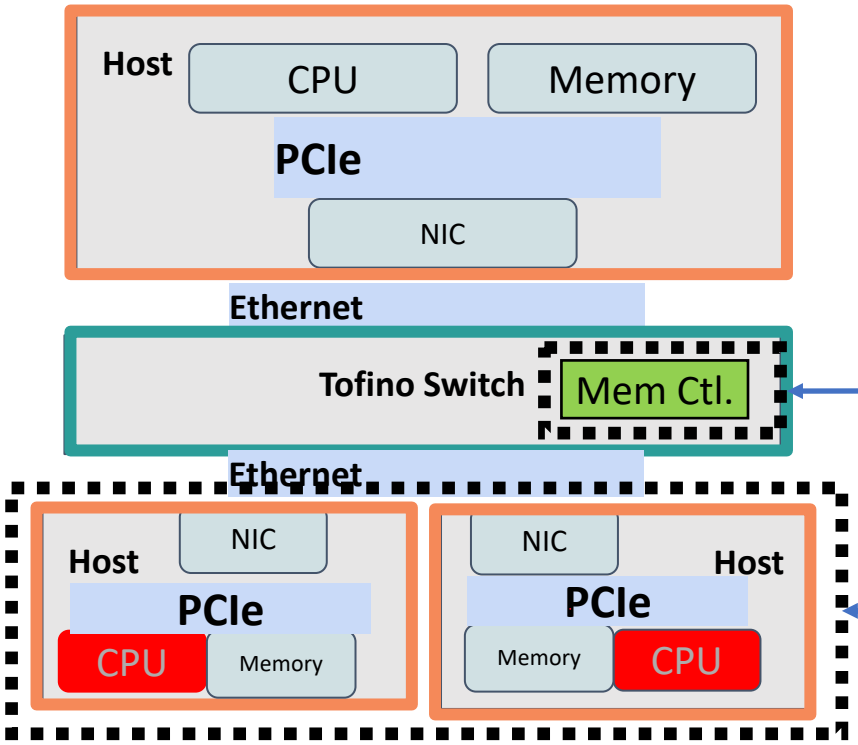
Example 1: Network-attached Disaggregated Memory



Emulated nodes in LegoOS [OSDI'18]

Disaggregated memory impl. in *SimBricks*

Example 2: In-network Support for Disaggregation

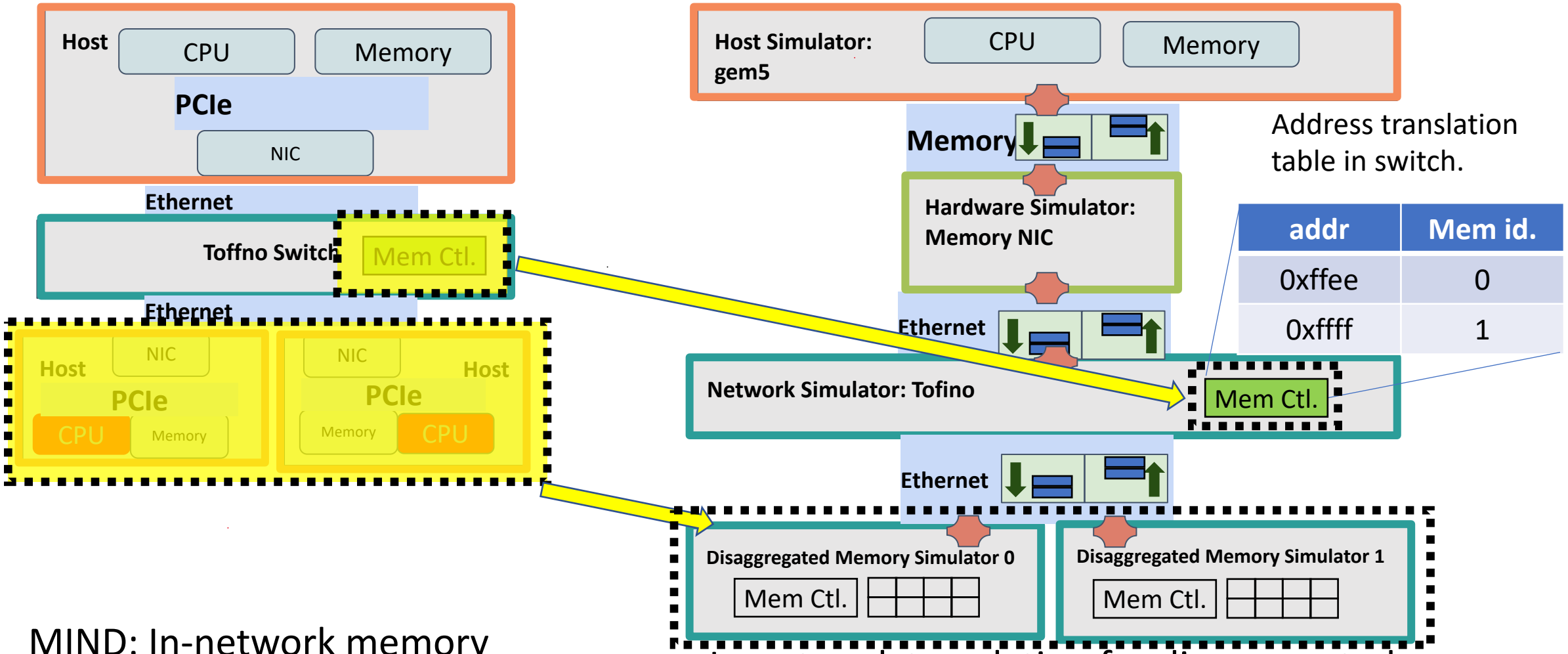


Memory management in switch

Emulate disaggregated memory nodes by avoiding using CPU on regular server

MIND [SOSP'21].

Example 2: In-network Support for Disaggregation



MIND: In-network memory management [SOSP'21].

In-network translation for disaggregated memory in ***SimBricks***

Example 2: In-network Support for Disaggregation

Preliminary Results

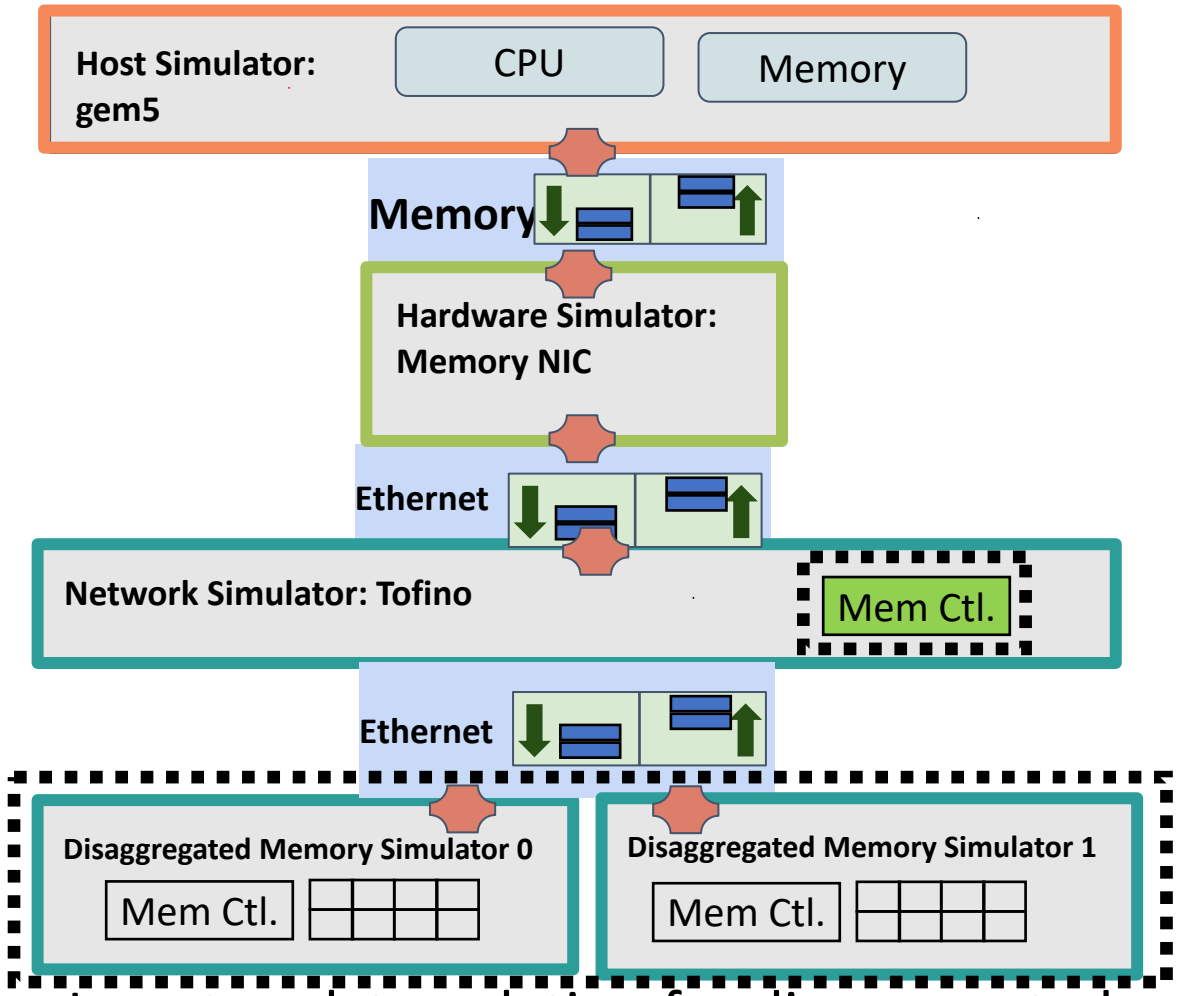
Benchmark: Sysbench Memory

Nodes: 3 compute, 2 memory

Simulation time: ~ 78min

Nodes: 20 compute, 5 memory

Simulation time: ~ 112min



In-network translation for disaggregated memory in ***SimBricks***

Future Directions

- Parallelize component simulators through decomposition
  - Use SimBricks techniques to connect and synchronize pieces
- Hardware accelerated simulation
  - Starting by integrating FireSim
- Validation through composition
  - Are modular combinations of individually validated component simulators valid?
- Implementation:
  - Interfaces: CXL / Cache-Coherent memory interface
  - RDMA NIC Simulator

Conclusion

SimBricks can enable full system evaluation for disaggregated systems

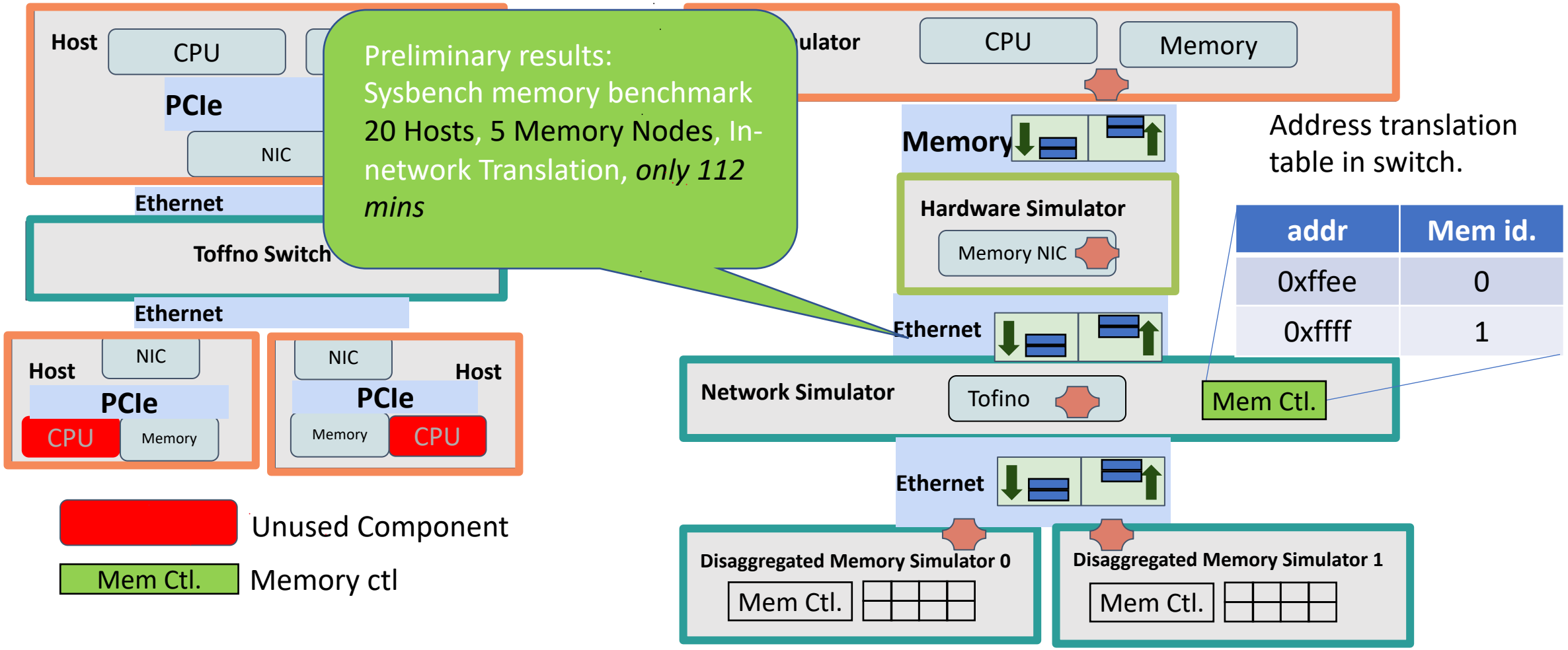
- Even with radical hardware changes
  - (Including simulating RTL with performance beyond FPGAs)
- Scales to simulate 100s-1000s of nodes

Preliminary work, please reach out if you are interested!

https://simbricks.github.io

https://github.com/simbricks

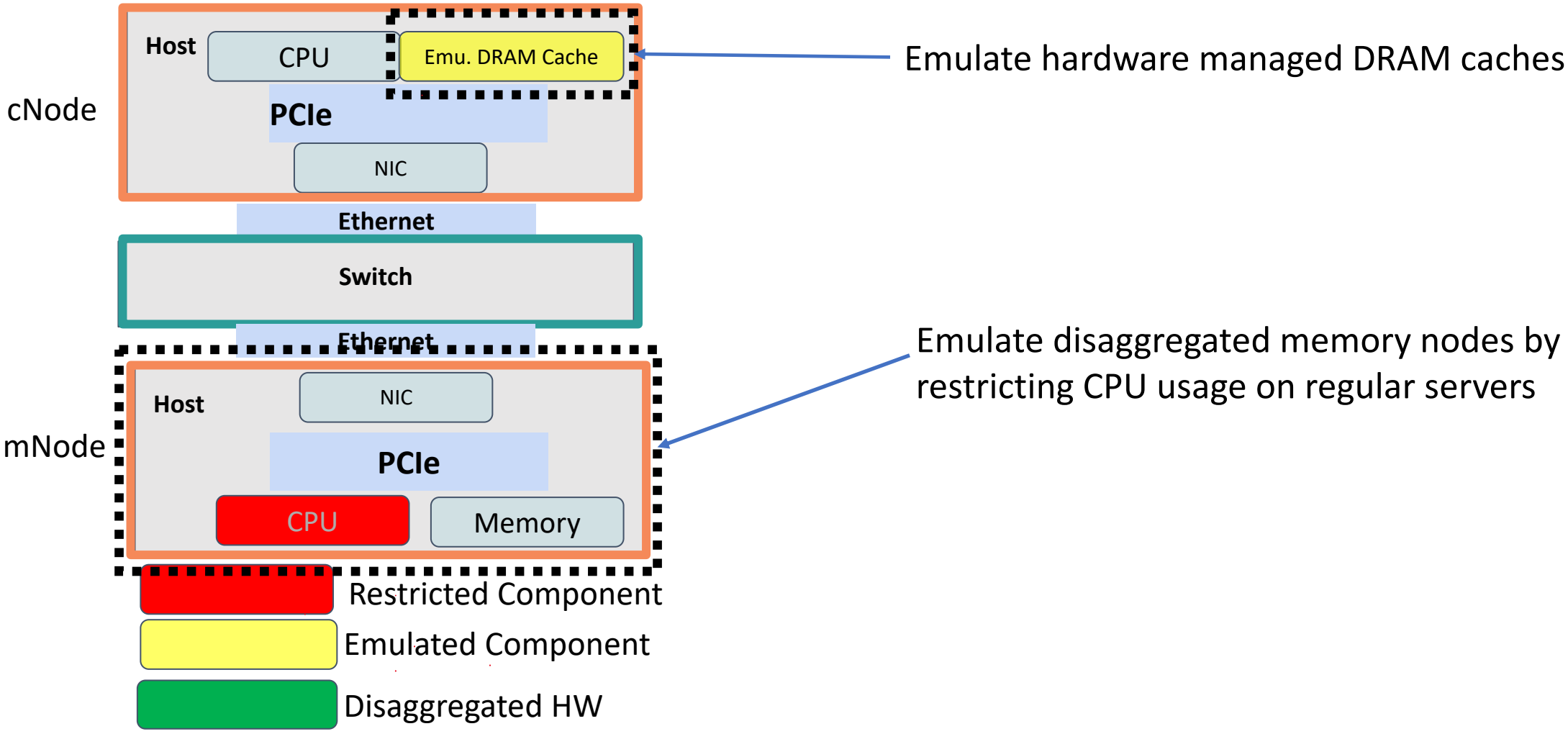
Example 2: In-network Support for Disaggregation



MIND [OSDI'21].

In-network translation for disaggregated memory in ***Simbricks***

Example 1: Network-attached Disaggregated Memory

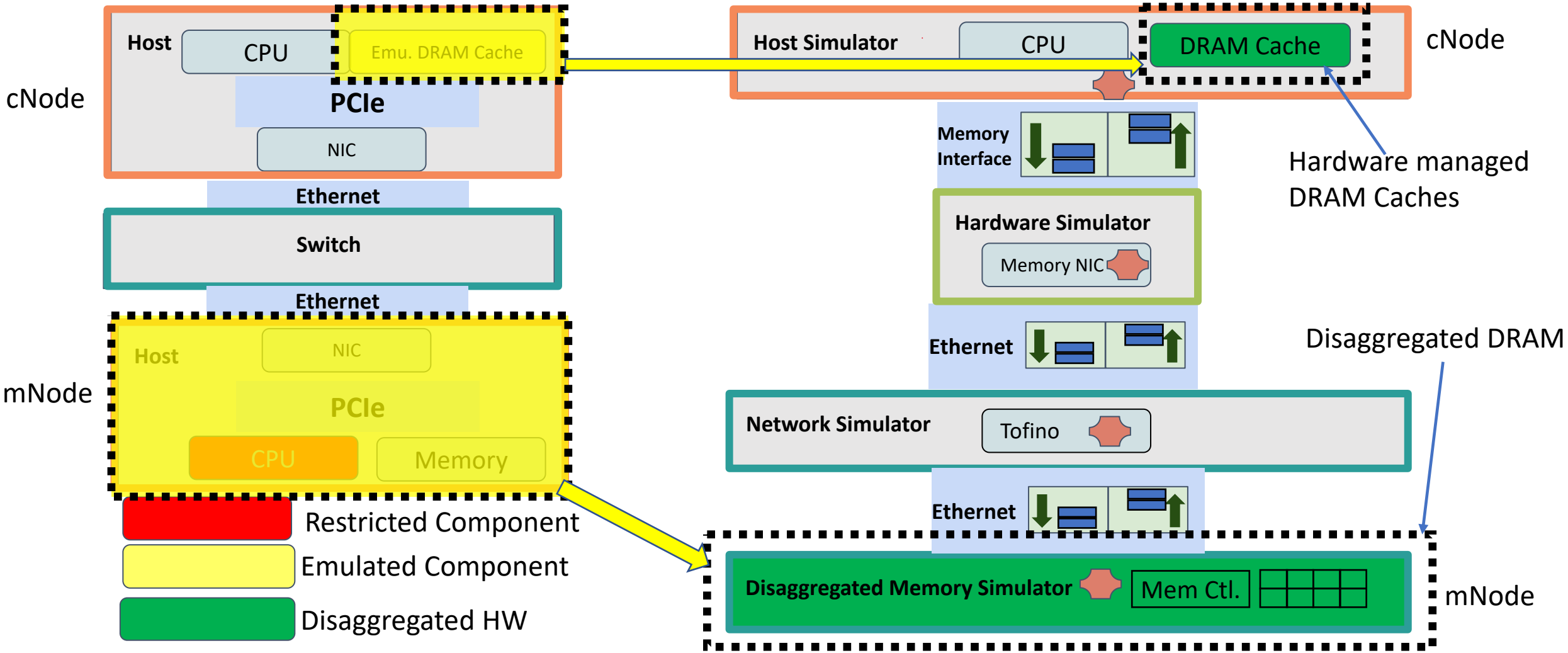


Emulate hardware managed DRAM caches

Emulate disaggregated memory nodes by restricting CPU usage on regular servers

Emulated nodes in LegoOS [OSDI'18]

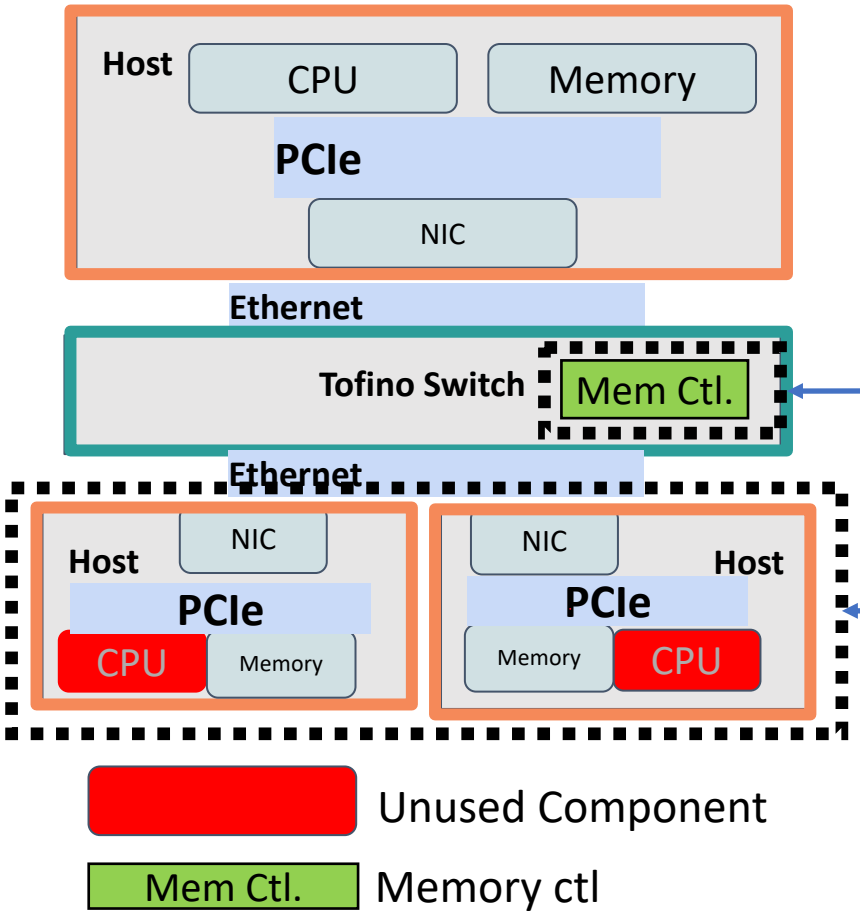
Example 1: Network-attached Disaggregated Memory



Emulated nodes in LegoOS [OSDI'18]

Disaggregated memory impl. in *SimBricks*

Example 2: In-network Support for Disaggregation

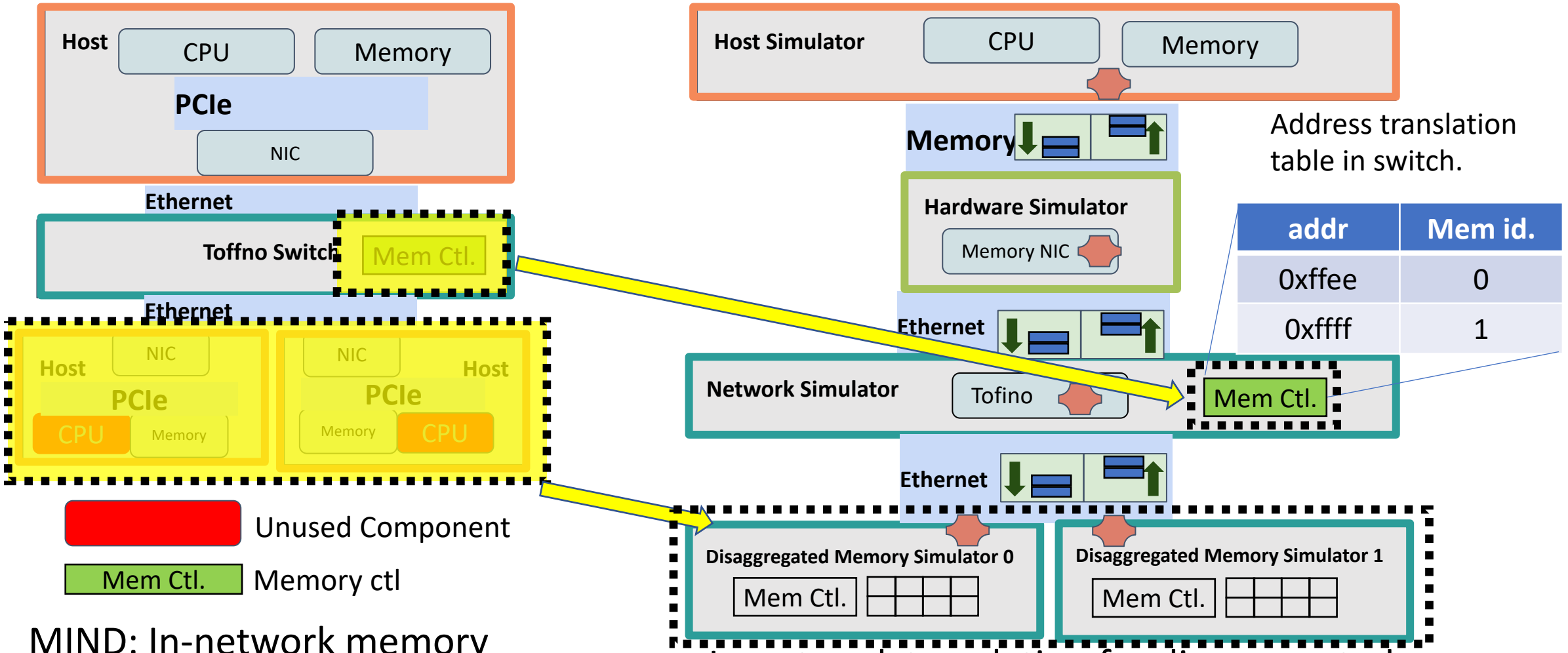


Memory management in switch

Emulate disaggregated memory nodes by avoiding using CPU on regular server

MIND [OSDI'21].

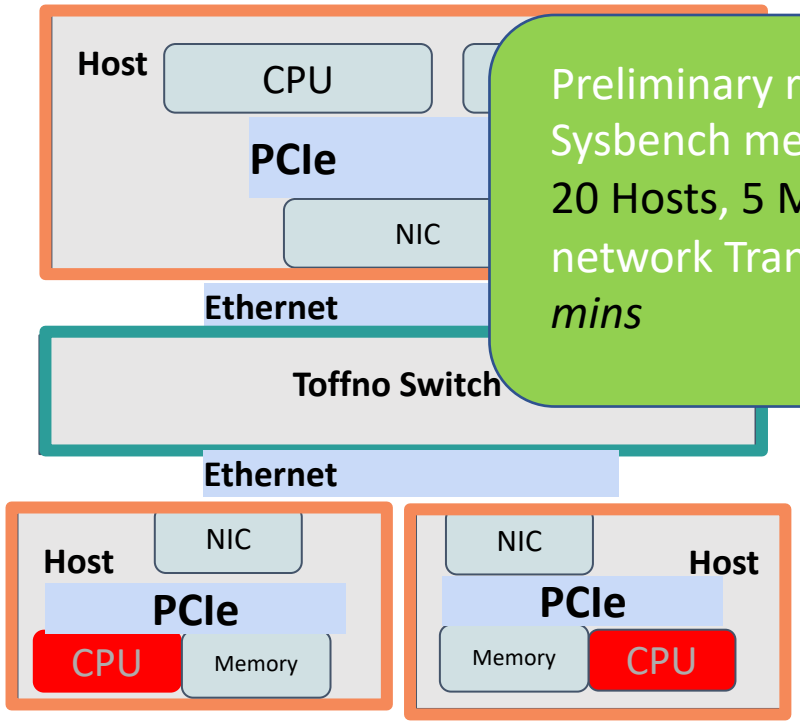
Example 2: In-network Support for Disaggregation



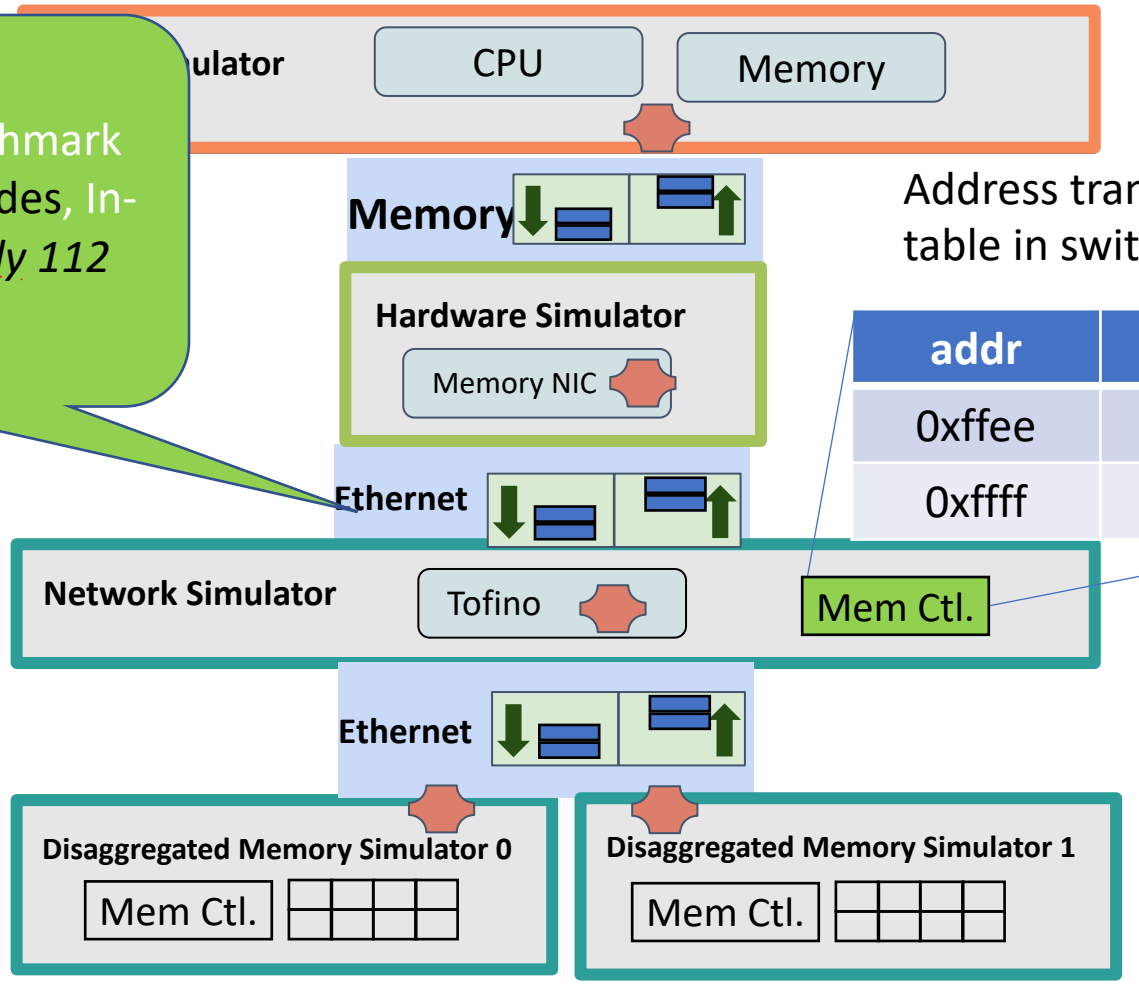
MIND: In-network memory management [OSDI'21].

In-network translation for disaggregated memory in ***Simbricks***

Example 2: In-network Support for Disaggregation



Preliminary results:
Sysbench memory benchmark
20 Hosts, 5 Memory Nodes, In-network Translation, *only 112 mins*



Address translation table in switch.

addr	Mem id.
0xffee	0
0xffff	1

Unused Component

Mem Ctl. Memory ctl

MIND [OSDI'21].

In-network translation for disaggregated memory in ***Simbricks***