How Close Are We to GPU Data Processing?

Toward Disaggregated Data Systems in the AI Hardware Era

Jigao Luo

2026

Systems Group, TU Darmstadt

TECHNISCHE UNIVERSITÄT DARMSTADT

SYSTEMS

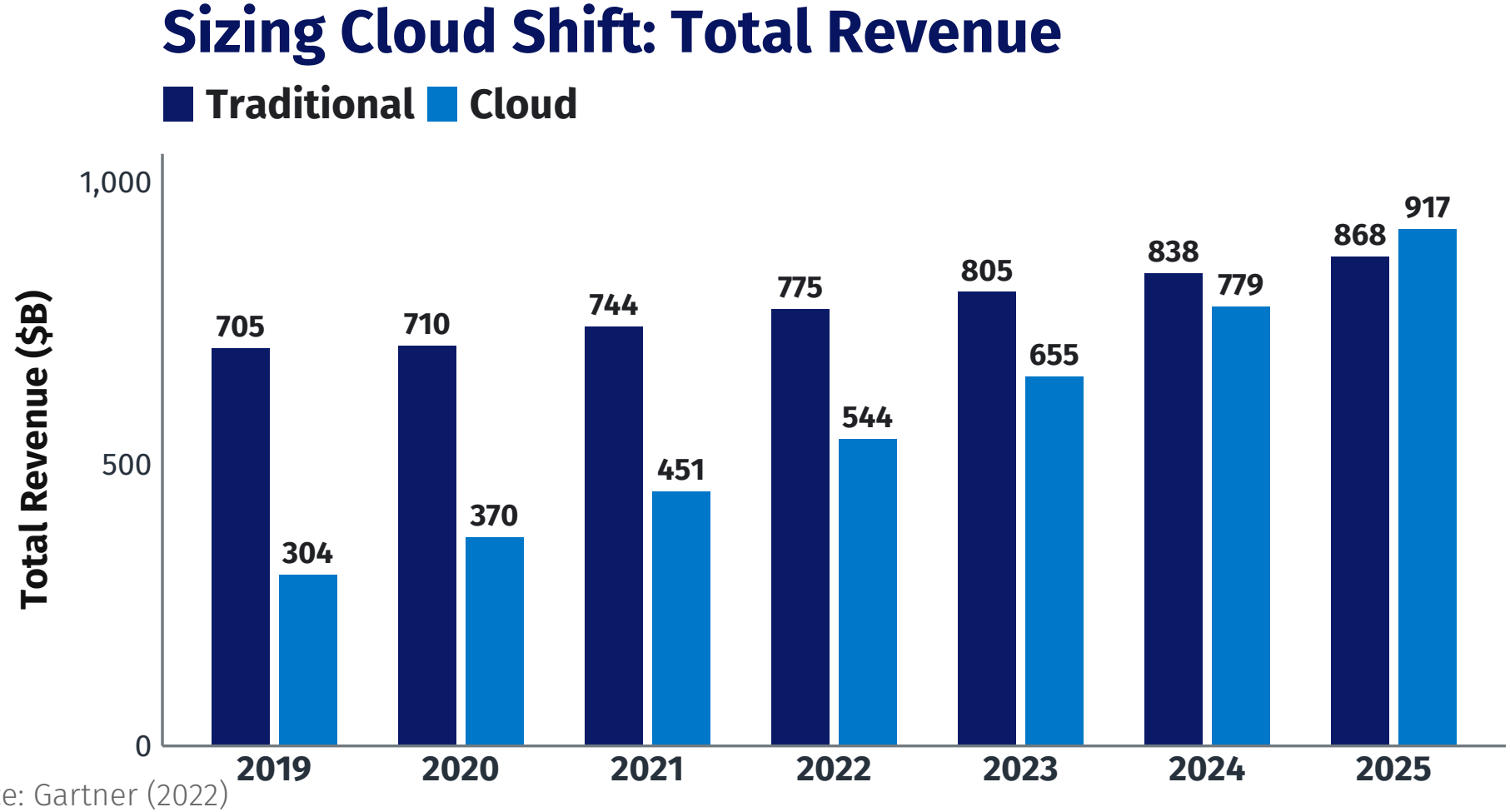
About Me

- Jigao Luo, third-year PhD student at TU Darmstadt
- Advisor: Prof. Carsten Binnig
- Research: disaggregated GPU data systems
- NVIDIA cuDF external contributor: Parquet reader
- Research Intern at Microsoft and Zilliz

Talk Structure

- Background: GPU Data Processing
- My Research
  - Overview
  - GPU & Parquet: DaMoN 26
  - PystachIO: VLDB 26
- The Road Ahead

Cloud Is a Success Story



Source: Gartner (2022)

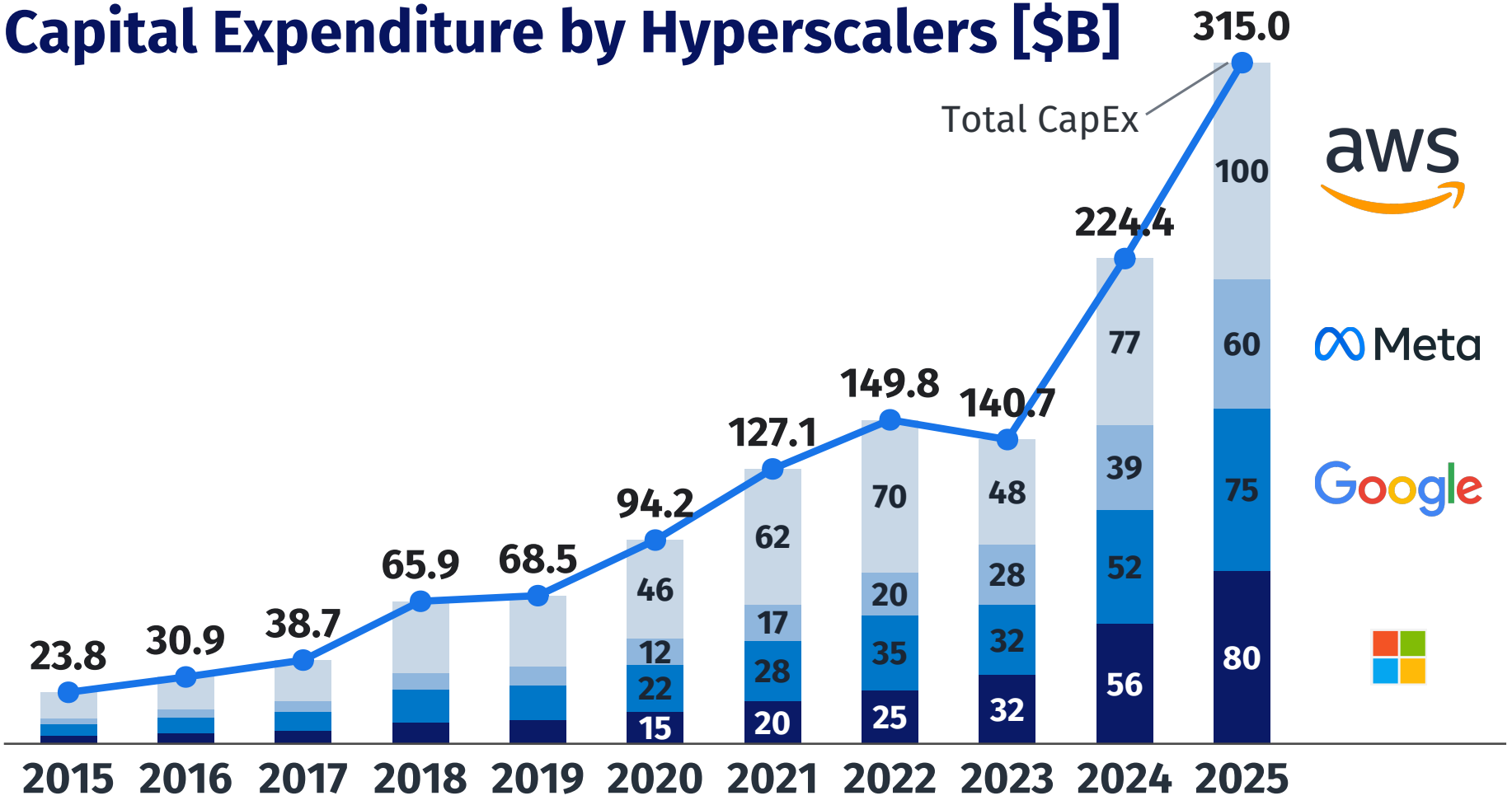
Cloud Database Is Also a Success Story

SaaS IPO Ranking

Rank	Company	IPO Year	IPO Size ($B)
1	**Snowflake**	2020	3.4
2	UiPath	2021	1.34
3	Dropbox	2018	0.756
4	Zoom	2019	0.751
5	Datadog	2019	0.648

AI Investment Is Shaping the Future

Capital Expenditure by Hyperscalers [$B]



Source: DC Pulse

How to Ride the AI Investment Wave for DB?

How to Ride the AI Investment Wave for DB?

- Issue: Existing GPU DBs remain single-node only

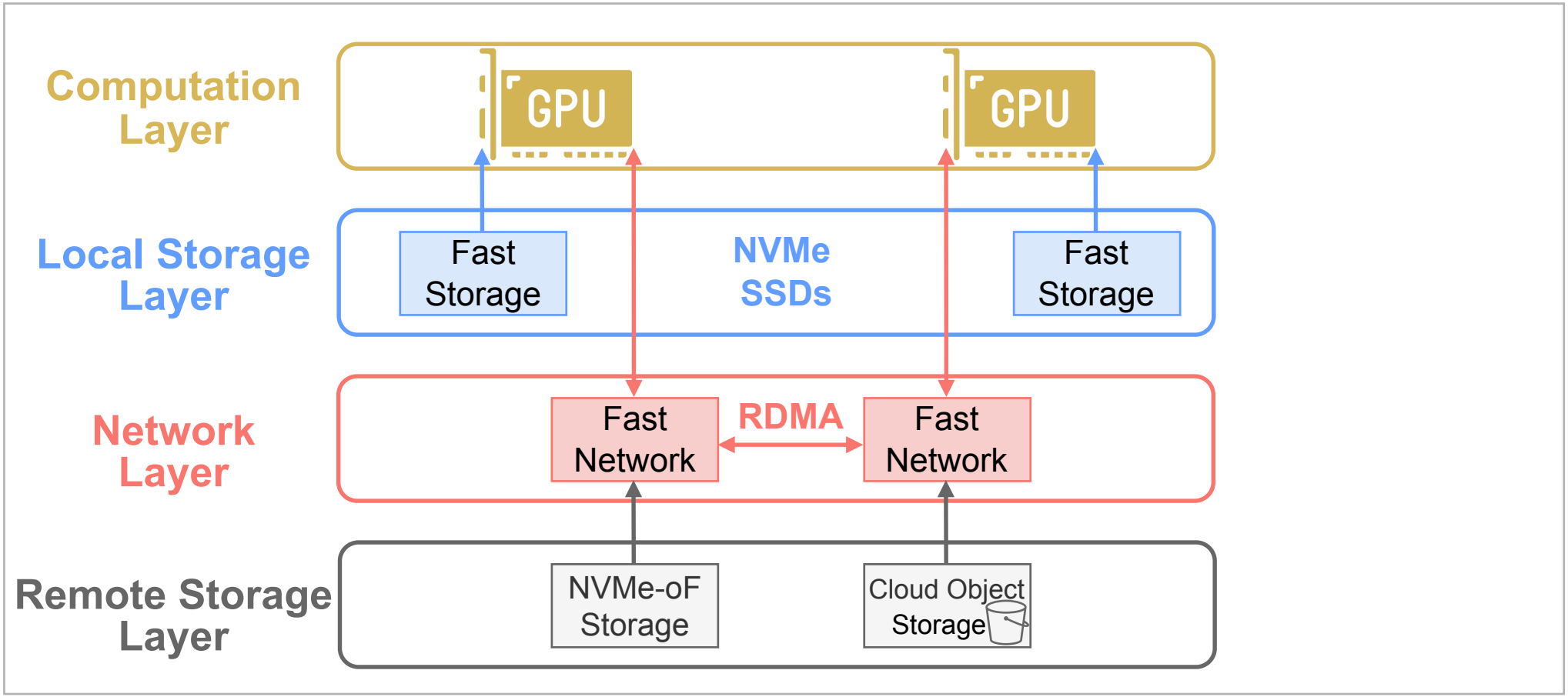
How to Ride the AI Investment Wave for DB?

- Issue: Existing GPU DBs remain single-node only
- **RQ: How do we build a GPU DB for disaggregation?**

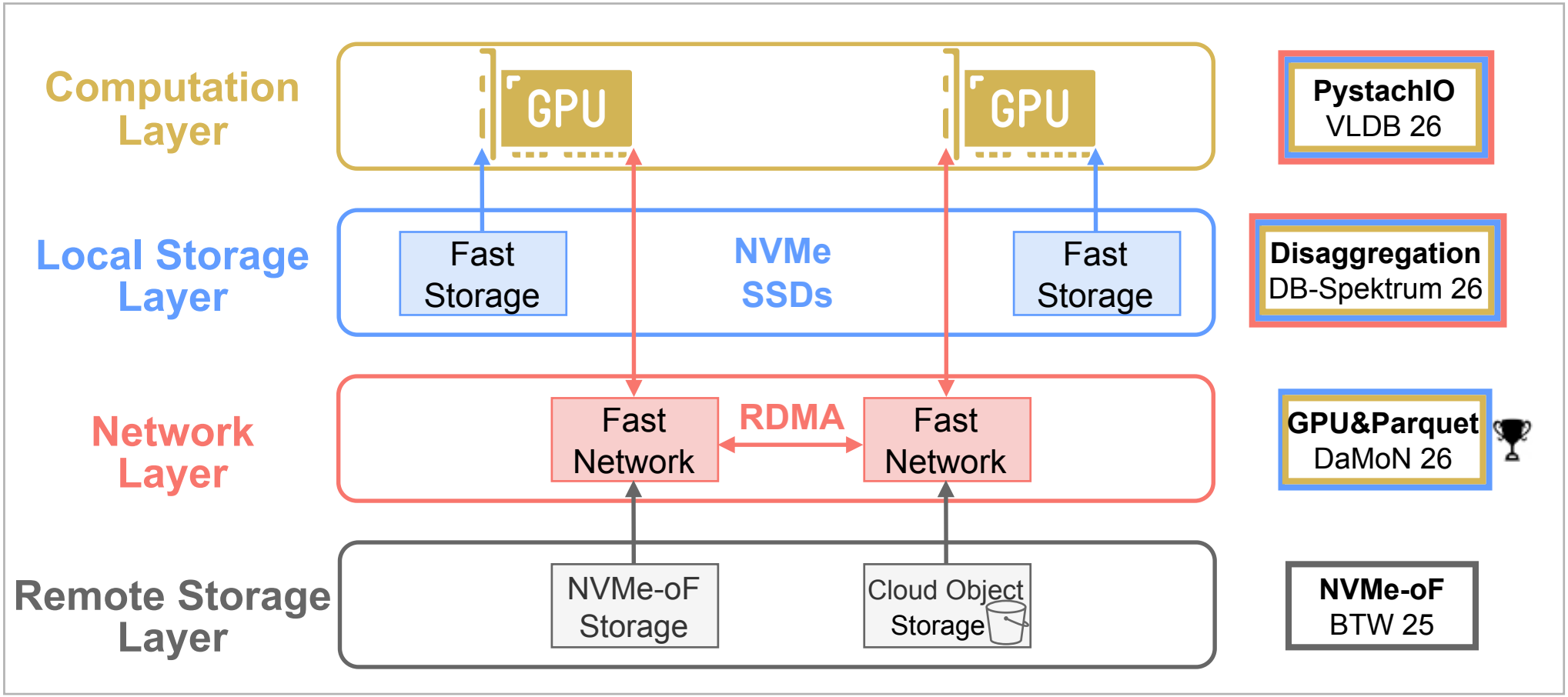
How to Ride the AI Investment Wave for DB?

- Issue: Existing GPU DBs remain single-node only
- **RQ: How do we build a GPU DB for disaggregation?**
- Challenges from fast networks and fast storage

My Research

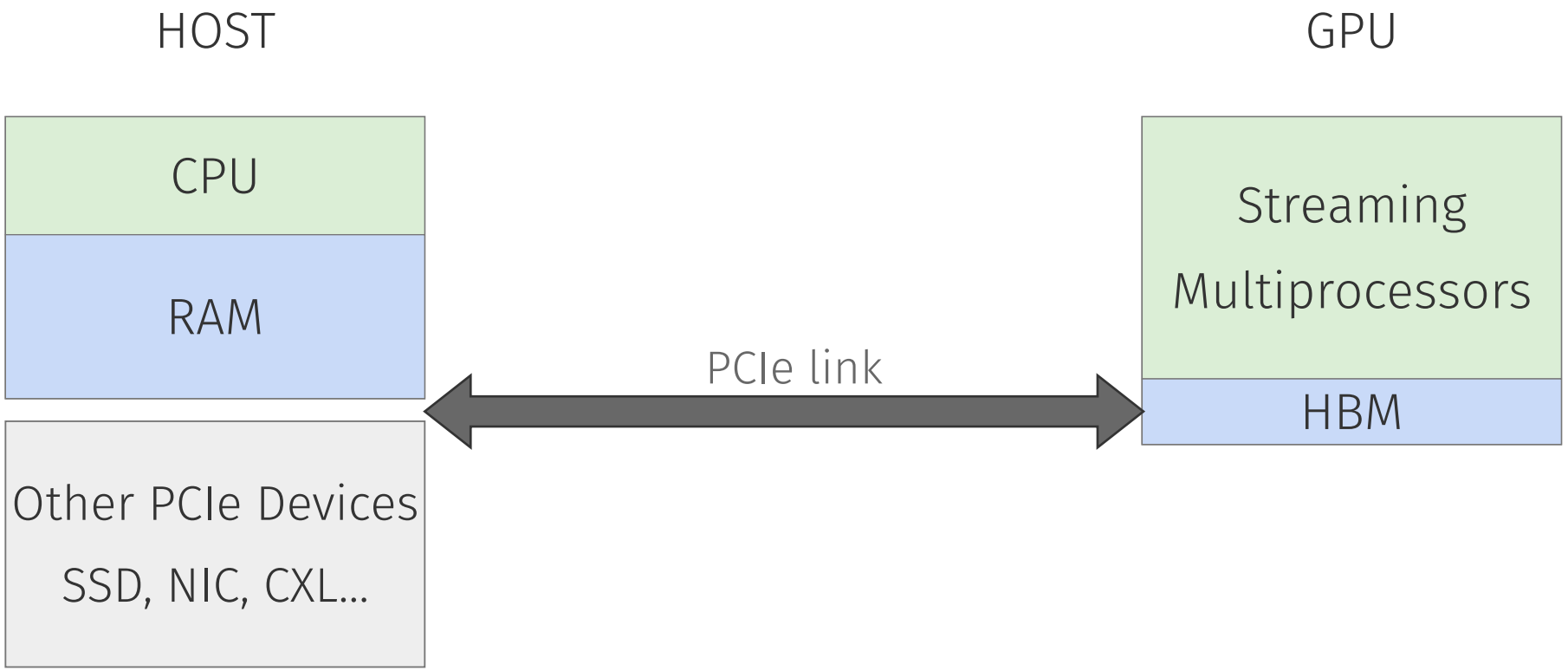

Computation Layer
GPU
GPU
Local Storage Layer
Fast Storage
NVMe SSDs
Fast Storage
Network Layer
Fast Network
RDMA
Fast Network
Remote Storage Layer
NVMe-oF Storage
Cloud Object Storage

My Research Status



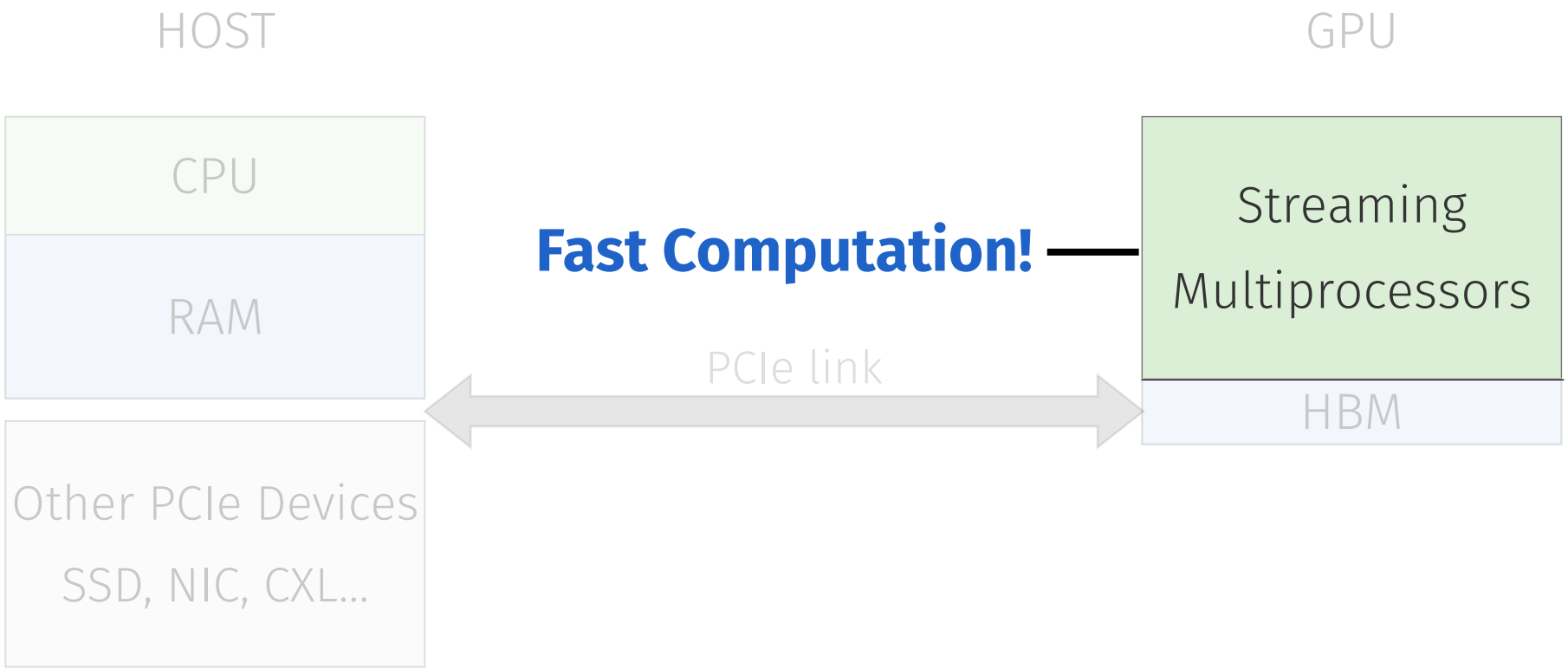
Background: GPU Data Processing

GPU System Primitives



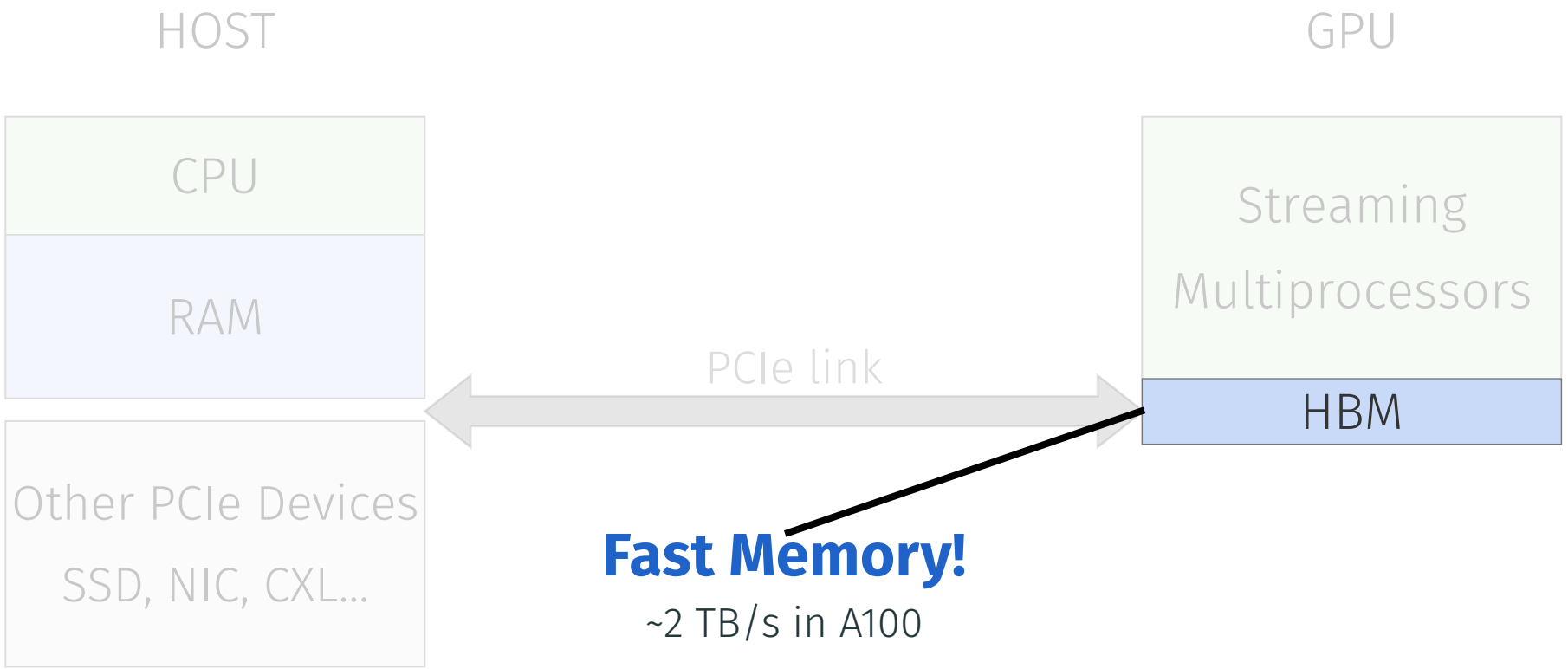
* NVLink C2C (chip-to-chip) excluded.

GPU System Primitives



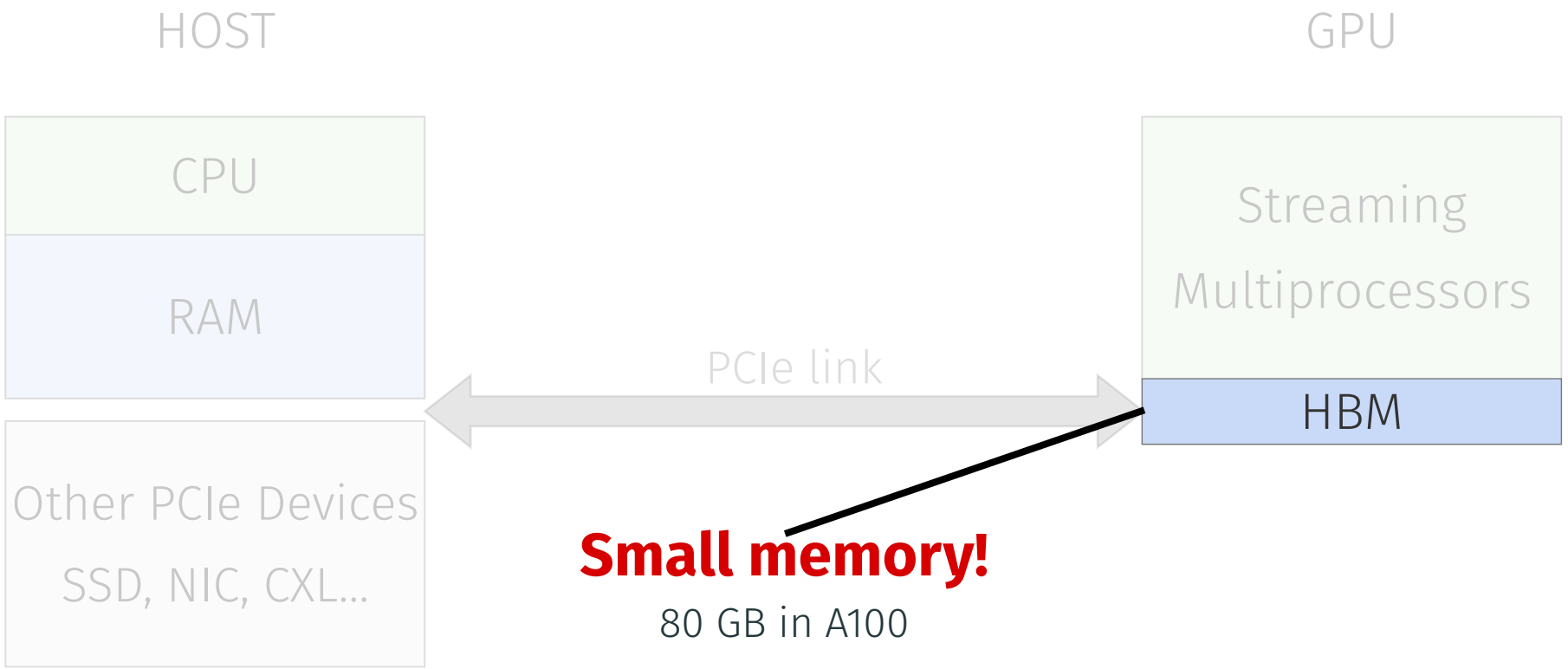
* NVLink C2C (chip-to-chip) excluded.

GPU System Primitives



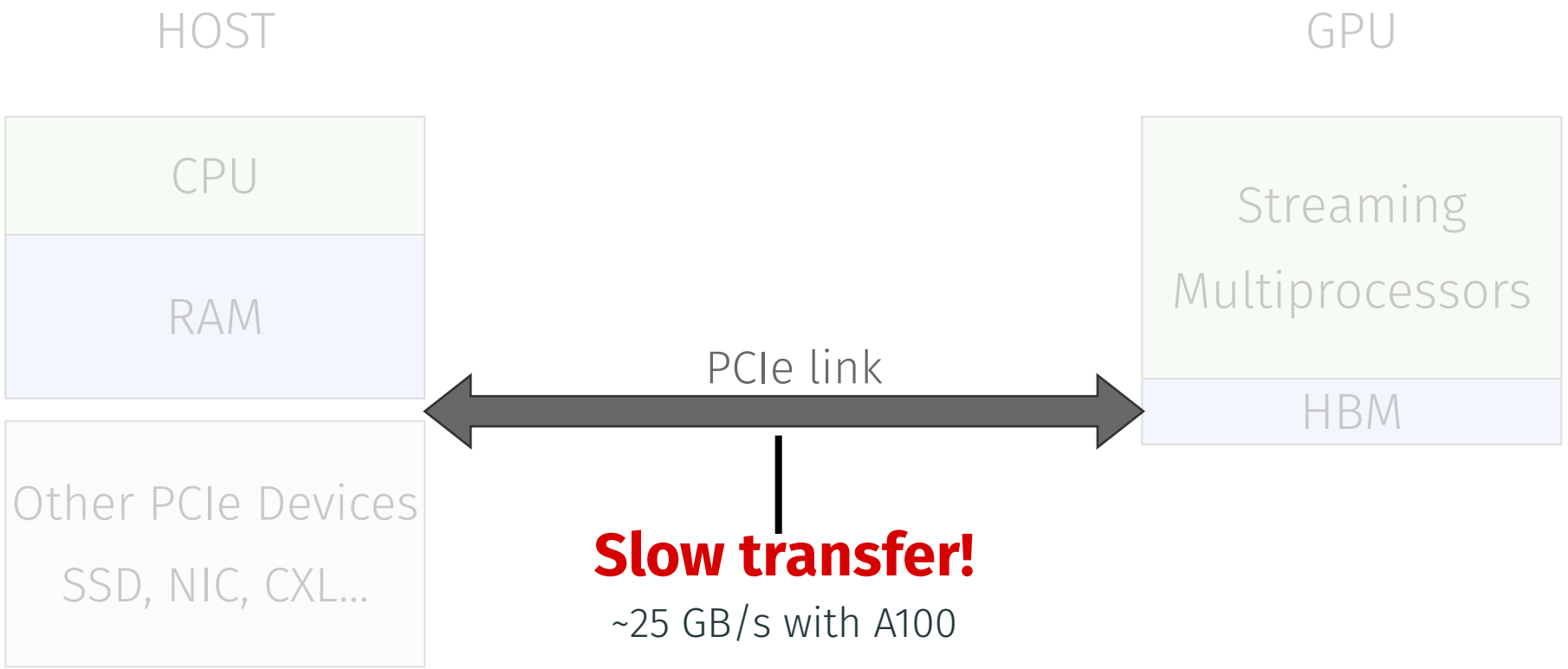
* NVLink C2C (chip-to-chip) excluded.

GPU System Primitives: Problems & Challenges



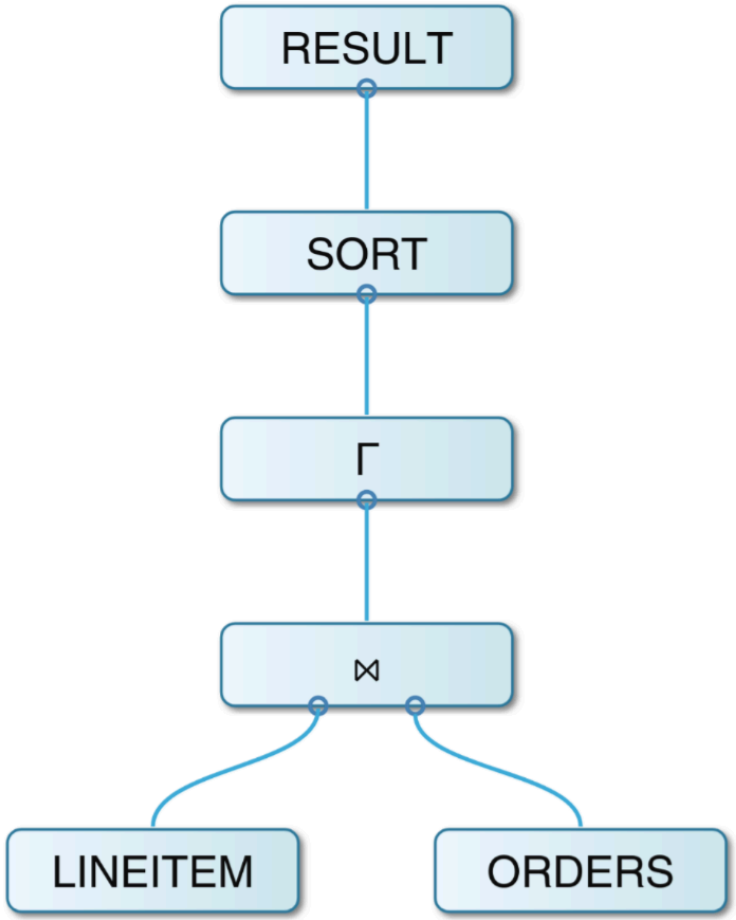
* NVLink C2C (chip-to-chip) excluded.

GPU System Primitives: Problems & Challenges

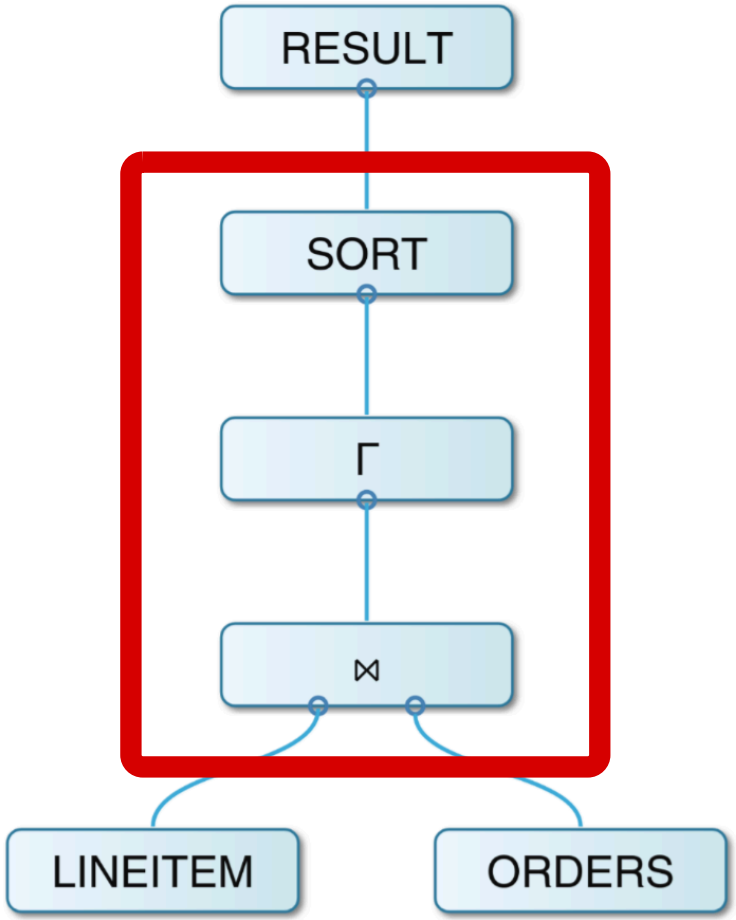


* NVLink C2C (chip-to-chip) excluded.

GPU Data Systems

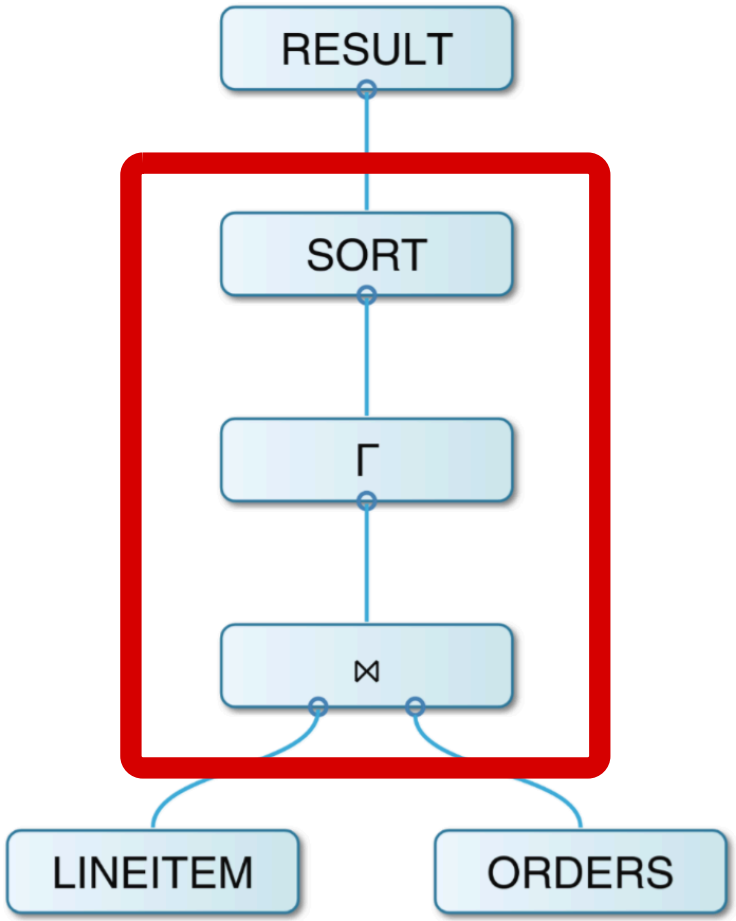


GPU Data Systems



1. join kernel
2. groupby kernel
3. sort kernel

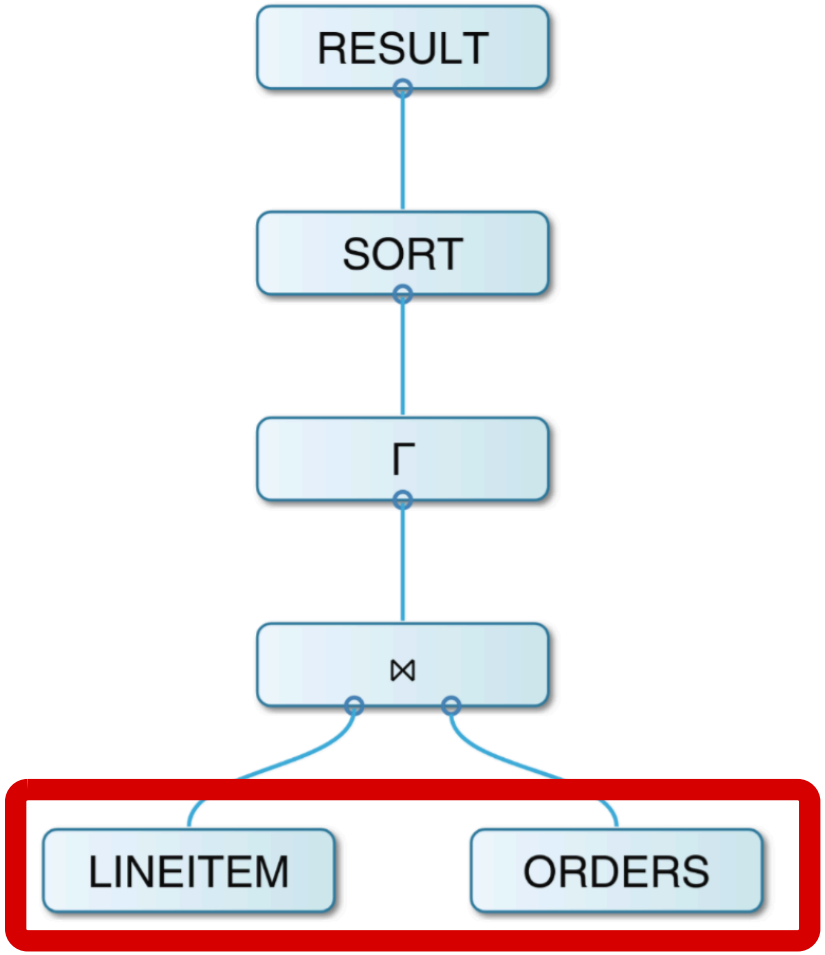
GPU Data Systems



1. join kernel
2. groupby kernel
3. sort kernel

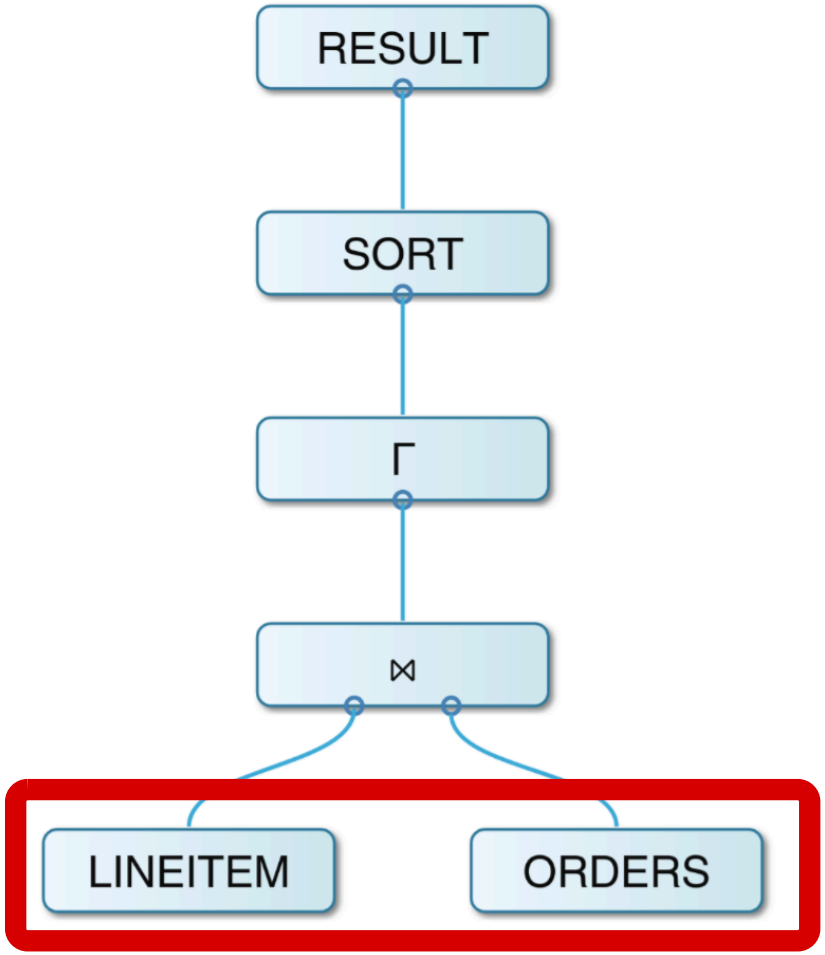
Not in this talk

GPU Data Systems



Where to put base tables?

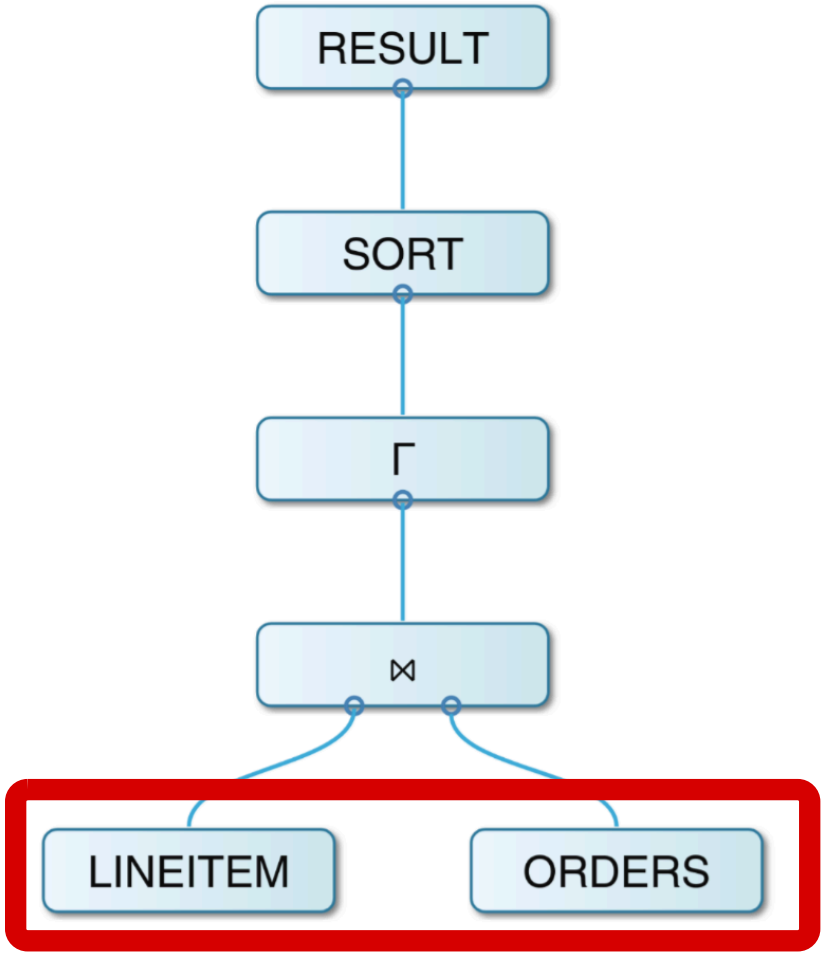
GPU Data Systems



Where to put base tables?

- GPU Memory: HBM

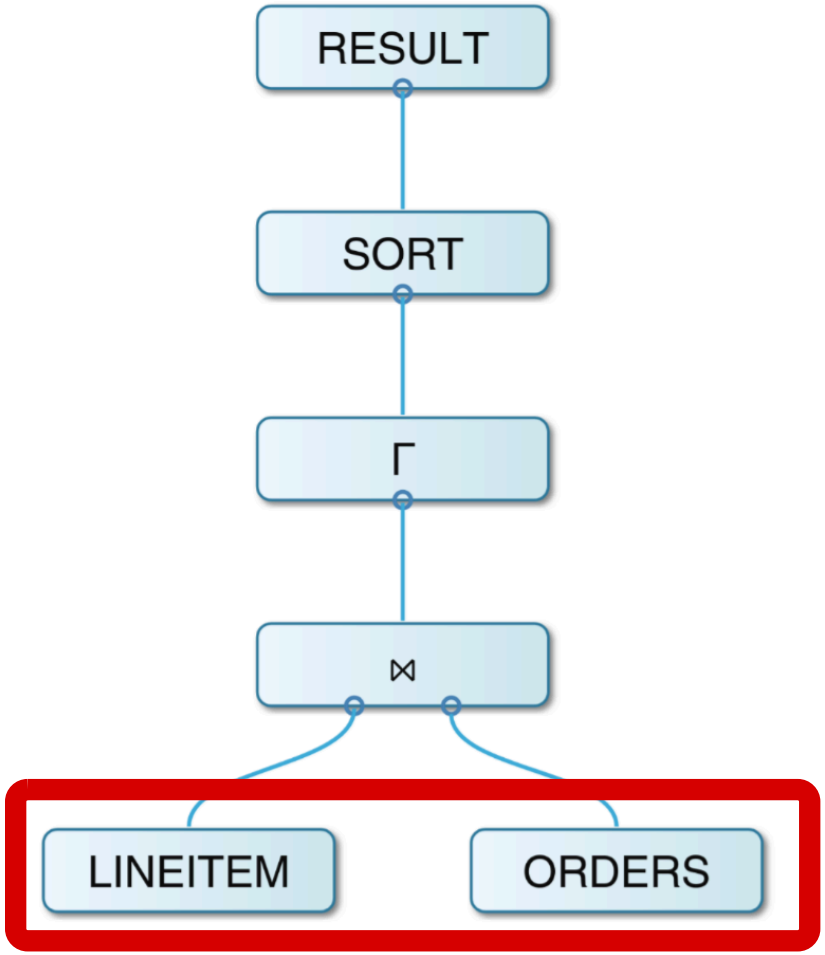
GPU Data Systems



Where to put base tables?

- GPU Memory: HBM
- CPU Memory

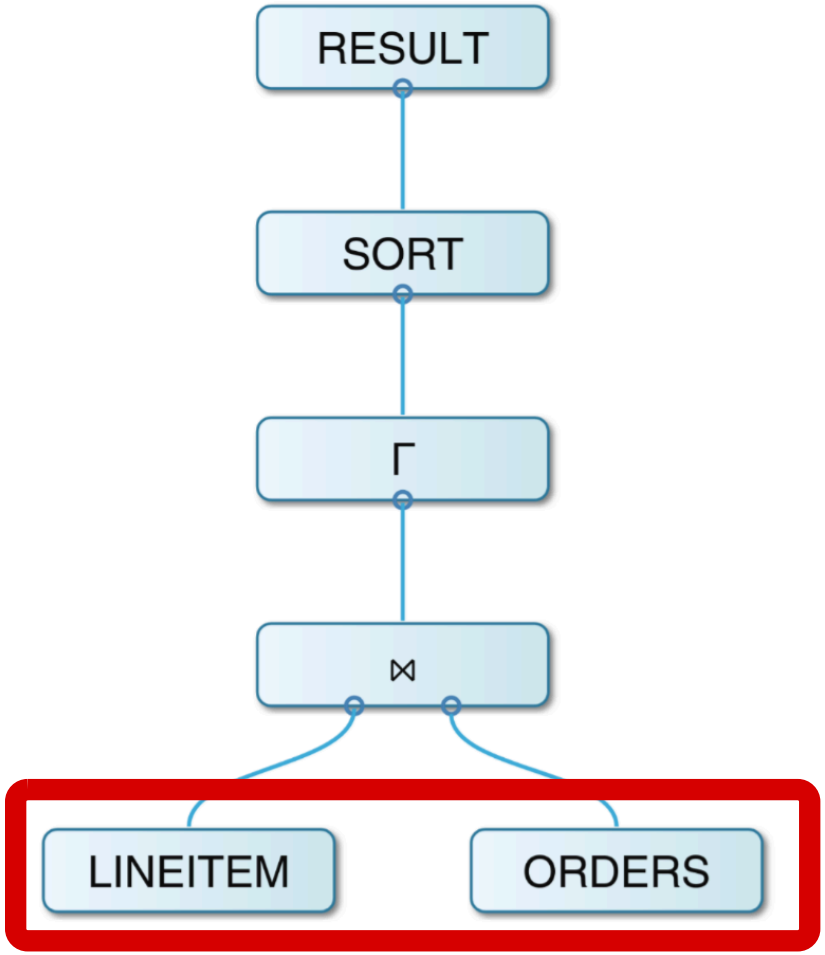
GPU Data Systems



Where to put base tables?

- GPU Memory: HBM
- CPU Memory
- SSDs

GPU Data Systems



Where to put base tables?

- GPU Memory: HBM
- CPU Memory
- SSDs
- NICs: remote storage

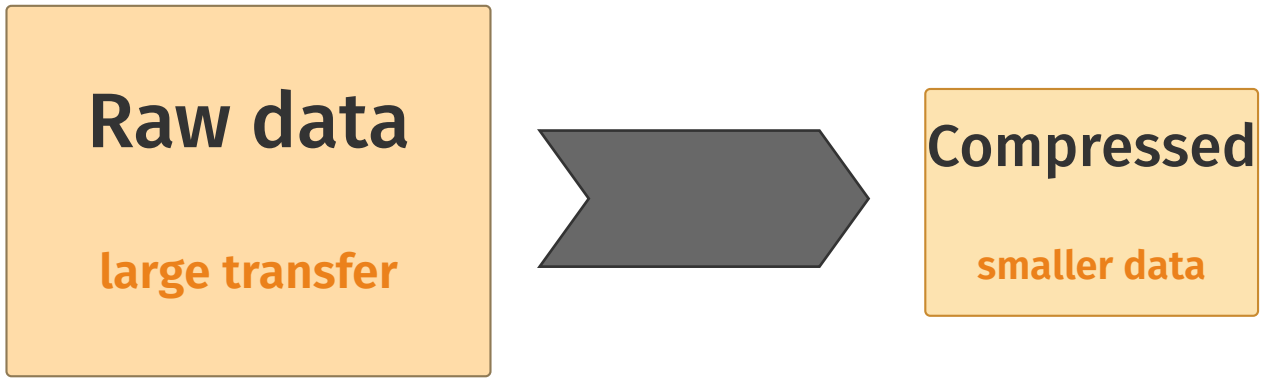
Figure Source: HyPer DB

Common Technique 1: Compression

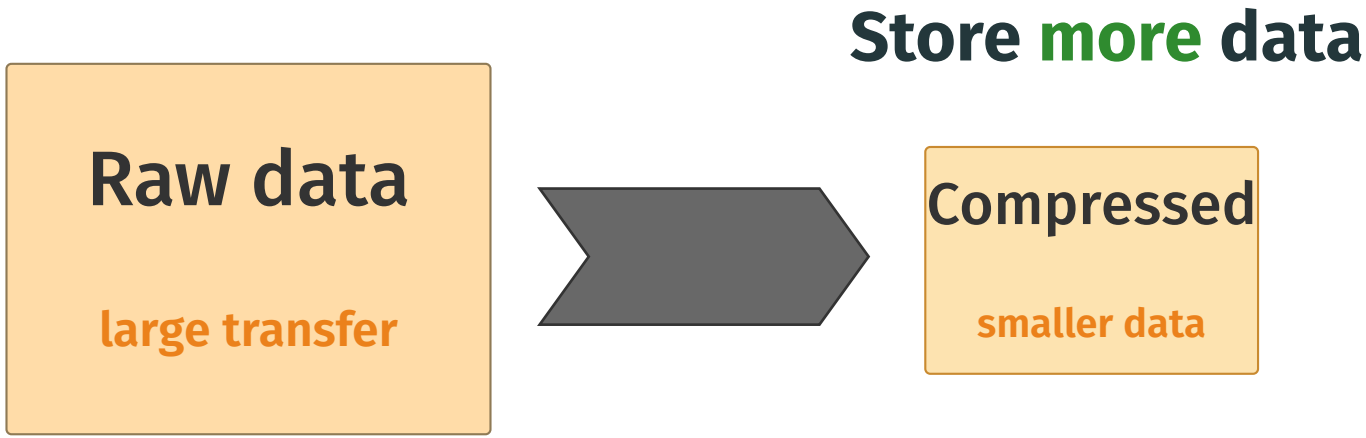
Raw data

large transfer

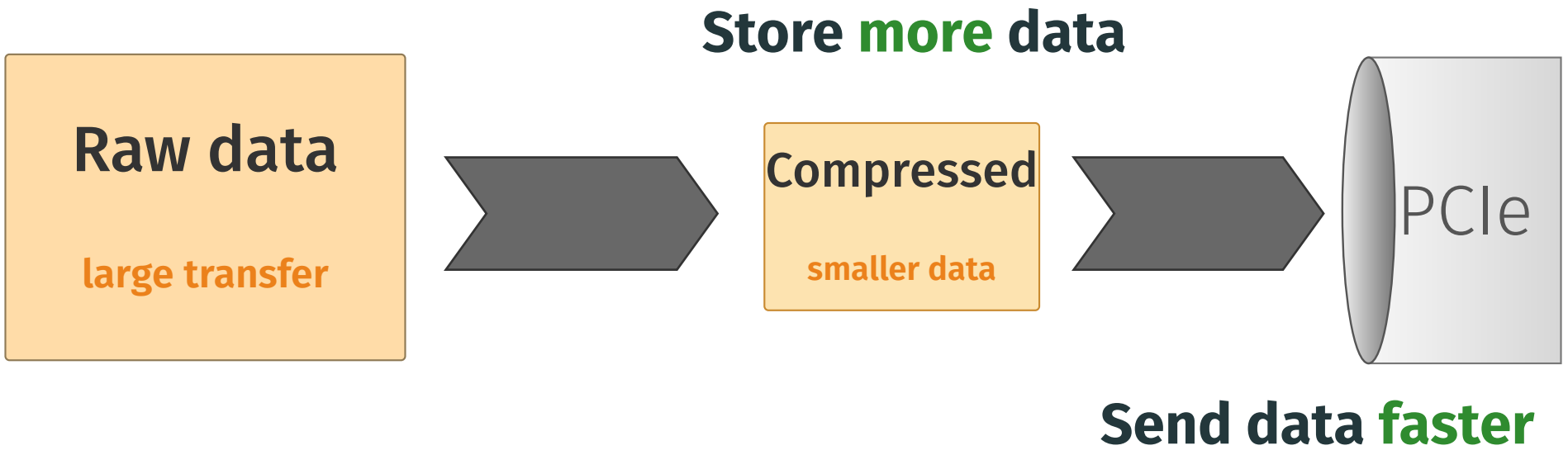
Common Technique 1: Compression



Common Technique 1: Compression

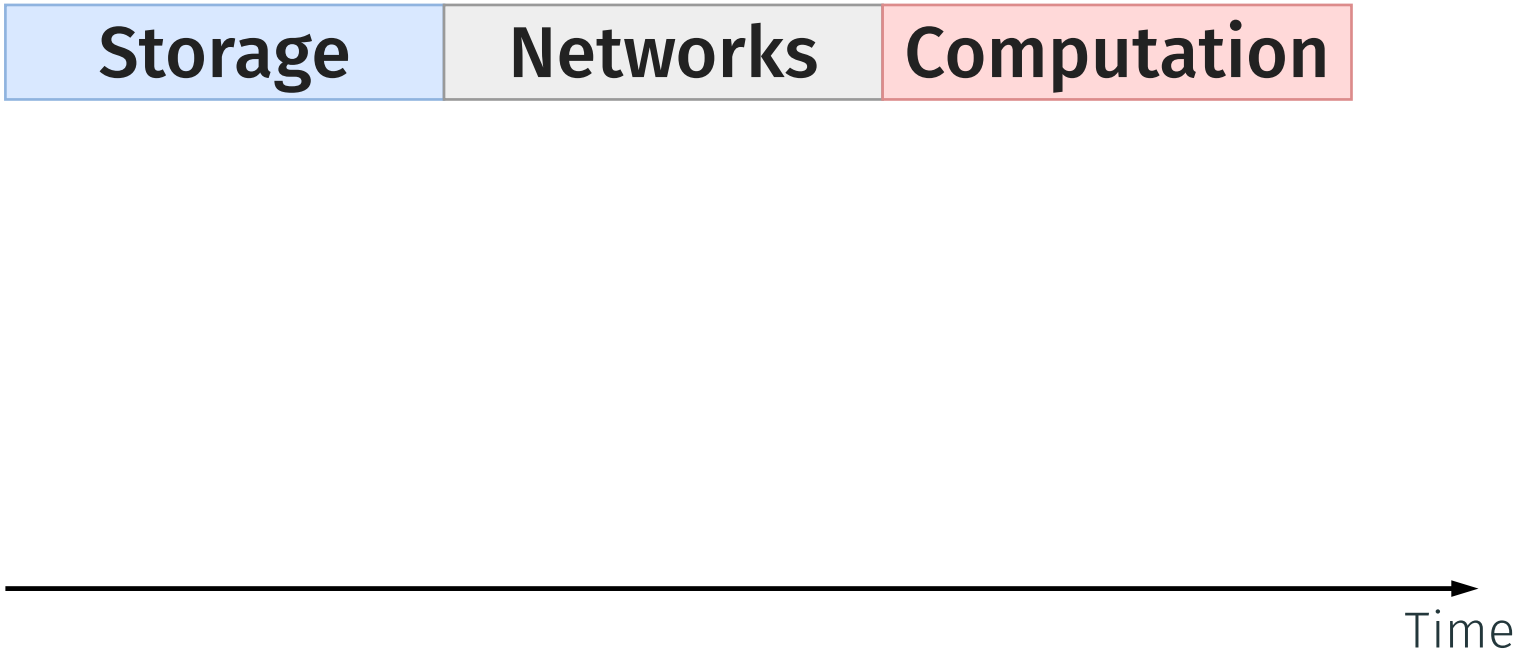


Common Technique 1: Compression



Common Technique 2: Overlapping

Blocking

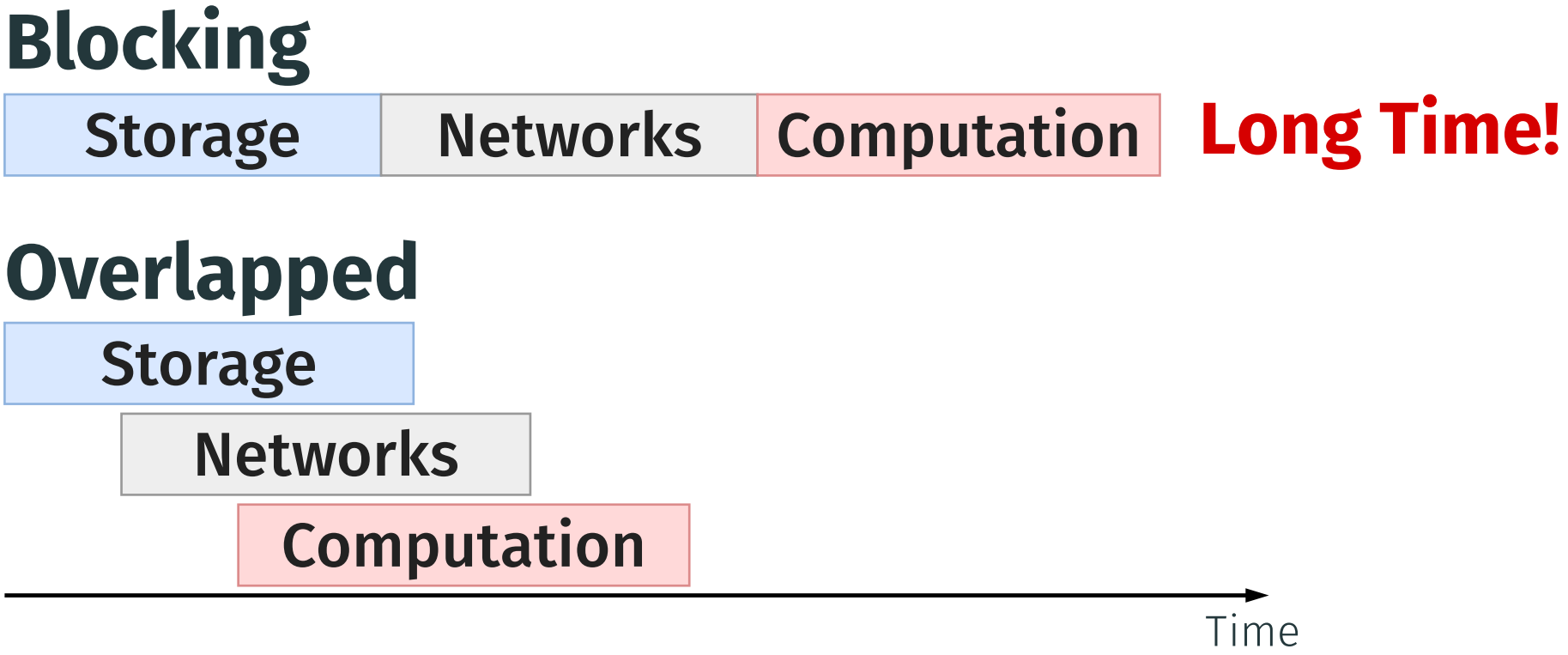


Common Technique 2: Overlapping

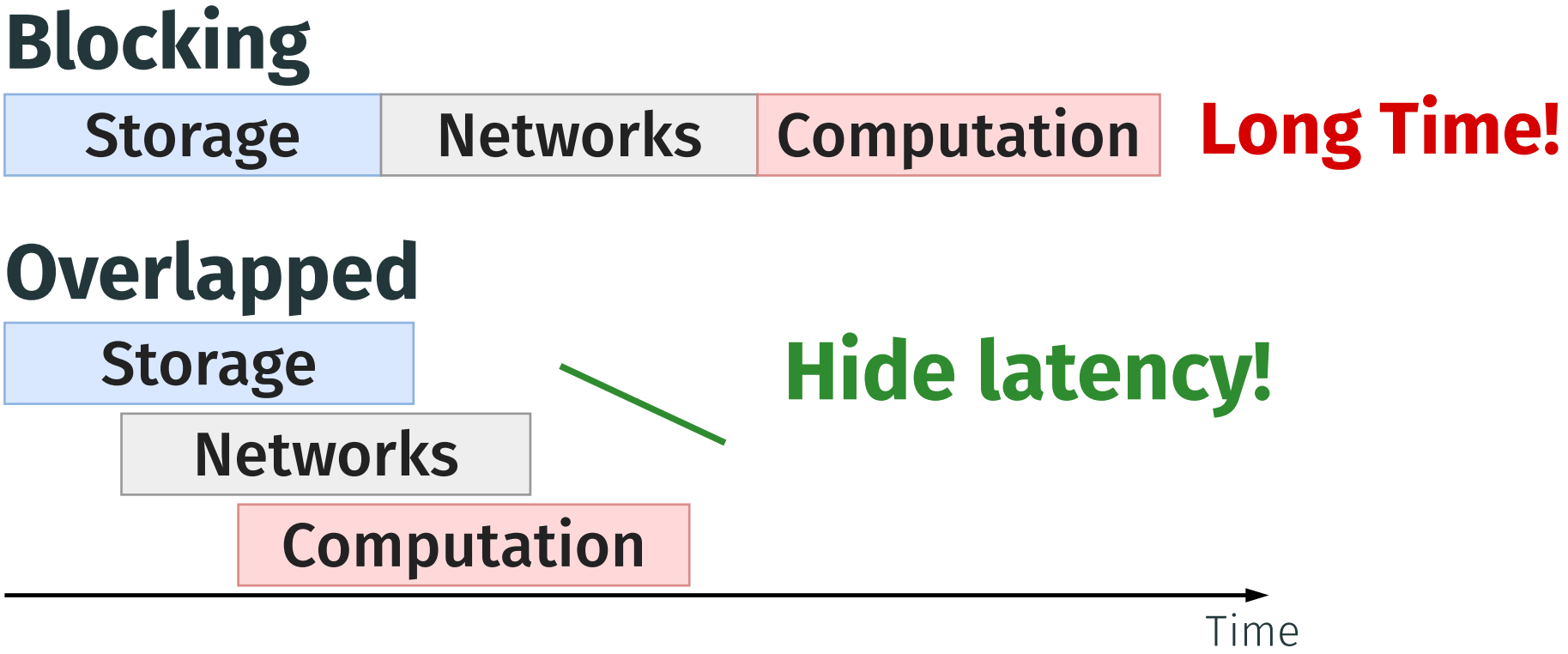
Blocking

Long Time!

Common Technique 2: Overlapping



Common Technique 2: Overlapping



1. GPU & Parquet, DaMoN 2026

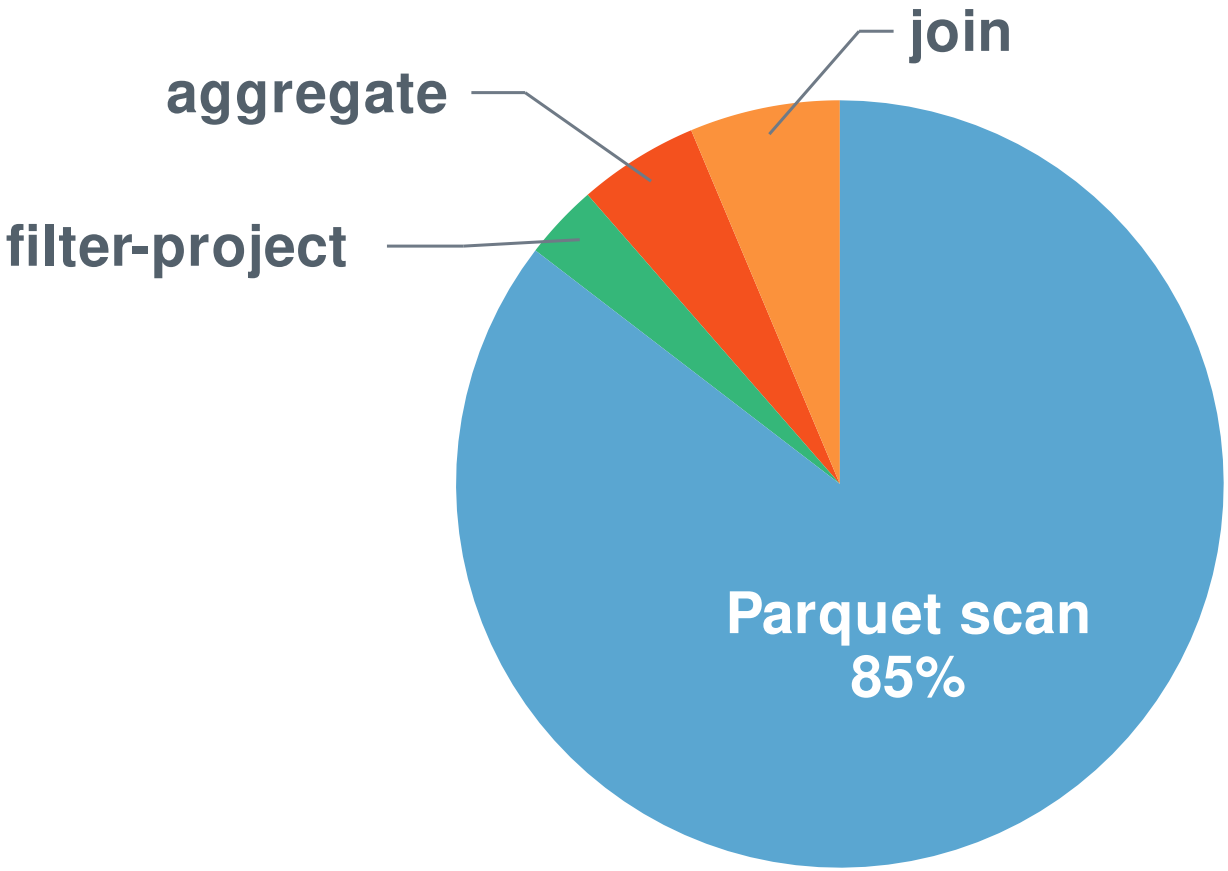
Problem: GPU Bottleneck With Parquet

Problem: GPU Bottleneck With Parquet

Is Parquet Bad for GPUs?

Problem: GPU Bottleneck With Parquet

Is Parquet Bad for GPUs?



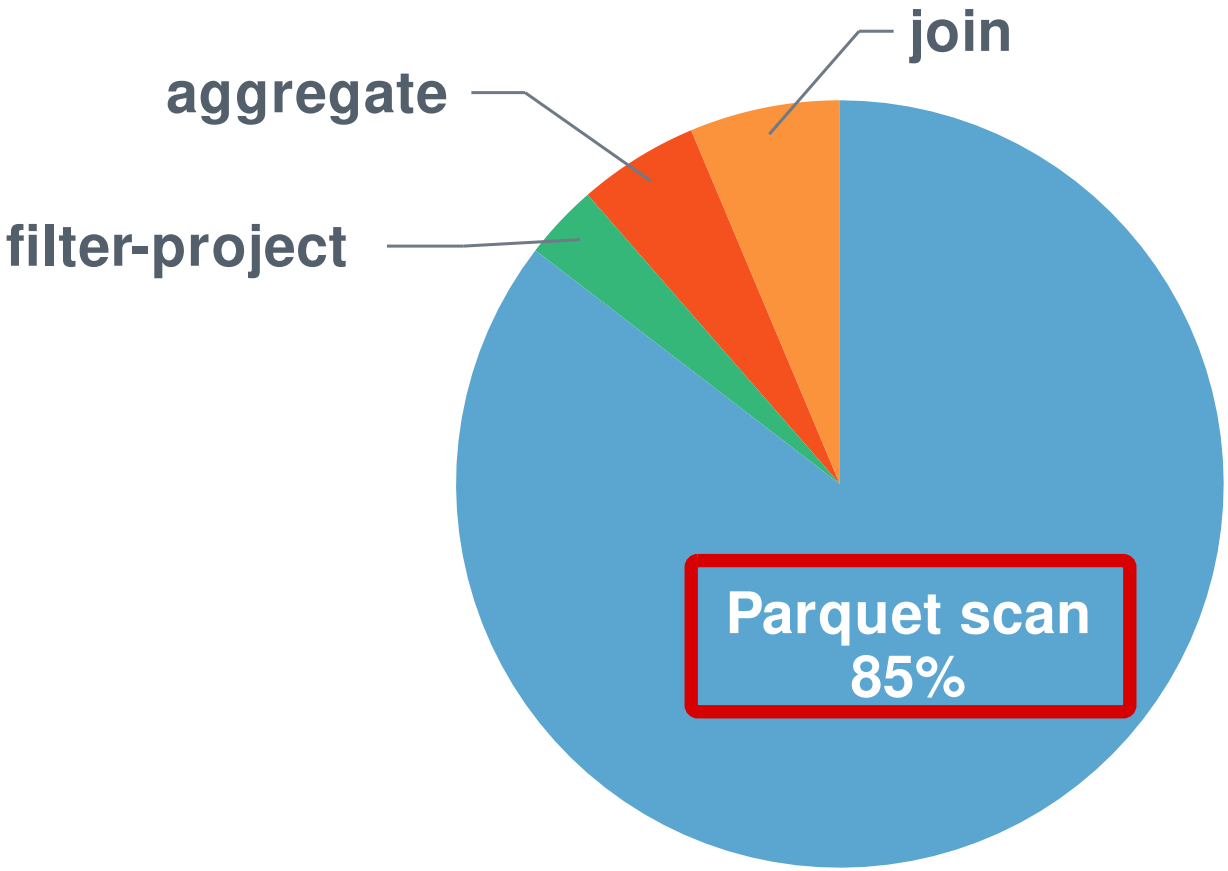
GPU TPC-H SF100 runtime breakdown.

Source: Accelerating Velox with RAPIDS cuDF, Velox-cuDF Team, 2025

Problem: GPU Bottleneck With Parquet

Is Parquet Bad for GPUs?

- **85%** in Parquet Scan



GPU TPC-H SF100 runtime breakdown.

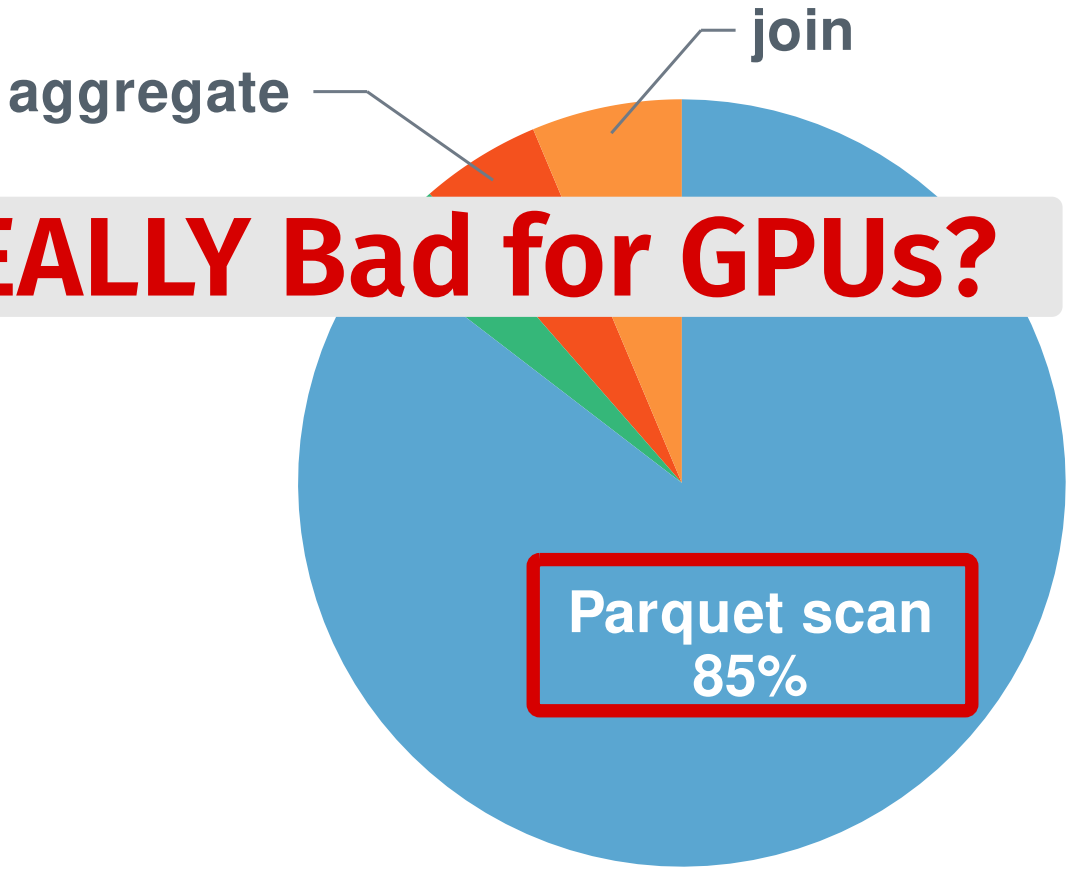
Source: Accelerating Velox with RAPIDS cuDF, Velox-cuDF Team, 2025

Problem: GPU Bottleneck With Parquet

Is Parquet REALLY Bad for GPUs?

Is Parquet Bad for GPUs?

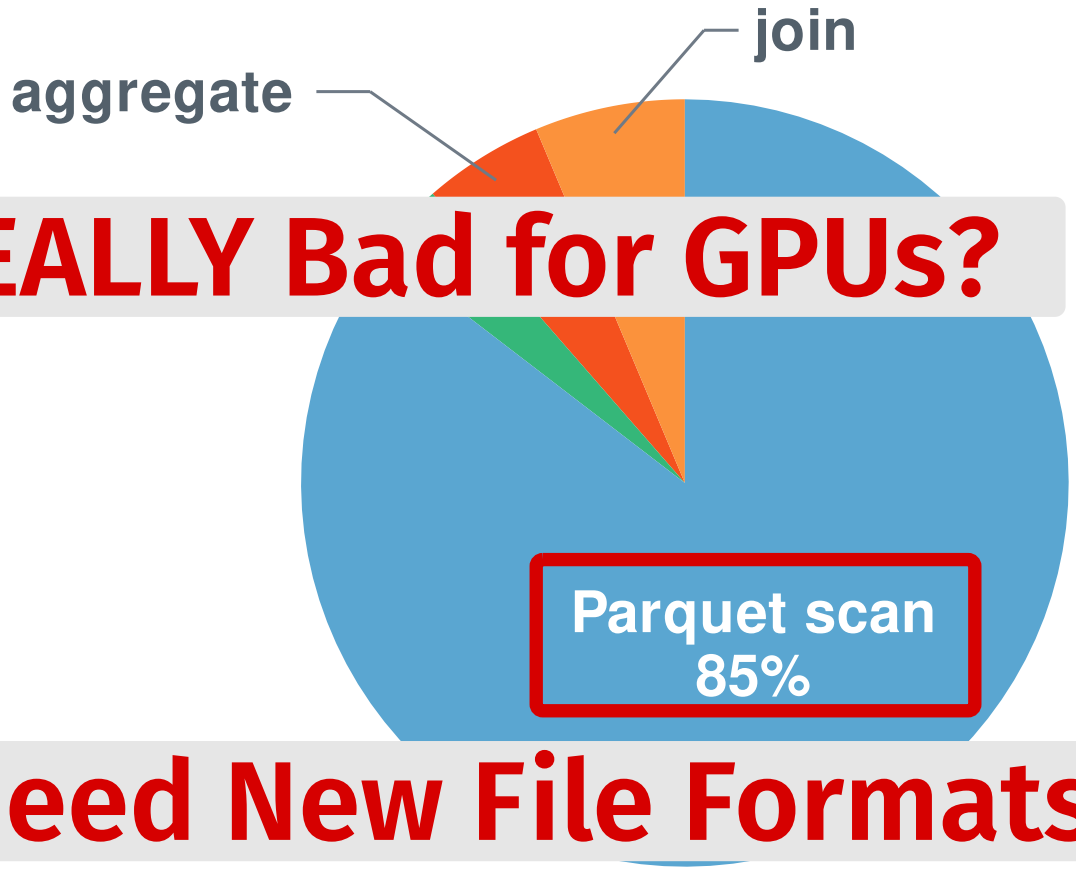
- **85%** in Parquet Scan



GPU TPC-H SF100 runtime breakdown.

Source: Accelerating Velox with RAPIDS cuDF, Velox-cuDF Team, 2025

Problem: GPU Bottleneck With Parquet



Is Parquet REALLY Bad for GPUs?

Is Parquet Bad for GPUs?

- **85%** in Parquet Scan

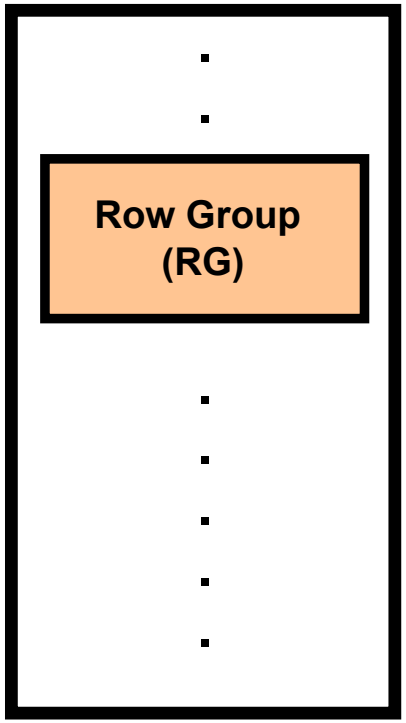
Do GPUs REALLY Need New File Formats?

GPU TPC-H SF100 runtime breakdown.

Source: Accelerating Velox with RAPIDS cuDF, Velox-cuDF Team, 2025

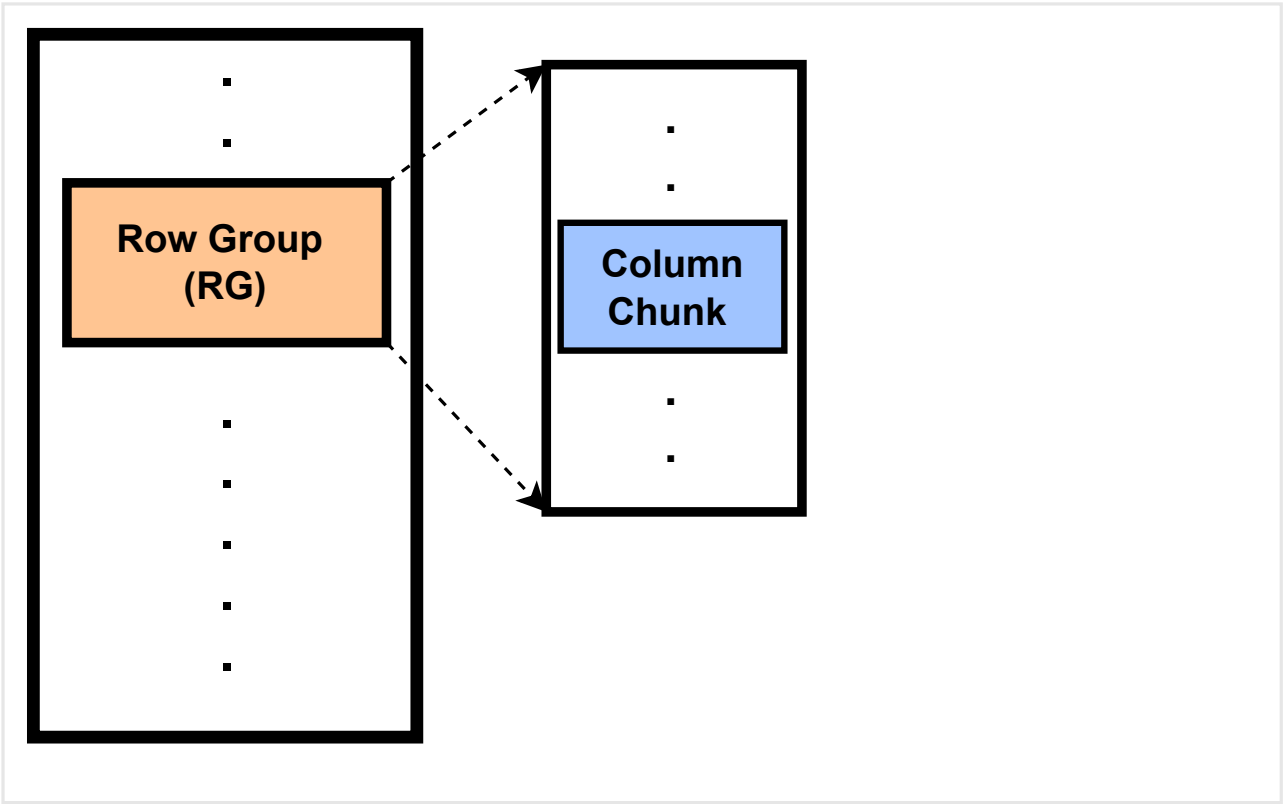
Background: Apache Parquet

- Row group



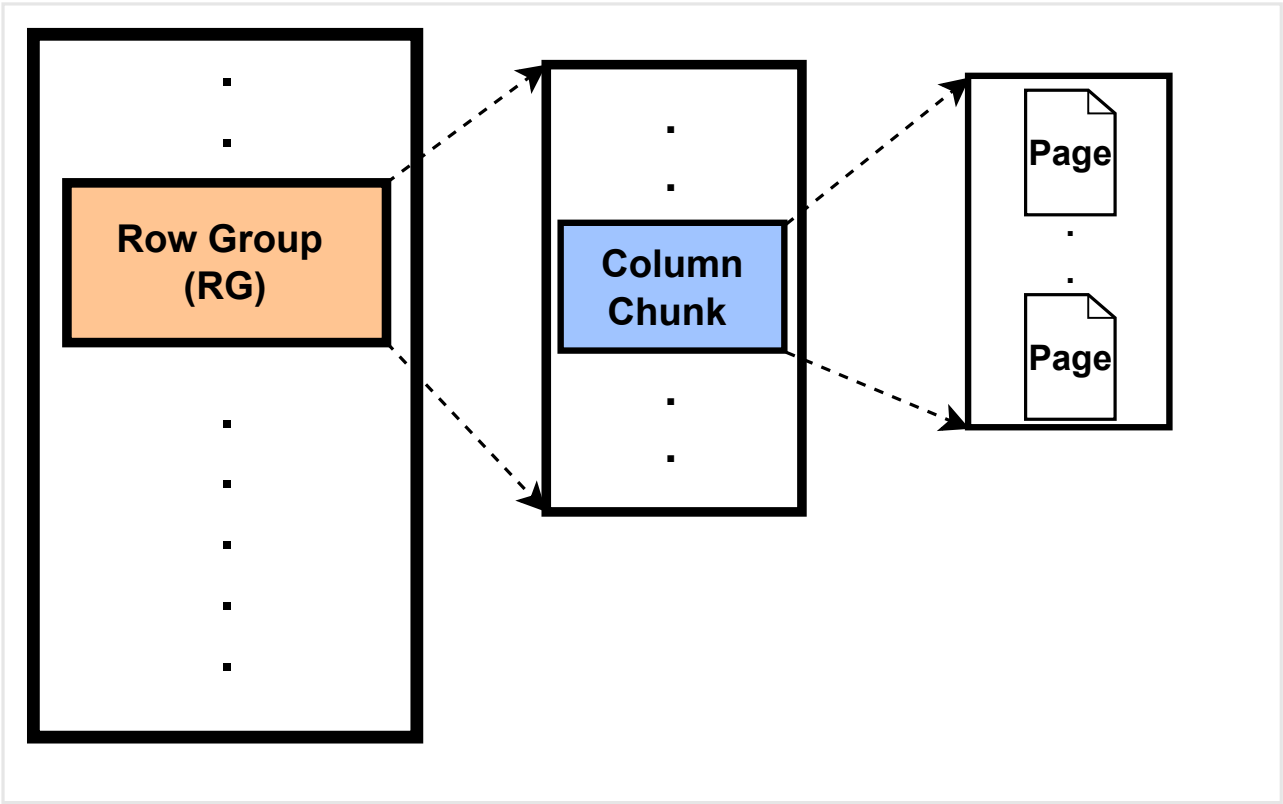
Background: Apache Parquet

- Row group
- Column chunk



Background: Apache Parquet

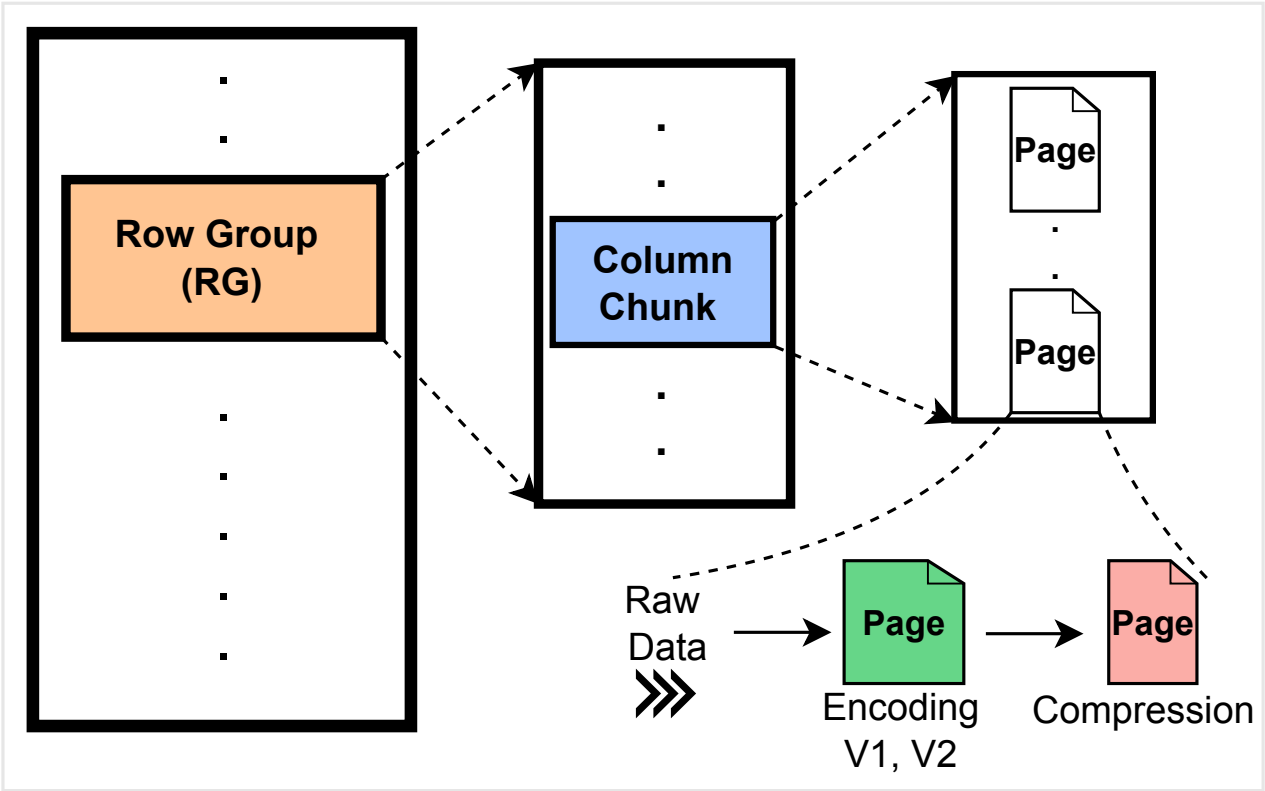
- Row group
- Column chunk
- Pages:



Parquet

Background: Apache Parquet

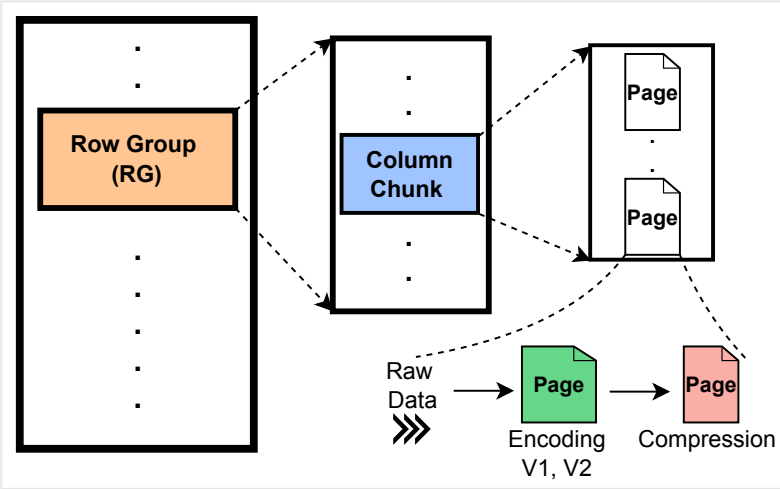
- Row group
- Column chunk
- Pages:
  - Encoded
  - Compressed



Parquet

Parquet Configuration Defaults for CPUs

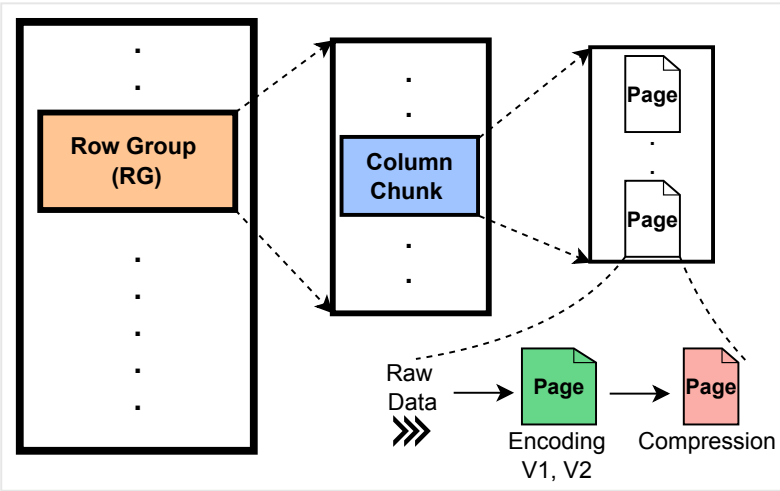
CPU Parquet Config. Defaults in **DuckDB**:



Parquet Configuration Defaults for CPUs

CPU Parquet Config. Defaults in **DuckDB**:

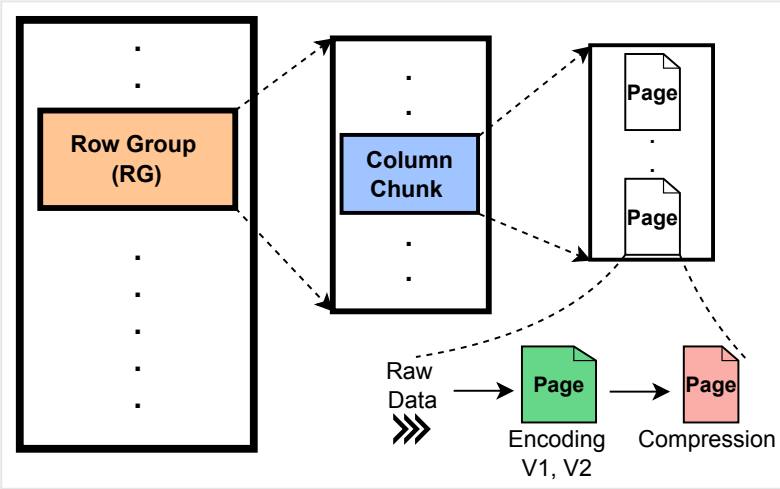
- Row group: 122880 rows



Parquet Configuration Defaults for CPUs

CPU Parquet Config. Defaults in DuckDB:

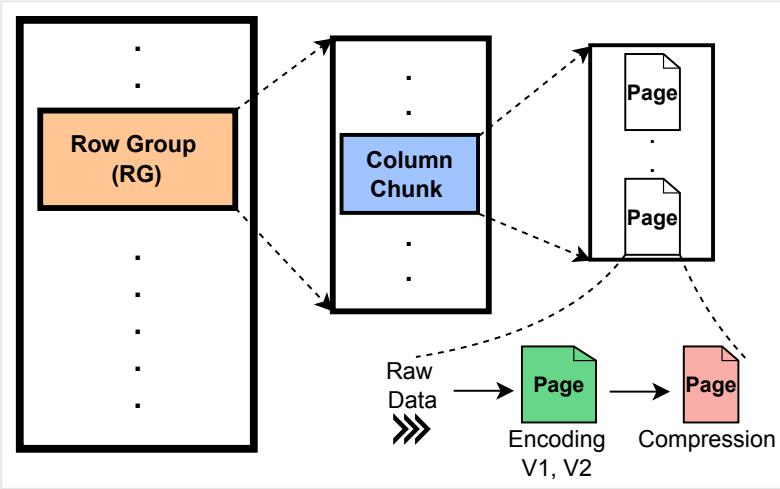
- Row group: 122880 rows
- Column chunk: 1 page only



Parquet Configuration Defaults for CPUs

CPU Parquet Config. Defaults in DuckDB:

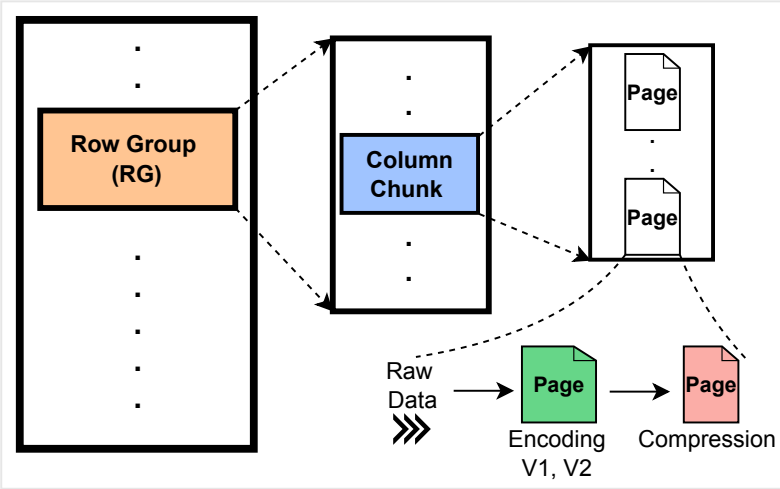
- Row group: 122880 rows
- Column chunk: 1 page only
- Encoding: V1 only



Parquet Configuration Defaults for CPUs

CPU Parquet Config. Defaults in **DuckDB**:

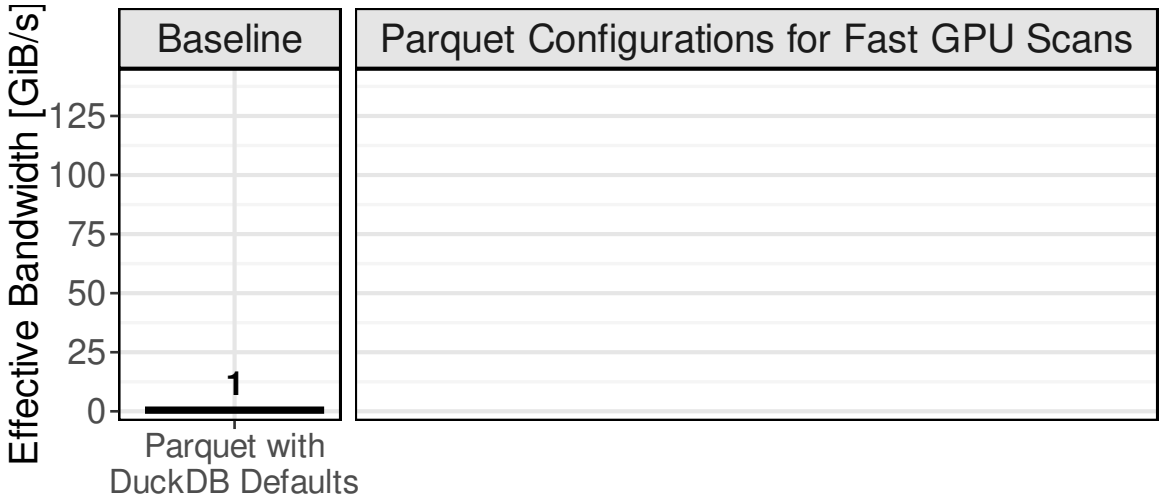
- Row group: 122880 rows
- Column chunk: 1 page only
- Encoding: V1 only
- Compression: used even without size reduction



Issue: CPU Defaults Config. ≠ GPU Optimal Config.

CPU Parquet Config. Defaults in DuckDB:

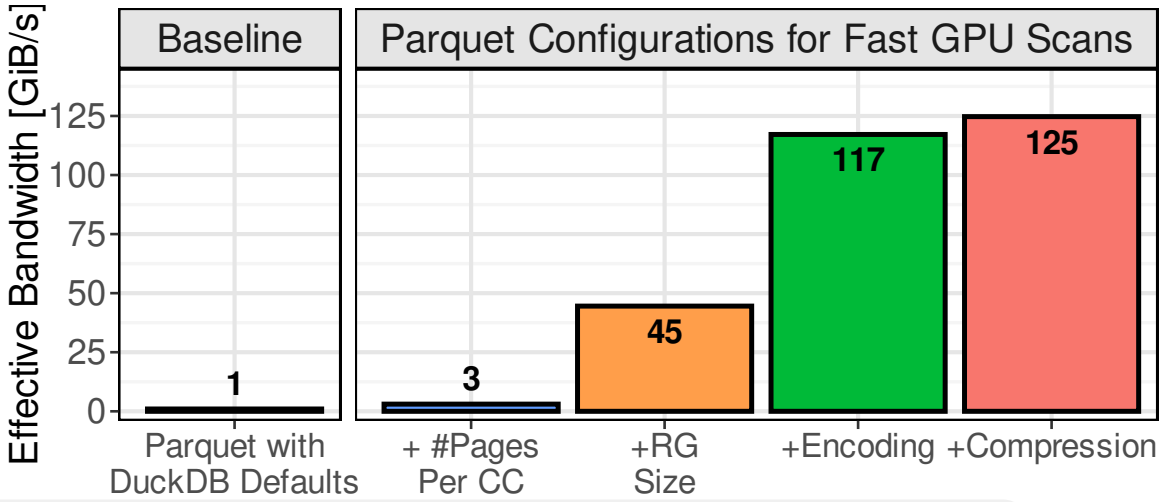
- Row group: 122880 rows
- Column chunk: 1 page only
- Encoding: V1 only
- Compression: used even without size reduction



Issue: CPU Defaults Config. ≠ GPU Optimal Config.

CPU Parquet Config. Defaults in DuckDB:

- Row group: 122880 rows
- Column chunk: 1 page only
- Encoding: V1 only
- Compression: used even without size reduction

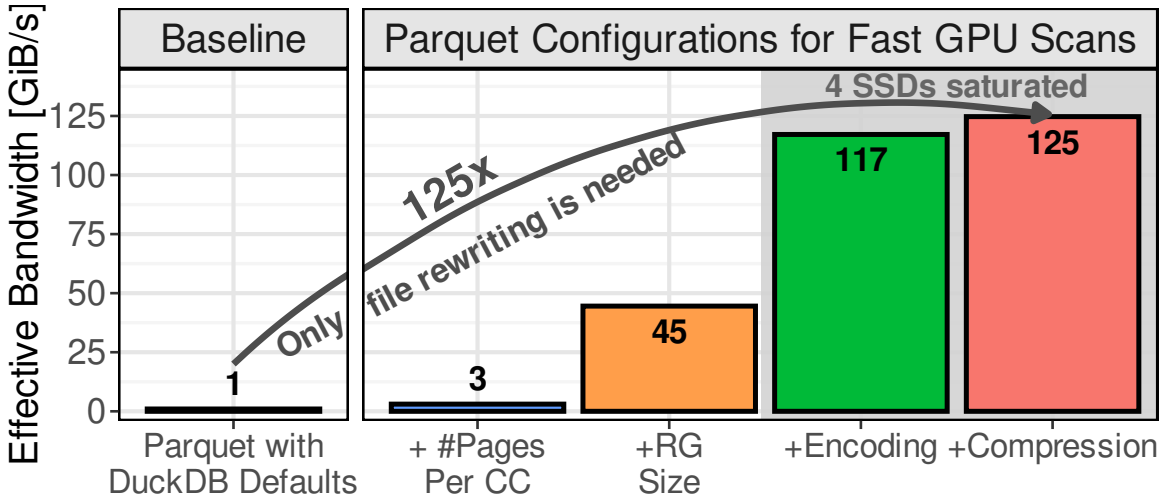


CPU Defaults Config. ≠ GPU Optimal Config.

Issue: CPU Defaults Config. ≠ GPU Optimal Config.

CPU Parquet Config. Defaults in DuckDB:

- Row group: 122880 rows
- Column chunk: 1 page only
- Encoding: V1 only
- Compression: used even without size reduction



2. PystachIO, VLDB 2026

What is missing in current GPU DBs?

	CUDA-X Lib
GPU Compute	✓
Storage	✓
Network	✓

What is missing in current GPU DBs?

	CUDA-X Lib	TQP
GPU Compute	✓	✓
Storage	✓	✗
Network	✓	✗

What is missing in current GPU DBs?

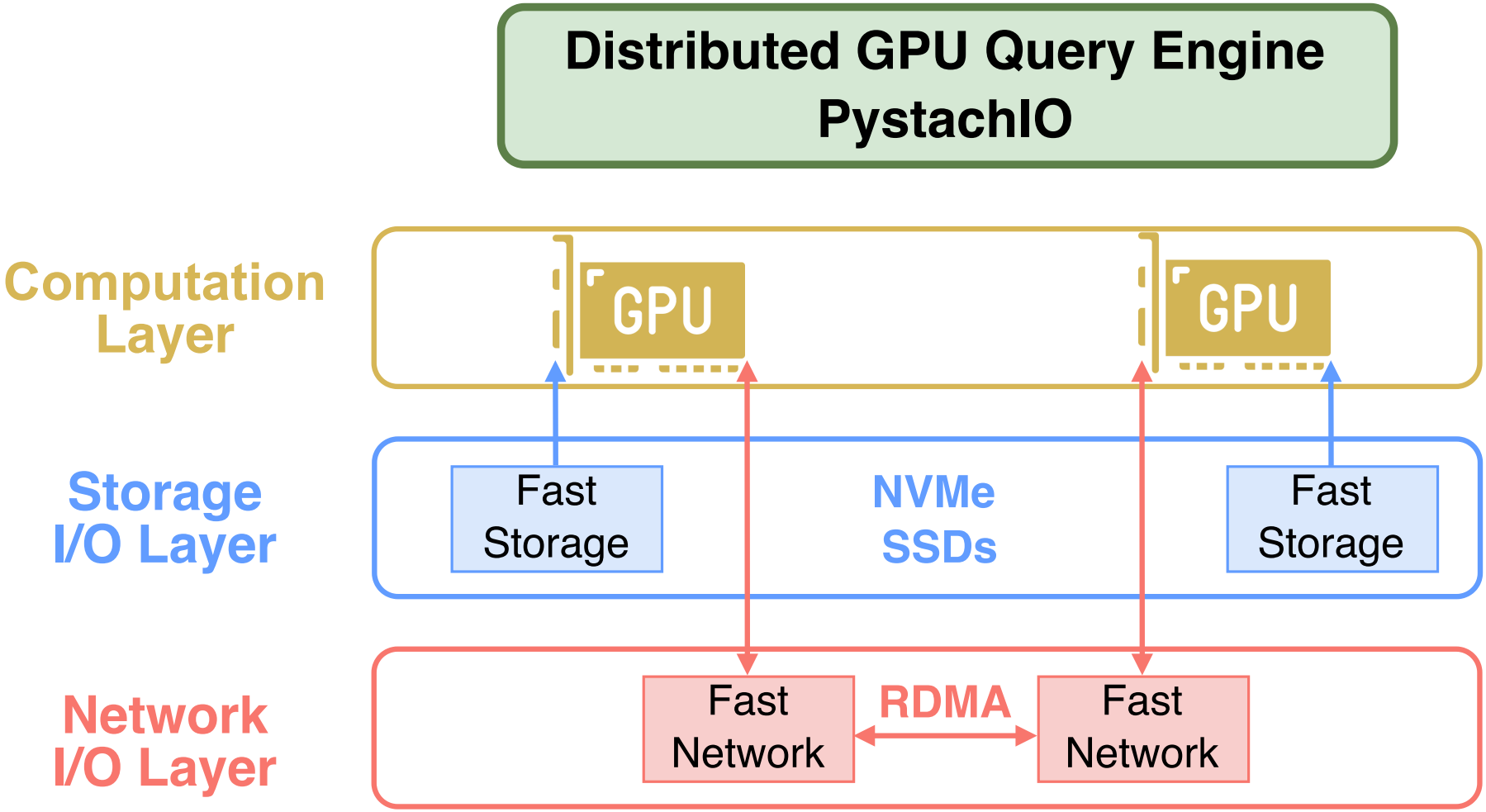
	CUDA-X Lib	TQP	Cloud OLAP
GPU Compute	✓	✓	✗
Storage	✓	✗	✓
Network	✓	✗	✓

What is missing in current GPU DBs?

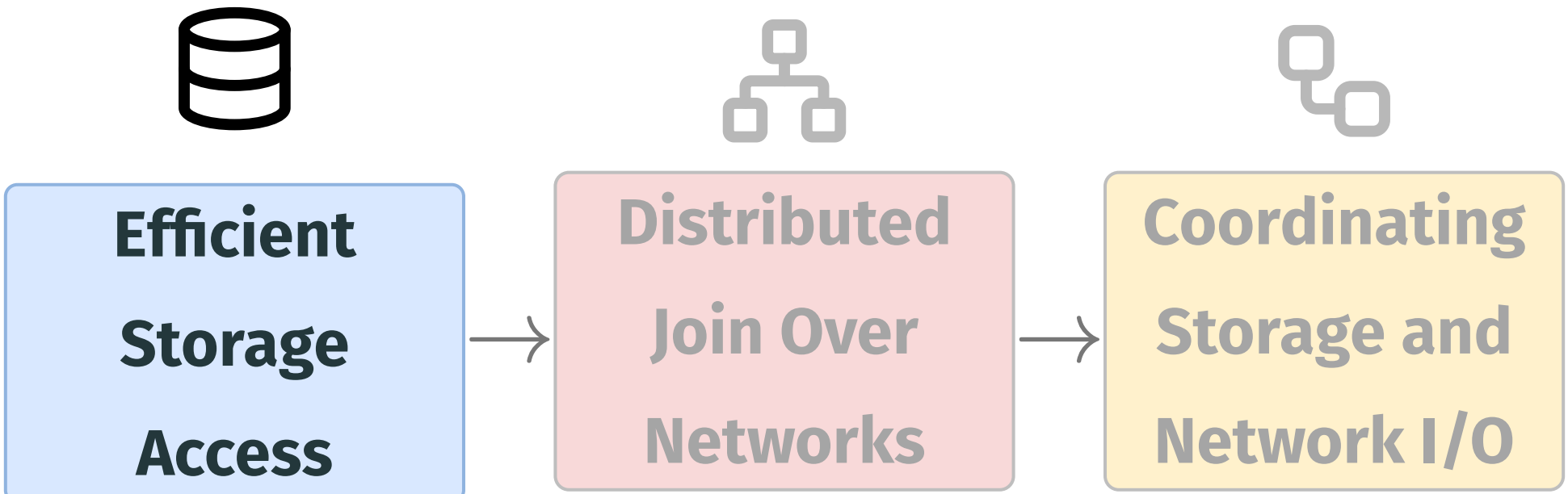
	CUDA-X Lib	TQP	Cloud OLAP
GI			
Network	✓	✗	✓

How to build a distributed GPU DB with storage & network?

PystachIO

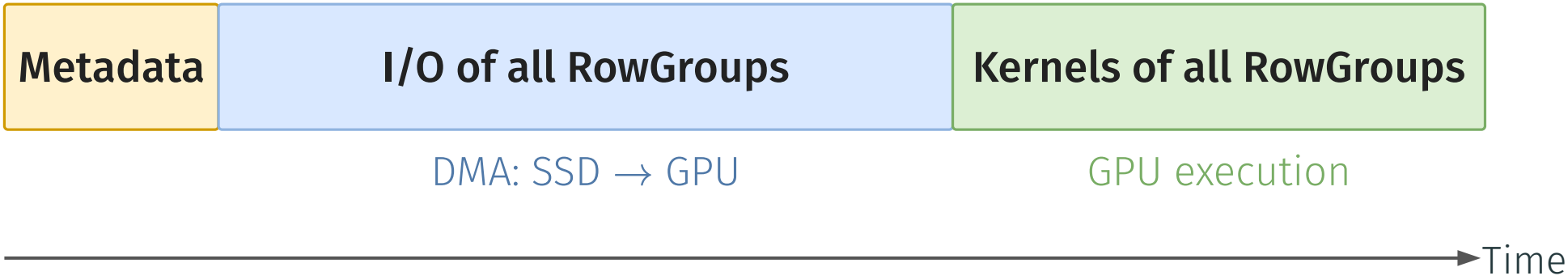

Distributed GPU Query Engine
PystachIO
Computation Layer
GPU
GPU
Storage I/O Layer
Fast Storage
NVMe SSDs
Fast Storage
Network I/O Layer
Fast Network
RDMA
Fast Network

PystachIO: Storage



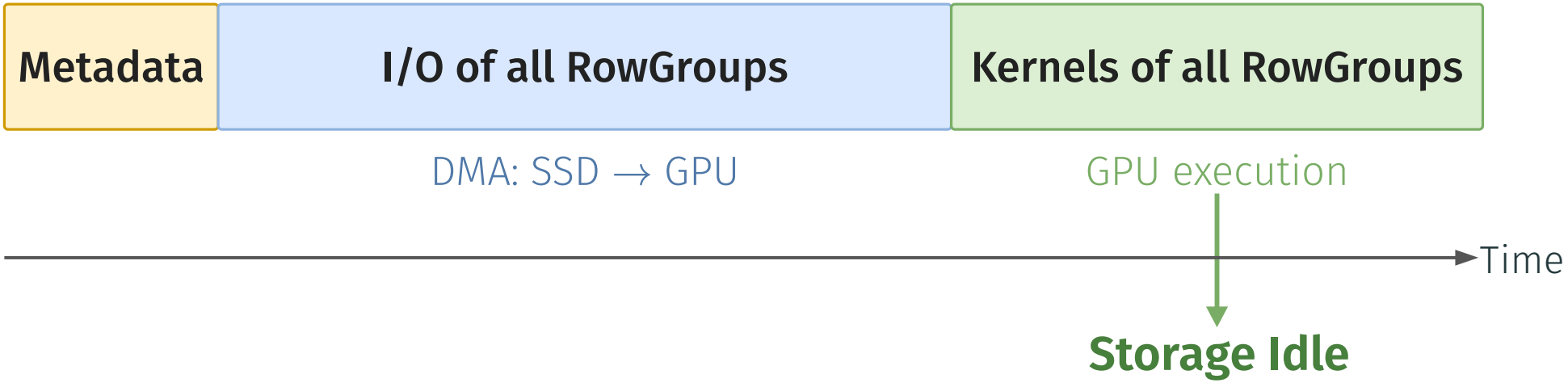
Inefficient Storage Access: Blocking Parquet Reader

`read_parquet()` in one CPU thread:



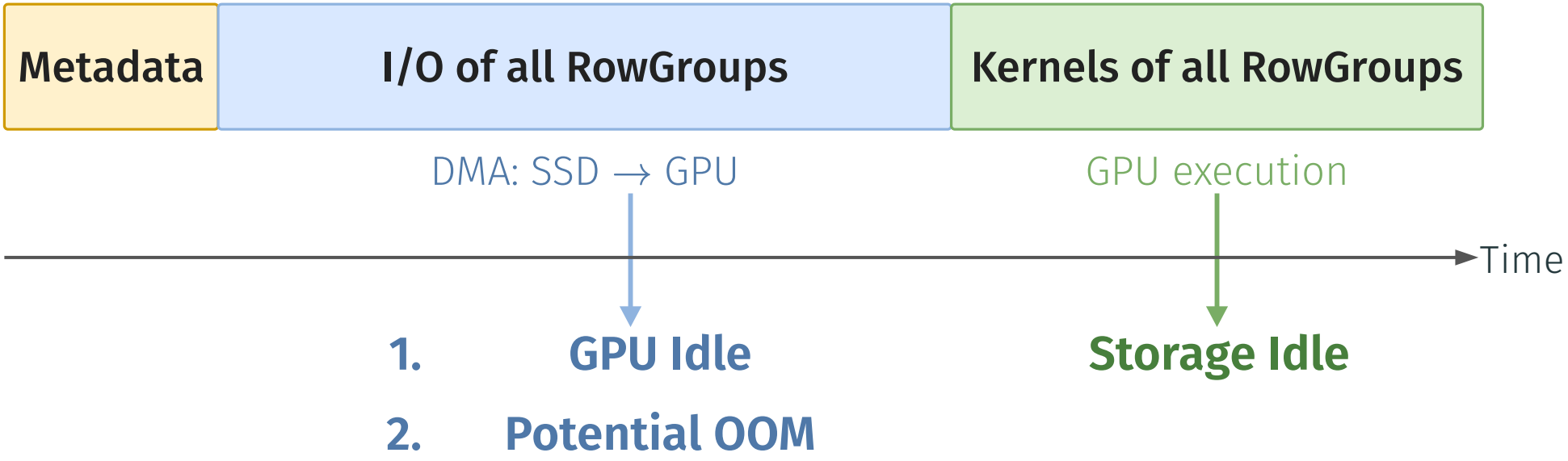
Inefficient Storage Access: Blocking Parquet Reader

`read_parquet()` in one CPU thread:

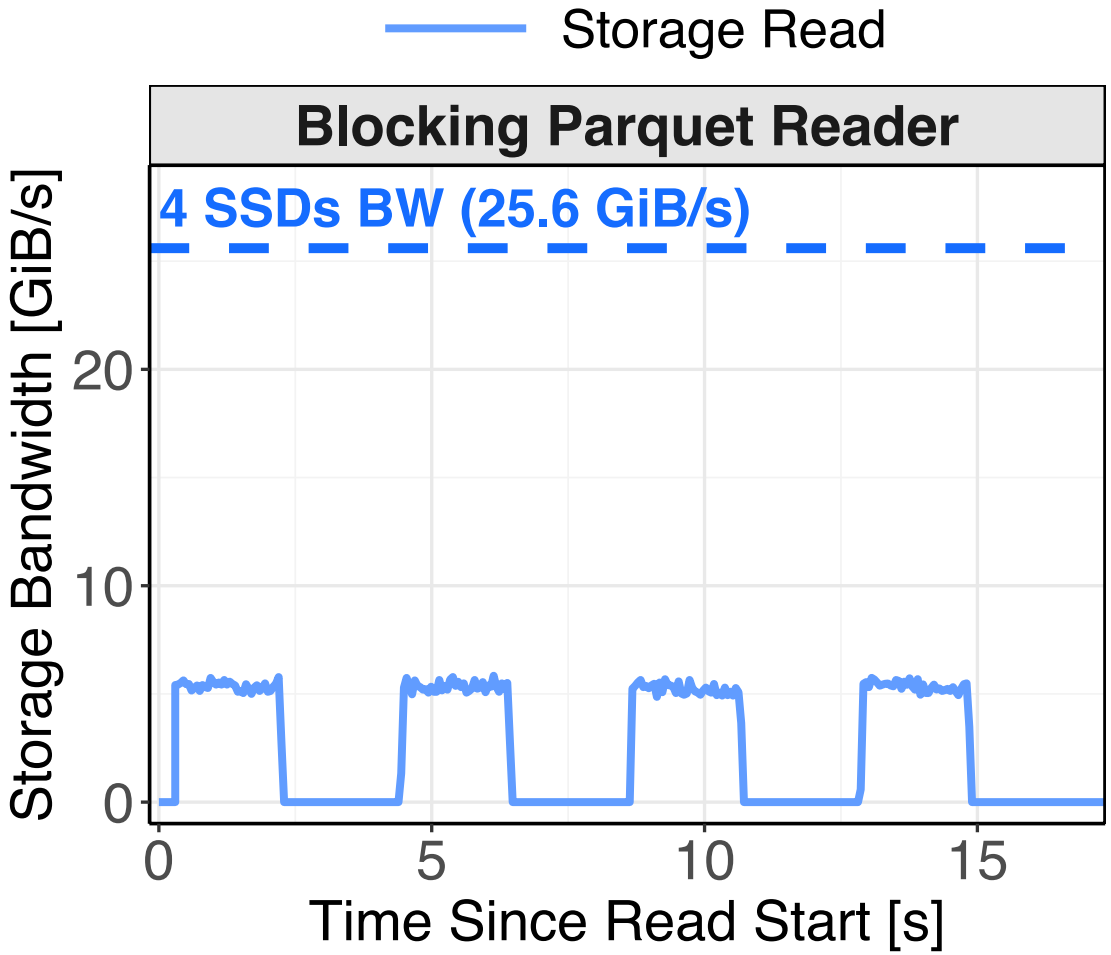


Inefficient Storage Access: Blocking Parquet Reader

`read_parquet()` in one CPU thread:

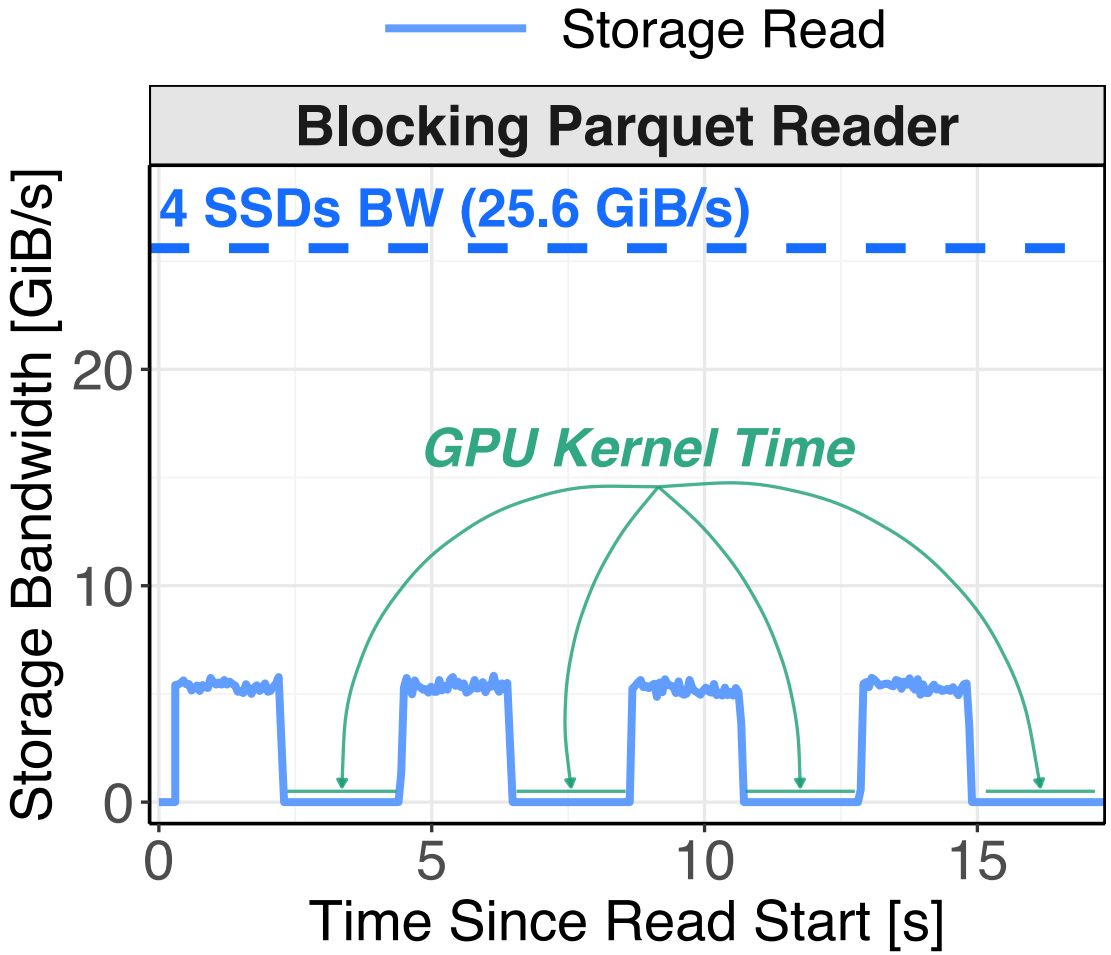


Inefficient Storage Access: Blocking Parquet Reader



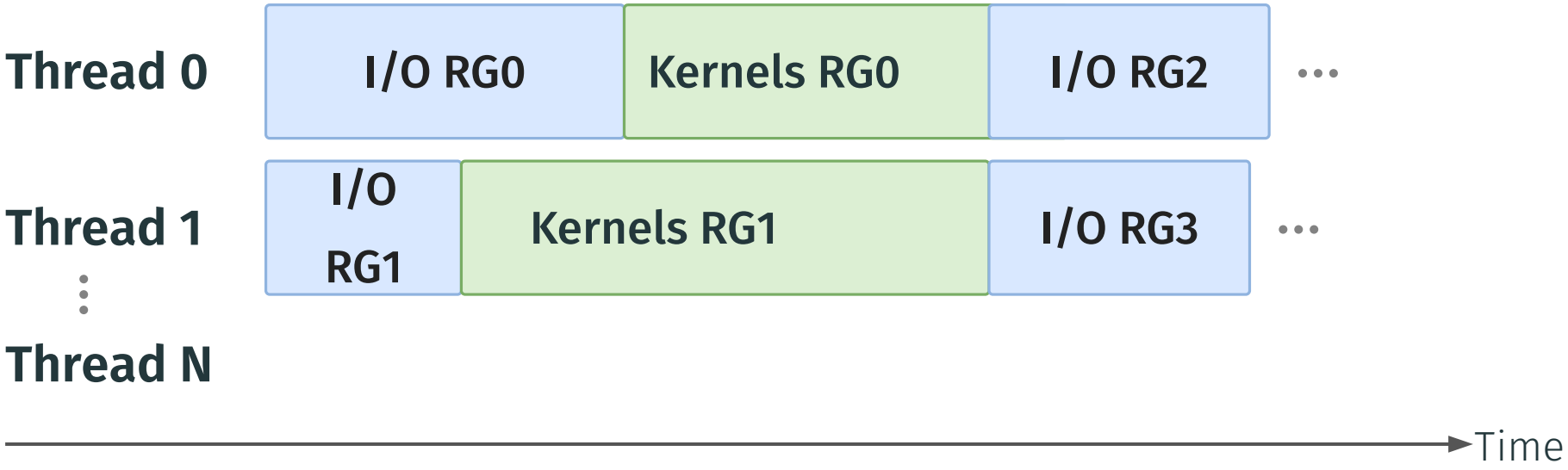
Storage microbenchmark with one GPU (A100, 40GB) and 4 NVMe SSD in PCIe 4.0.

Inefficient Storage Access: Blocking Parquet Reader

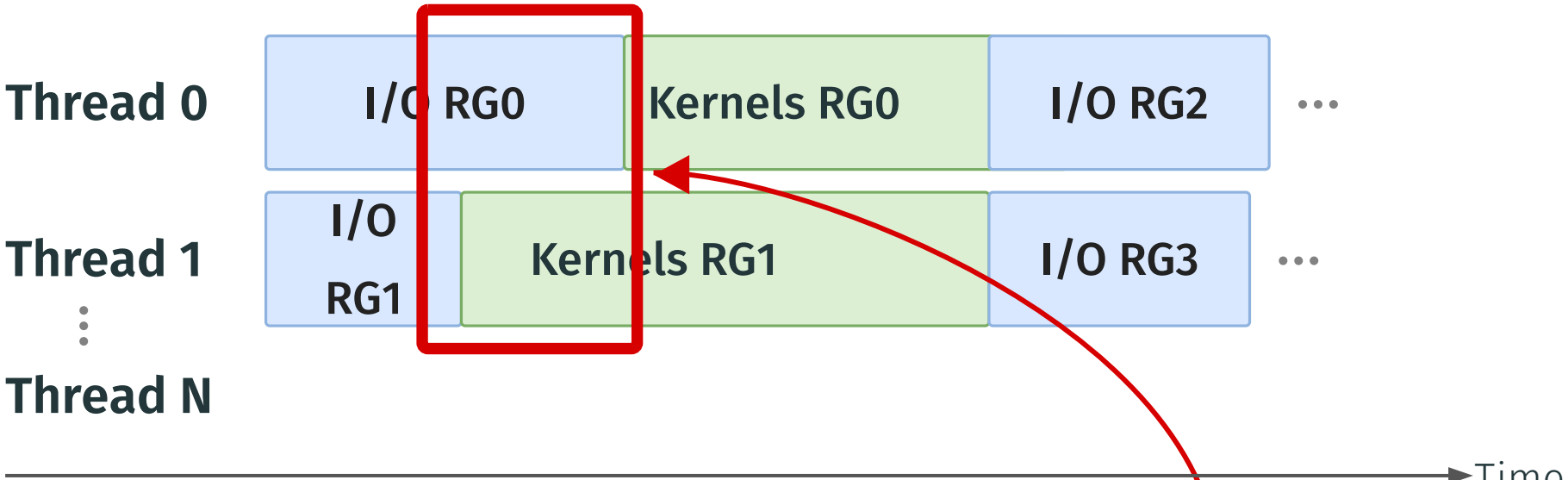


Storage microbenchmark with one GPU (A100, 40GB) and 4 NVMe SSD in PCIe 4.0.

Efficient Storage Access: Chunking

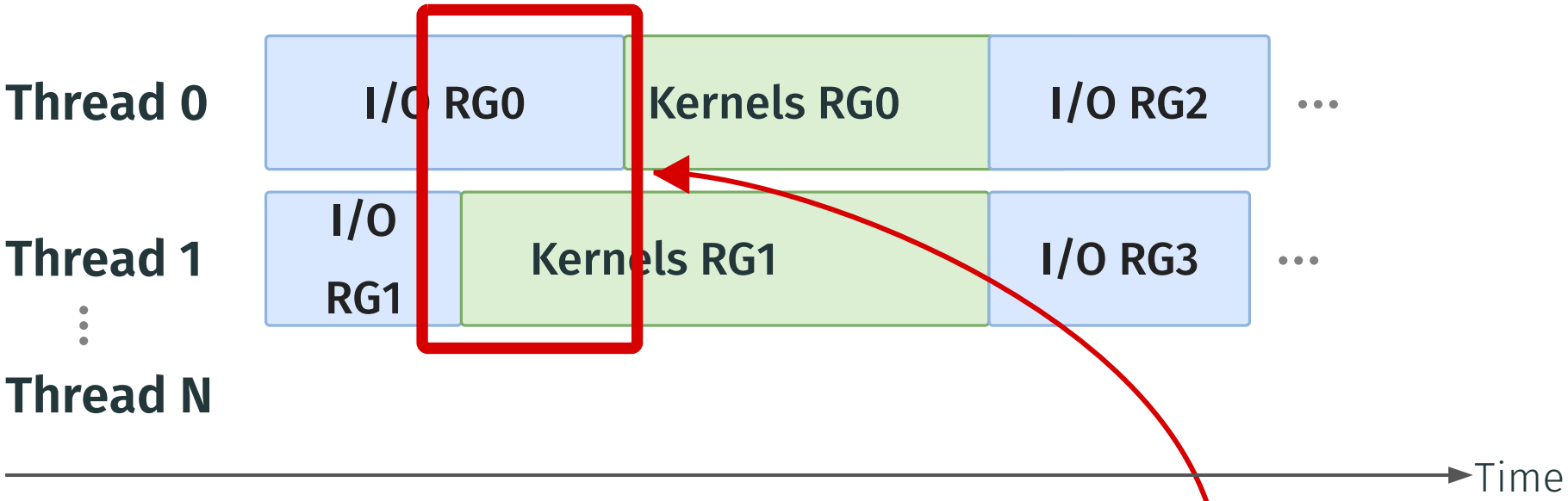


Efficient Storage Access: Chunking



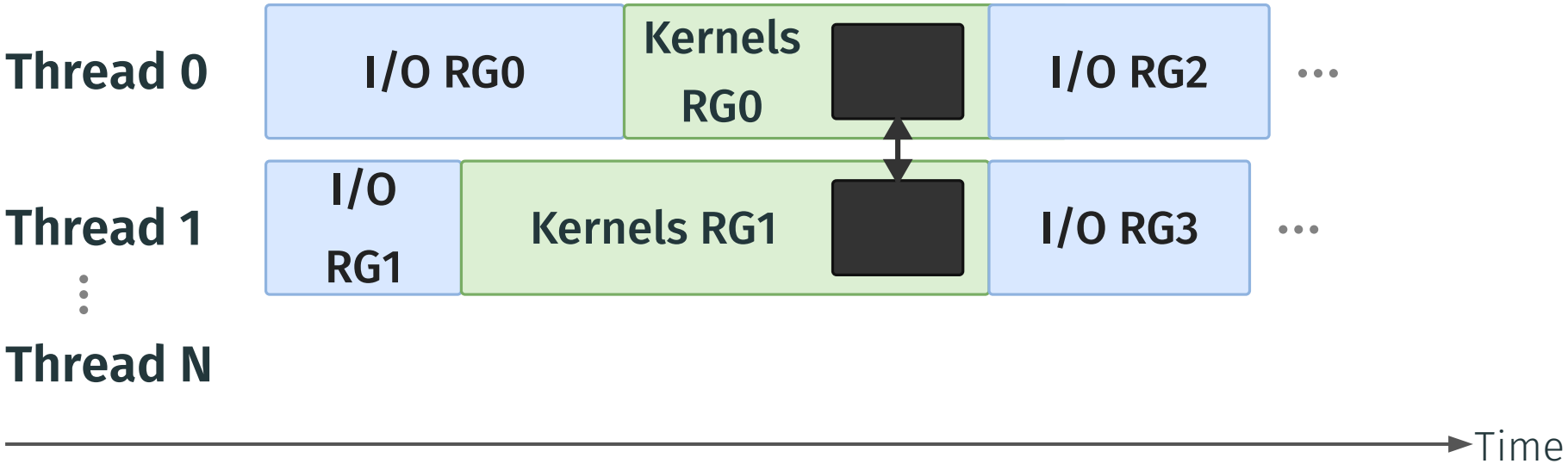
1. Overlapping I/O and Computation

Efficient Storage Access: Chunking

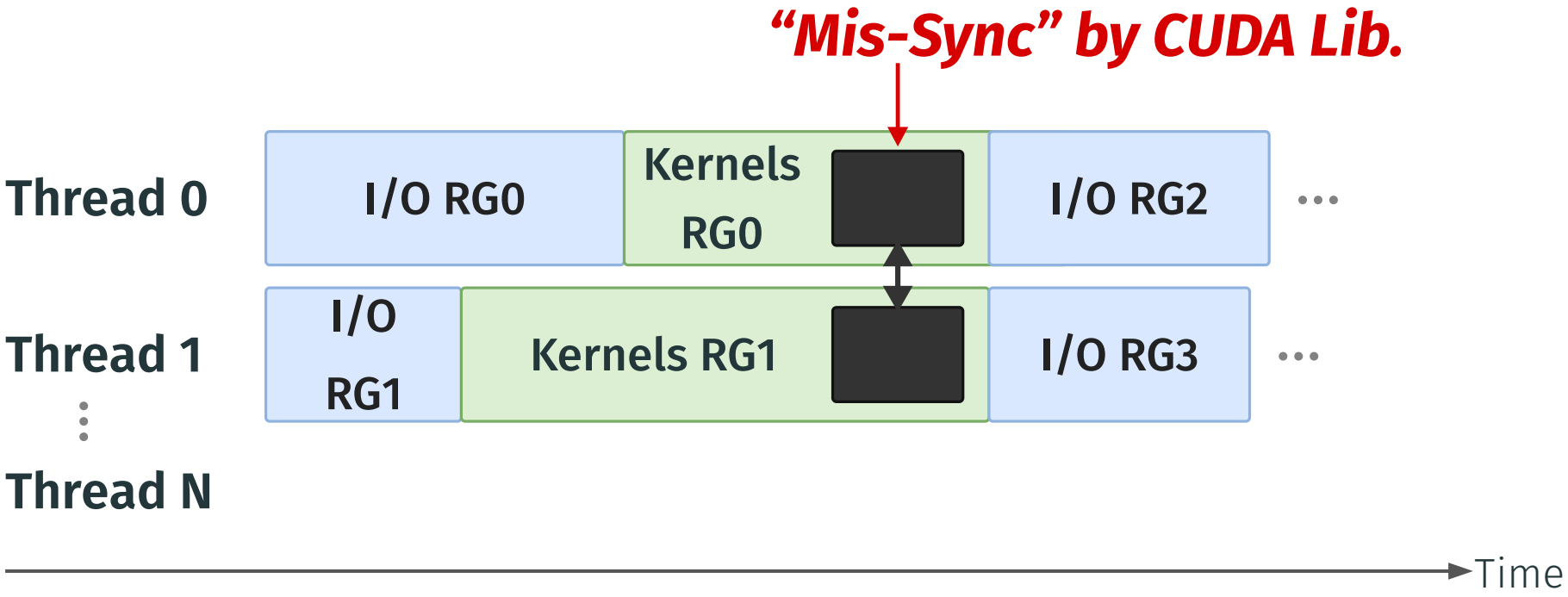


1. Overlapping I/O and Computation
2. Less Memory Pressure

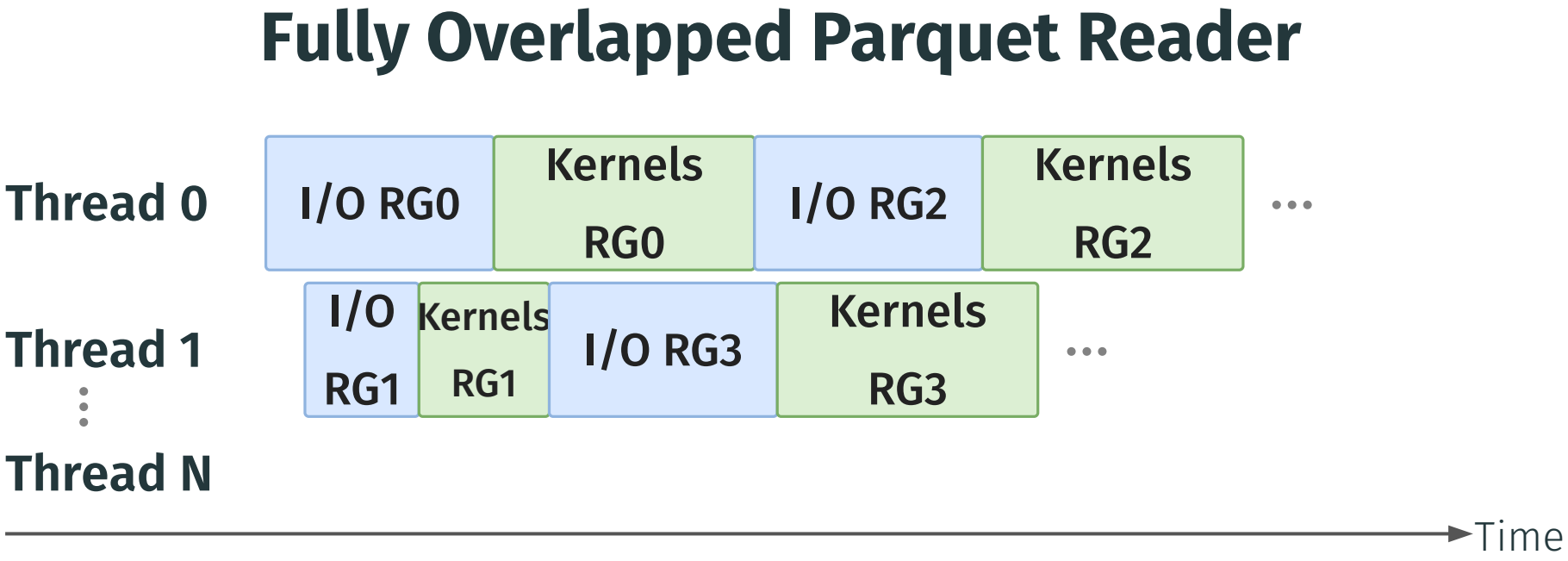
Efficient Storage Access: Mis-Sync Elimination



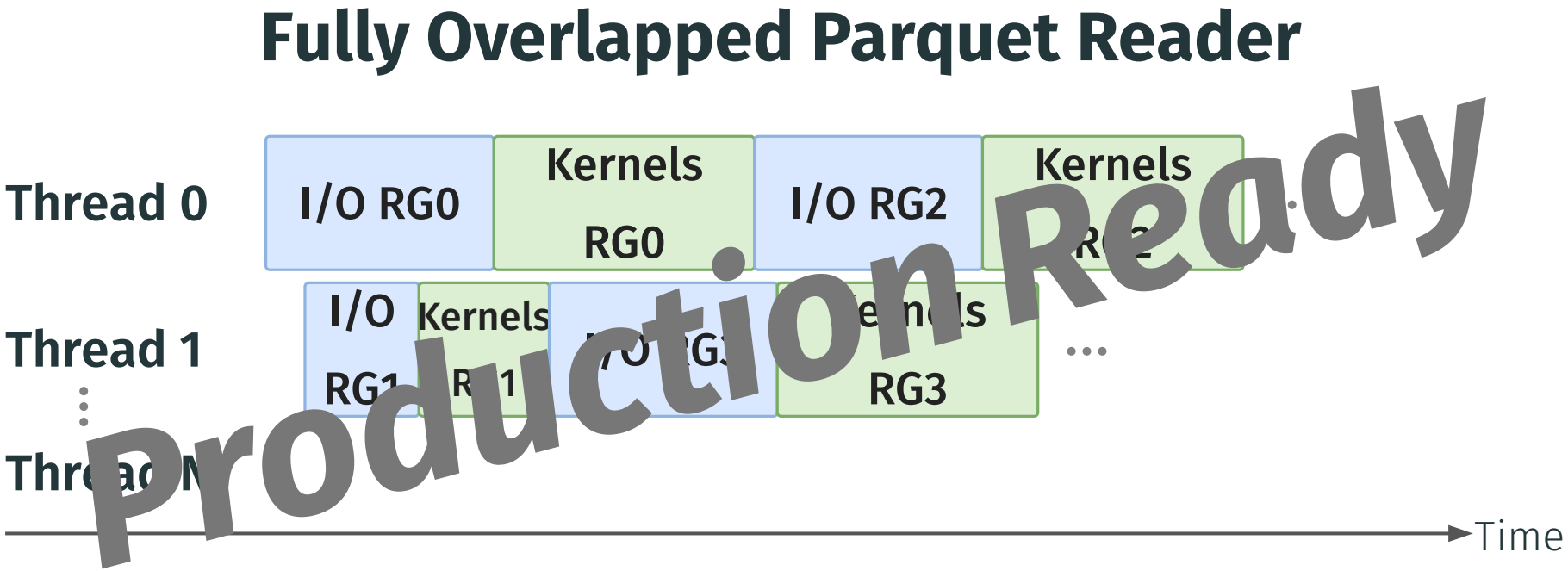
Efficient Storage Access: Mis-Sync Elimination



Efficient Storage Access: Mis-Sync Elimination

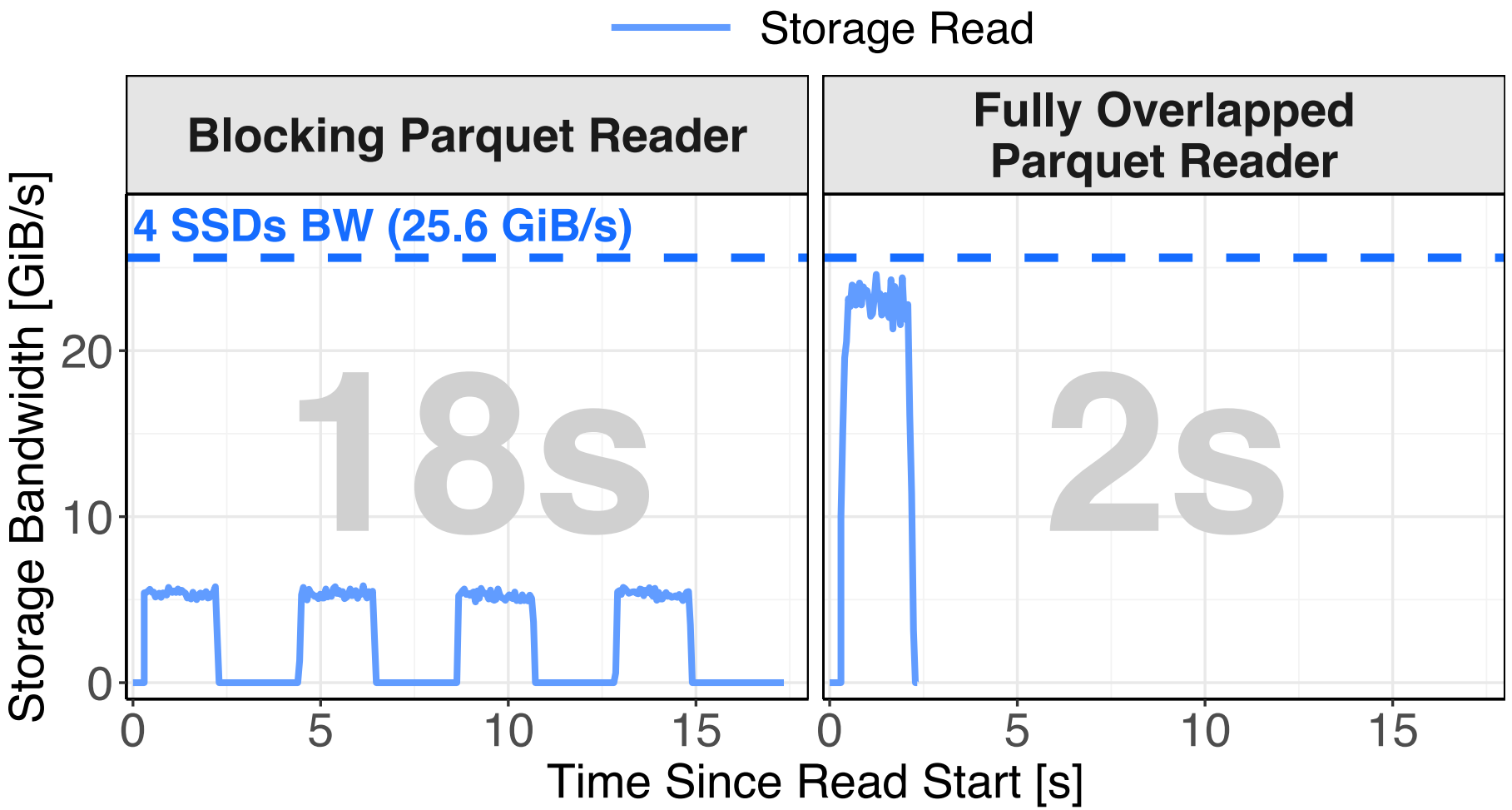


Efficient Storage Access: Mis-Sync Elimination



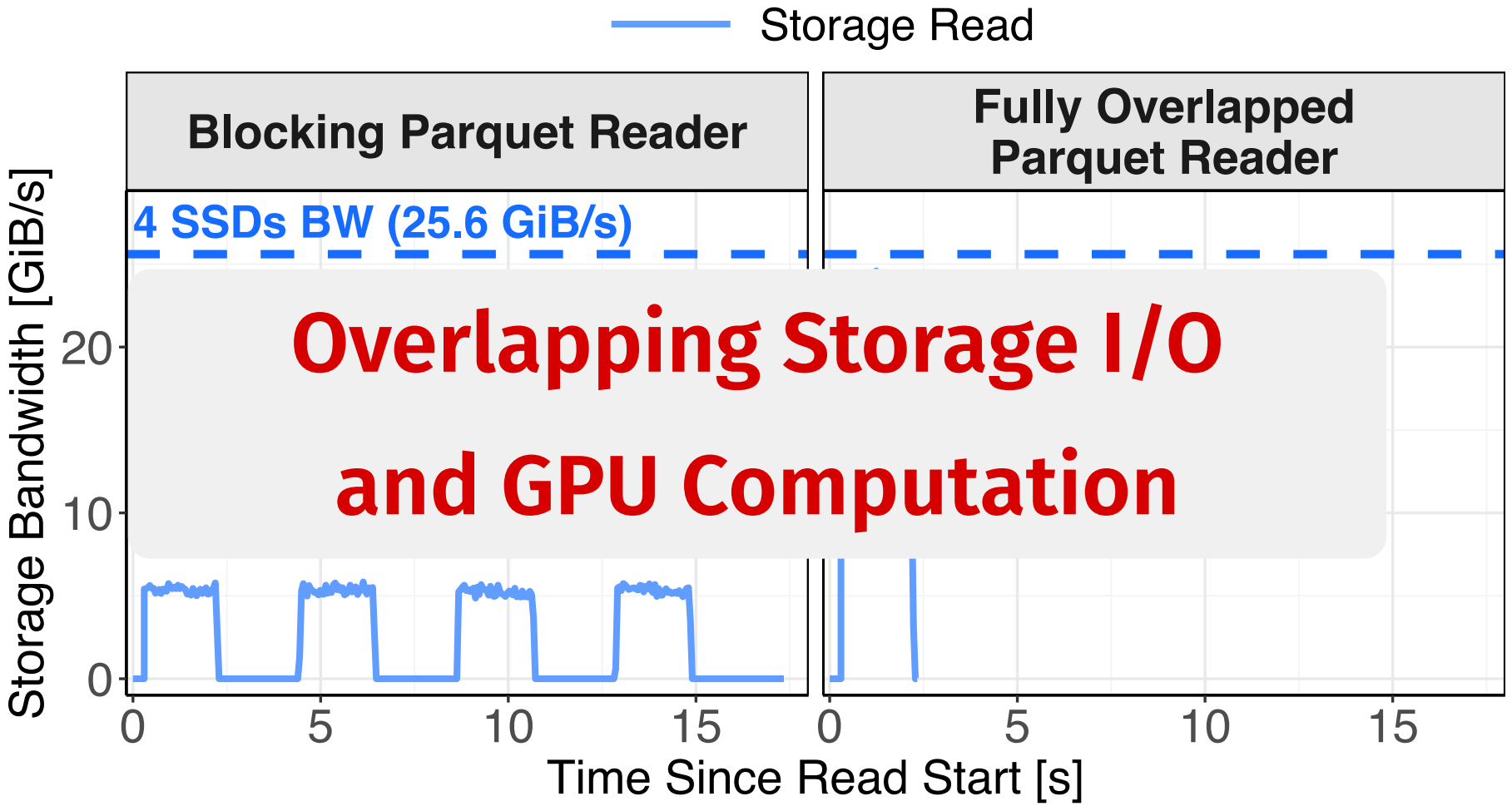
cuDF issue #18892: [Story] Towards a faster Parquet reader with pipelining and multistream optimization, Jigao Luo.
cuDF issue #21272: Migrate libcudf from Thrust to CUB-based algorithms, Bradley Dice.

Efficient Storage Access: Mis-Sync Elimination



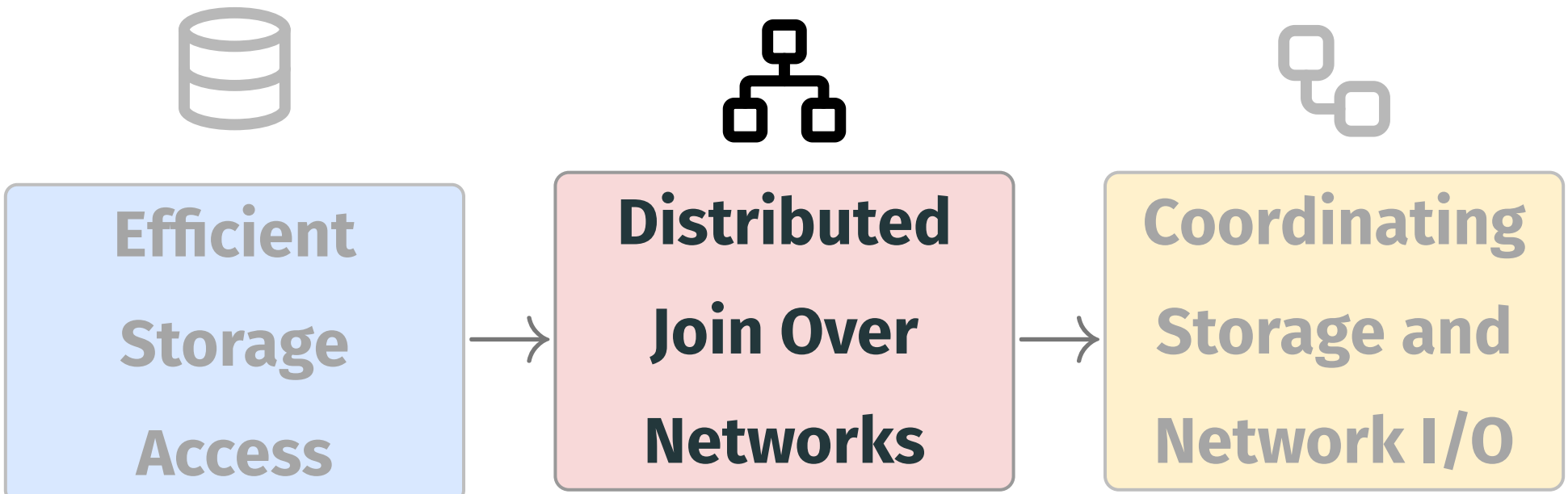
Storage microbenchmark with one GPU (A100, 40GB) and 4 NVMe SSD in PCIe 4.0.

Efficient Storage Access: Mis-Sync Elimination

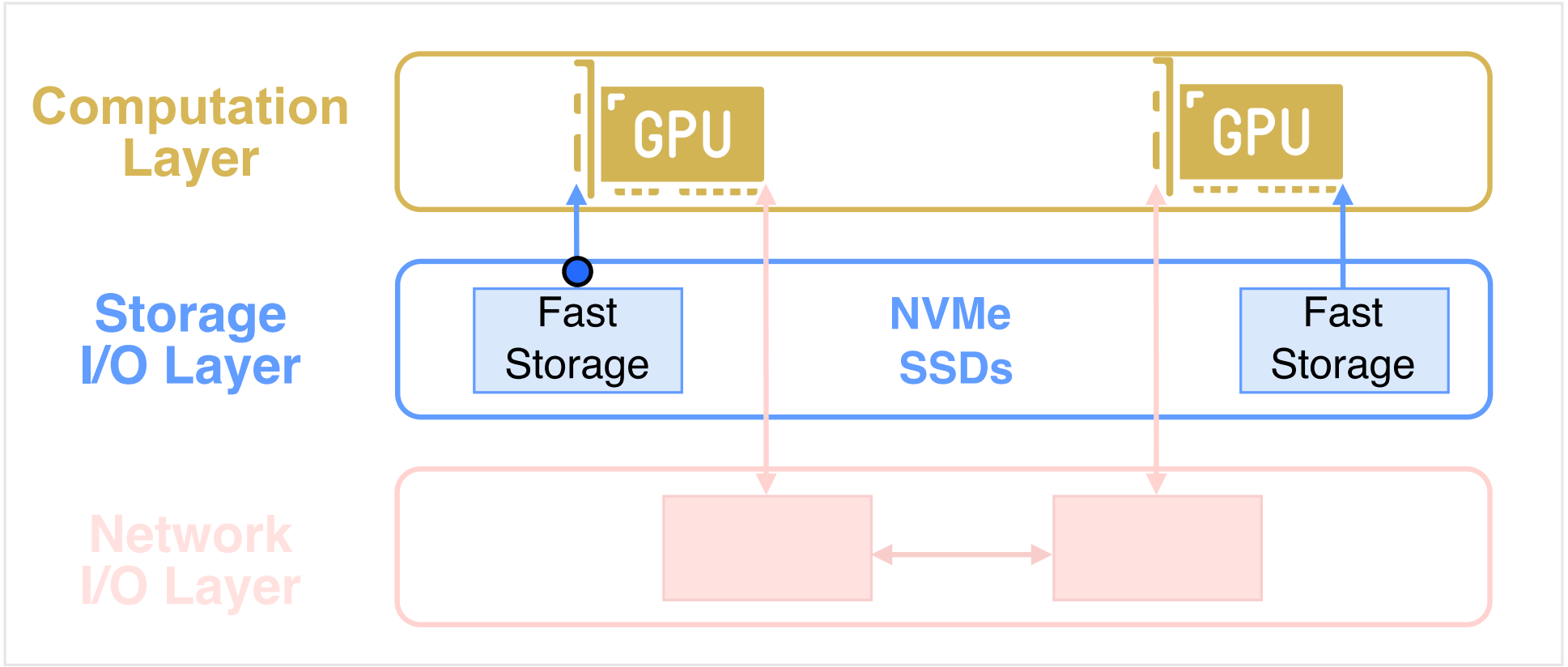


Storage microbenchmark with one GPU (A100, 40GB) and 4 NVMe SSD in PCIe 4.0.

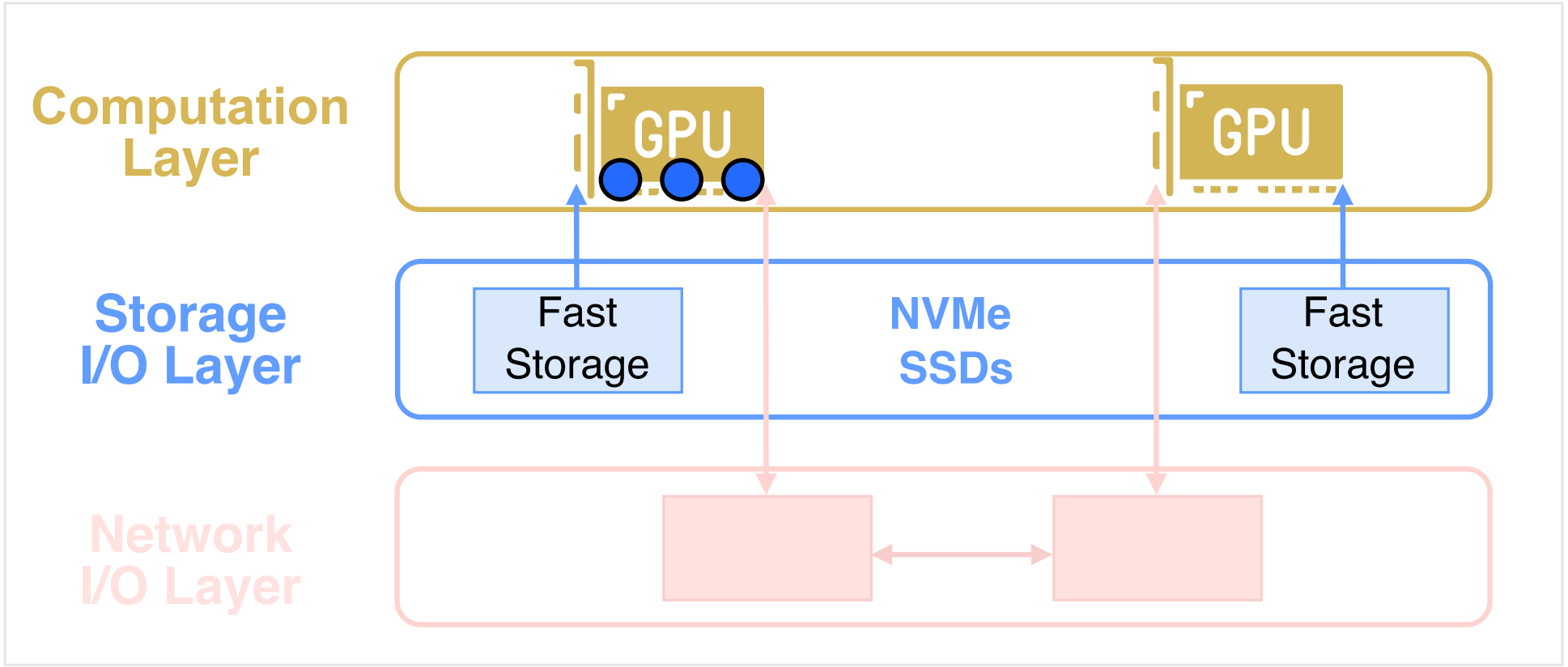
PystachIO: Network



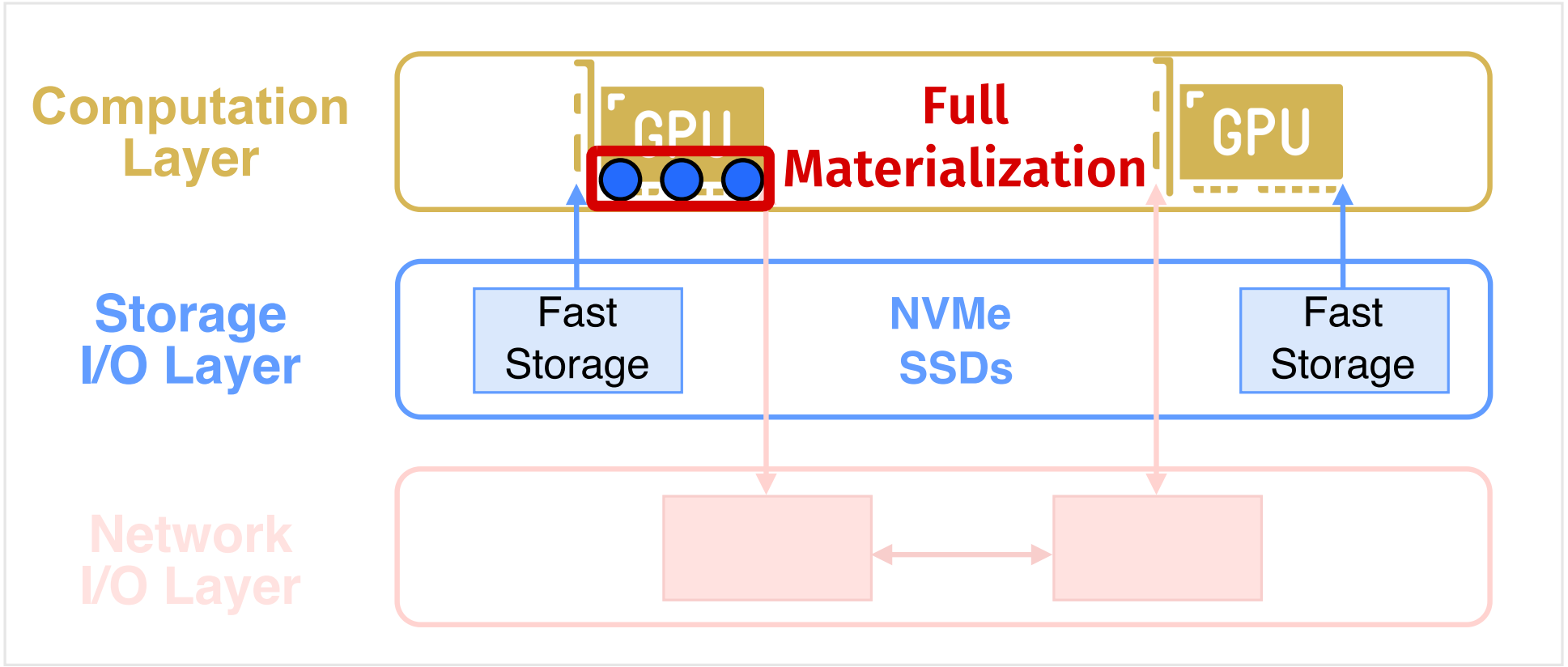
Run a Distributed Query



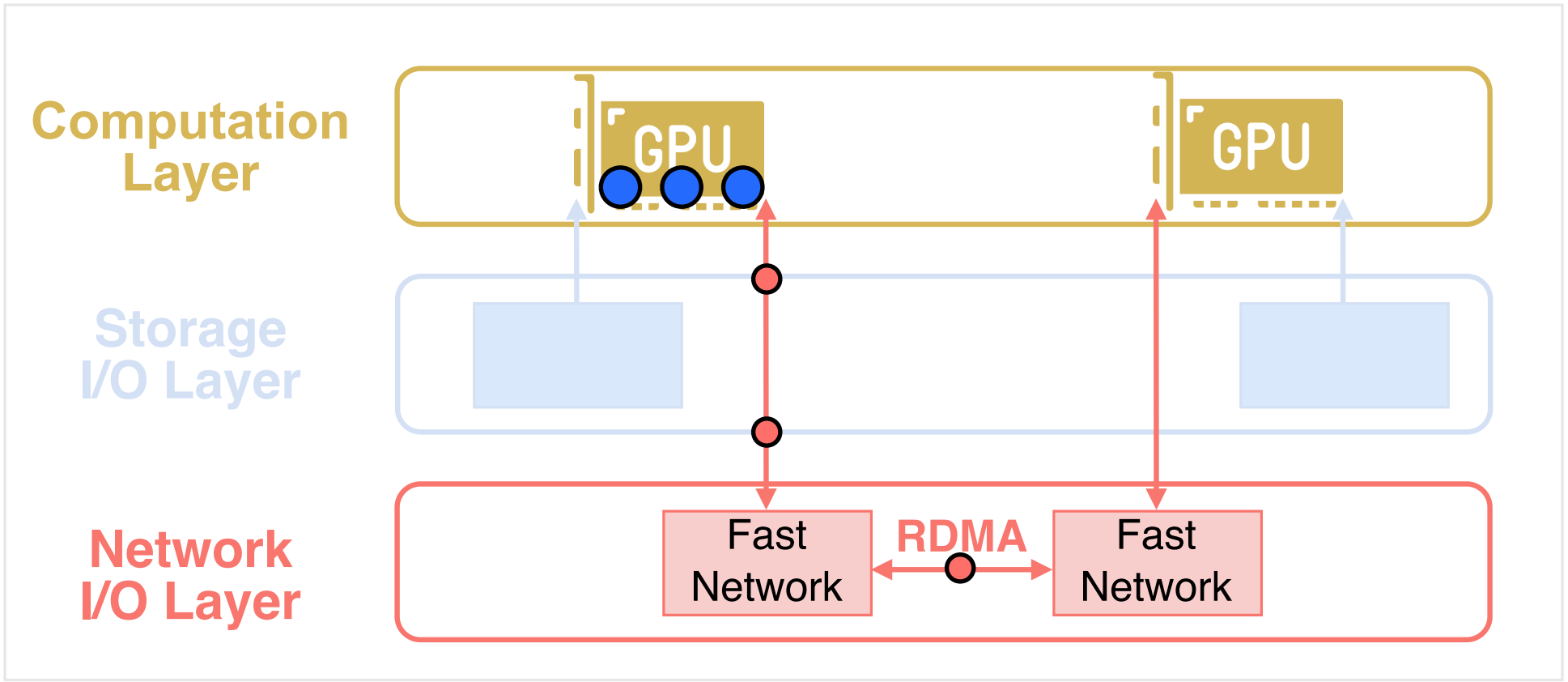
Run a Distributed Query



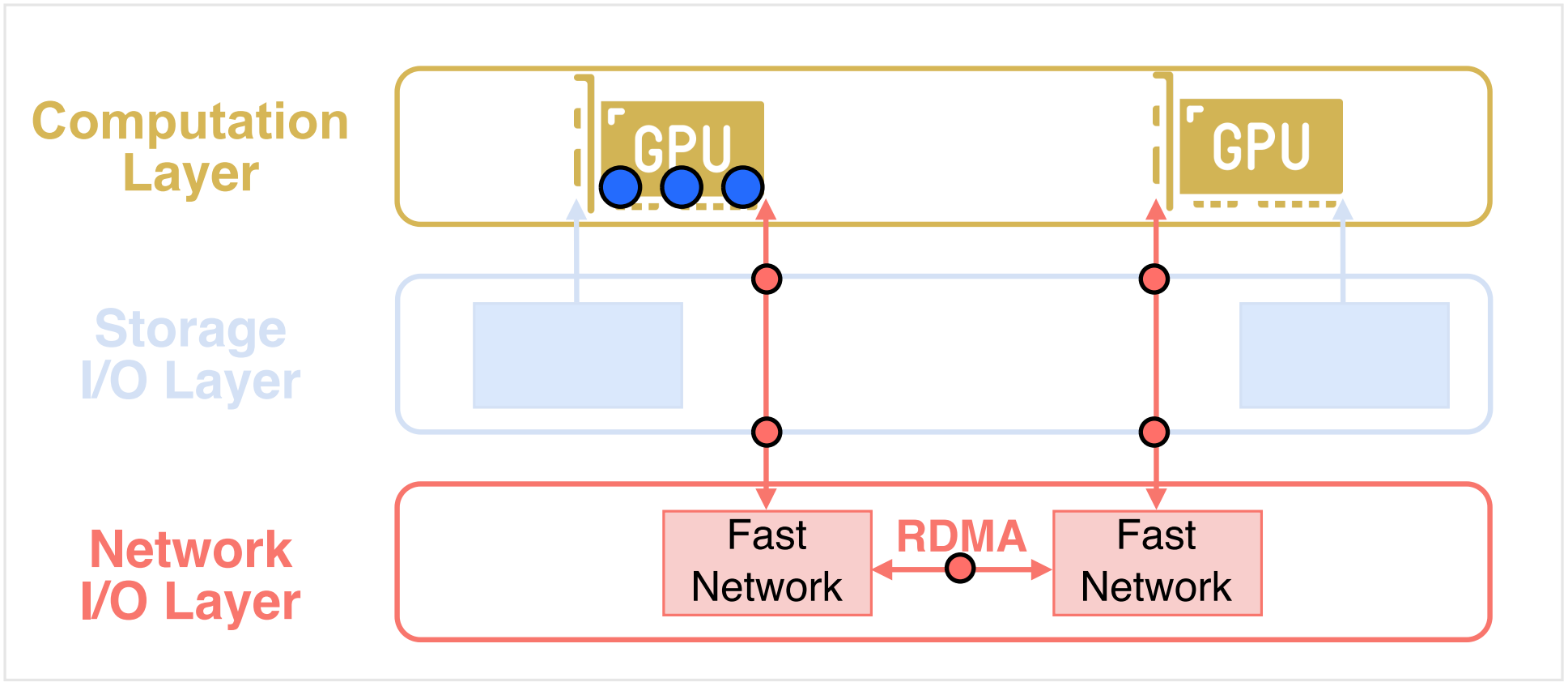
Run a Distributed Query



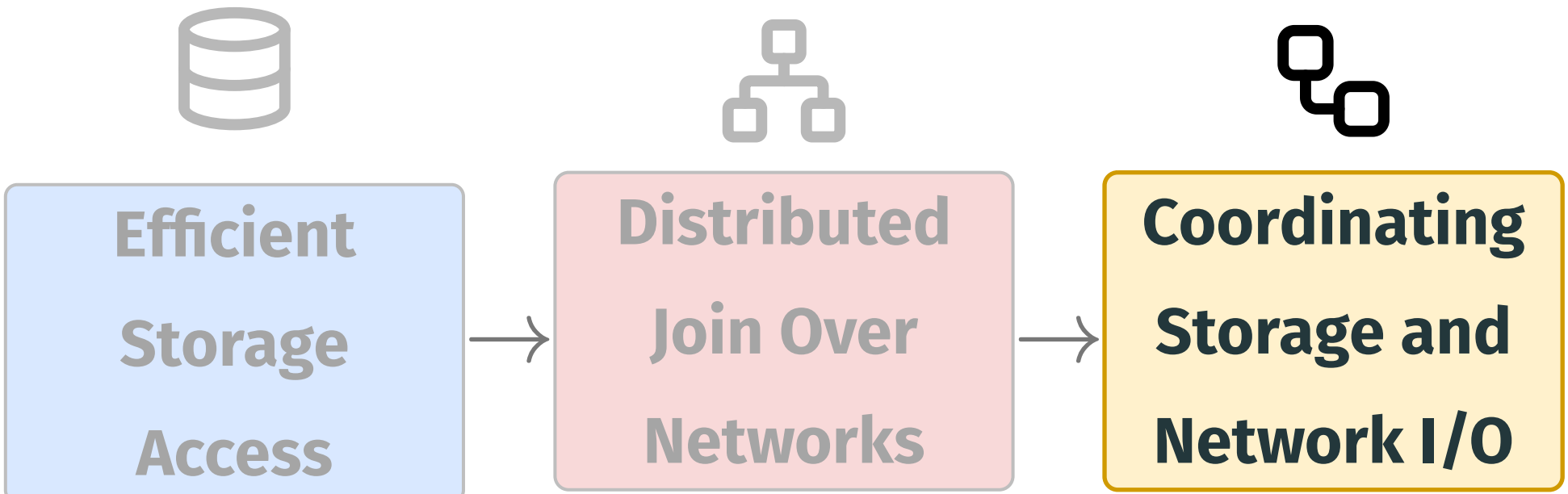
Run a Distributed Query



Run a Distributed Query

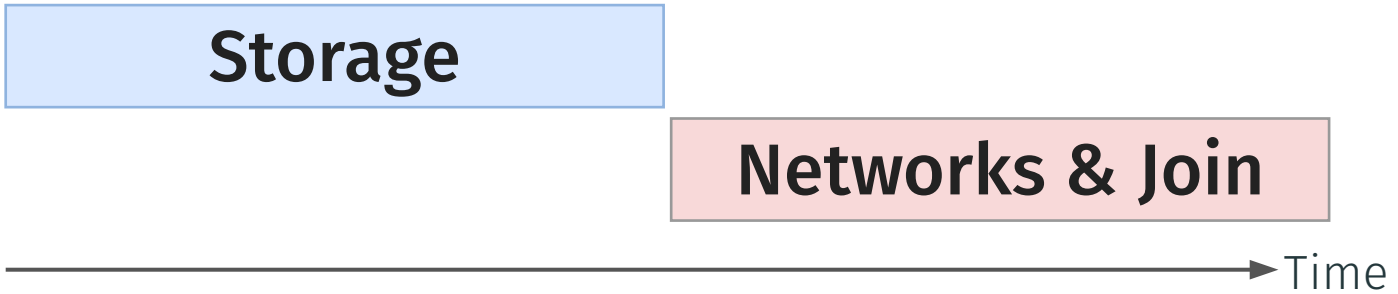


PystachIO: Coordinating I/O



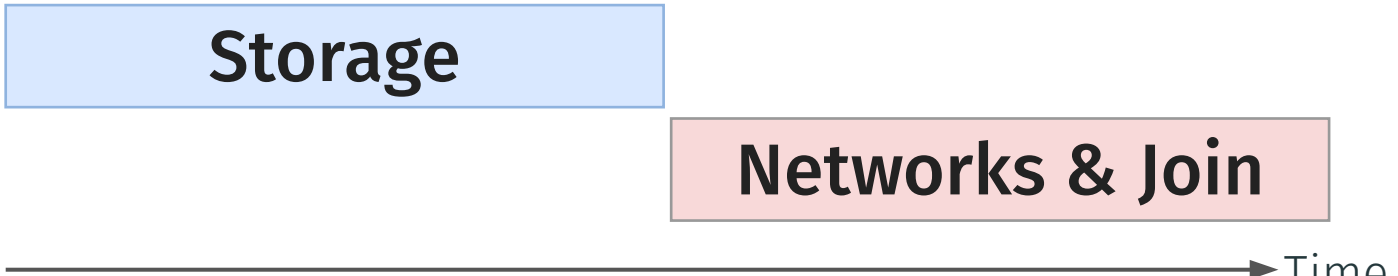
Fast Storage, then Fast Network

- Storage optimized
- Network optimized



Fast Storage, then Fast Network

- Storage optimized
- Network optimized



– Slow Runtime

– High Risk of OOM

– Only One I/O Device Active

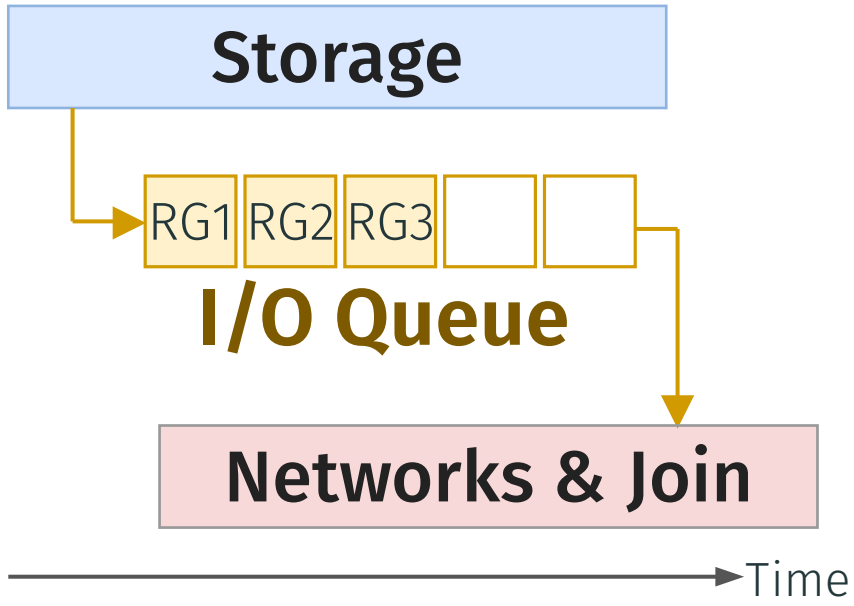
Fast Storage, then Fast Network

- Storage
- Networ

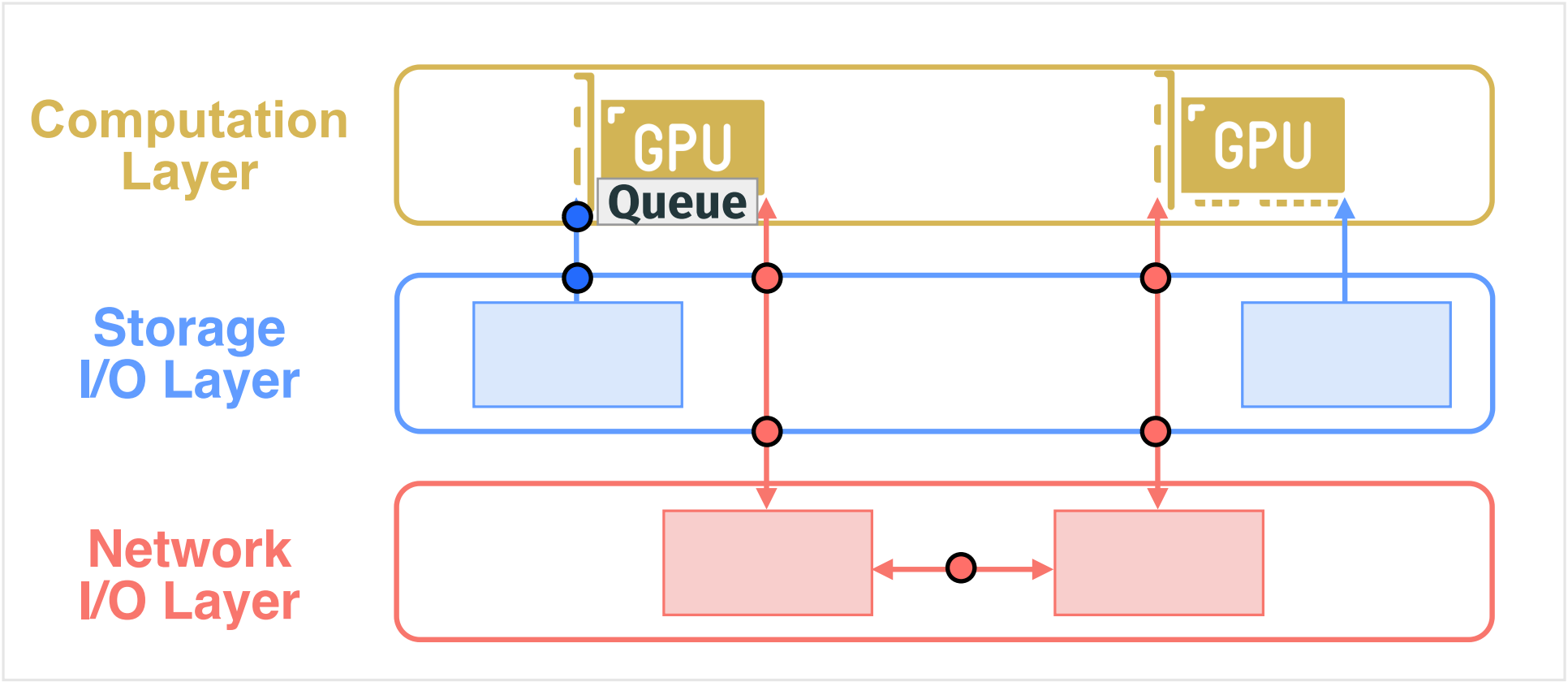
Is It Possible to Overlap Storage and Network?

I/O Queue For Coordination

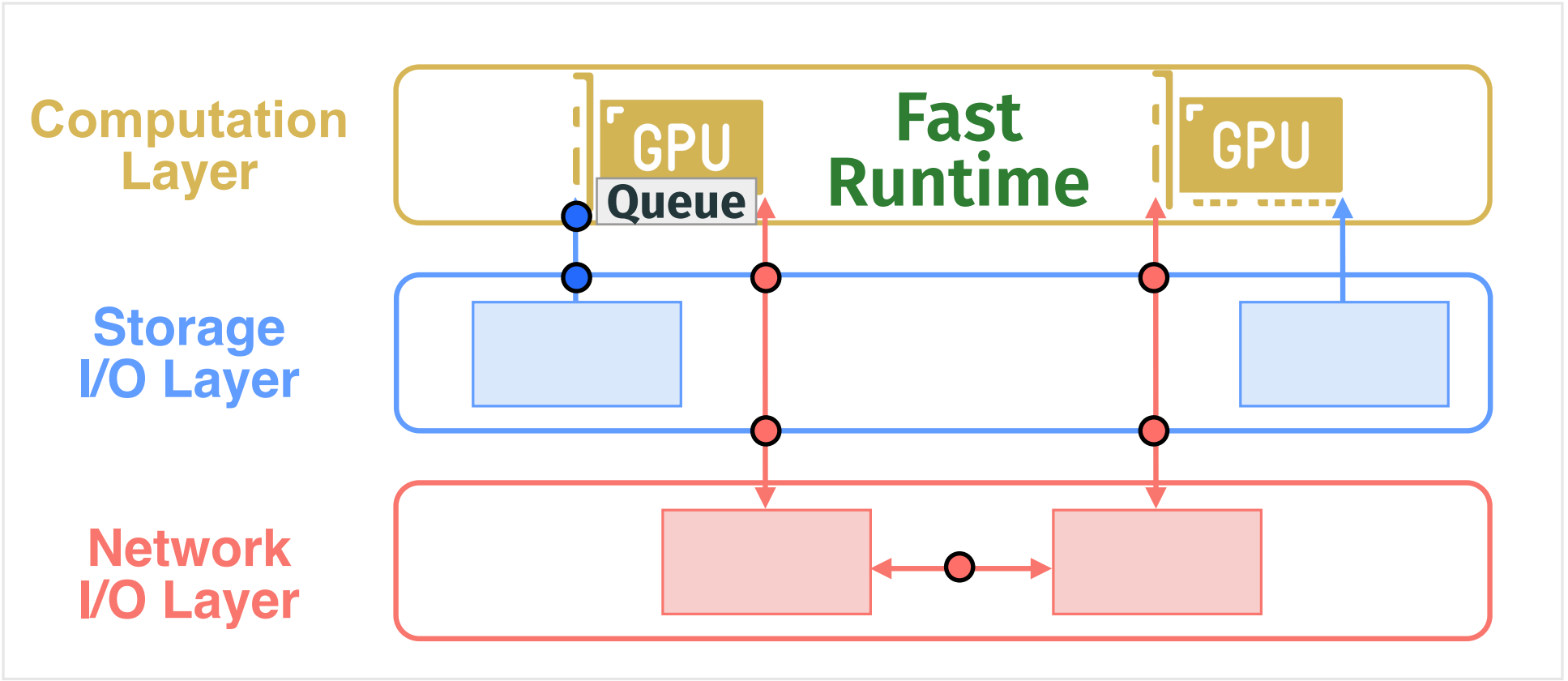
- Storage: enqueuing
- Network & Join: dequeuing
- Then network shuffling



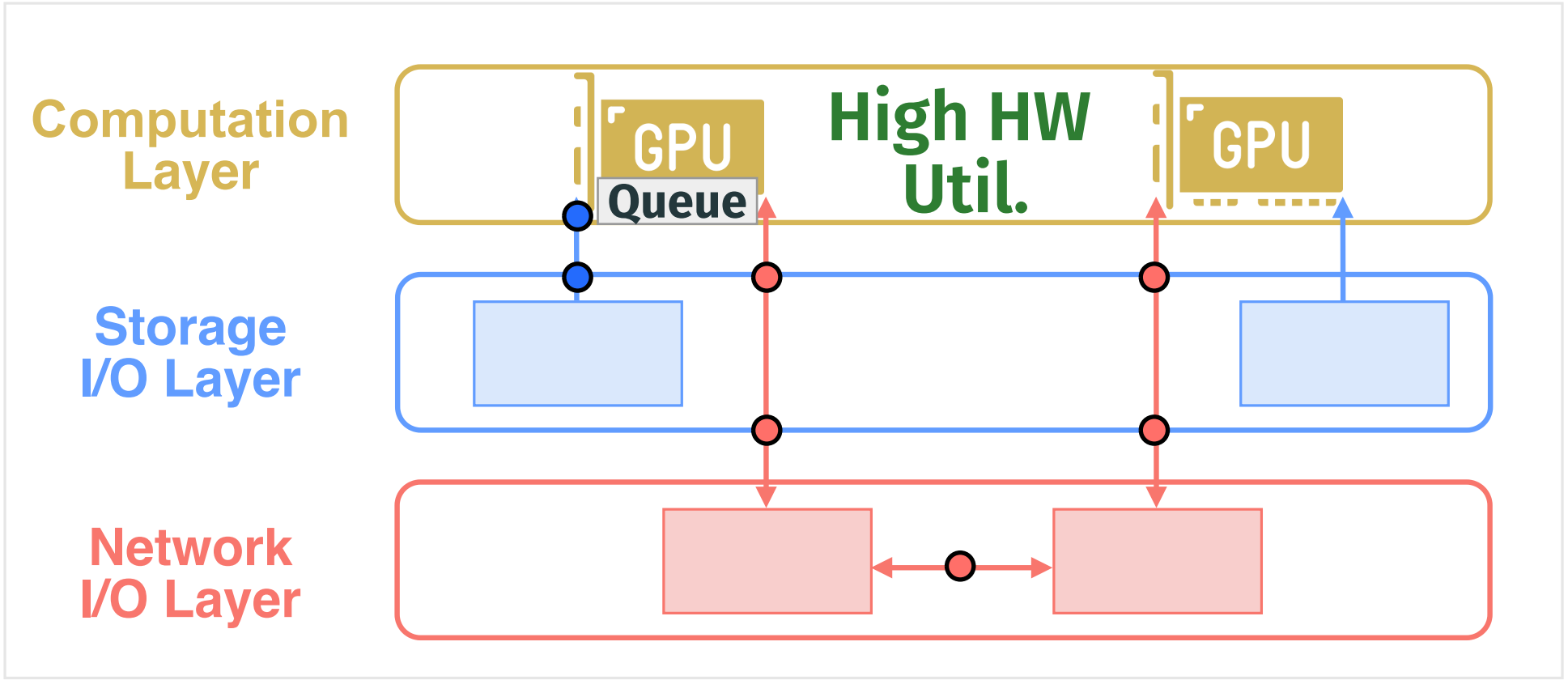
Overlapping Network, Storage, and Computation



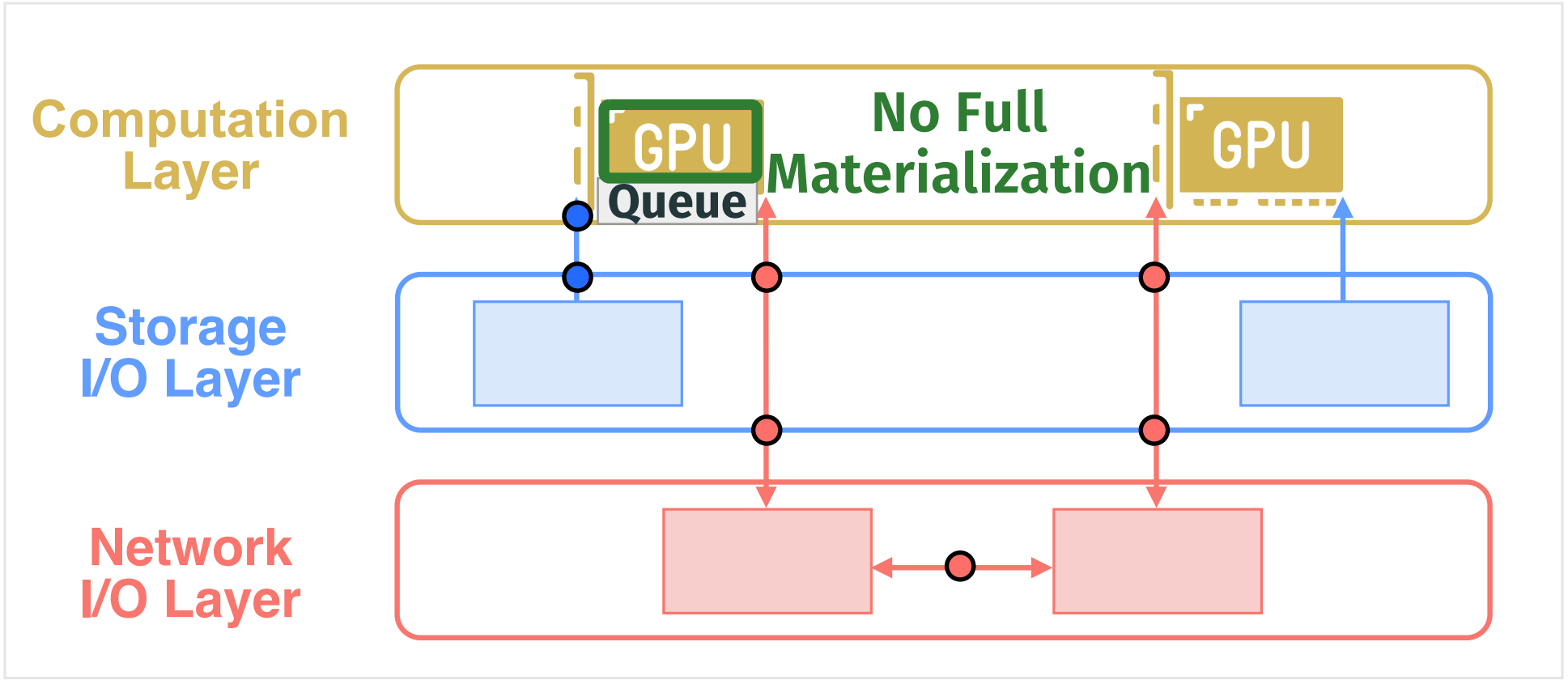
Overlapping Network, Storage, and Computation



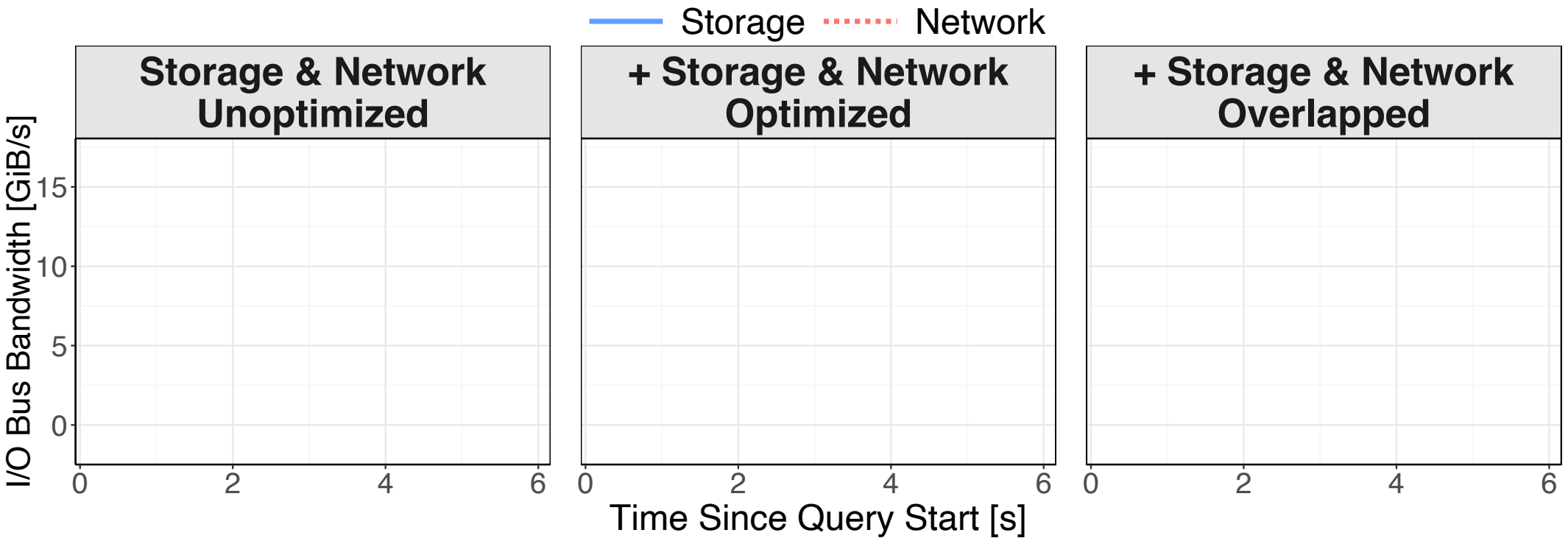
Overlapping Network, Storage, and Computation



Overlapping Network, Storage, and Computation

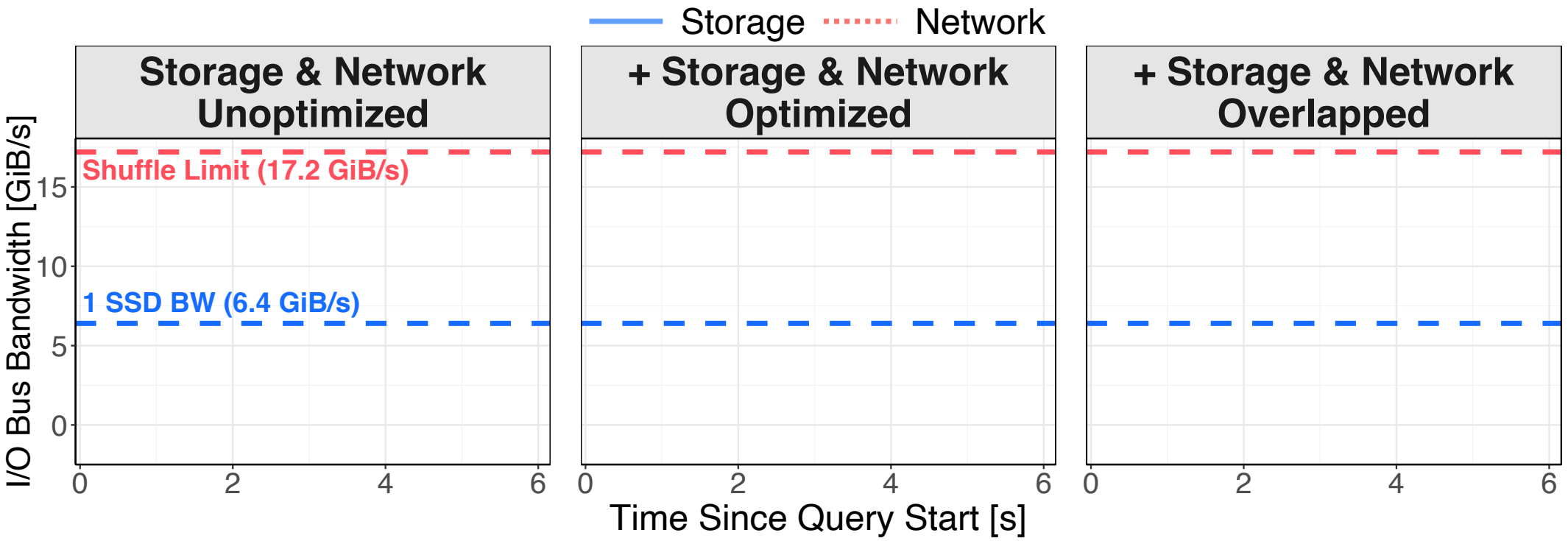


Overlapping Network, Storage, and Computation



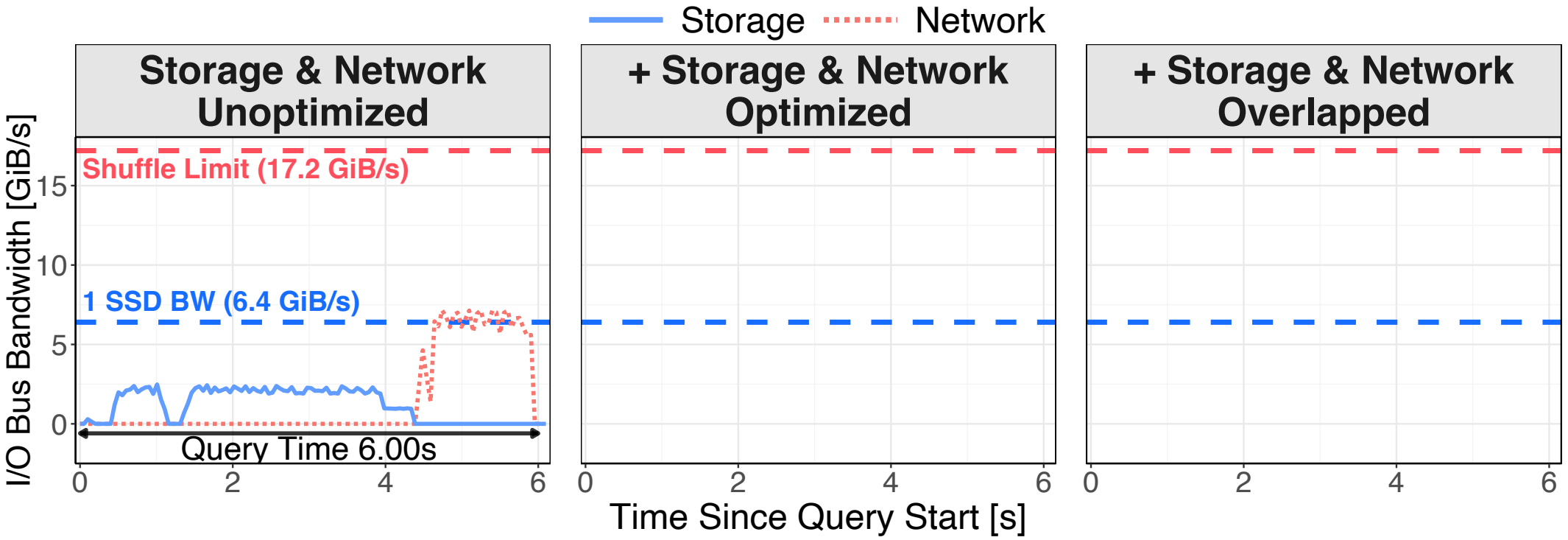
TPC-H Query 3 (SF 500) runtime executed on 2 GPUs (Each GPU has one SSD and one NIC)

Overlapping Network, Storage, and Computation



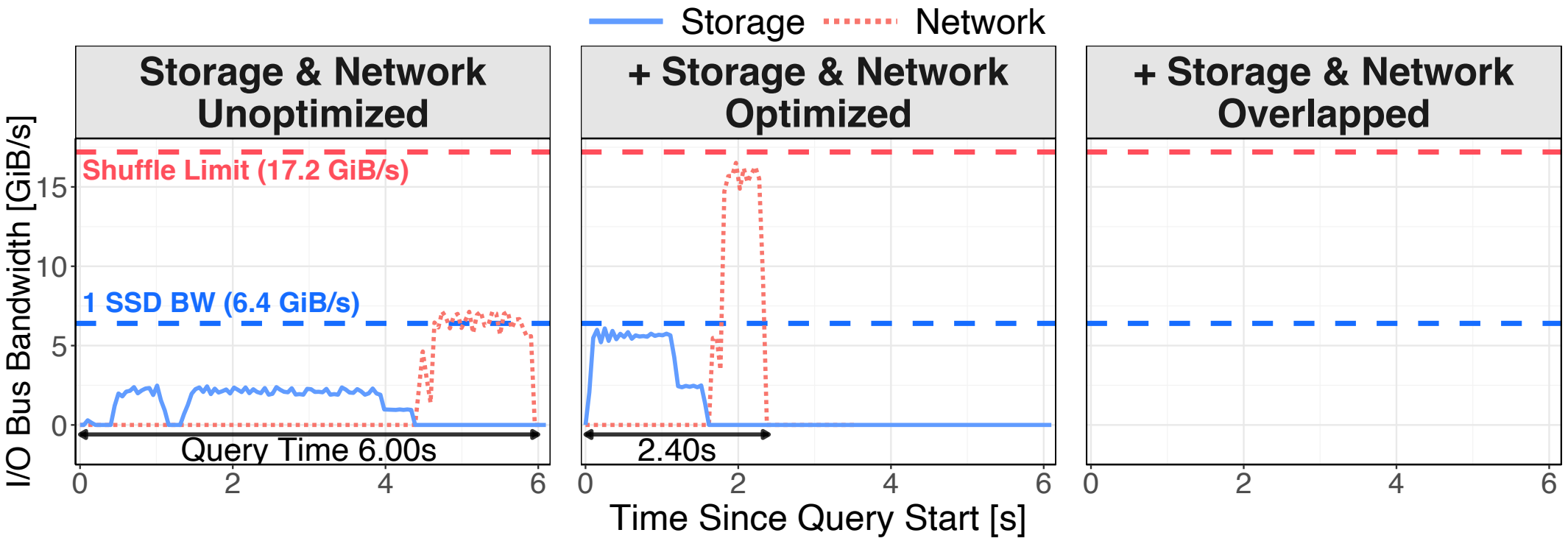
TPC-H Query 3 (SF 500) runtime executed on 2 GPUs (Each GPU has one SSD and one NIC)

Overlapping Network, Storage, and Computation



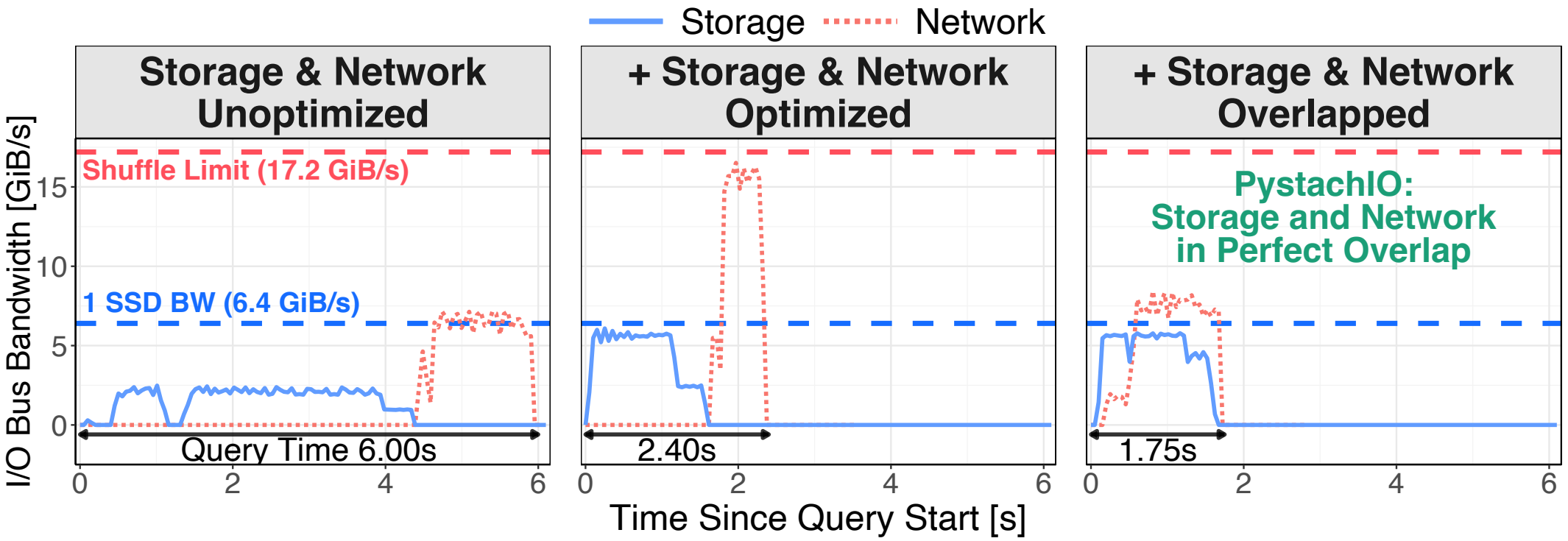
TPC-H Query 3 (SF 500) runtime executed on 2 GPUs (Each GPU has one SSD and one NIC)

Overlapping Network, Storage, and Computation



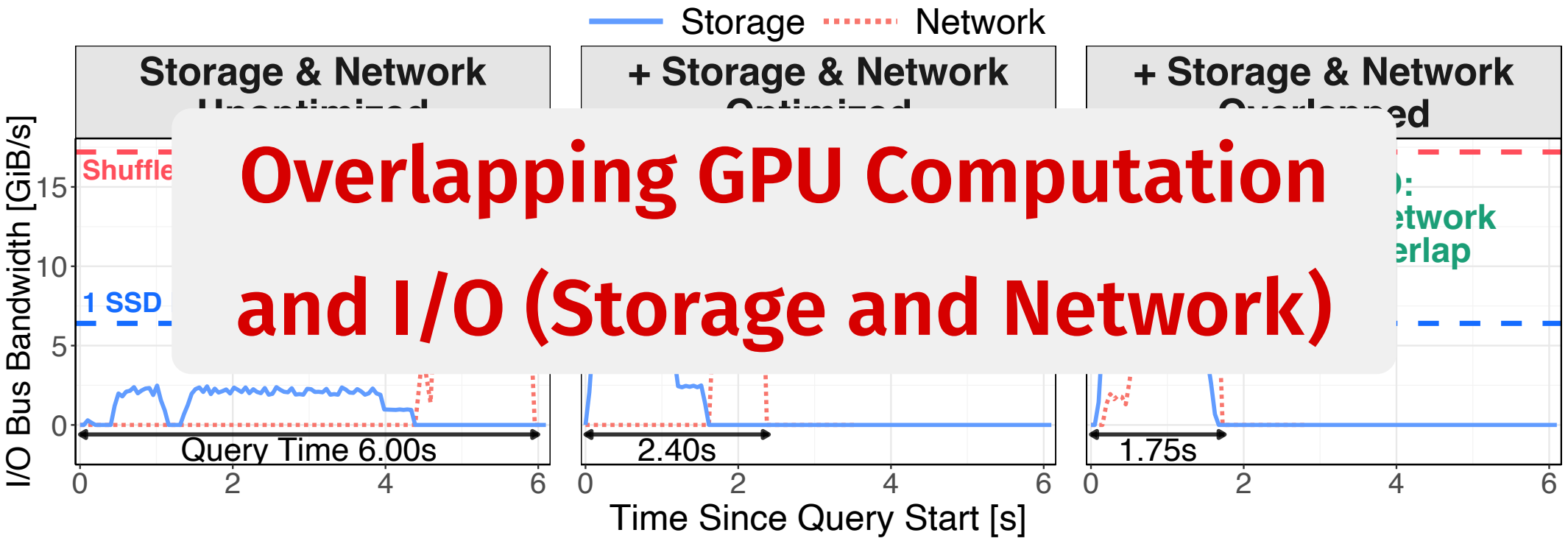
TPC-H Query 3 (SF 500) runtime executed on 2 GPUs (Each GPU has one SSD and one NIC)

Overlapping Network, Storage, and Computation



TPC-H Query 3 (SF 500) runtime executed on 2 GPUs (Each GPU has one SSD and one NIC)

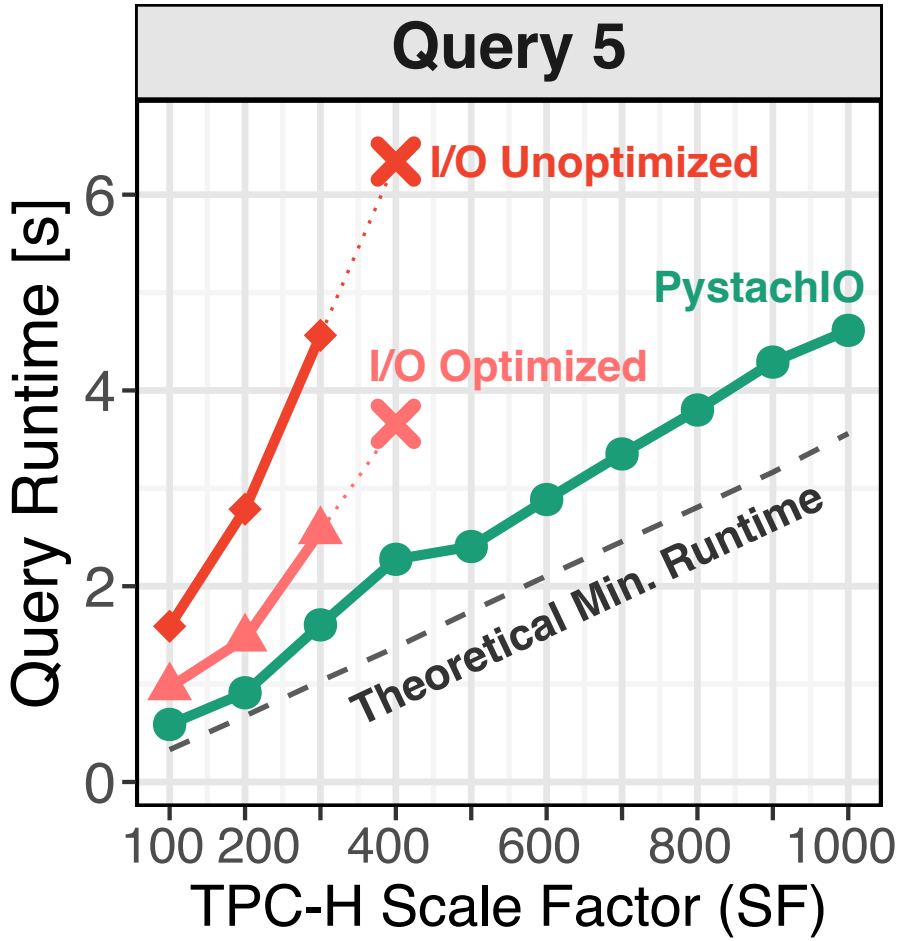
Overlapping Network, Storage, and Computation



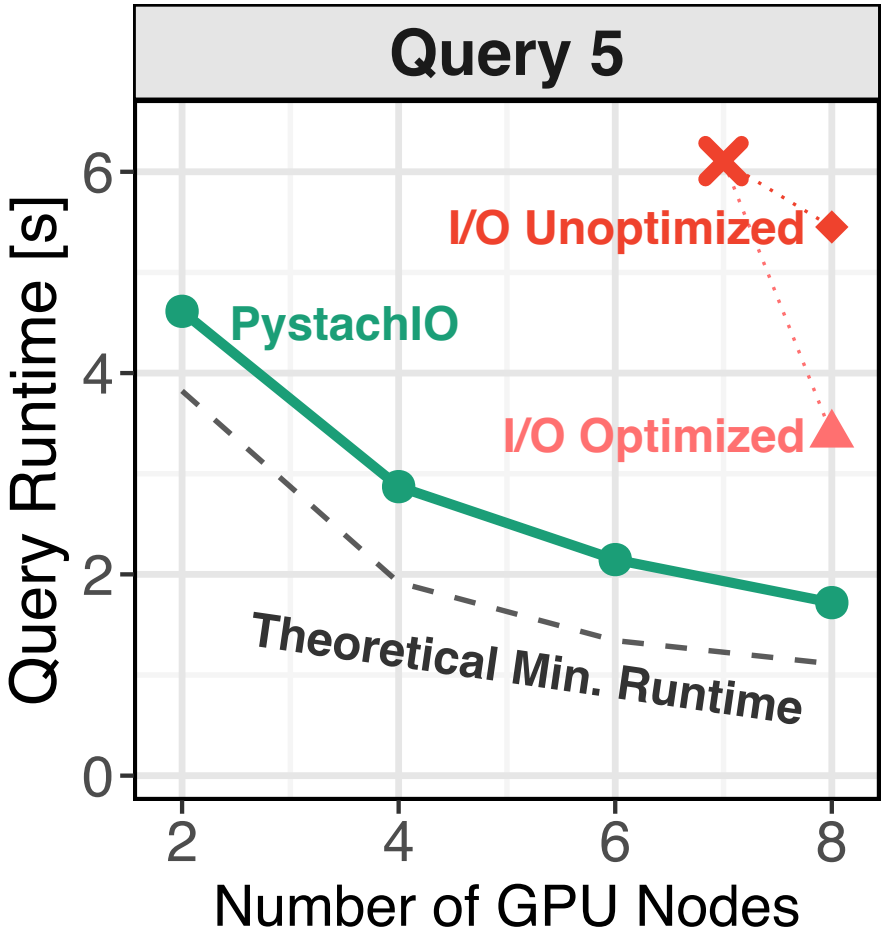
Overlapping GPU Computation and I/O (Storage and Network)

TPC-H Query 3 (SF 500) runtime executed on 2 GPUs (Each GPU has one SSD and one NIC)

Scalability: Dataset Size with 2 GPUs



Scalability: Number of GPUs with SF1000



The Road Ahead:

How Close Are We to **Speed-of-Light** GPU Data Processing?

1. The Curse of S3

What is S3?

- Extremely cheap!

1. The Curse of S3

What is S3?

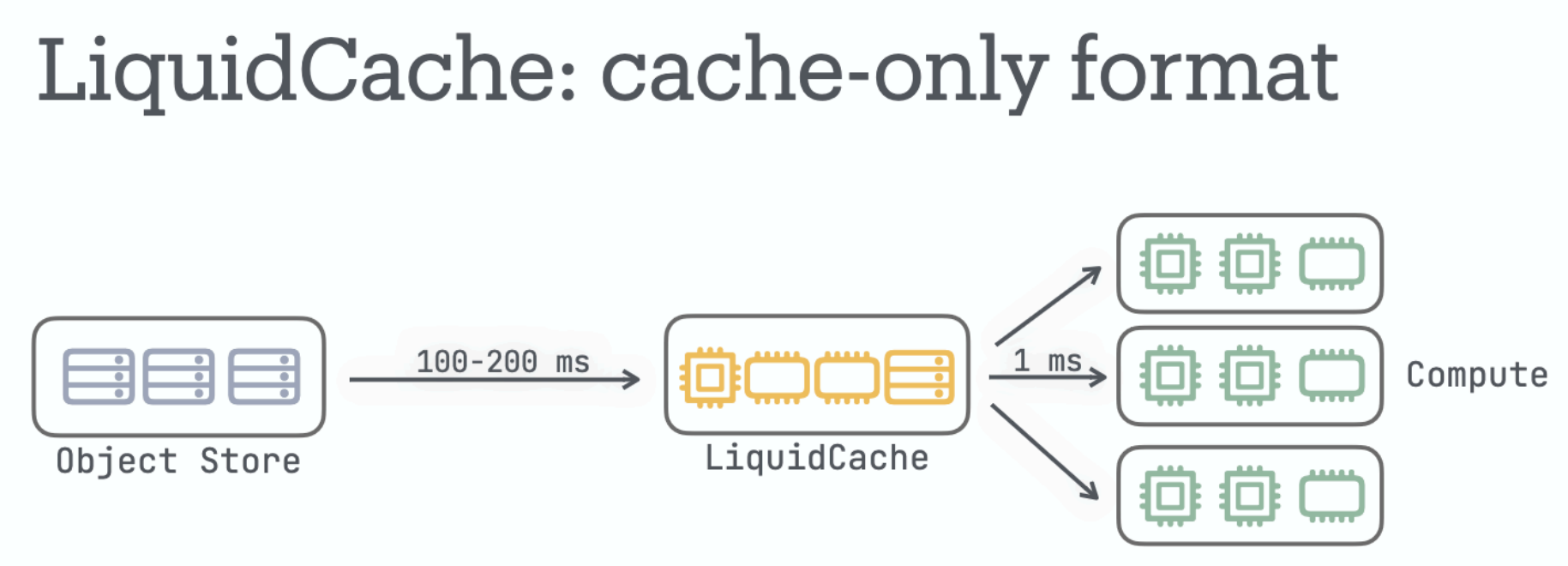
- Extremely cheap!
- Bandwidth: < 1 GB/s per connection
- Latency: ~100ms P99
- CPU-only: DNS, TLS, and HTTP

1. The Curse of S3

Why S3 is the Curse?

- GPUs idle due to S3 latency
- GPUs underutilized due to S3 bandwidth
- Moving the data path back to a CPU-centric system
- GPUs extremely expensive!

1. Caching Layer above S3



Xiangpeng Hao et al., *LiquidCache: Efficient Pushdown Caching for Cloud-Native Data Analytics*, PVLDB 18(13)

2. Team Parquet or Team SomethingElse?

2. Team Parquet or Team SomethingElse?

> *"If you want to go fast, go alone;*
> *if you want to go far, go together."*

Julien Le Dem, *Column Storage for the AI Era*, Iceberg Summit 2026

2. Team Parquet or Team SomethingElse?

Future File Format & GPU

- Hugging Face 🤗 × NVIDIA: collaboration
- **~50%** of Hugging Face 🤗 datasets in Parquet
- Parquet is for DB and AI!

2. Team Parquet or Team SomethingElse?

Future File Format & GPU

- Hugging Face 🤗 × NVIDIA: collaboration
- **~50%** of Hugging Face 🤗 datasets in Parquet
- Parquet is for DB and AI!
- Future Parquet: evolving the open standard

2. Team Parquet or Team SomethingElse?

Future File Format & GPU

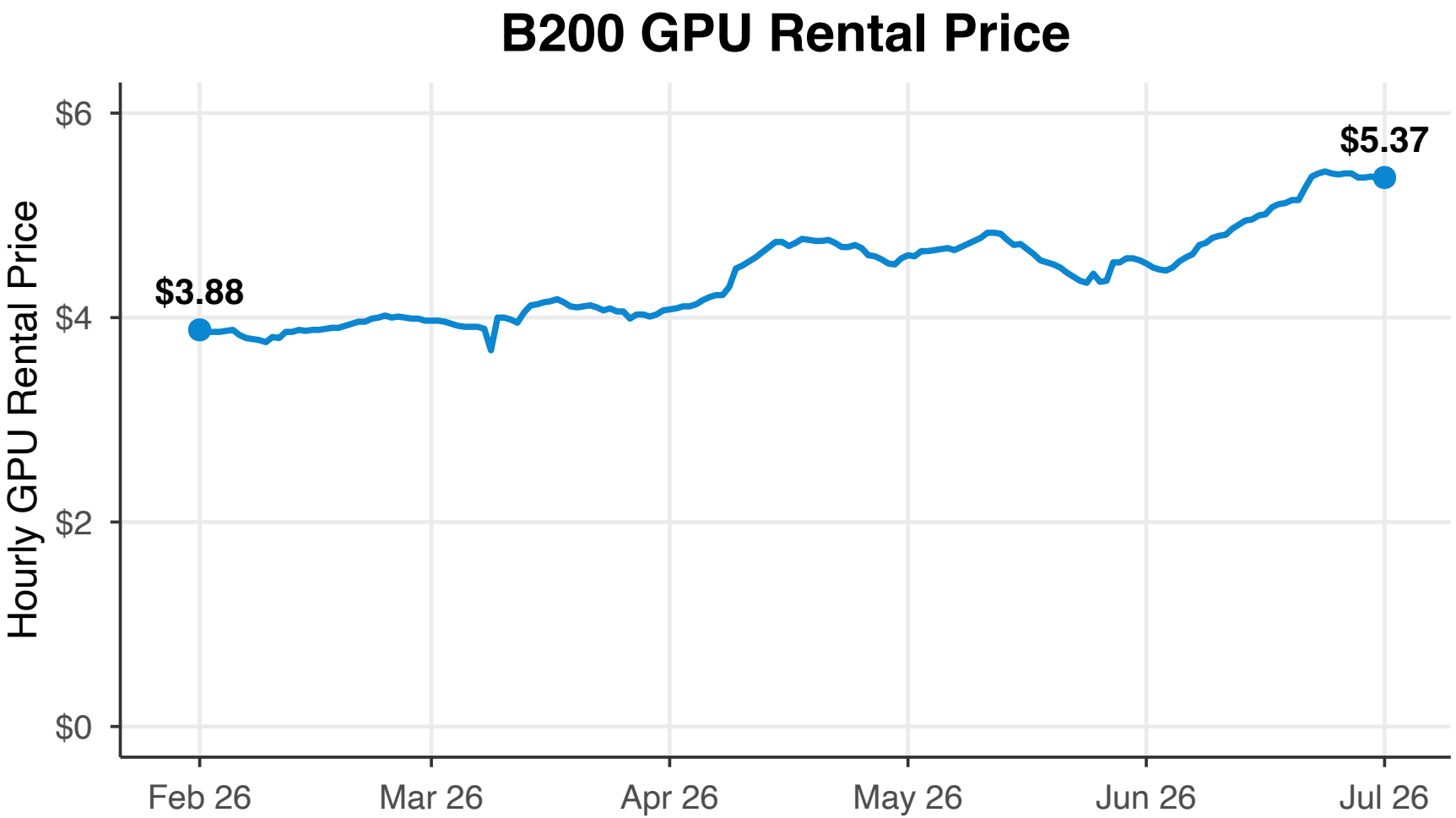
- Hugging Face 🤗 × NVIDIA: collaboration
- **~50%** of Hugging Face 🤗 datasets in Parquet
- Parquet is for DB and AI!
- Future Parquet: evolving the open standard
- I/O: BaM (SCADA)? GDS?*

* Data Path Fusion in GPU for Analytical Query Processing (arXiv:2605.10511)

2. Team Parquet or Team SomethingElse?

- File Formats: Software Campus × Snowflake
- Up to €115,000 · two years · from 2027
- Working with Russell Spitzer

3. The Unspeakable Cost



Source: SemiAnalysis GPU Pricing Index

3. The Unspeakable Cost

> *"There's a lot more money behind LLMs today. So why use GPUs for DB?"*

> *"... would be good to show the GPU compute and memory utilization ..."*

3. The Unspeakable Cost

What is the perfect GPU for DB?

- Large HBM
- Less SMs
- Recent PCIe Gen
- Low monetary cost -> Old GPU

Thanks For Your Attention!

Acknowledgement

Special thanks to my colleague Nils for our many fruitful discussions!

Acknowledgement

Special thanks to my colleague Nils for our many fruitful discussions!

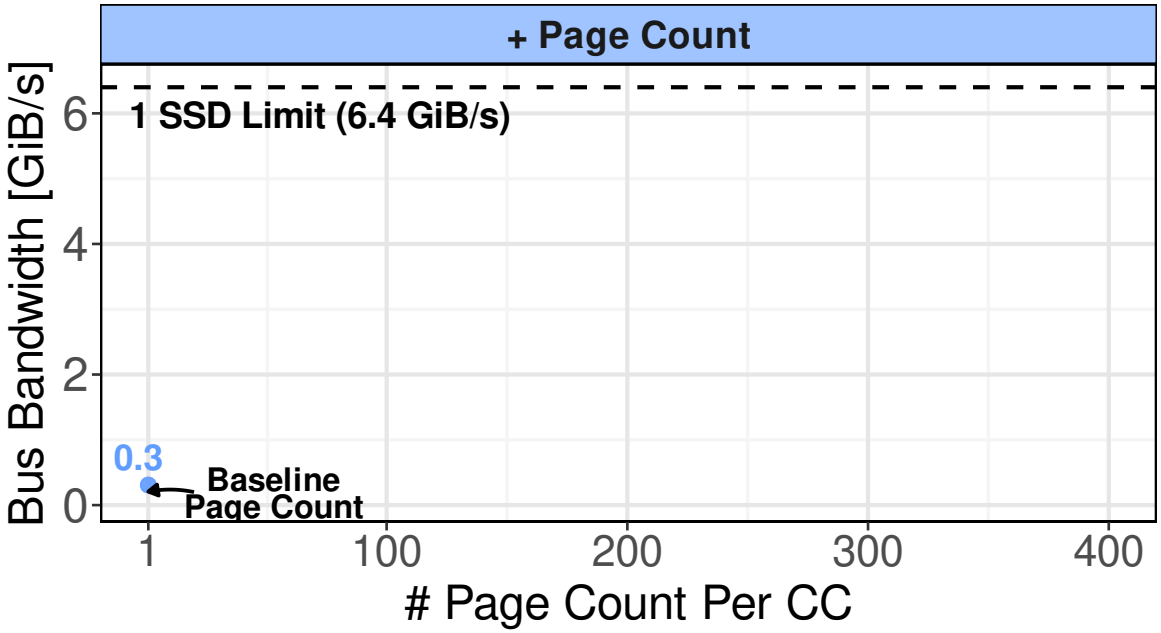
These slides are inspired by the following works. Many thanks to the authors:

- **DBs+GPUs: Where are we and what's next?**, Bowen Wu, Systems Group ETHz
- **G-ALP: Rethinking Light-weight Encodings for GPUs** @ DaMoN25, Sven Hepkema, ex-CWI DB & Systems Group ETHz
- **GPU Databases – The New Modality of Data Analytics**, Xiangyao Yu, University of Wisconsin-Madison
- **SIGMOD 26 Keynote: A New Golden Era for Data Management**, Gustavo Alonso, Systems Group ETHz
- **The Composable Codex**, Voltron Data

Backup: DaMoN 2026

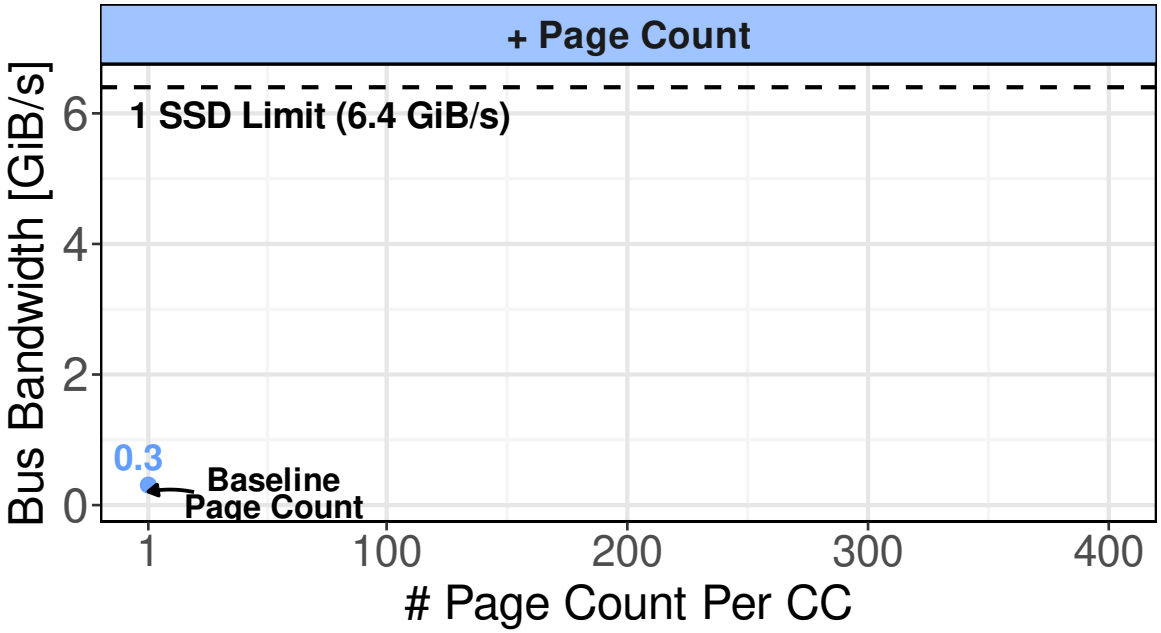
[BACKUP] Config. 1: Page Count Per Column Chunk

- Baseline of **1** due to CPU parallelism



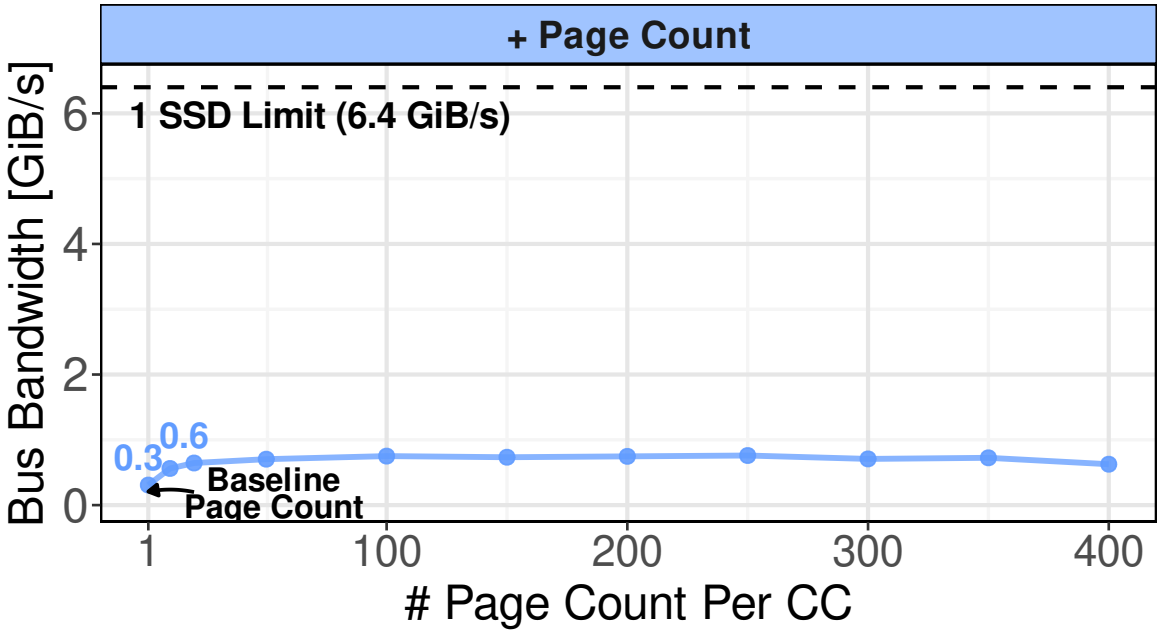
[BACKUP] Config. 1: Page Count Per Column Chunk

- Baseline of **1** due to CPU parallelism
- #pages mapped to GPU kernel parameter



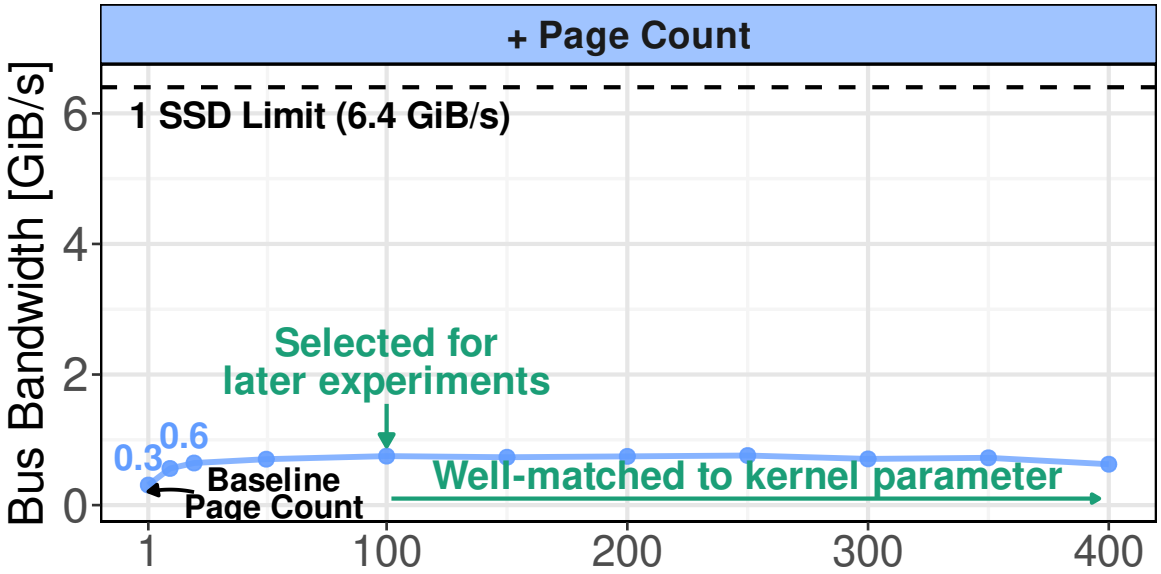
[BACKUP] Config. 1: Page Count Per Column Chunk

- Baseline of **1** due to CPU parallelism
- #pages mapped to GPU kernel parameter
- Too few pages underutilize GPU



[BACKUP] Config. 1: Page Count Per Column Chunk

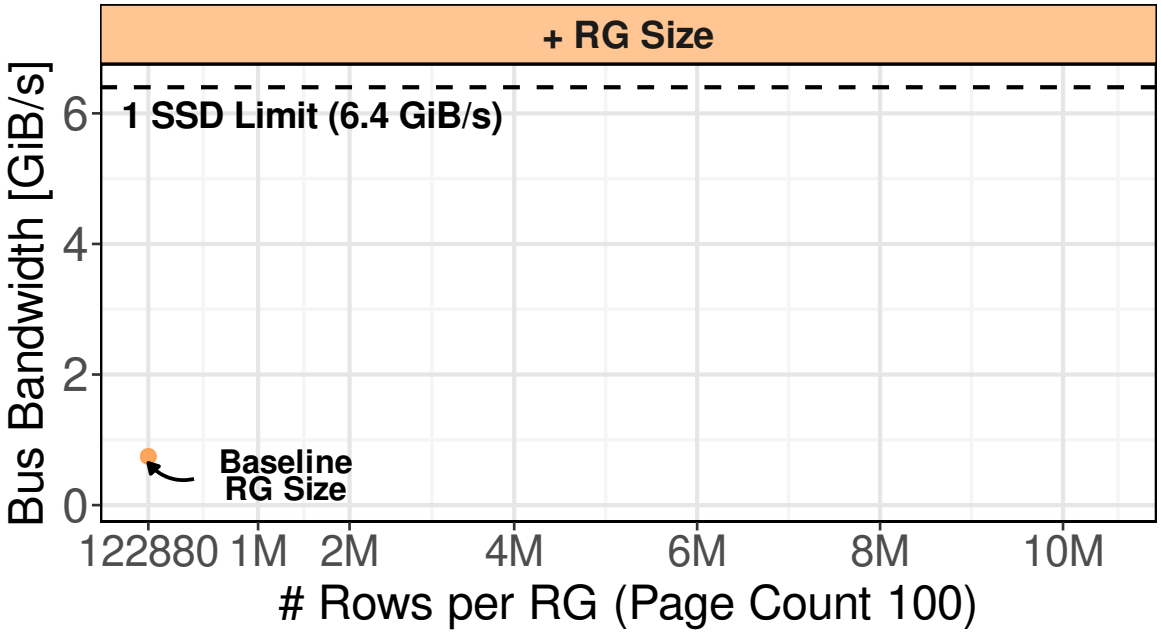
- Baseline of **1** due to CPU parallelism
- #pages mapped to GPU kernel parameter
- Too few pages underutilize GPU



Insight 1: Increase page count to match the recommended GPU kernel parameter

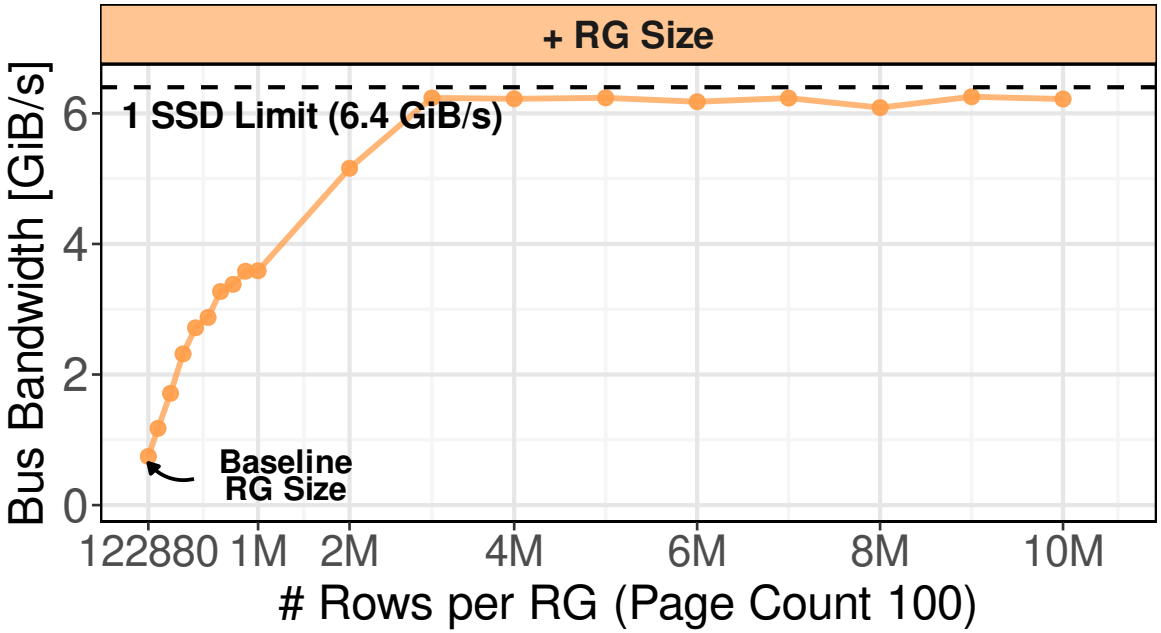
[BACKUP] Config. 2: Row Group Size

- GPU I/O stack differs from CPU's



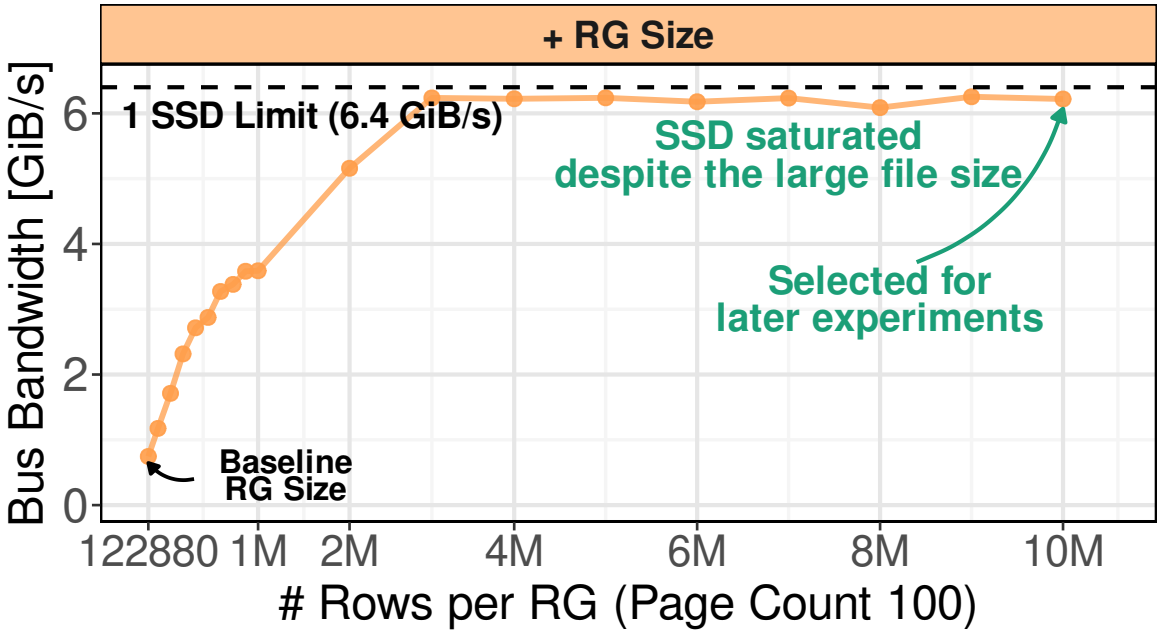
[BACKUP] Config. 2: Row Group Size

- GPU I/O stack differs from CPU's
- Larger RG improve bandwidth



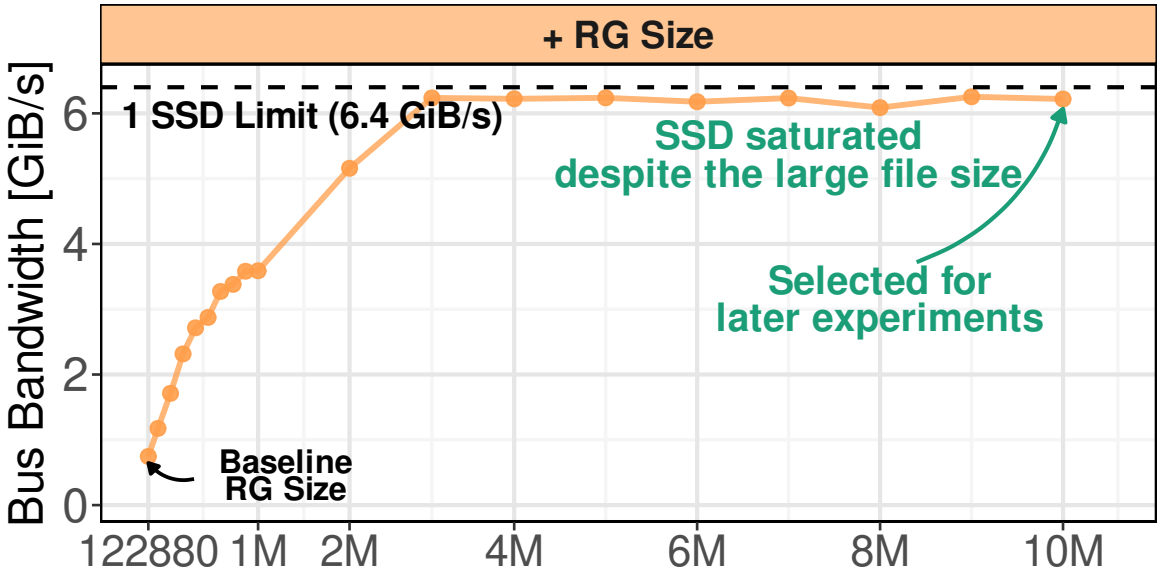
[BACKUP] Config. 2: Row Group Size

- GPU I/O stack differs from CPU's
- Larger RG improve bandwidth



[BACKUP] Config. 2: Row Group Size

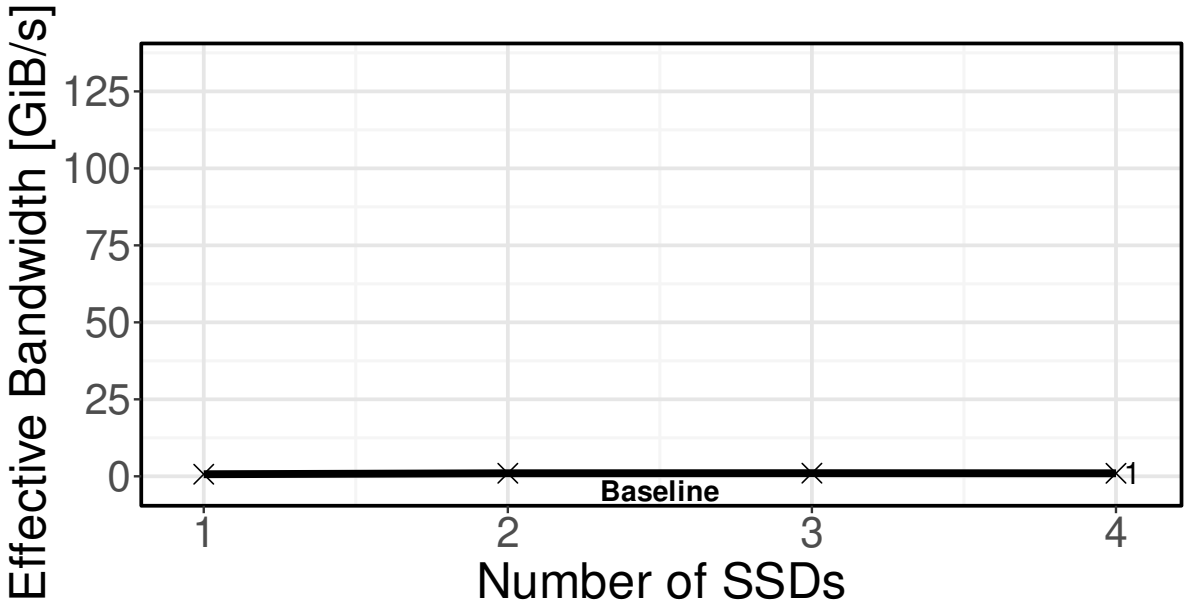
- GPU I/O stack differs from CPU's
- Larger RG improve bandwidth



Insight 2: Million-row RG sizes are preferred for GPU I/O stacks

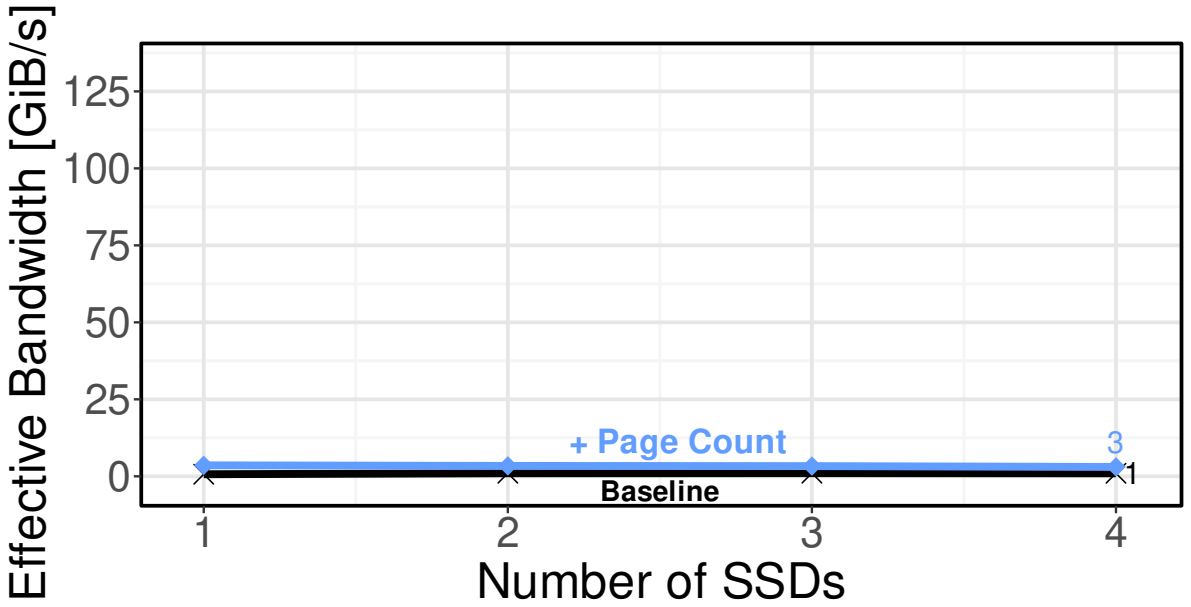
[BACKUP] Reading 4 SSDs

- Single SSD saturated
- Scaling to multiple SSDs



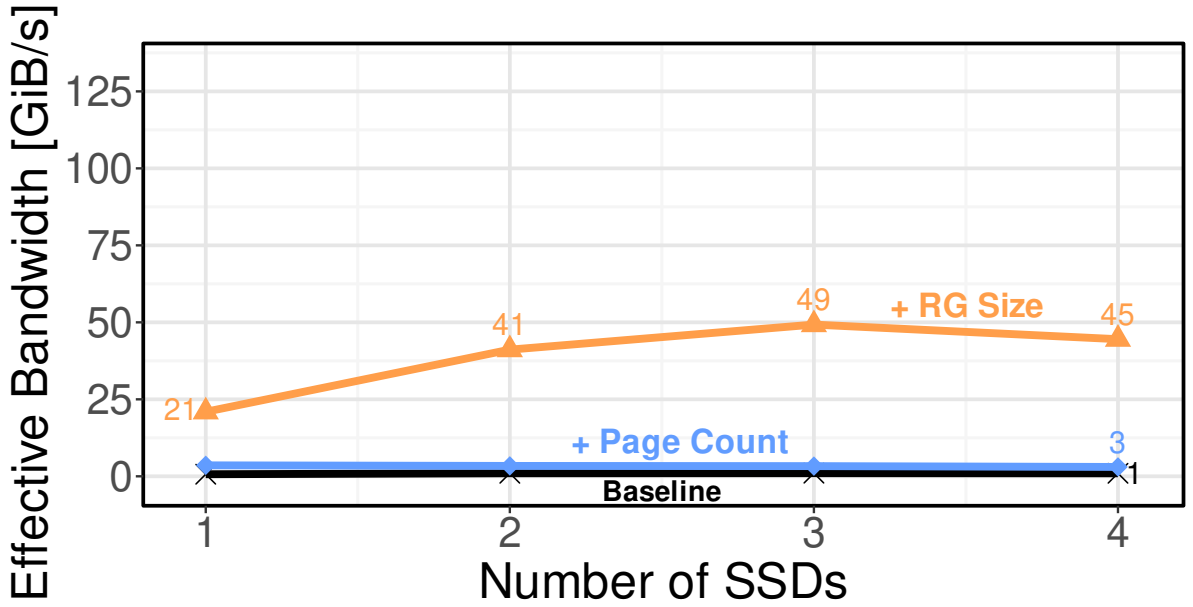
[BACKUP] Reading 4 SSDs

- Single SSD saturated
- Scaling to multiple SSDs



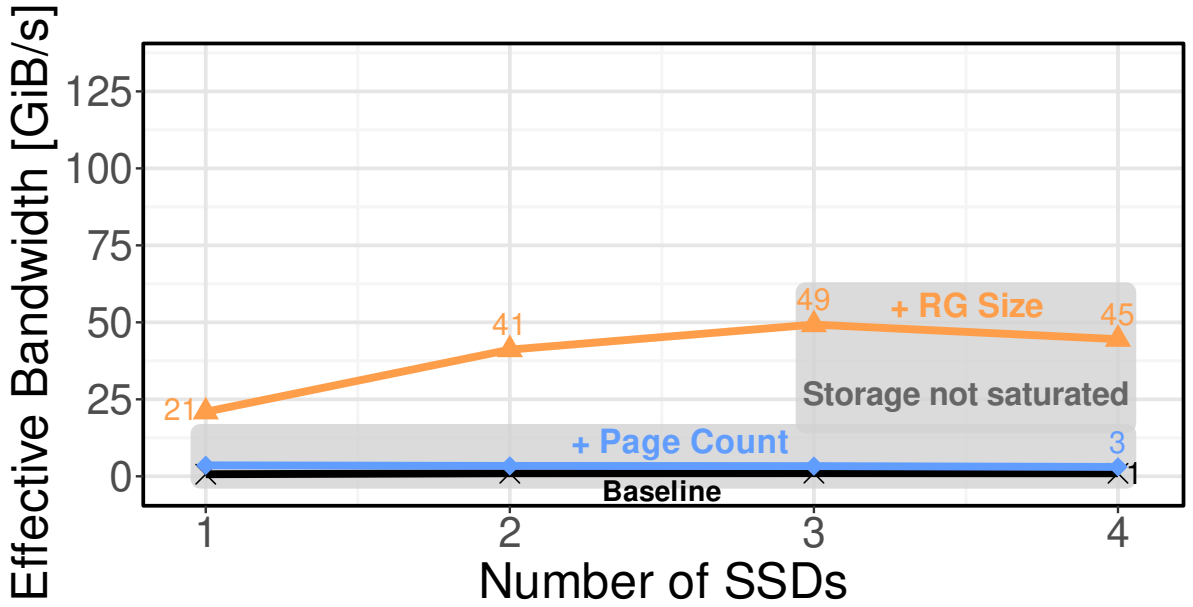
[BACKUP] Reading 4 SSDs

- Single SSD saturated
- Scaling to multiple SSDs



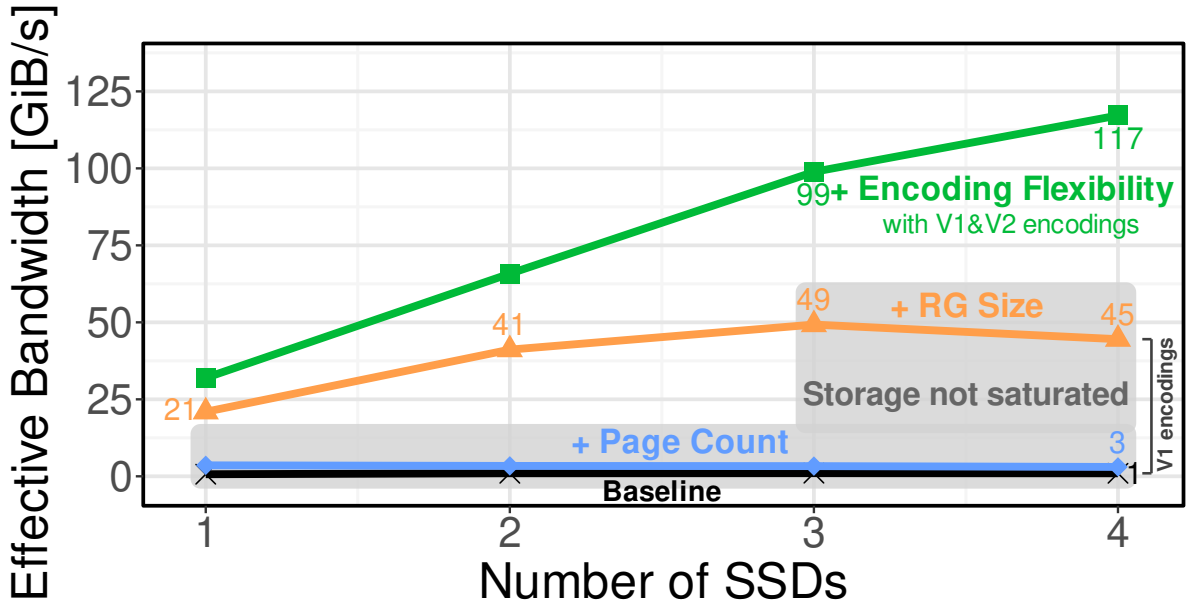
[BACKUP] Reading 4 SSDs

- Single SSD saturated
- Scaling to multiple SSDs



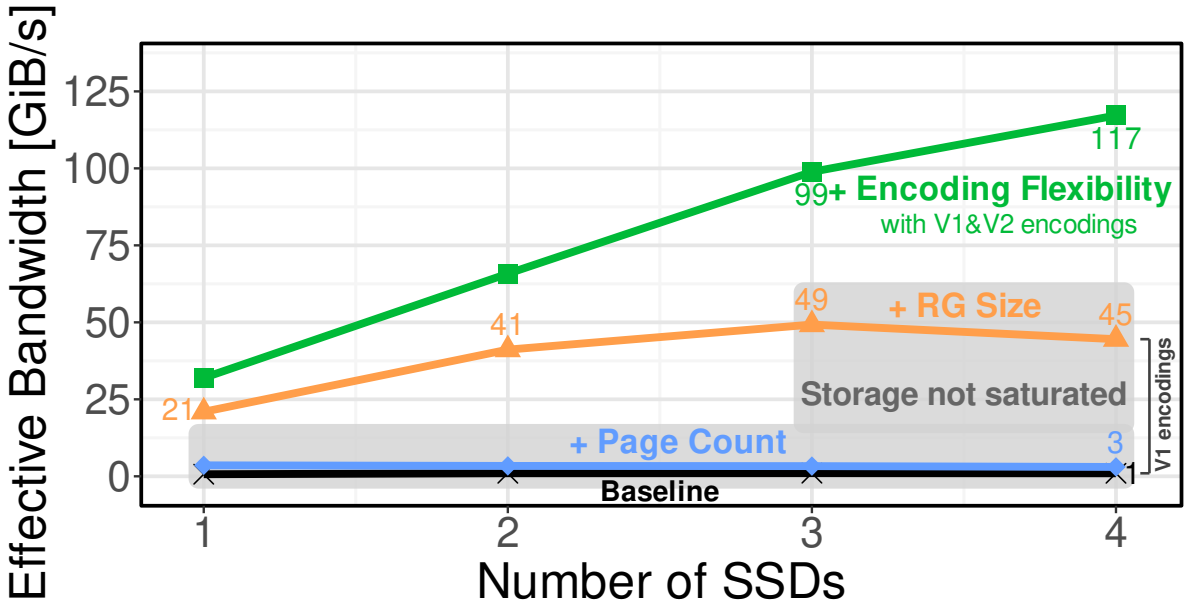
[BACKUP] Config. 3: Encoding Flexibility

- Consider both V1 and V2



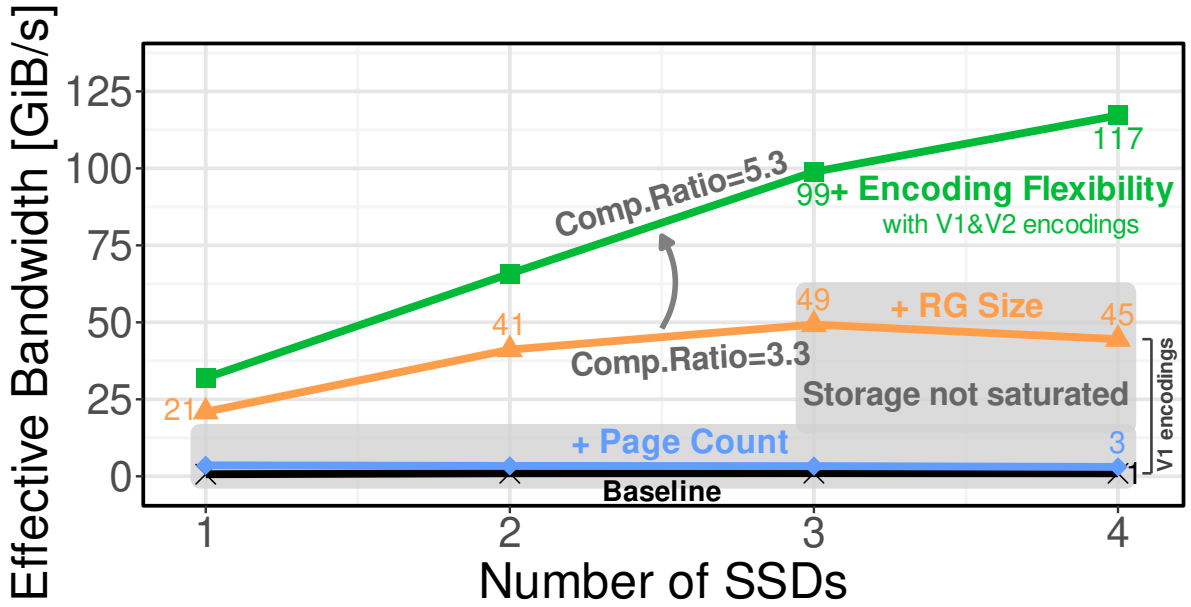
[BACKUP] Config. 3: Encoding Flexibility

- Consider both V1 and V2
- Finalize the encoding producing the smallest size



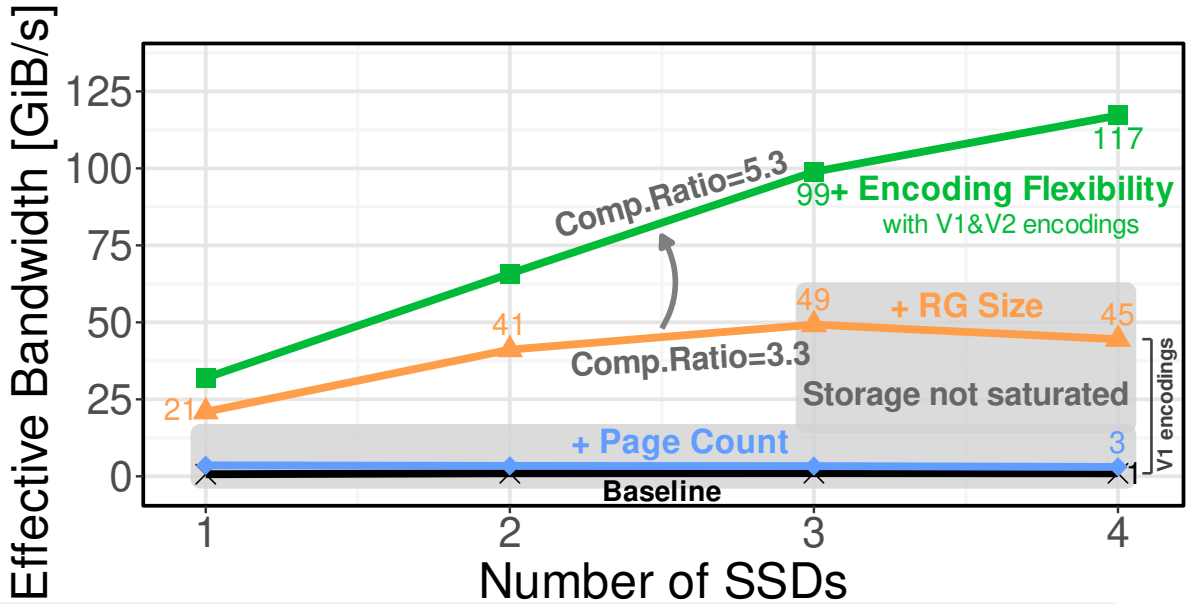
[BACKUP] Config. 3: Encoding Flexibility

- Consider both V1 and V2
- Finalize the encoding producing the smallest size
- Improve compression ratio



[BACKUP] Config. 3: Encoding Flexibility

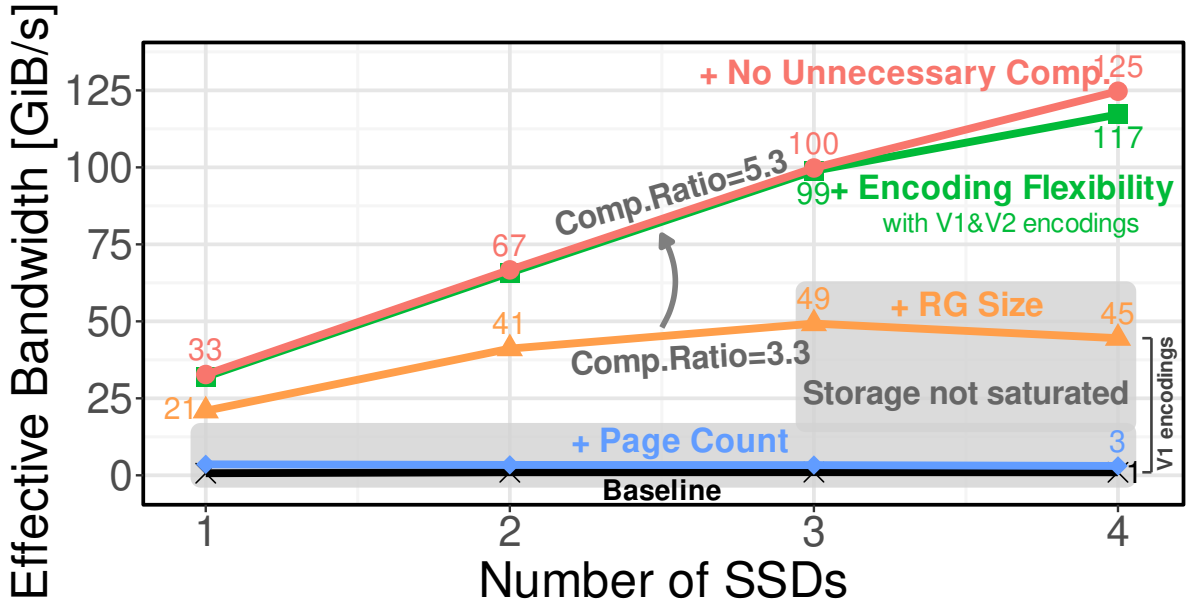
- Consider both V1 and V2
- Finalize the encoding producing the smallest size
- Improve compression ratio



Insight 3: Flexibly choosing encodings for size reduction

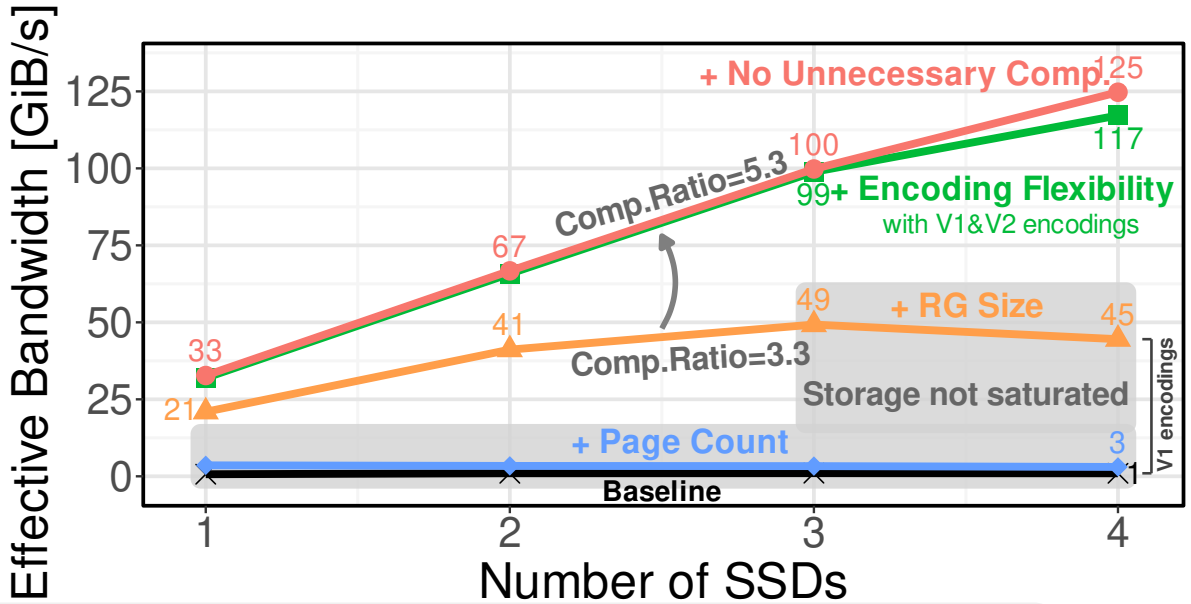
[BACKUP] Config. 4: No Unnecessary Compression

- Compress only when size reduction is meaningful
- Otherwise, uncompressed



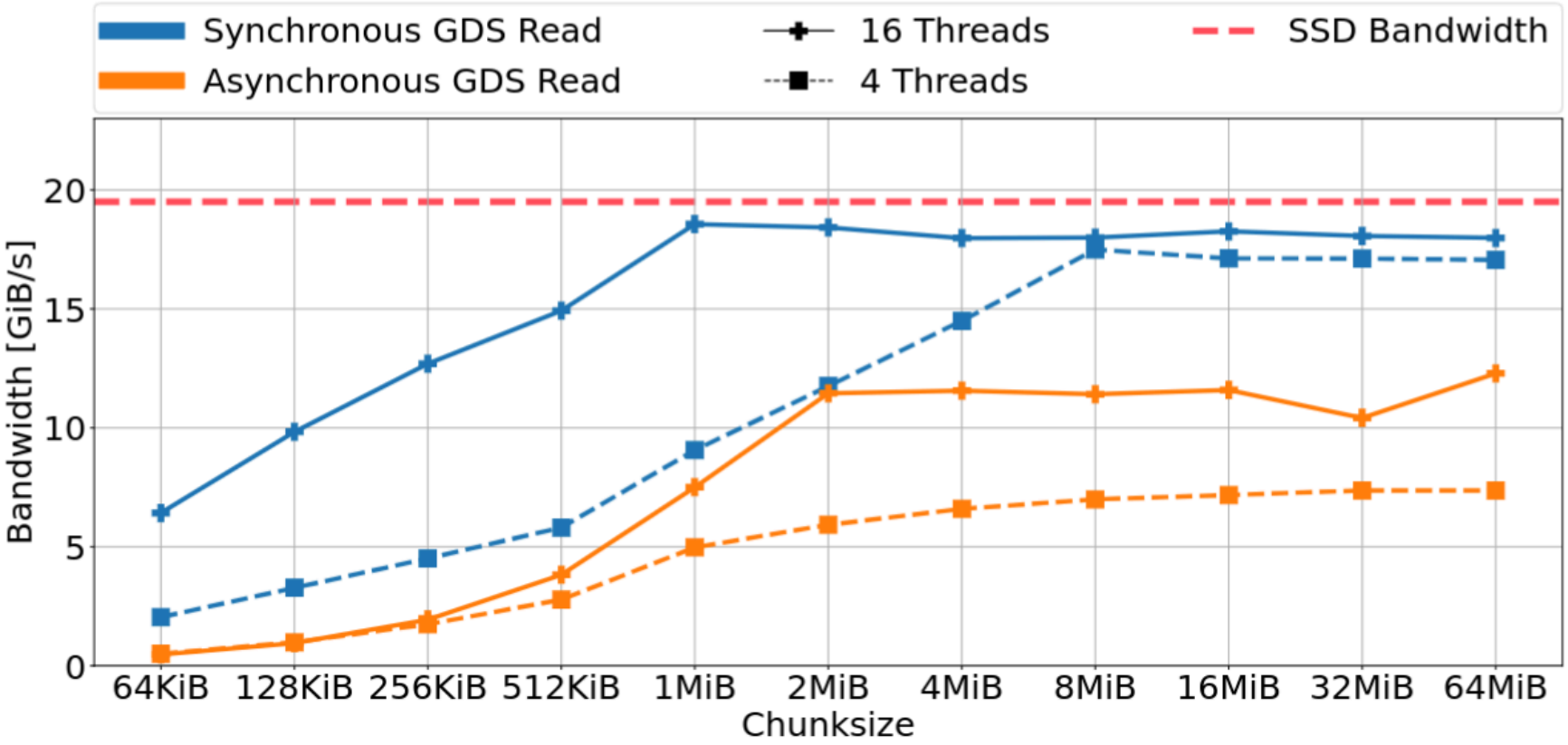
[BACKUP] Config. 4: No Unnecessary Compression

- Compress only when size reduction is meaningful
- Otherwise, uncompressed



Insight 4: Skip compression if no size reduction

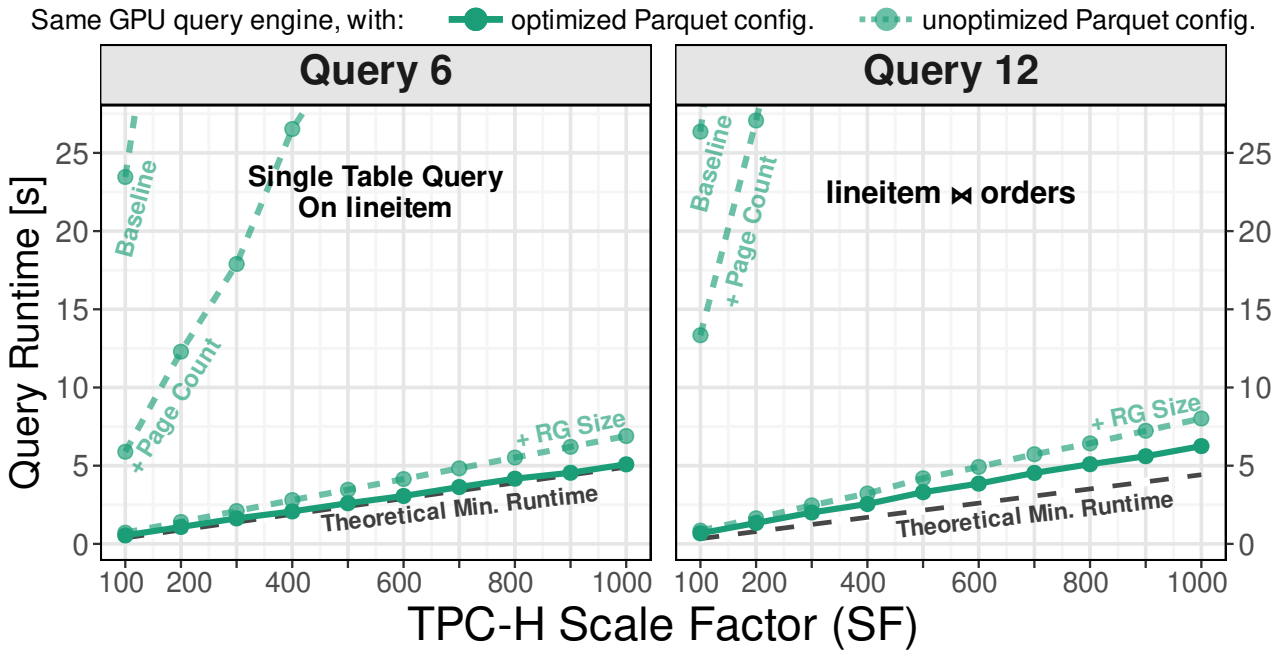
[BACKUP] NVIDIA GPU Direct Storage (GDS): Read Size



Source: Nils Boeschen 🎓, Tobias Ziegler, and Carsten Binnig. **GOLAP: A GPU-in-Data-Path Architecture for High-Speed OLAP**

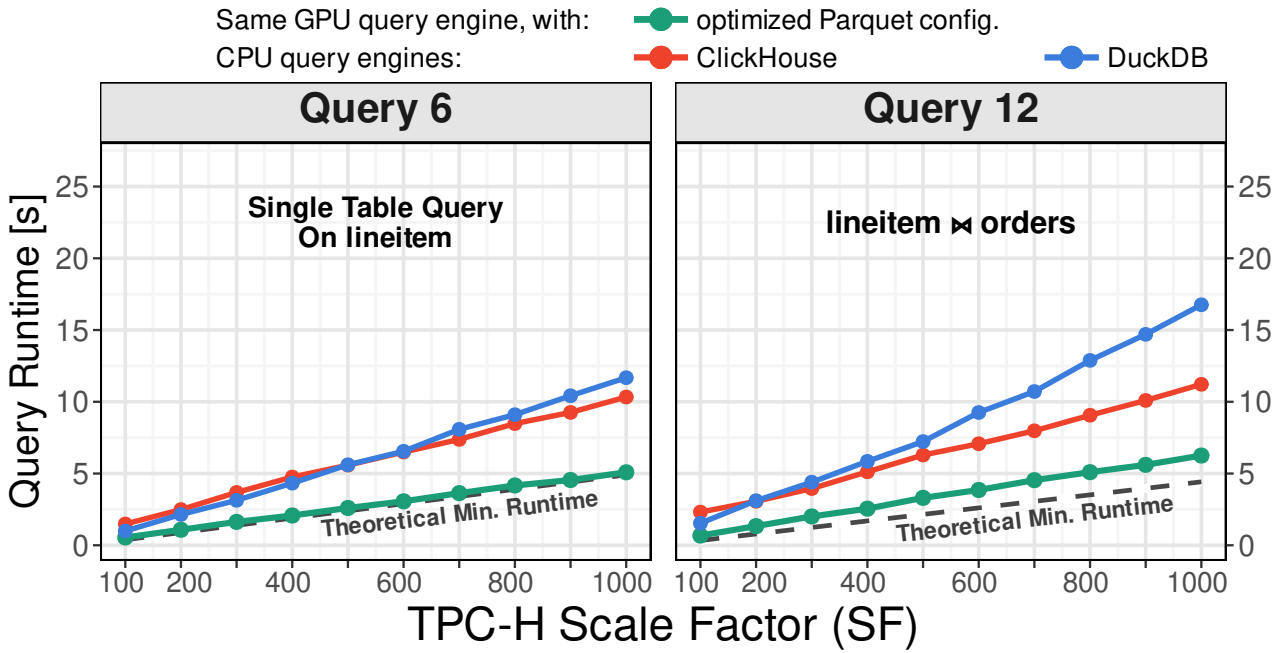
[BACKUP] Query Processing With Parquet Improvements

- Parquet improvements carry to queries



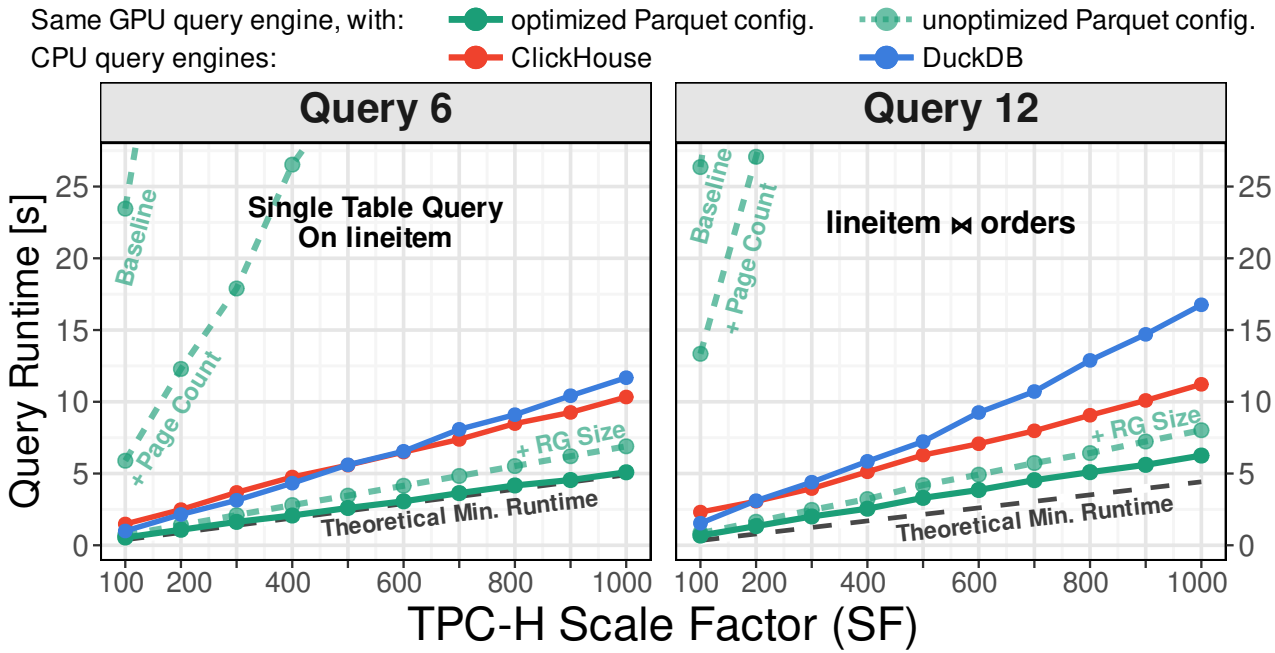
[BACKUP] Query Processing With Parquet Improvements

- Parquet improvements carry to queries



[BACKUP] Query Processing With Parquet Improvements

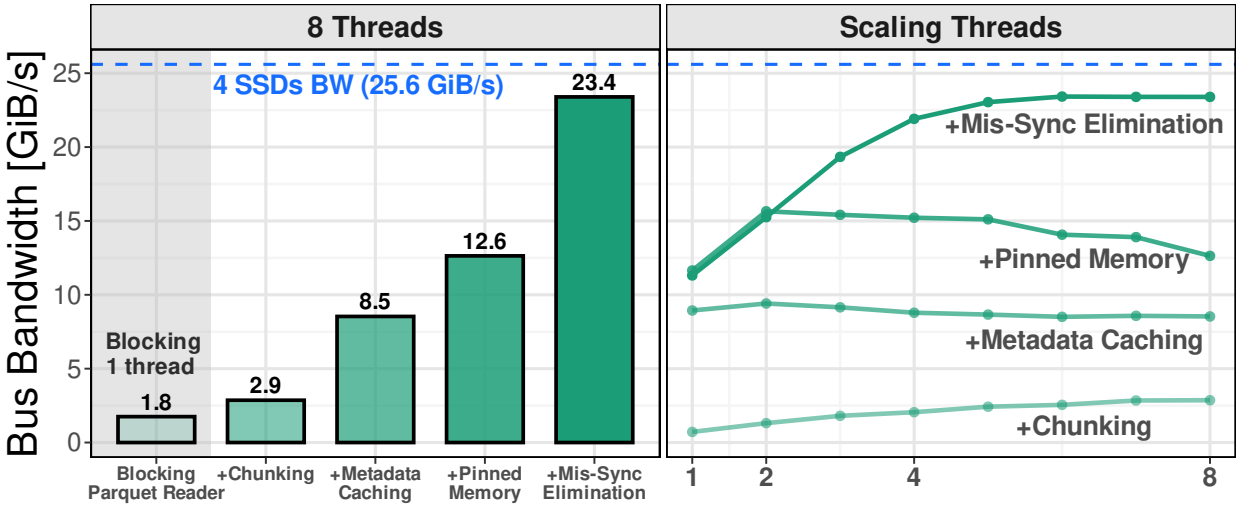
- Parquet improvements carry to queries



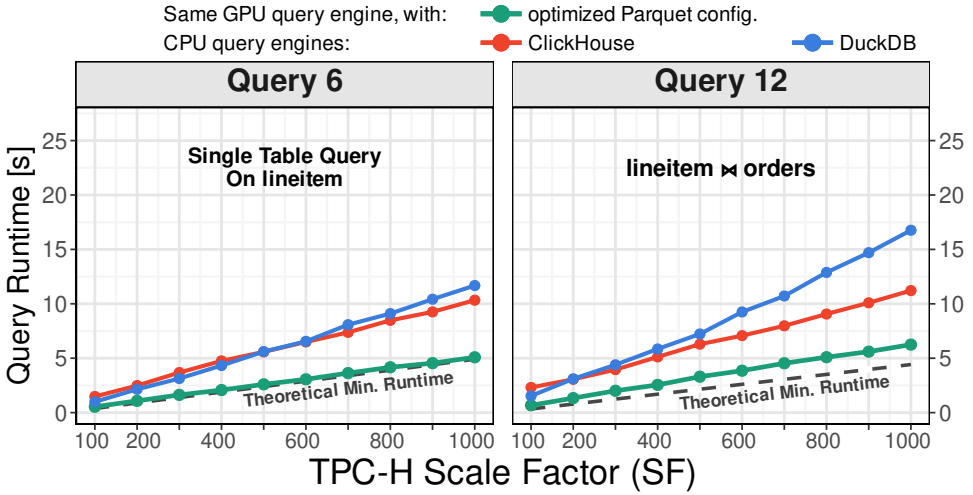
[BACKUP] Overlapping Reader: Optimizations

- Built on NVIDIA RAPIDS cuDF
- Merged upstream

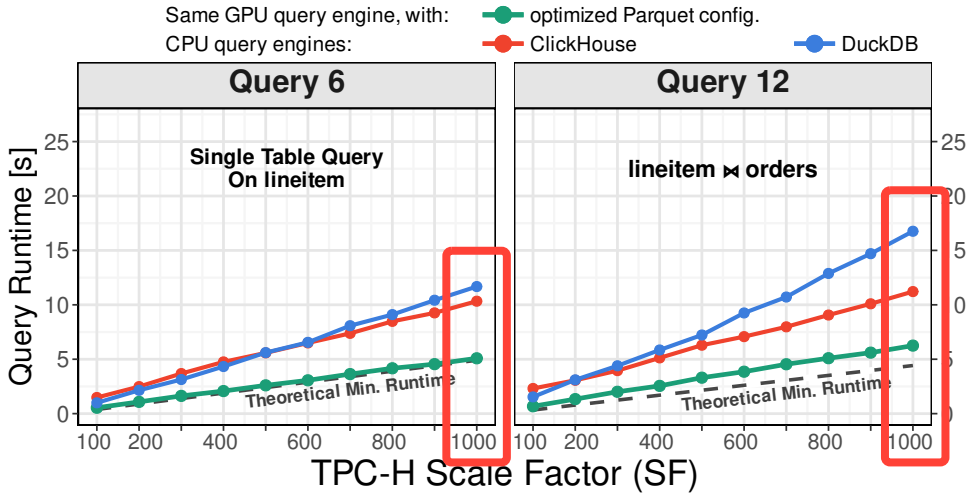
cuDF: [Story Issue] Towards a faster Parquet reader with pipelining and multistream optimization: https://github.com/rapidsai/cudf/issues/18892



[BACKUP] “Are GPUs Expensive?”



[BACKUP] “Are GPUs Expensive?”



	Runtime Slowdown vs. PystachIO		Relative TCO vs. PystachIO	
Query	ClickHouse	DuckDB	ClickHouse	DuckDB
Q6	2.0x	2.3x	5.0x	5.6x
Q12	1.8x	2.7x	4.4x	6.6x

Table 1: SF1000 single-node runtime slowdown and relative TCO.