Pump Up the Volume: Processing Large Data on GPUs with Fast Interconnects

Clemens Lutz, Sebastian Breß, Steffen Zeuch, Tilmann Rabl, Volker Markl

TU berlin

DFKI German Research Center for Artificial Intelligence

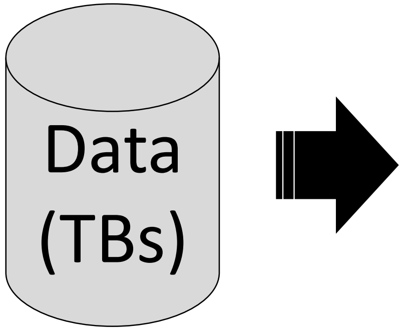
HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

Goal

*Scale GPU-accelerated data management to **arbitrary data volumes**.*

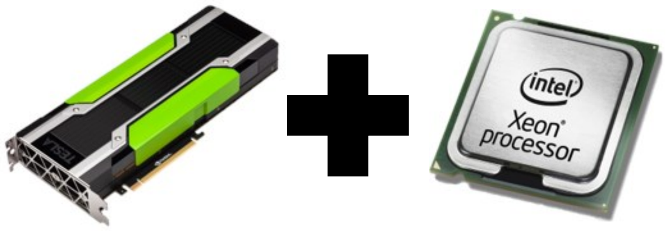
Goal

Scale GPU-accelerated data management to ***arbitrary data volumes****.*



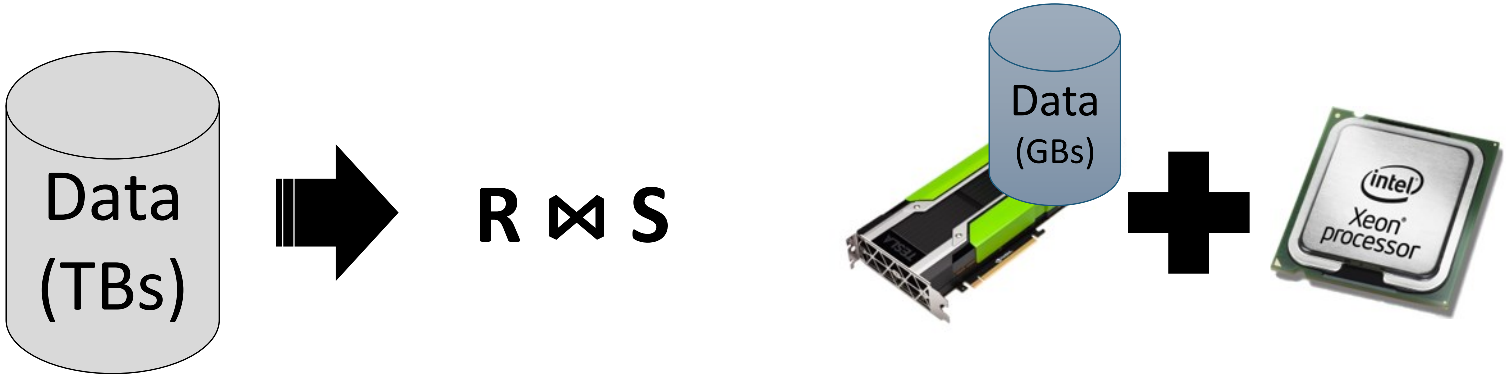
Data (TBs)

R ⋈ S



Goal

Scale GPU-accelerated data management to ***arbitrary data volumes.***

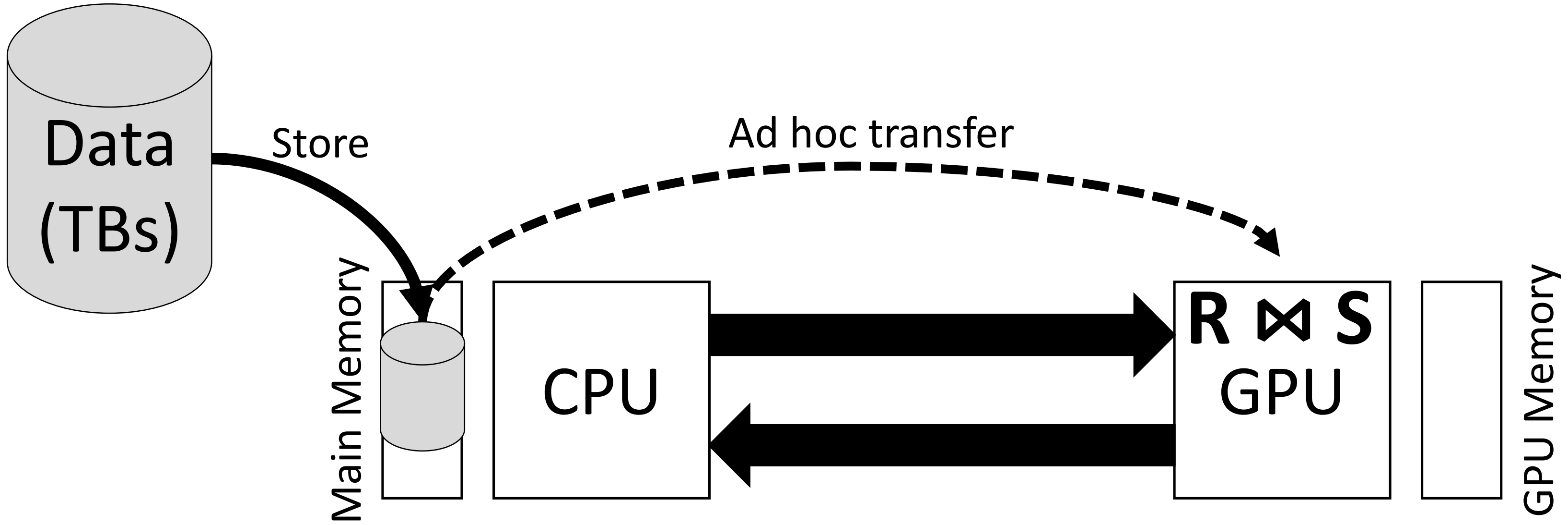


Data (TBs)

R ⋈ S

Data (GBs)

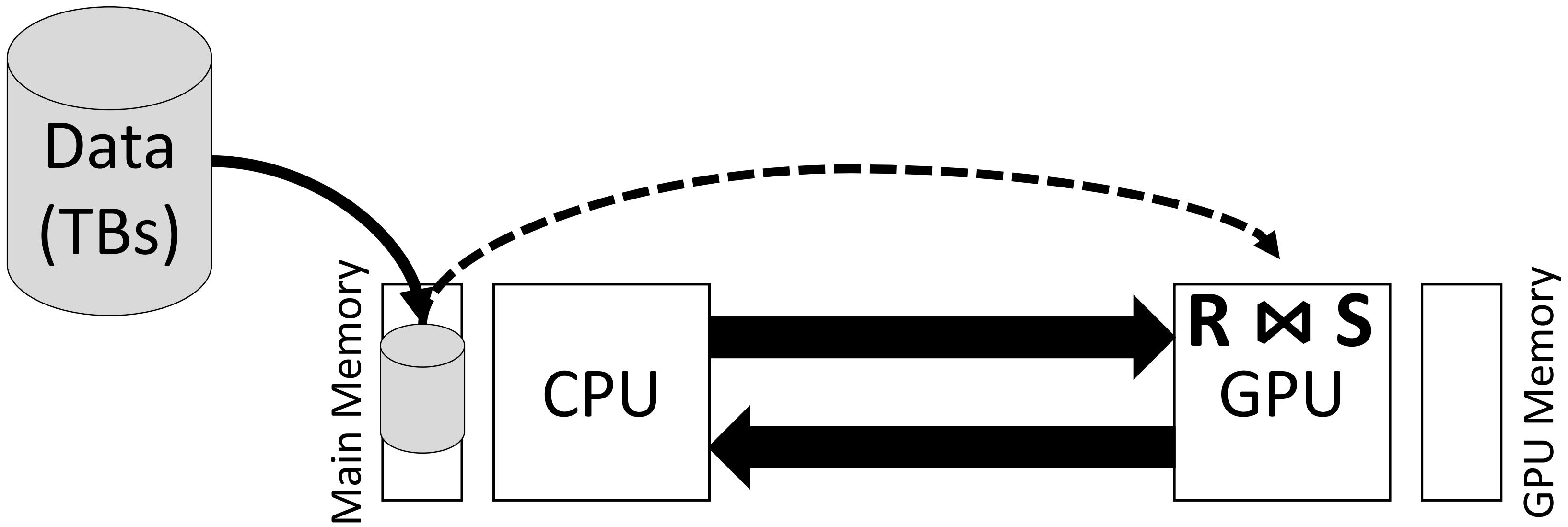
Problem 1: Transfer Bandwidth



Today’s GPU databases:

- Store data in main memory
- Perform data processing on GPU

Problem 1: Transfer Bandwidth



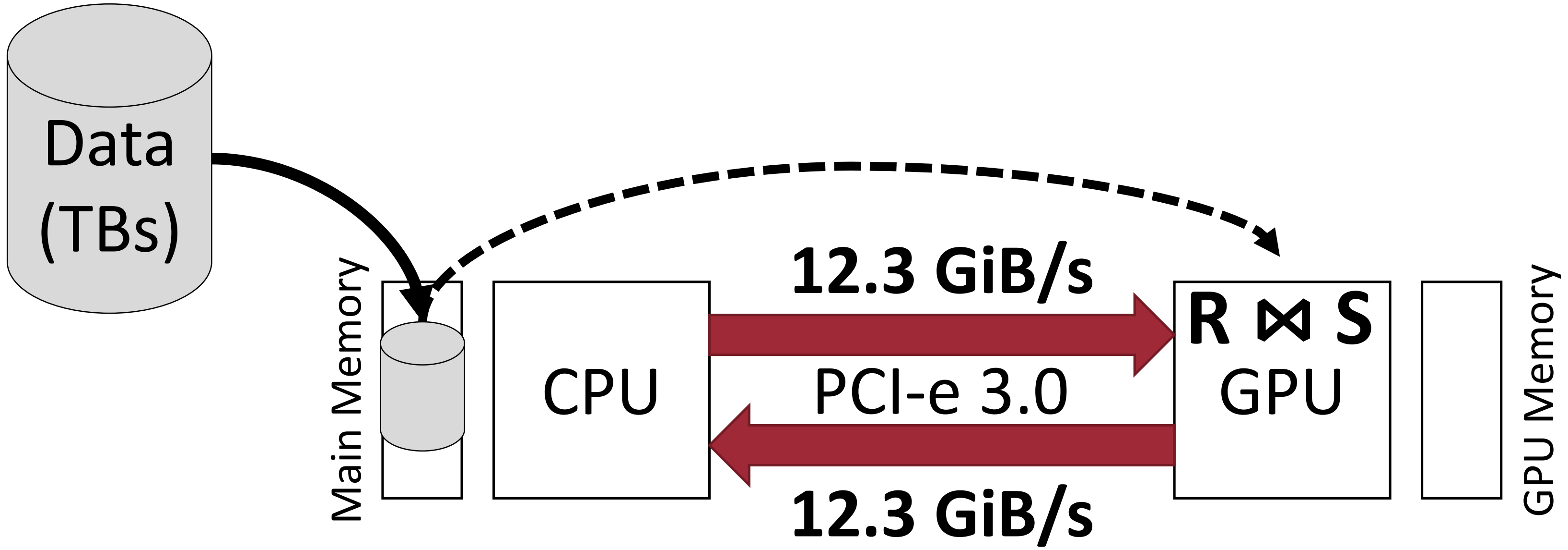
blazingSQL

omni·sci

kinetica

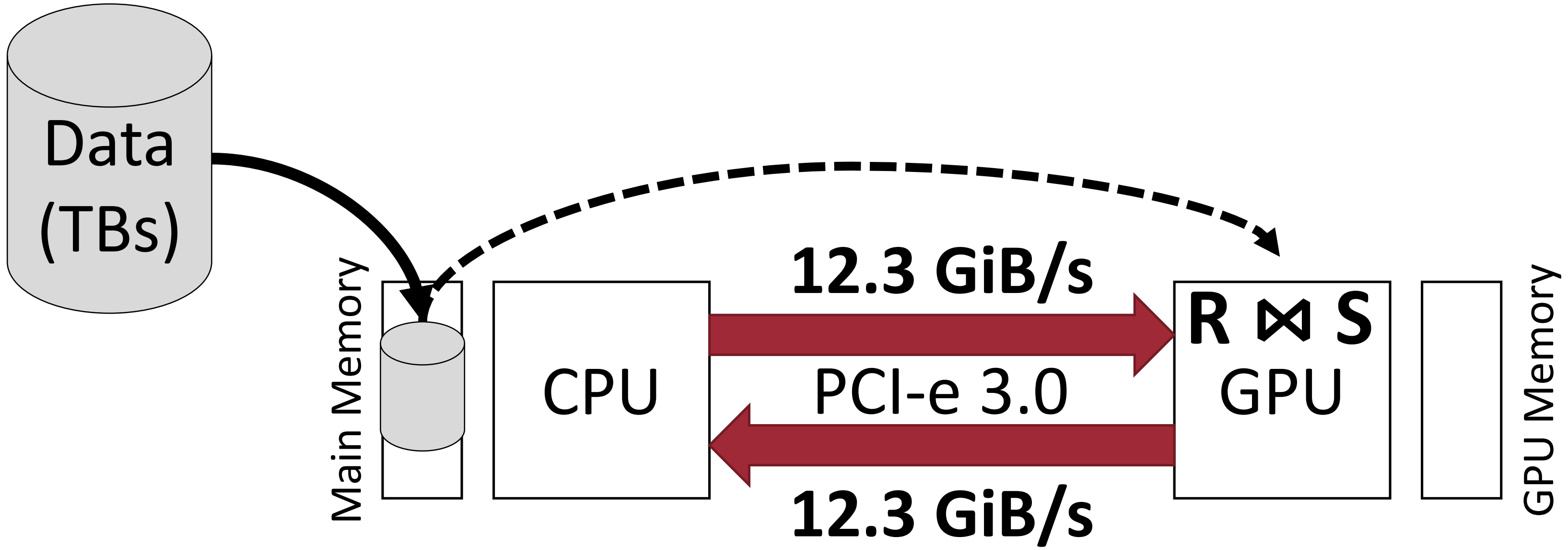
SQREAM

Problem 1: Transfer Bandwidth



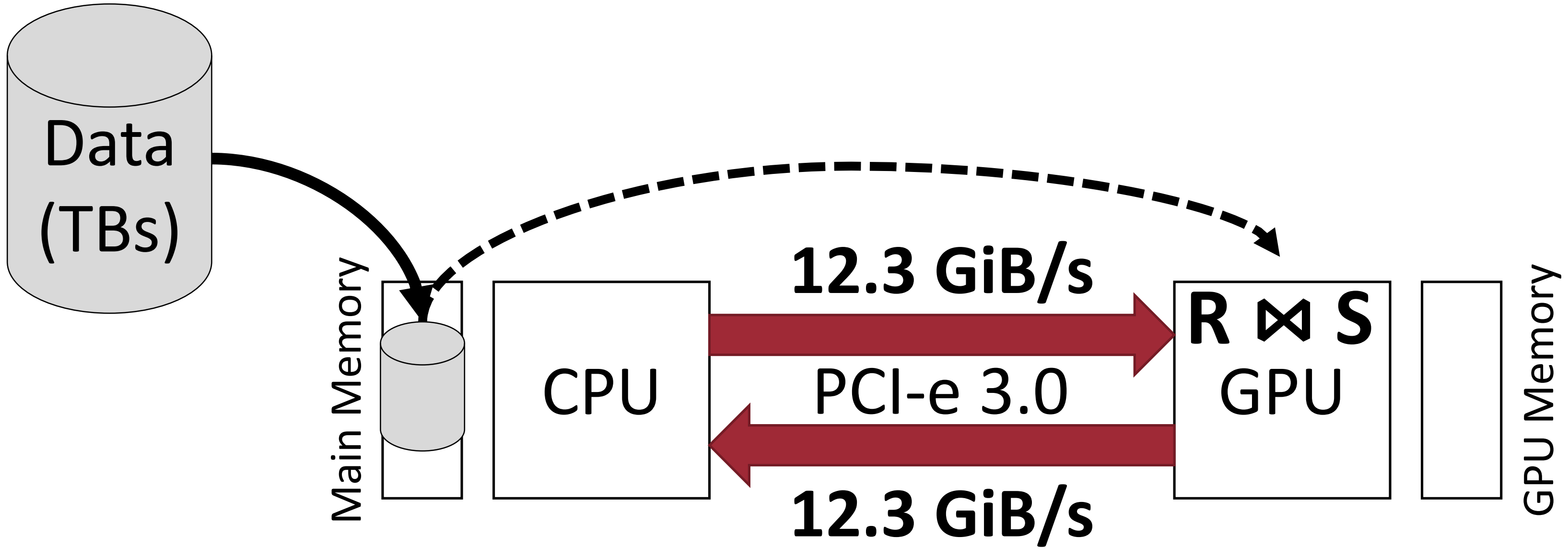
- Ad hoc data transfer over PCI-e 3.0
- GPU capable of much higher throughput

Problem 1: Transfer Bandwidth



Interconnect bandwidth & GPU memory capacity limit scalability

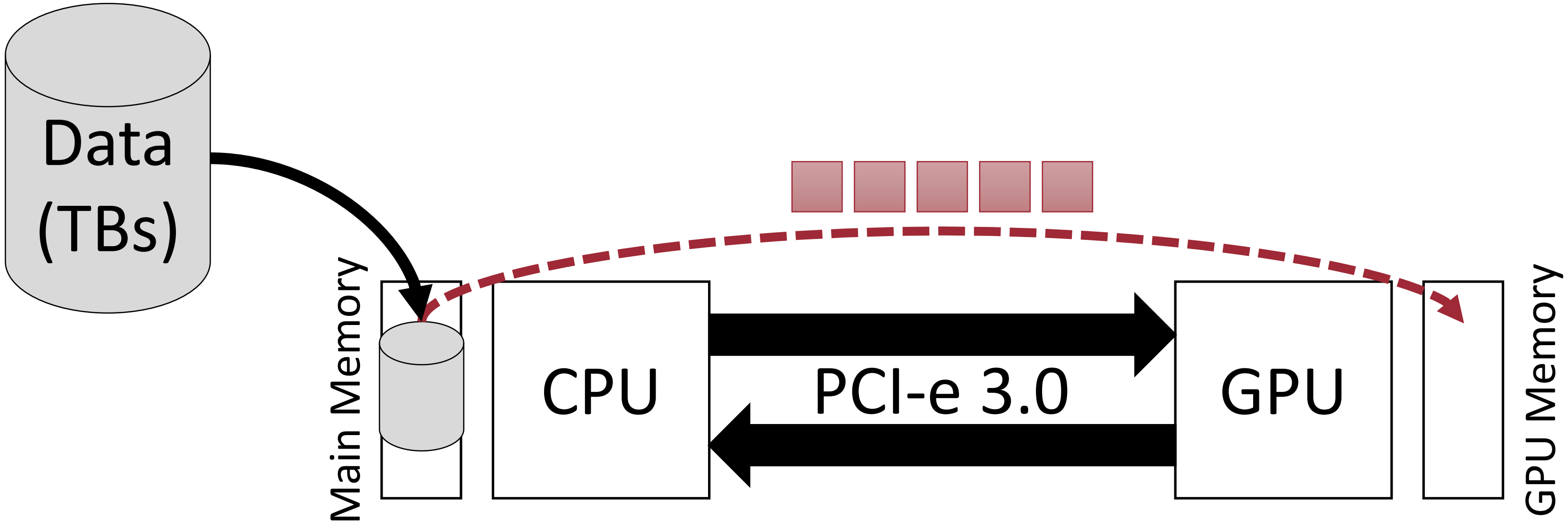
Problem 1: Transfer Bandwidth



Interconnect bandwidth & GPU memory capacity limit scalability

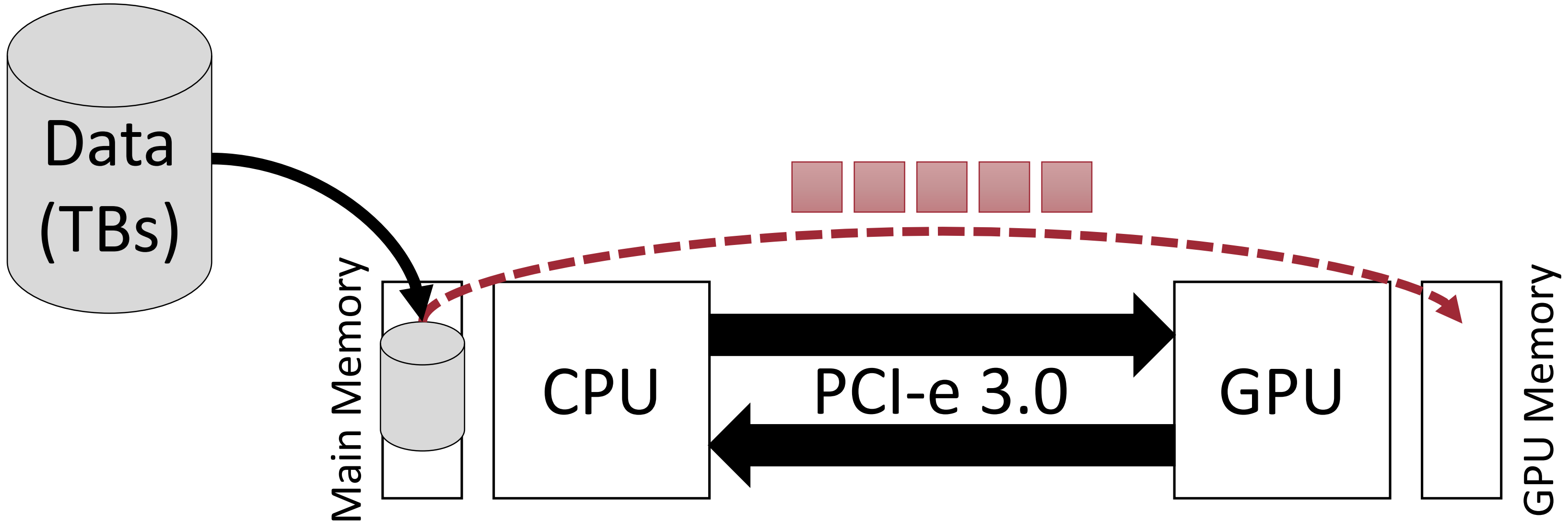
“Transfer bottleneck”

Problem 2: Course-Grained Cooperation



- Data-dependent memory accesses not possible
- Fine-grained CPU+GPU cooperation not possible

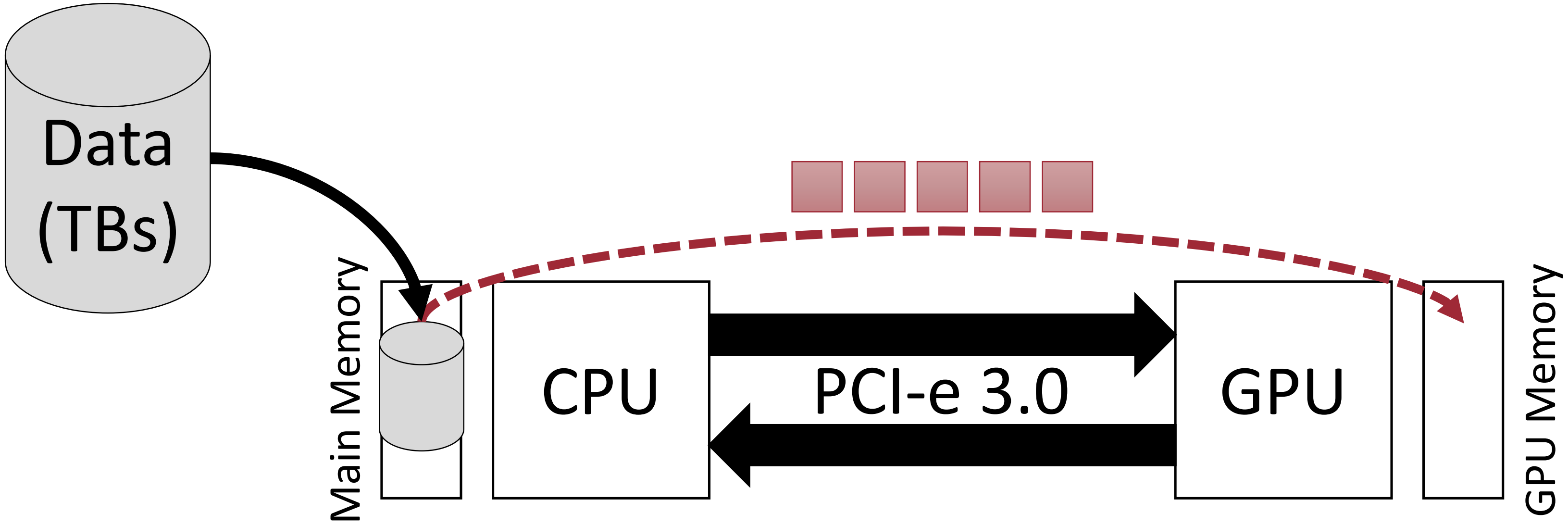
Problem 2: Course-Grained Cooperation



- Data-dependent memory accesses not possible
- Fine-grained CPU+GPU cooperation not possible

} Data structures

Problem 2: Course-Grained Cooperation



Non-cache coherence limits design space & co-processing

Game Changer

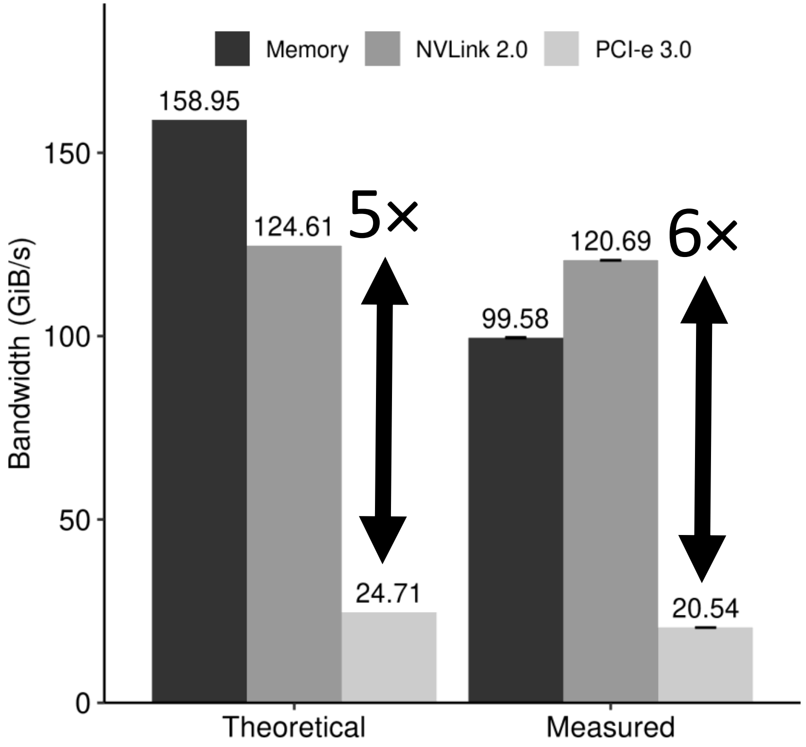
- Fast interconnects

  e.g., NVLink 2.0, Infinity Fabric, CXL

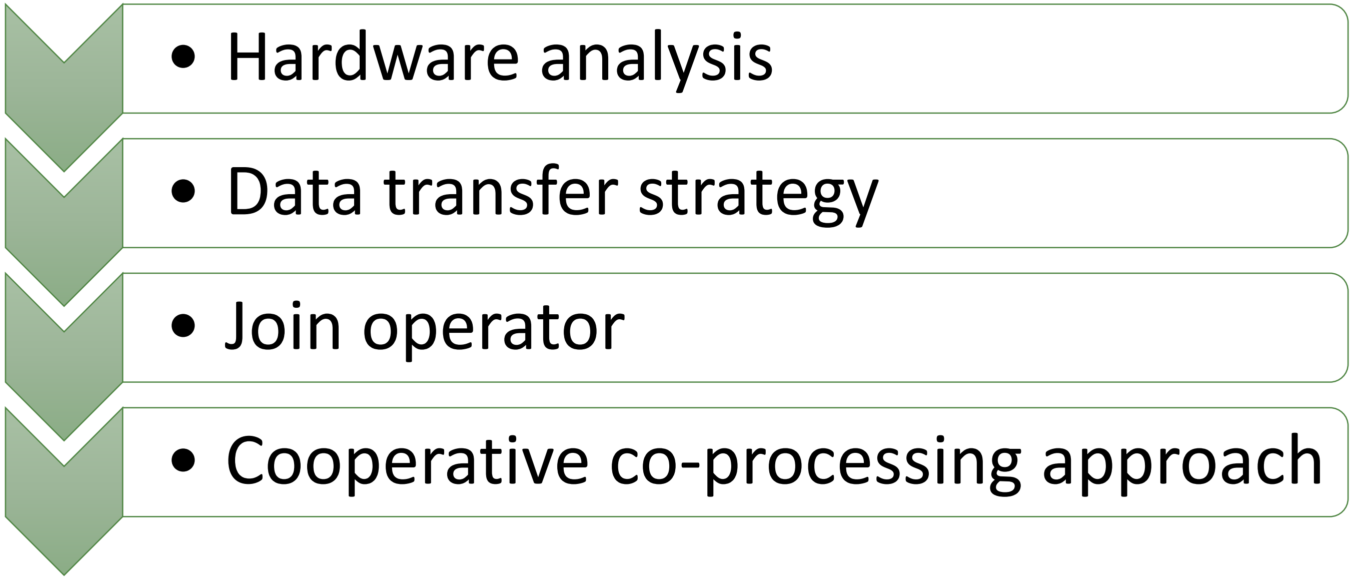
- High bandwidth (124 GiB/s total)
- System-wide cache-coherence

  - data-dependent memory access

  - fine-grained CPU+GPU cooperation

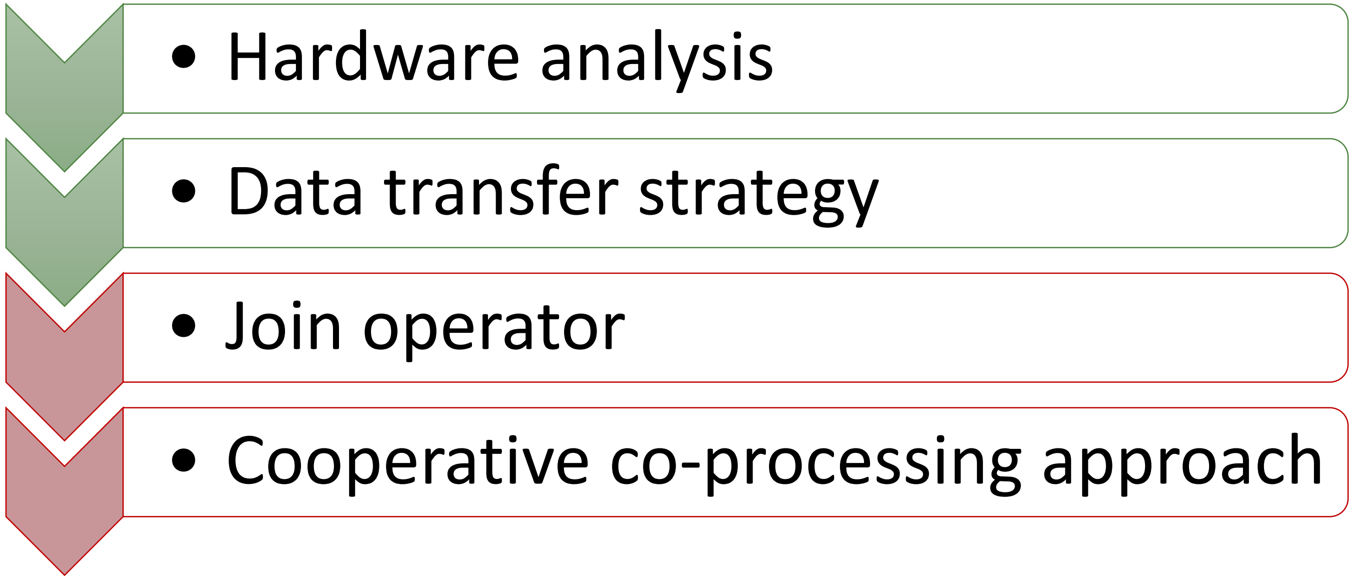


Contributions



- Hardware analysis
- Data transfer strategy
- Join operator
- Cooperative co-processing approach

Contributions



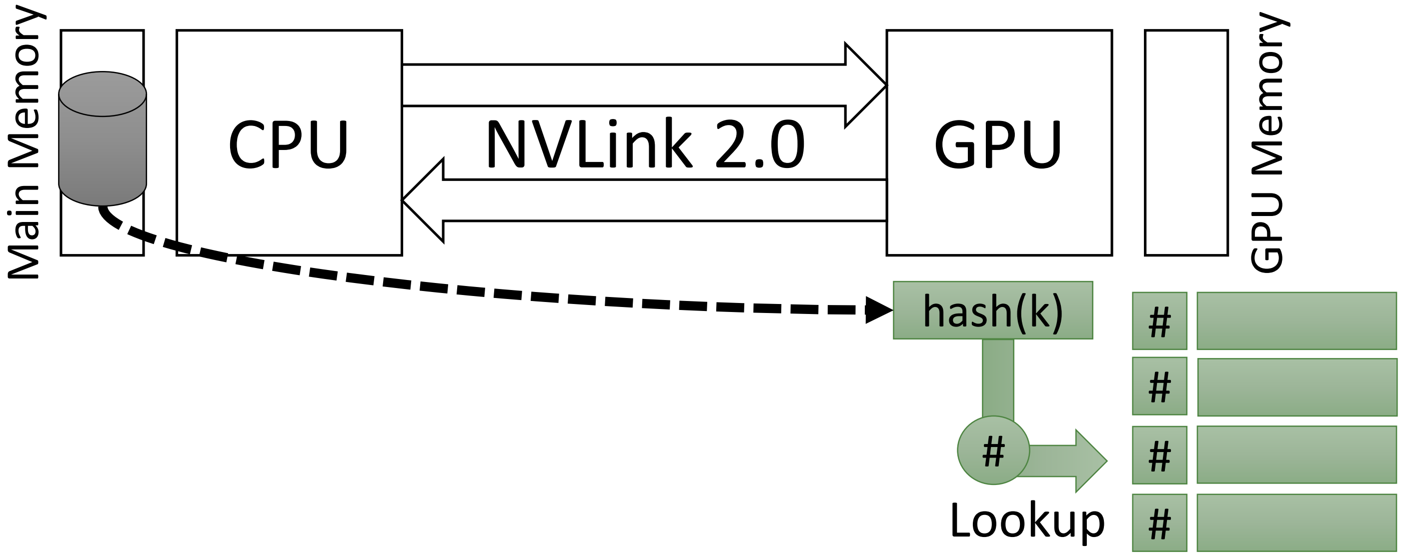
- Hardware analysis
- Data transfer strategy
- Join operator
- Cooperative co-processing approach

Solution

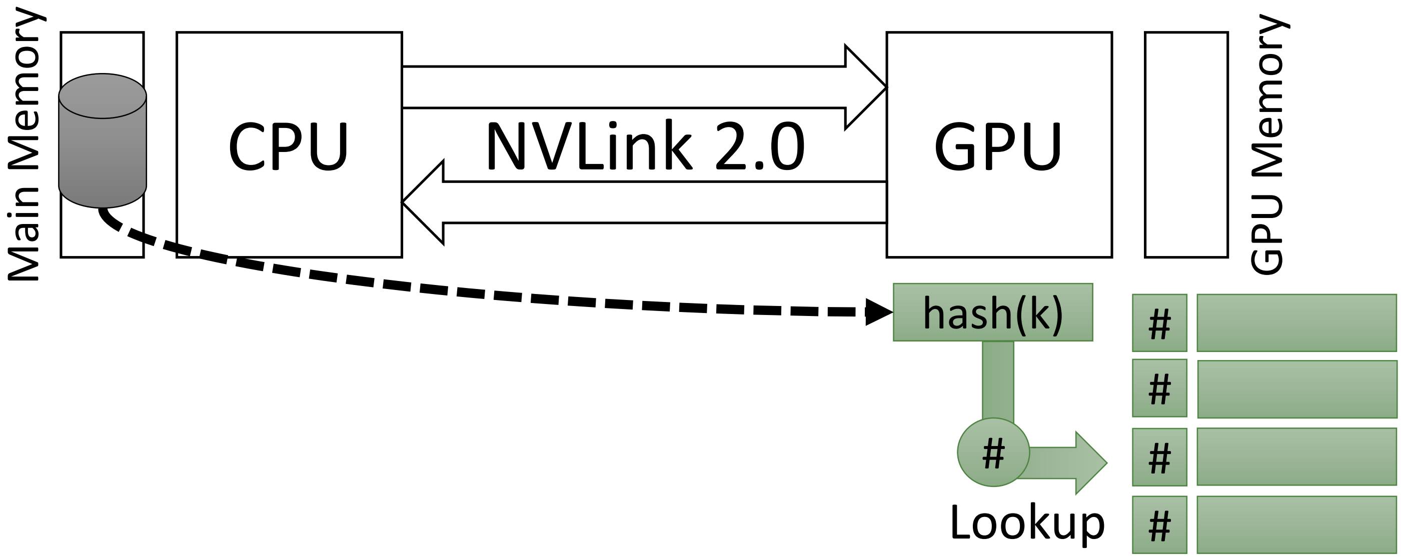
Hash Join

- Probe-side scaling
- Build-side scaling
- GPU+CPU Cooperation

Probe-Side Scaling

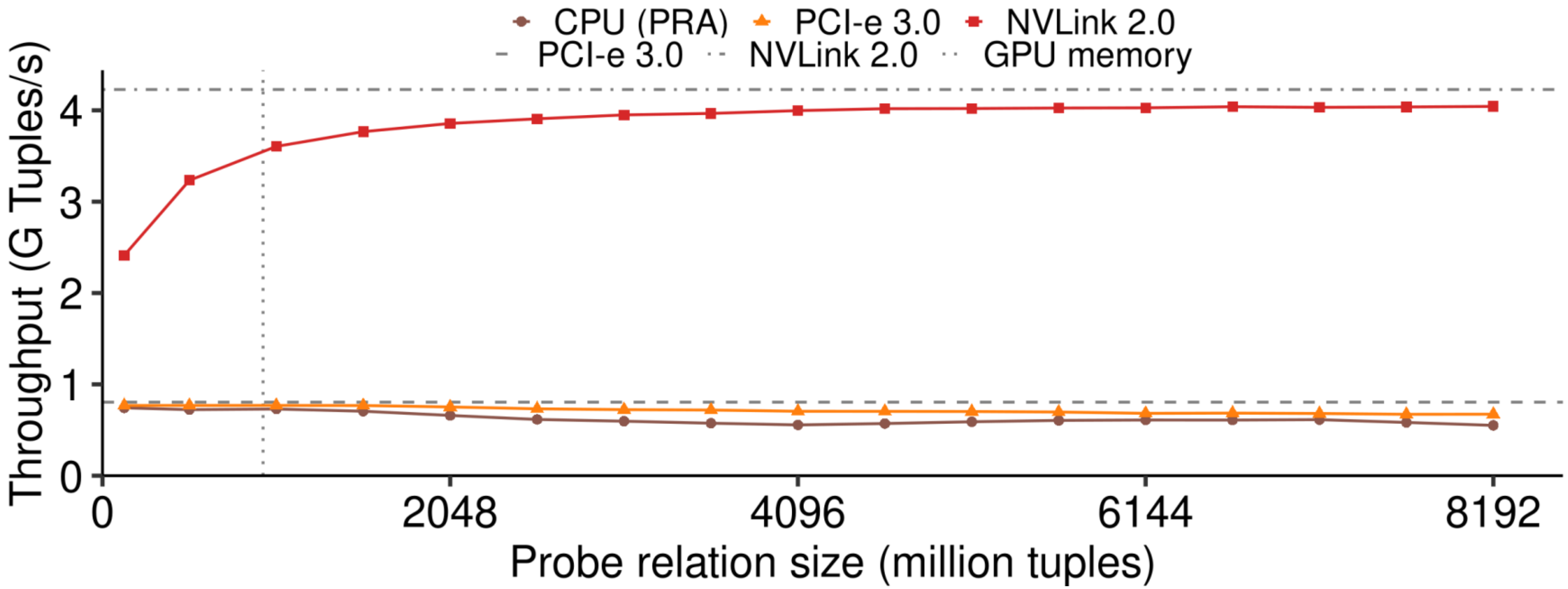


Probe-Side Scaling



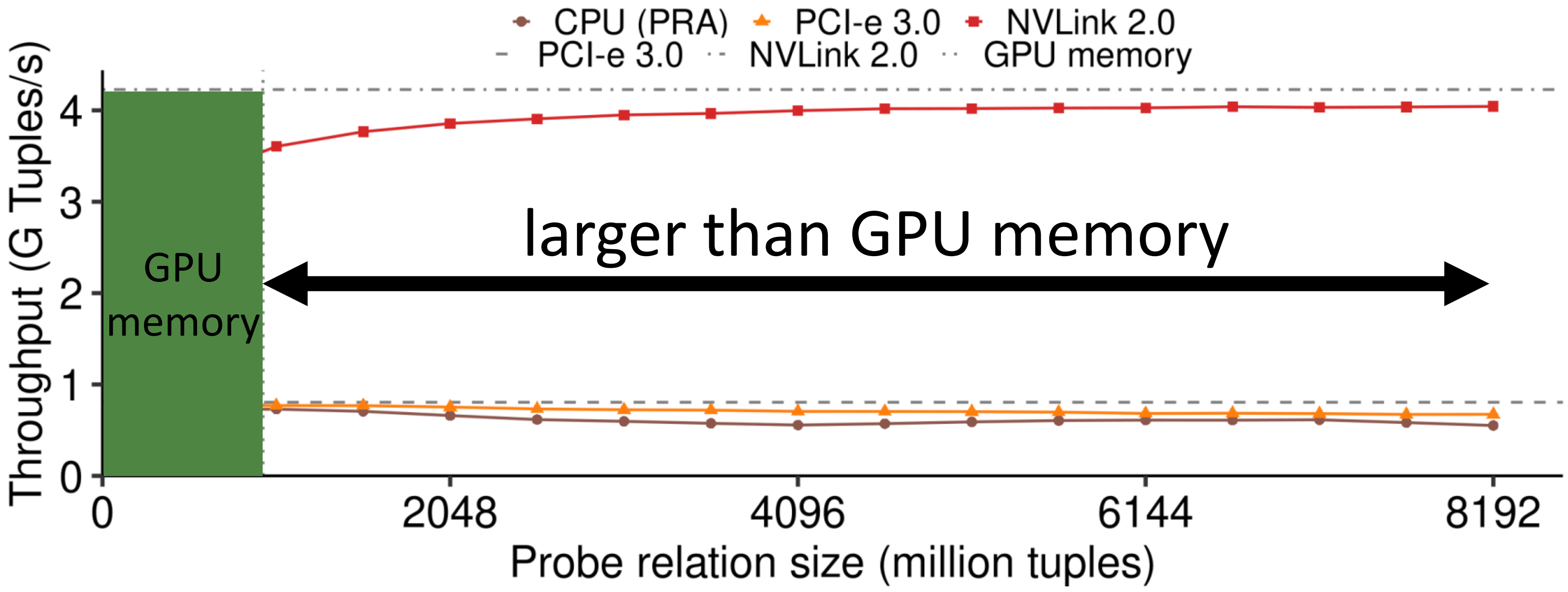
Interconnect feature: High bandwidth

Probe-Side Scaling



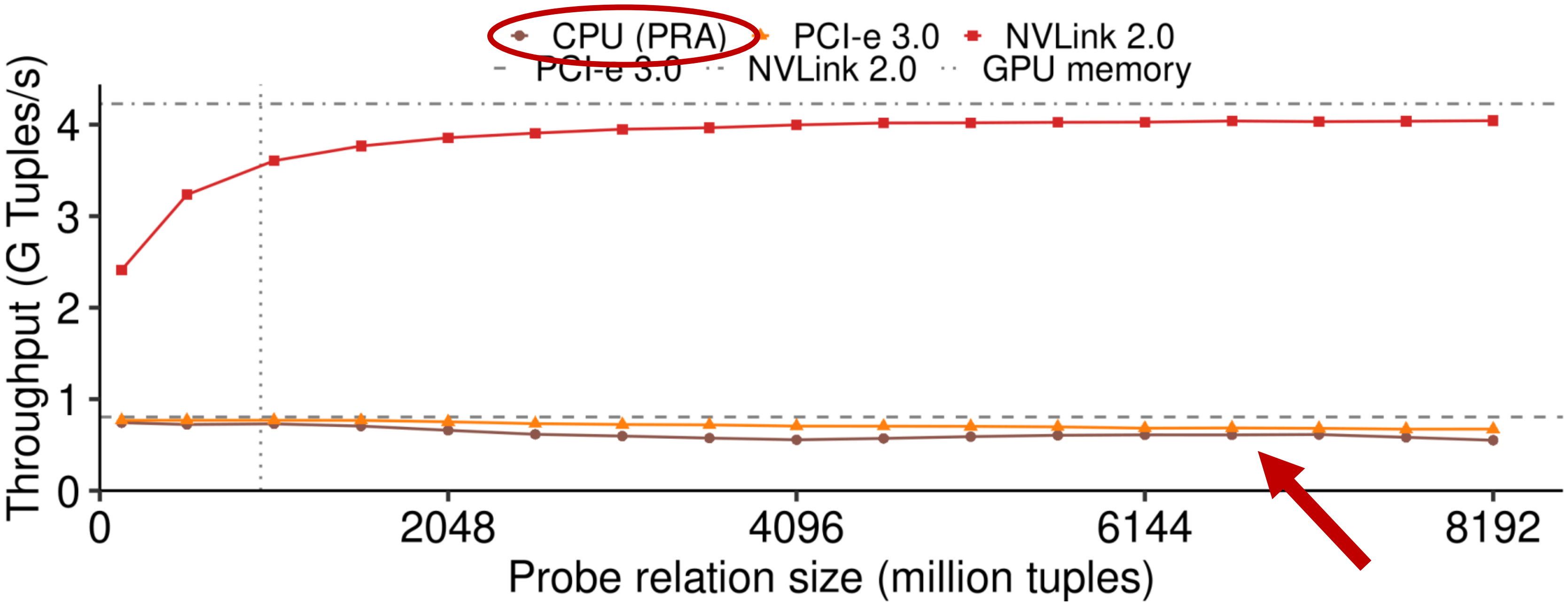
- Up to 2⋈122 GiB

Probe-Side Scaling



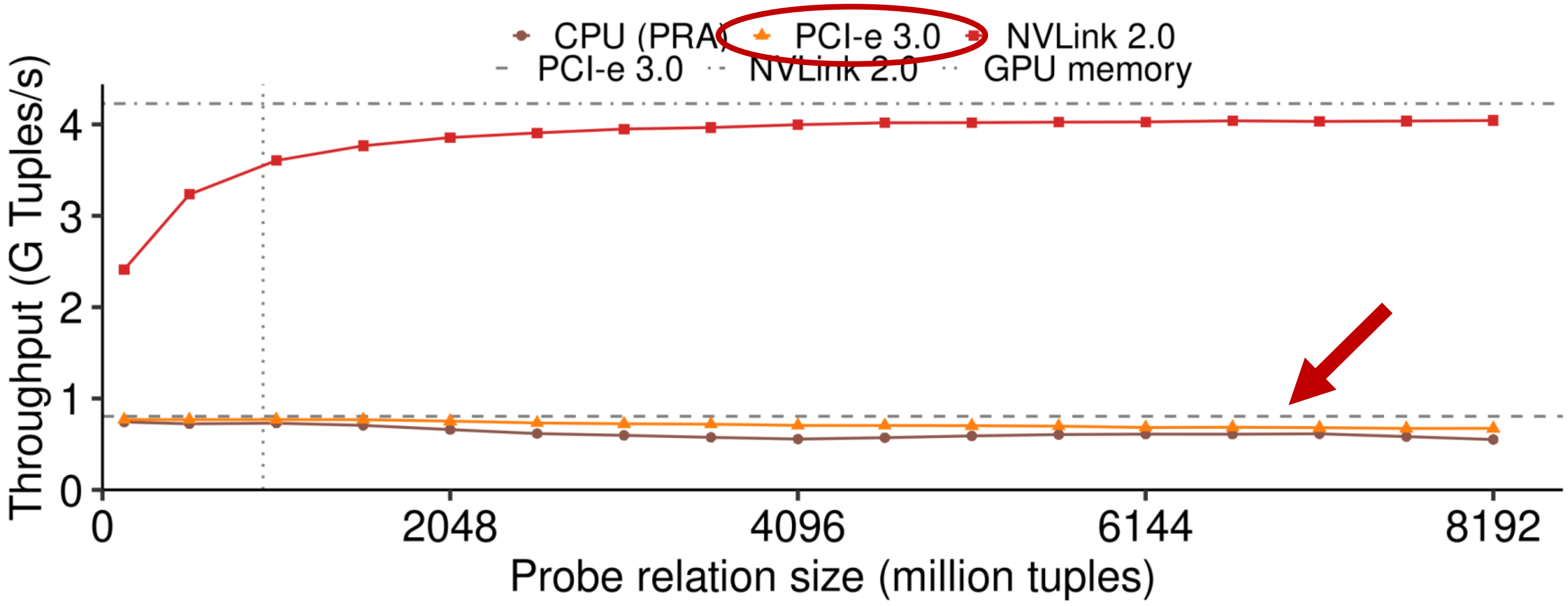
- Up to 2⋈122 GiB

Probe-Side Scaling



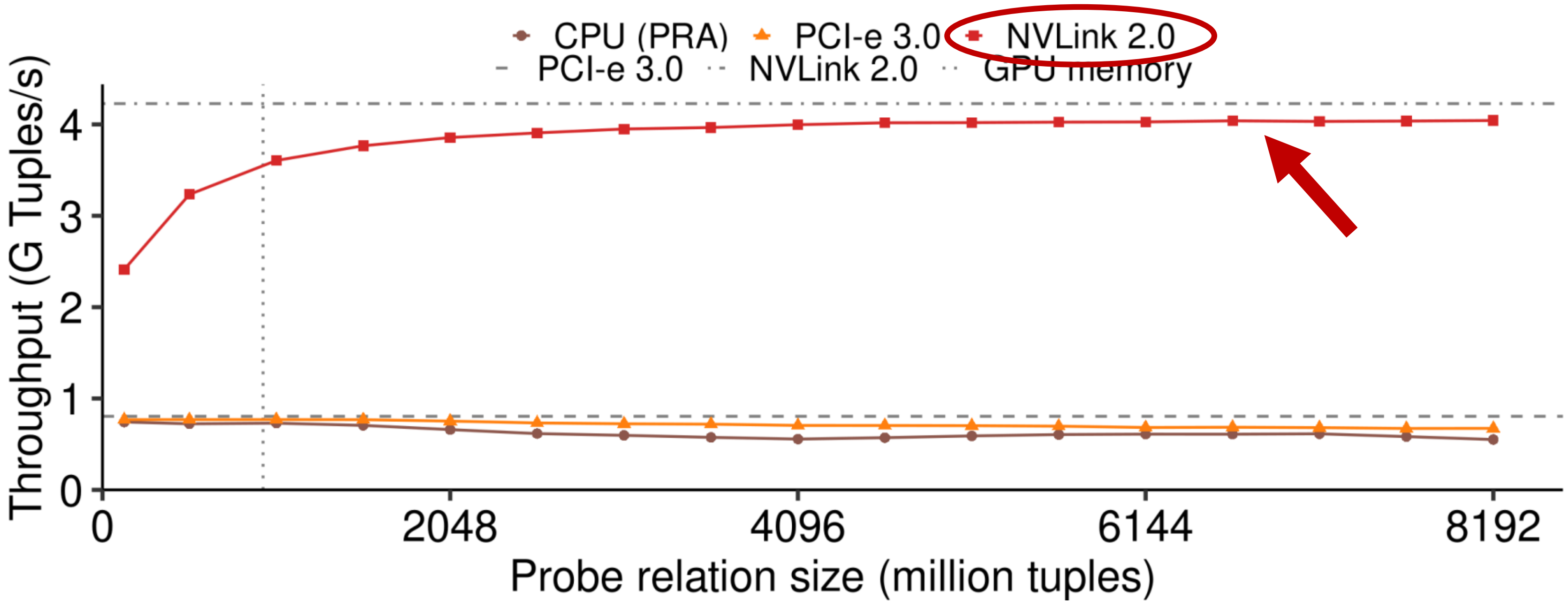
- Up to 2⋈122 GiB
- CPU baseline: Radix-partitioned hash join

Probe-Side Scaling



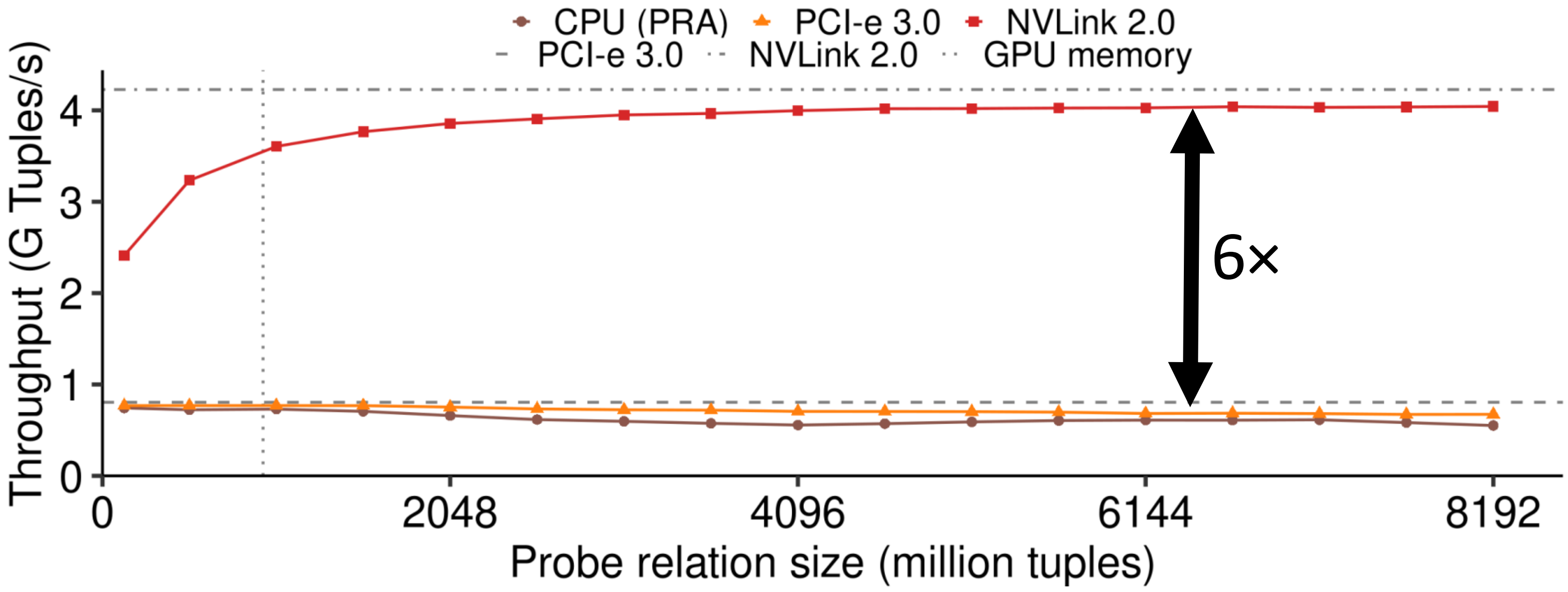
- Up to 2⋈122 GiB
- CPU baseline: Radix-partitioned hash join

Probe-Side Scaling



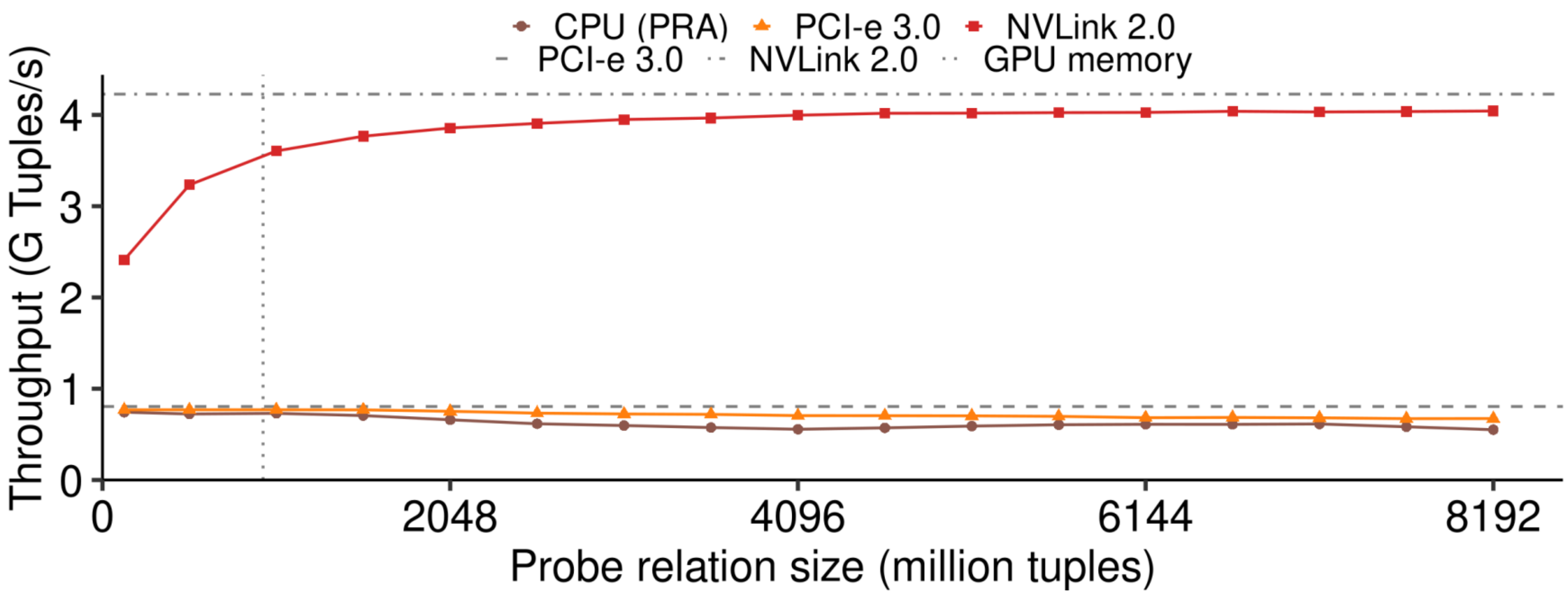
- Up to 2⋈122 GiB
- CPU baseline: Radix-partitioned hash join

Probe-Side Scaling



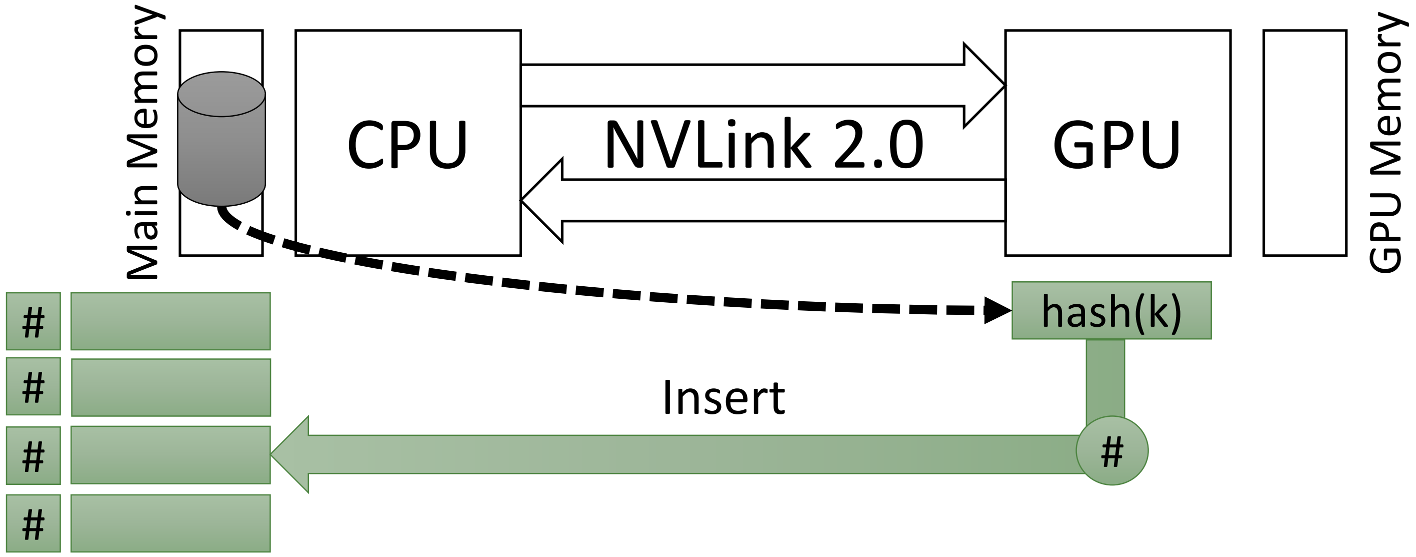
- Up to 2⋈122 GiB
- CPU baseline: Radix-partitioned hash join

Probe-Side Scaling

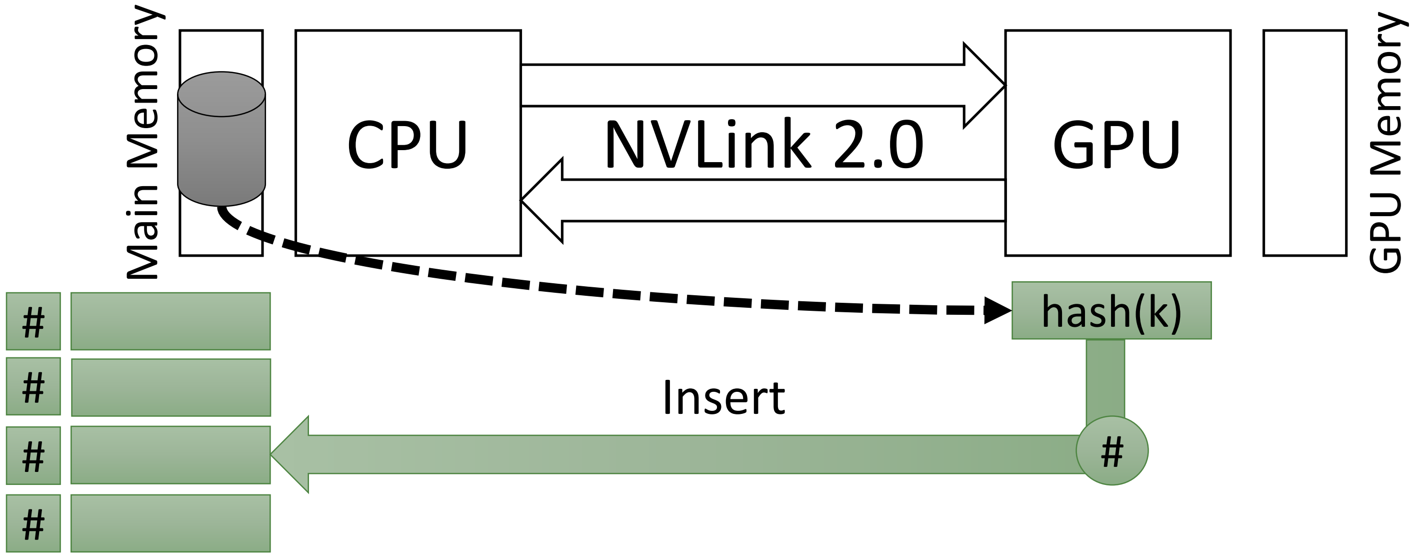


GPUs can efficiently process large, out-of-core data

Build-Side Scaling

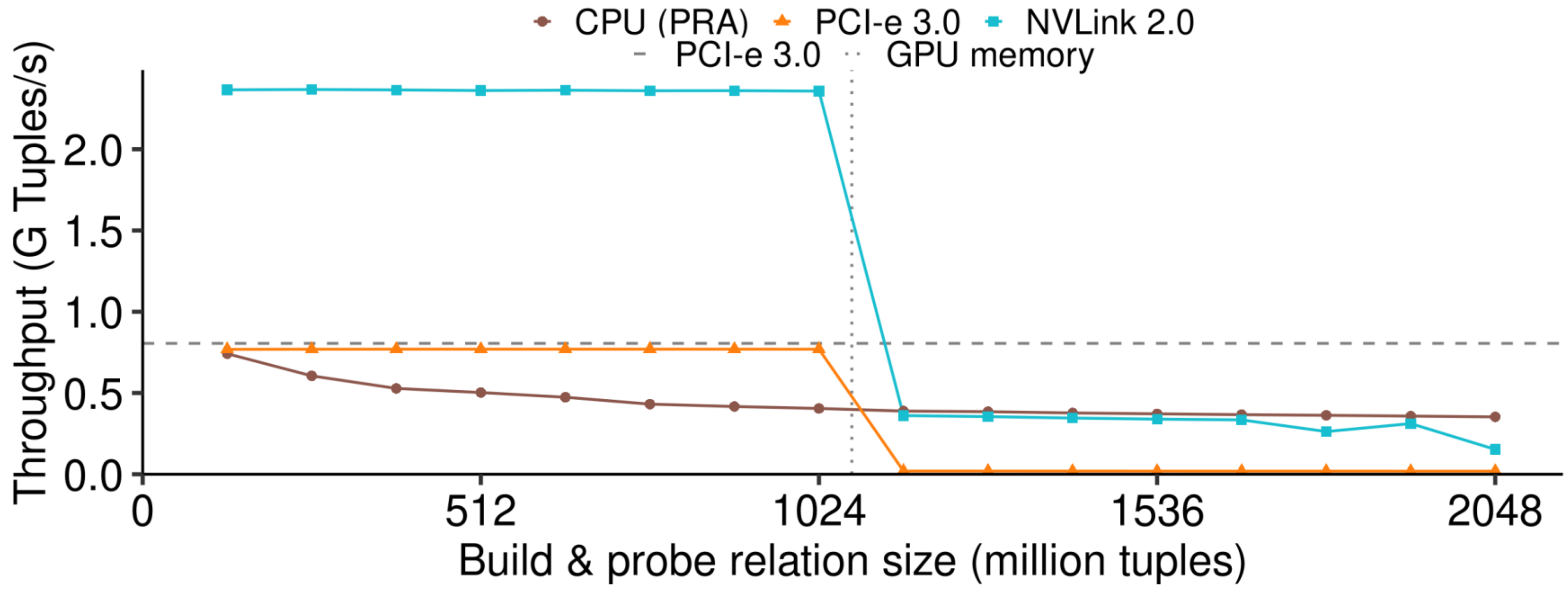


Build-Side Scaling



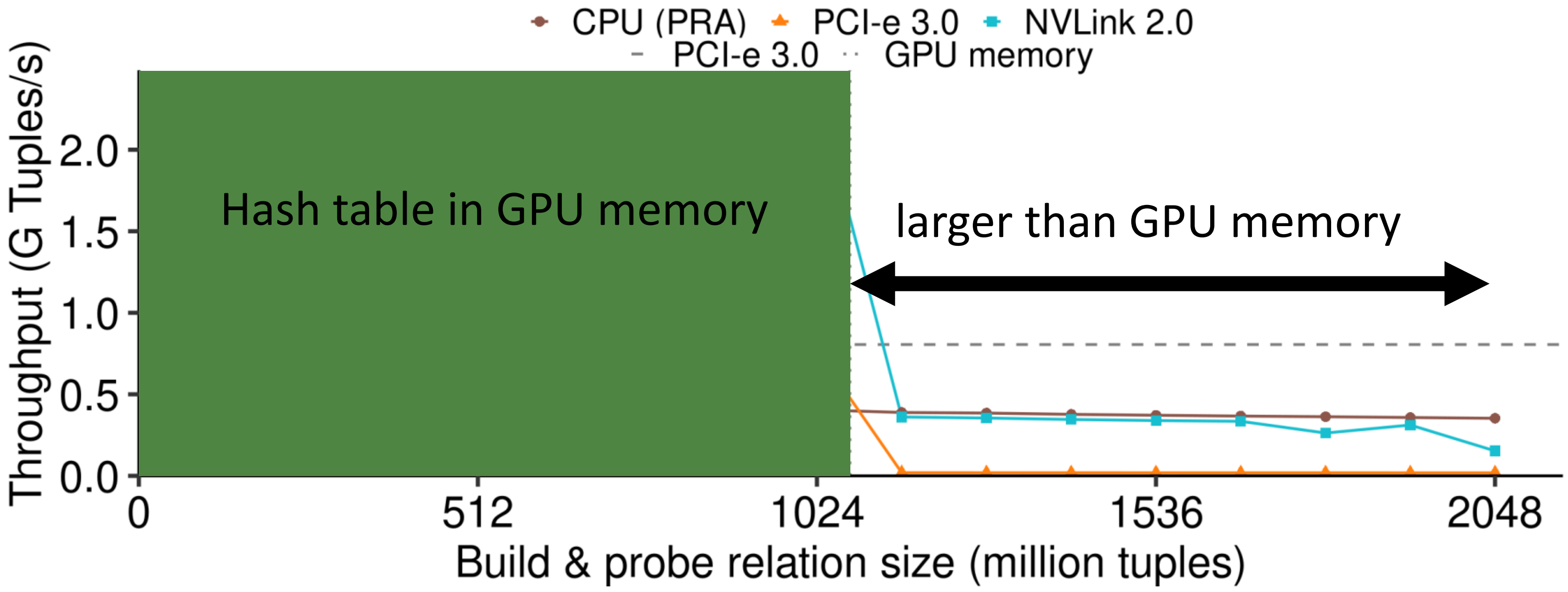
Interconnect feature: Data-dependent memory access

Build-Side Scaling



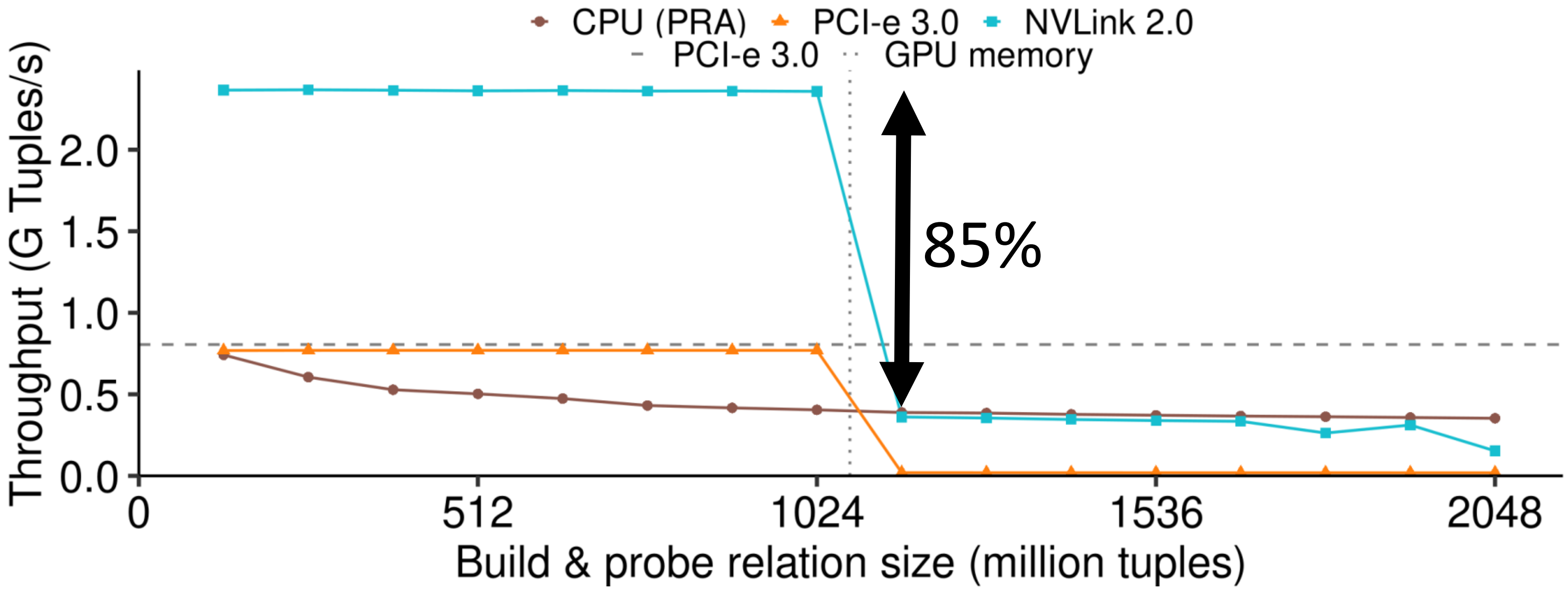
- Up to 30⋈30 GiB with a 30 GiB hash table **= 90 GiB**

Build-Side Scaling



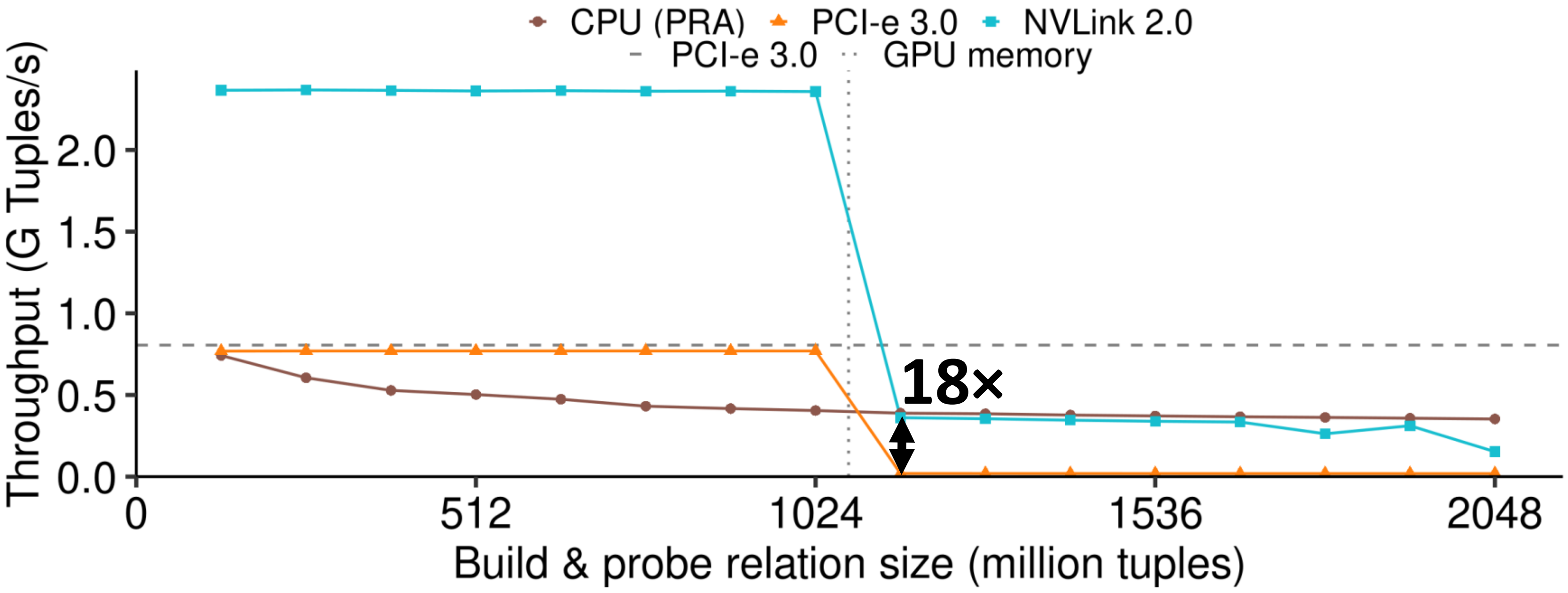
- Up to 30⋈30 GiB with a 30 GiB hash table = **90 GiB**

Build-Side Scaling



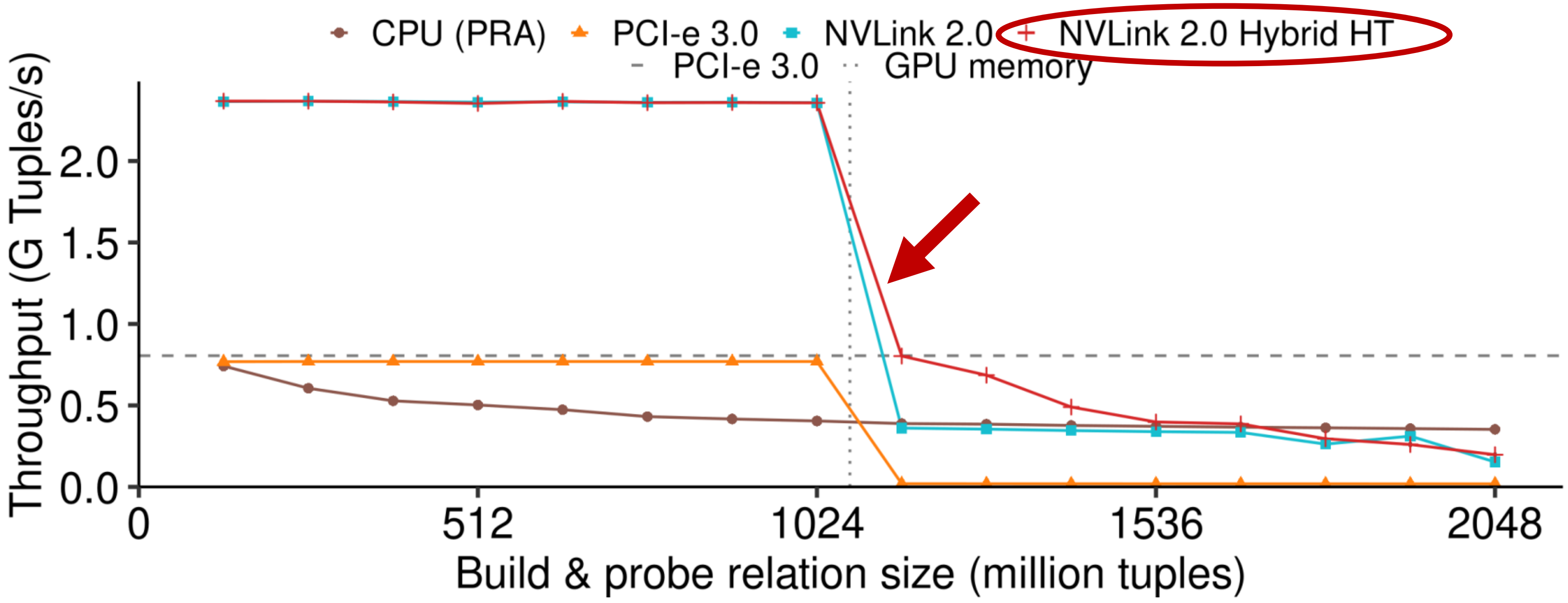
- Up to 30⋈30 GiB with a 30 GiB hash table = **90 GiB**

Build-Side Scaling



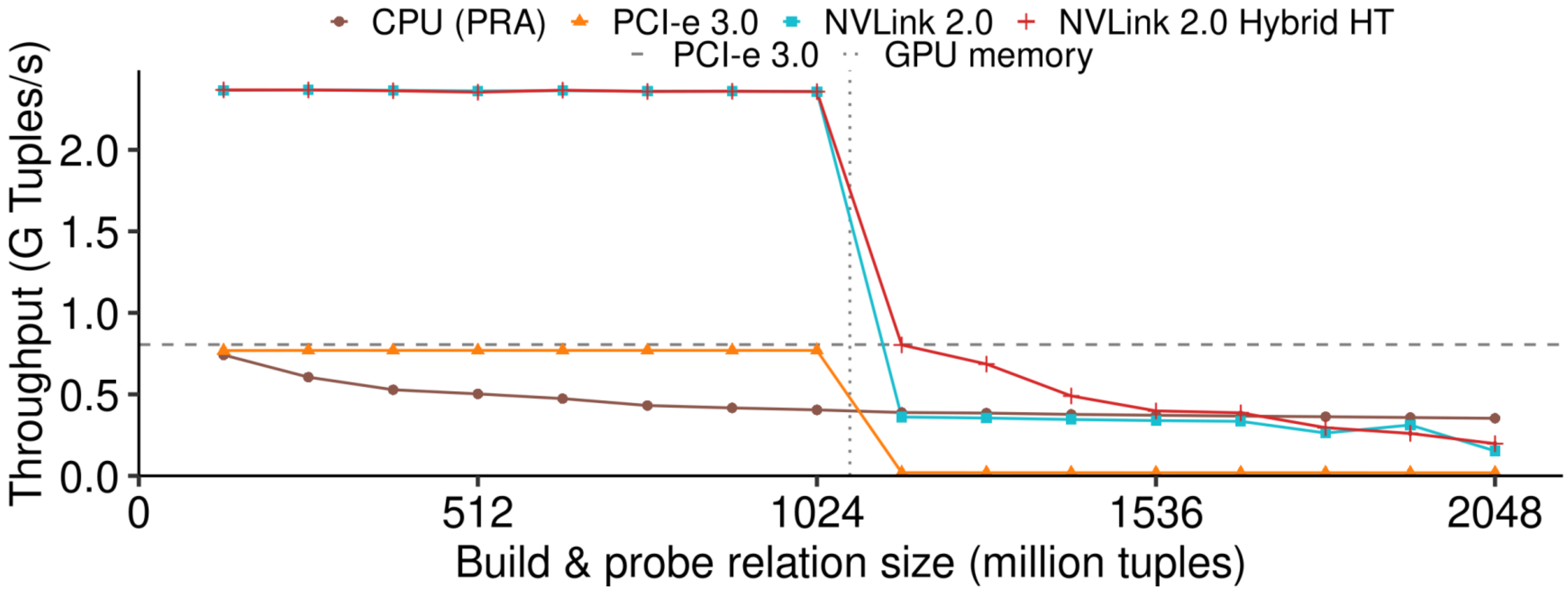
- Up to 30⋈30 GiB with a 30 GiB hash table **= 90 GiB**

Build-Side Scaling



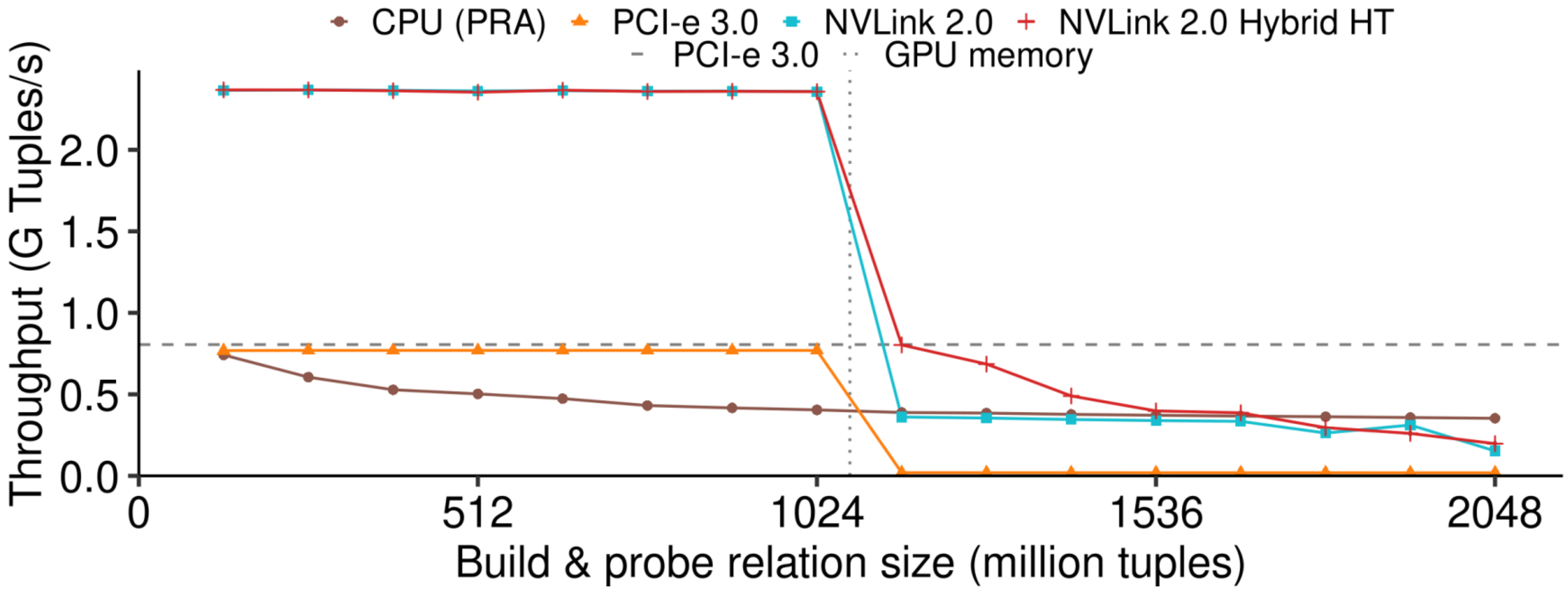
- Up to 30⋈30 GiB with a 30 GiB hash table **= 90 GiB**
- Hybrid hash table spills to CPU memory

Build-Side Scaling



GPUs are able to operate on large, out-of-core data structures

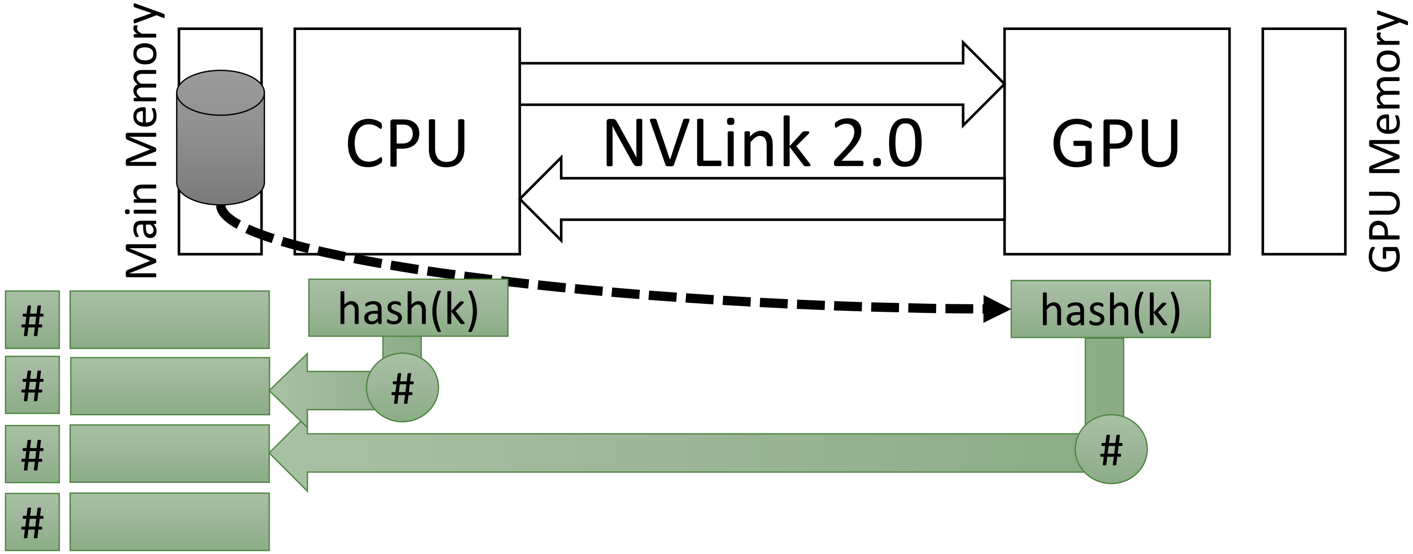
Build-Side Scaling



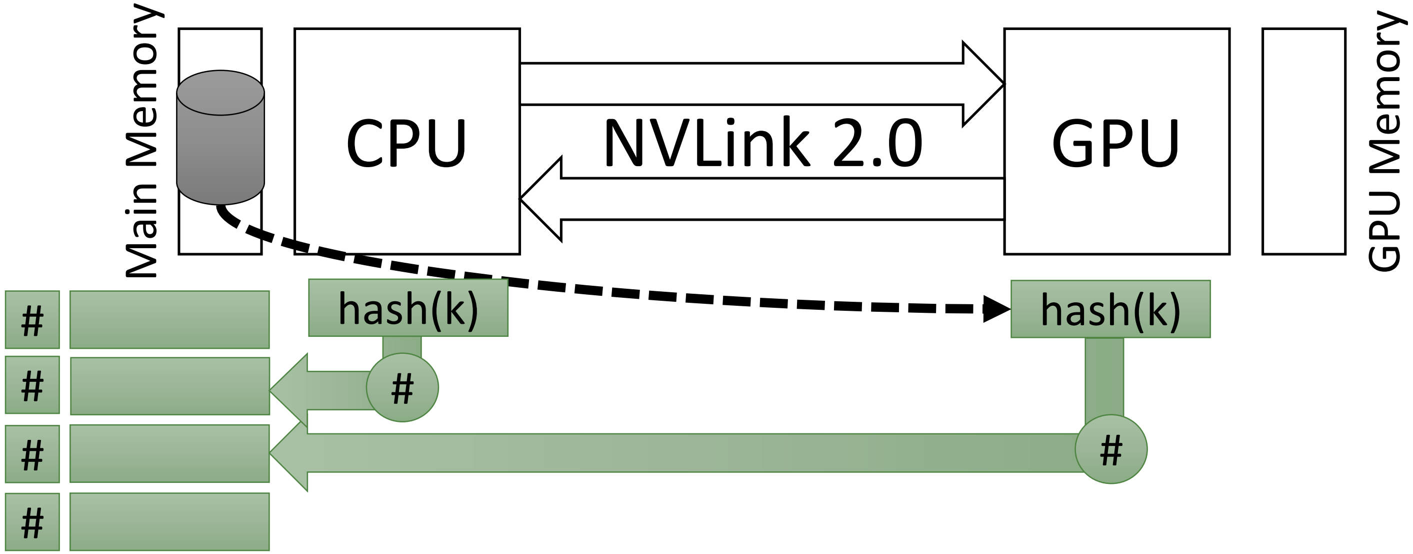
GPUs are able to operate on large, out-of-core data structures

… but should cache data structures in GPU memory

GPU+CPU Cooperation



GPU+CPU Cooperation

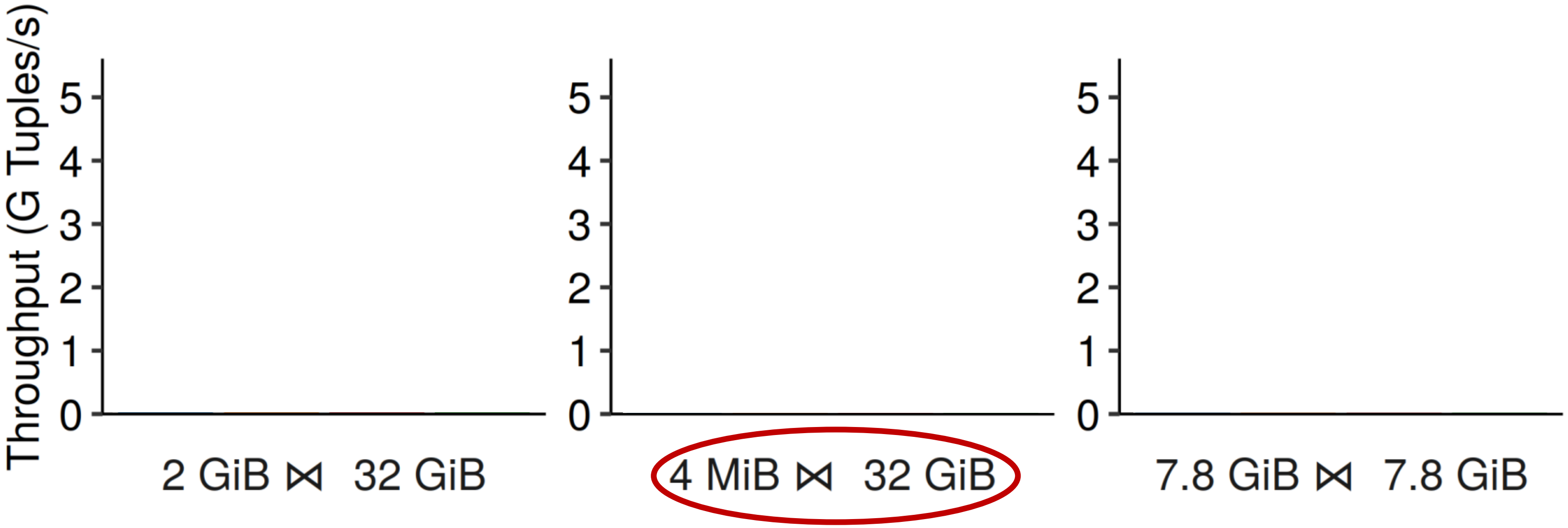


Interconnect feature: Cache coherence

GPU+CPU Cooperation

Throughput (G Tuples/s)
5
4
3
2
1
0
2 GiB ⋈ 32 GiB
4 MiB ⋈ 32 GiB
7.8 GiB ⋈ 7.8 GiB

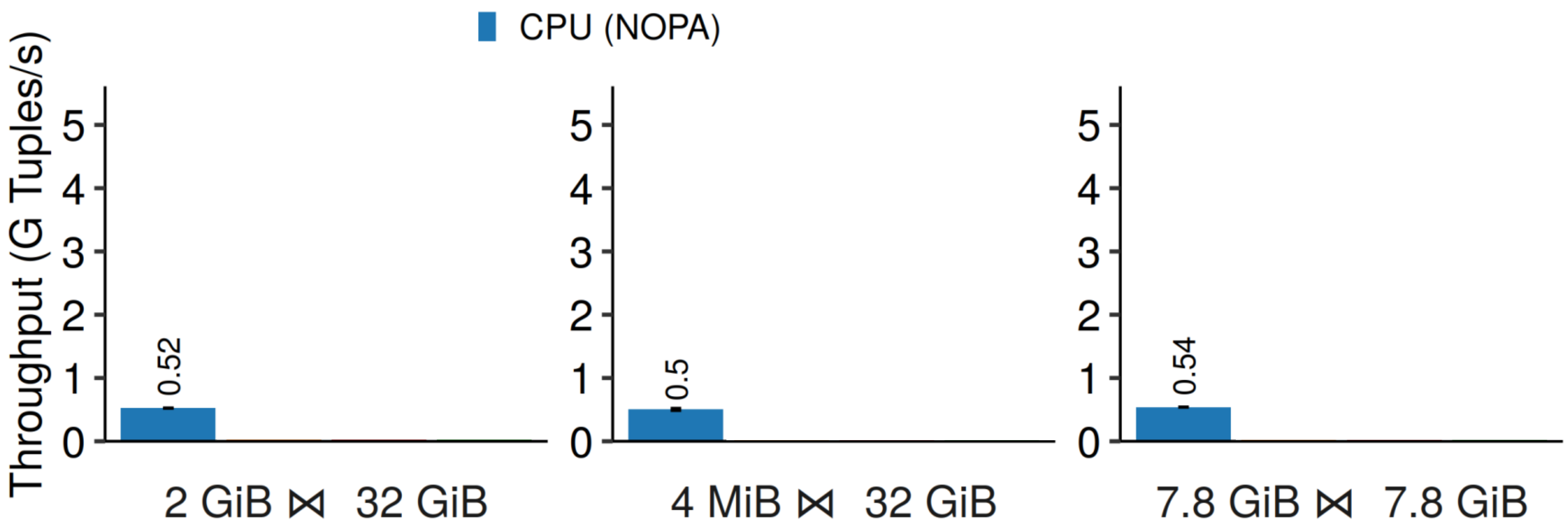
GPU+CPU Cooperation



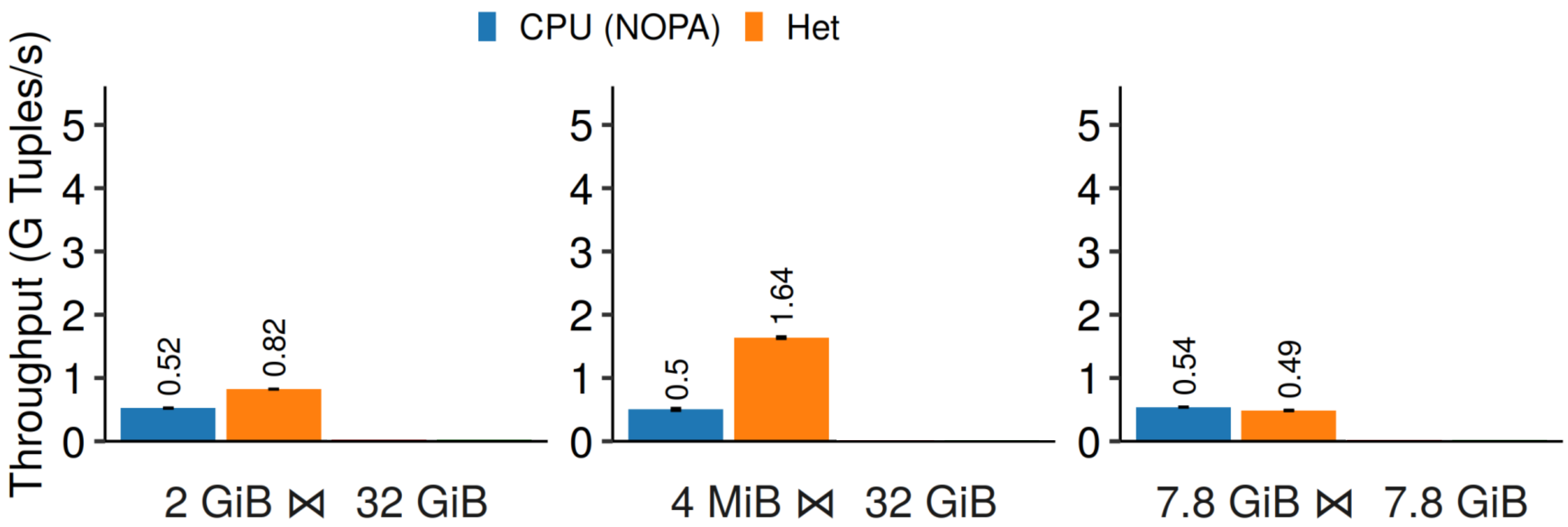
GPU+CPU Cooperation

Throughput (G Tuples/s)
0
1
2
3
4
5
2 GiB ⋈ 32 GiB
4 MiB ⋈ 32 GiB
7.8 GiB ⋈ 7.8 GiB

GPU+CPU Cooperation

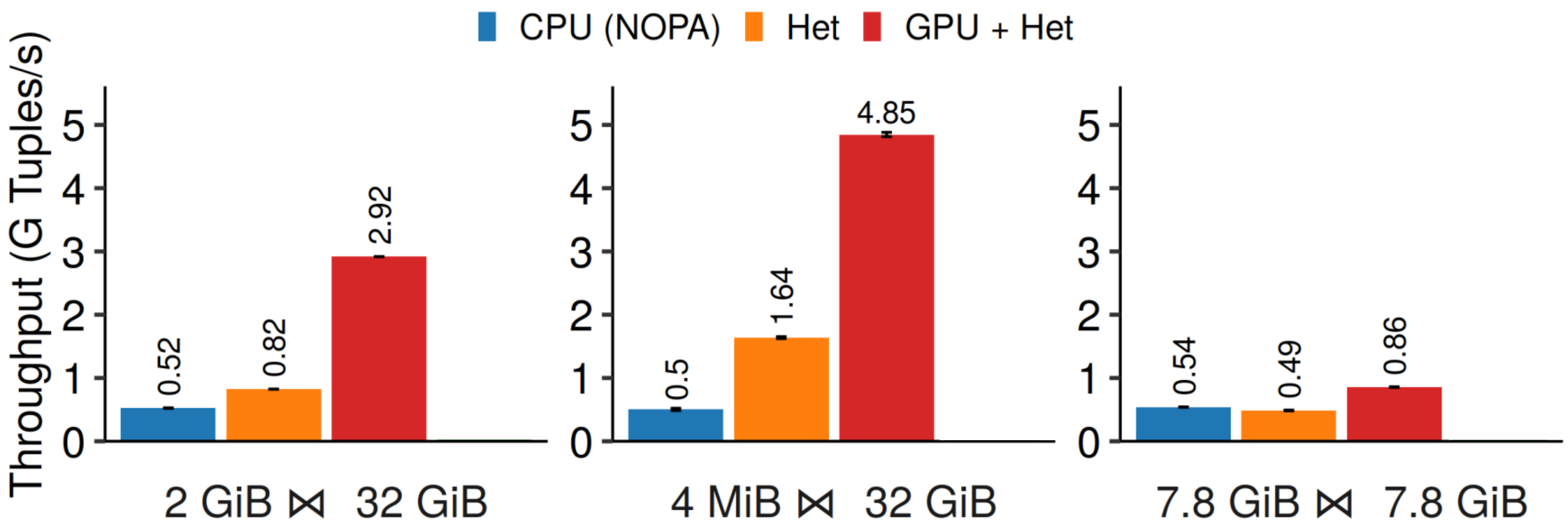


GPU+CPU Cooperation



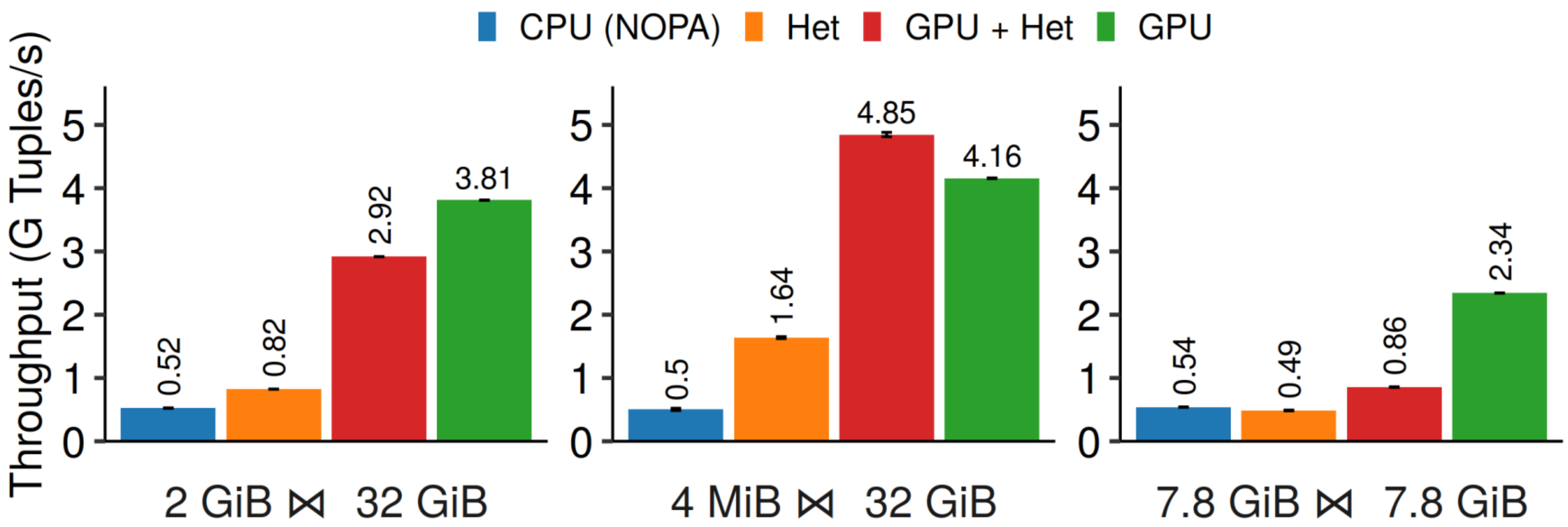
- Heterogeneous → GPU & CPU share hash table in CPU memory

GPU+CPU Cooperation



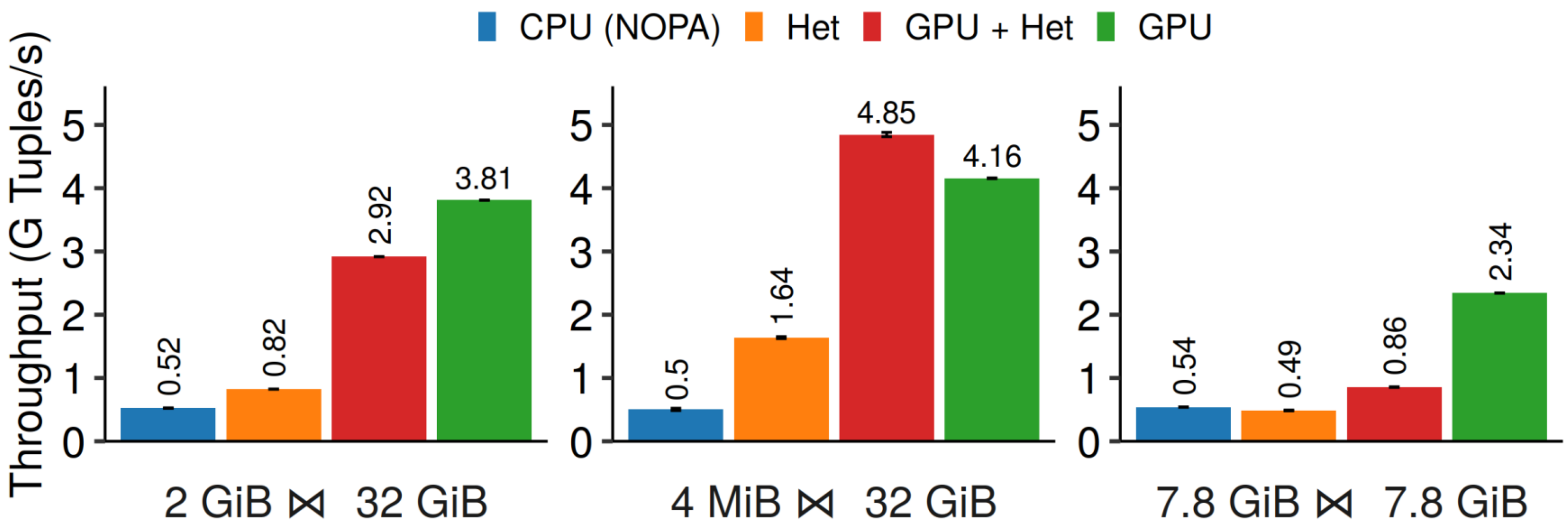
- Heterogeneous → GPU & CPU share hash table in CPU memory
- GPU + Het → GPU builds hash table in GPU memory, then copy it to CPU memory

GPU+CPU Cooperation



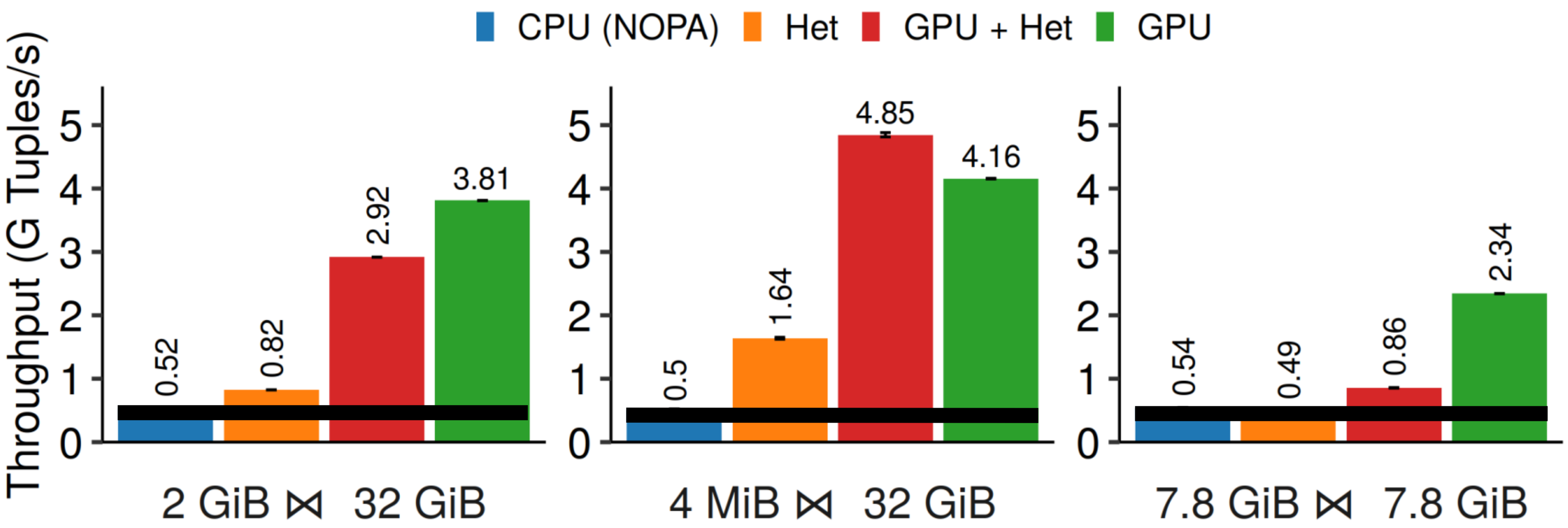
- Heterogeneous → GPU & CPU share hash table in CPU memory
- GPU + Het → GPU builds hash table in GPU memory, then copy it to CPU memory

GPU+CPU Cooperation



Scaling-up using co-processors makes performance more robust

GPU+CPU Cooperation



Scaling-up using co-processors makes performance more robust

Avoids worst-case, but using only GPU can be faster

Conclusion

*We explore **in which ways** fast interconnects **benefit databases**:*

- Out-of-core **data sets**
- Out-of-core **data structures**
- **Fine-grained** cooperative co-processing

www.clemenslutz.com

clemens.lutz@dfki.de