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

Sangjin Choi*, Taeksoo Kim*, Jinwoo Jeong, Rachata Ausavarungnirun,
Myeongjae Jeon, Youngjin Kwon, Jeongseob Ahn

KAIST

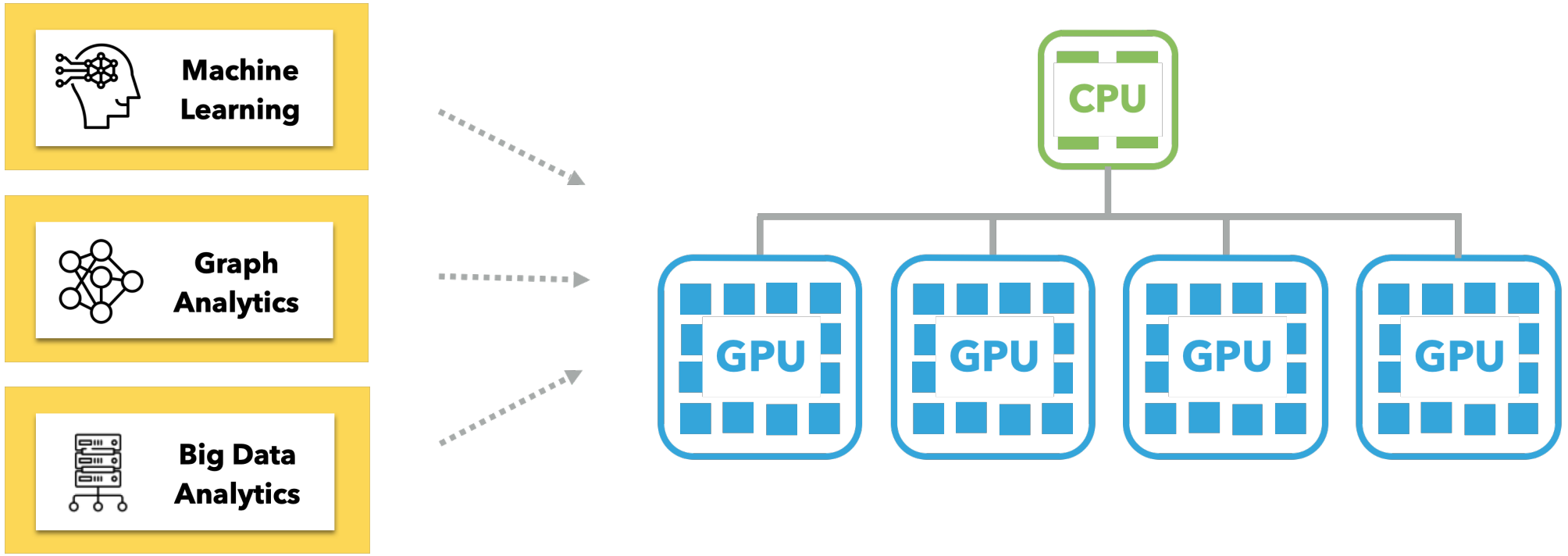


UNIST

*Co-first author

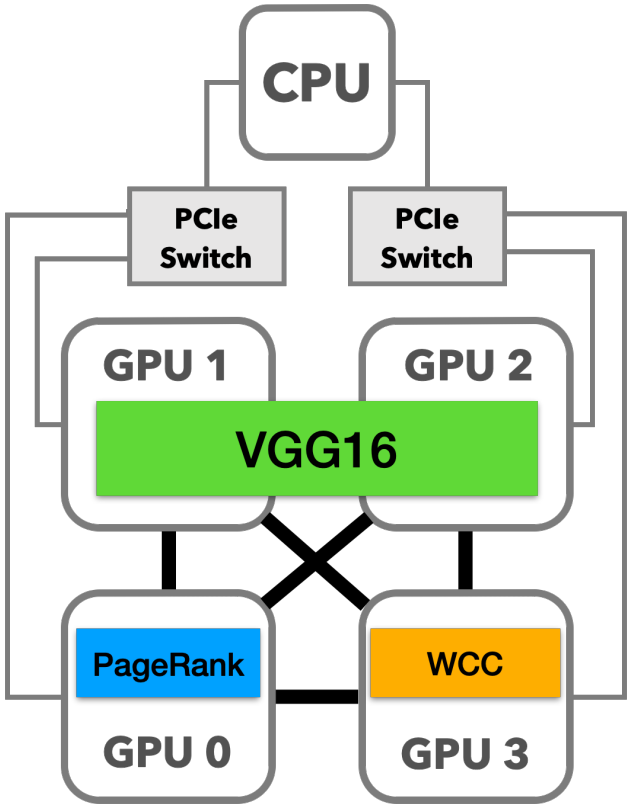
# Shared multi-GPU environment

- Shared multi-GPU servers to reduce the cost of infrastructure

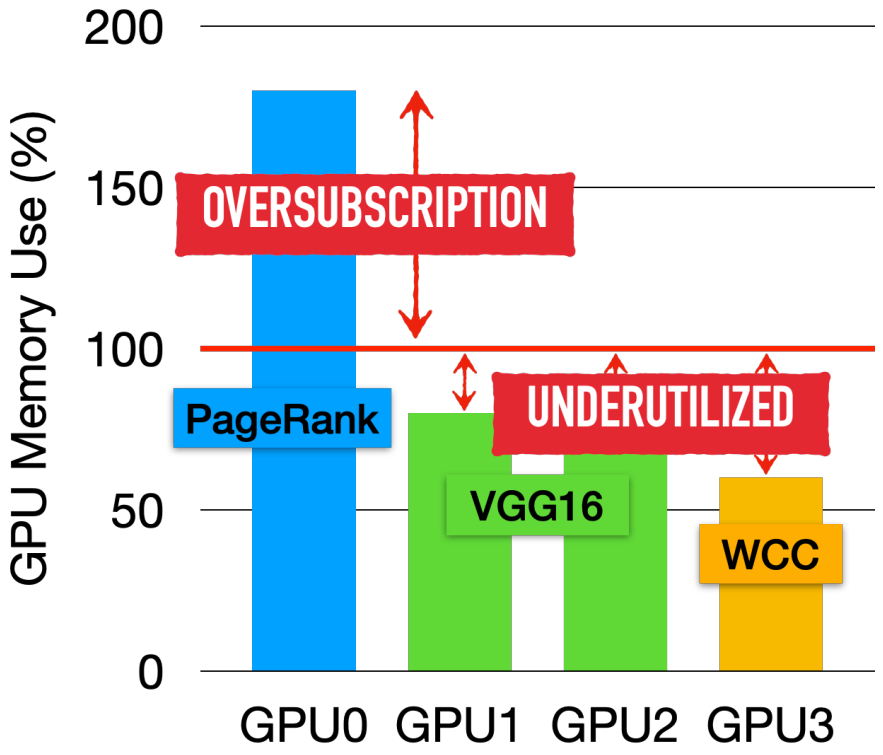



**Workloads are consolidated to improve resource utilization**

# Memory space across GPUs is not fully utilized

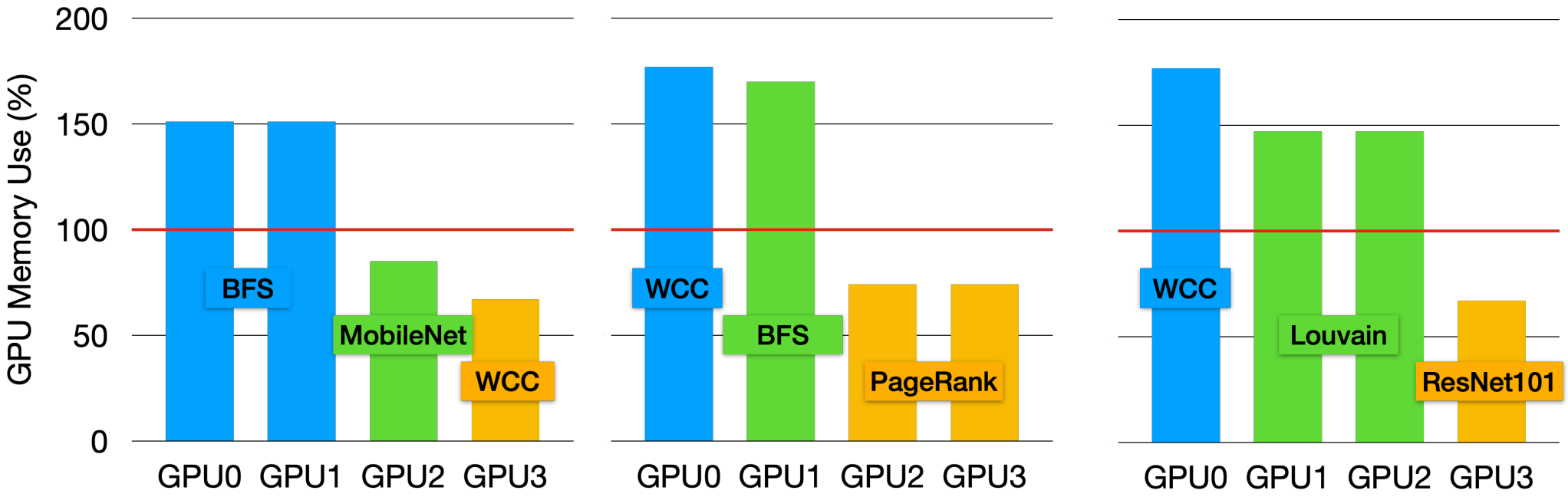



AWS p3.8xlarge instance



# Memory space across GPUs is not fully utilized

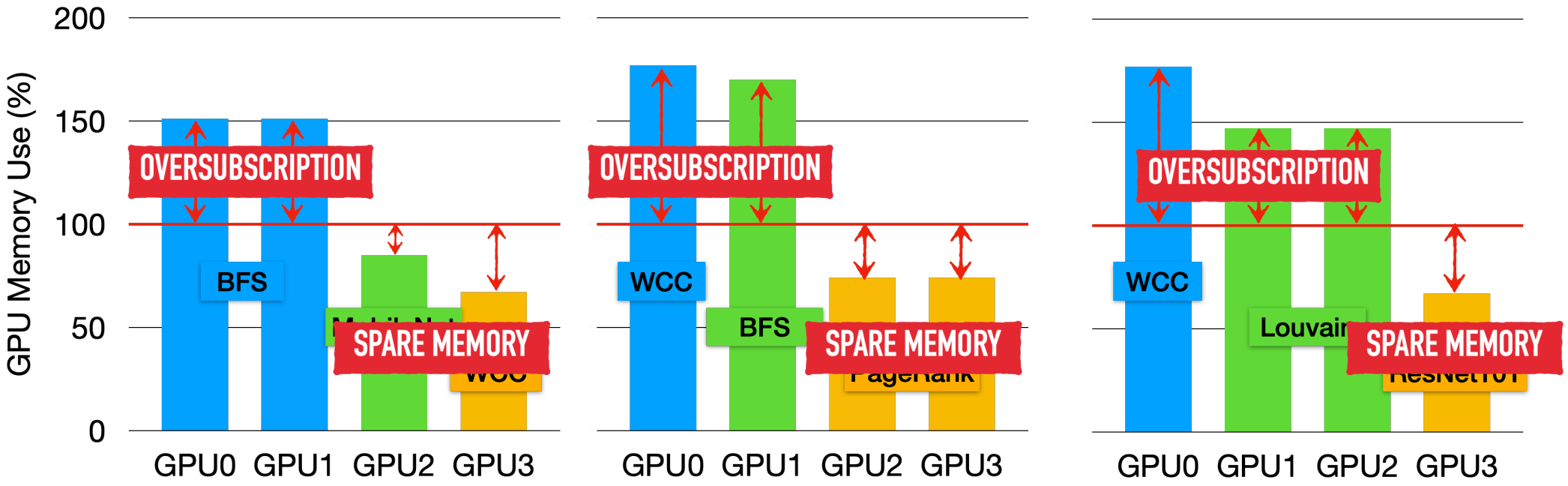
- Each workload has highly varying memory demands
- GPU memory is not fully utilized in shared GPU clusters (e.g., Microsoft GPU cluster[1])




[1] Jeon el al., Multi-tenant GPU Clusters for Deep Learning Workloads: Analysis and Implications. Technical Report MSR-TR-2018-13.

# Memory space across GPUs is not fully utilized

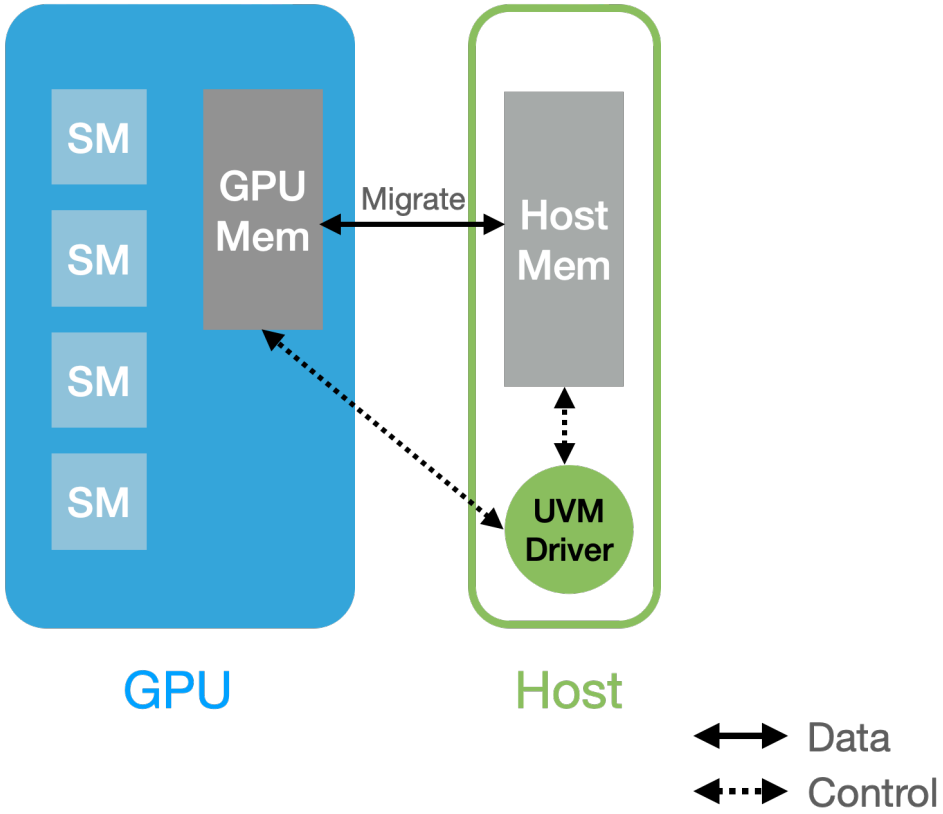
- Each workload has highly varying memory demands
- GPU memory is not fully utilized in shared GPU clusters (e.g., Microsoft GPU cluster[1])




[1] Jeon el al., Multi-tenant GPU Clusters for Deep Learning Workloads: Analysis and Implications. Technical Report MSR-TR-2018-13.

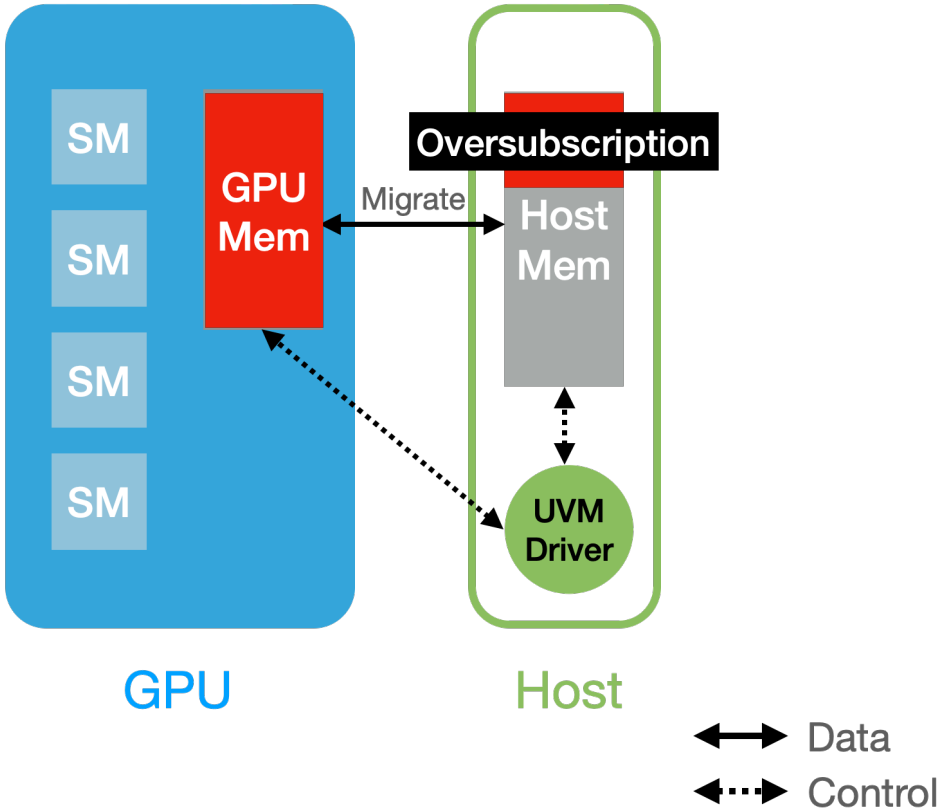
# Performance overhead for memory oversubscription

- Unified Virtual Memory (UVM)
  - Swapping GPU memory to host memory



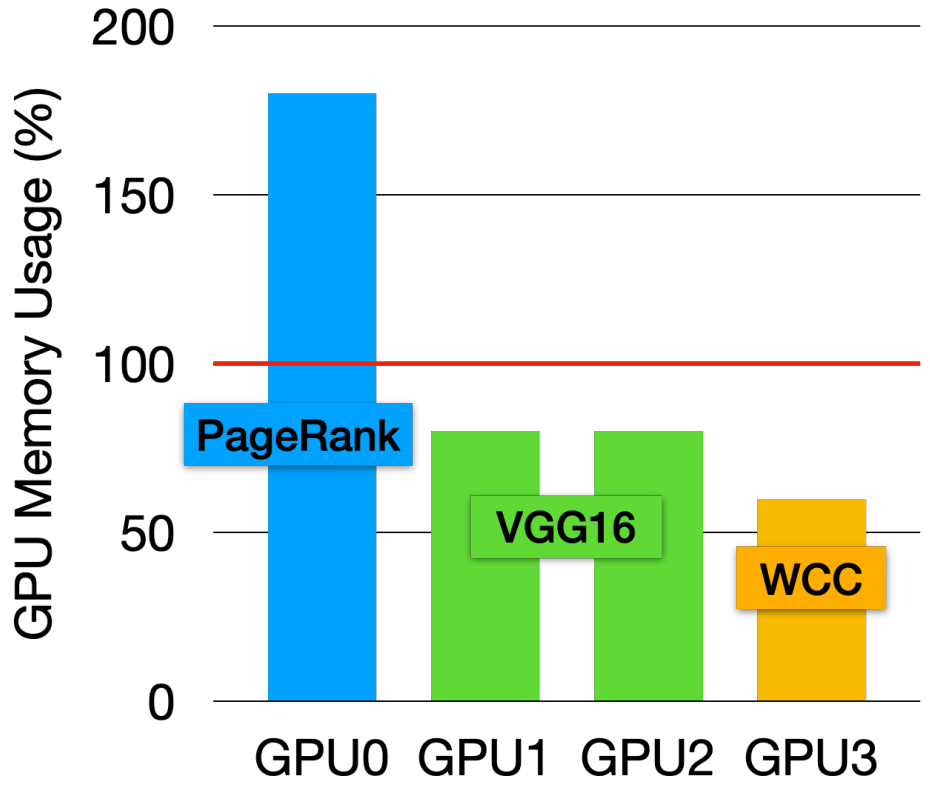
# Performance overhead for memory oversubscription

- Unified Virtual Memory (UVM)
  - Swapping GPU memory to host memory
- Cost for memory oversubscription is high
  - **2x ~ 64x** longer to complete with 40% oversubcription[1]

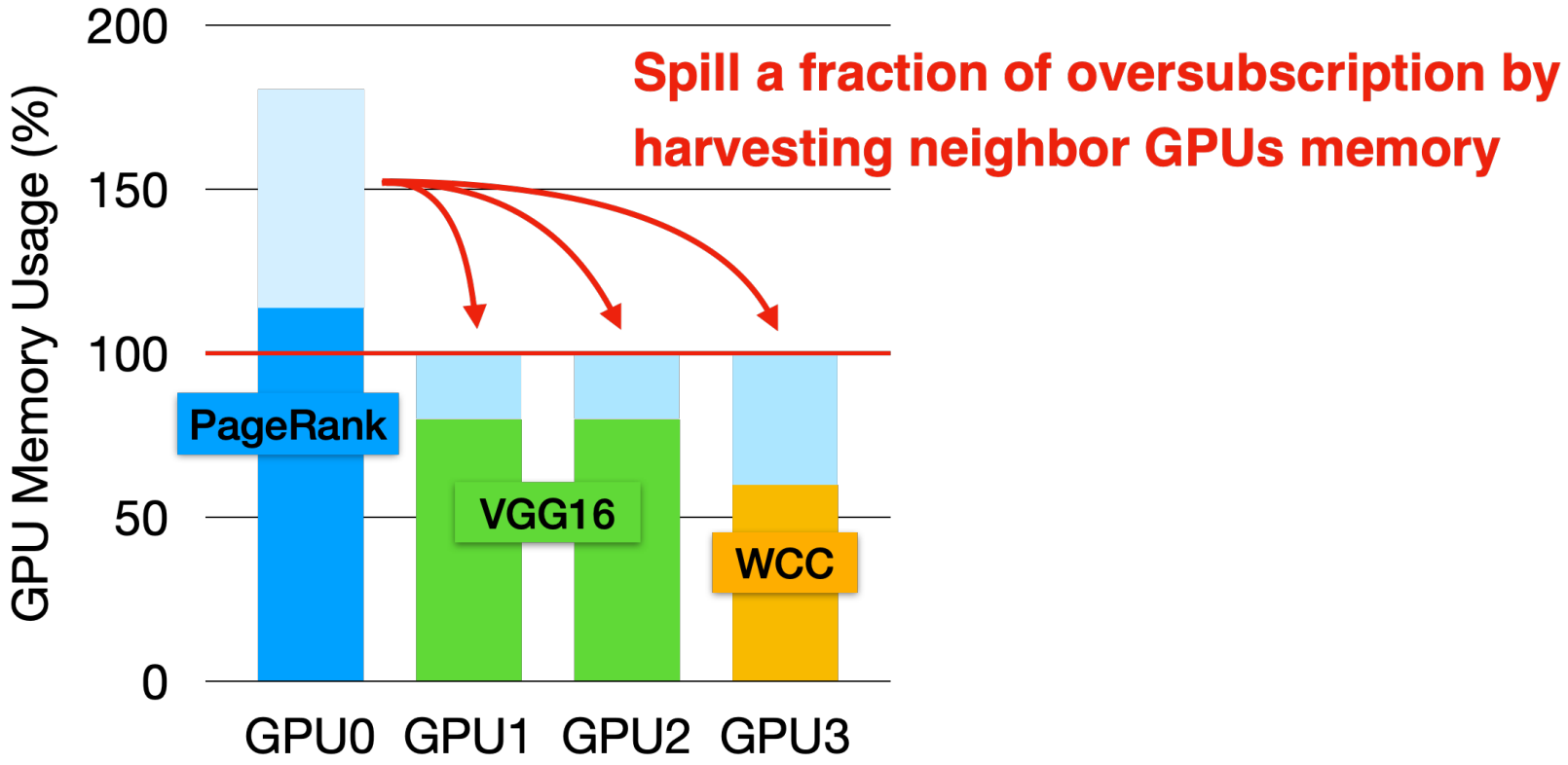



[1] E. Choukse et al., "Buddy Compression: Enabling Larger Memory for Deep Learning and HPC Workloads on GPUs," 2020 ACM/IEEE 47th Annual International Symposium on Computer Architecture (ISCA), 2020, pp. 926-939, doi: 10.1109/ISCA45697.2020.00080.

# Our approach: Harvesting neighbor GPU memory

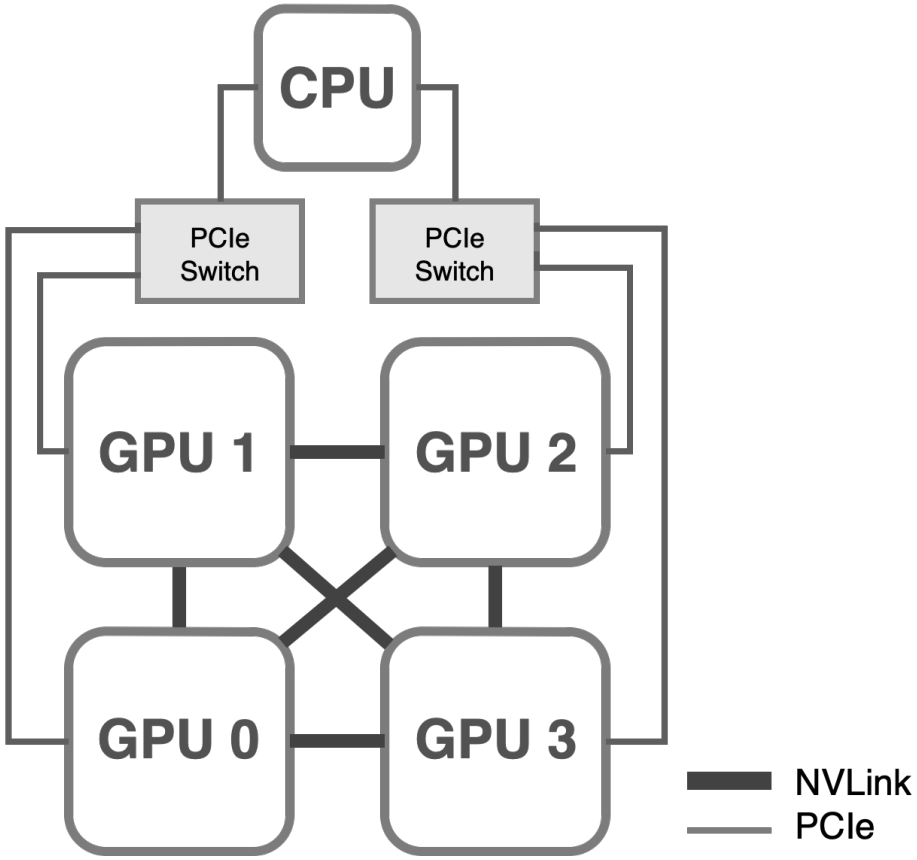


# Our approach: Harvesting neighbor GPU memory




**How to efficiently access neighbor GPU memory?**

# NVLink provides fast interconnect to neighbor GPU

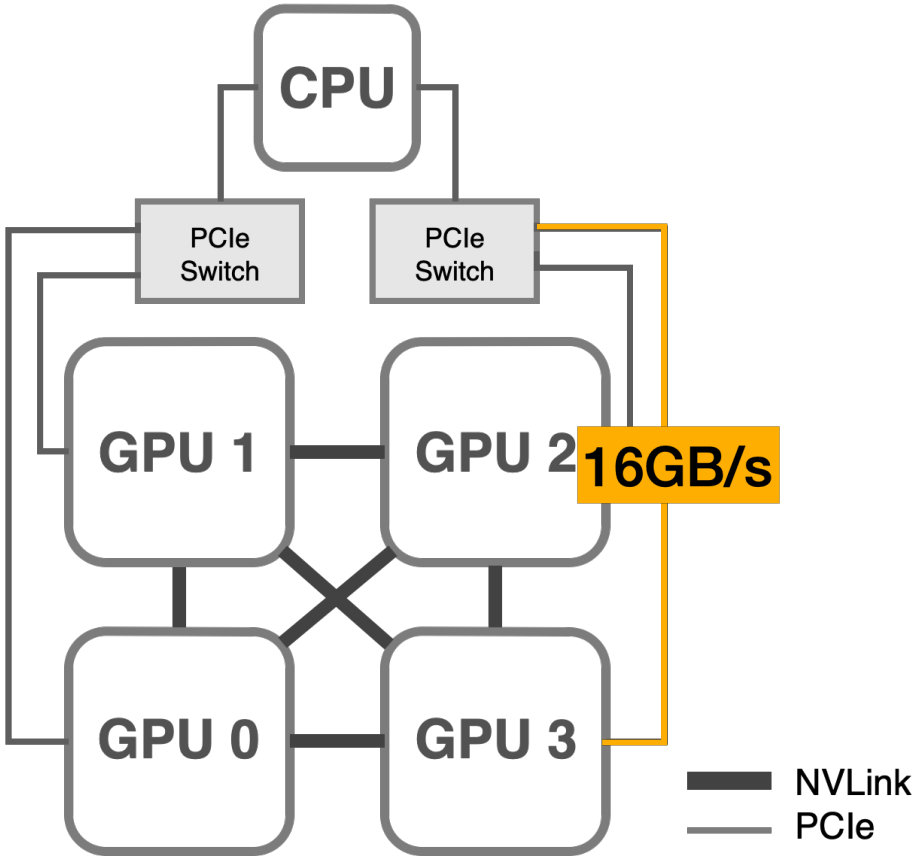



| | PCIe | NVLink |
|---|---|---|
| Throughput (GB/s) | 12.3 | 40.1 (3.3x) |
| Latency (µs) | 16.7 | 5.1 (3.2x) |

2MB Copy

NVLink 2.0 x 2
PCIe 3.0 x 16

# NVLink provides fast interconnect to neighbor GPU

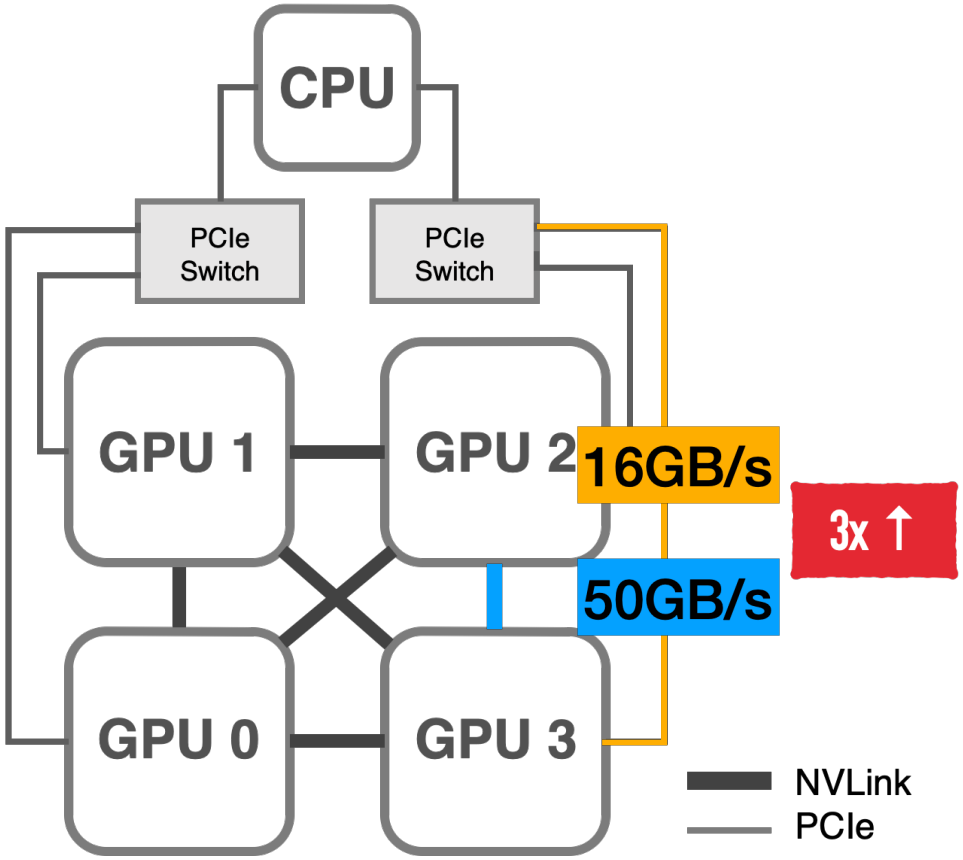



| | PCIe | NVLink |
|---|---|---|
| Throughput (GB/s) | 12.3 | 40.1 (3.3x) |
| Latency (µs) | 16.7 | 5.1 (3.2x) |

2MB Copy

NVLink 2.0 x 2
PCIe 3.0 x 16

# NVLink provides fast interconnect to neighbor GPU

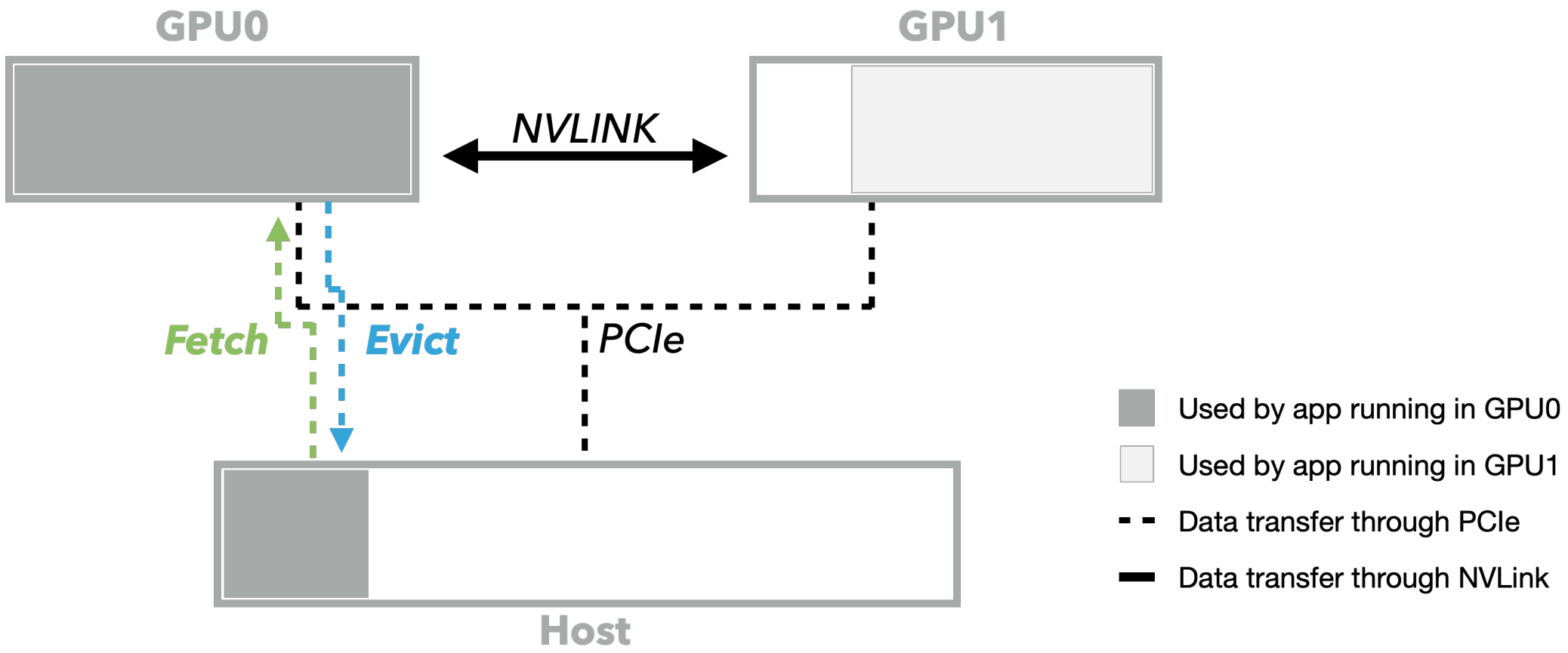



| | PCIe | NVLink |
|---|---|---|
| Throughput (GB/s) | 12.3 | 40.1 (3.3x) |
| Latency (µs) | 16.7 | 5.1 (3.2x) |

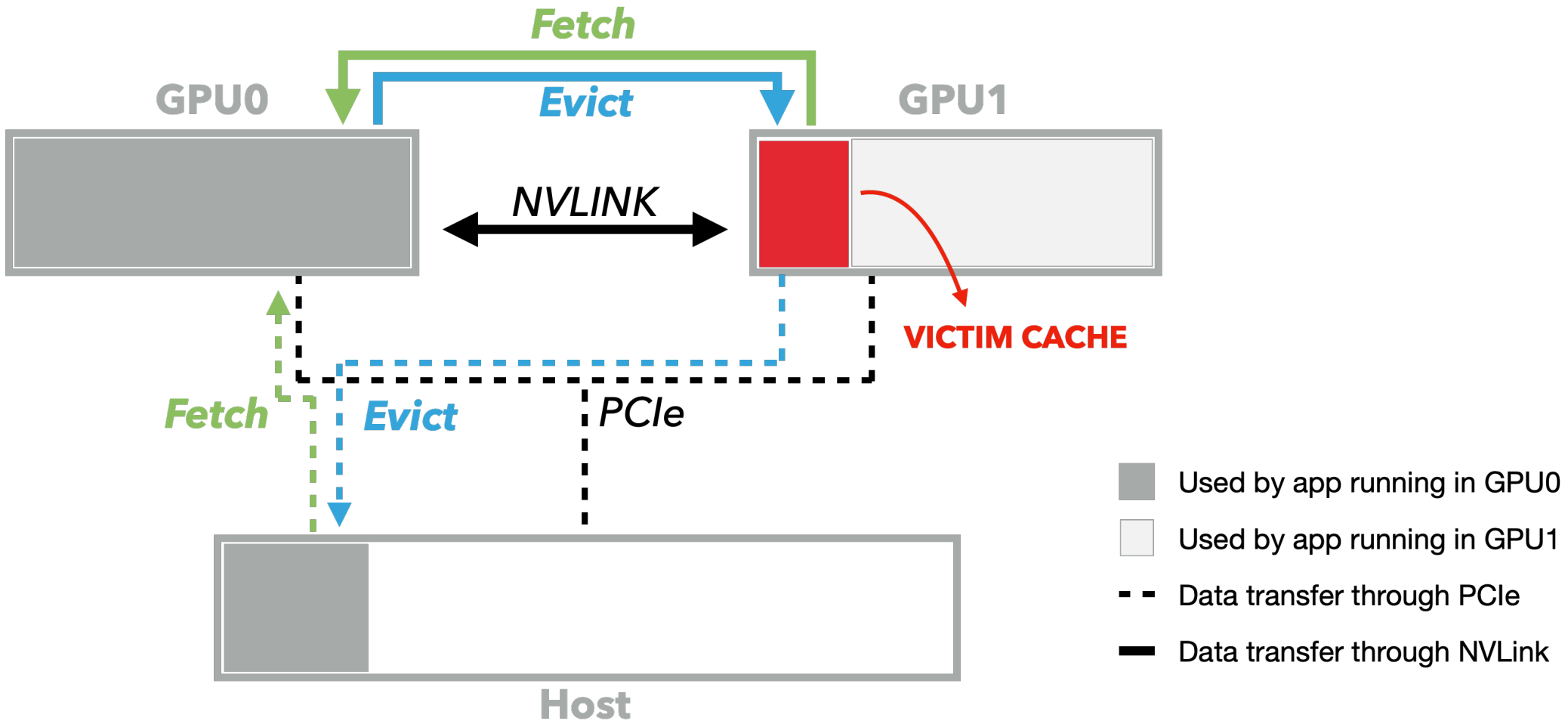
2MB Copy

NVLink 2.0 x 2
PCIe 3.0 x 16

# Unified Virtual Memory



# Data-path: Hierarchical Unified Virtual Memory



# Goals of Hierarchical Unified Virtual Memory

Reducing performance cost of memory oversubscription

| | |
|---|---|
| **Effective Harvesting** | Harvest small and temporarily available spare memory of neighbor GPU<br>Reduce eviction/fetch latency with spare memory |
| **Minimal Interference** | Minimize performance impact of workloads running in neighbor GPU |
| **Framework-agnostic** | No modification of applications or frameworks |

# Memory manager for HUVM: memHarvester

Centralized coordinator for data-path in HUVM

1. Hiding eviction latency
2. Reducing fetch latency
3. Hiding fetch latency

Effective Harvesting

Minimal Interference

Framework-agnostic

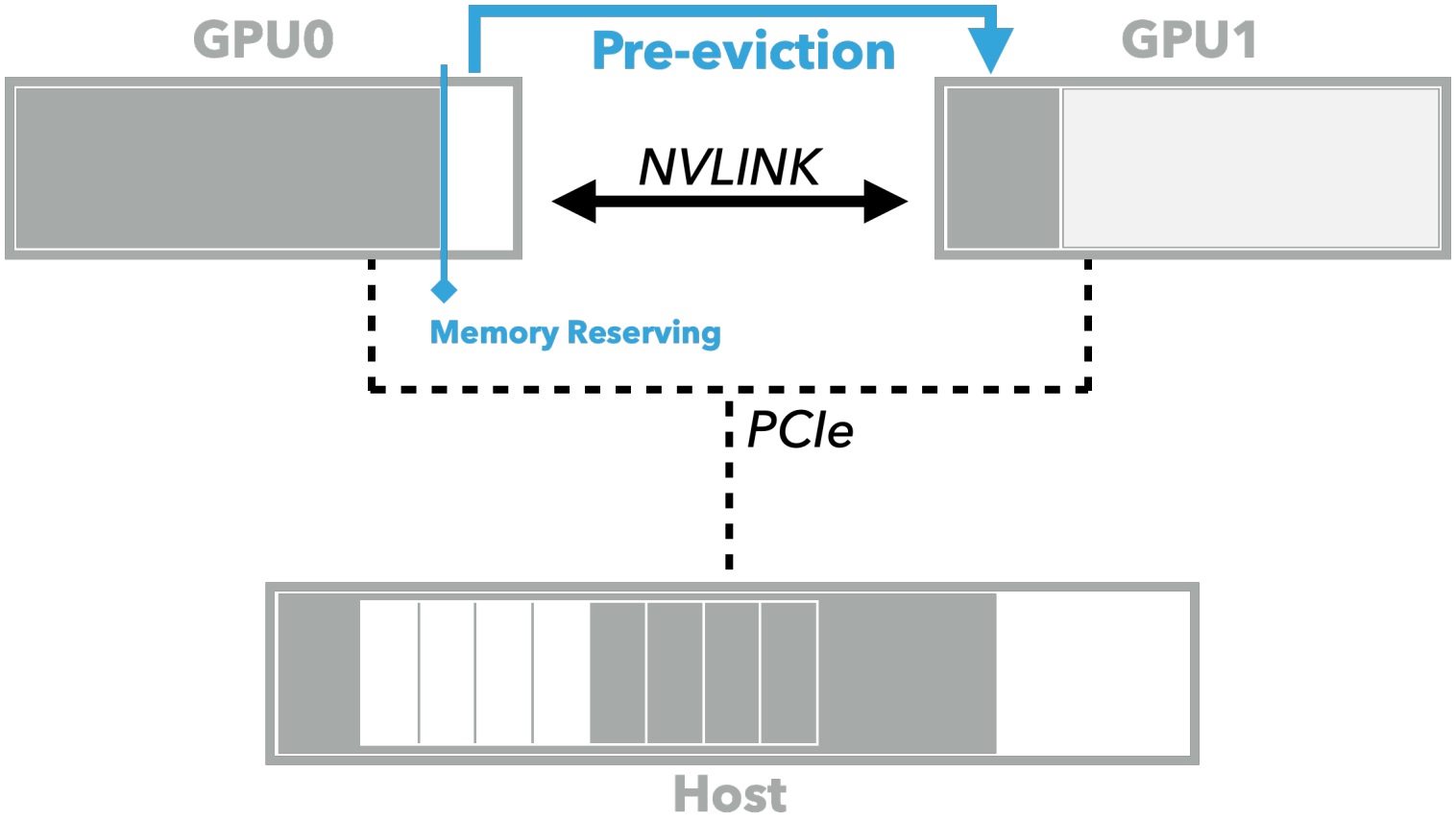
# 1. Hiding eviction latency

2. Reducing fetch latency

3. Hiding fetch latency

- ✓ Effective Harvesting
- Minimal Interference
- ✓ Framework-agnostic

Reduce # of on-demand evictions



# 1. Hiding eviction latency

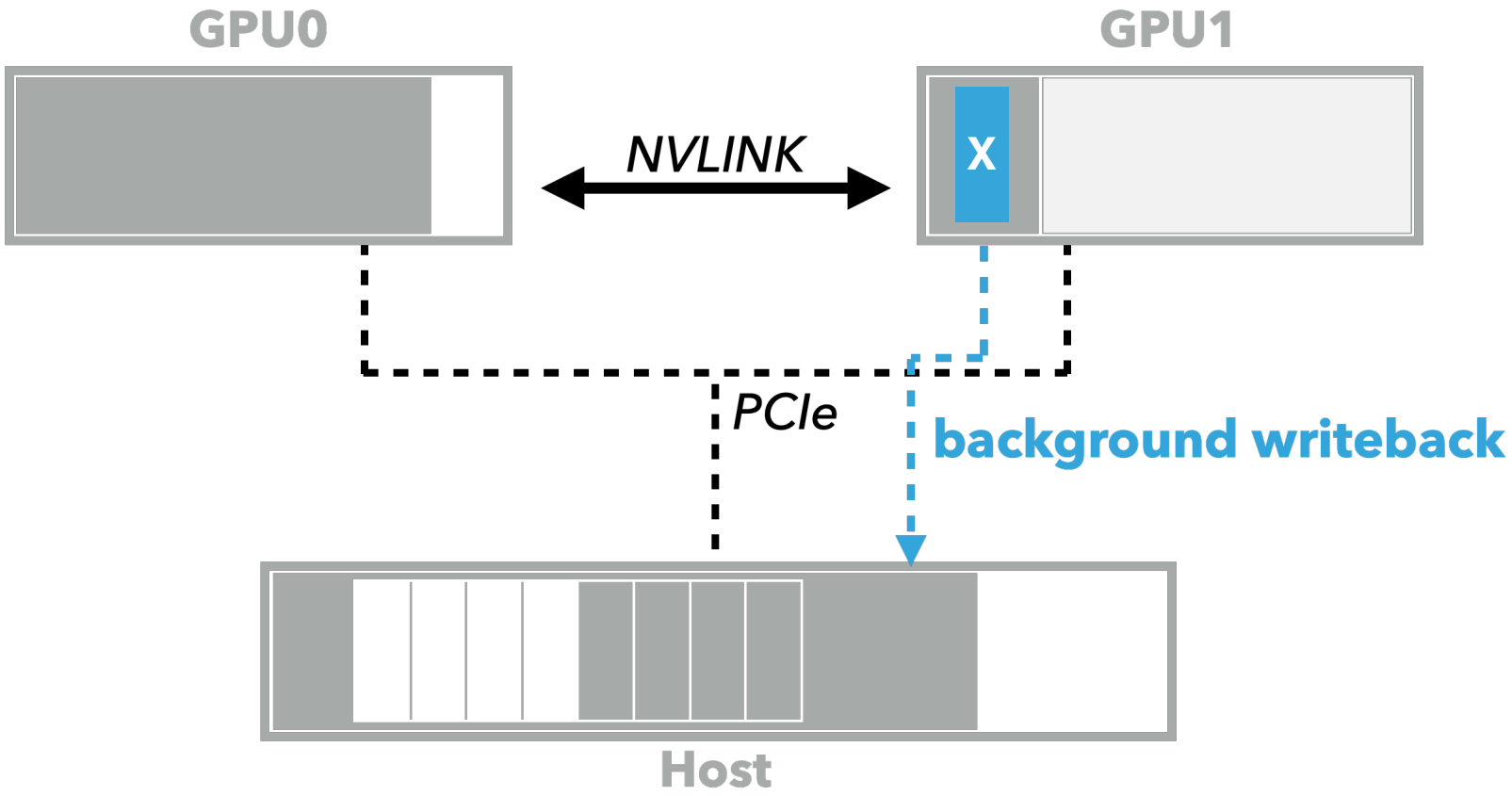
2. Reducing fetch latency

3. Hiding fetch latency

Effective Harvesting

✓ Minimal Interference

✓ Framework-agnostic



# 1. Hiding eviction latency

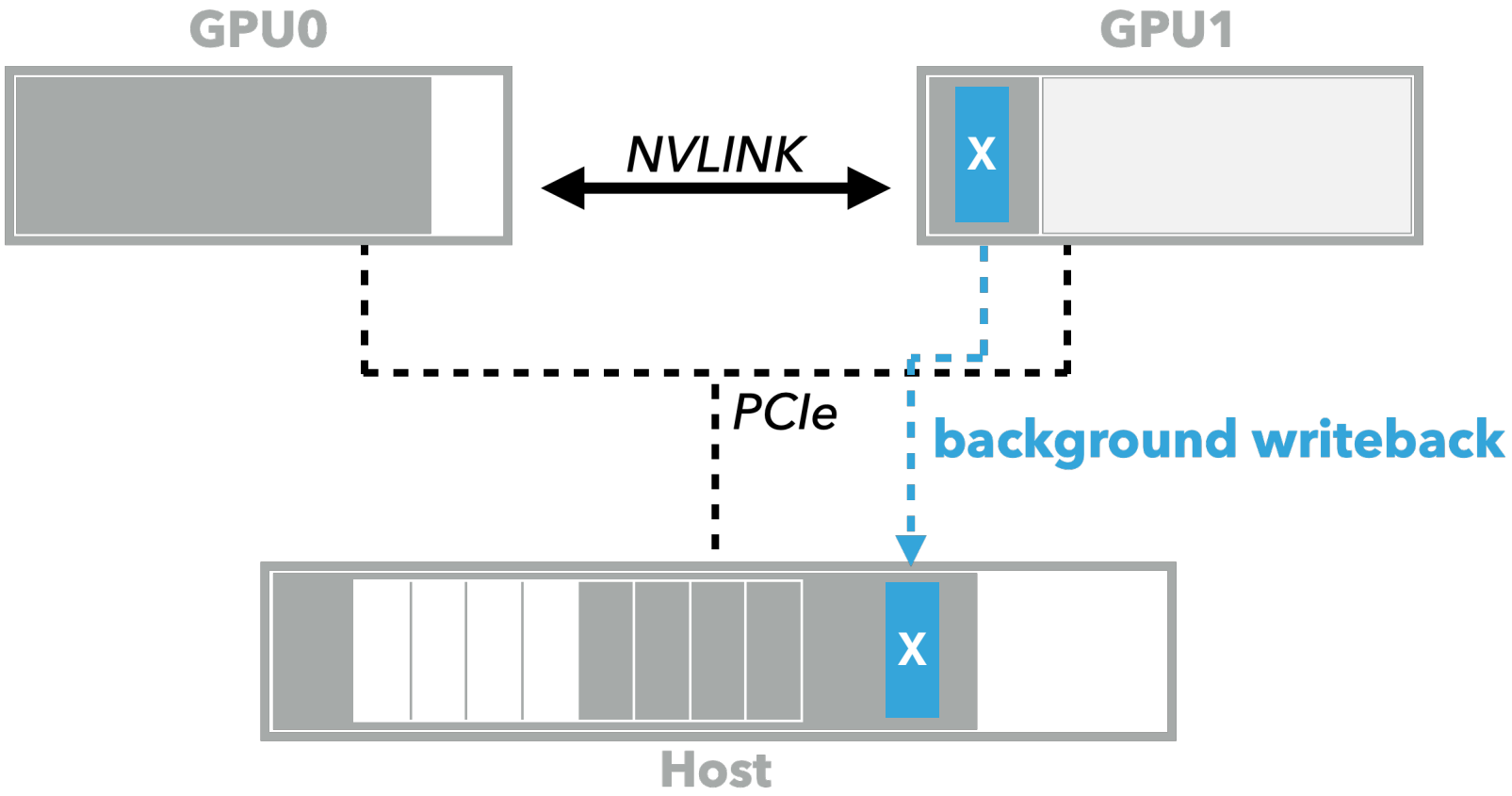
2. Reducing fetch latency

3. Hiding fetch latency

Effective Harvesting

✓ Minimal Interference

✓ Framework-agnostic



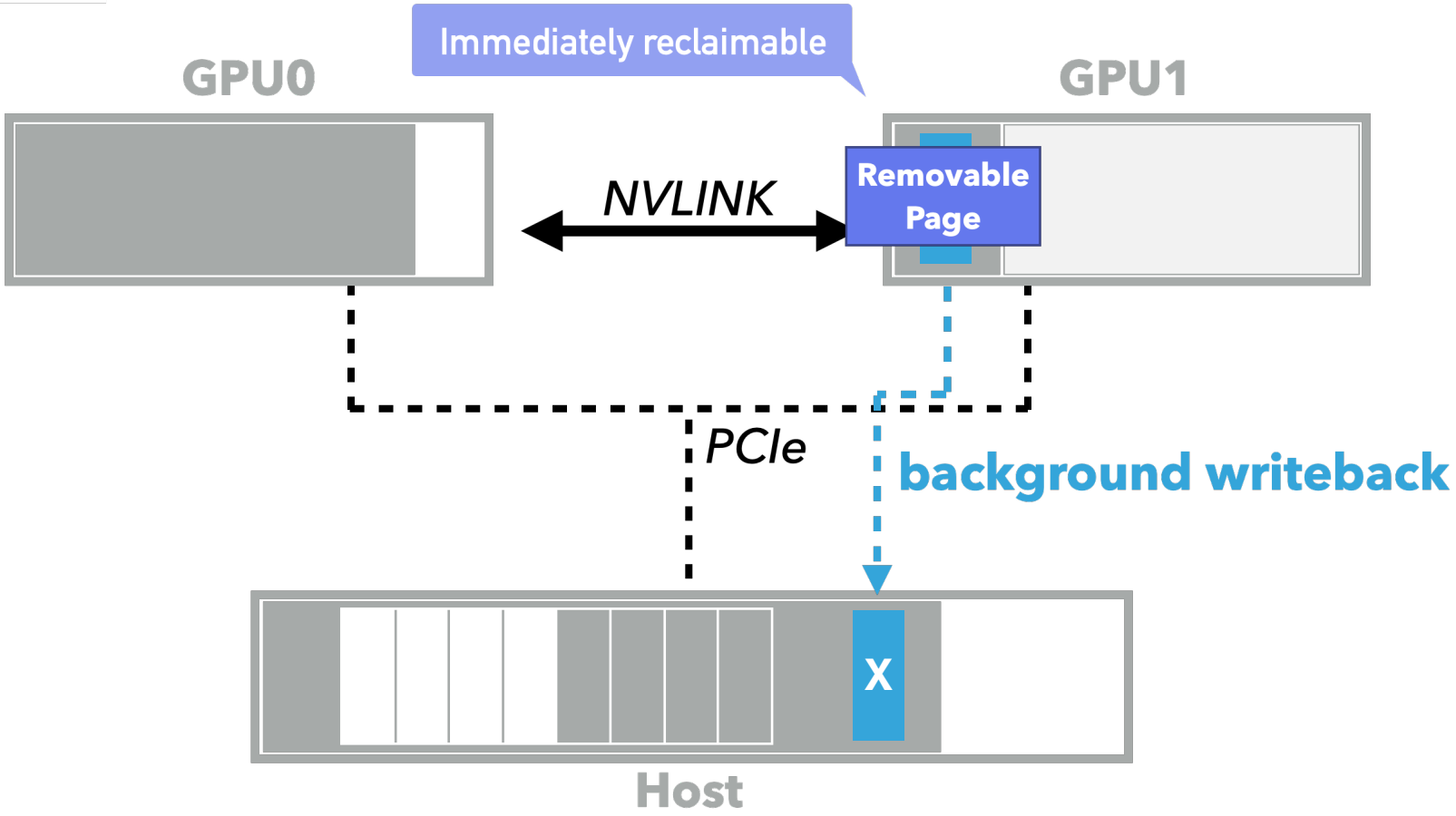
# 1. Hiding eviction latency

2. Reducing fetch latency

3. Hiding fetch latency

Effective Harvesting

- Minimal Interference
- Framework-agnostic



# 1. Hiding eviction latency

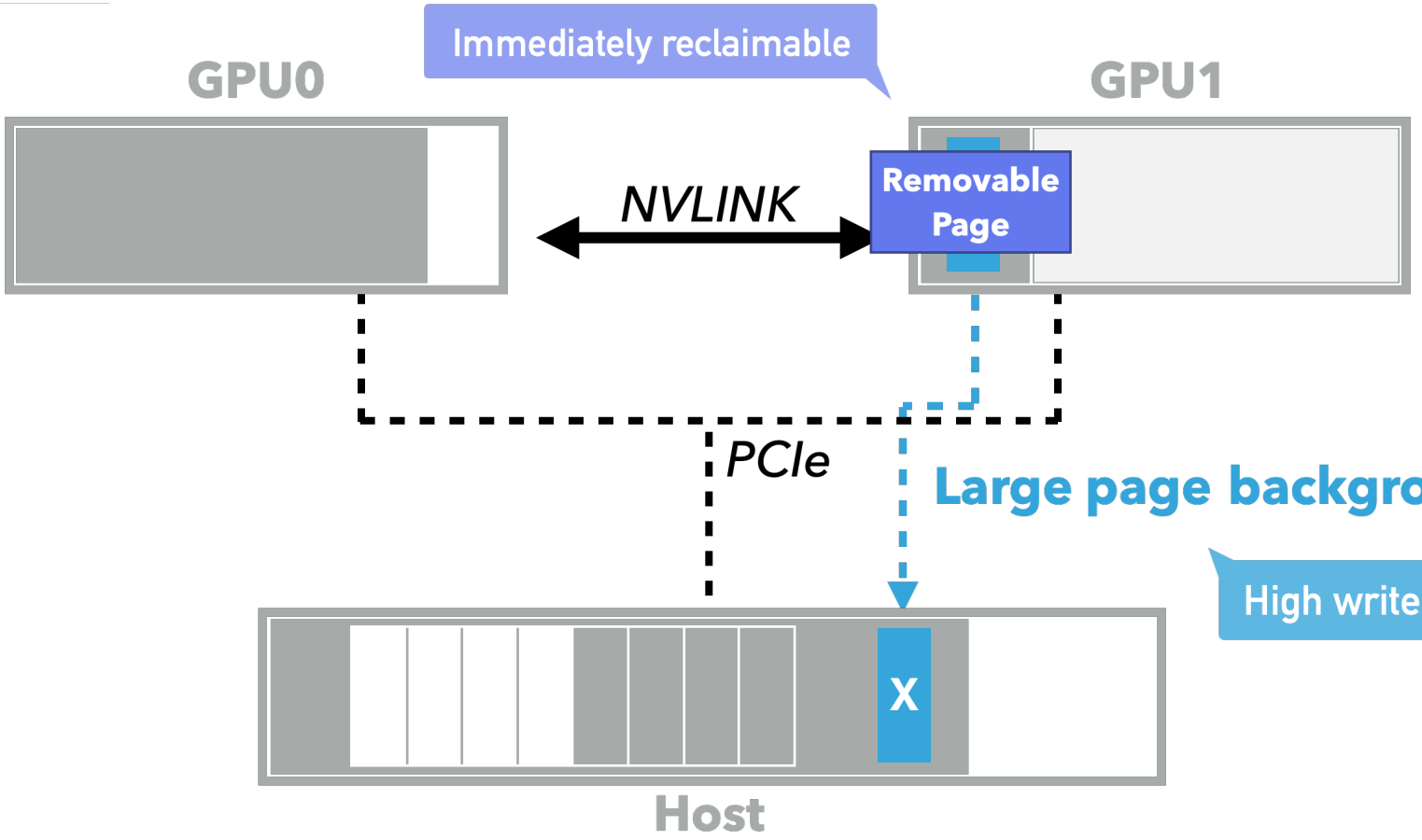
2. Reducing fetch latency

3. Hiding fetch latency

Effective Harvesting

✓ Minimal Interference

✓ Framework-agnostic

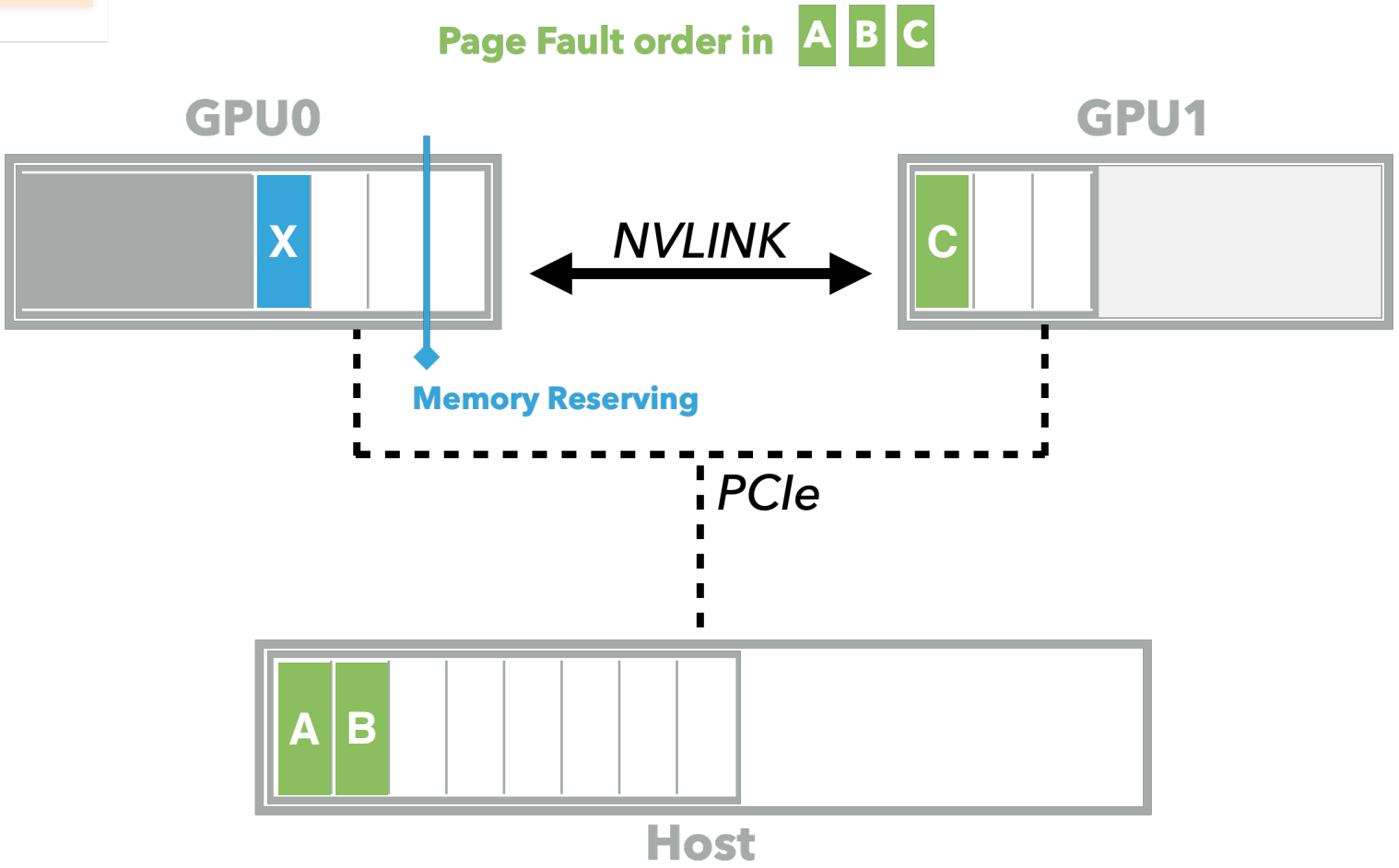


# 1. Hiding eviction latency

2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

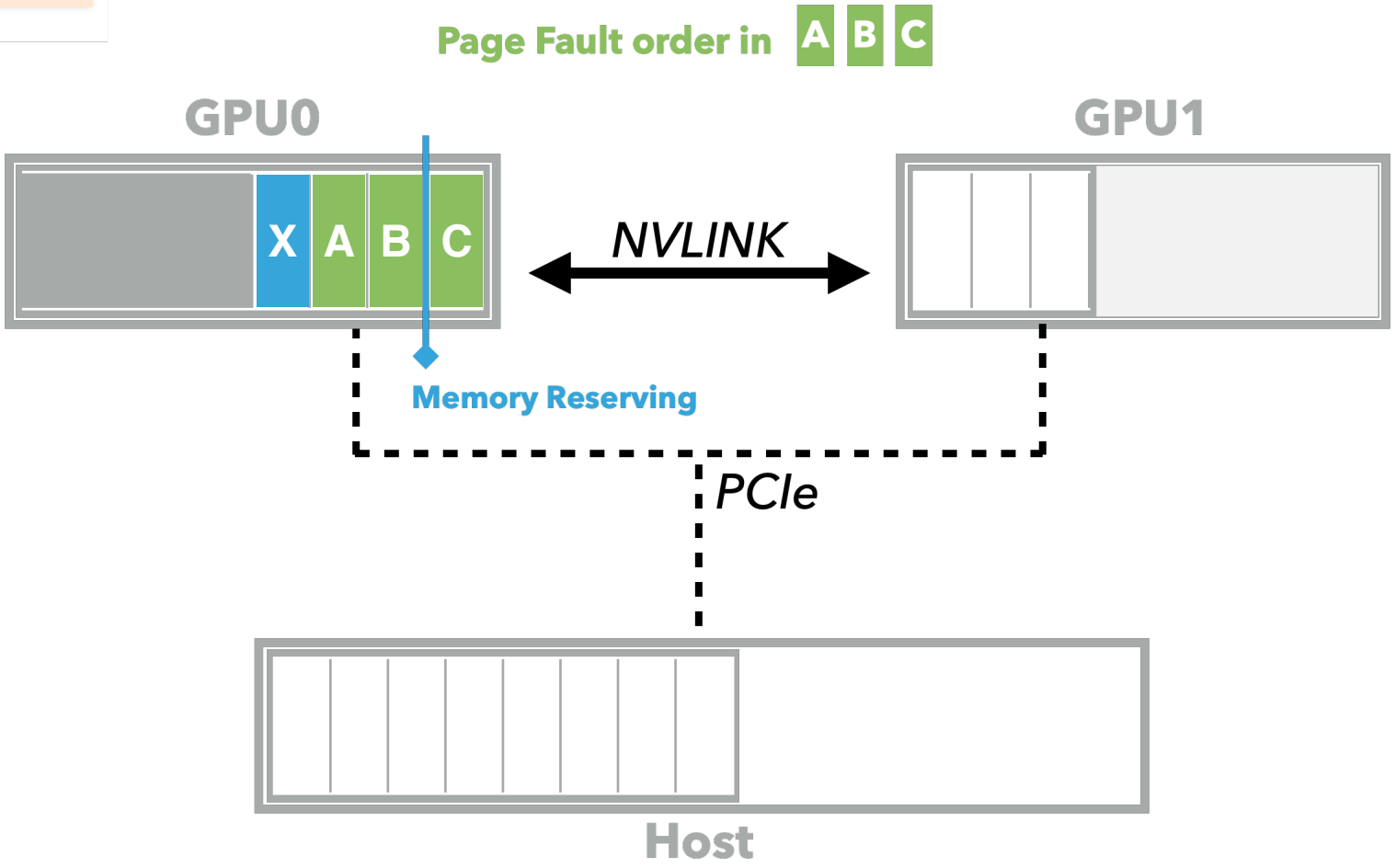


# 1. Hiding eviction latency

2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

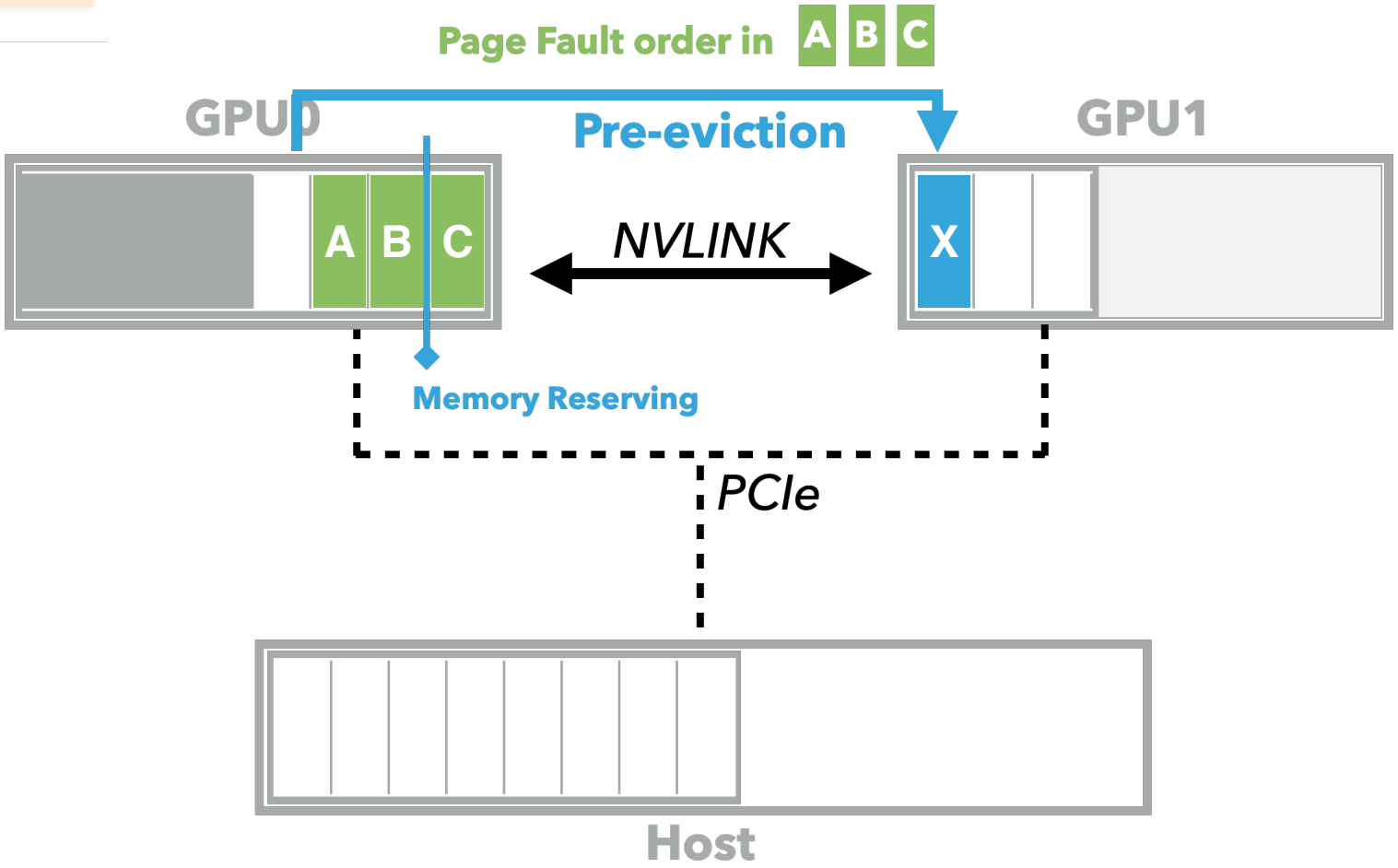


# 1. Hiding eviction latency

2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

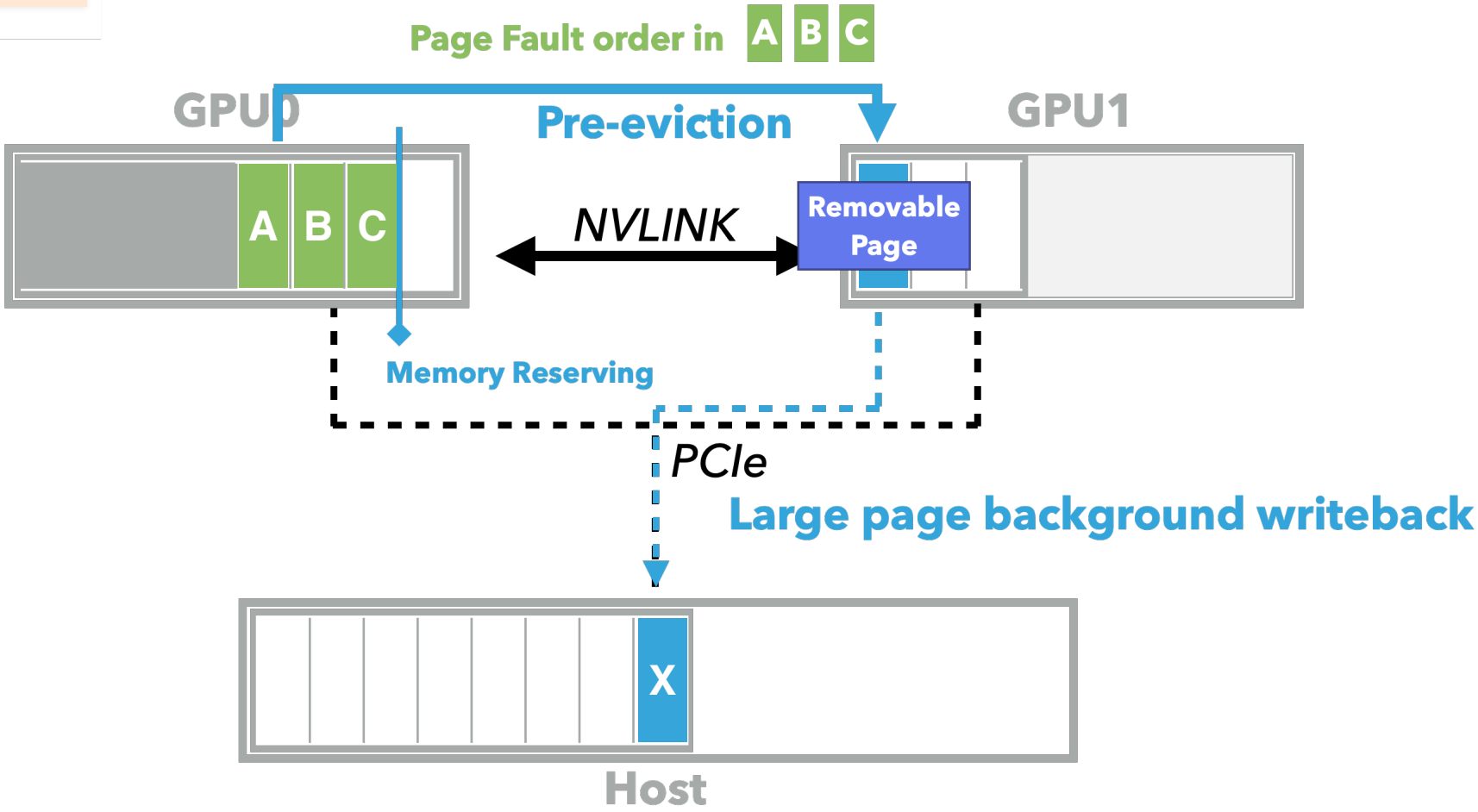


# 1. Hiding eviction latency

2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

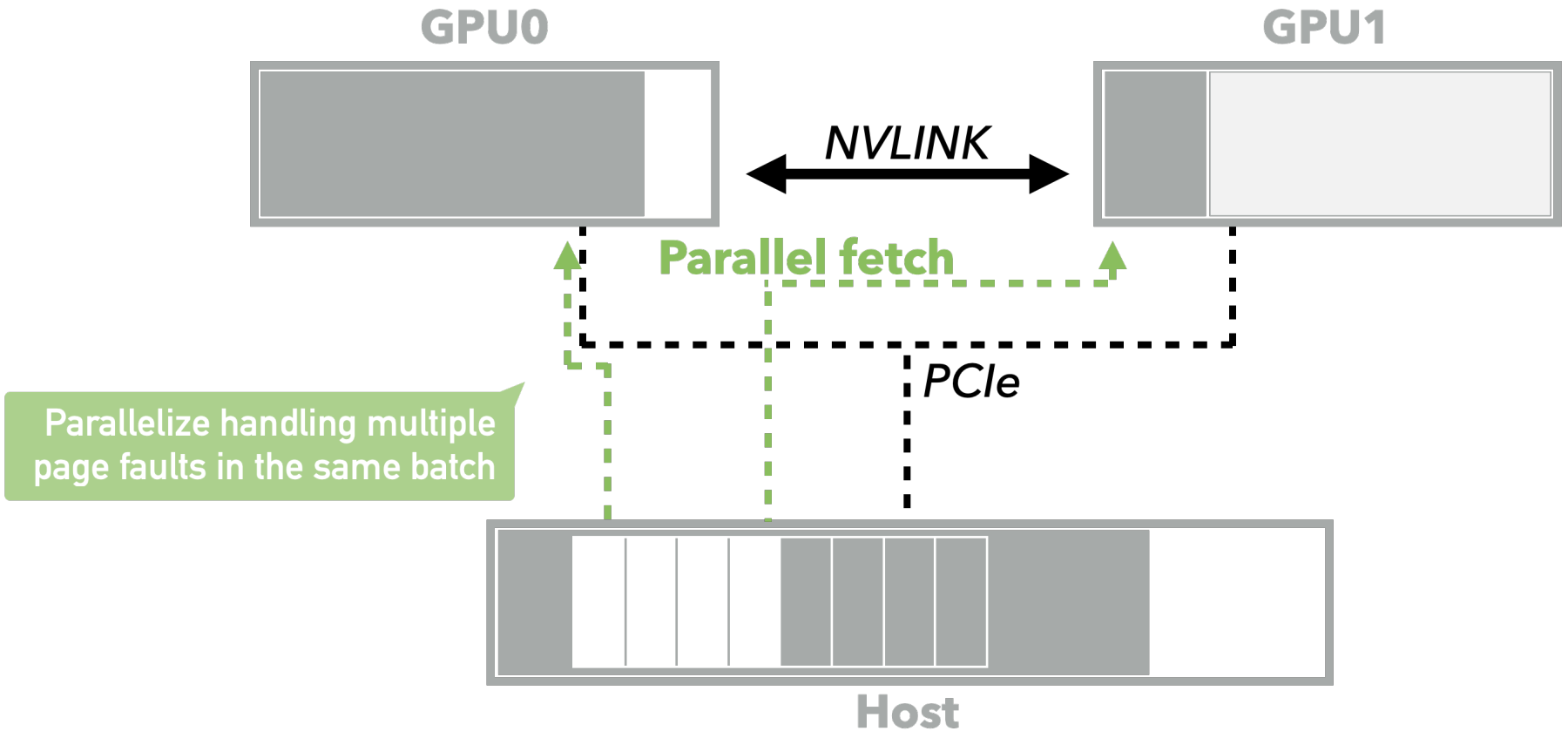


1. Hiding eviction latency

# 2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

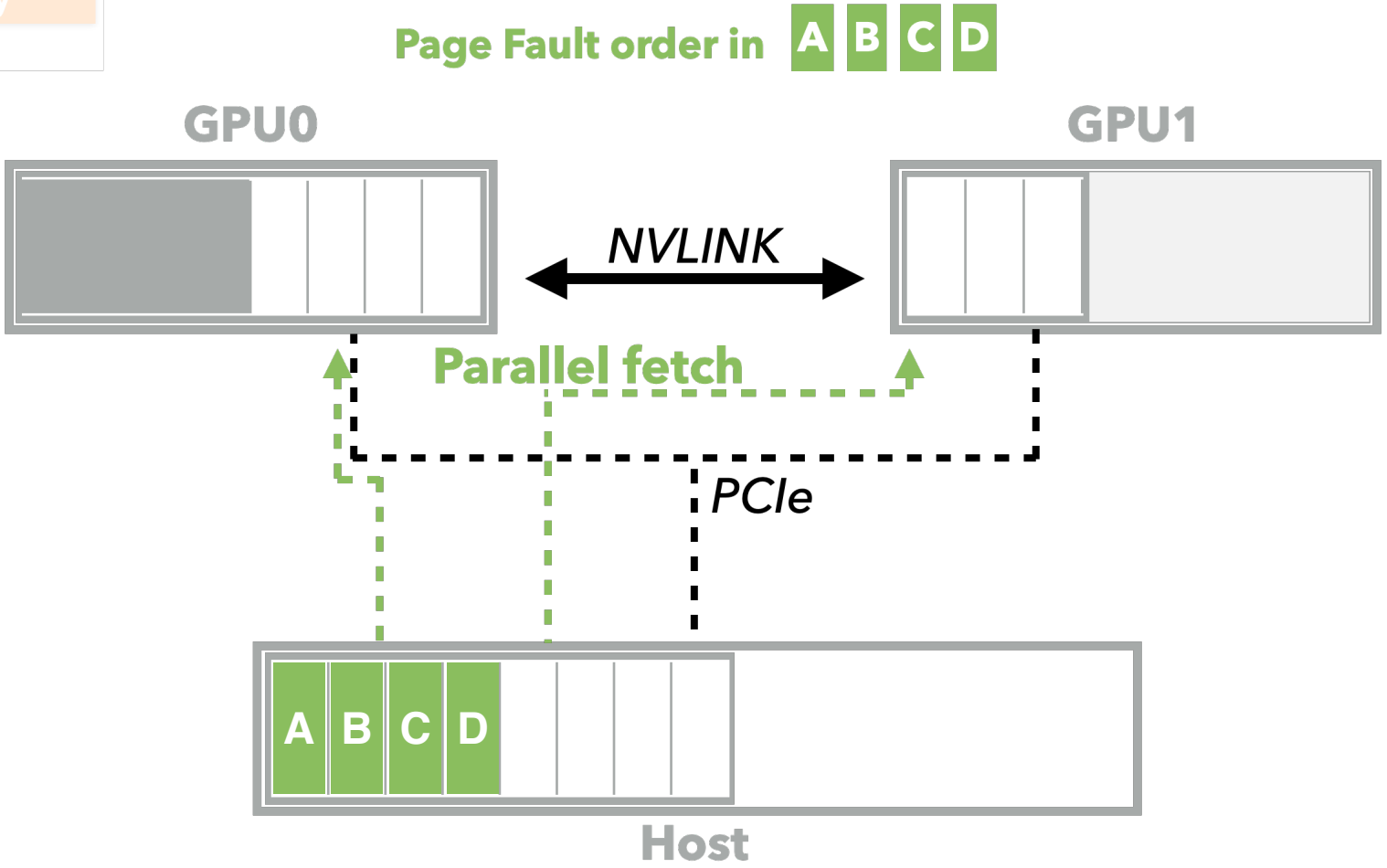


1. Hiding eviction latency

# 2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

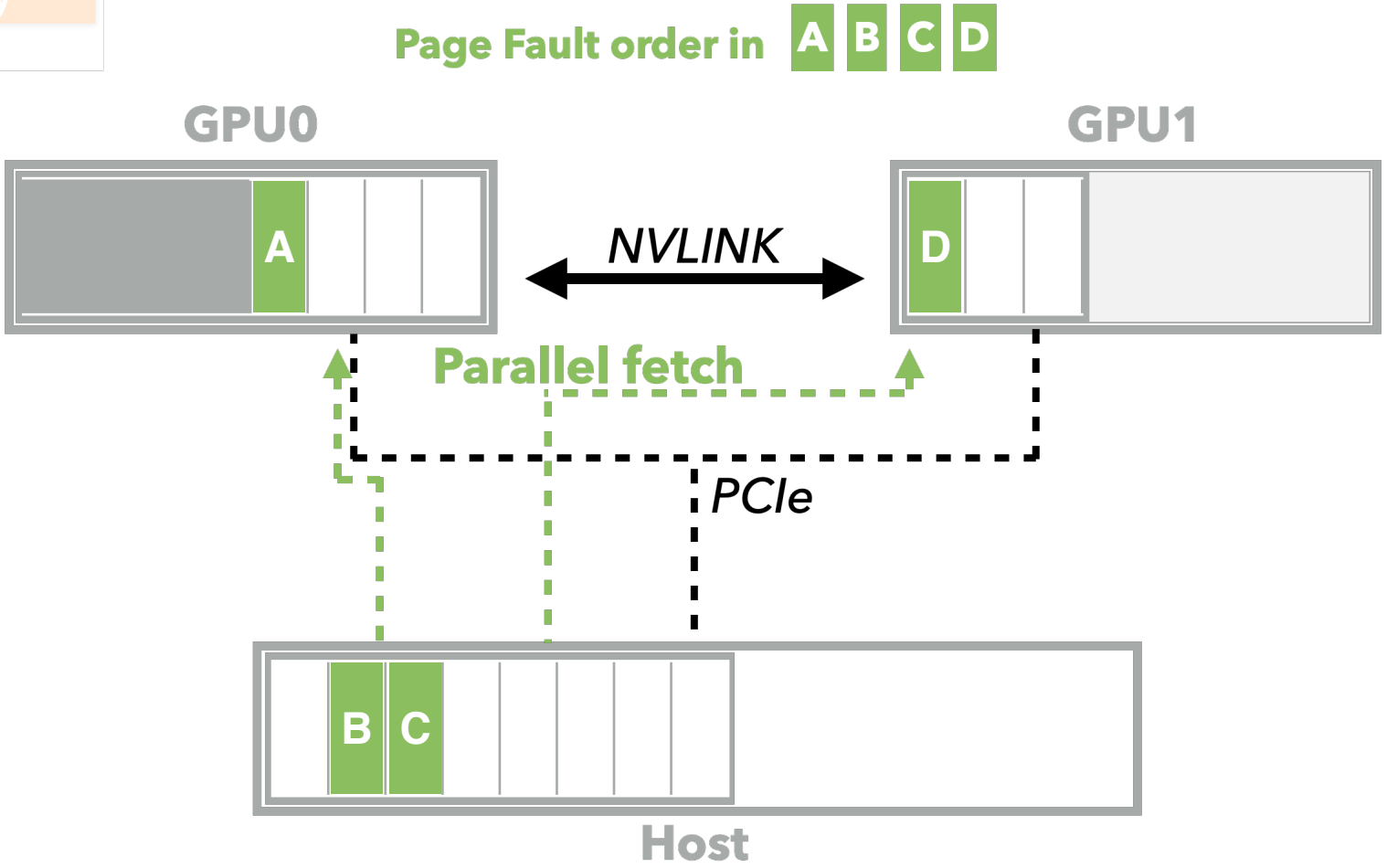


1. Hiding eviction latency

# 2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

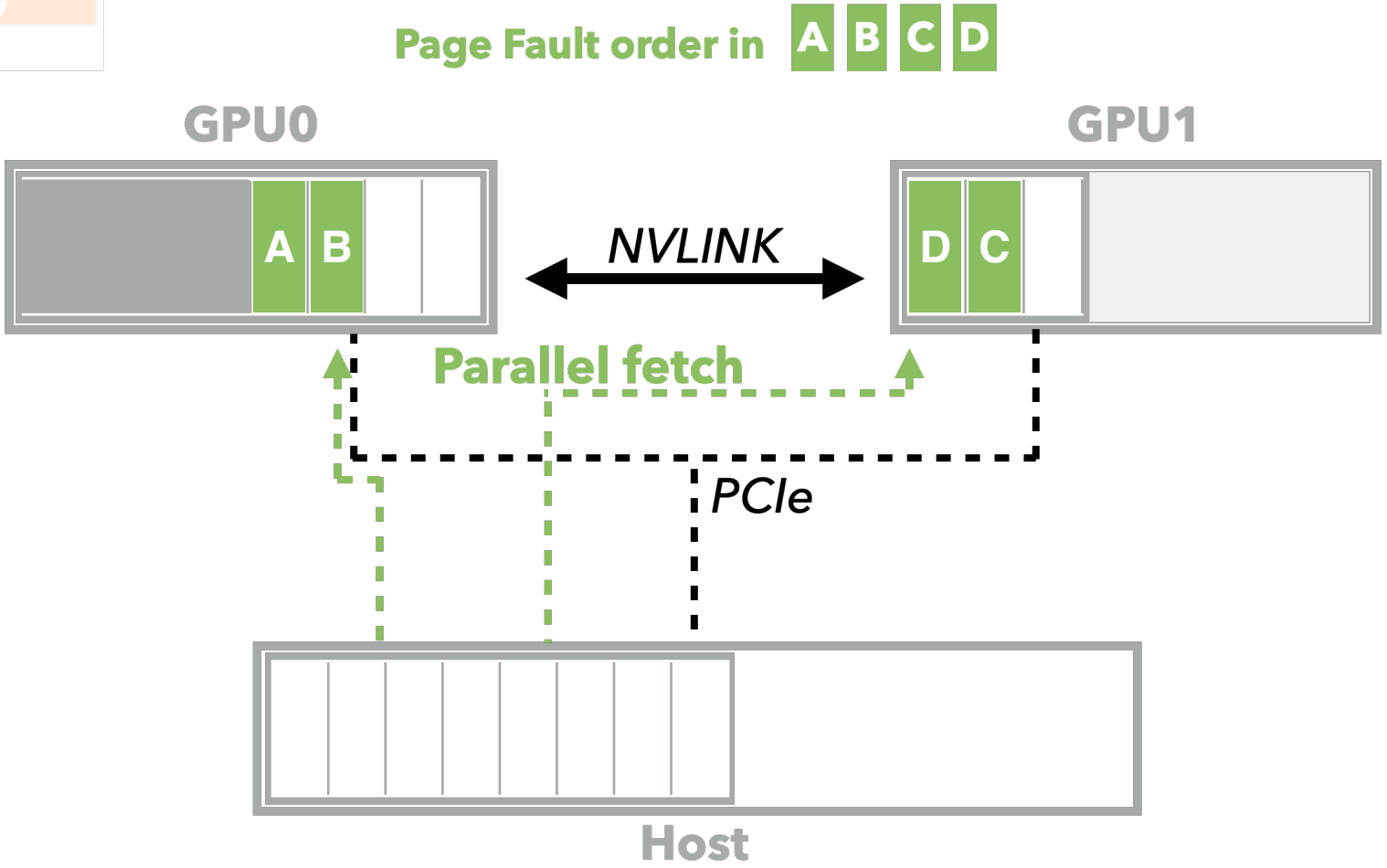


1. Hiding eviction latency

# 2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

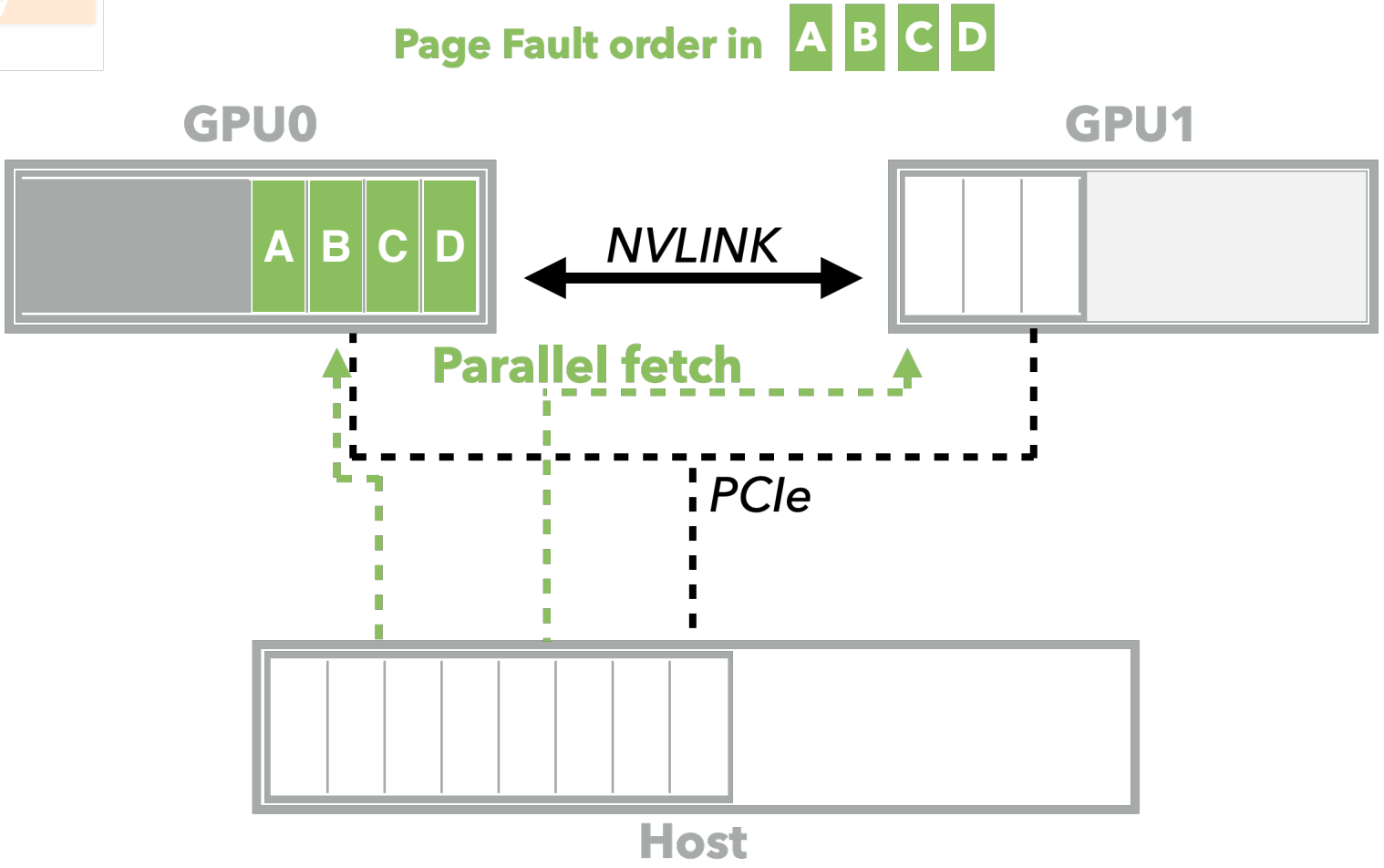


1. Hiding eviction latency

# 2. Reducing fetch latency

3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

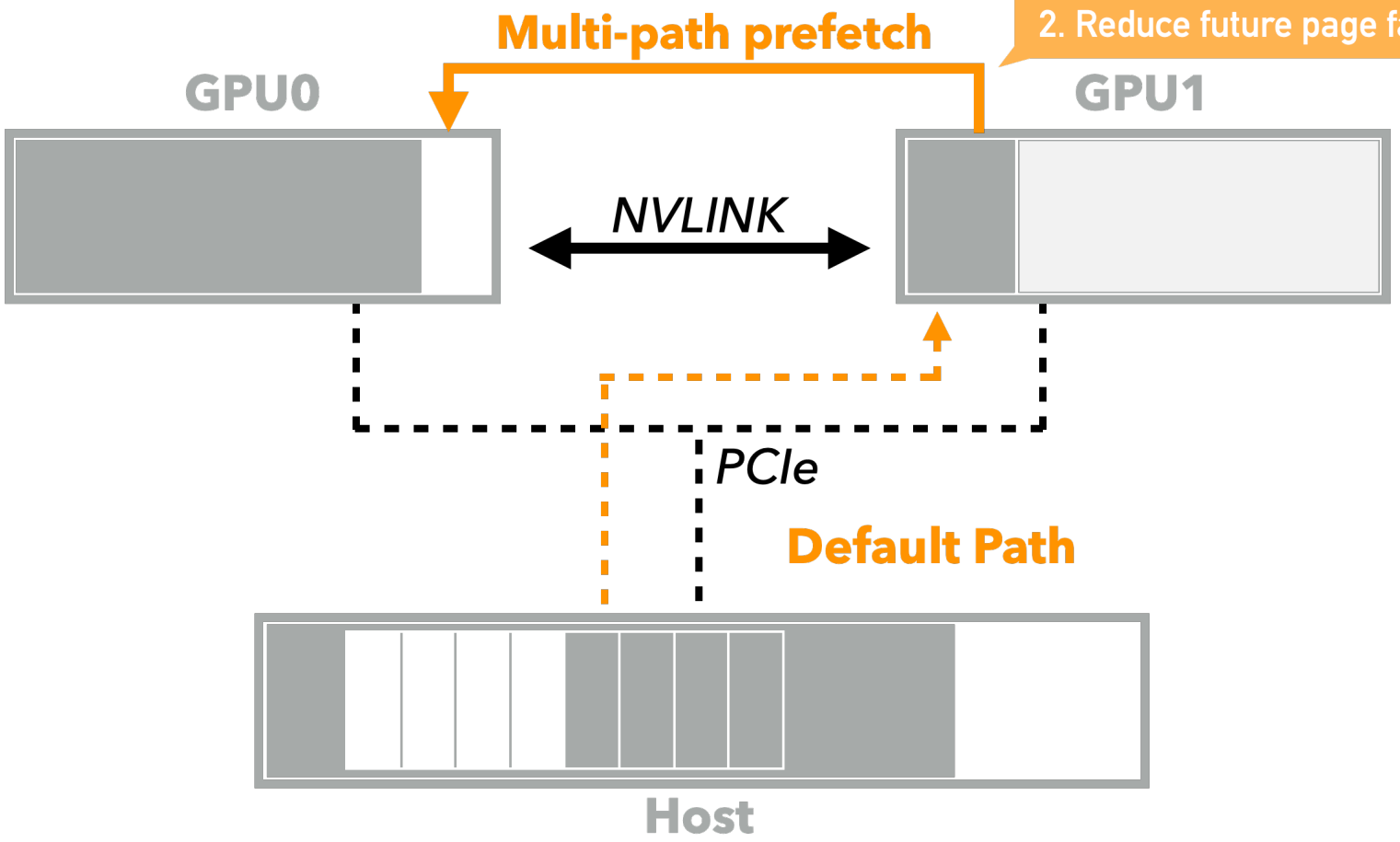


1. Hiding eviction latency
2. Reducing fetch latency

# 3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

1. Reduce page faults by proactive prefetching
2. Reduce future page fault latency

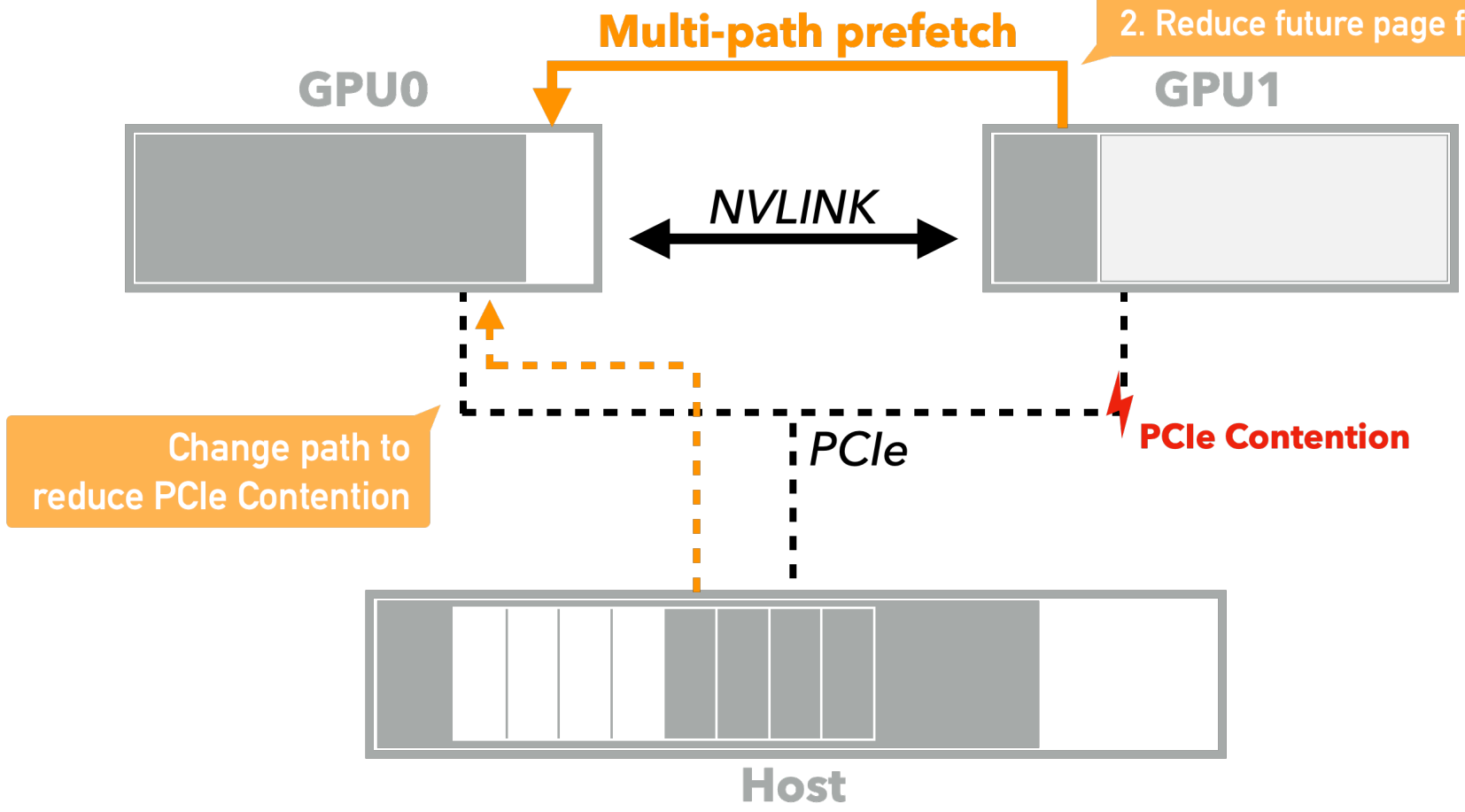


1. Hiding eviction latency
2. Reducing fetch latency

# 3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

1. Reduce page faults by proactive prefetching
2. Reduce future page fault latency

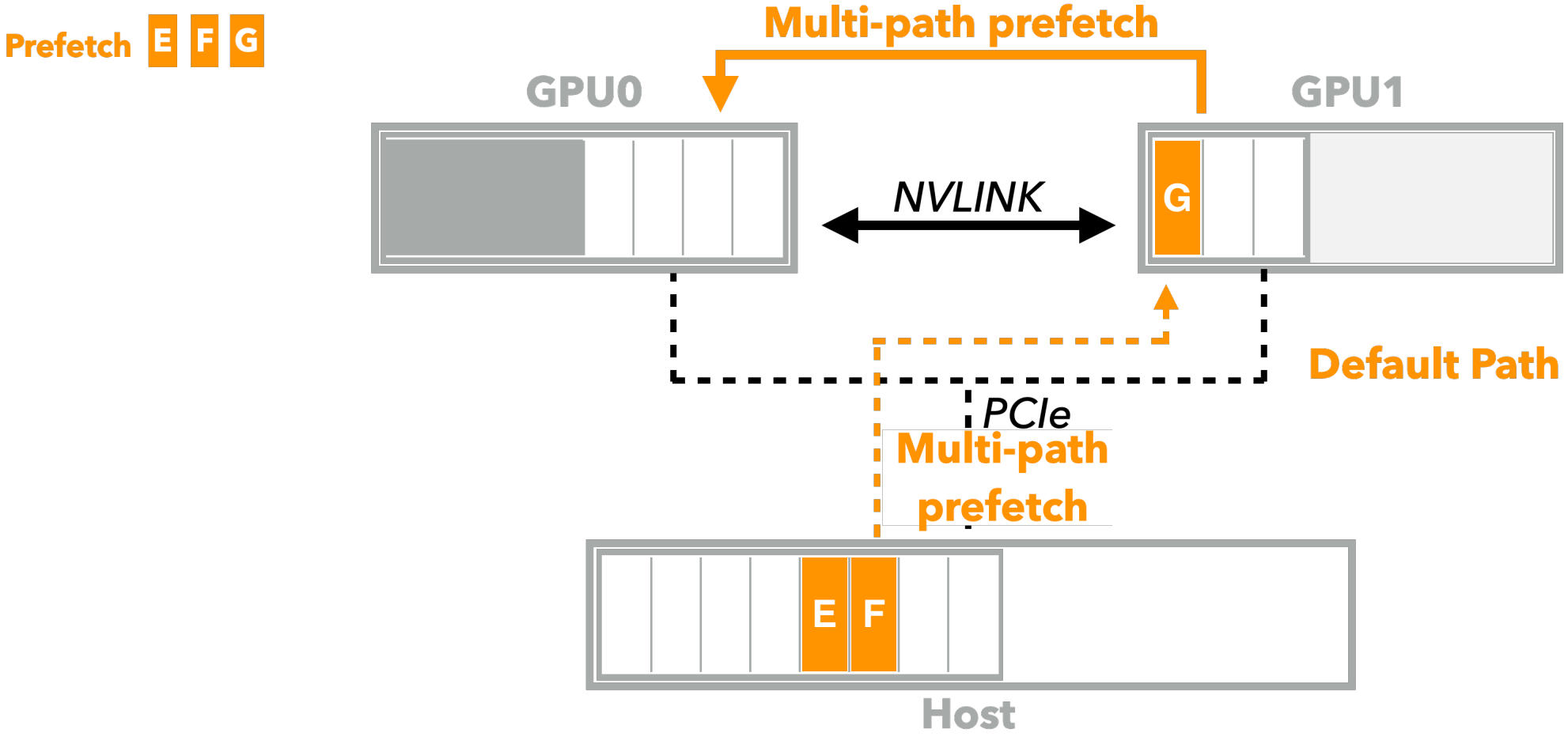


1. Hiding eviction latency

2. Reducing fetch latency

# 3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

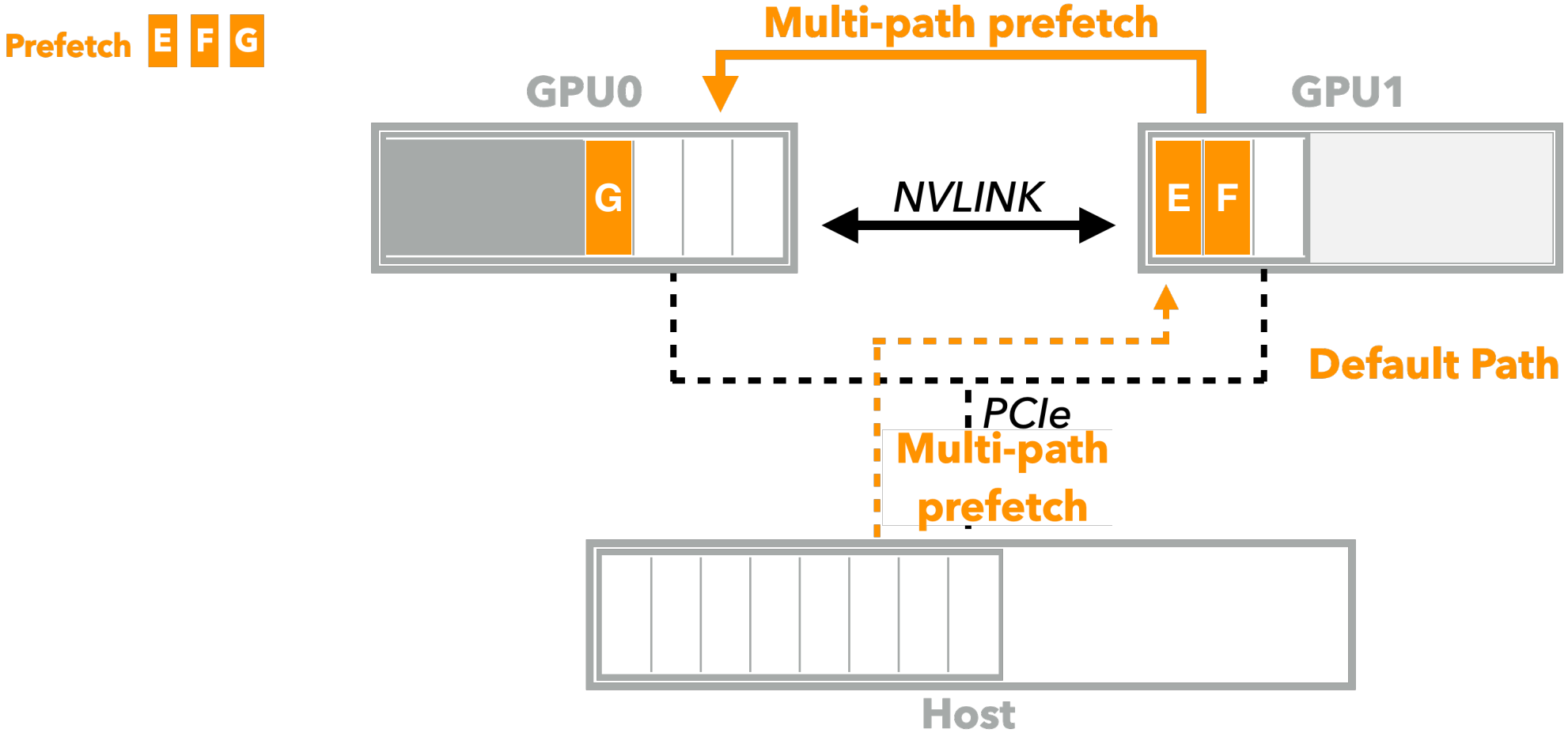


1. Hiding eviction latency

2. Reducing fetch latency

# 3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

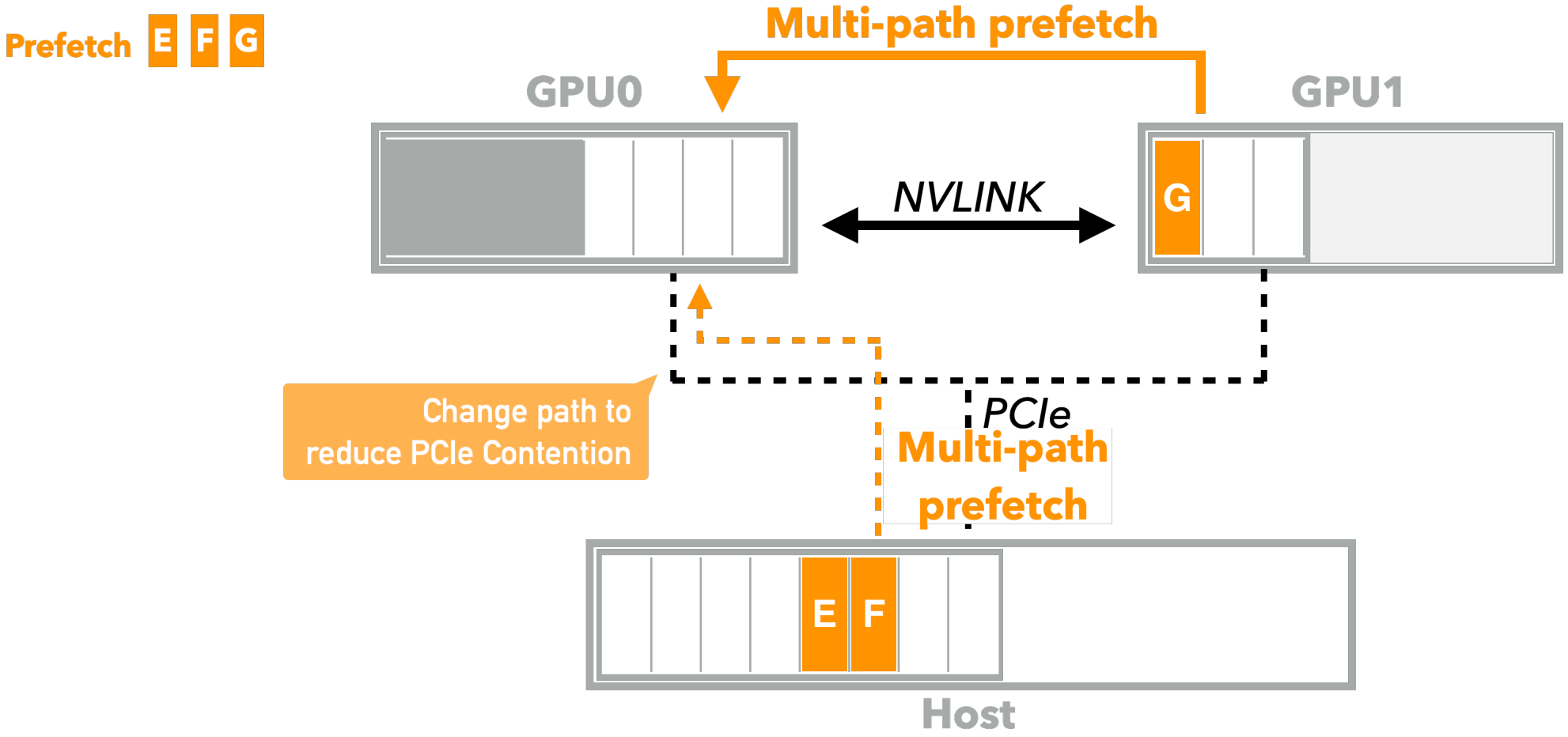


1. Hiding eviction latency

2. Reducing fetch latency

# 3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

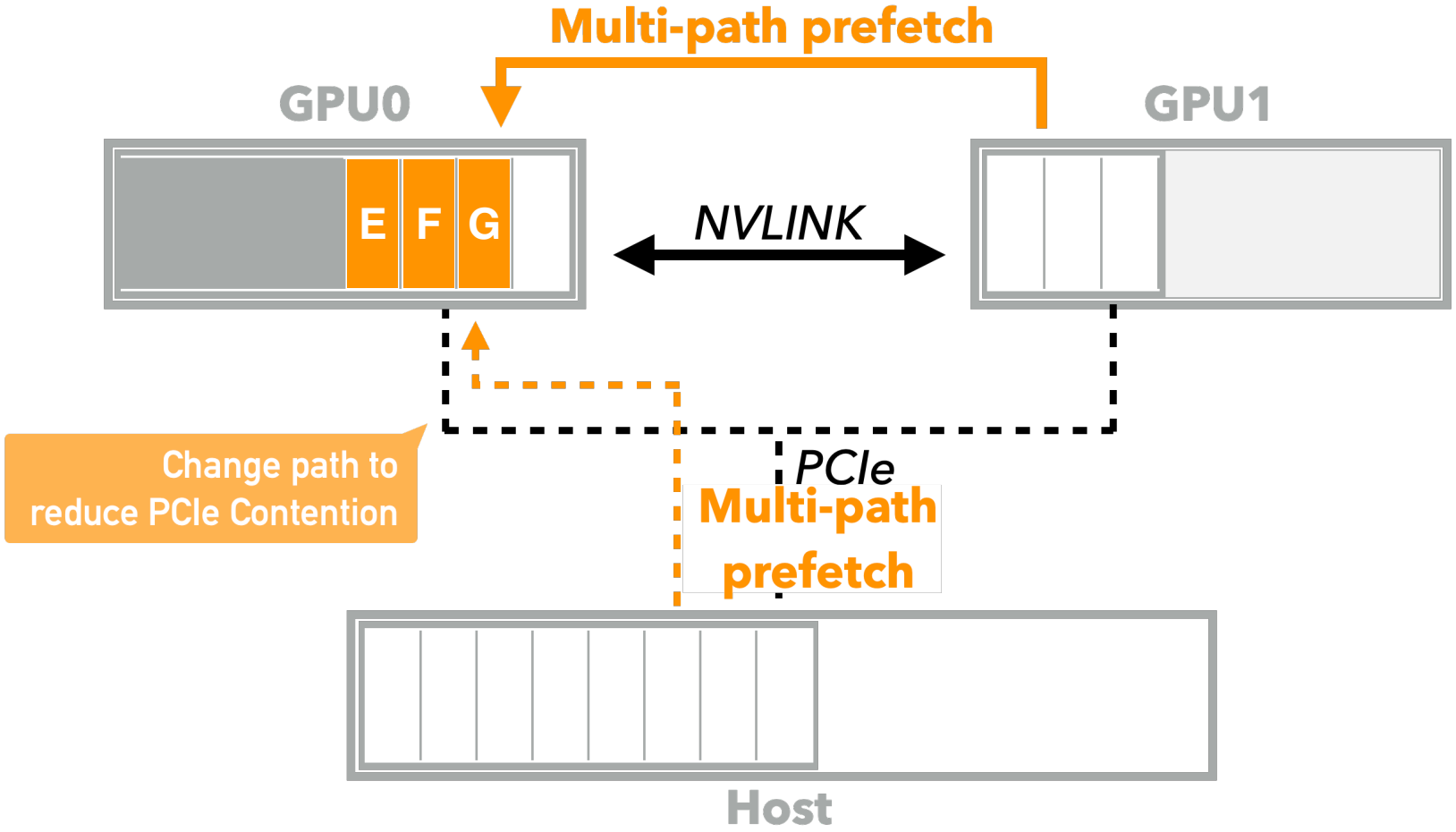


1. Hiding eviction latency
2. Reducing fetch latency

# 3. Hiding fetch latency

- Effective Harvesting
- Minimal Interference
- Framework-agnostic

Prefetch E F G



# Goals of Hierarchical Unified Virtual Memory

Reducing performance cost of memory oversubscription

| | |
|---|---|
| **Effective Harvesting** | Harvest small and temporarily available spare memory of neighbor GPU<br>Reduce eviction/fetch latency with spare memory |
| **Minimal Interference** | Minimize performance impact of workloads running in neighbor GPU |
| **Framework-agnostic** | No modification of applications or frameworks |

# Memory manager for HUVM: memHarvester

Centralized coordinator for data-path in HUVM

**Effective Harvesting**

Harvest small and temporarily available spare m[illegible] of neighb[illegible]

Pre-eviction

Parallel Fetch

Multi-path Prefetch

Reduce eviction/fetch latency with spare memory

**Minimal Interference**

Minimize performance impact of workloads runnin[illegible] GPU

Removable Page

**Framework-agnostic**

No mo[illegible] of applications or frameworks

Leverage UVM

# Evaluation

- System configuration (AWS p3.8xlarge)

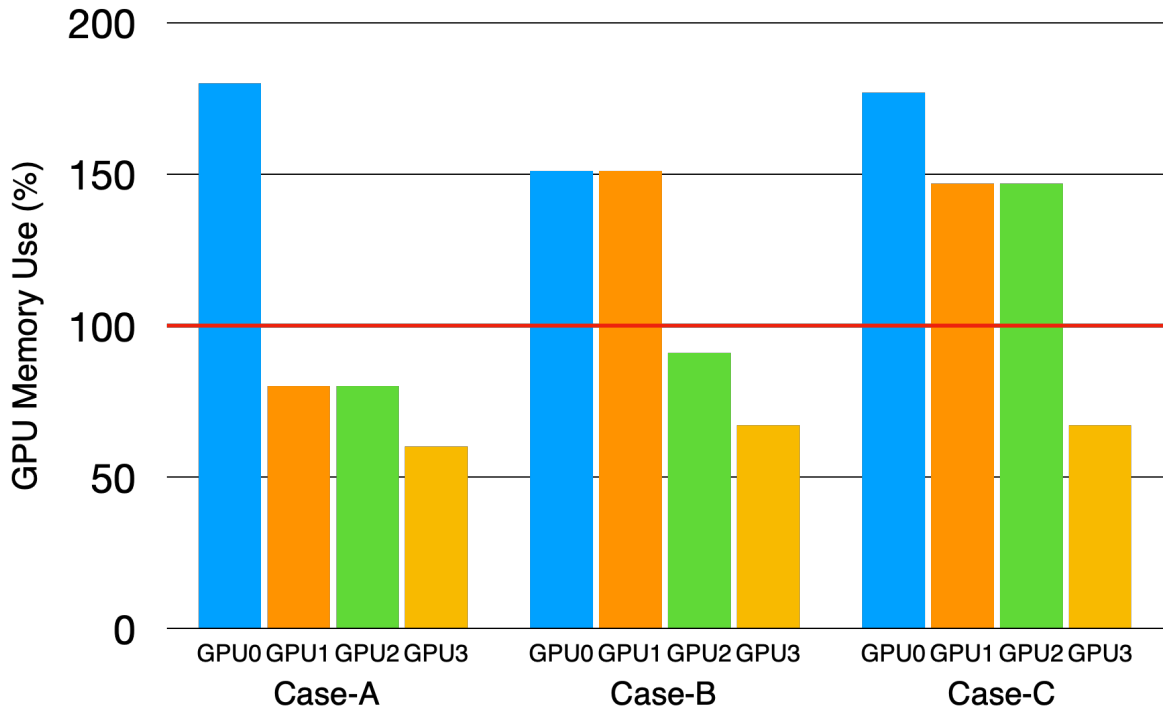
| | |
|---|---|
| **GPU** | NVIDIA V100 (x4) (**16 GB each**)<br>(Connected through **NVLink** bridge) |
| **Processors** | Intel Xeon Skylake vCPU (x32) |
| **Memory** | 224GB DDR4 |
| **Driver Version** | NVIDIA Driver 460.67 |

- Benchmarks
  - cuGraph (version 21.12)
  - PyTorch (version 1.10.1)

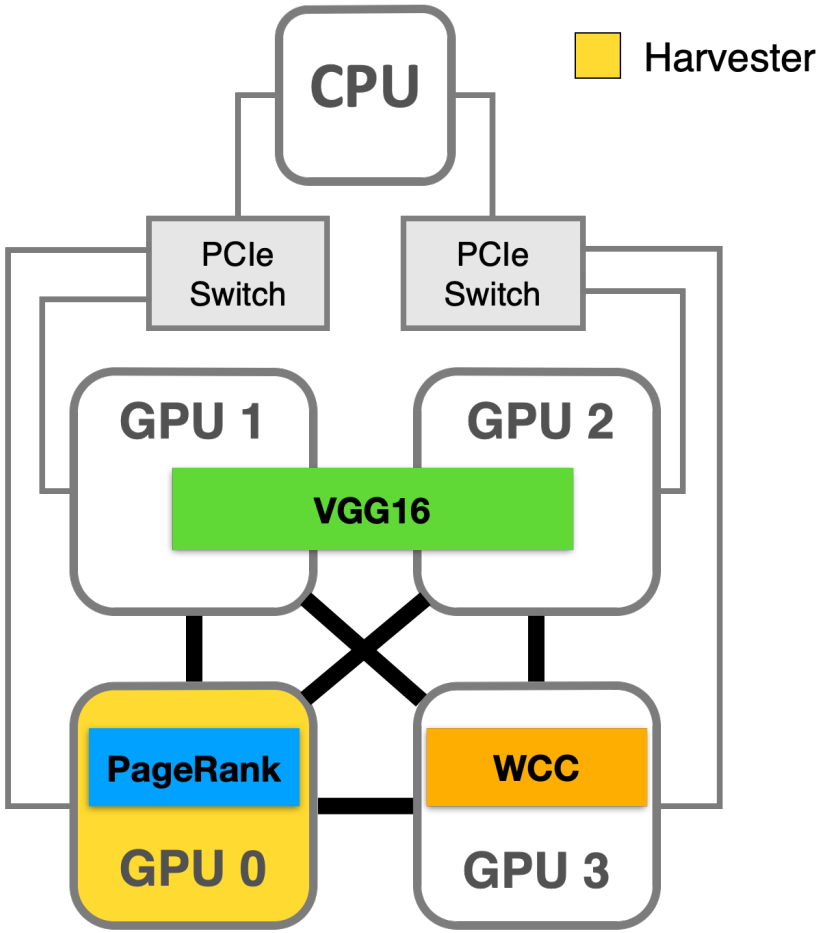
# Workload running scenarios

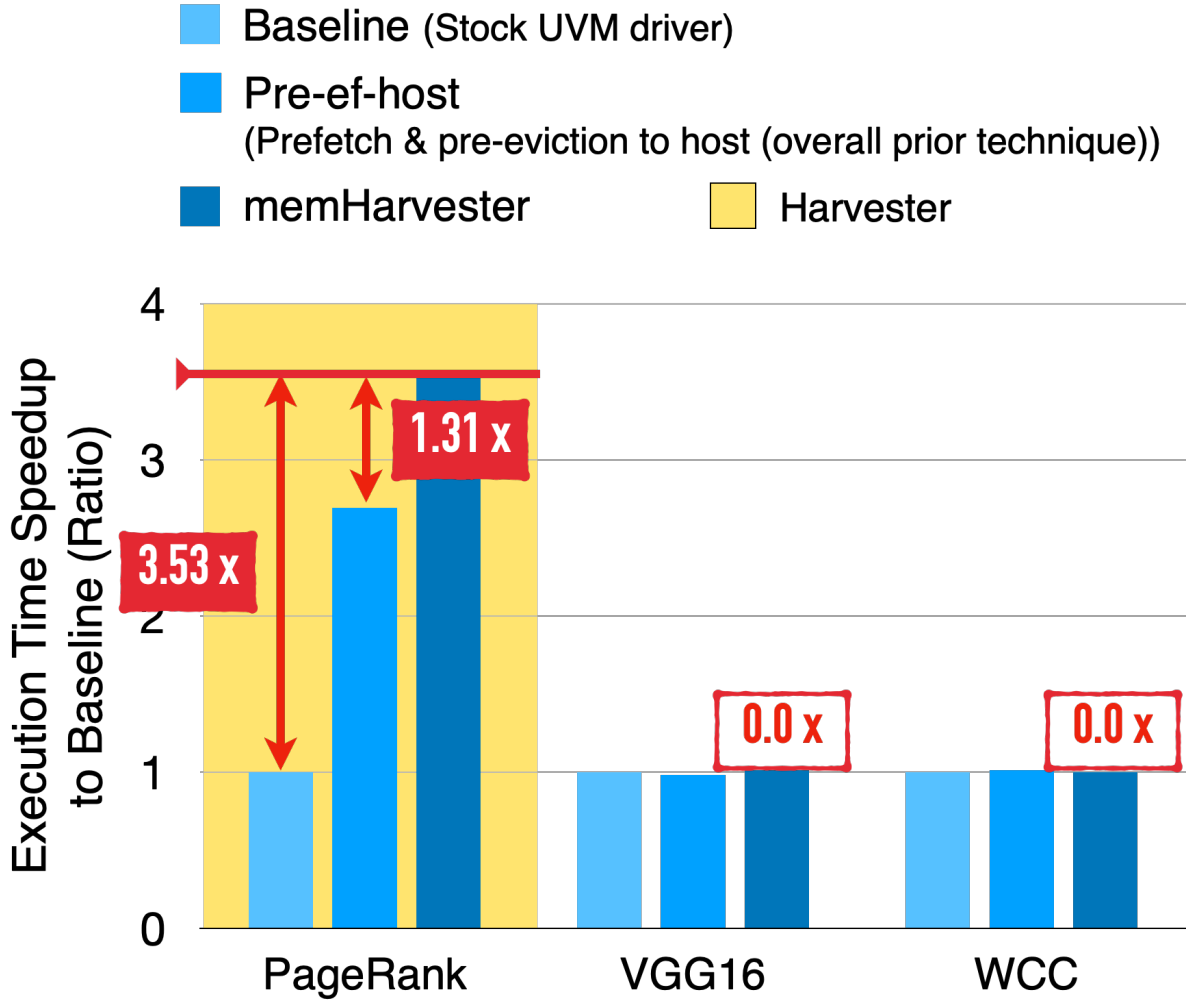
Harvester

| | GPU-0 | GPU-1 | GPU-2 | GPU-3 |
|---|---|---|---|---|
| Case-A | PageRank<br>soc-twitter-2010 | VGG16<br>Batch 256 | | WCC<br>soc-sinaweibo |
| Case-B | BFS<br>web-uk-2005 | | MobileNet<br>Batch 256 | ResNet101<br>Batch 64 |
| Case-C | WCC<br>soc-twitter-2010 | Louvain<br>web-uk-2005 | | ResNet101<br>Batch 64 |

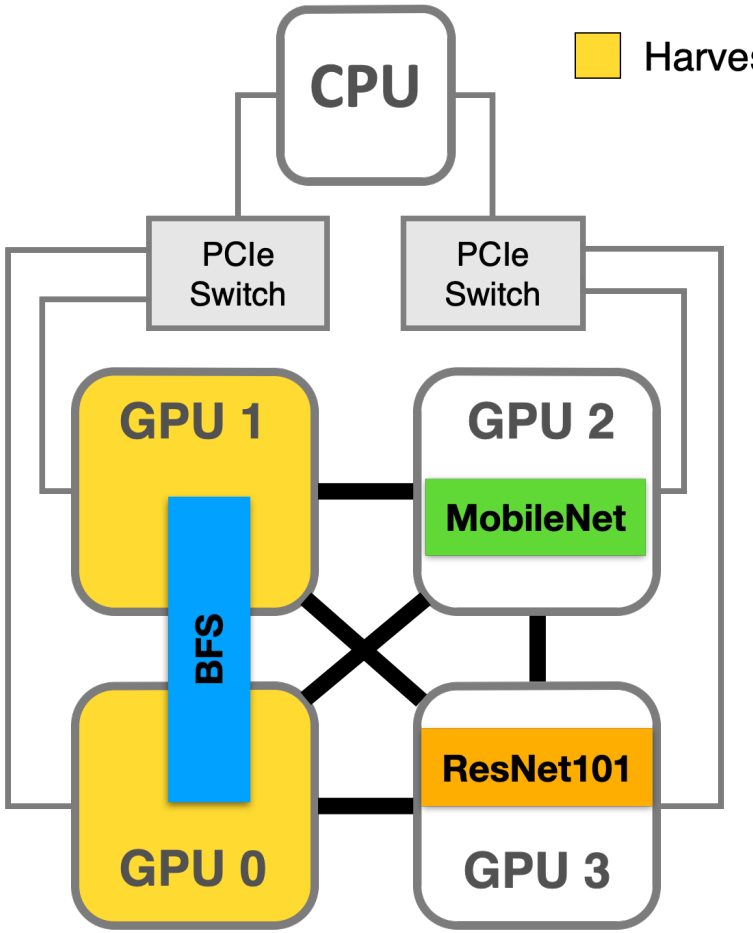


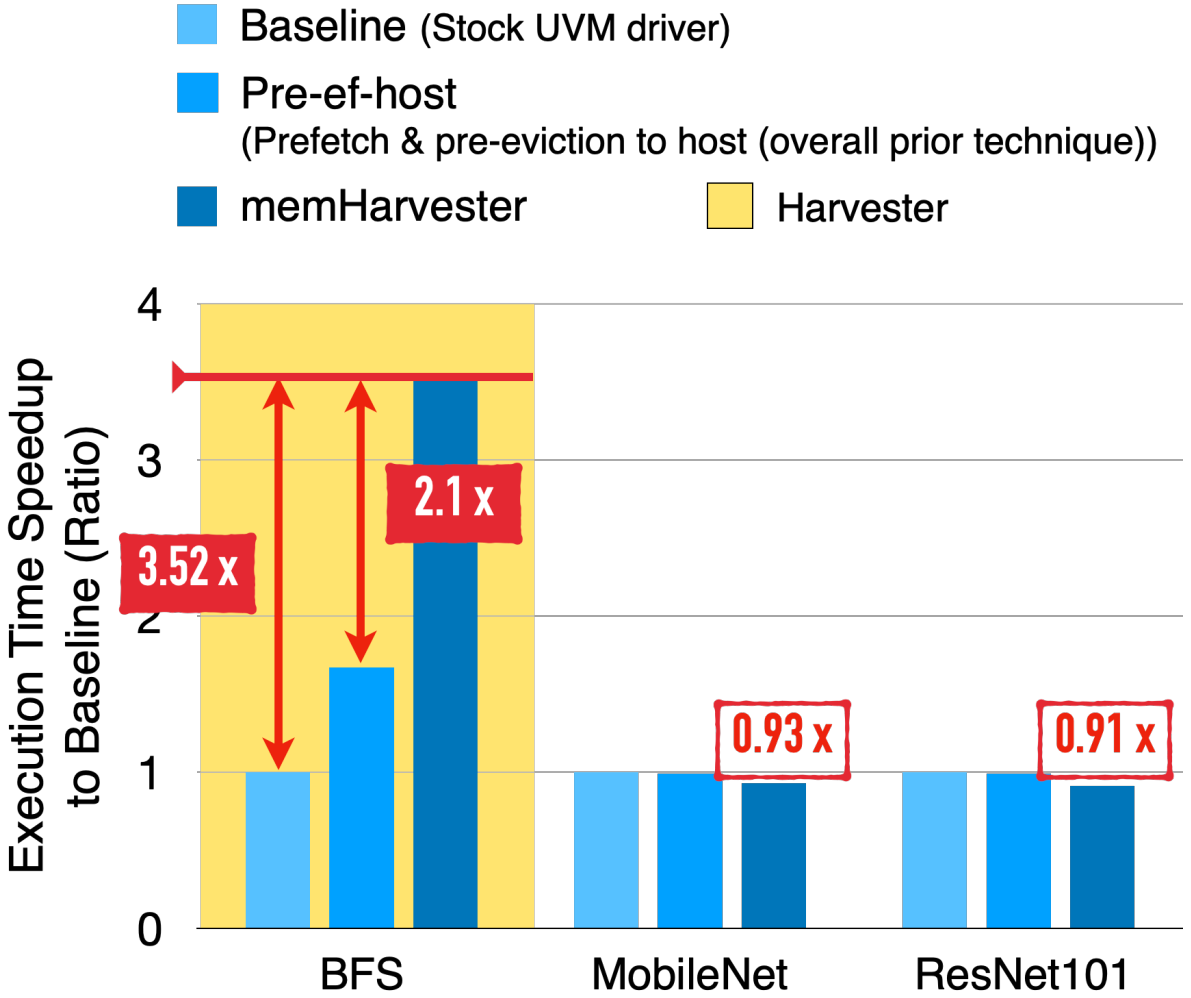
# Case-A (1 harvests 3)



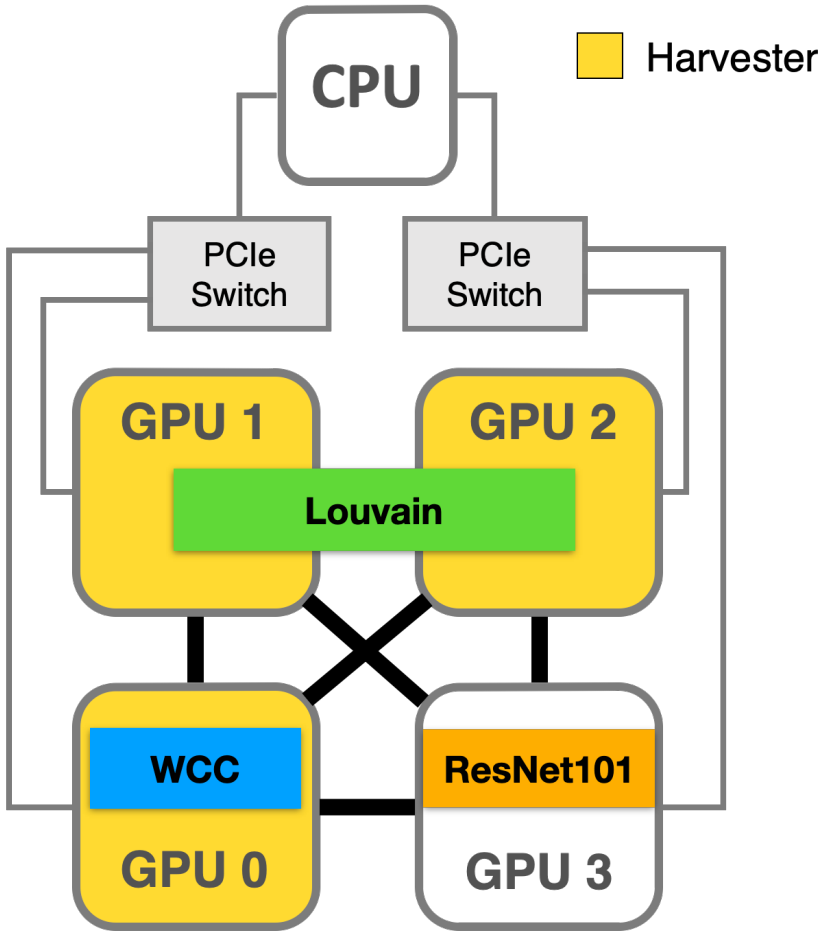



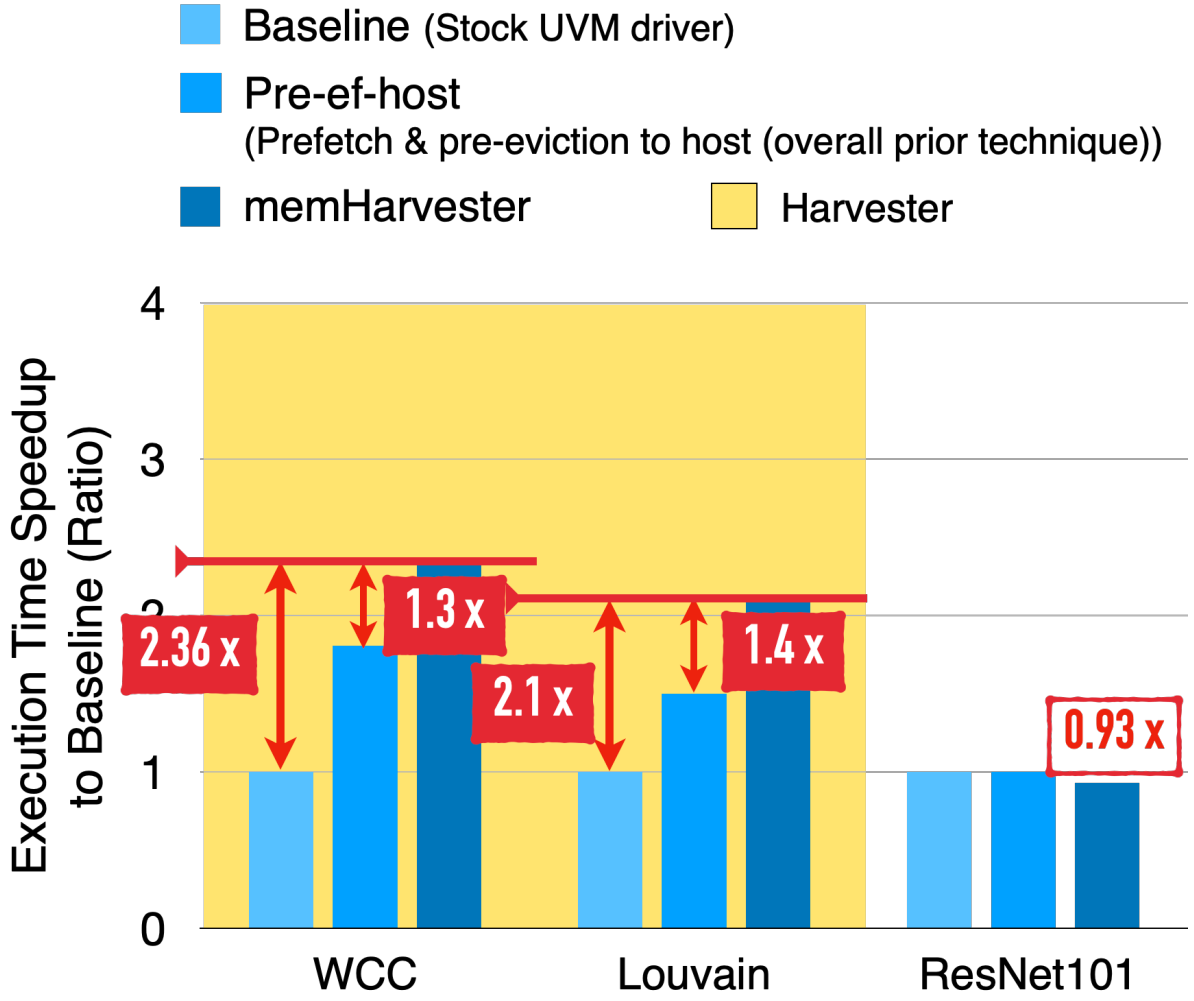
# Case-B (2 harvests 2)



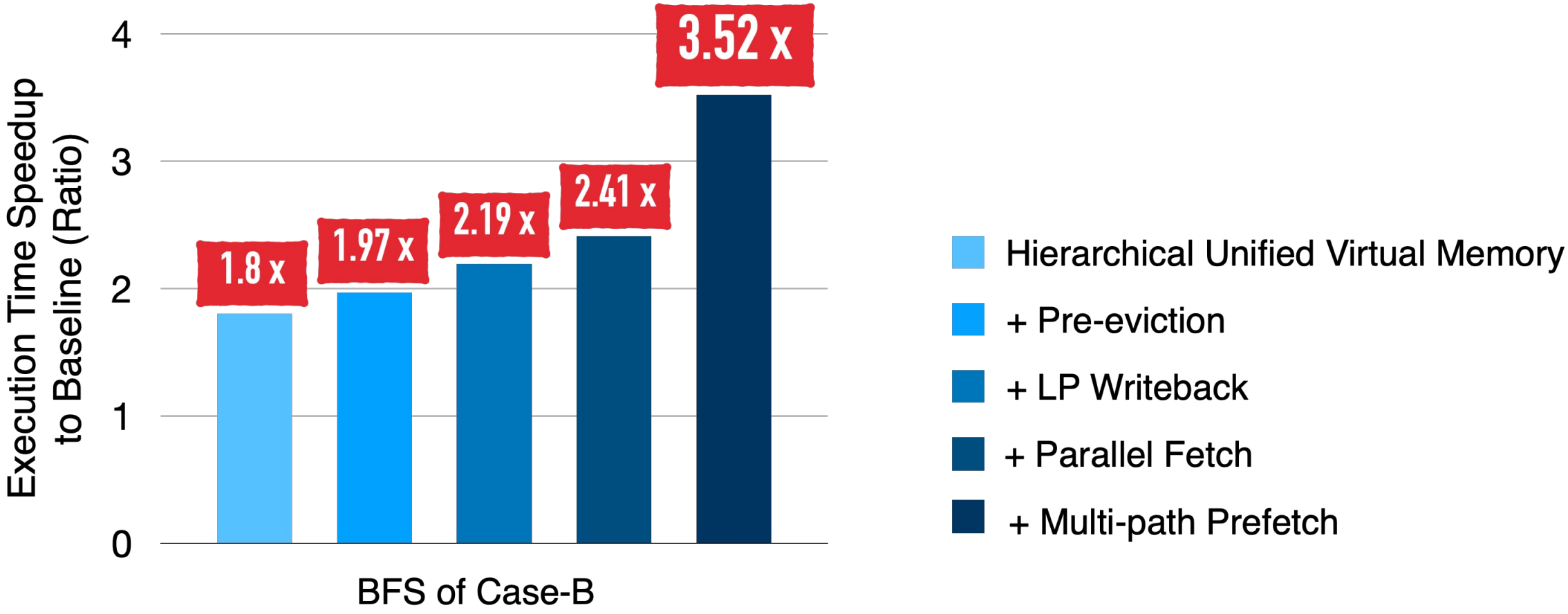



# Case-C (3 harvests 1)

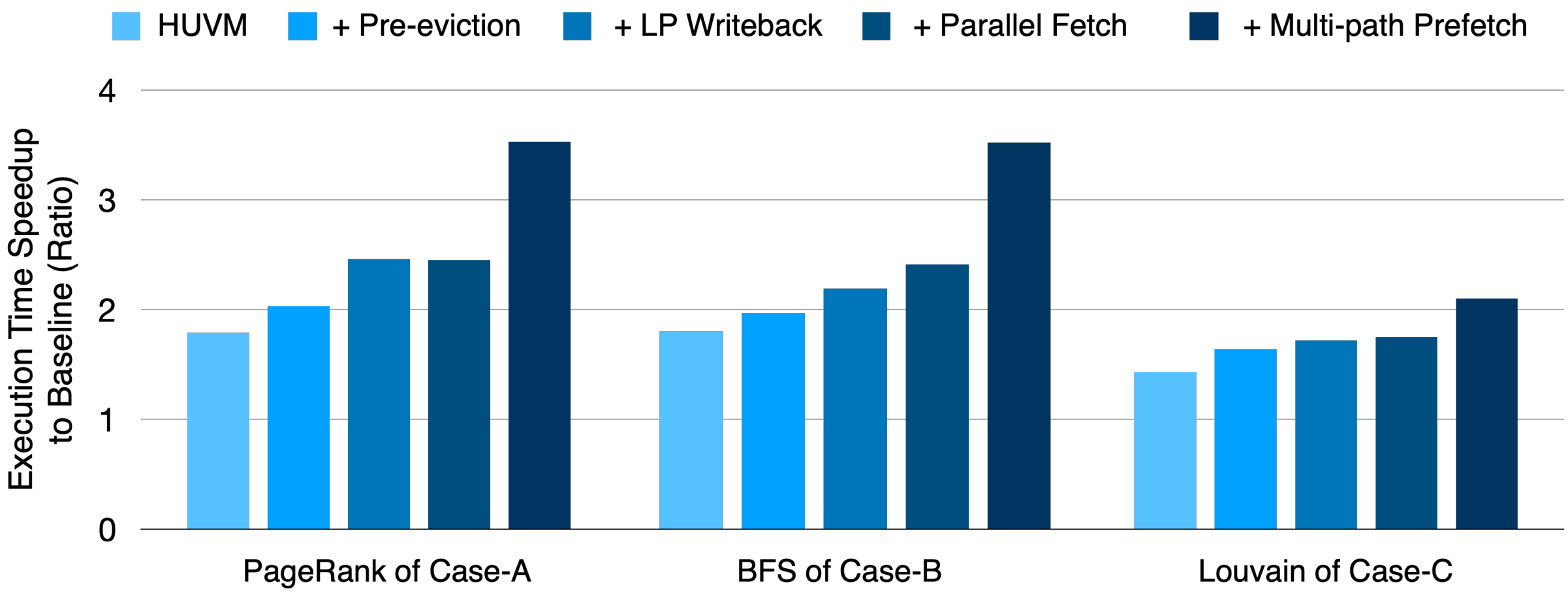


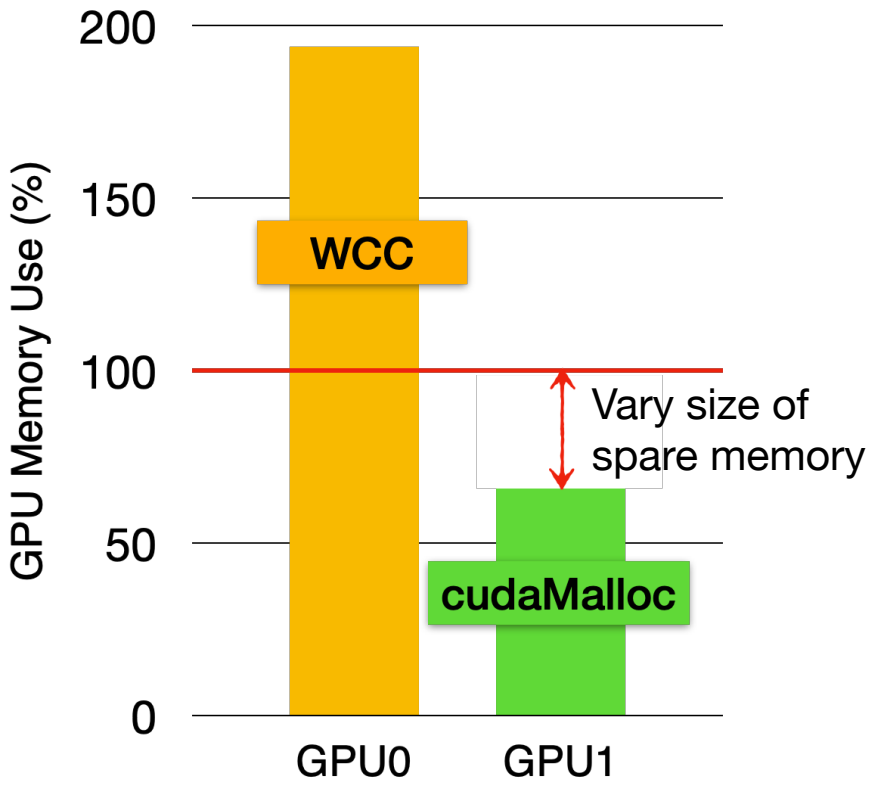
# Individual technique performance breakdown

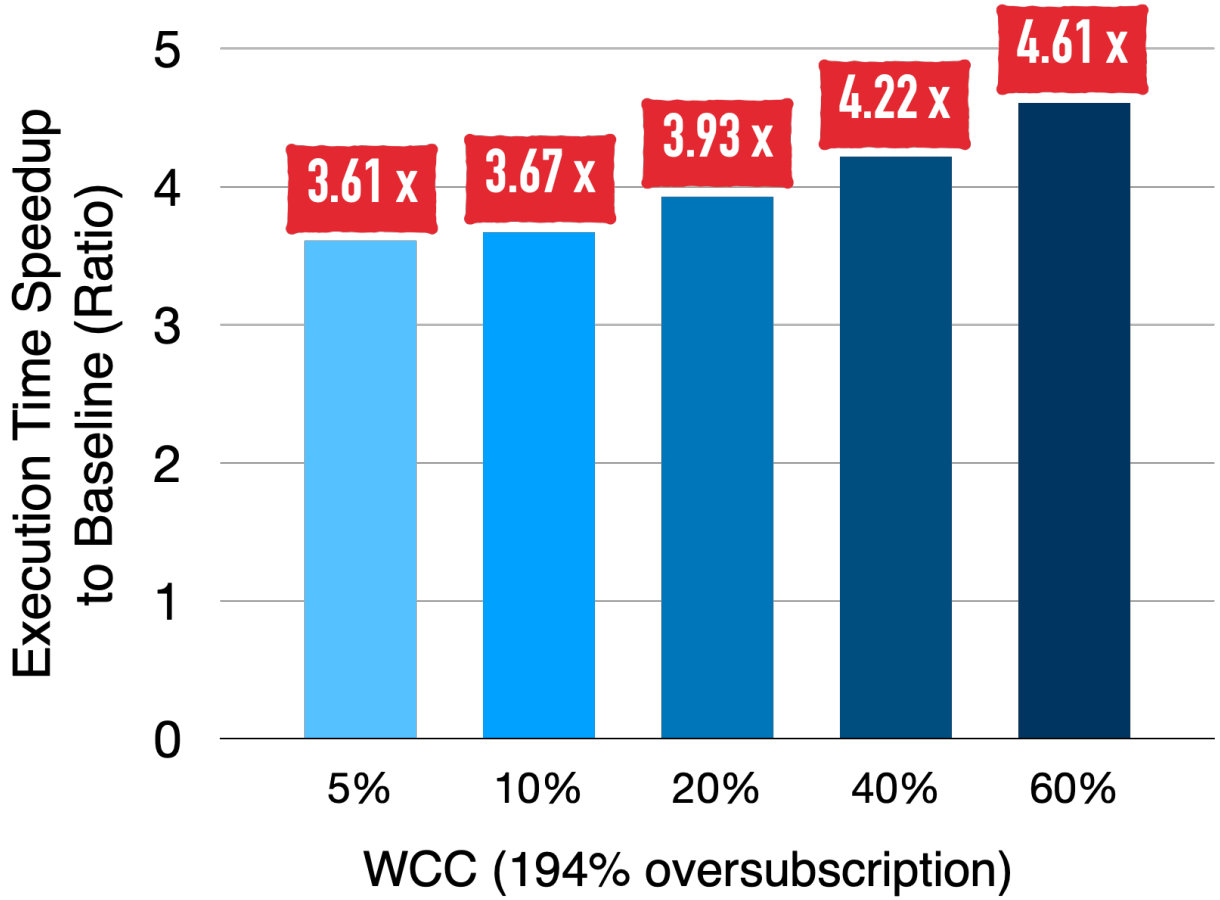


# Individual technique performance breakdown



# Sensitivity to size of spare memory



# Conclusion

**Problem**

Resource imbalance in shared multi-GPU server
Significant performance overhead for memory oversubscription

**Approach**

1. Spill fraction of oversubscription to neighbor GPUs
2. Build new data-path with fast interconnect (NVLink)
3. Propose memory manager for HUVM

**Result**

Hierarchically unified virtual memory (w/ memory manager)

Performance improvement compared to naive UVM **3x ↑**
W/ small fraction of spare memory
W/ Less performance interference **< 10%**