GPS: A Global Publish-Subscribe Model for Multi-GPU Memory Management

International Symposium on Microarchitecture (MICRO) 2021

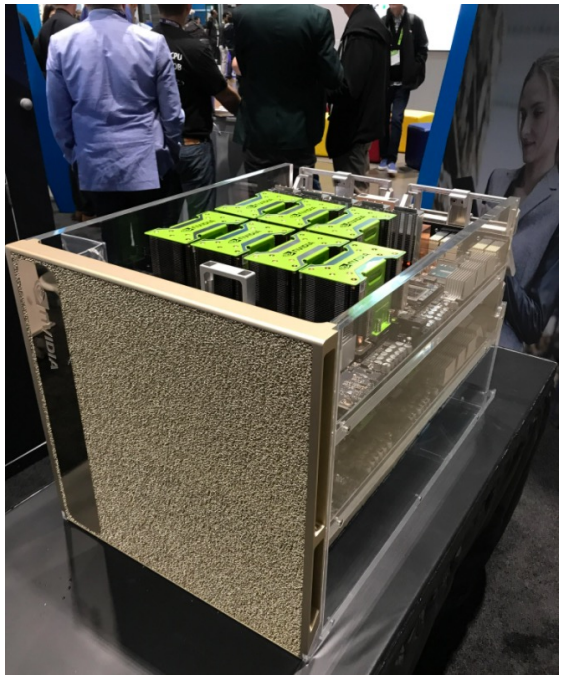
October 18-22, 2021

Harini Muthukrishnan (University of Michigan), Daniel Lustig (NVIDIA),

David Nellans (NVIDIA), Thomas Wenisch (University of Michigan)

Multi-GPU Systems for High Performance Computing

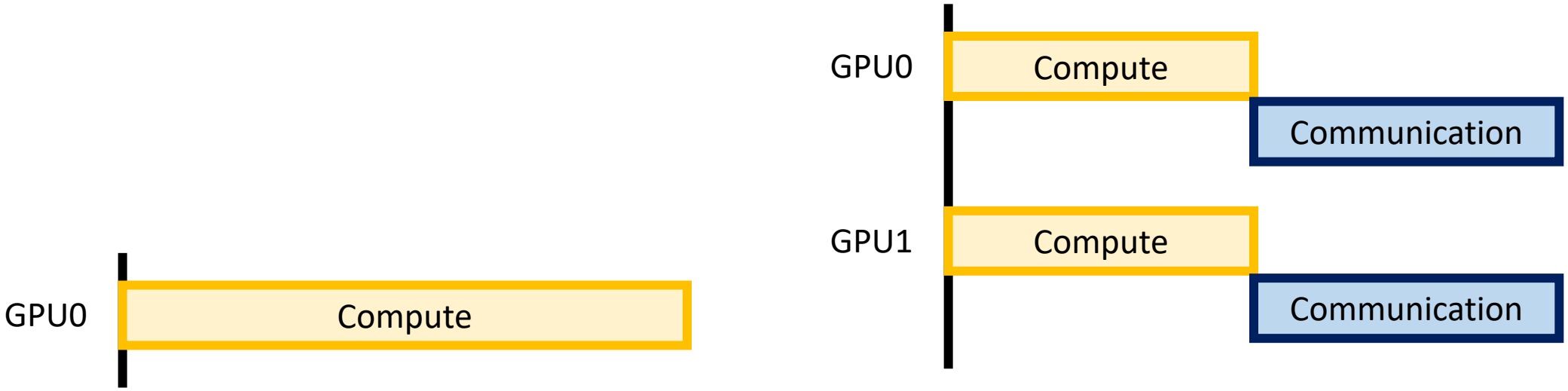
- GPUs well suited for HPC applications
- Single GPU often insufficient to exploit all parallelism
- Multi-GPU systems enable further scaling



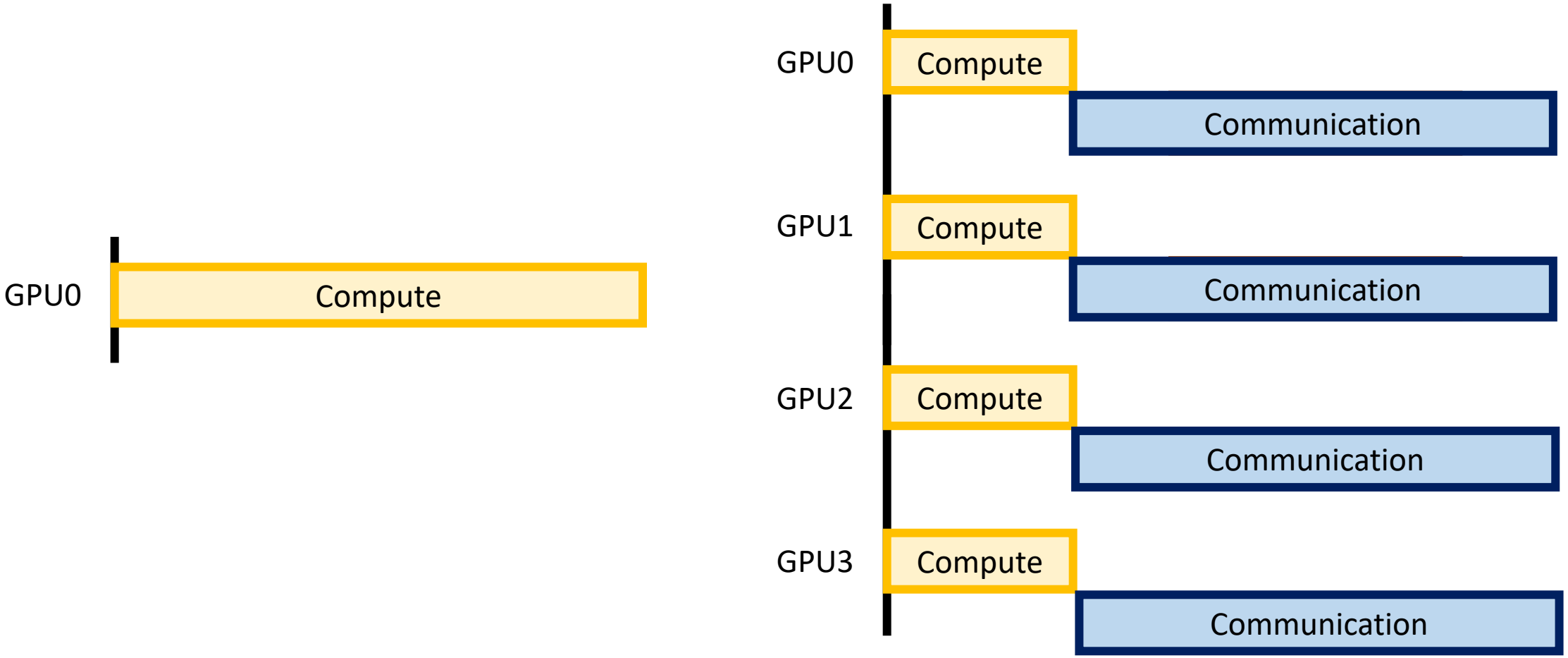
NVIDIA DGX-2

Inter-GPU Communication Hampers Scalability

Inter-GPU Communication Hampers Scalability



Inter-GPU Communication Hampers Scalability



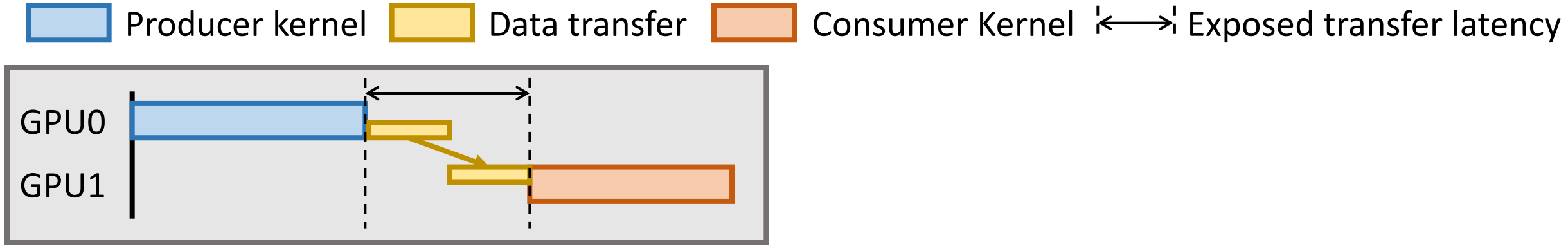
Inter-GPU communication remains the primary barrier to multi-GPU strong scaling

Contributions

- GPS: publish-subscribe memory management for multi-GPU systems
- Pages replicated on subscribed GPUs, updated via proactive stores
- A dynamic unsubscription technique to conserve interconnect BW
- Global memory's programmability at GPU-local memory performance

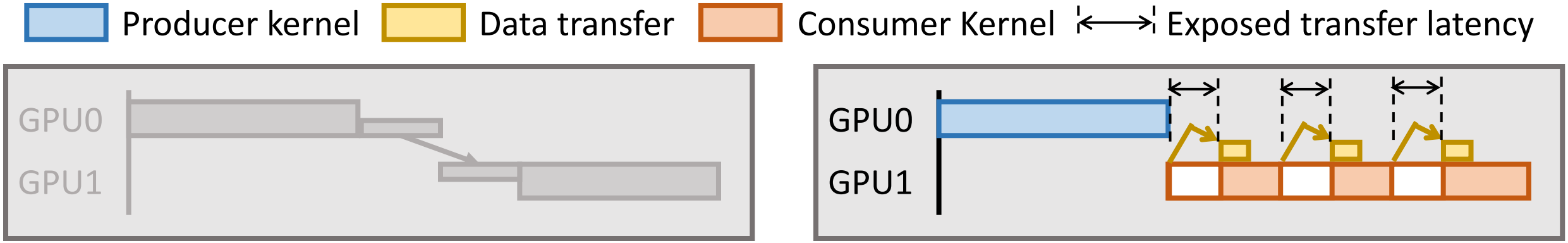
Achieves 7.9x performance improvement on a 16-GPU system
4x better than the next best programming paradigm

Inter-GPU Communication Challenges



Bulk DMA (cudaMemcpy) exposes transfer latency

Inter-GPU Communication Challenges

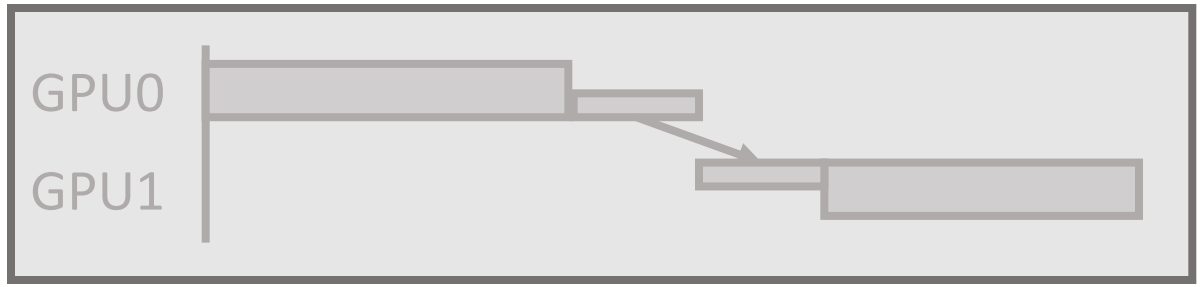


Bulk DMA (cudaMemcpy) exposes transfer latency

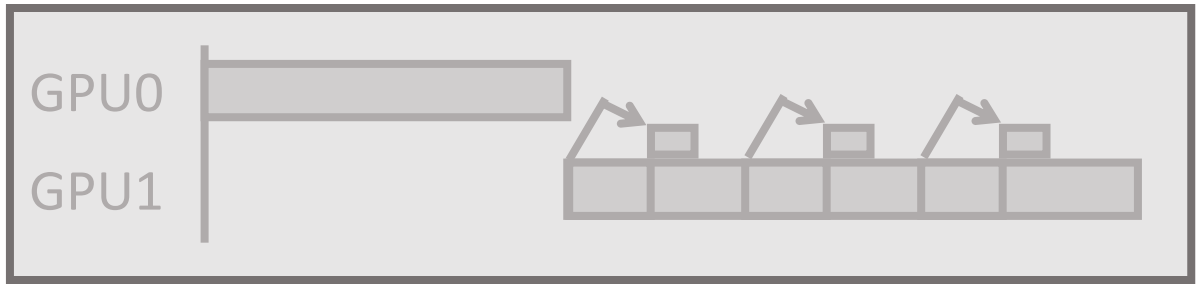
Peer-to-peer loads expose remote load latency

Inter-GPU Communication Challenges

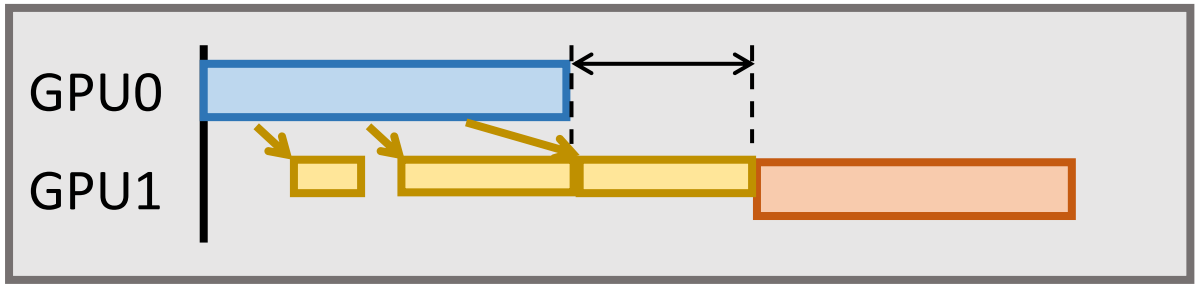
Producer kernel Data transfer Consumer Kernel Exposed transfer latency



Bulk DMA (cudaMemcpy) exposes transfer latency



Peer-to-peer loads expose remote load latency

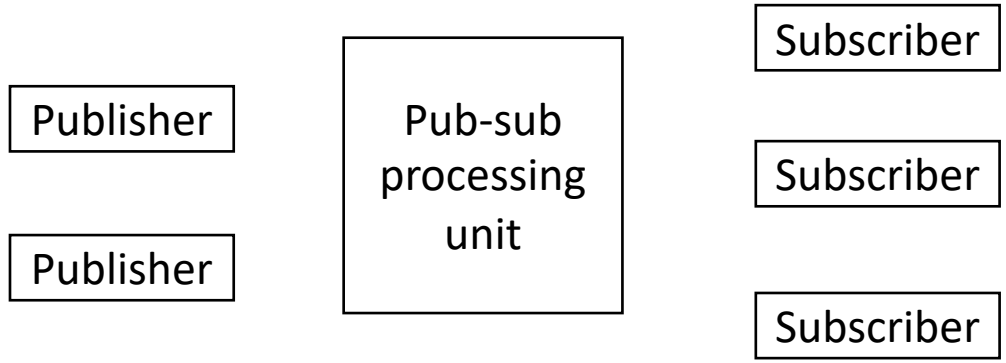


Peer-to-peer stores provides fine-grained overlap of compute and communication

Unneeded transfers waste interconnect bandwidth

Publish-Subscribe Model

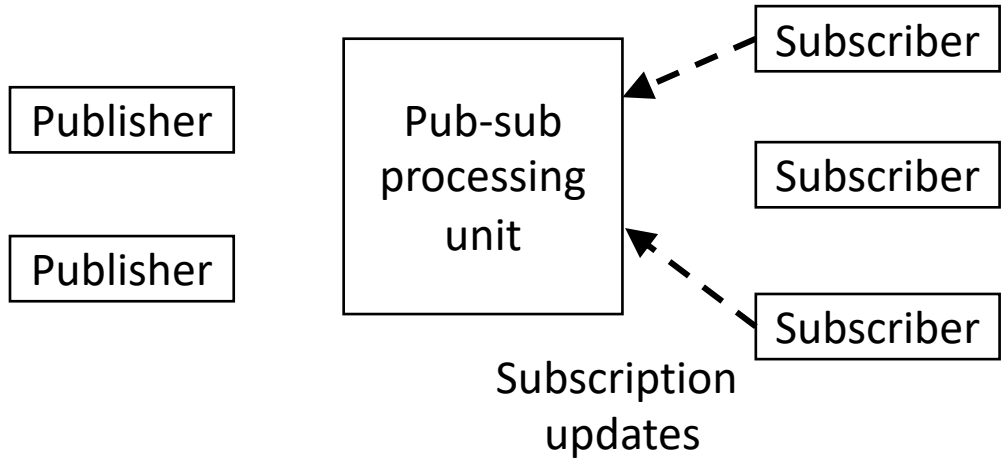
- Not all GPUs read pages written by others
- All-to-all store broadcasts waste BW
- Publish-subscribe model:
  - GPUs can subscribe to pages they read
  - Stores forwarded only to subscribers



Publish-subscribe model can save precious interconnect bandwidth

Publish-Subscribe Model

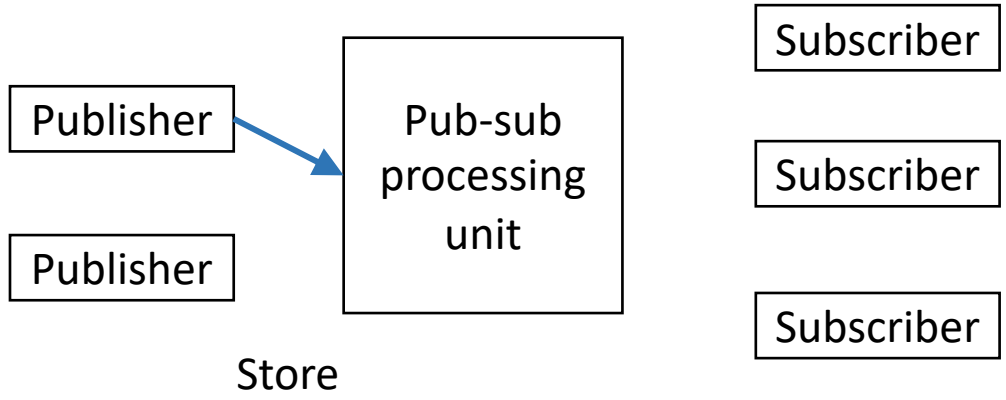
- Not all GPUs read pages written by others
- All-to-all store broadcasts waste BW
- Publish-subscribe model:
  - GPUs can subscribe to pages they read
  - Stores forwarded only to subscribers



Publish-subscribe model can save precious interconnect bandwidth

Publish-Subscribe Model

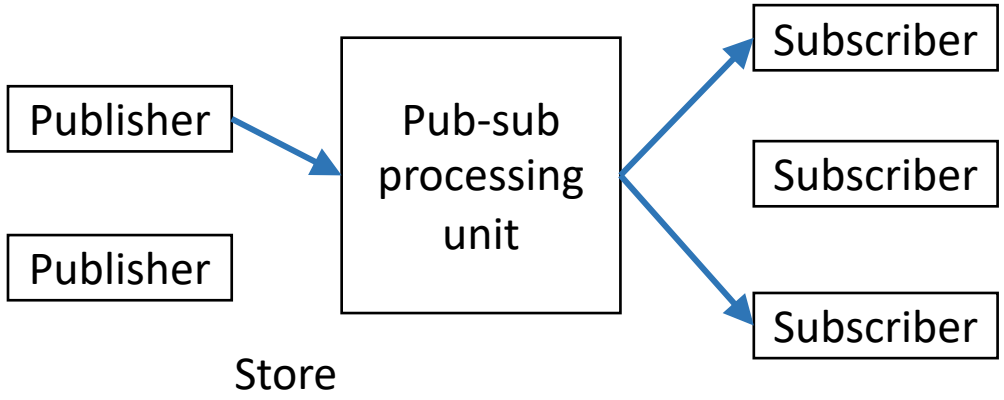
- Not all GPUs read pages written by others
- All-to-all store broadcasts waste BW
- Publish-subscribe model:
  - GPUs can subscribe to pages they read
  - Stores forwarded only to subscribers



Publish-subscribe model can save precious interconnect bandwidth

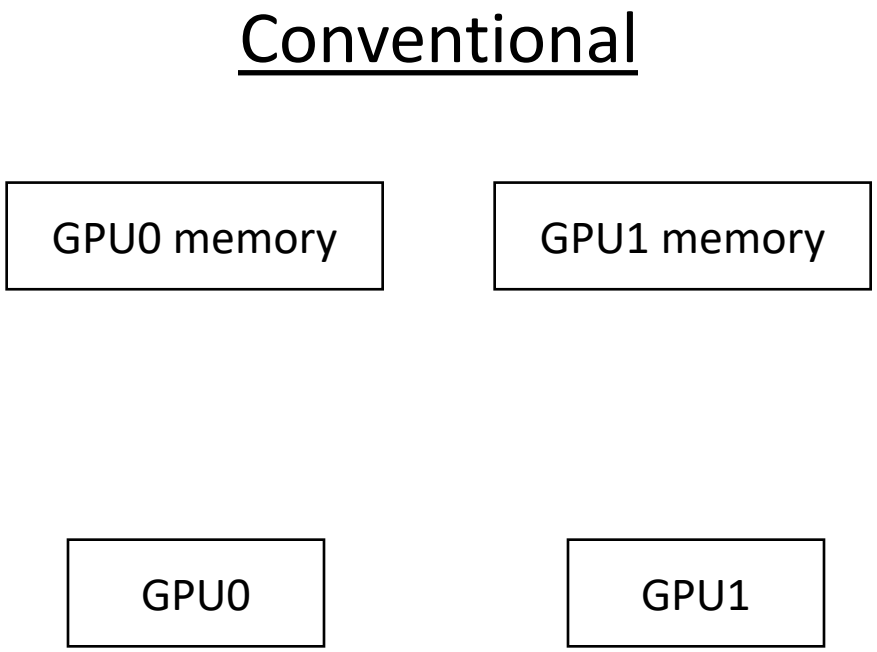
Publish-Subscribe Model

- Not all GPUs read pages written by others
- All-to-all store broadcasts waste BW
- Publish-subscribe model:
  - GPUs can subscribe to pages they read
  - Stores forwarded only to subscribers

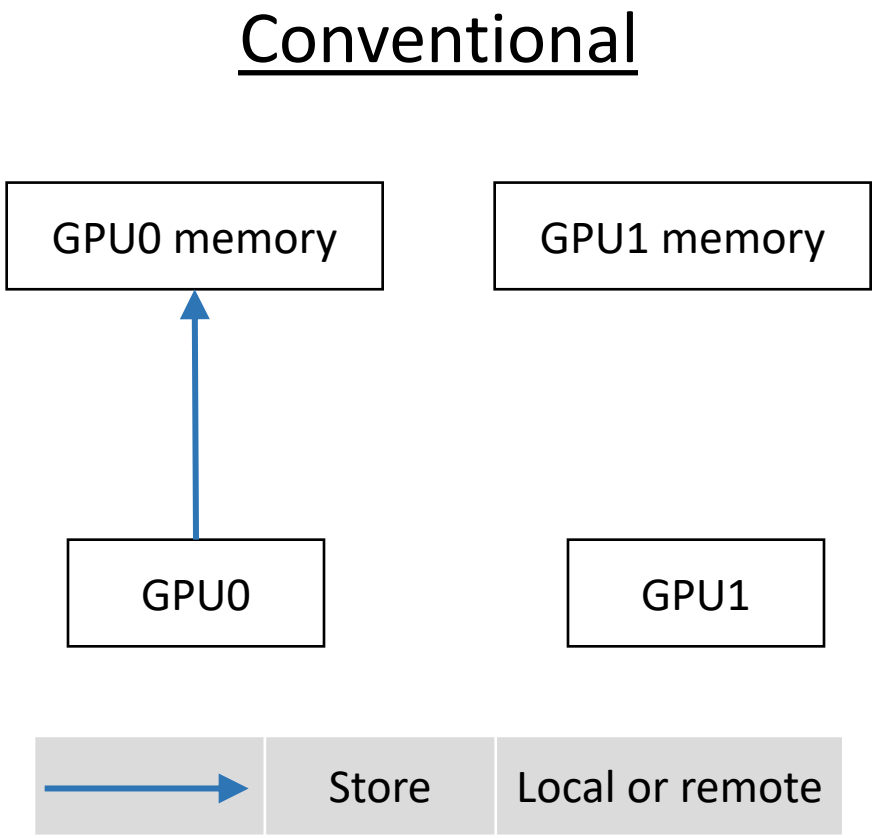


Publish-subscribe model can save precious interconnect bandwidth

Conventional vs. GPS Accesses

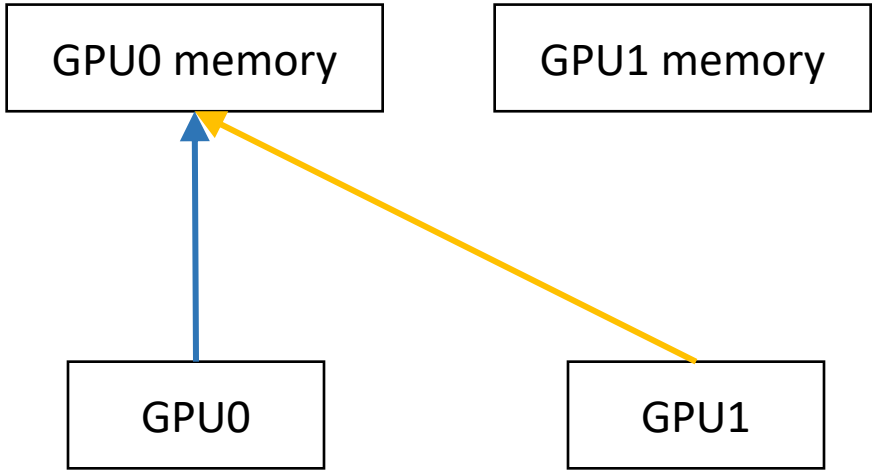


Conventional vs. GPS Accesses



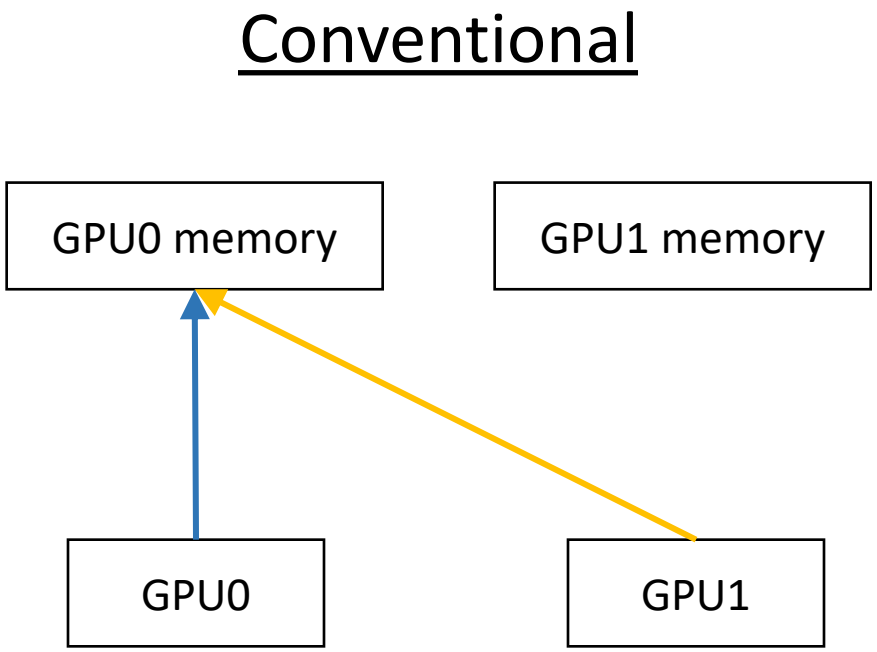
Conventional vs. GPS Accesses

Conventional

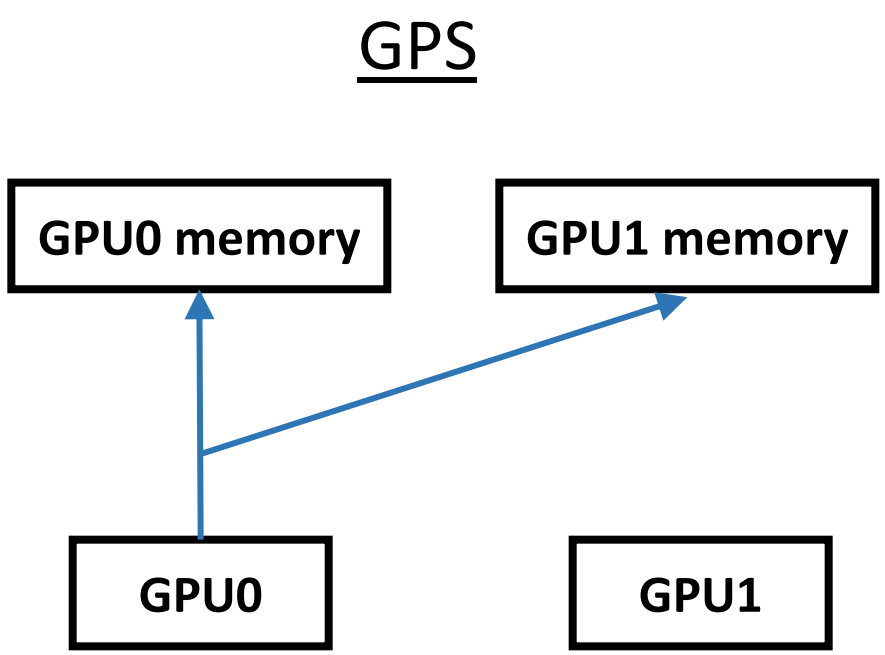


→	Store	Local or remote
→	Load	Local or remote

Conventional vs. GPS Accesses



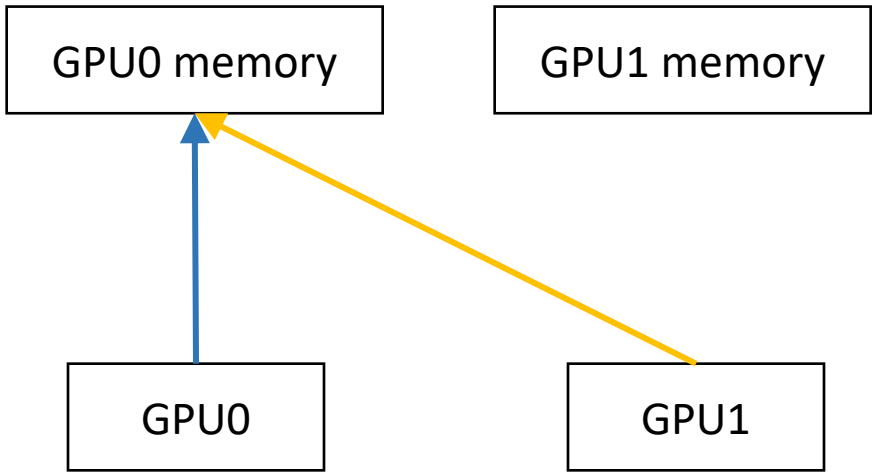
→	Store	Local or remote
→	Load	Local or remote



→	Store	Local AND remote

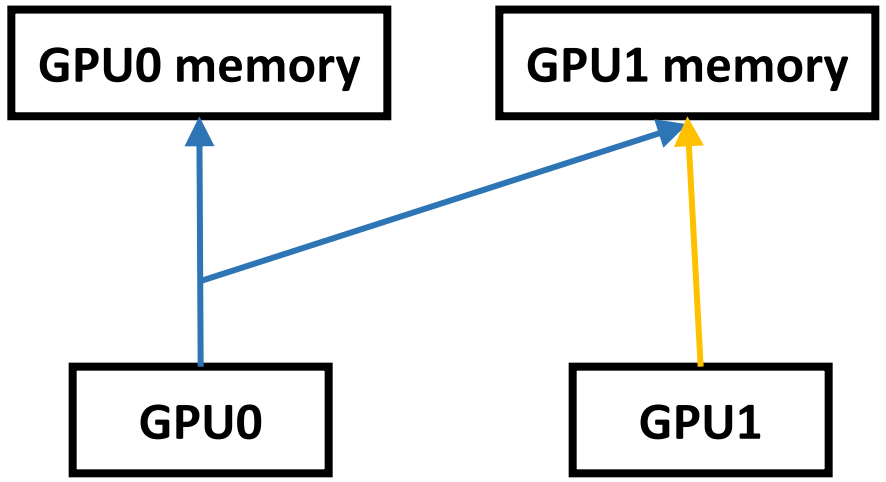
Conventional vs. GPS Accesses

Conventional



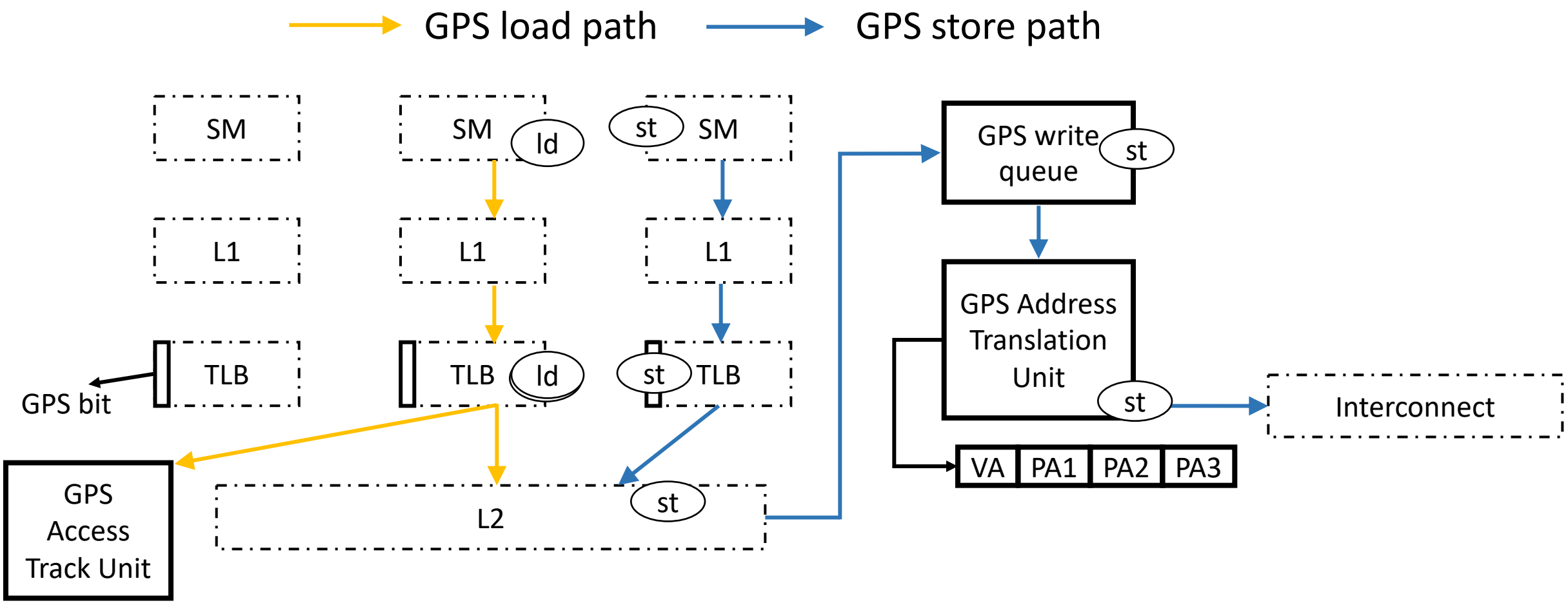
→	Store	Local or remote
→	Load	Local or remote

GPS

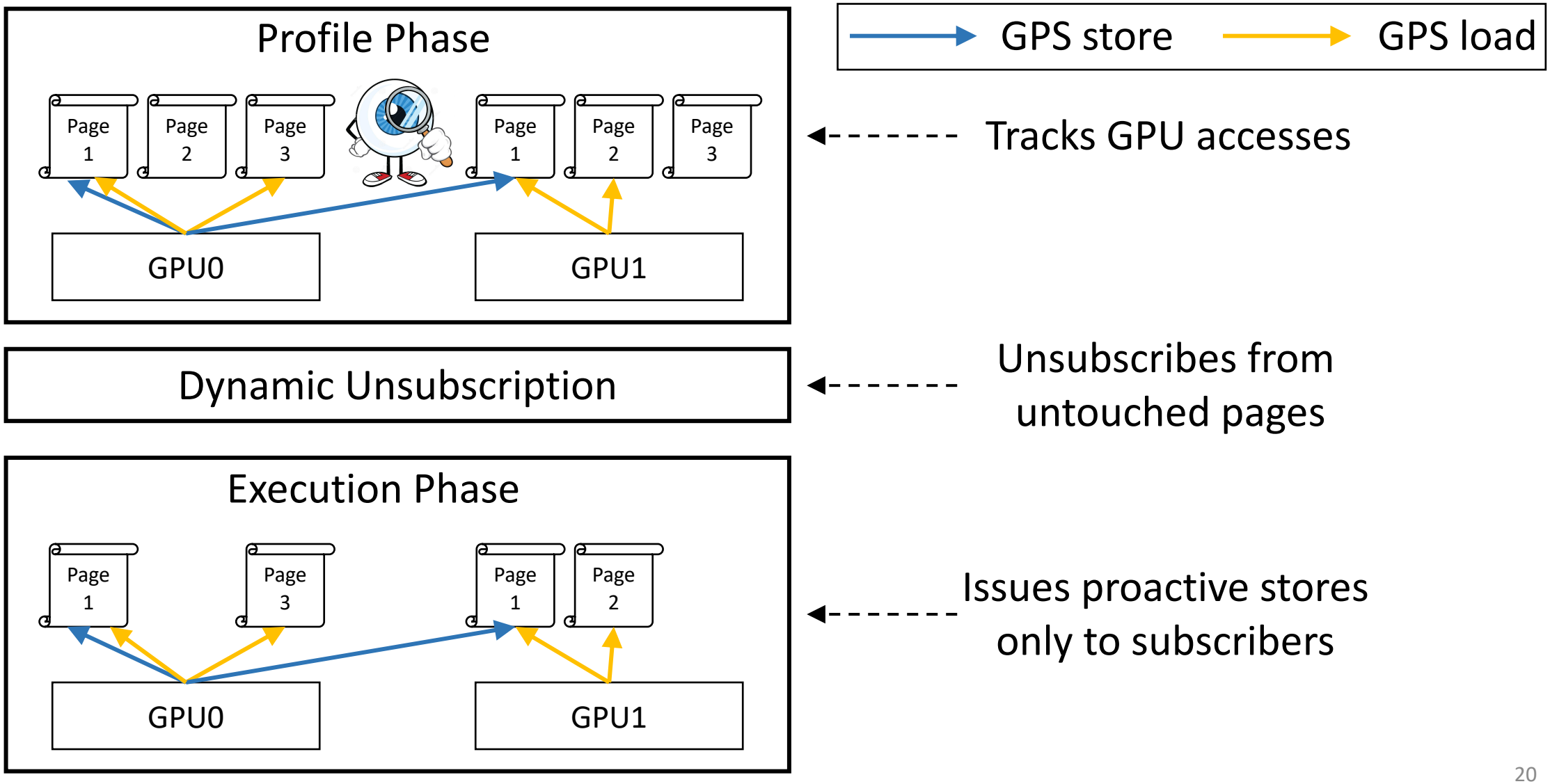


→	Store	Local AND remote
→	Load	Local

GPS Microarchitecture



Automatic Subscription Management



GPS Programming Interface

```
int main() {
    cudaMallocGPS(&mat, matsize);
    cudaMallocGPS(&vec1, vecsize);
    cudaMallocGPS(&vec2, vecsize);

    for(int iter=0; iter < MAX_ITER; iter++) {
        if(iter==0) cuGPSTrackingStart();
        for(int d=0; d < num_devices; d++) {
            cudaSetDevice(d);
            kernel<<<blocks, threads, stream[d]>>>(mat, vec1, vec2);
            kernel<<<blocks, threads, stream[d]>>>(mat, vec2, vec1);
        }
        if(iter==0) cuGPSTrackingStop();
    }
```

Allocates memory in the GPS VA

Start profile phase

End profile phase

Simple and intuitive extensions to Unified Memory for integrating GPS into applications

Evaluation Methodology

Simulation Framework

NVArchSim + NVBit

Interconnects

PCIe 3.0

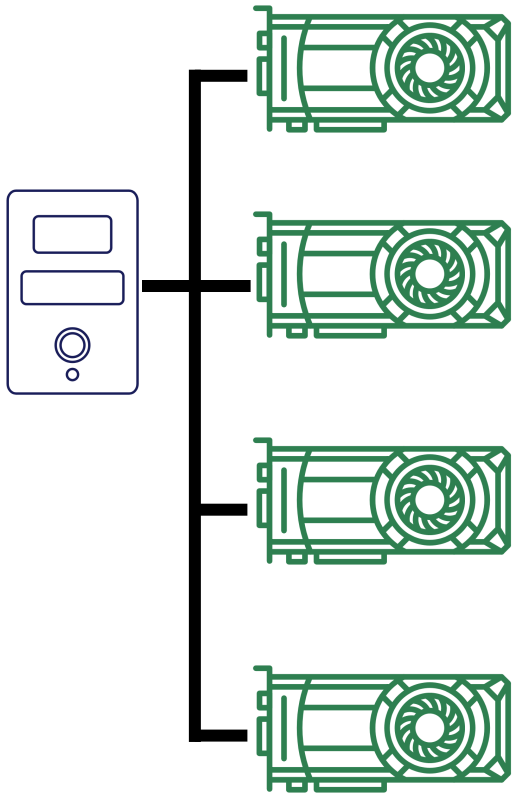
PCIe 4.0

PCIe 5.0

PCIe 6.0 (projected)

GPU Architecture

Volta



Workloads

Scientific computing

Medical imaging

Graph processing

Recommender systems

Number of GPUs

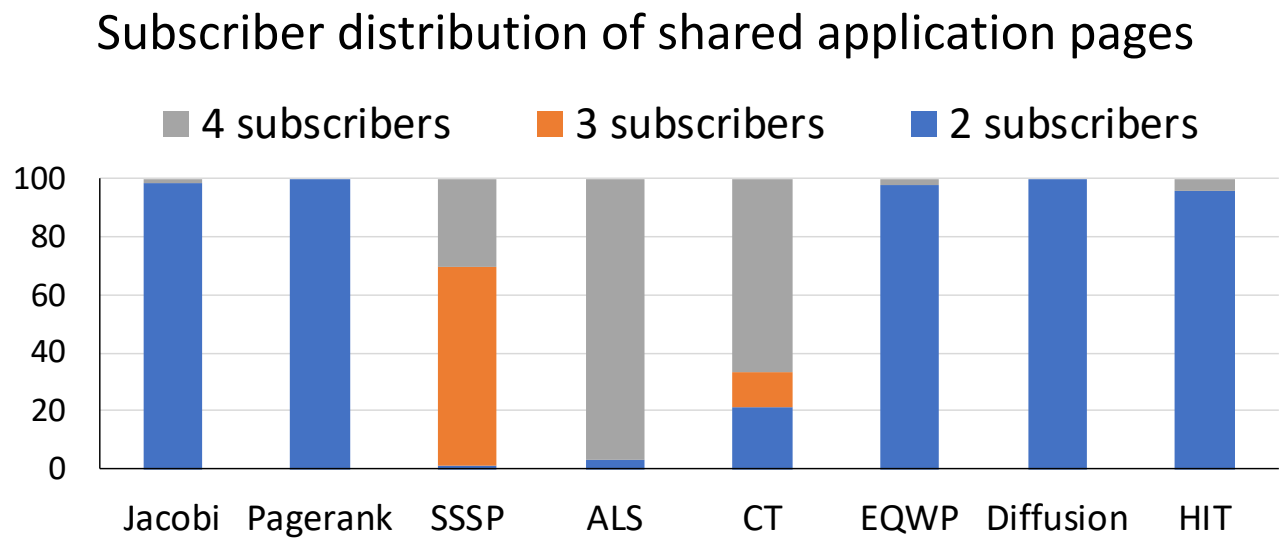
1,4,16

Evaluation: Programming Paradigms

Technique	Description
UM	Unified Memory with fault-based page migration
UM + hints	UM with hand-coded cudaMemAdvise hints
cudaMemcpy	cudaMemcpy only at kernel boundaries
Peer-to-peer loads	Fine-grained remote demand loads
GPS	**GPS with automatic subscription management**
Infinite Interconnect BW	All data is available locally without data transfer costs

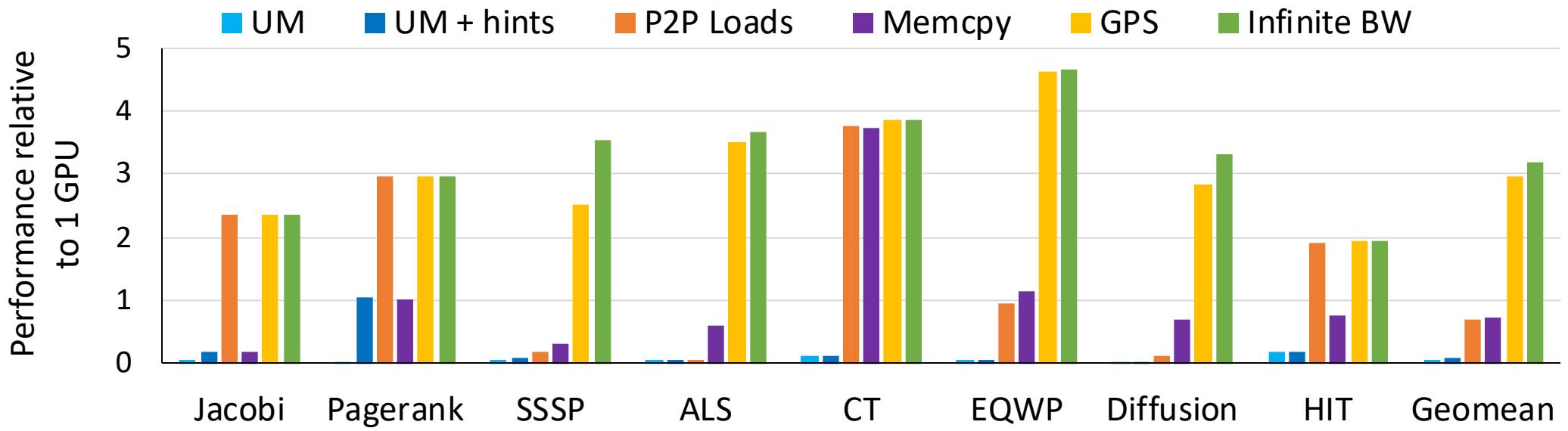
Subscription Benefits

- Subscription set varies across apps
- Automatic subscription leads to ~35% reduction in subscribed pages count

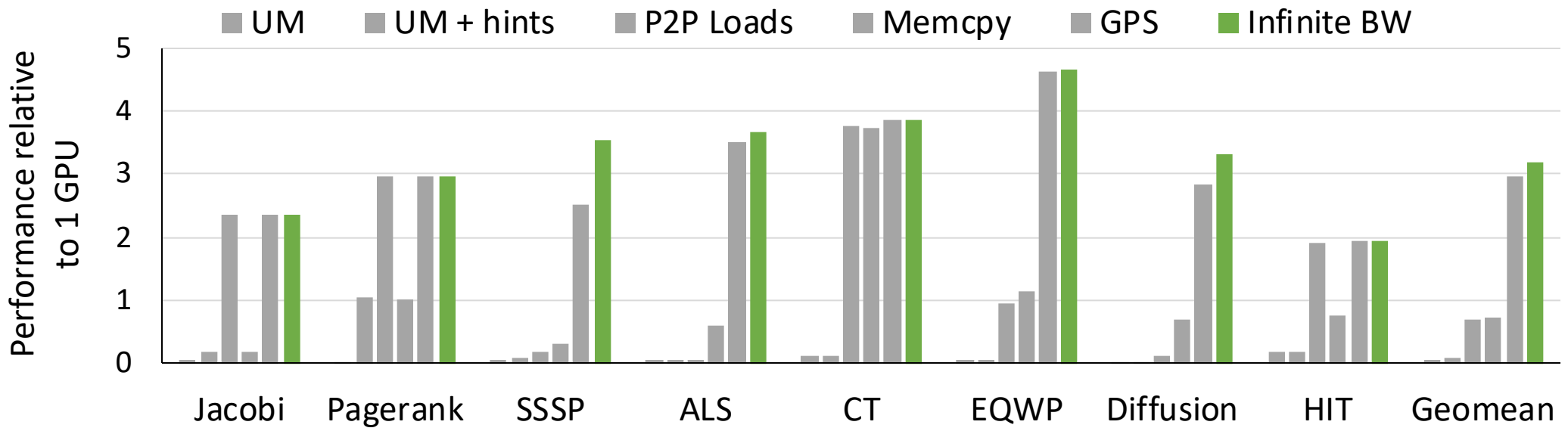


GPS subscriptions result in interconnect bandwidth savings for all pages with less than 4 subscribers

4-GPU Speedups

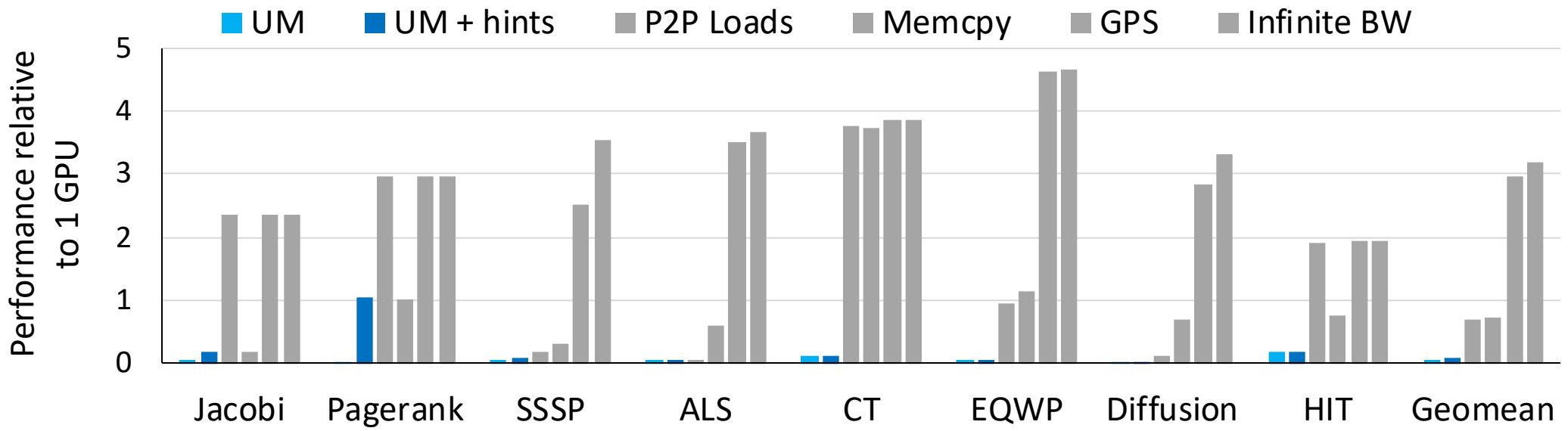


4-GPU Speedups



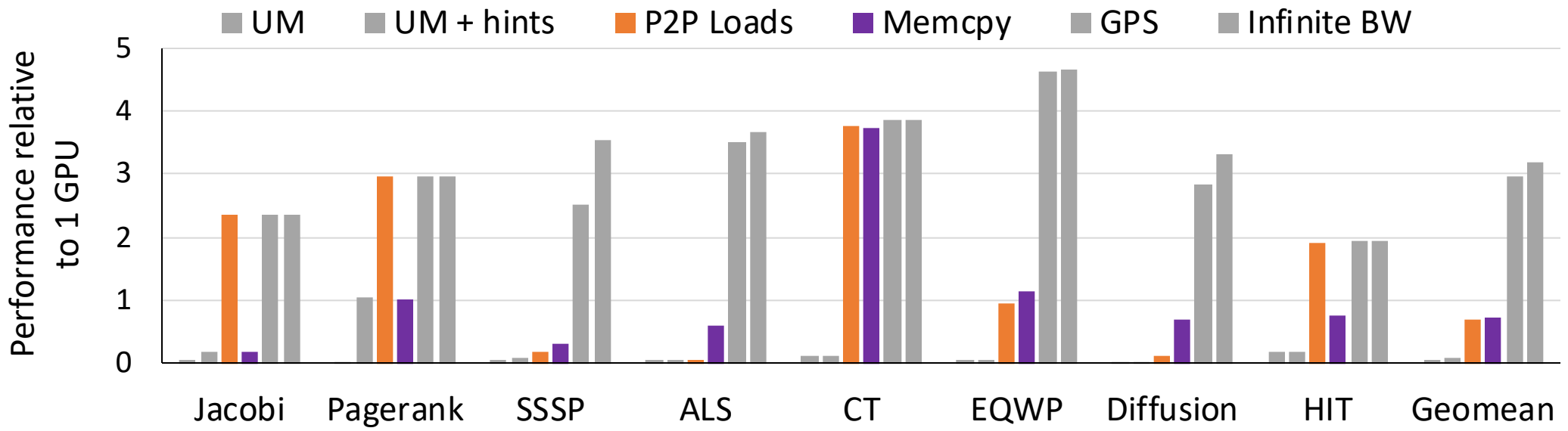
Infinite BW shows near-linear speedups demonstrating the highly parallel nature of the applications

4-GPU Speedups



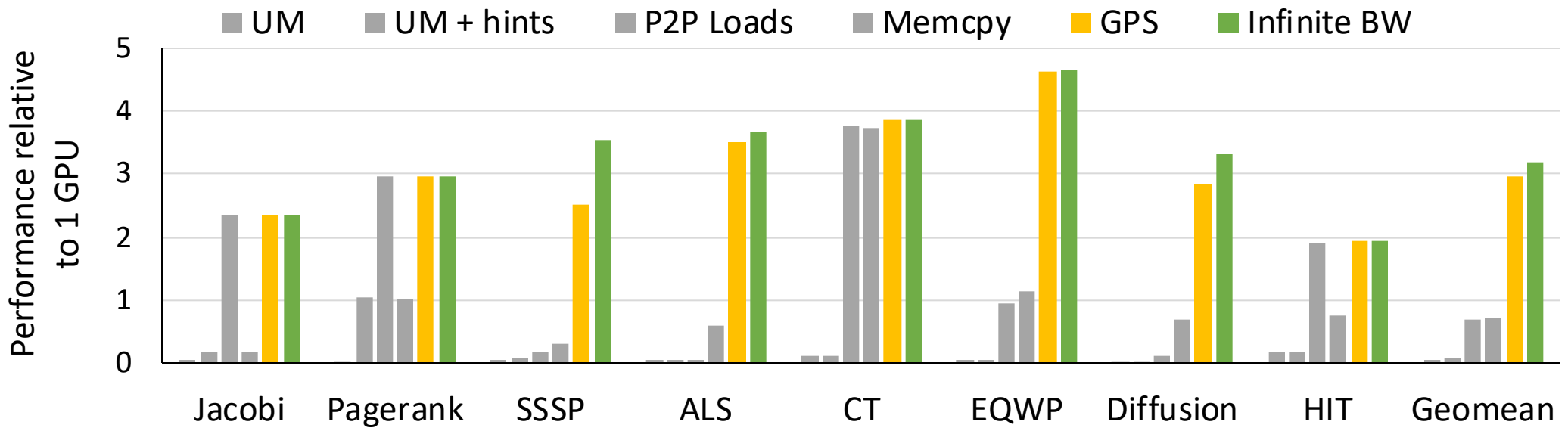
Unified Memory underperforms 1 GPU across applications

4-GPU Speedups



P2P loads and Memcpy scale well for a few applications but perform poorly on average

4-GPU Speedups



GPS scales well across all applications capturing 94% of the available opportunity

Additional Data In The Paper…

- Effectiveness of unsubscription mechanism
  - Enables significant reduction in total data moved over interconnect
- Sensitivity to system configuration
  - GPS improves strong scaling across GPU counts and interconnect architectures
- Sensitivity to GPS microarchitectural parameters
  - Chip area required to implement GPS components is minimal

Conclusion

- Multi-GPU strong scaling is bound by interconnect BW
  - Efficient bulk DMA transfers are hard to achieve in practice
  - Peer-to-peer transfers are efficient only if locality can be carefully managed
- GPS performs intelligent HW/SW-based memory management
  - Selective page replication + proactive remote stores improve read locality
  - Achieves 3x and 7.9x over 1 GPU on 4 and 16 GPU systems respectively
  - Provides a new pathway to future multi-GPU performance scalability

GPS: A Global Publish-Subscribe Model for Multi-GPU Memory Management

Harini Muthukrishnan (University of Michigan), Daniel Lustig (NVIDIA),
David Nellans (NVIDIA), Thomas Wenisch (University of Michigan)

Contact: harinim@umich.edu