Memory Harvesting in Multi-GPU Systems with Hierarchical Unified Virtual Memory

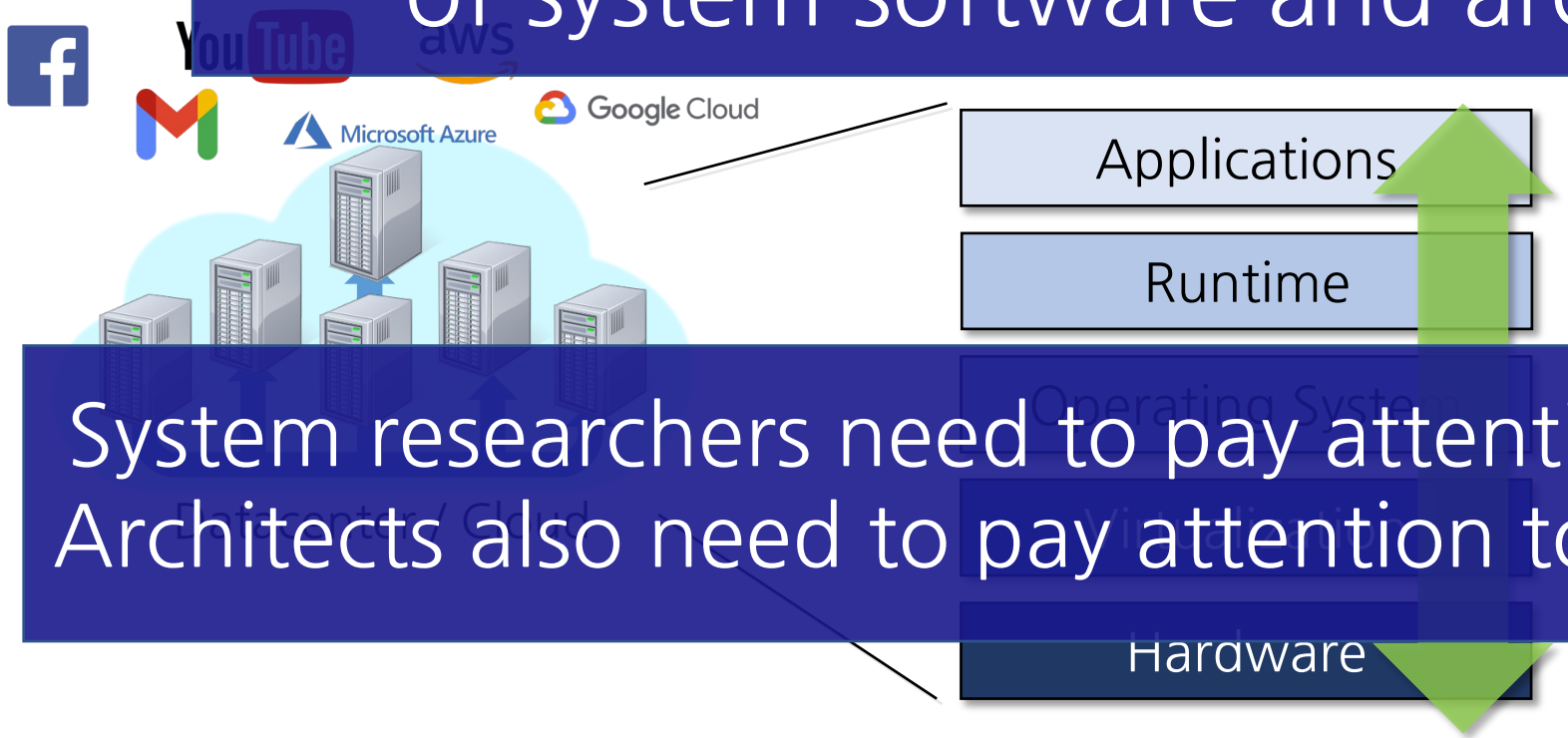
Jeongseob Ahn (안정섭)

AJOU UNIVERSITY

Source: Sangjin Choi[1], Taeksoo Kim[1], et al., “Memory Harvesting in Multi-GPU Systems with Hierarchical Unified Virtual Memory”, USENIX ATC 2022

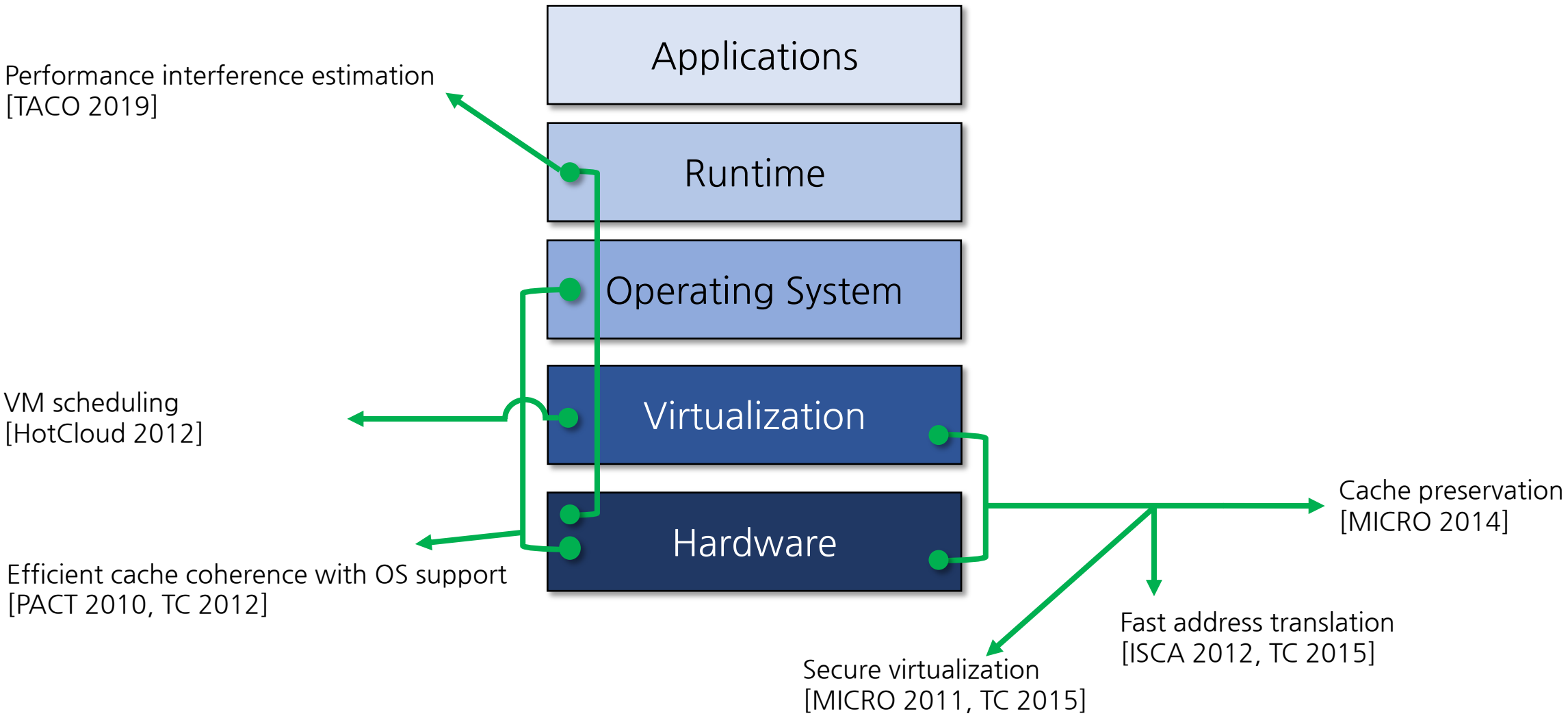
Research theme

Vision: Designing systems with interaction of system software and architecture

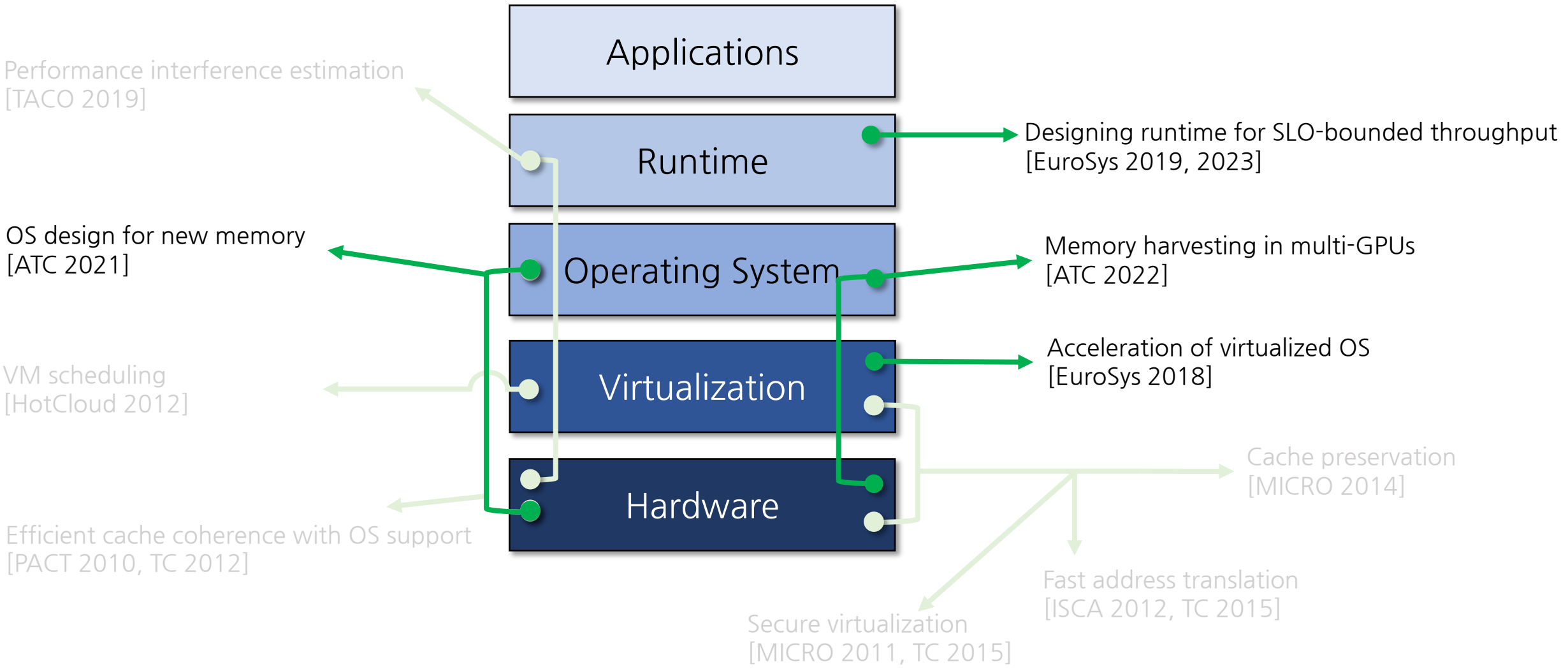


System researchers need to pay attention to architecture
Architects also need to pay attention to systems software

My research *before* Ajou Univ.



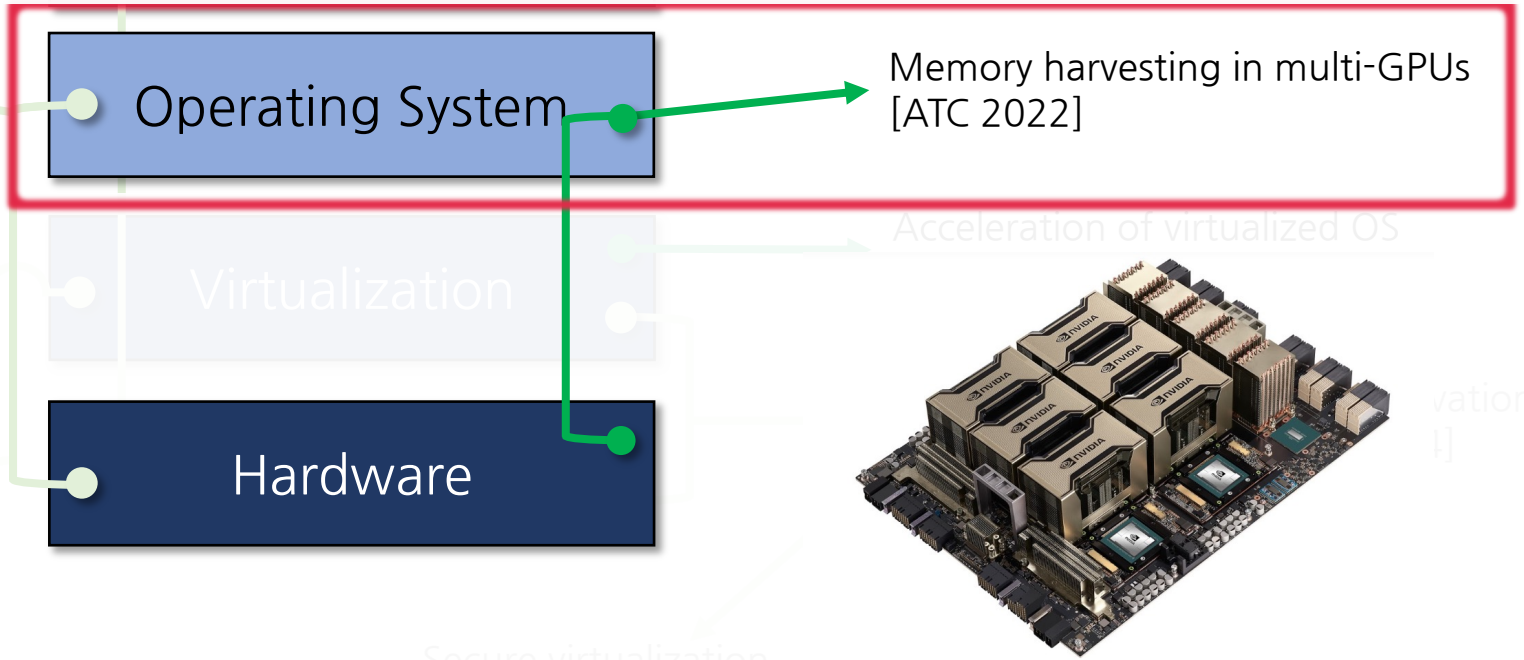
My research *at* Ajou Univ.



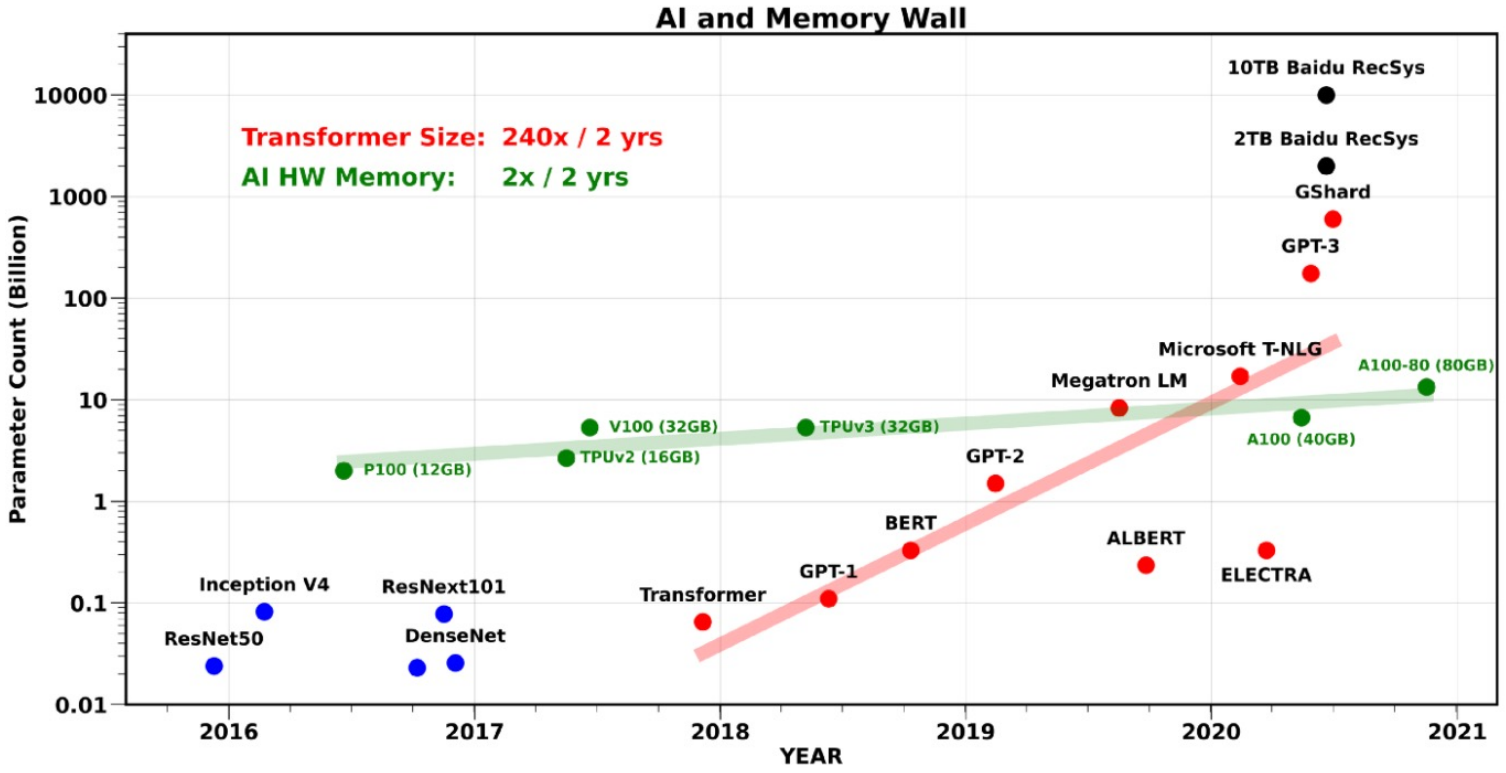
Focus of Today's Talk: **Memory capacity wall**

Every new hardware needs new software support!

Design principle: make new memory hierarchy elaborate

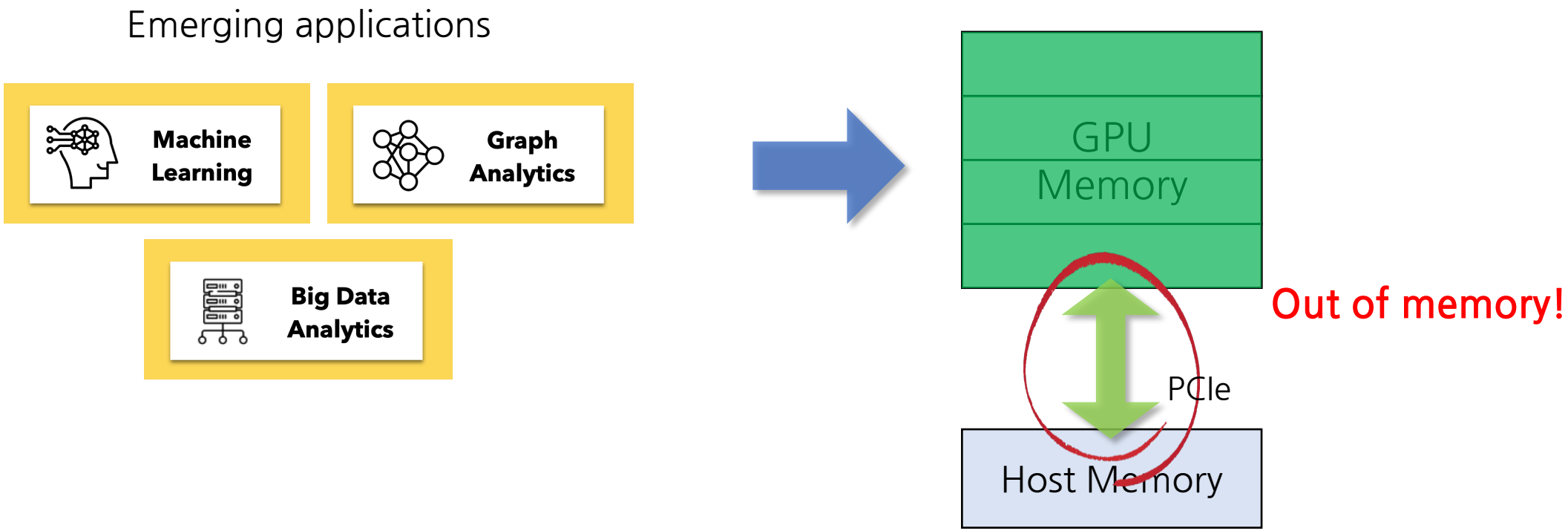


Memory capacity wall in GPUs



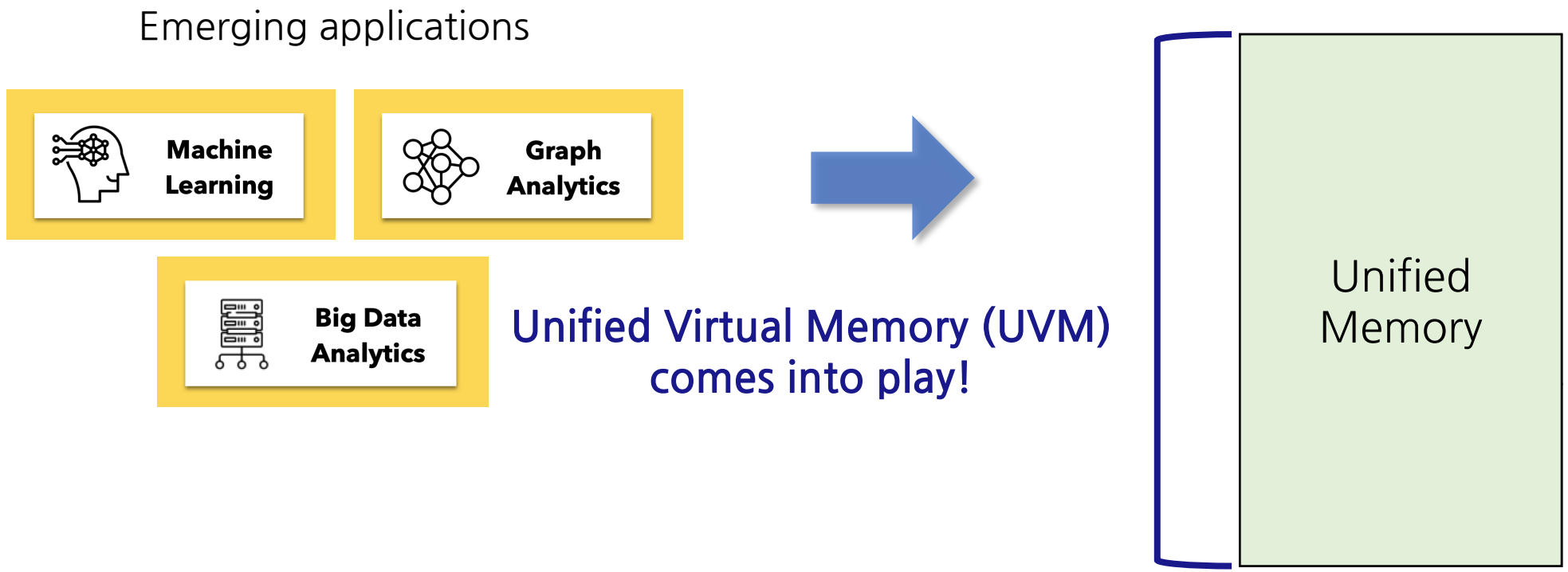
- Training AI models requires 3-4x more memory than the size of parameters
- Model size increases 240x per 2 years whereas HW memory size increase 2x per 2 years

Current status for memory capacity wall in GPU



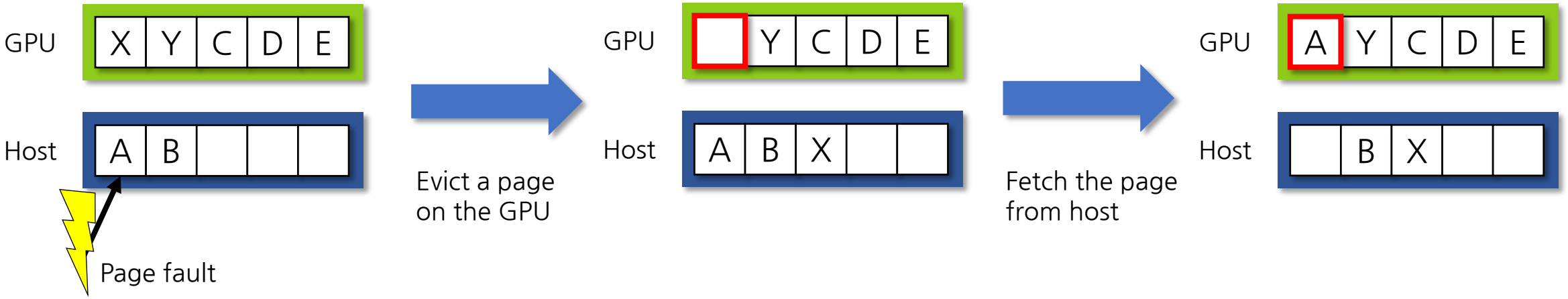
- Using host memory as an extension of GPU memory
- Data moves between host memory and GPU memory through PCIe

Current status for memory capacity wall in GPU



- GPU driver provides the unified address space across the GPU and host
- Thanks to the **paging** mechanism

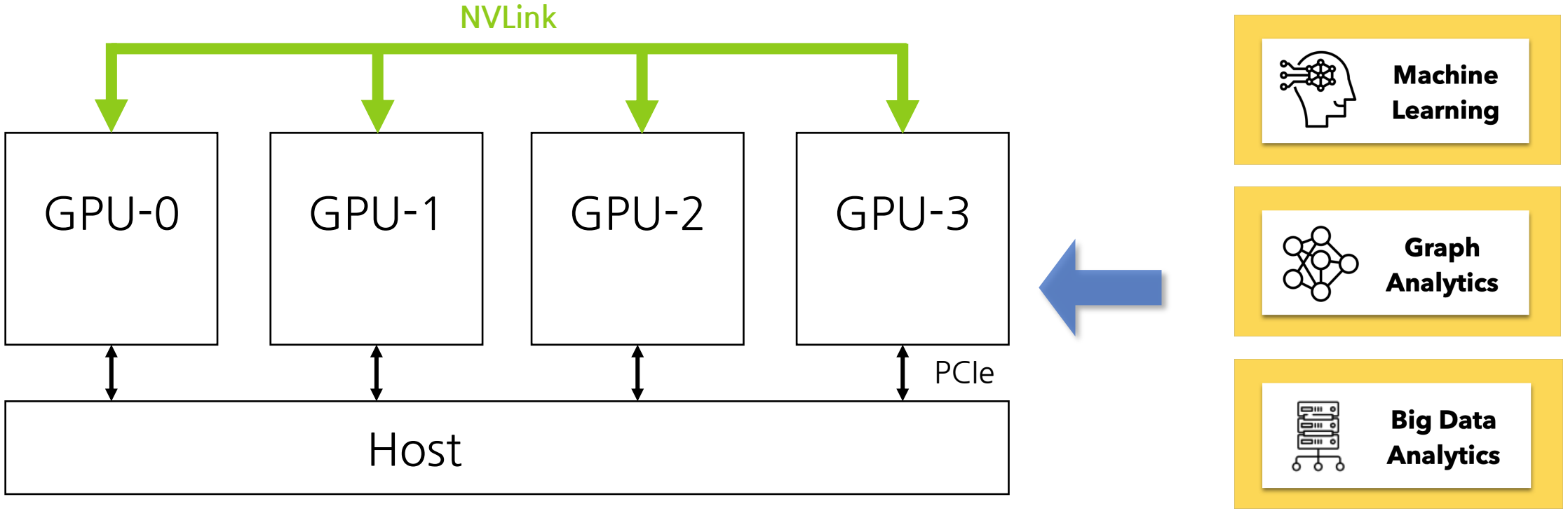
Background on unified virtual memory



Page fault latency = eviction latency + fetch latency

Modern Multi-GPU servers

Providing better bandwidth and throughput than PCIe
→ Lead to fast GPU-to-GPU communication

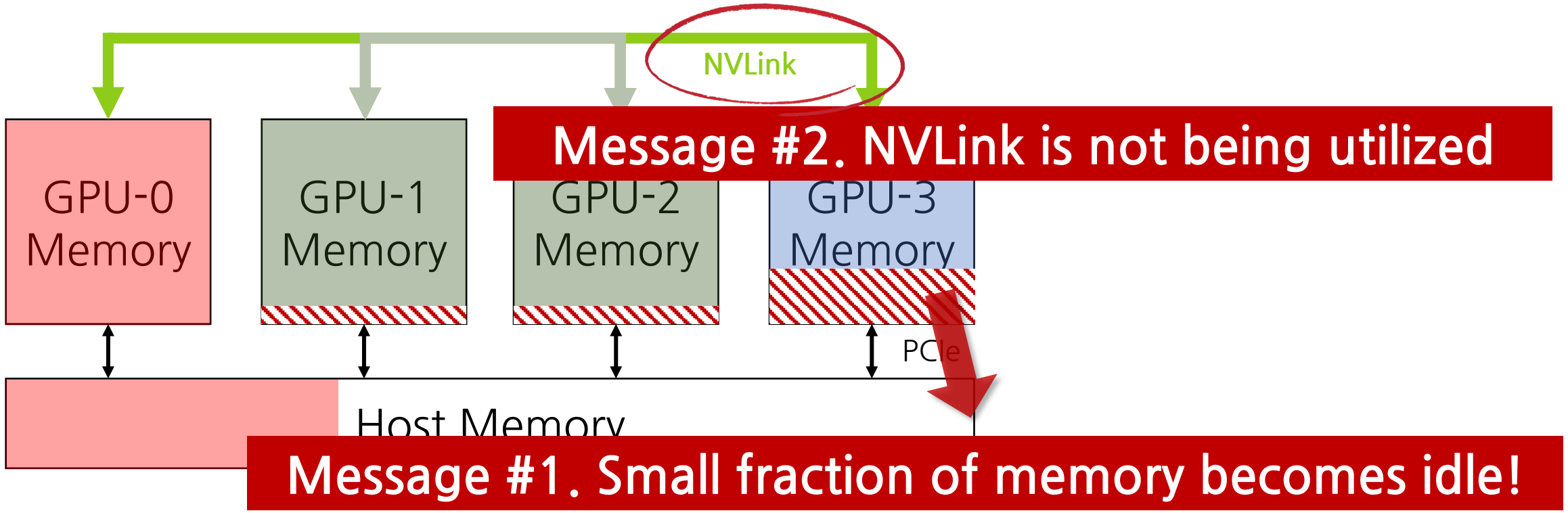


Hosting multiple workloads from users (e.g., employees in a company)

How does UVM work on multiple GPUs?

	GPU-0	GPU-1	GPU-2	GPU-3
Workload	Pagerank	VGG16		WCC
Dataset	soc-Twitter	256 batch		soc-sinaweibo
Memory usage	1.87x	0.96x		0.79x

Scenario → Three apps share a 4-GPU server (multi-job / multi-GPU case)



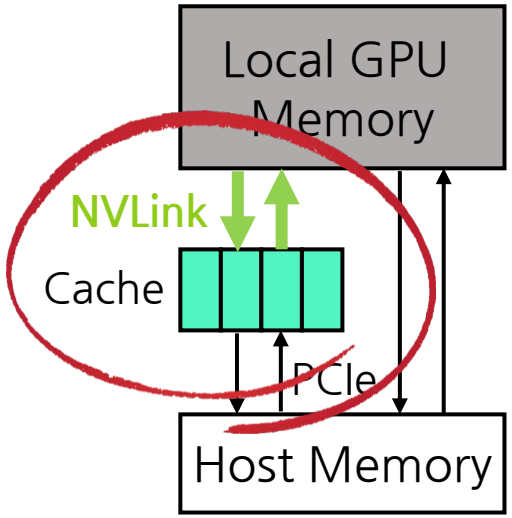
HUVM: hierarchical unified virtual memory

Message #1. Small fraction of memory becomes idle!

Message #2. NVLink is not being utilized

Harvesting idle memory in neighbor GPUs to lower performance overhead under memory oversubscription

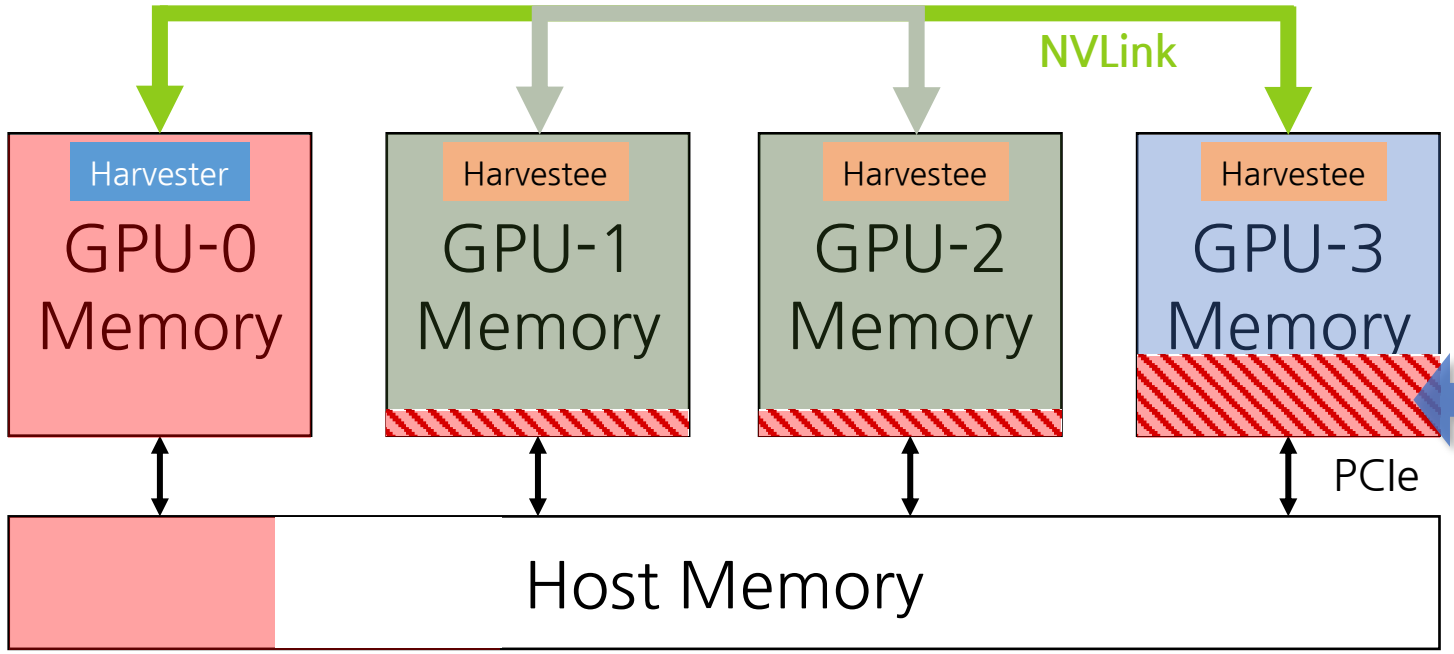
Hiding latency from/to host memory!



Memory harvesting

	GPU-0	GPU-1	GPU-2	GPU-3
Workload	Pagerank	VGG16		WCC
Dataset	soc-Twitter	256 batch		soc-sinaweibo
Memory usage	1.87x	0.96x		0.79x

Scenario → Three apps share a 4-GPU server (multi-job / multi-GPU case)

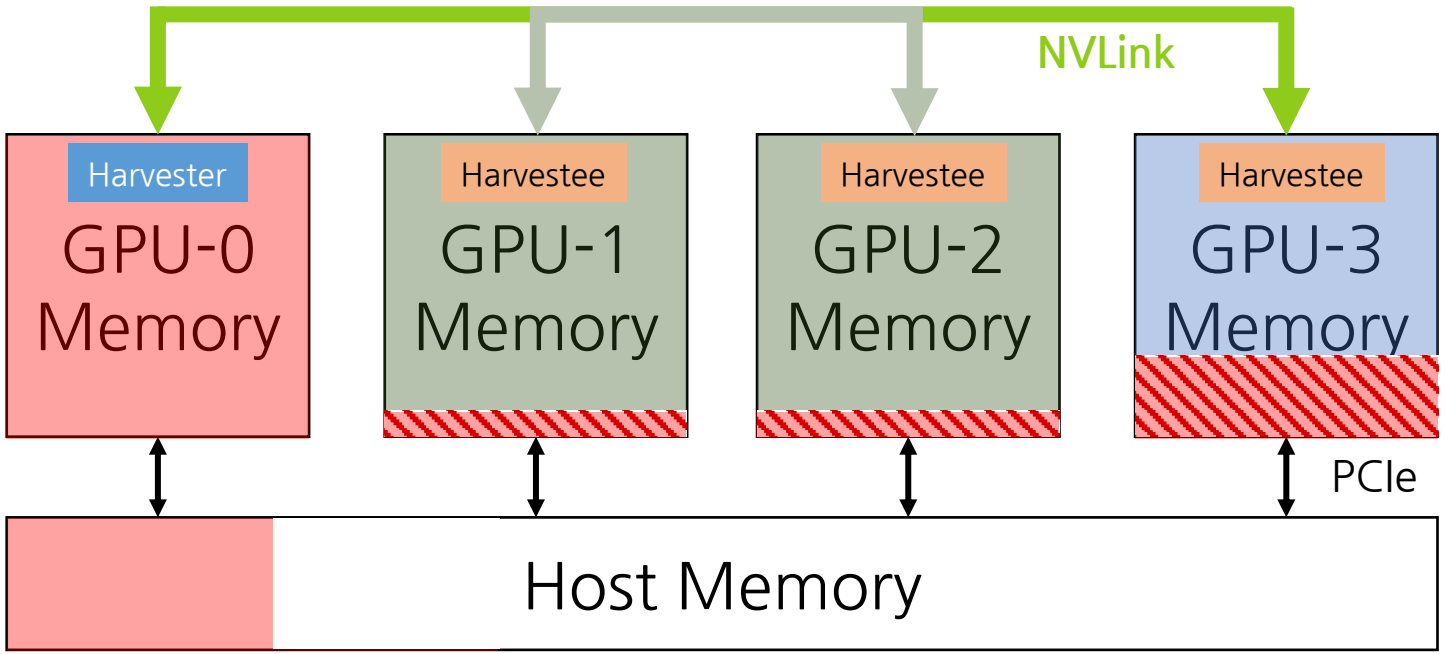


HUVM enables GPU-0 to harvest the idle memory of neighbor GPUs

Design challenges

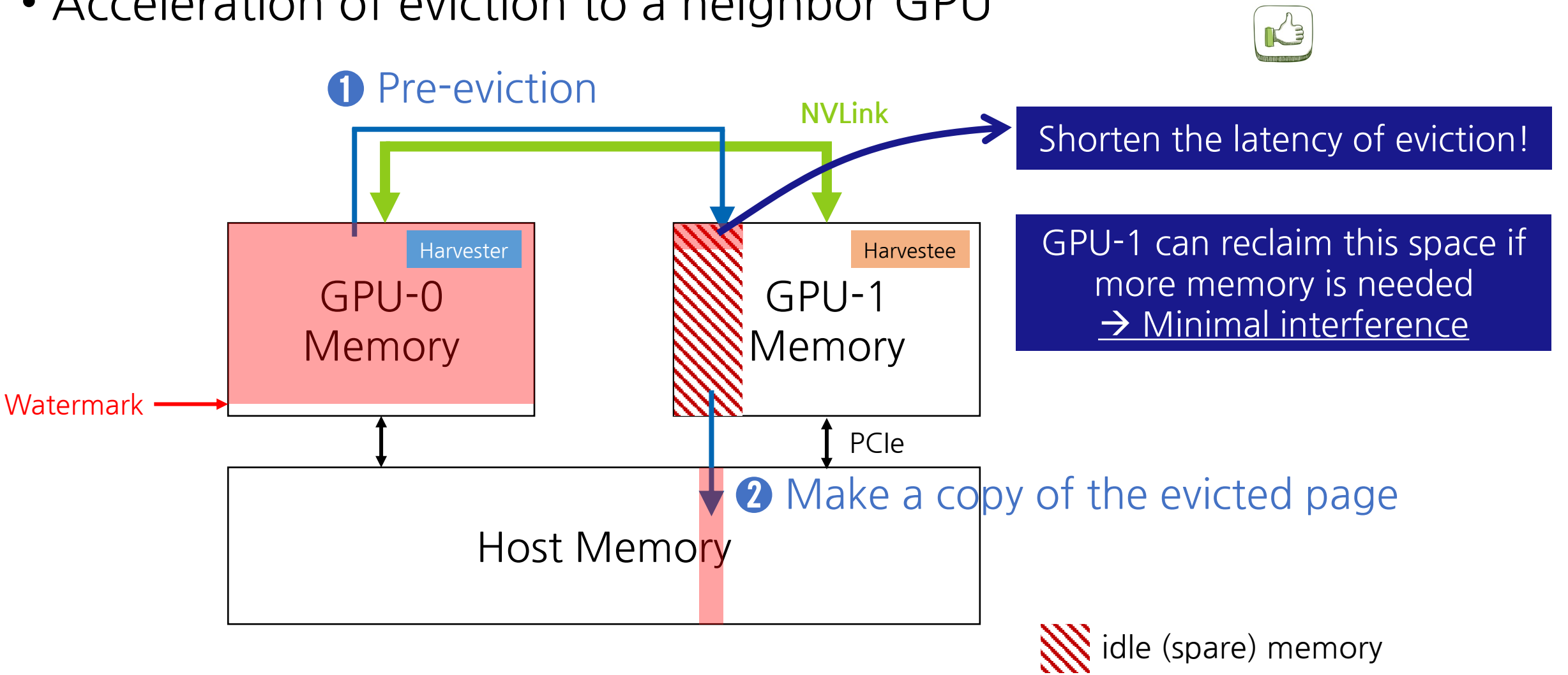
1. Effective harvesting

2. Minimal interference



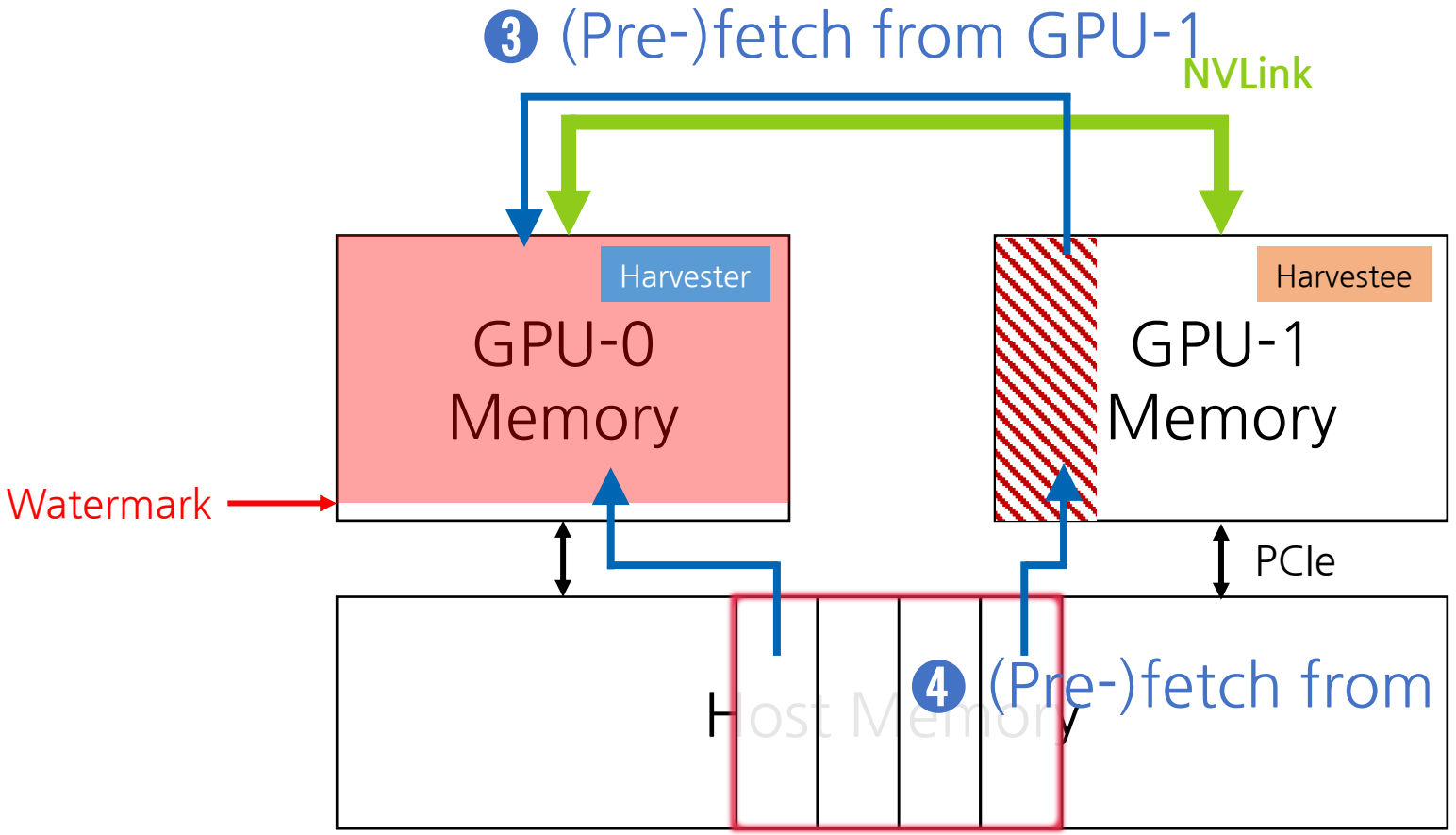
Hiding eviction latency to host

- Acceleration of eviction to a neighbor GPU



Hiding fetch latency from host

- Acceleration of fetch from a neighbor GPU



Shorten the latency of fetch!

When prefetching, we can exploit path diversity (PCIe and NVLink)

We use a simple next page and stride prefetcher

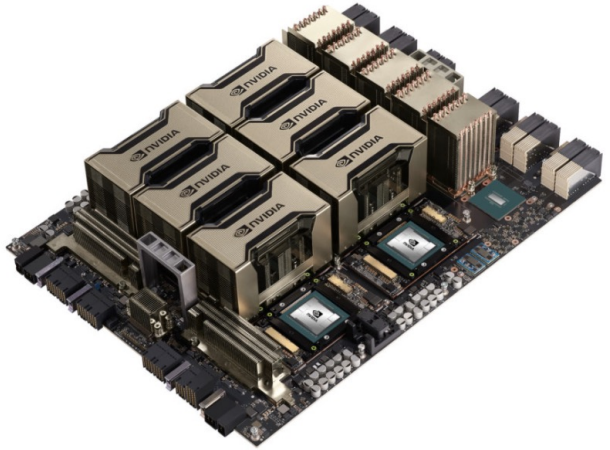
idle (spare) memory

HUVM: benefits

- Better utilization of memory across GPUs
  - Idle (spare) memory now available for other applications
- Latency hiding to/from host memory
  - Lowers performance penalty under memory oversubscription
- High applicability
  - Without modification to applications or frameworks
  - Implemented in the GPU driver layer → easy to use

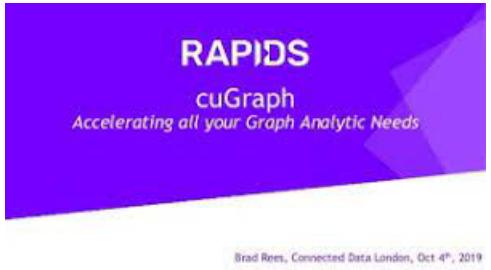
Experimental environments

- Real system: AWS `p3.8xlarge` instance
  - GPU: NVIDIA V100 GPU (16GB) x 4 with NVSwitch
  - CPU: Intel Xeon E5-2685 v4 @ 2.3GHz
  - Host memory: 244GB DDR4
  - OS: Ubuntu 20.04.3
  - GPU driver: Modified NVIDIA driver 460.67 (publicly available soon)



- Benchmarks
  - PyTorch DNN workloads
  - RAPIDS cuGraph workloads

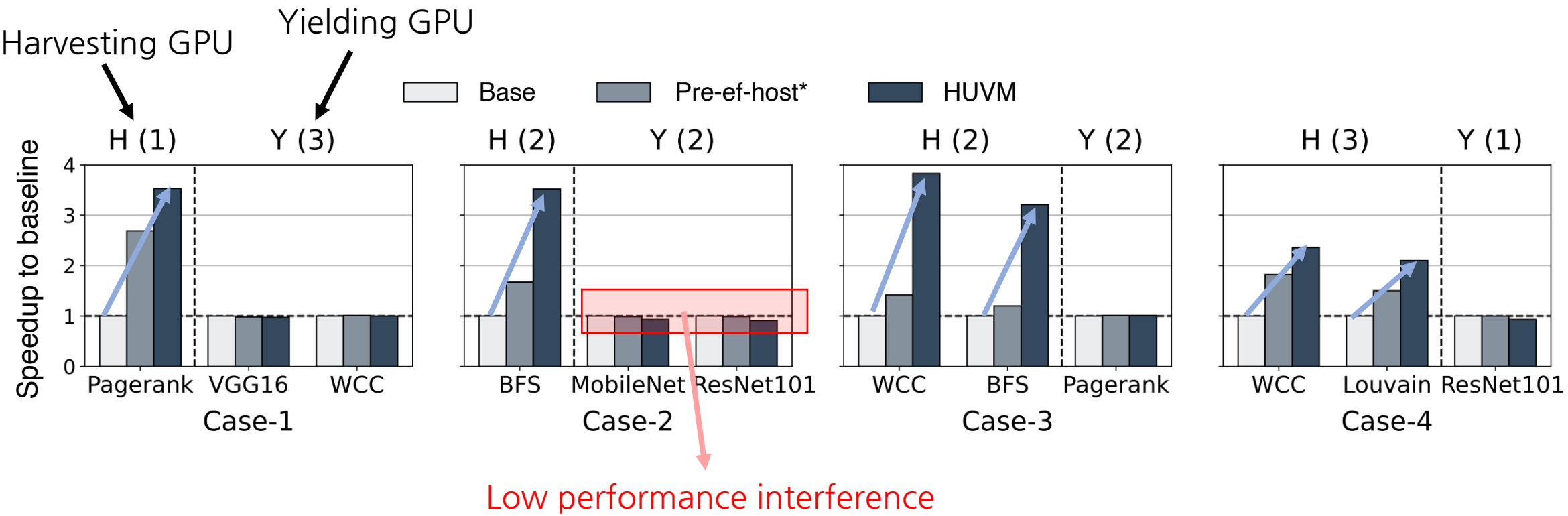
PyTorch



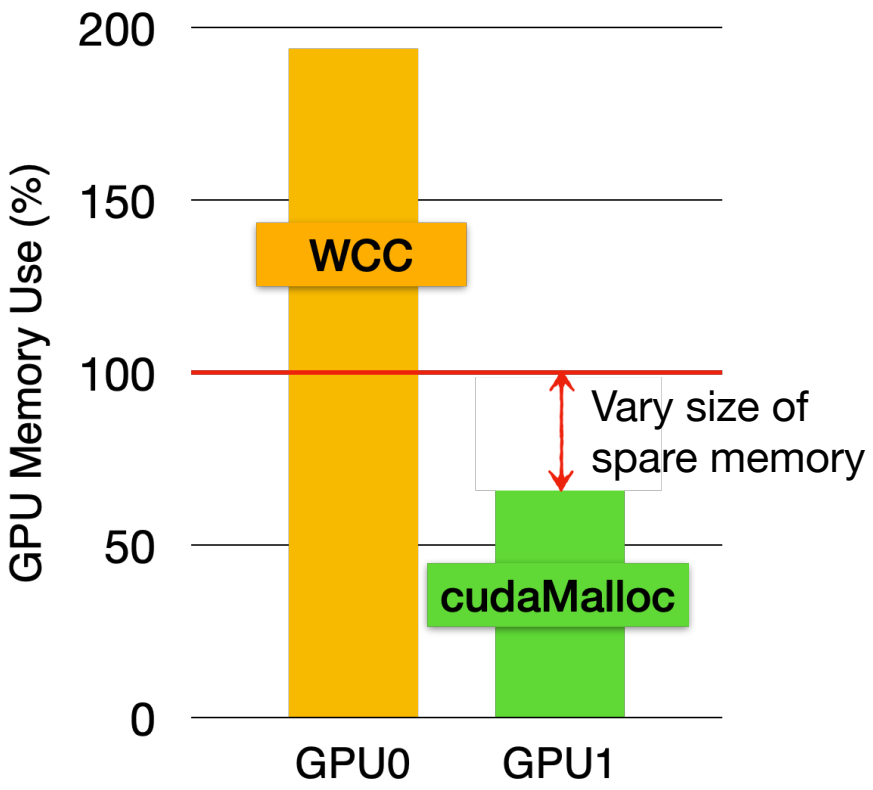
Performance evaluation

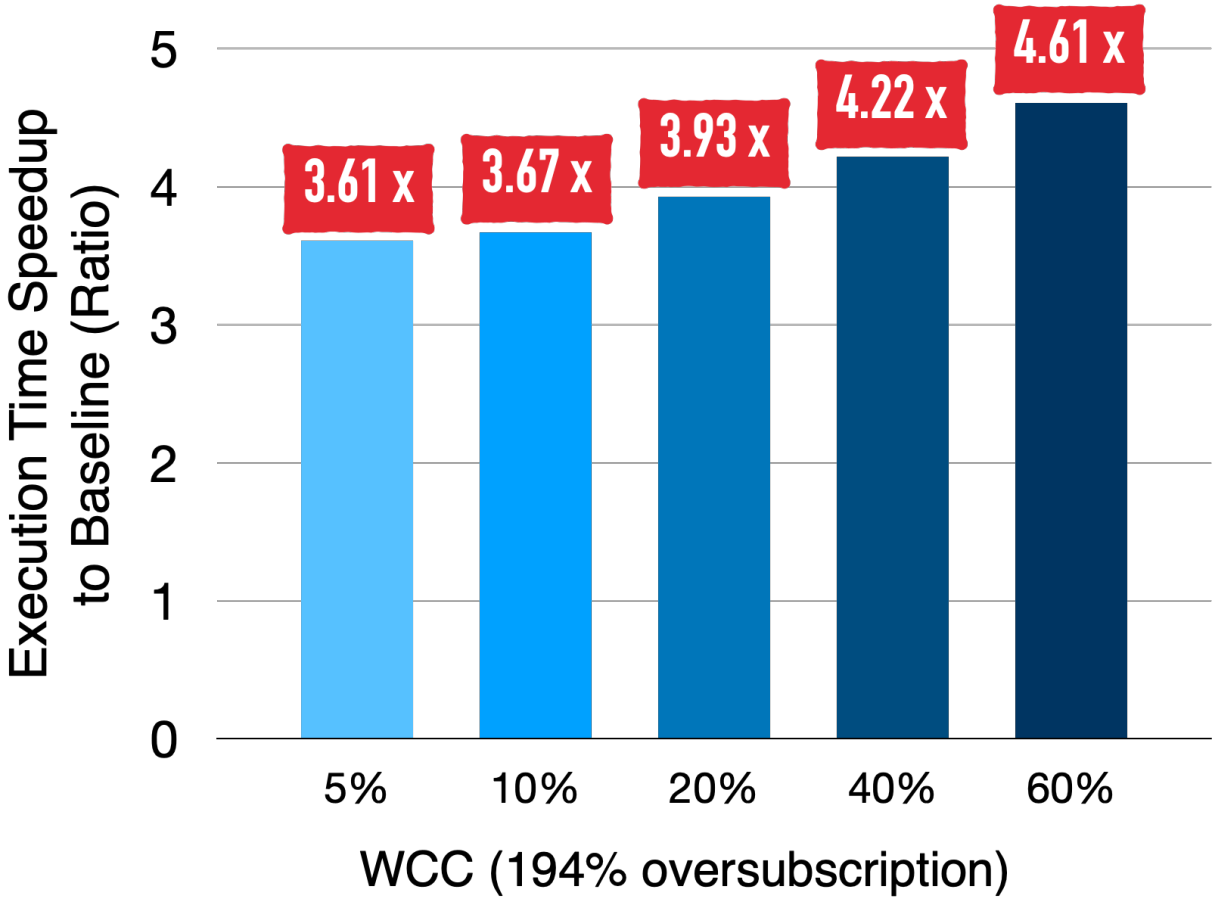
*Pre-ef-host: Pre-eviction & pre-fetch to/from host

Similar to vDNN (MICRO 2016)



Sensitivity to size of spare memory





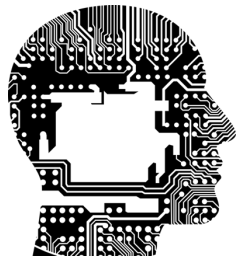
Summary of HUVM

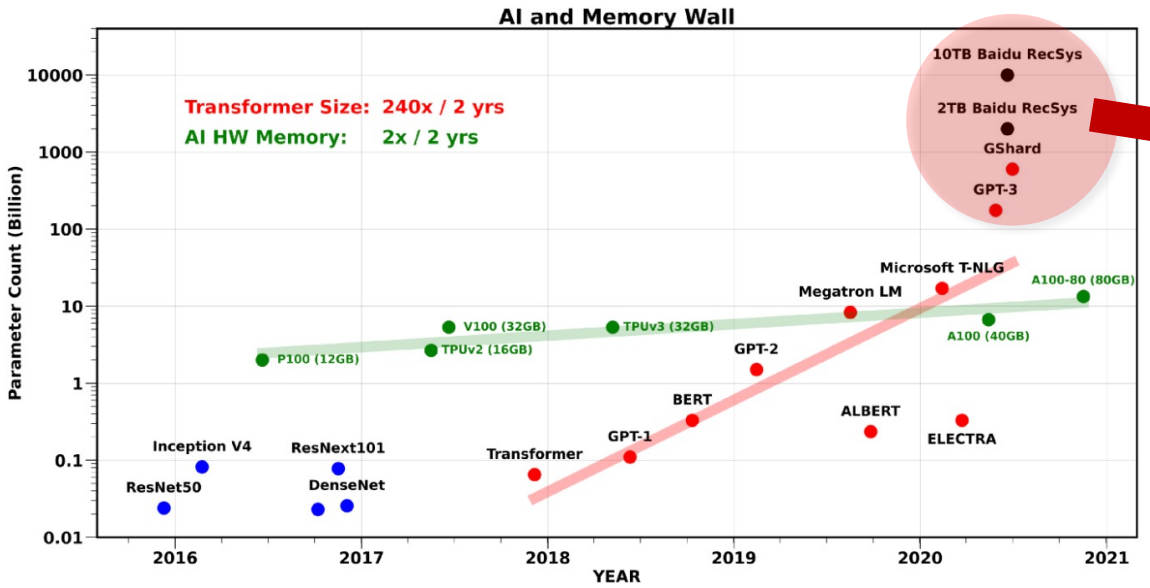
- Current UVM is not sufficient for multi-GPU systems
- HUVM: **make new memory hierarchy elaborate**
  - Harvesting idle memory of neighbor GPUs by leveraging high speed interconnect
- Future work
  - Proactively reclaim the memory which is allocated to a GPU process but not being utilized for a while
    - e.g., DNN training frameworks

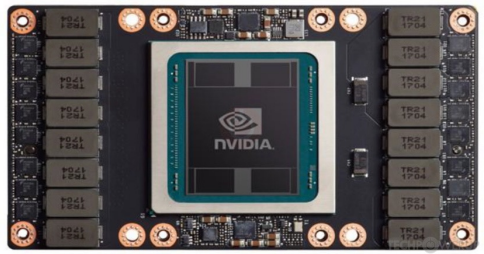
On-going & Future Research

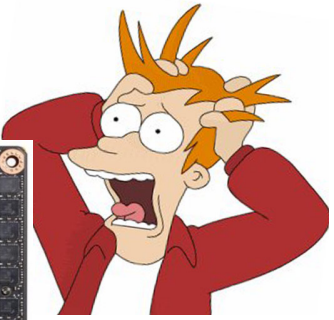
Systems for Machine Learning

- Acceleration of deep learning inferences
- Performance Isolation across multiple DL inferences
- Efficient GPU-to-GPU communication









Cannot fit in a single GPU even for inferencing!

Thank You!

jsahn@ajou.ac.kr

Jeongseob Ahn (안정섭)

AJOU UNIVERSITY