CS 764: Topics in Database Management Systems

Lecture 25: GPU Databases

Xiangyao Yu

11/30/2020

Announcements

Prepare for the project presentation next week

Submit the final project report to the paper review website by Dec. 18

Today's Papers: GPU Databases

A Study of the Fundamental Performance Characteristics of GPUs and CPUs for Database Analytics

Anil Shanbhag
MIT
anil@csail.mit.edu

Samuel Madden
MIT
madden@csail.mit.edu

Xiangyao Yu
University of Wisconsin-Madison
yxy@cs.wisc.edu

ABSTRACT

There has been significant amount of excitement and recent work on GPU-based database systems. Previous work has claimed that these systems can perform orders of magnitude better than CPU-based database systems on analytical workloads such as those found in decision support and business intelligence applications. A hardware expert would view these claims with suspicion. Given the general notion that database operators are memory-bandwidth bound, one would expect the maximum gain to be roughly equal to the ratio of the memory bandwidth of GPU to that of CPU. In this paper, we adopt a model-based approach to understand when and why the performance gains of running queries on GPUs vs on CPUs vary from the bandwidth ratio (which is roughly 16× on modern hardware). We propose Crystal, a library of parallel routines that can be combined together to

CCS CONCEPTS

• **Computer systems organization** → **Heterogeneous (hybrid) systems**; • **Information systems** → **Query operators**.

ACM Reference Format:
Anil Shanbhag, Samuel Madden, and Xiangyao Yu. 2020. A Study of the Fundamental Performance Characteristics of GPUs and CPUs for Database Analytics. In *2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3318464.3380595

1 INTRODUCTION

In the past decade, special-purpose graphics processing units (GPUs) have evolved into general purpose computing devices, with the advent of general purpose parallel programming models, such as CUDA [3] and OpenCL [7]. Because of

SIGMOD 2020

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

Clemens Lutz
clemens.lutz@dfki.de
DFKI GmbH
Berlin, Germany

Sebastian Breß
sebastian.bress@tu-berlin.de
TU Berlin
Berlin, Germany

Steffen Zeuch
steffen.zeuch@dfki.de
DFKI GmbH
Berlin, Germany

Tilmann Rabl
tilmann.rabl@hpi.de
HPI, University of Potsdam
Potsdam, Germany

Volker Markl
volker.markl@tu-berlin.de
DFKI GmbH, TU Berlin
Berlin, Germany

ABSTRACT

GPUs have long been discussed as accelerators for database query processing because of their high processing power and memory bandwidth. However, two main challenges limit the utility of GPUs for large-scale data processing: (1) the on-board memory capacity is too small to store large data sets, yet (2) the interconnect bandwidth to CPU main-memory is insufficient for ad hoc data transfers. As a result, GPU-based systems and algorithms run into a transfer bottleneck and do not scale to large data sets. In practice, CPUs process large-scale data faster than GPUs with current technology.

In this paper, we investigate how a fast interconnect can resolve these scalability limitations using the example of NVLink 2.0. NVLink 2.0 is a new interconnect technology

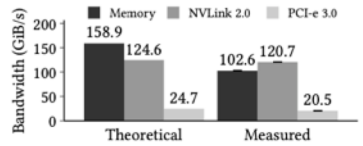


Figure 1: NVLink 2.0 eliminates the GPU's main-memory access disadvantage compared to the CPU.

ACM Reference Format:
Clemens Lutz, Sebastian Breß, Steffen Zeuch, Tilmann Rabl, and Volker Markl. 2020. Pump Up the Volume: Processing Large Data on GPUs with Fast Interconnects. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 17 pages.

SIGMOD 2020 (best paper award)

Outline

GPU architecture

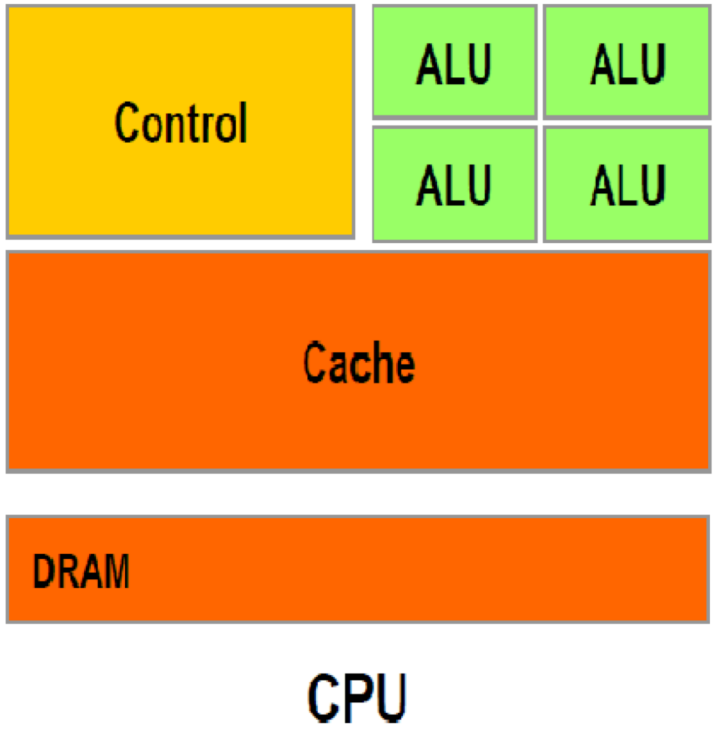
Challenges of GPU database

Paper 1: Crystal GPU database

Paper 2: NVLink GPU database

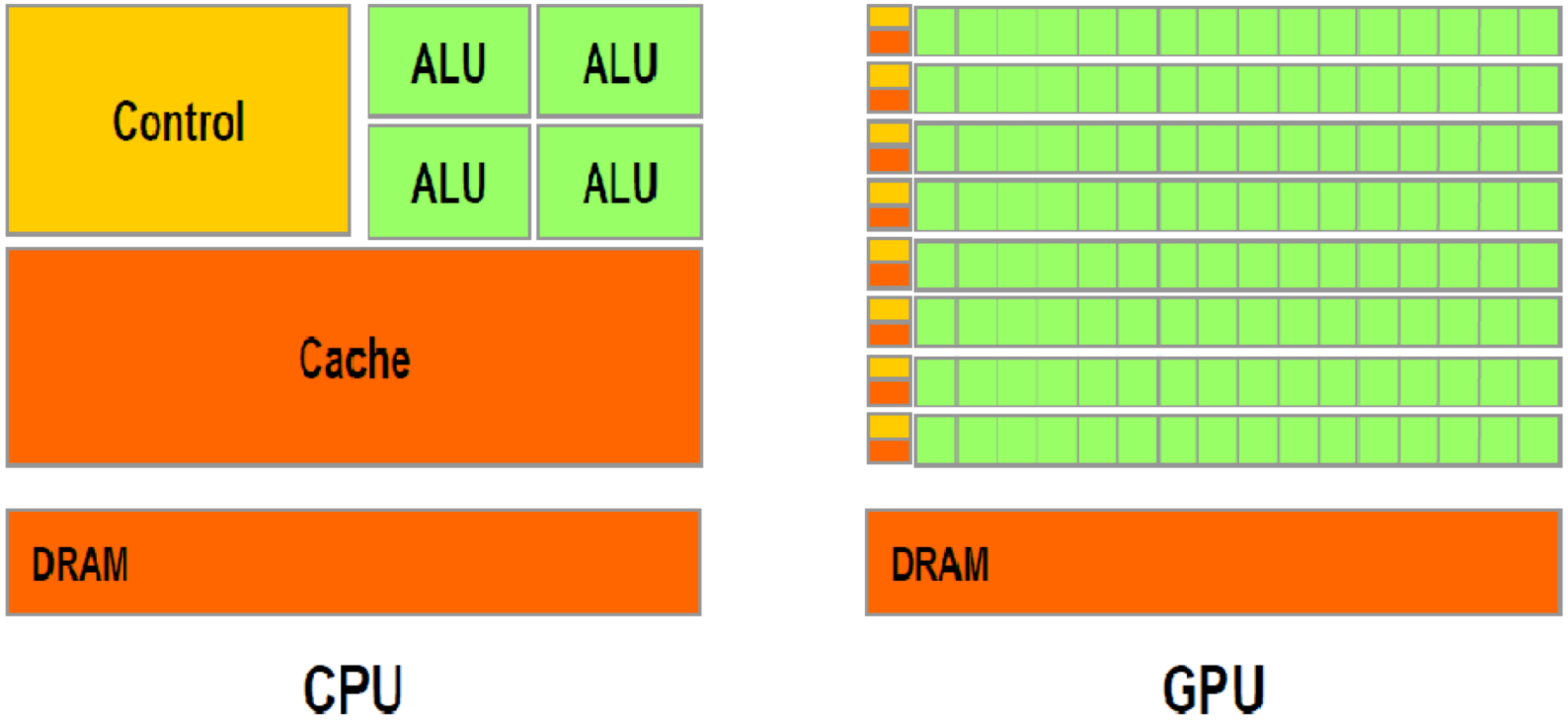
Summary

CPU vs. GPU



CPU: A few powerful cores with large caches. Optimized for sequential computation

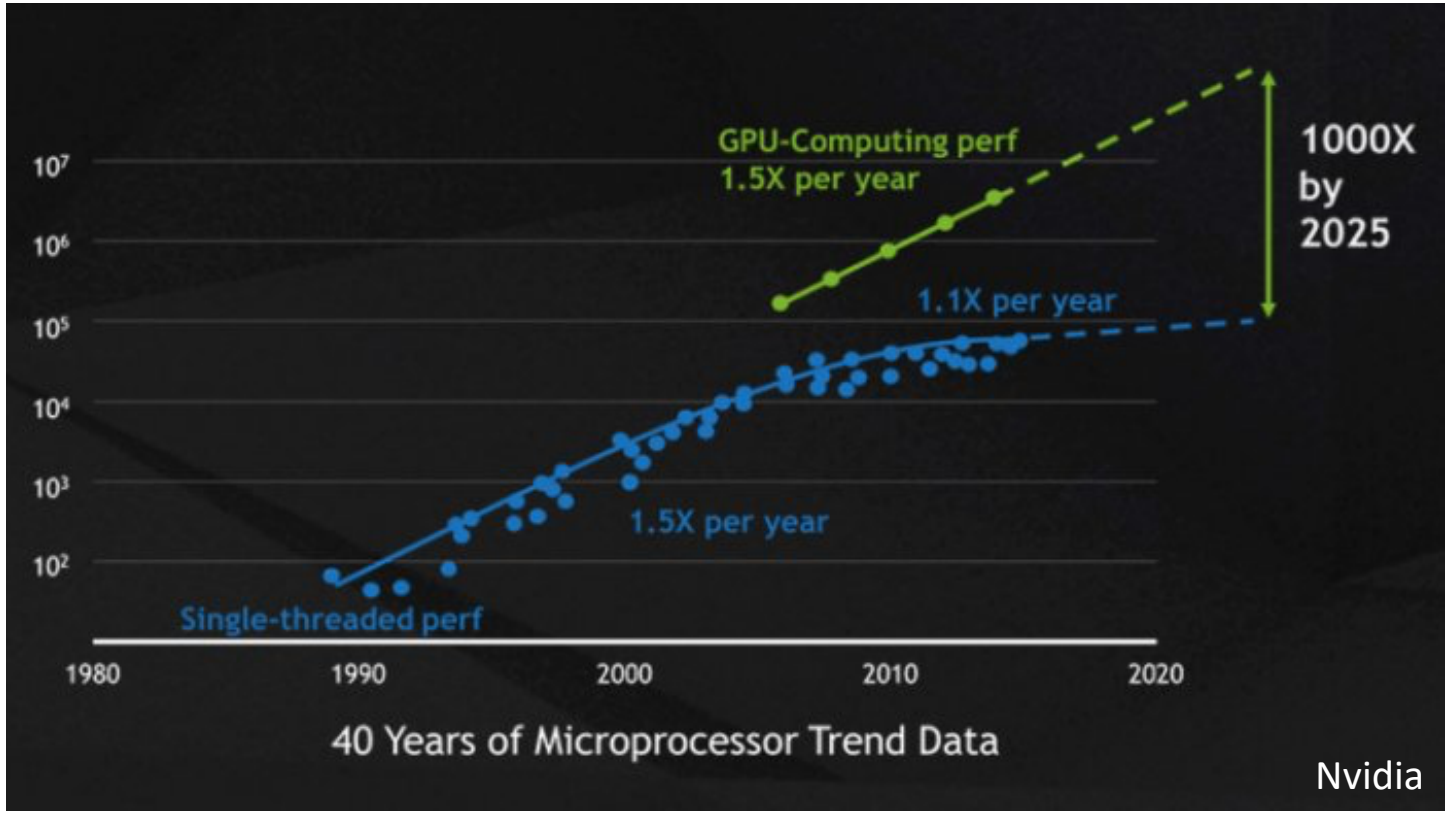
CPU vs. GPU



CPU: A few powerful cores with large caches. Optimized for sequential computation

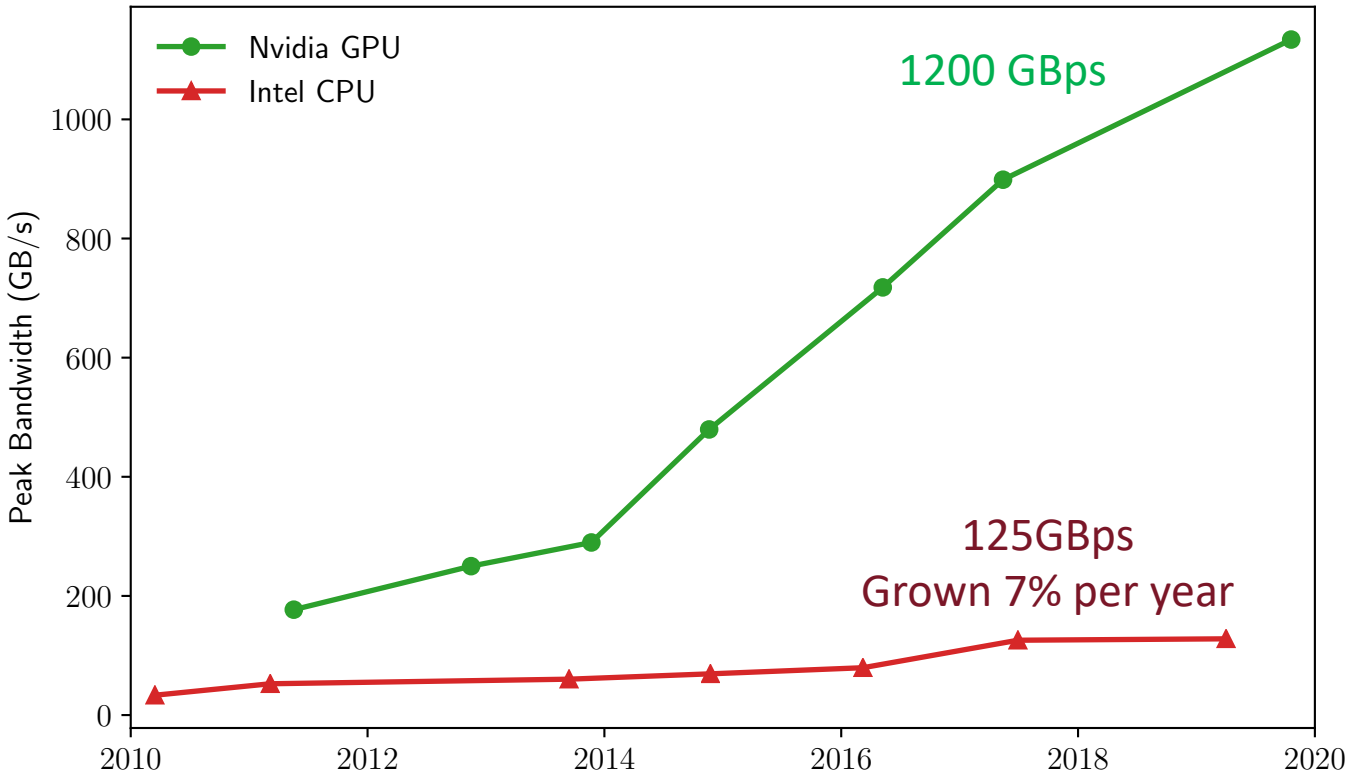
GPU: Many small cores. Optimized for parallel computation

CPU vs. GPU – Processing Units



	Throughput	Power	Throughput/Power
Intel Skylake	128 GFLOPS/4 Cores	100+ Watts	~1 GFLOPS/Watt
NVIDIA V100	15 TFLOPS	200+ Watts	~75 GFLOPS/Watt

CPU vs. GPU — Memory Bandwidth



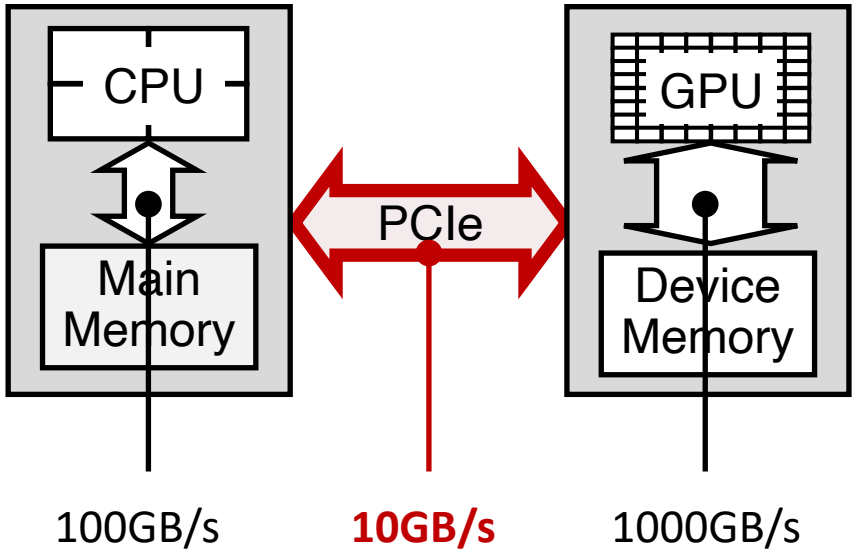
GPU has one order of magnitude higher memory bandwidth than CPU

Memory Bandwidth is the bottleneck for in-memory analytics

A natural idea: **use GPUs for data analytics**

GPU-DB Limitations

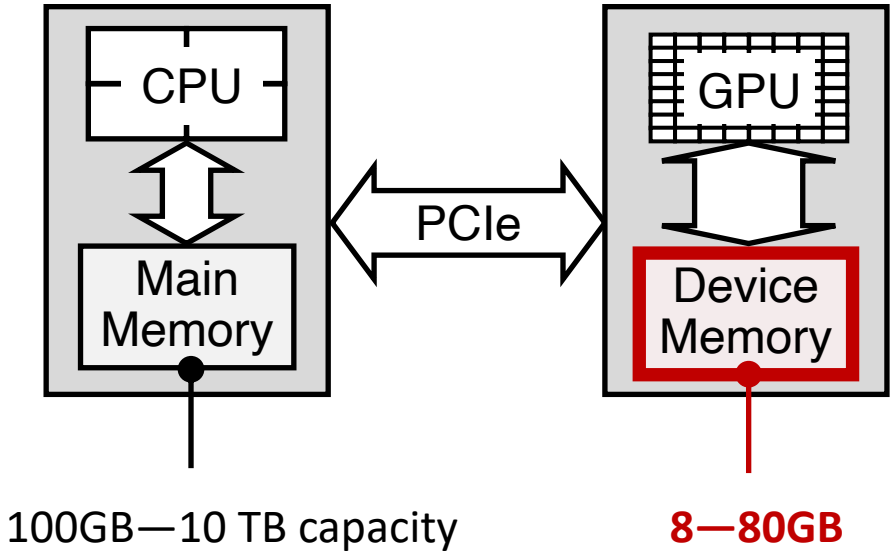
Limitation 1: Low interconnect bandwidth



GPU-DB Limitations

Limitation 1: Low interconnect bandwidth

Limitation 2: Small GPU memory capacity

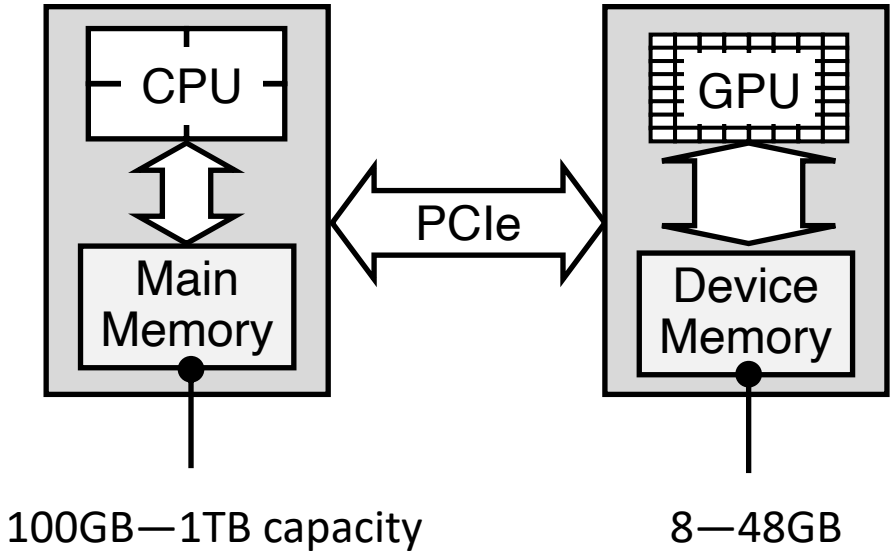


GPU-DB Limitations

Limitation 1: Low interconnect bandwidth

Limitation 2: Small GPU memory capacity

Limitation 3: Coarse-grained cooperation of CPU and GPU



Paper 1: Crystal GPU DB

A Study of the Fundamental Performance Characteristics of GPUs and CPUs for Database Analytics

Anil Shanbhag
MIT
anil@csail.mit.edu

Samuel Madden
MIT
madden@csail.mit.edu

Xiangyao Yu
University of Wisconsin-Madison
yxy@cs.wisc.edu

ABSTRACT

There has been significant amount of excitement and recent work on GPU-based database systems. Previous work has claimed that these systems can perform orders of magnitude better than CPU-based database systems on analytical workloads such as those found in decision support and business intelligence applications. A hardware expert would view these claims with suspicion. Given the general notion that database operators are memory-bandwidth bound, one would expect the maximum gain to be roughly equal to the ratio of the memory bandwidth of GPU to that of CPU. In this paper, we adopt a model-based approach to understand when and why the performance gains of running queries on GPUs vs on CPUs vary from the bandwidth ratio (which is roughly 16× on modern hardware). We propose Crystal, a library of parallel routines that can be combined together to

CCS CONCEPTS

• **Computer systems organization** → **Heterogeneous (hybrid) systems**; • **Information systems** → **Query operators**.

ACM Reference Format:
Anil Shanbhag, Samuel Madden, and Xiangyao Yu. 2020. A Study of the Fundamental Performance Characteristics of GPUs and CPUs for Database Analytics. In *2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3318464.3380595

1 INTRODUCTION

In the past decade, special-purpose graphics processing units (GPUs) have evolved into general purpose computing devices, with the advent of general purpose parallel programming models, such as CUDA [3] and OpenCL [7]. Because of

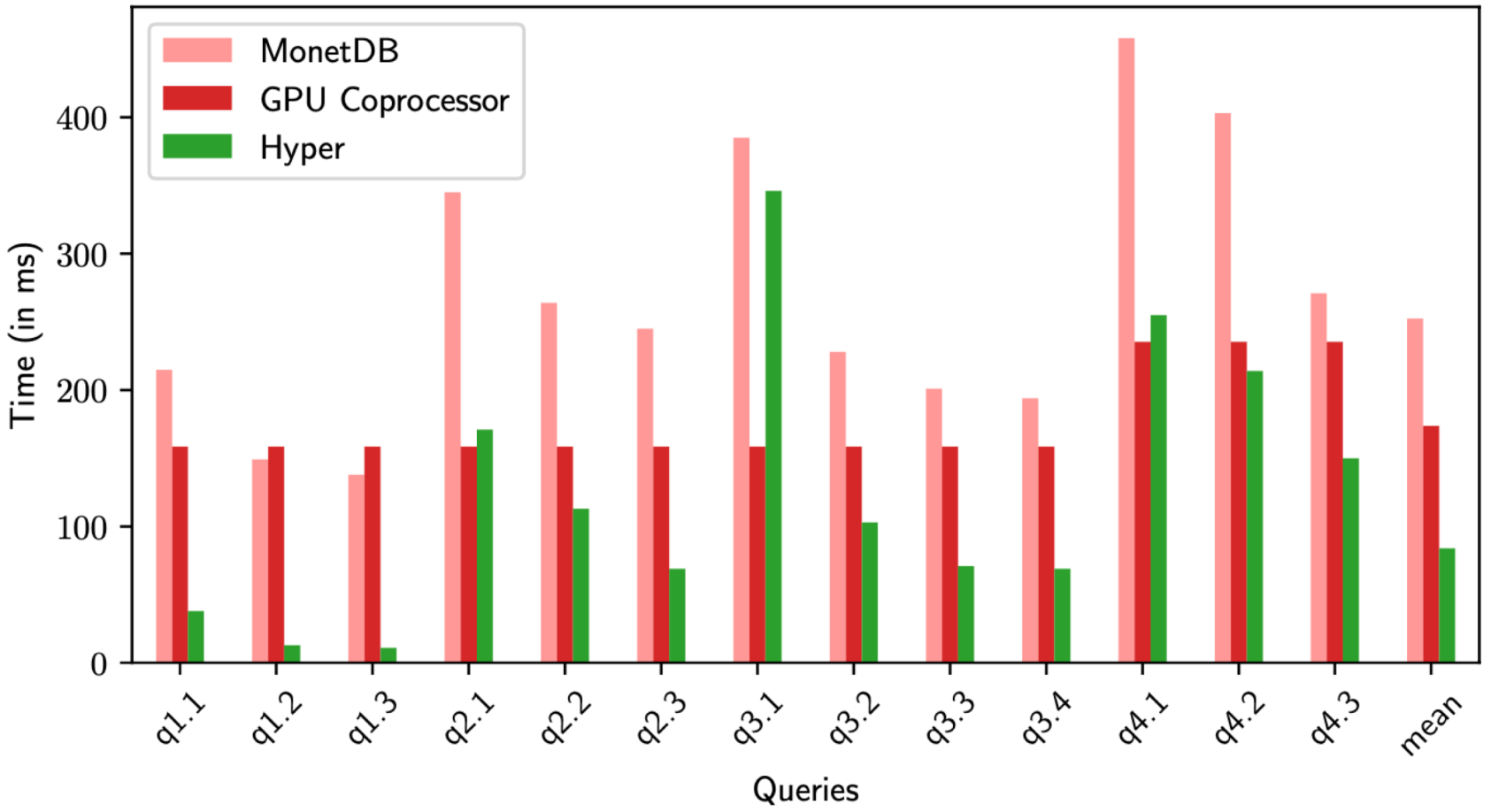
SIGMOD 2020

GPU Database Operation Mode

Coprocessor mode: Every query loads data from CPU memory to GPU

GPU-only mode: Store working set in GPU memory and run the entire query on GPU

CPU-only vs. Coprocessor



Key observation: With efficient implementations that can saturate memory bandwidth

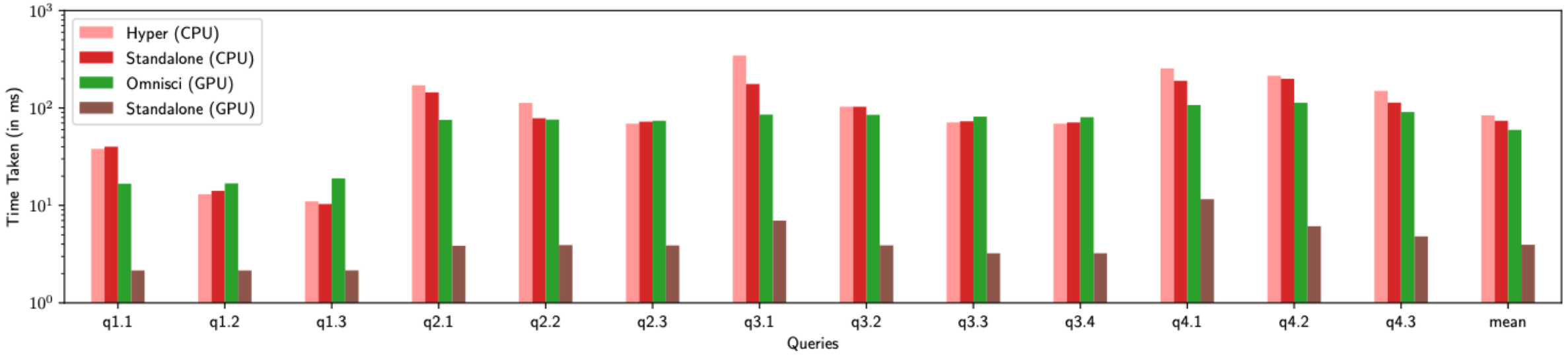
GPU-only > CPU-only > coprocessor

Star-Schema Benchmark

Platform	CPU	GPU
Model	Intel i7-6900	Nvidia V100
Cores	8 (16 with SMT)	5000
Memory Capacity	64 GB	32 GB
L1 Size	32KB/Core	16KB/SM
L2 Size	256KB/Core	6MB (Total)
L3 Size	20MB (Total)	-
Read Bandwidth	53GBps	880GBps
Write Bandwidth	55GBps	880GBps
L1 Bandwidth	-	10.7TBps
L2 Bandwidth	-	2.2TBps
L3 Bandwidth	157GBps	-

Crystal-based implementations always saturate GPU memory bandwidth

GPU is on average **25X** faster than CPU



Paper 2: GPU DB with NVLink

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

Clemens Lutz
clemens.lutz@dfki.de
DFKI GmbH
Berlin, Germany

Sebastian Breß
sebastian.bress@tu-berlin.de
TU Berlin
Berlin, Germany

Steffen Zeuch
steffen.zeuch@dfki.de
DFKI GmbH
Berlin, Germany

Tilmann Rabl
tilmann.rabl@hpi.de
HPI, University of Potsdam
Potsdam, Germany

Volker Markl
volker.markl@tu-berlin.de
DFKI GmbH, TU Berlin
Berlin, Germany

ABSTRACT

GPUs have long been discussed as accelerators for database query processing because of their high processing power and memory bandwidth. However, two main challenges limit the utility of GPUs for large-scale data processing: (1) the on-board memory capacity is too small to store large data sets, yet (2) the interconnect bandwidth to CPU main-memory is insufficient for ad hoc data transfers. As a result, GPU-based systems and algorithms run into a transfer bottleneck and do not scale to large data sets. In practice, CPUs process large-scale data faster than GPUs with current technology.

In this paper, we investigate how a fast interconnect can resolve these scalability limitations using the example of NVLink 2.0. NVLink 2.0 is a new interconnect technology

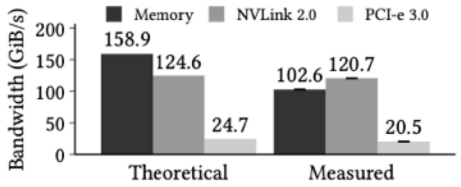
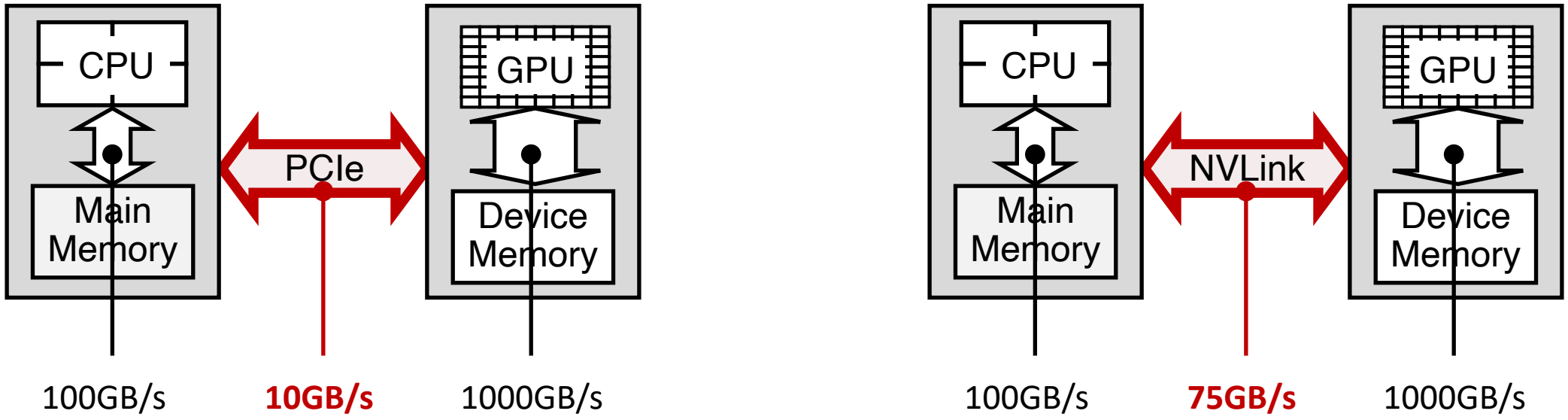


Figure 1: NVLink 2.0 eliminates the GPU's main-memory access disadvantage compared to the CPU.

ACM Reference Format:
Clemens Lutz, Sebastian Breß, Steffen Zeuch, Tilmann Rabl, and Volker Markl. 2020. Pump Up the Volume: Processing Large Data on GPUs with Fast Interconnects. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 17 pages.

SIGMOD 2020

Emerging Fast Interconnect



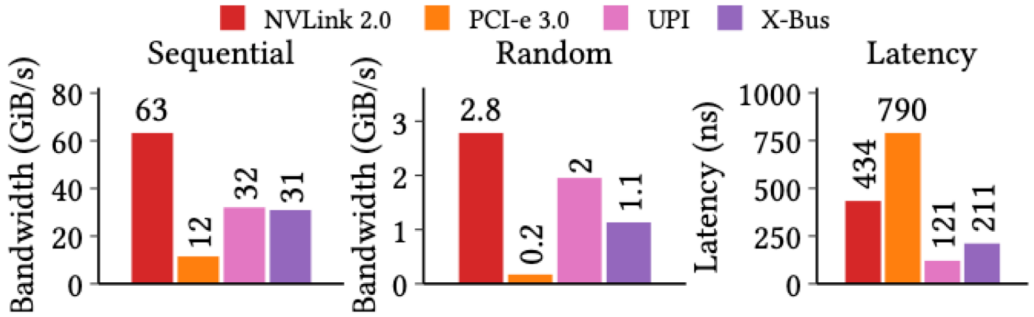
Fast Interconnect can solve the PCIe bottleneck

Emerging alternative interconnect technologies:

- NVLink
- Infinity Fabric
- Compute Express Link (CXL)

NVLink Bandwidth and Latency

NVLink has much higher bandwidth than PCIe

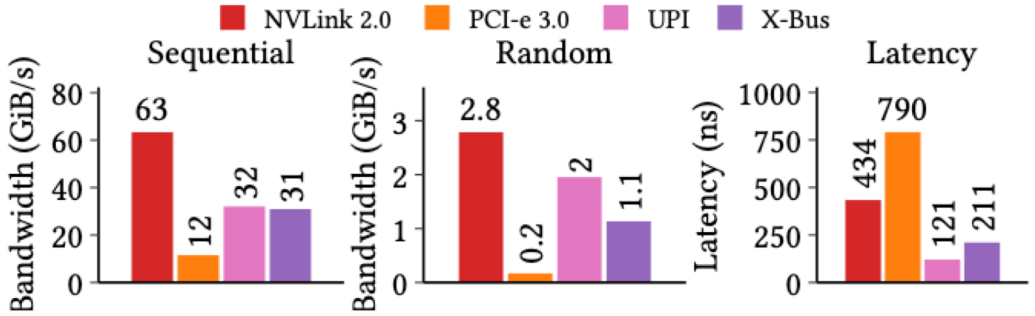


(a) NVLink 2.0 vs. CPU & GPU Interconnects.

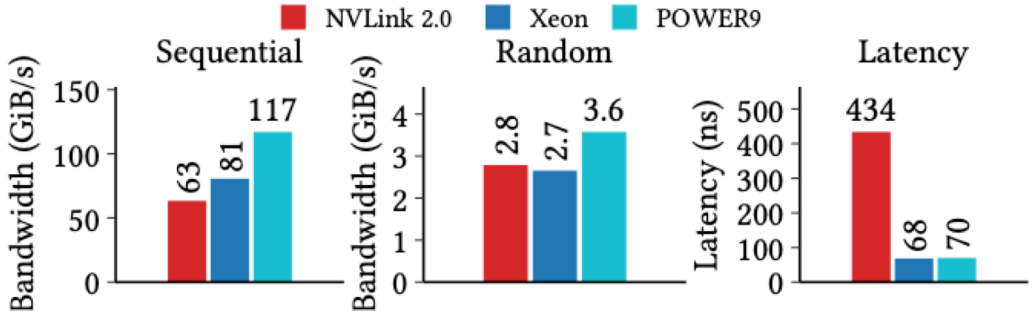
NVLink Bandwidth and Latency

NVLink has much higher bandwidth than PCIe

NVLink has comparable bandwidth as CPU local memory



(a) NVLink 2.0 vs. CPU & GPU Interconnects.



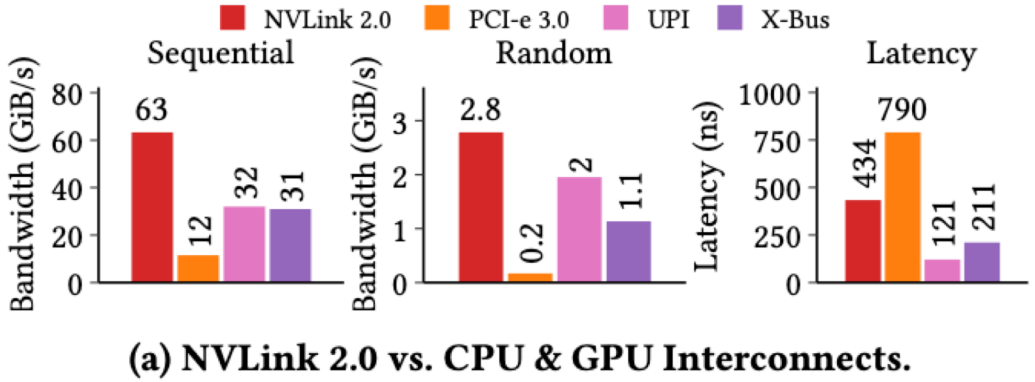
(b) NVLink 2.0 vs. CPU memory.

NVLink Bandwidth and Latency

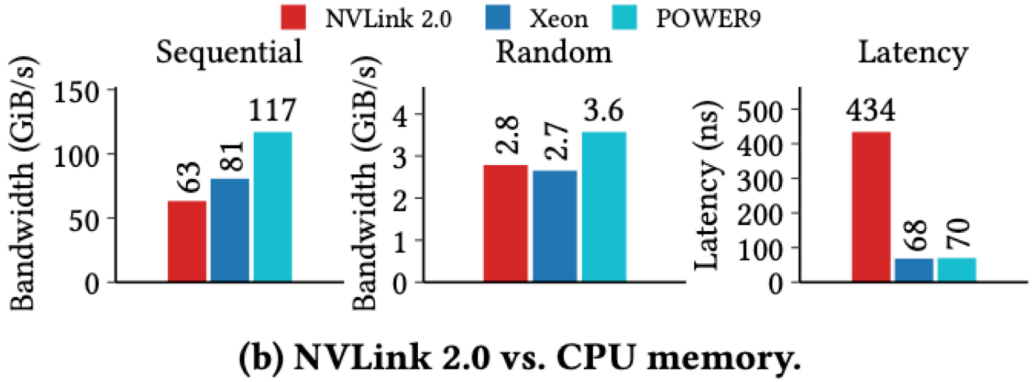
NVLink has much higher bandwidth than PCIe

NVLink has comparable bandwidth as CPU local memory

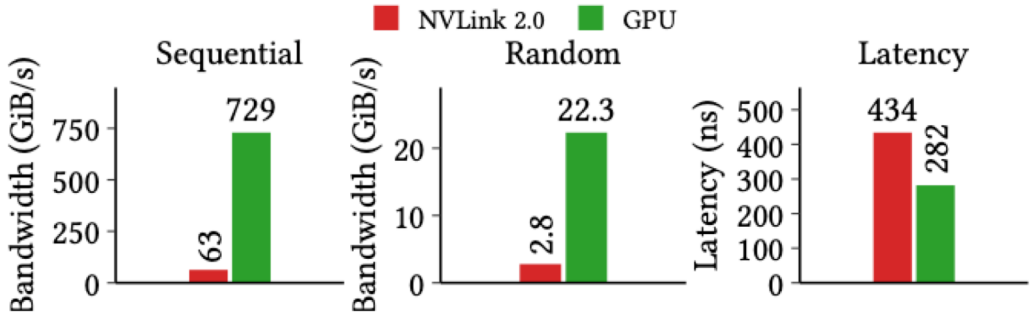
NVLink bandwidth has much lower bandwidth than GPU memory



(a) NVLink 2.0 vs. CPU & GPU Interconnects.



(b) NVLink 2.0 vs. CPU memory.



(c) NVLink 2.0 vs. GPU memory.

GPU Transfer Methods

Table 1: An overview of GPU transfer methods.

Method	Semantics	Level	Granularity	Memory
Pageable Copy Staged Copy Dynamic Pinning	Push	SW	Chunk	Pageable
Pinned Copy				Pinned
UM Prefetch				Unified
UM Migration	Pull	OS	Page	Unified
Zero-Copy		HW	Byte	Pinned
Coherence				Pageable

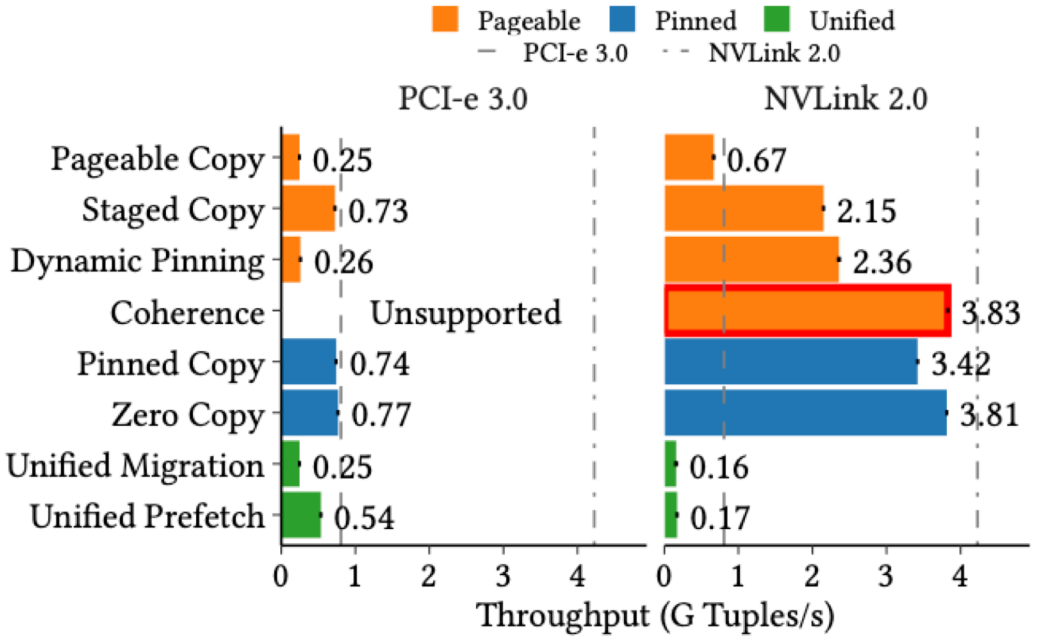


Figure 12: No-partitioning hash join using different transfer methods for PCI-e 3.0 and NVLink 2.0.

Pinned copy and **zero copy** can saturate PCIe bandwidth

Coherence can saturate NVLink bandwidth

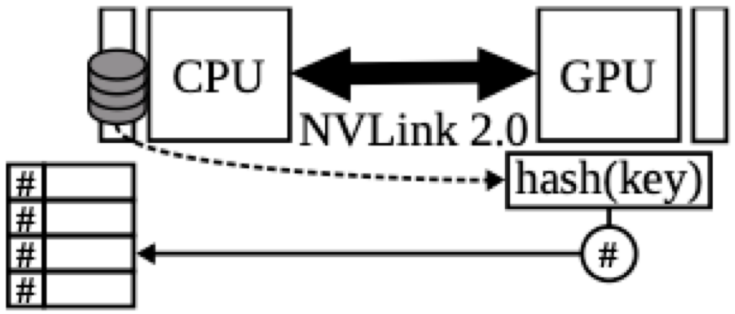
Non-Partitioned Hash Join Methods

Build phase: build the hash table using inner relation R

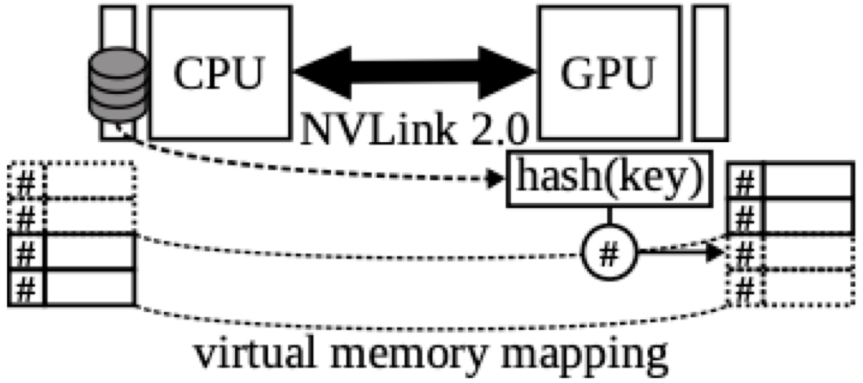
Probe phase: lookup hash table for each record in outer relation S

Hash Join – Build Phase

Build phase: build the hash table using inner relation R



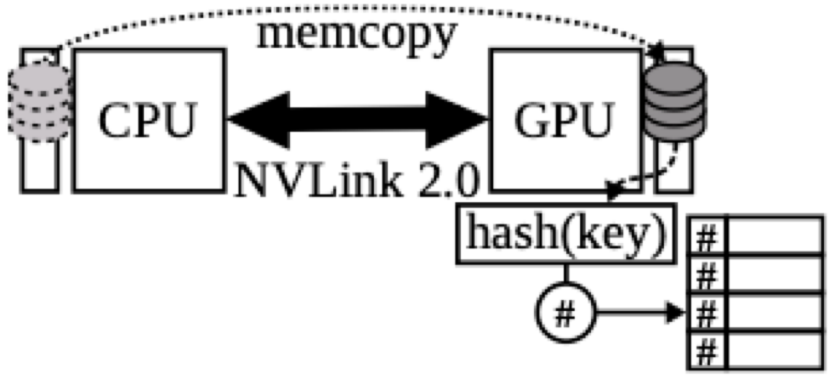
(a) Data and hash table in CPU memory.



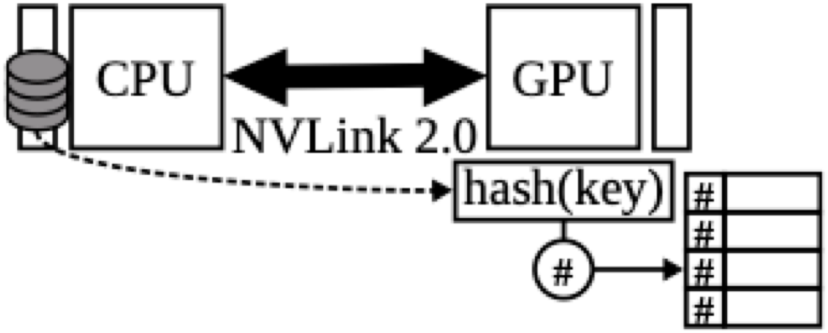
(b) Data in CPU memory and hash table spills from GPU memory into CPU memory.

Hash Join – Probe Phase

Probe phase: lookup hash table for each record in outer relation S

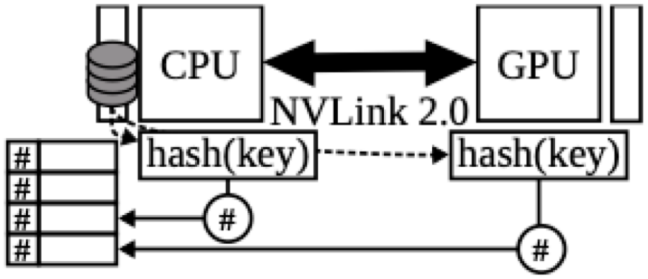


(a) Data and hash table in GPU memory.

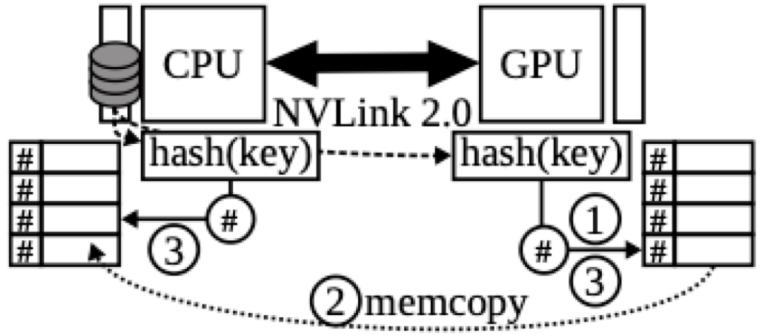


(b) Data in CPU memory and hash table in GPU memory.

Hash Join



(a) Cooperatively process join on CPU and GPU with hash table in CPU memory.



(b) Build hash table on GPU, copy the hash table to processor-local memories, and then cooperatively probe on CPU and GPU.

This **hybrid design** subsumes the previous designs in the paper

- Dynamically schedule tasks to both CPU and GPU

Hash Table Locality

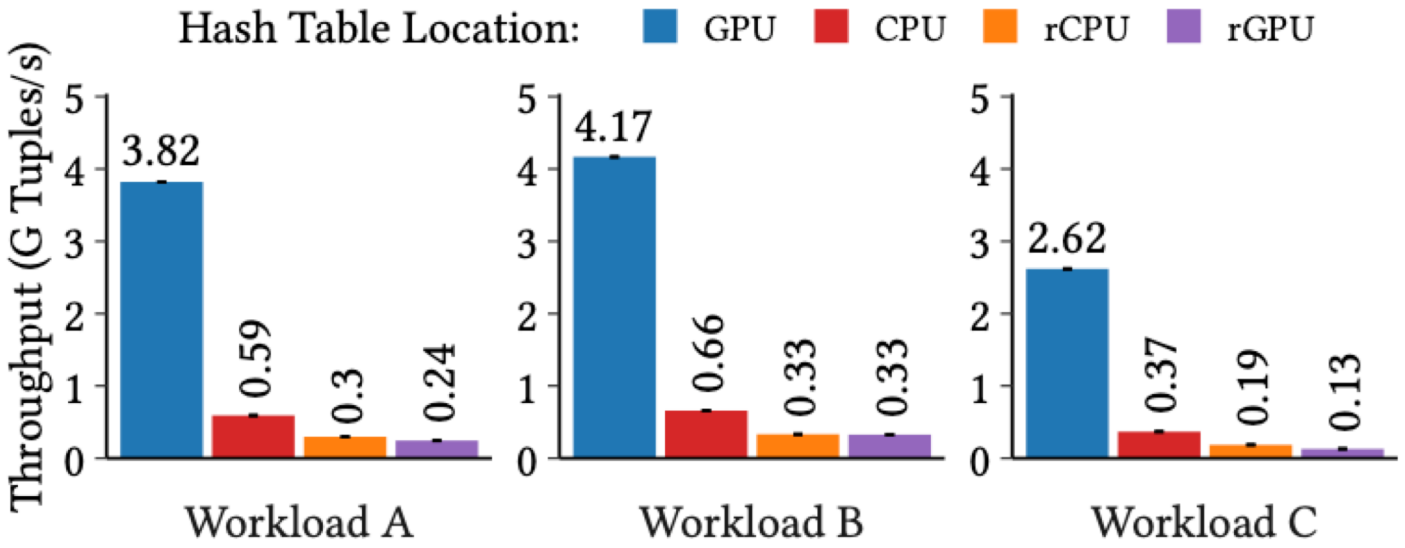


Figure 14: Join performance of the GPU when the hash table is located on different processors, increasing the number of interconnect hops from 0 to 3.

Best performance achieved when the hash table is in GPU memory

Scaling Data Size in TPC-H Q6

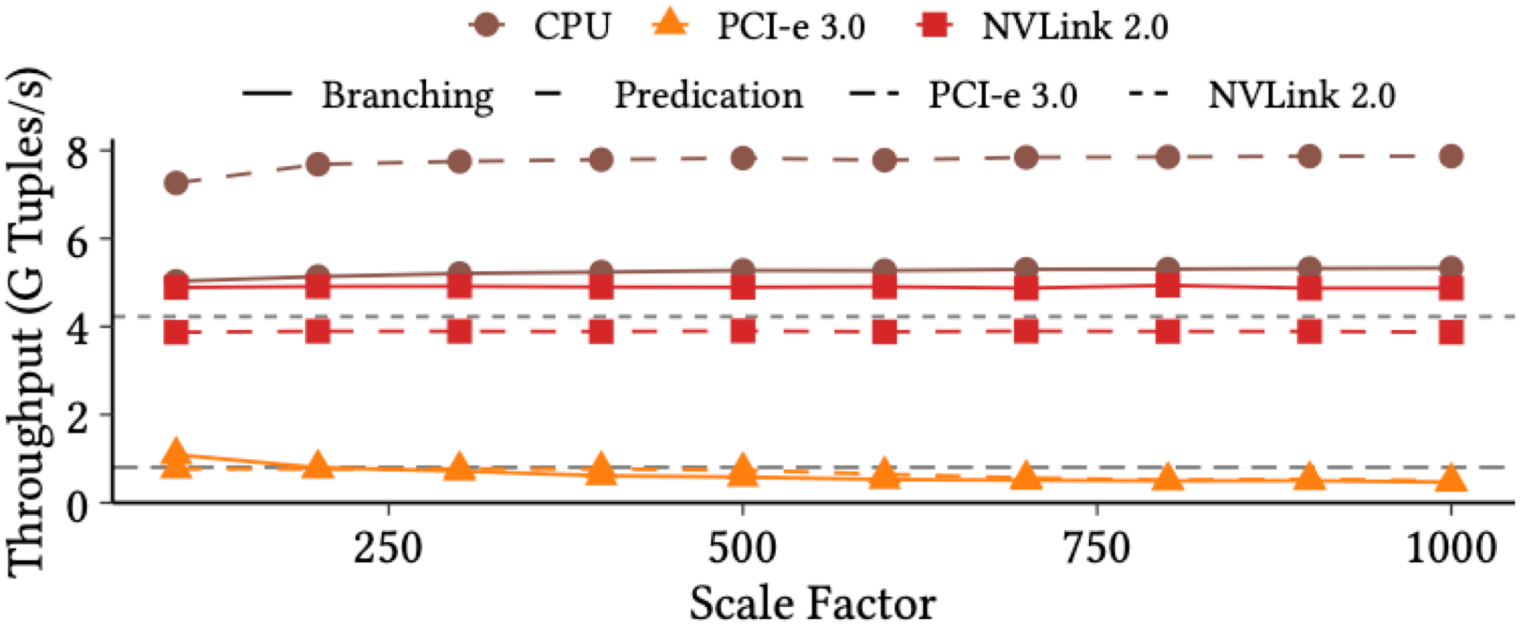


Figure 15: Scaling the data size of TPC-H query 6.

TPC-H Q6 contains a simple scan + aggregation with no join

Running the query on CPU leads to the highest performance

Scaling the Probe Side Relation

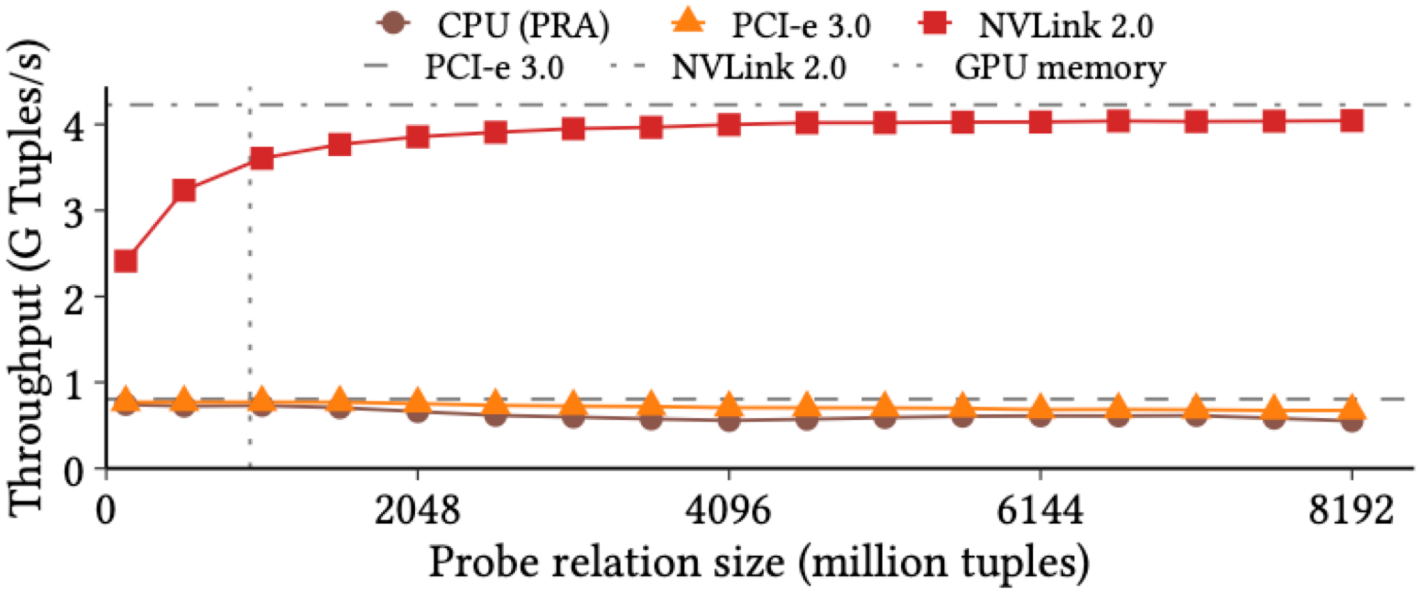


Figure 16: Scaling the probe-side relation.

NVLink is faster than both PCIe and CPU only

Scaling the Build Side Relation

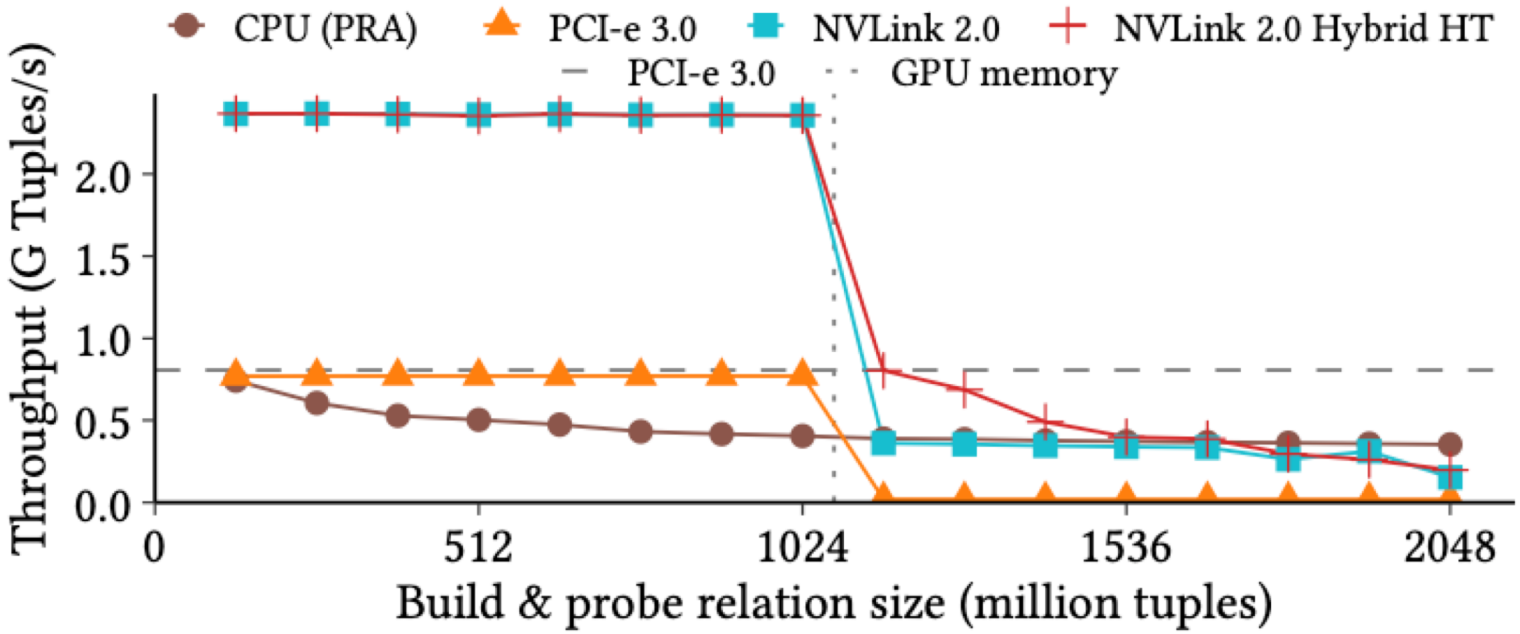


Figure 17: Scaling the build-side relation.

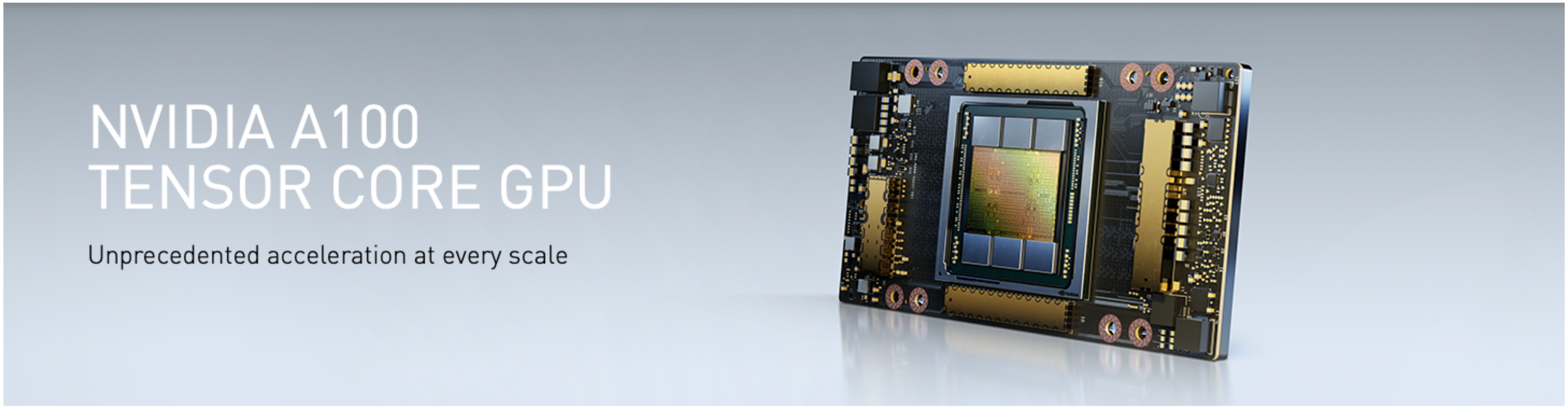
Performance drops when the hash table does not fit in GPU memory

Discussion

	Crystal	NVLink
Query Type	SPJA analytical queries	Non-partitioned hash join
Execution Model	Data fits in GPU memory	Coprocessor
Interconnect	PCIe 3.0	NVLink 2.0

Research question: How to maximize GPU database performance with different interconnect technology?

Latest A100 GPU



Available in 40GB and **80GB** memory versions, A100 80GB debuts the world's fastest memory bandwidth at over **2 terabytes per second (TB/s)** to run the largest models and datasets.

November 16, 2020

Q/A – GPU Databases

Some parameters (like batch size for task scheduling) complicates the configuration of such a database?

Techniques proposed in this paper work for operations besides joins?

What characteristics make the GPU's join processing rate faster than CPU's?

The hash table placement decision appears to be completely discrete

General optimization for DB with heterogeneous hardware?

Optimizing DB with TPU?

Next Lecture

Submit review for

- Jiacheng Yang, et al. F1 Lightning: HTAP as a Service, VLDB 2020