OASIS: Object-Based Analytics Storage for Multi-Layer SQL Offloading in Scientific Tabular Workloads

Soon Hwang, Junhyeok Park, Junghyun Ryu, Seonghoon Ahn, Jeoungahn Park, Jeongjin Lee, Jinna Yang, Soonyeal Yang, Jungki Noh, Qing Zheng, Woosuk Chung, Hoshik Kim, and Youngjae Kim

Presenter: Soon Hwang / Sogang University

Sogang University | SK hynix | Los Alamos National Laboratory









SOGANG UNIVERSITY

DISCOS

Contents

Background

Motivation

Design

Evaluation

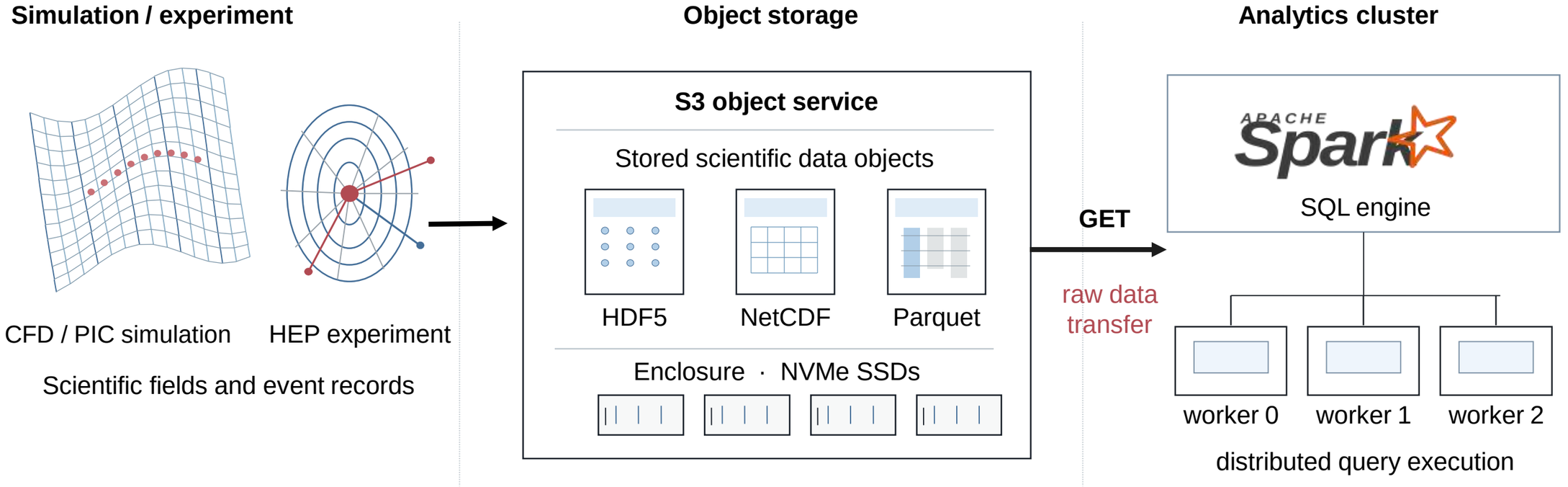
Conclusion

Background

SOGANG UNIVERSITY

DISCOS

Scientific Analytics on Disaggregated Storage

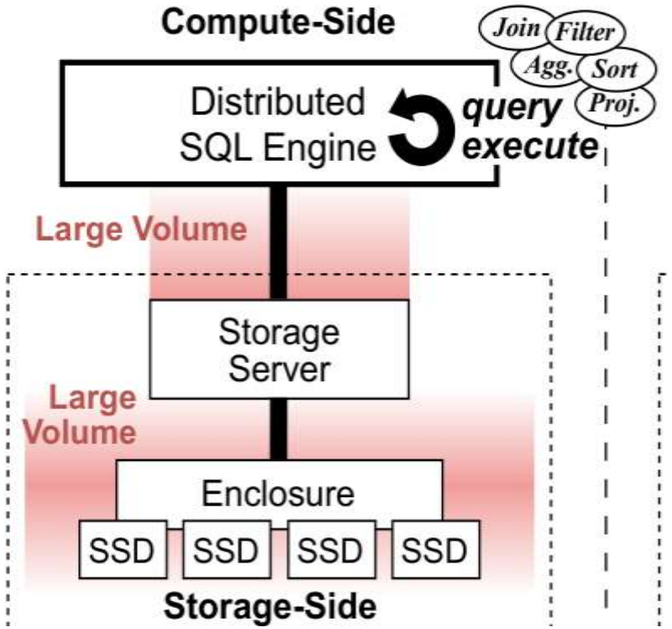


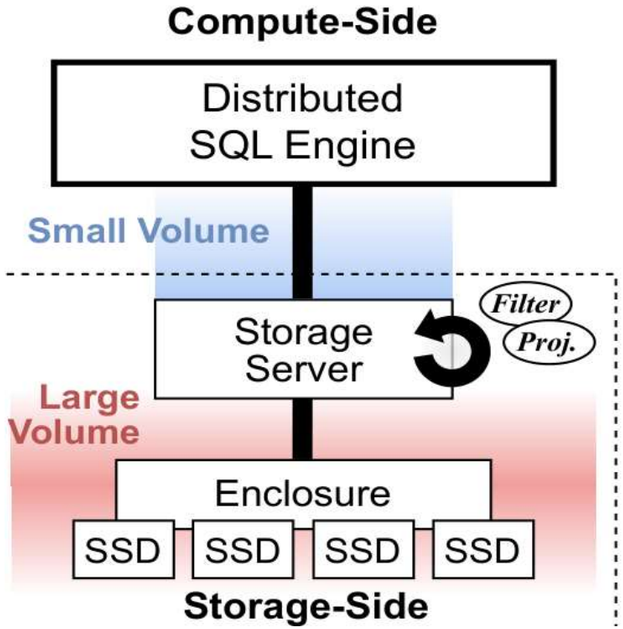
- Scientific datasets are stored as HDF5 [1], NetCDF [2], and Parquet [3] objects in remote storage.
- Spark executes analytics separately, requiring data movement across the compute–storage boundary.

SOGANG UNIVERSITY

DISCOS

Existing Computation-Enabled Object Storage





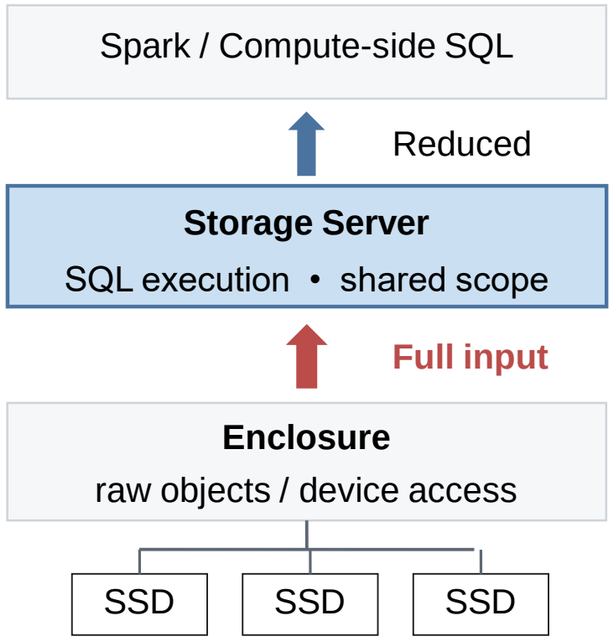
- Server-centric COS (S3 Select [4], MinIO Select [5], Skyhook [6]) runs offloaded operators at the storage server.
- Traffic to Spark is reduced, but raw data still moves from the Enclosure to the Storage Server.

SOGANG UNIVERSITY

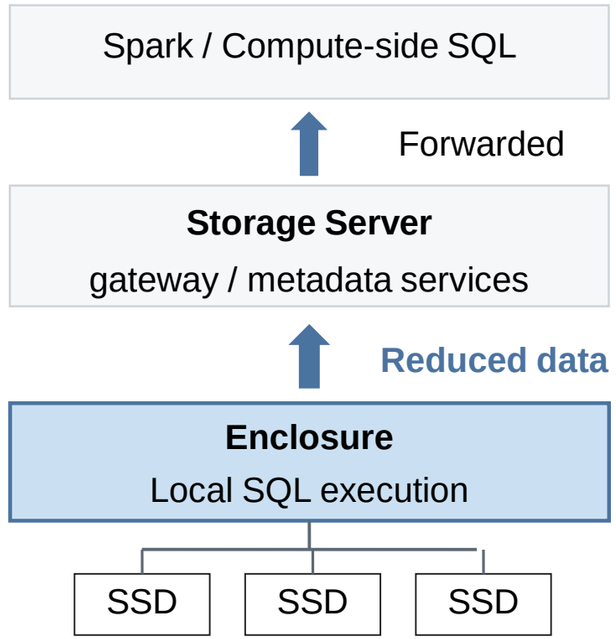
DISCOS

Where Existing Offloading Performs Computation

Server-centric COS [4]–[6]



Enclosure-based execution [7]



- Server-centric COS provides shared-scope execution, but moves the full input from Enclosure to Server.
- Enclosure-based execution reduces data early, but lacks coordinated global finalization across enclosures.

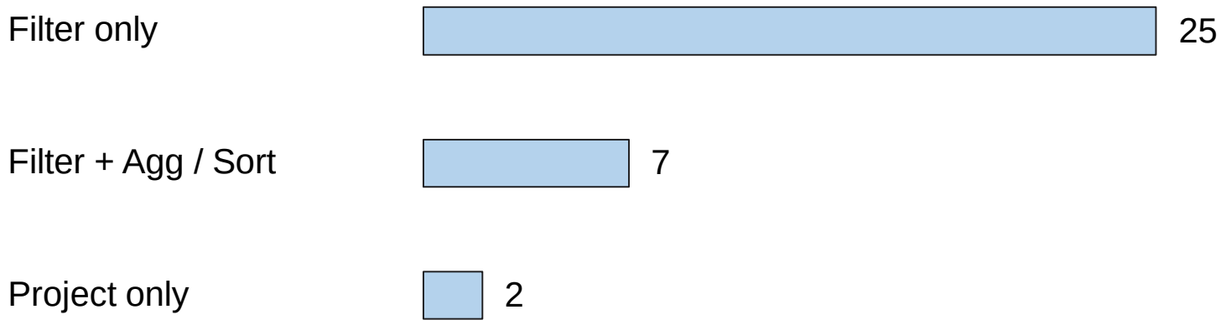
Motivation

SOGANG UNIVERSITY

DISCOS

Scientific Workload Observations

34 CFD / HEP / PIC queries [8], [9], [10]–[12]



Post-operator output / input

1.85%

mean

0.0002%

median

Scientific types: float, double, and arrays; HDF5 [1] and NetCDF [2] support scientific fields.

- O1 — Local reduction and shared-scope finalization coexist in scientific queries.
- O2 — Early operators often eliminate most intermediate data.
- O3 — Both storage layers must preserve scientific data types and columnar semantics.

SOGANG UNIVERSITY

Why Multi-Layer Execution Is Necessary

Requirements derived from workload observations

R1 Early reduction	O1: local + global operators; O2: strong early data reduction
R2 Shared finalization	O1: final Aggregate and global Sort require coordination
R3 Scientific semantics	O3: preserve scalar, array, and columnar formats

Single-tier systems do not satisfy all requirements

System	R1	R2	R3	Execution scope
S3 Select / MinIO [4,5]	No	No	No	Storage server
Skyhook [6]	No	Partial	Yes	Ceph OSD / server
Enclosure-based [7]	Yes	No	Partial	Near-device / enclosure
OASIS	Yes	Yes	Yes	Enclosure + server

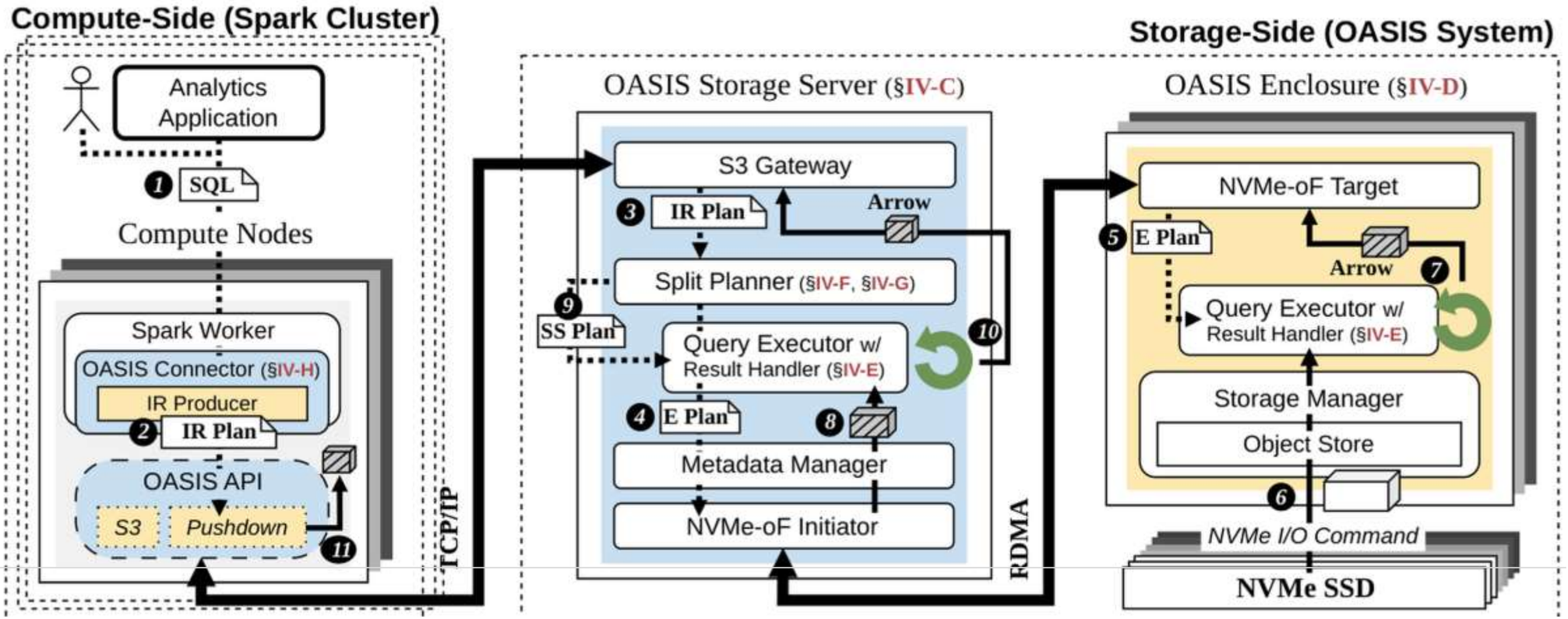
- The key challenge is execution-boundary placement across storage layers, not just operator selection.

Design

SOGANG UNIVERSITY

DISCOS

OASIS System Overview

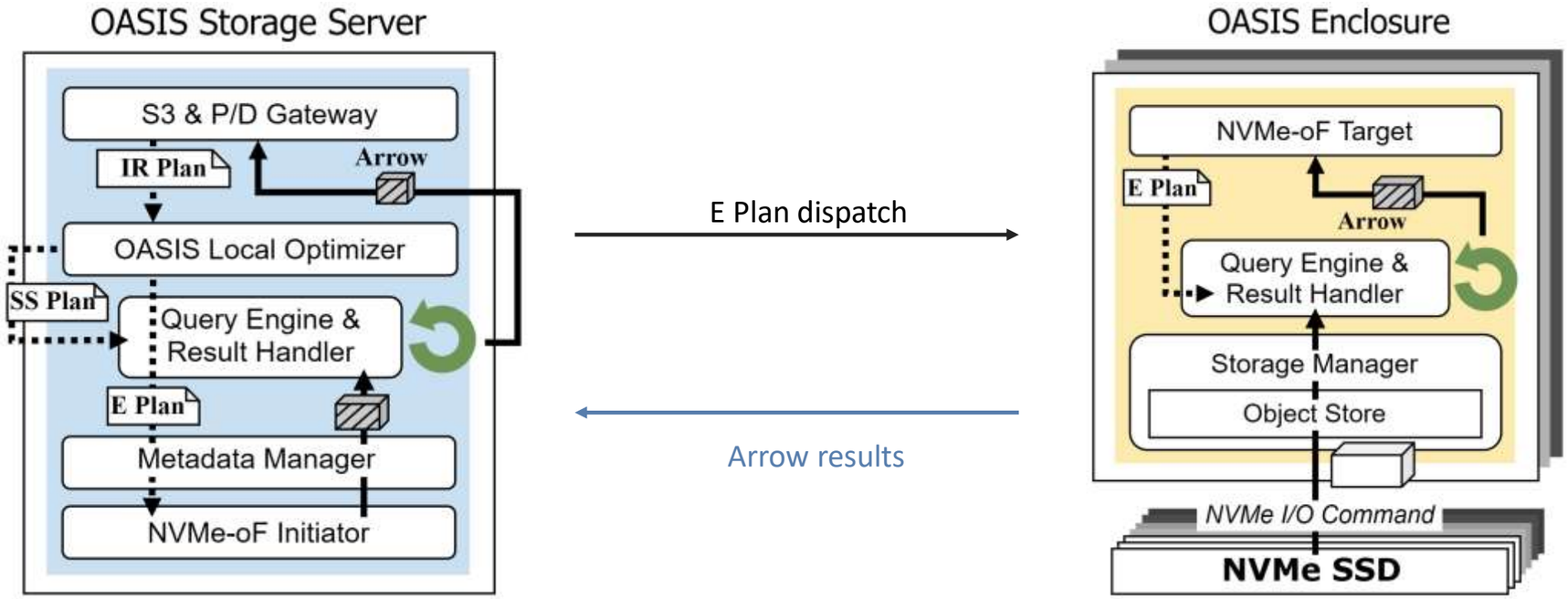


- Spark submits a Substrait [16] fragment; SODA chooses the Enclosure (E Plan) / Server (SS Plan) split.
- SPDK-based Enclosure access and DuckDB [18] enable local execution; Arrow [17] carries intermediate batches.

SOGANG UNIVERSITY

DISCOS

OASIS Storage-Side Components



- Gateway — receives S3 and IR [16].
- SODA — selects the E/SS split.
- Metadata Manager — finds objects.
- SS Executor — finalizes results.
- NVMe-oF Target — receives E Plan.
- Storage Manager — reads Parquet.
- E Executor — runs near SSDs.
- Result Handler — streams Arrow [17].

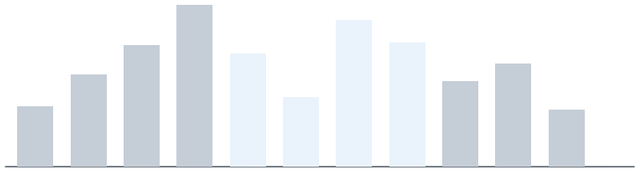
SOGANG UNIVERSITY

DISCOS

SODA: Data-Size Estimation and Split Selection

Goal: minimize the estimated intermediate bytes transferred from Enclosure to Storage Server.

1. Collect statistics



Predicate range overlaps sampled bins

- 0.5–5% sampled data; 254 bins per nume ric column.
- Read widths from Parquet metadata.

2. Estimate output sizes

Filter	Histogram overlap → passing rows
Project	Selected columns → tuple width
Aggregate	Filtered bins → group/state es timate

3. Select a split

Final Aggregate / Sort

E → SS boundary

Project / Partial Agg

ReadFilter

Compare only feasible candidate cuts.

- Semantic constraint — Final Aggregate and global Sort remain at the Storage Server.
- Resource constraint — Oversized partial states or unknown-selectivity operators remain at the Storage Server.

SOGANG UNIVERSITY

SODA Example: Original IR Plan

Offloaded Substrait IR Plan

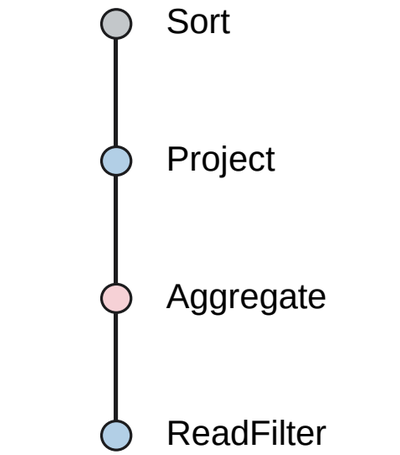
URIs · Extensions

Relations

root: sort
- project
  - aggregate
    - readFilter

Outputs · result schema

Relation Tree (root → leaf)



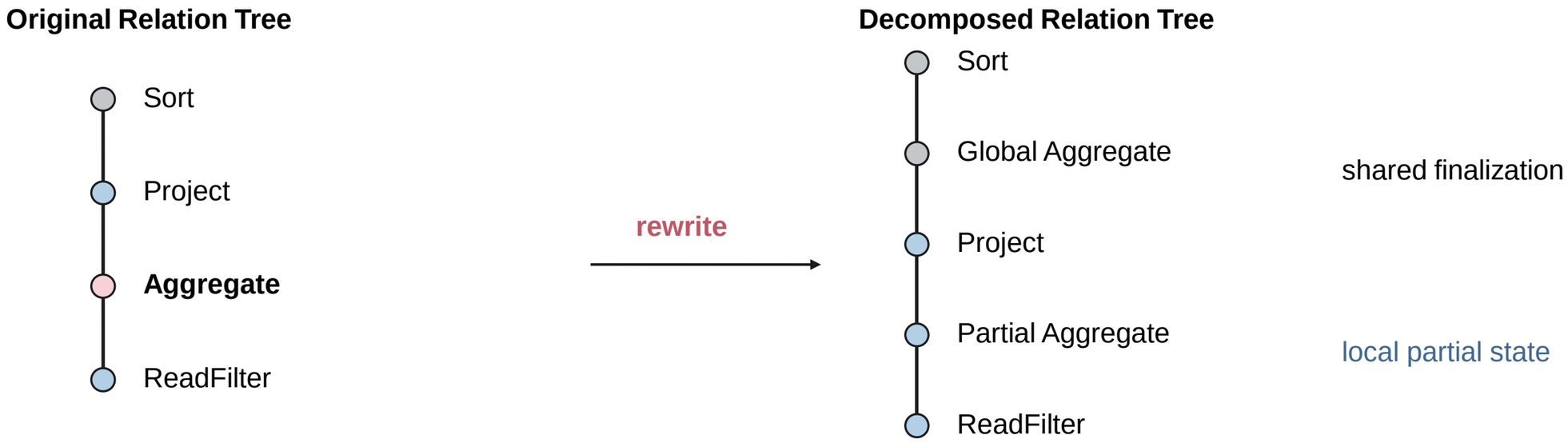
enclosure-local shared scope rewrite target

- SODA receives an already-offloaded linear plan: Sort → Project → Aggregate → ReadFilter.
- The Substrait [16] relation tree exposes legal operator boundaries and the intermediate schema.

SOGANG UNIVERSITY

DISCOS

SODA Example: Decompose Aggregate



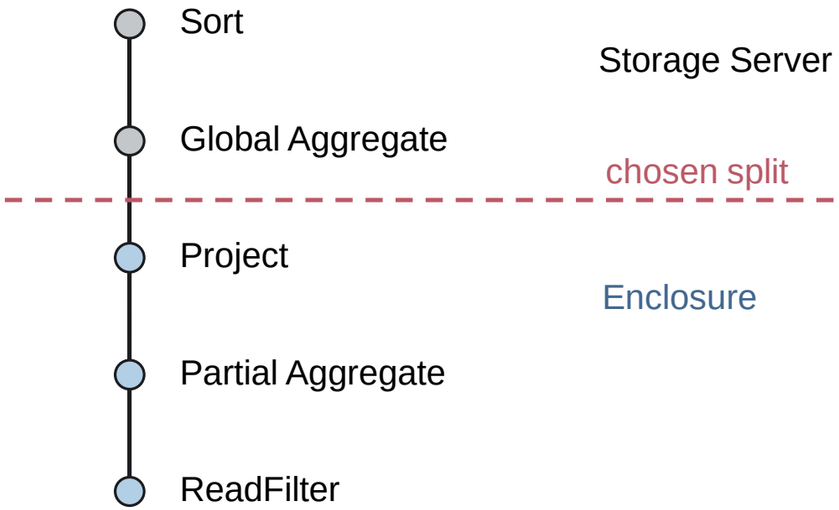
- Aggregate is rewritten into a Partial Aggregate plus a Global Aggregate.
- Partial states can be generated locally; merging/finalization requires the Storage Server.

SOGANG UNIVERSITY

DISCOS

SODA Example: Choose the Split Point

Rewritten Relation Tree



Feasible candidate boundaries

- After ReadFilter
- After Partial Aggregate
- **After Project ← selected**

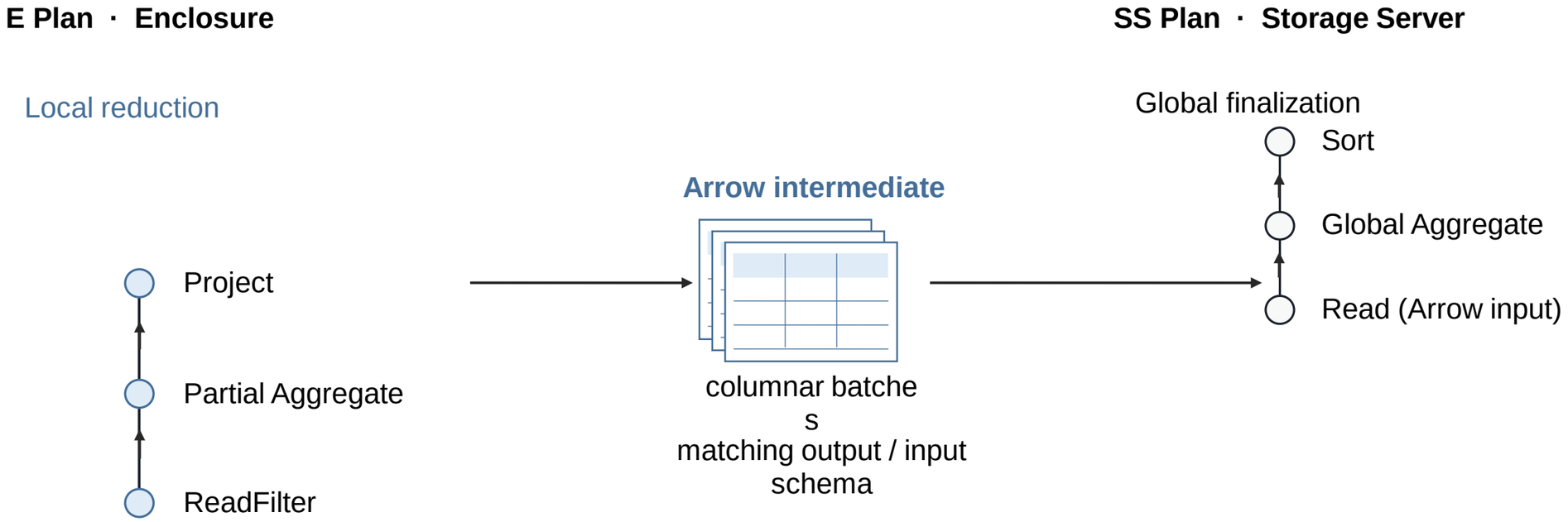
compare estimated E → SS bytes at each boundary

- SODA estimates the intermediate bytes at candidate boundaries, then excludes semantically or memory-infea sible choices.
- For this example it places ReadFilter, Partial Aggregate, and Project in the Enclosure.

SOGANG UNIVERSITY

DISCOS

SODA Example: Generate E Plan and SS Plan

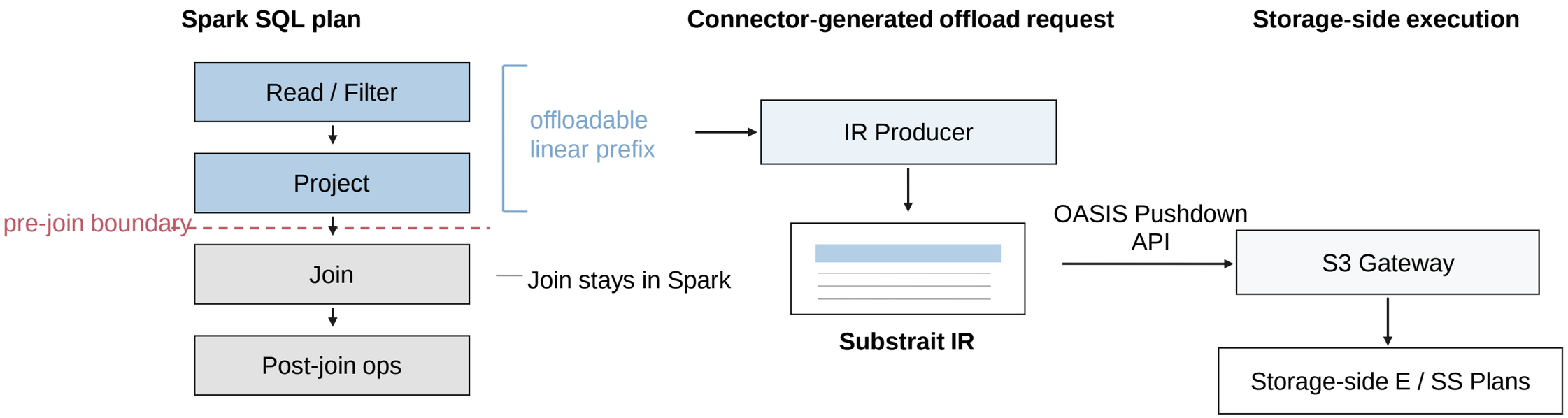


- URIs, extensions, and referenced schemas are preserved in both subplans.
- The Enclosure sends Arrow [17] batches; a new Read passes them to the SS Plan.
- The Storage Server finalizes the global Aggregate and Sort.

SOGANG UNIVERSITY

DISCOS

Spark Connector



- The client connector chooses an offloadable linear prefix; Join and downstream operators remain in Spark.
- The IR Producer translates the prefix into Substrait [16] and submits it through the OASIS API.

SOGANG UNIVERSITY

DISCOS

Evaluation

SOGANG UNIVERSITY

DISCOS

Experimental Setup

Spark: 1 driver + 2 executors 10 GbE → **Storage Server** 100 GbE → **Enclosure (2 × 1 TB NVMe)**

Hardware configuration (per node; OASIS Table IV)

Role	Cores	GHz	RAM	SSD
Spark driver	64	3.9	386 GB	–
Spark executor ×2	112	3.1	128 GB	–
Storage Server	48	3.9	64 GB	–
Enclosure	16	2.3	64 GB	2 × 1 TB

Software and comparison settings

In-storage executor	DuckDB v0.8.0
Direct object manager	SPDK v23.09 (BDEV)
S3-compatible gateway	VersityGW v2.49.2
Baseline deployments	MinIO; Ceph Quincy v17.2

Evaluation workloads

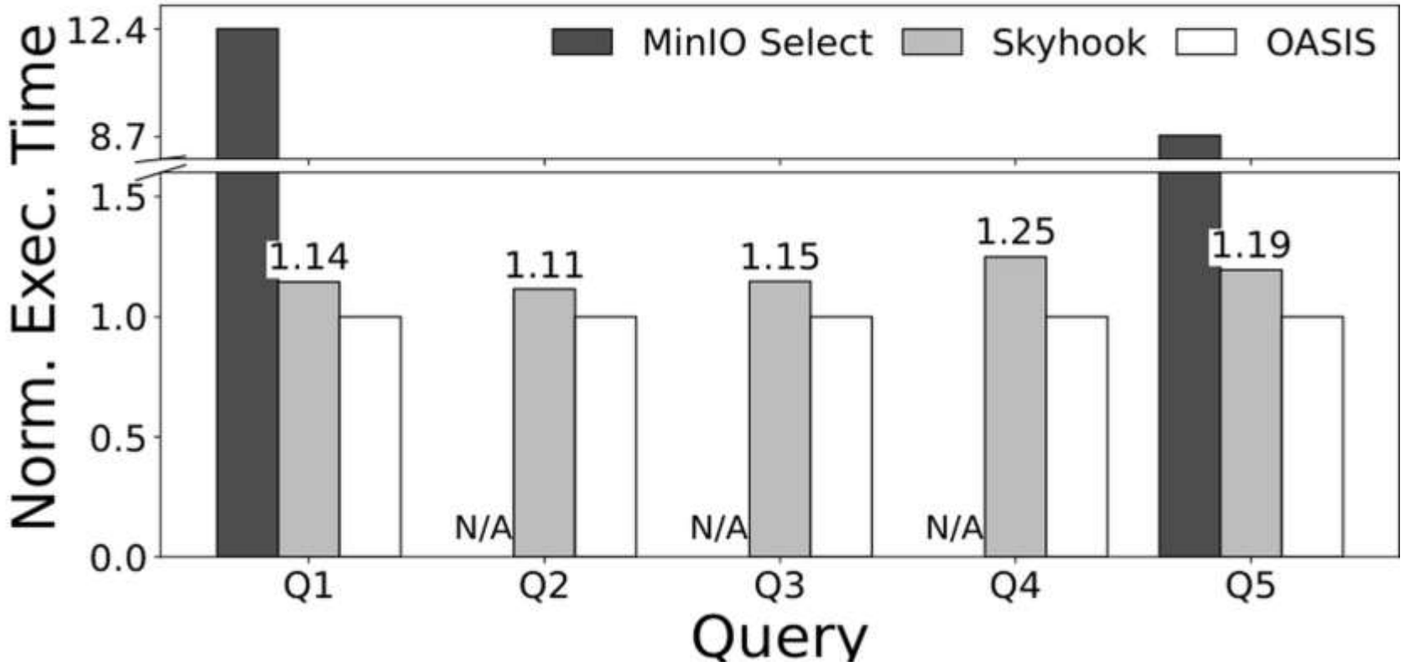
Query	Dataset / domain	Input	Primary operators
Q1	Laghos [8] · CFD	20 GB	Filter / Aggregate / Sort
Q2	DeepWater [9] · CFD	13 GB	Filter / Project
Q3	DeepWater [9] · CFD	30 GB	Filter / Aggregate
Q4	CMS Open Data [19] · HEP	12 GB	Array expressions
Q5	TPC-H Q4 · SQL benchmark	1.3 GB	Pre-join prefix

- Baselines: Spark-Only, MinIO Select [5], Skyhook [6], SS-Only, and OASIS; five runs per case (median).

SOGANG UNIVERSITY

DISCOS

Comparison with Existing COS Systems

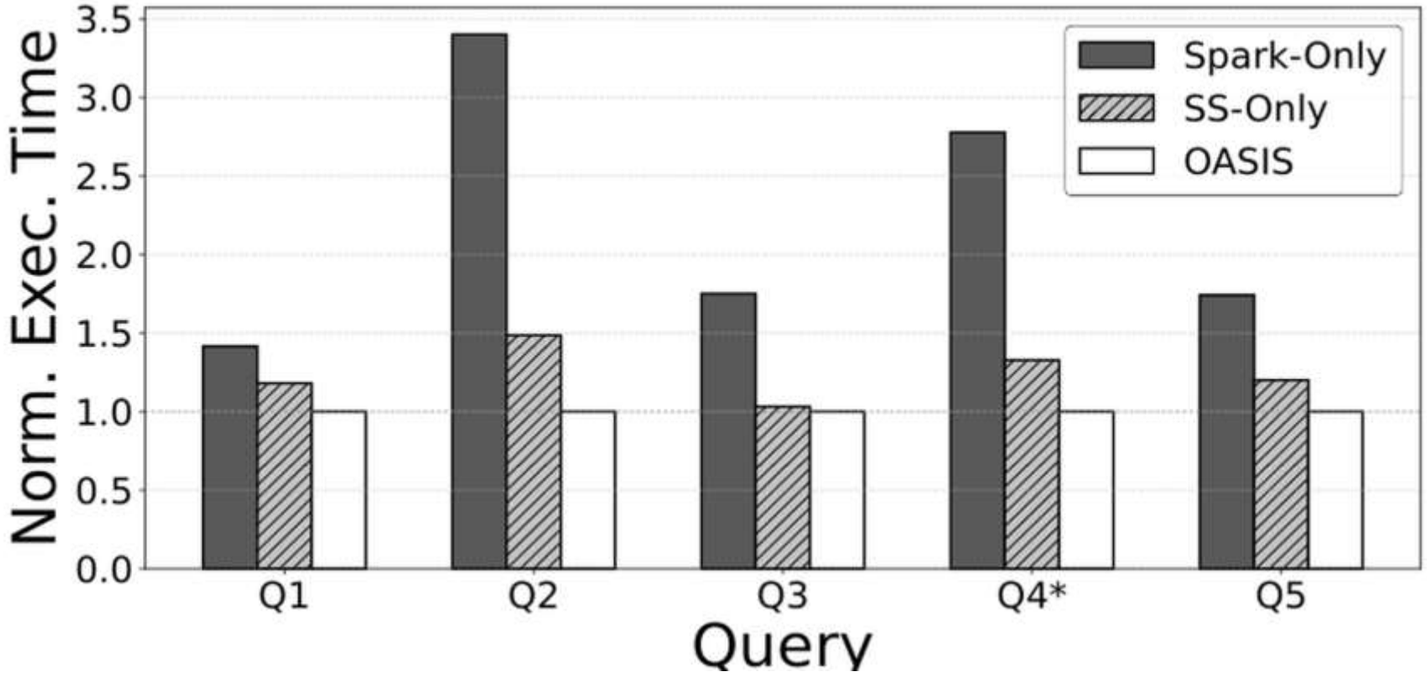


- Skyhook takes 1.11–1.25× the OASIS execution time across applicable queries (OASIS Fig. 3).
- MinIO Select [5] cannot execute several scientific-type or expression workloads natively.

SOGANG UNIVERSITY

DISCOS

Effect of Hierarchical Execution



- For Q1–Q3, both use the same DuckDB [18] executor; the placement policy is what changes.
- In OASIS Fig. 4, SS-Only takes up to approximately 1.48× the OASIS query time.

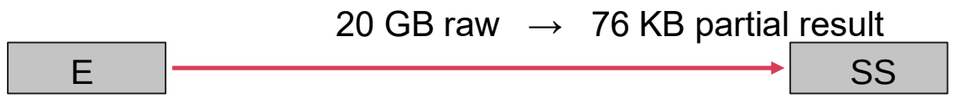
SOGANG UNIVERSITY

Enclosure–Server Data Movement

Measured bytes transferred from Enclosure to Storage Server

Query	SS-Only	OASIS
Q1 Laghos	**20 GB**	**76 KB**
Q2 DeepWater	**13.19 GB**	**52.89 MB**
Q3 DeepWater	**30 GB**	**228 B**

Q1 transfer path



- Q1 E→SS movement: 20 GB (SS-Only) → 76 KB (OASIS), a 99.998% reduction (Table VII).
- Both approaches can return small results to Spark; OASIS additionally eliminates storage-internal traffic.

SOGANG UNIVERSITY

DISCOS

SODA vs. Selective-Only Pushdown

Same enclosure and server executors — different operator placement policy

Query	SO-PD bytes	SODA bytes	SO-PD latency	SODA latency
Q1	450 KB	**76 KB**	402.0 s	369.4 s
Q2	52.89 MB	52.89 MB	56.0 s	57.5 s
Q3	9.7 GB	**228 B**	369.0 s	359.4 s

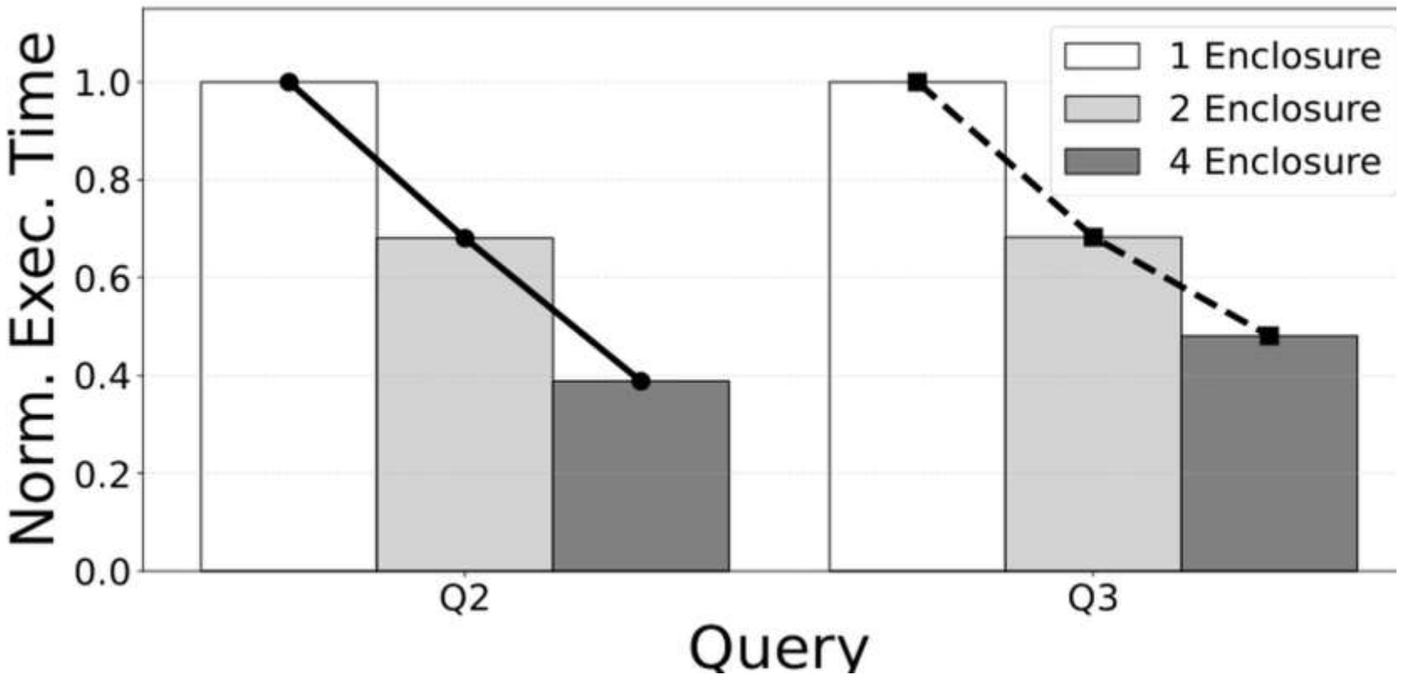


- SO-PD represents selective/fixed-boundary methods such as AnalyzeThis [13] and Skytether [14].
- For Q3, SODA cuts E→SS traffic from 9.7 GB to 228 B, with 2.6% less latency (OASIS Table VIII).

SOGANG UNIVERSITY

DISCOS

Multiple-Enclosure Scalability



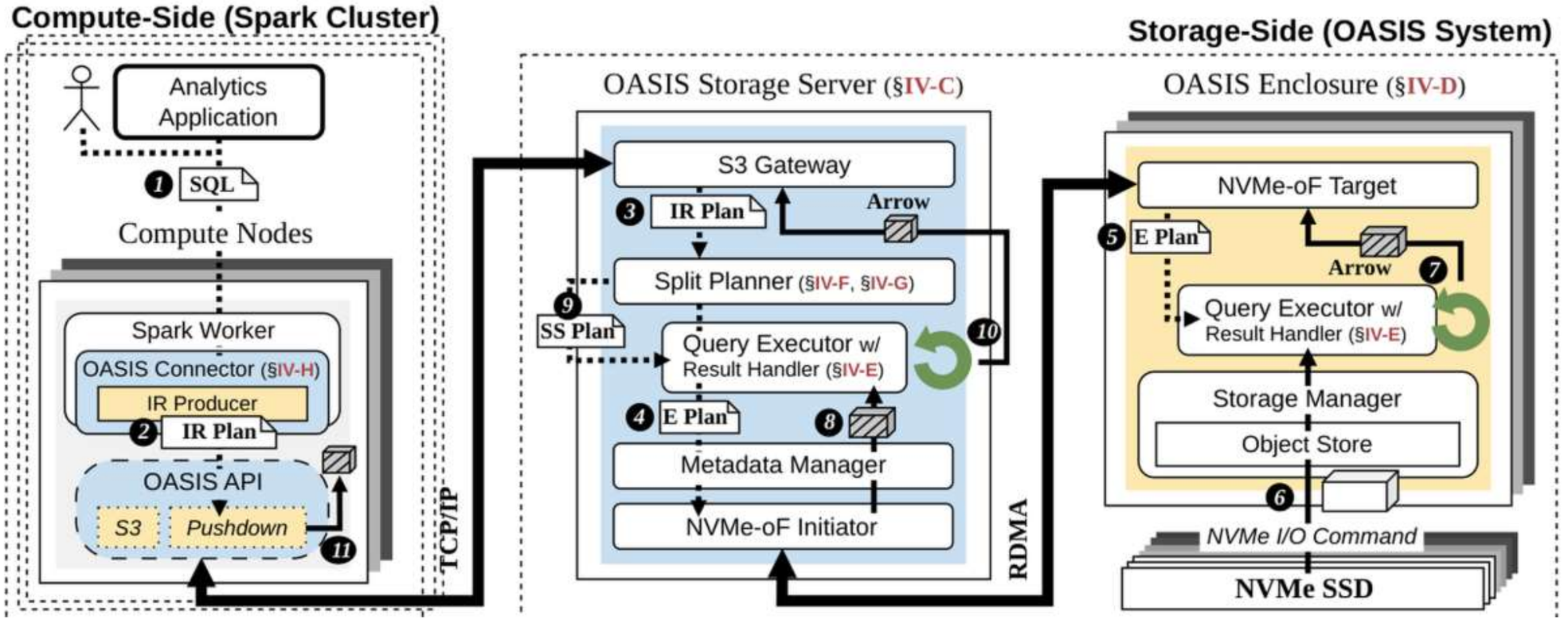
Scaling configuration: Sidewinder SW-100 boards over PCIe QDMA (separate from the base testbed).

- With four enclosures, normalized latency is 0.39 for Q2 (Filter only) and 0.48 for Q3 (Aggregate).
- Q3 scales less because server-side final aggregation requires shared coordination.

SOGANG UNIVERSITY

DISCOS

Conclusion



- OASIS distributes offloaded SQL across Enclosure and Storage Server.
- SODA chooses the boundary under semantic and memory constraints.
- 99.998% less E→SS traffic vs. SS-Only; Skyhook/OASIS time: 1.11–1.25×.

SOGANG UNIVERSITY

DISCOS

Thank you!

Contact

- Soon Hwang / soonhw@sogang.ac.kr
- Youngjae Kim / youkim@sogang.ac.kr

  https://sites.google.com/site/youkim/home

- Data-Intensive AI Computing & Systems Laboratory

  https://discos.sogang.ac.kr/

Scan for more from our lab

discos.sogang.ac.kr/publications-kim

SOGANG UNIVERSITY

References

[1] M. Folk et al., “An Overview of the HDF5 Technology Suite,” EDBT/ICDT AD, 2011.

[2] UCAR/Unidata, “Network Common Data Form (NetCDF),” 2026.

[3] Apache Parquet, “File Format Documentation,” https://parquet.apache.org/docs/file-format/, 2026.

[4] R. Hunt, “S3 Select and Glacier Select,” AWS, 2017.

[5] MinIO, “mc sql Command: Run SQL Queries on Object Data,” 2025.

[6] J. Chakraborty et al., “Skyhook: Towards an Arrow-Native Storage System,” CCGrid, 2022.

[7] J. Menon, “Computational Storage: Directions at Fungible,” SNIA SDC, 2021.

[8] Los Alamos National Laboratory, “Laghos Sample Dataset,” 2024.

[9] Los Alamos National Laboratory, “Deep Water Impact Dataset,” 2024.

[10] L. Canali, “Apache Spark for High Energy Physics,” benchmark repository, 2024.

[11] Los Alamos National Laboratory, “C2 VPIC Sample Dataset,” 2022.

[12] A. Huebl et al., “openPMD-api,” 2018.

[13] H. Sim et al., “AnalyzeThis: An Analysis Workflow-Aware Storage System,” SC, 2015.

[14] A. Montana et al., “A Moveable Beast: Partitioning Data and Compute,” 2023.

[15] Y. Yang et al., “FlexPushdownDB,” VLDB Journal, 2024.

[16] Substrait Project, “Substrait,” 2024.

[17] Apache Software Foundation, “Apache Arrow,” 2024.

[18] DuckDB Foundation, “DuckDB,” 2024.

[19] CMS Collaboration, “SingleMu Primary Dataset,” CERN Open Data, 2017.

[20] I. Jo et al., “YourSQL: A High-Performance Database System Leveraging In-Storage Computing,” PVLDB, 2016.