Specialization, Disaggregation and Streaming: the new frontiers for data processing

Gustavo Alonso

Systems Group

Department of Computer Science

ETH Zurich, Switzerland

Specialization

General purpose computing

Slow improvements lead to specialization

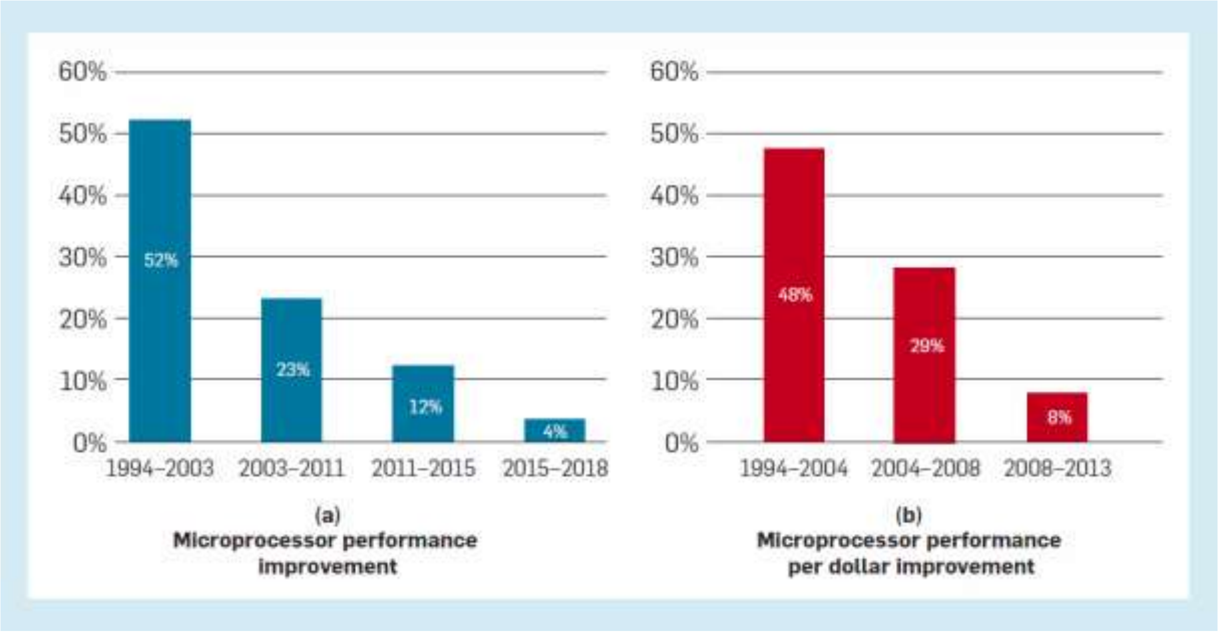
COMMUNICATIONS OF THE ACM

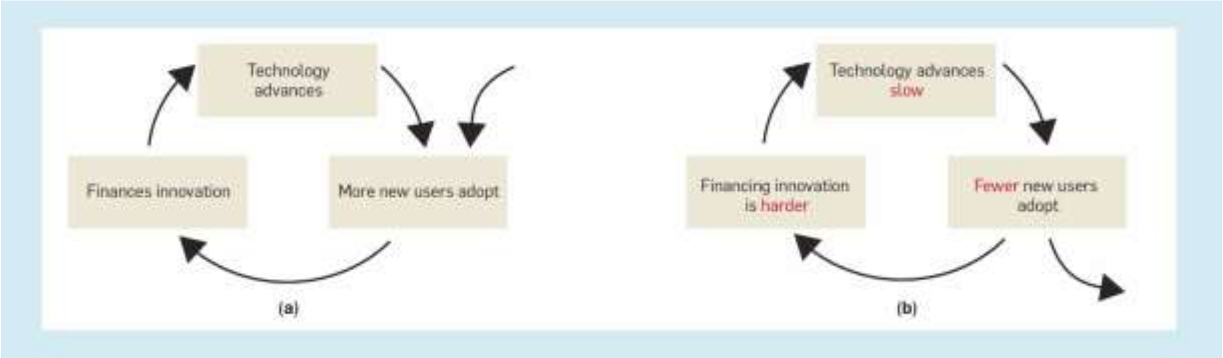
HOME | CURRENT ISSUE | NEWS | BLOGS | OPINION | RESEARCH | PRACTICE | CAREERS | ARCHIVE | VIDEOS

Home / Magazine Archive / March 2021 (Vol. 64, No. 3) / The Decline of Computers as a General Purpose Technology / Full Text

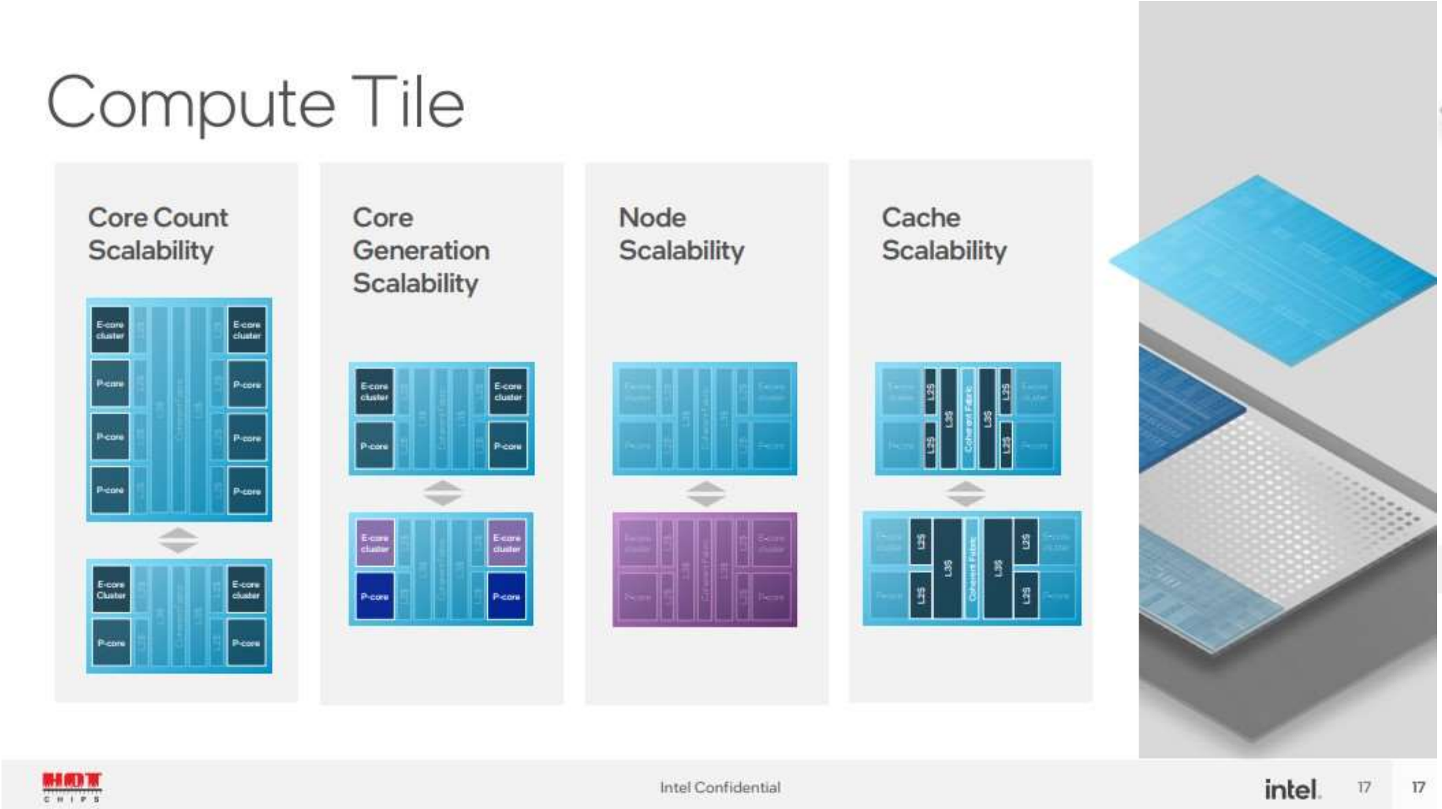
CONTRIBUTED ARTICLES

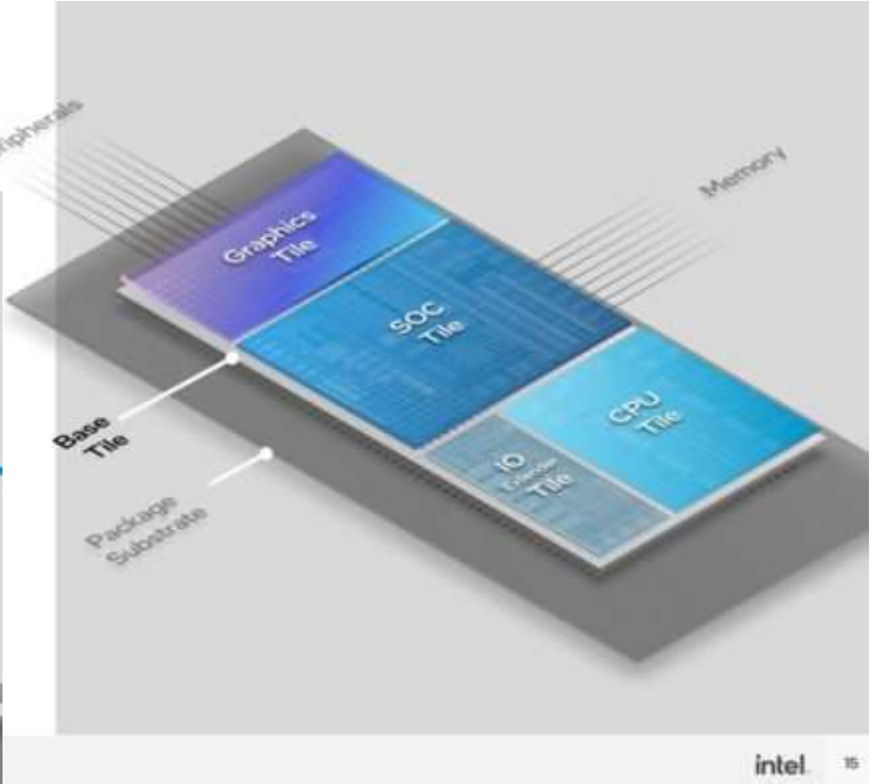
The Decline of Computers as a General Purpose Technology





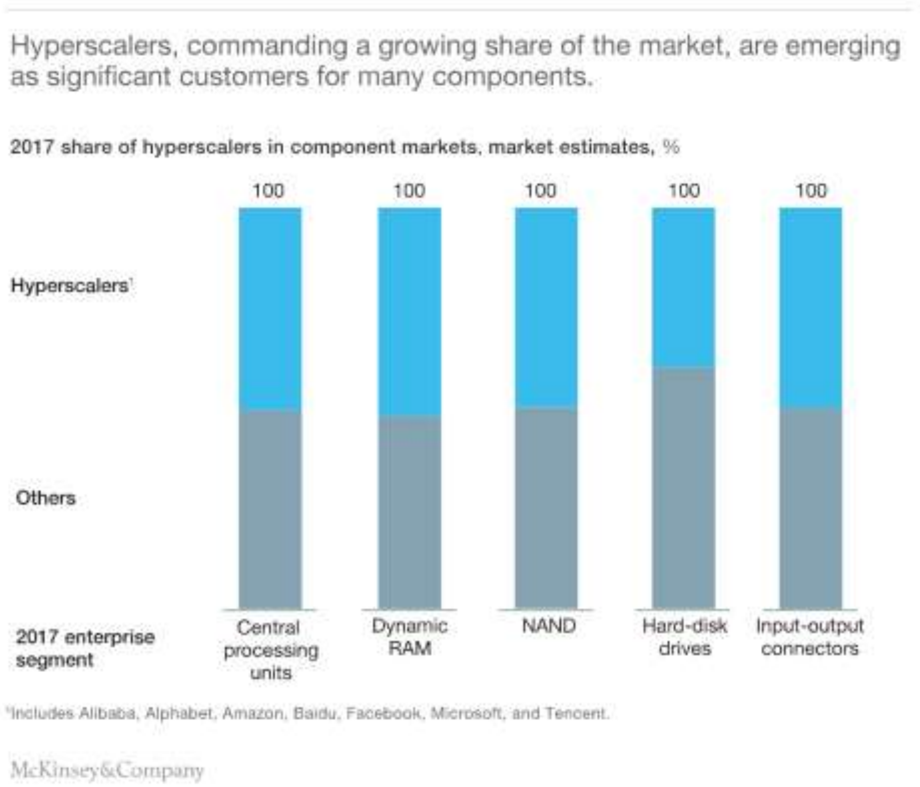
Processor specialization





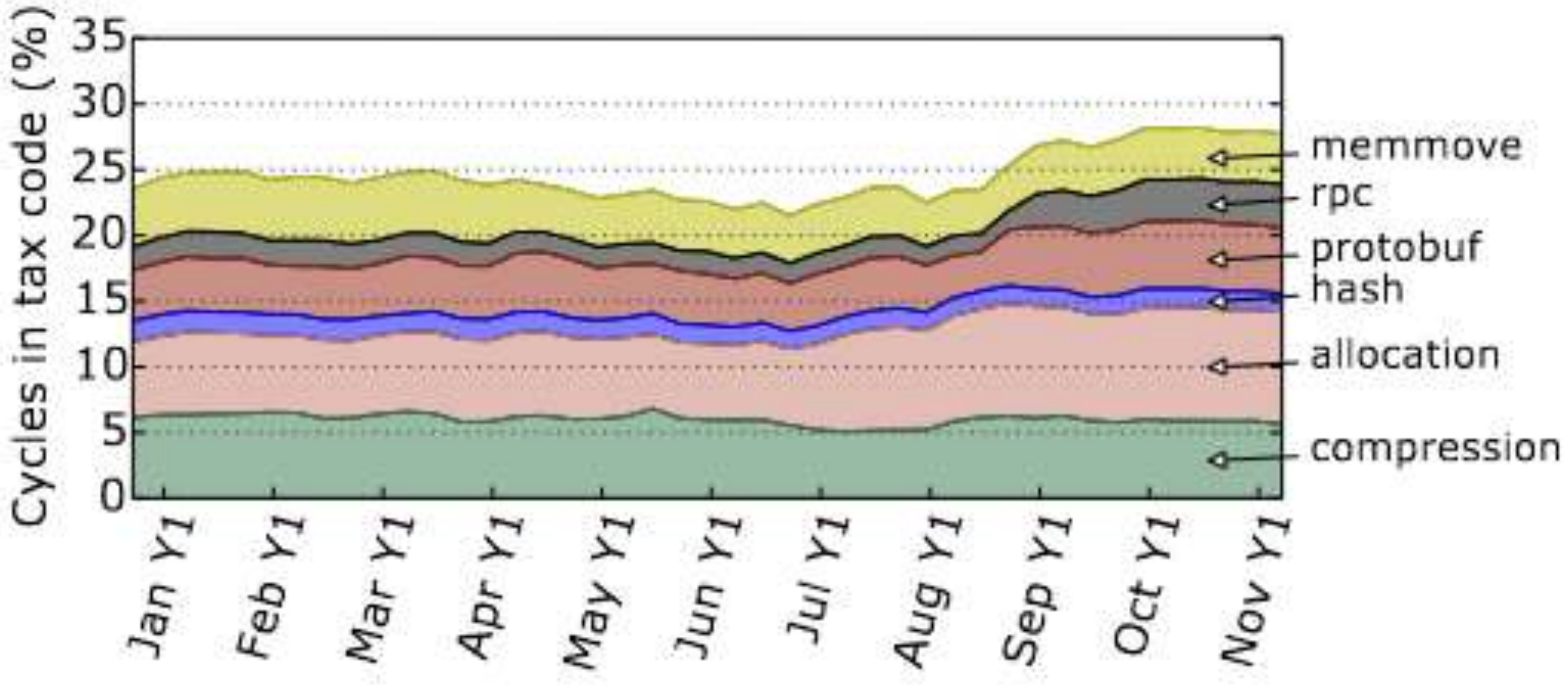
Driving specialization

- The cloud is the big game changer:
  - New business model
  - Economies of scale
  - Very large workloads
- Every hyper scaler is its own “Killer App”
  - The scale makes many things feasible
  - The gains have a very large multiplier



https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/how-high-tech-suppliers-are-responding-to-the-hyperscaler-opportunity

The Data Center Tax

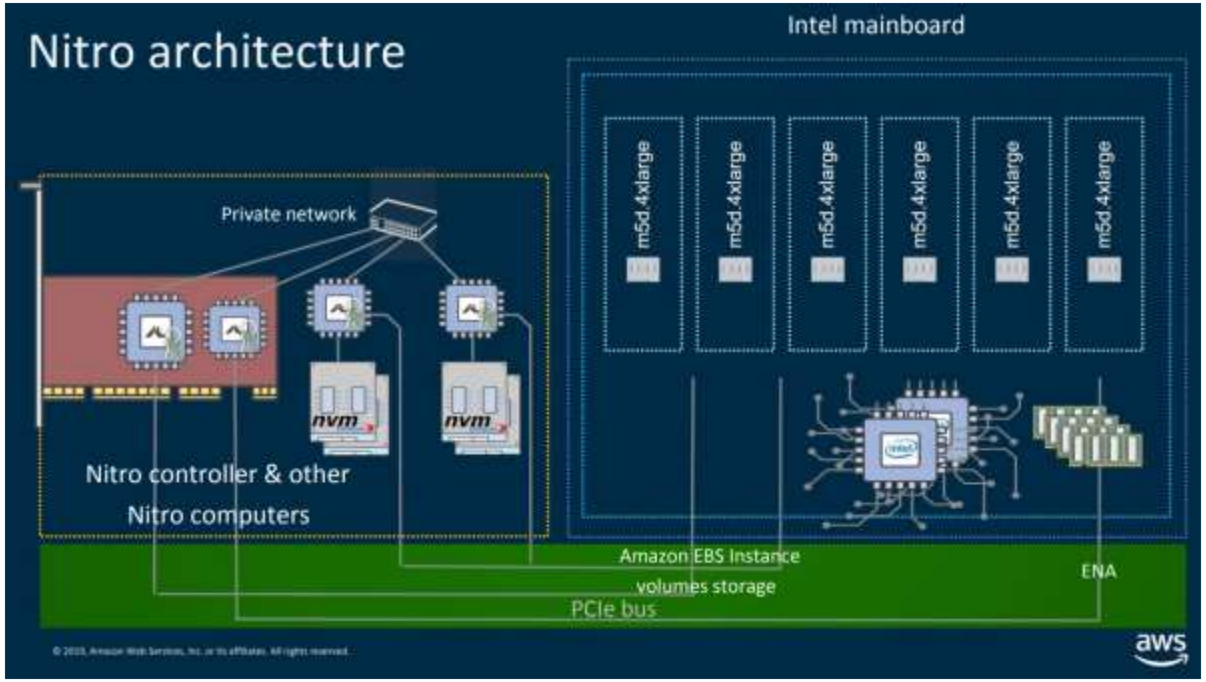


Profiling a warehouse-scale computer, ISCA 2015

Consequences:

Everything that is demanding enough and common enough is moving to dedicated accelerators

Running virtual machines



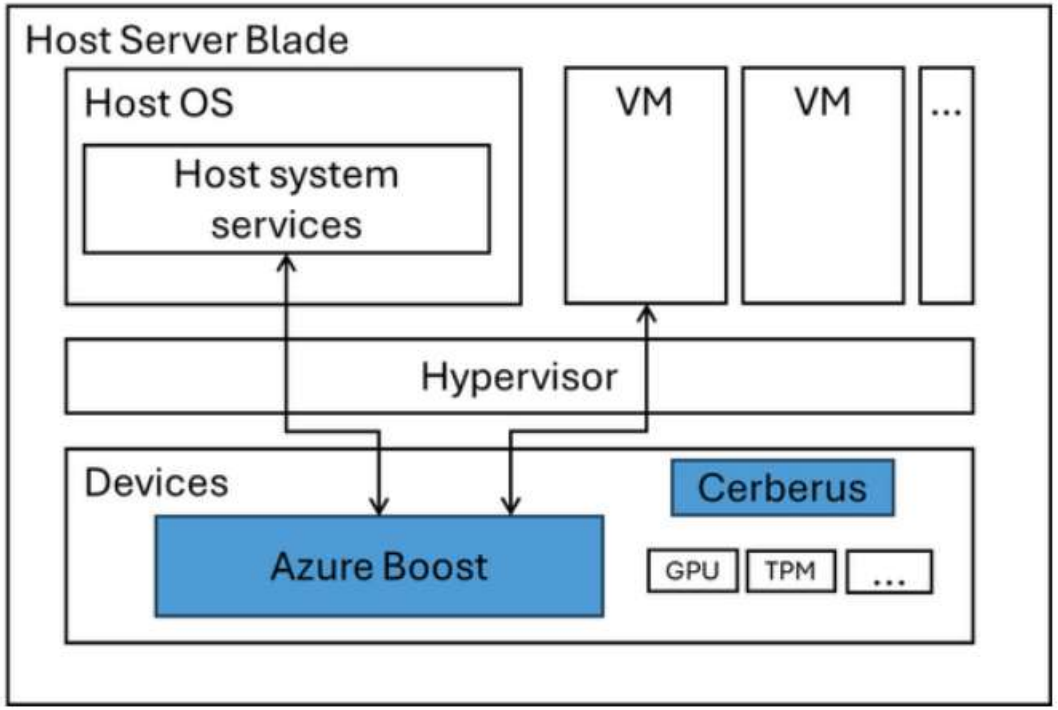
Microsoft Azure Boost

Security architecture components

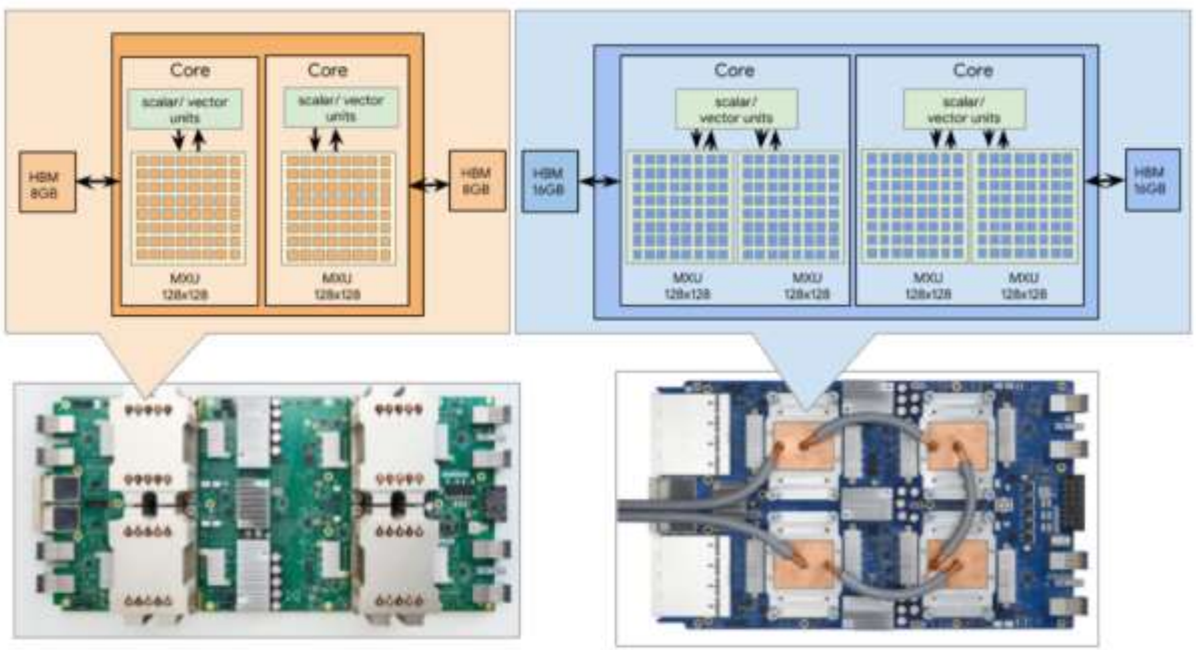
Designed to enhance Azure workload security, Azure Boost includes the following security components:

- **An independent hardware root of trust** - Cerberus fulfils NIST 800-193 certification.
- **Azure Boost system on chip (SoC)** – dedicated, Linux based system conducting management operations for the control plane.
- **Configurable field-programable gate array (FPGA)** – programable network and storage acceleration capabilities for the data plane.

Azure Boost SoCs pair with each host and work in tandem to create a more secure hosting infrastructure.

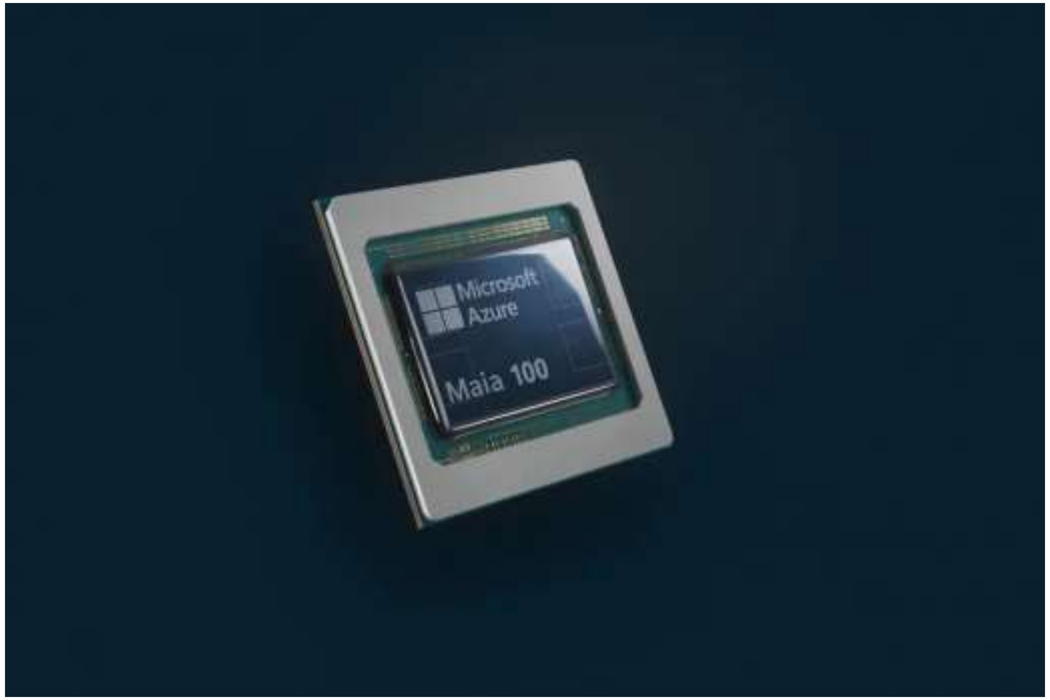


Accelerating ML/AI



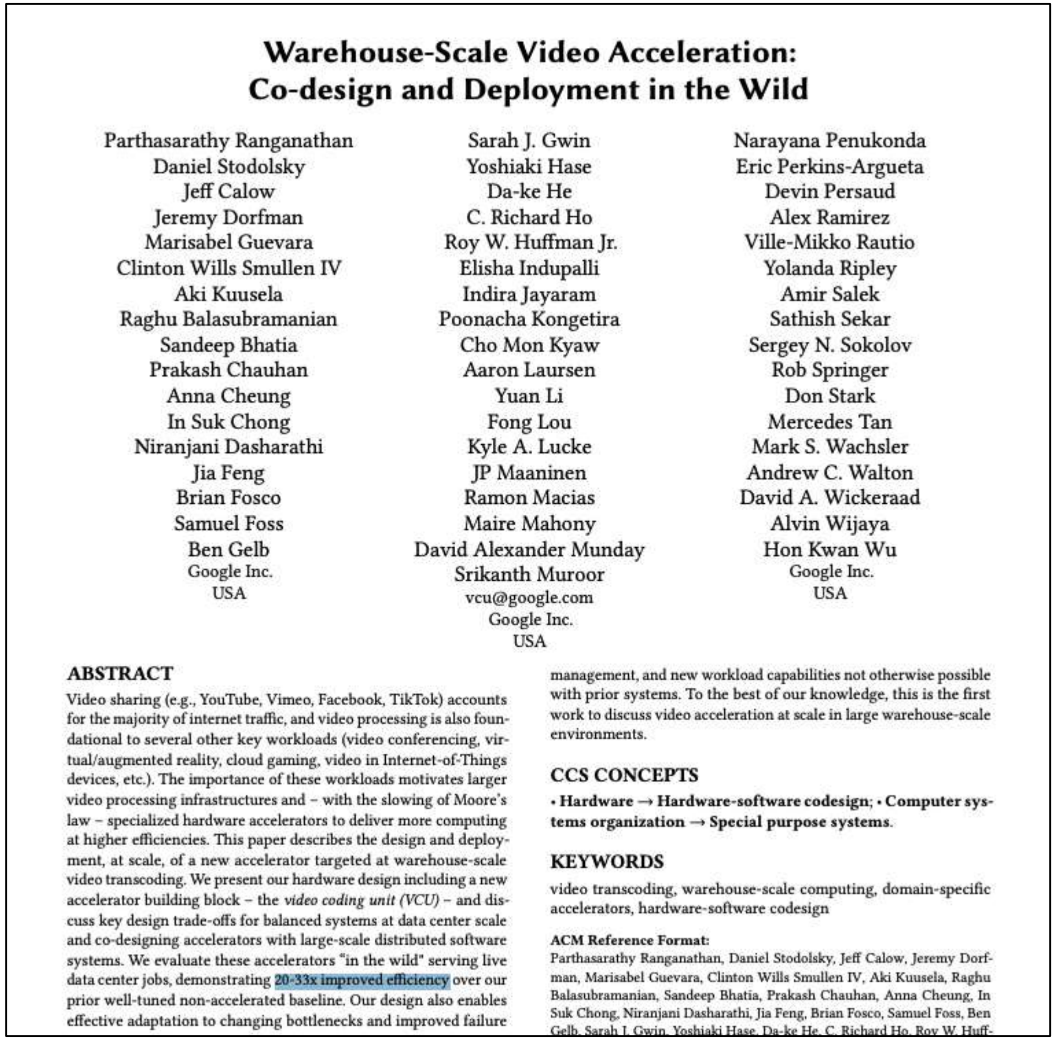
TPU v2 - 4 chips, 2 cores per chip

TPU v3 - 4 chips, 2 cores per chip



Accelerating video processing (VCU)

- 500+ hrs of video are uploaded to YouTube every minute!
- Need to transcode each video to diff formats & resolutions for diff devices
- VCU achieves 20-33x better compute-efficiency vs. optimized CPU baseline
- Use-cases:
  - video conferencing, livestreaming
  - virtual/augmented reality
  - cloud gaming
  - video in IoT devices
- See paper at ASPLOS'21: https://research.google/pubs/pub50300/



Warehouse-Scale Video Acceleration: Co-design and Deployment in the Wild

Parthasarathy Ranganathan, Daniel Stodolsky, Jeff Calow, Jeremy Dorfman, Marisabel Guevara, Clinton Wills Smullen IV, Aki Kuusela, Raghu Balasubramanian, Sandeep Bhatia, Prakash Chauhan, Anna Cheung, In Suk Chong, Niranjani Dasharathi, Jia Feng, Brian Fosco, Samuel Foss, Ben Gelb
Google Inc.
USA

Sarah J. Gwin, Yoshiaki Hase, Da-ke He, C. Richard Ho, Roy W. Huffman Jr., Elisha Indupalli, Indira Jayaram, Poonacha Kongetira, Cho Mon Kyaw, Aaron Laursen, Yuan Li, Fong Lou, Kyle A. Lucke, JP Maaninen, Ramon Macias, Maire Mahony, David Alexander Munday, Srikanth Muroor
vcu@google.com
Google Inc.
USA

Narayana Penukonda, Eric Perkins-Argueta, Devin Persaud, Alex Ramirez, Ville-Mikko Rautio, Yolanda Ripley, Amir Salek, Sathish Sekar, Sergey N. Sokolov, Rob Springer, Don Stark, Mercedes Tan, Mark S. Wachsler, Andrew C. Walton, David A. Wickeraad, Alvin Wijaya, Hon Kwan Wu
Google Inc.
USA

ABSTRACT

Video sharing (e.g., YouTube, Vimeo, Facebook, TikTok) accounts for the majority of internet traffic, and video processing is also foundational to several other key workloads (video conferencing, virtual/augmented reality, cloud gaming, video in Internet-of-Things devices, etc.). The importance of these workloads motivates larger video processing infrastructures and – with the slowing of Moore's law – specialized hardware accelerators to deliver more computing at higher efficiencies. This paper describes the design and deployment, at scale, of a new accelerator targeted at warehouse-scale video transcoding. We present our hardware design including a new accelerator building block – the *video coding unit (VCU)* – and discuss key design trade-offs for balanced systems at data center scale and co-designing accelerators with large-scale distributed software systems. We evaluate these accelerators "in the wild" serving live data center jobs, demonstrating 20-33x improved efficiency over our prior well-tuned non-accelerated baseline. Our design also enables effective adaptation to changing bottlenecks and improved failure management, and new workload capabilities not otherwise possible with prior systems. To the best of our knowledge, this is the first work to discuss video acceleration at scale in large warehouse-scale environments.

CCS CONCEPTS

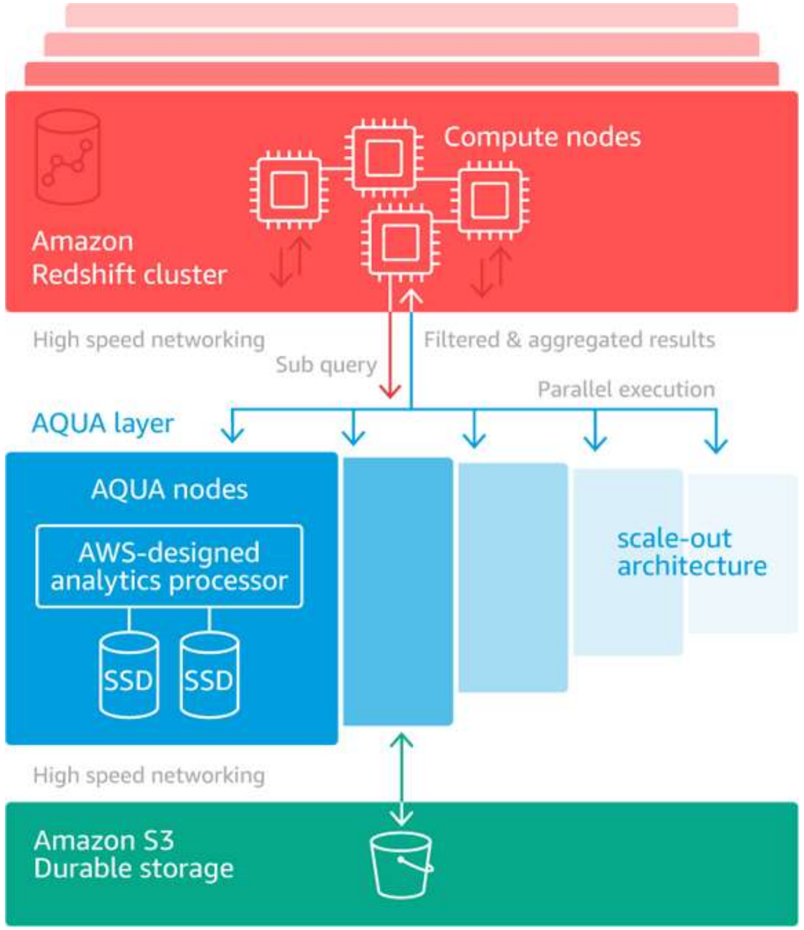
• Hardware → Hardware-software codesign; • Computer systems organization → Special purpose systems.

KEYWORDS

video transcoding, warehouse-scale computing, domain-specific accelerators, hardware-software codesign

ACM Reference Format:
Parthasarathy Ranganathan, Daniel Stodolsky, Jeff Calow, Jeremy Dorfman, Marisabel Guevara, Clinton Wills Smullen IV, Aki Kuusela, Raghu Balasubramanian, Sandeep Bhatia, Prakash Chauhan, Anna Cheung, In Suk Chong, Niranjani Dasharathi, Jia Feng, Brian Fosco, Samuel Foss, Ben Gelb, Sarah J. Gwin, Yoshiaki Hase, Da-ke He, C. Richard Ho, Roy W. Huff-

Cloud caches (Amazon Aqua)



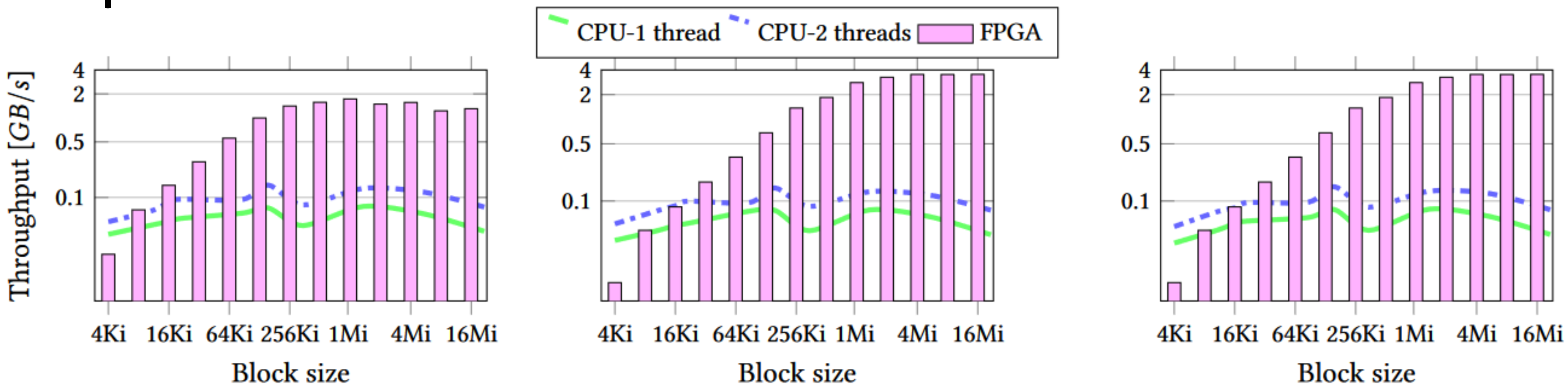
"AQUA is designed to deliver up to 10X performance on queries that perform large scans, aggregates, and filtering with LIKE and SIMILAR_TO predicates. Over time we expect to add support for additional queries."

https://aws.amazon.com/blogs/aws/new-aqua-advanced-query-accelerator-for-amazon-redshift/

Data Compression (Microsoft Zipline/Corsica)

https://azure.microsoft.com/en-us/blog/improved-cloud-service-performance-through-asic-acceleration/

Example



(a) Compression + AES-CBC (b) Compression + AES-CTR (c) Compression + AES-ECB

Figure 12: Full pipeline - with 1 and 2 threads on CPU vs. FPGA design. Note the logarithmic scale of the y axis.

Hardware Acceleration of Compression and Encryption in SAP HANA

Monica Chiosa*
ETH Zurich
monica.chiosa@inf.ethz.ch

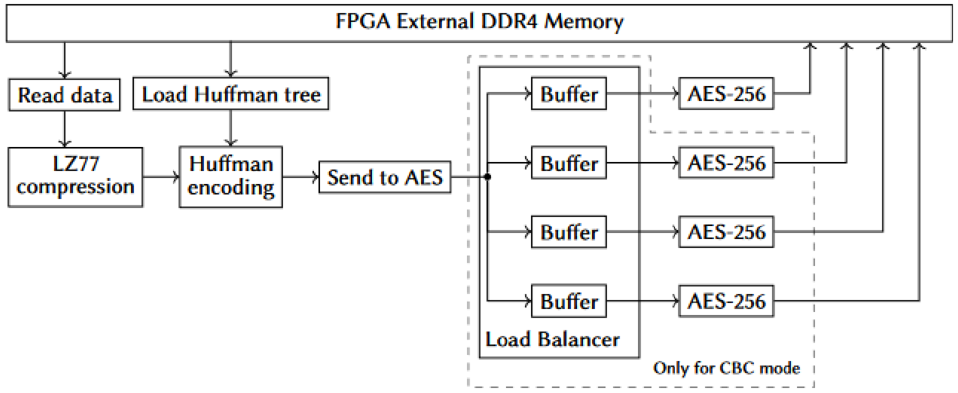
Fabio Maschi*
ETH Zurich
fabio.maschi@inf.ethz.ch

Ingo Müller
ETH Zurich
ingo.mueller@inf.ethz.ch

Gustavo Alonso
ETH Zurich
alonso@inf.ethz.ch

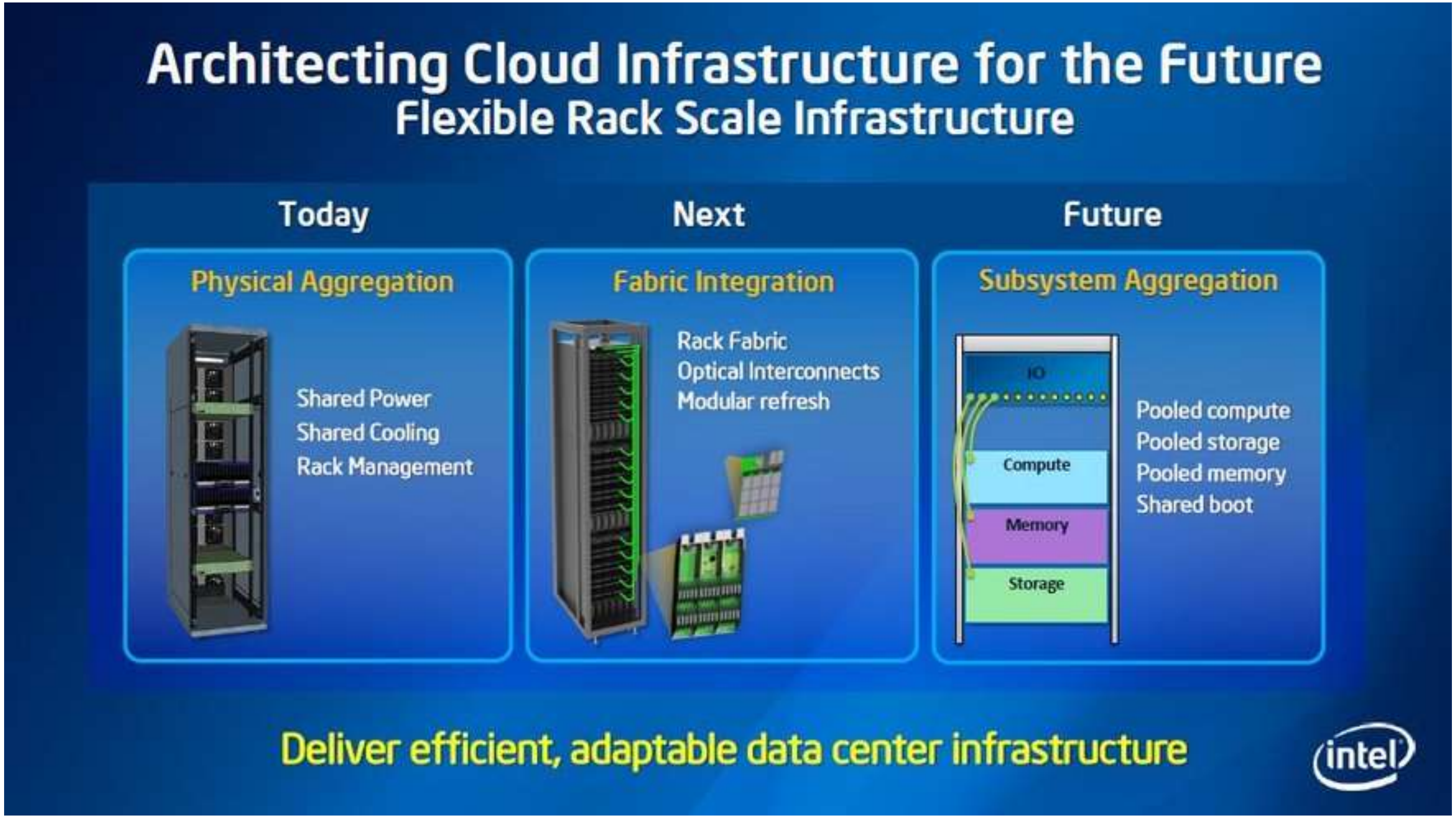
Norman May
SAP SE
norman.may@sap.com

VLDB 2022



Disaggregation

Trend towards disaggregation

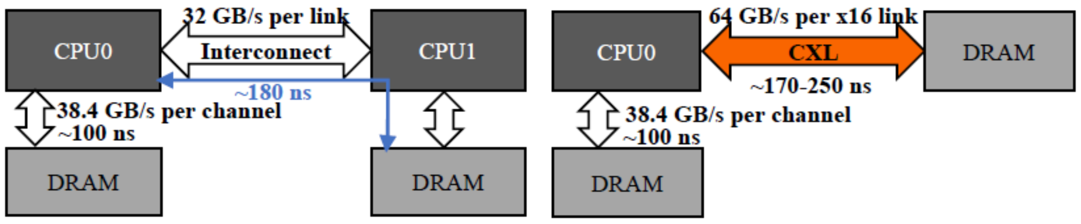


The future of accelerators

TPP: Transparent Page Placement for CXL-Enabled Tiered Memory

Hasan Al Maruf*, Hao Wang†, Abhishek Dhanotia†, Johannes Weiner†, Niket Agarwal†, Pallab Bhattacharya†, Chris Petersen†, Mosharaf Chowdhury*, Shobhit Kanaujia†, Prakash Chauhan†

University of Michigan* Meta Inc.†

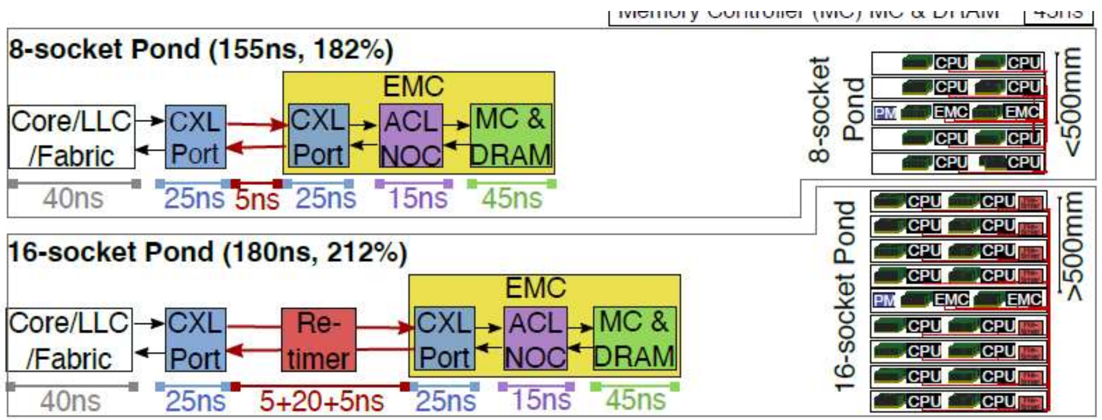


(a) *Without CXL* (b) *With CXL*

Pond: CXL-Based Memory Pooling Systems for Cloud Platforms

Huaicheng Li†, Daniel S. Berger*‡, Stanko Novakovic*, Lisa Hsu*, Dan Ernst*, Pantea Zardoshti*, Monish Shah*, Samir Rajadnya*, Scott Lee*, Ishwar Agarwal*, Mark D. Hill*°, Marcus Fontoura*, Ricardo Bianchini*

†Virginia Tech and CMU *Microsoft Azure ‡University of Washington °University of Wisconsin-Madison



Example

Farview: Disaggregated Memory with Operator Off-loading for Database Engines

Dario Korolija
dario.korolija@inf.ethz.ch
ETH Zurich
Switzerland

Dimitrios Koutsoukos
dkoutsou@inf.ethz.ch
ETH Zurich
Switzerland

Kimberly Keeton*
kimberlykeeton@acm.org
Hewlett Packard Labs
USA

Konstantin Taranov
konstantin.taranov@inf.ethz.ch
ETH Zurich
Switzerland

Dejan Milojičić
dejan.milojicic@hpe.com
Hewlett Packard Labs
USA

Gustavo Alonso
alonso@inf.ethz.ch
ETH Zurich
Switzerland

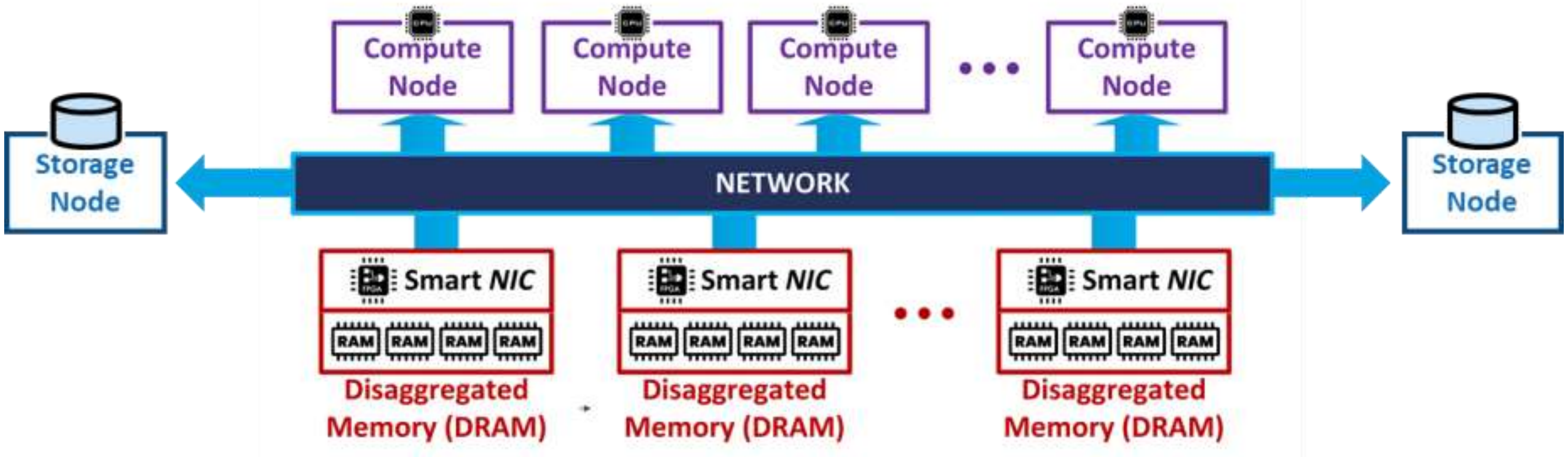
ABSTRACT

Cloud deployments disaggregate storage from compute, providing more flexibility to both the storage and compute layers. In this paper,

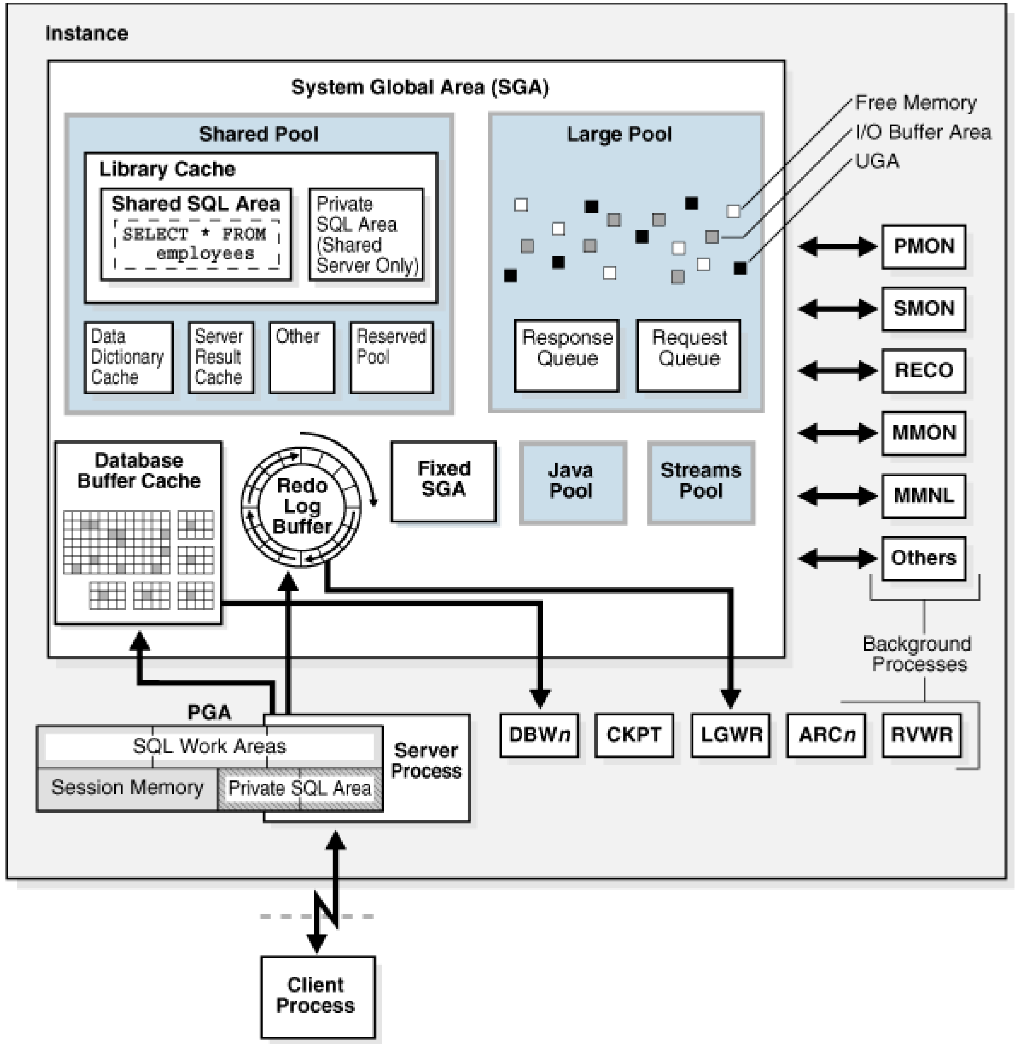
computing; and the amount of data to be processed keeps growing while DRAM capacity does not.

Optimized query plans typically push down selection and pro-

Disaggregated Memory (Farview)



Use case

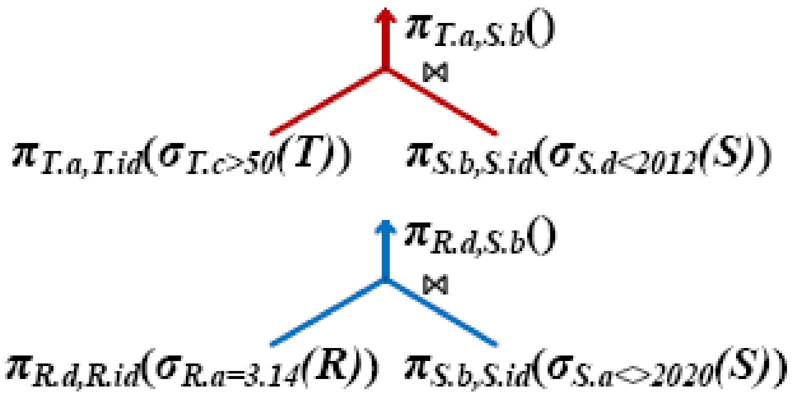


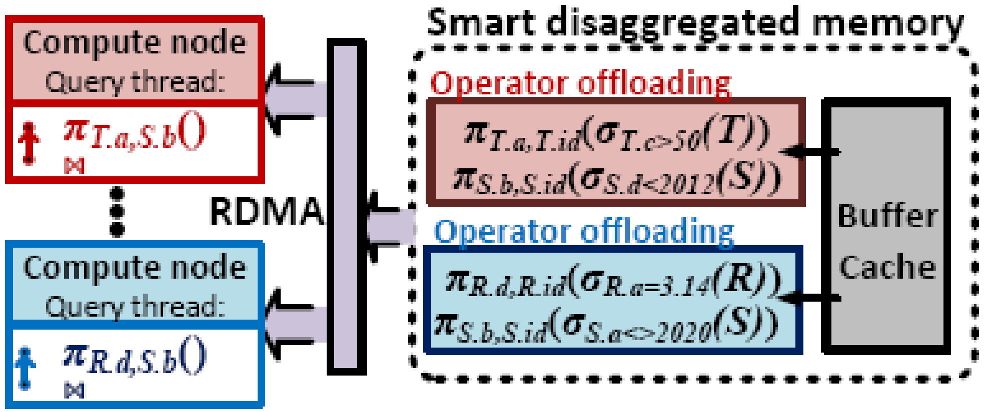
https://docs.oracle.com/cd/E11882_01/server.112/e40540/process.htm#CNCPT902

Smart Disaggregated Memory (Farview)

```
SELECT T.a, S.b
FROM T, S
WHERE T.id = S.id
AND T.c > 50 AND S.d < 2012;
```

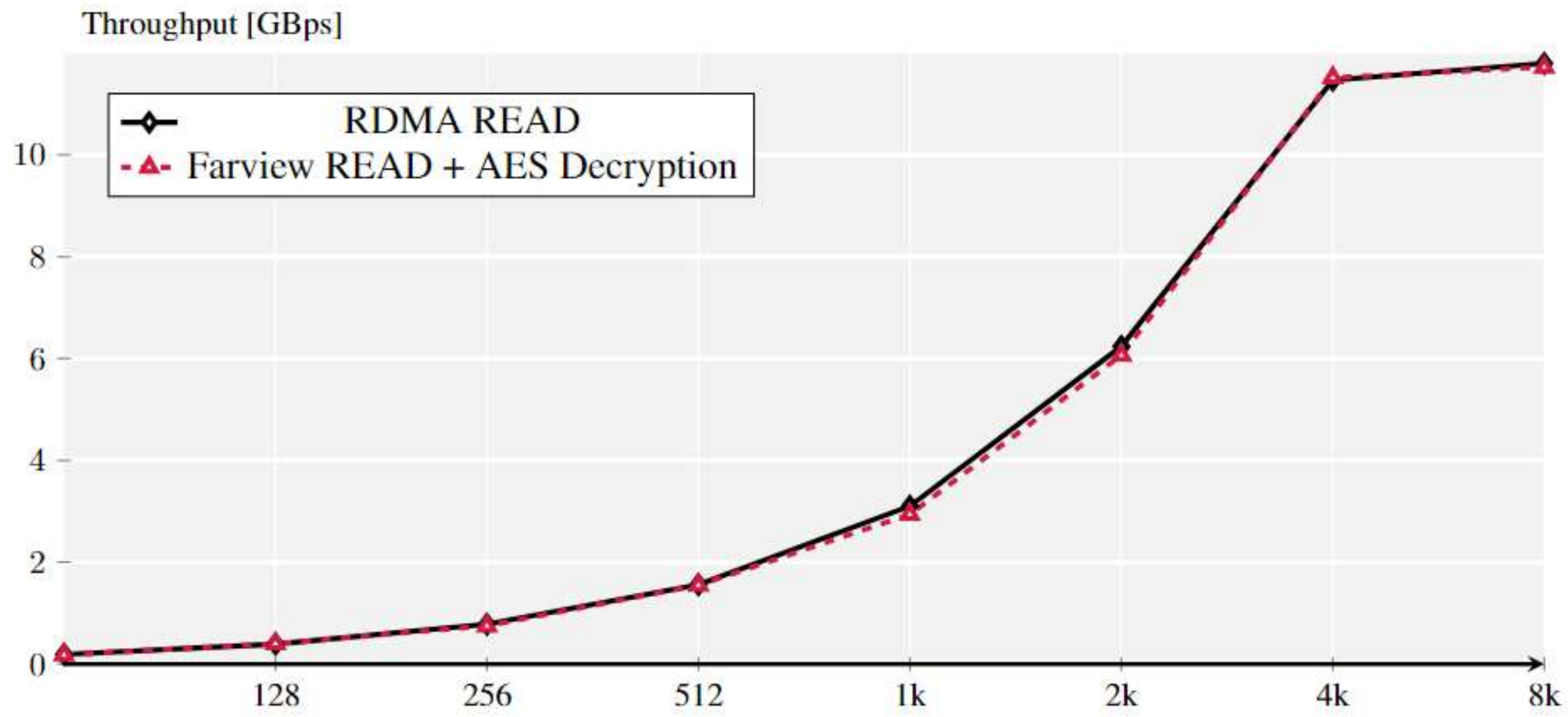
```
SELECT R.d, S.b
FROM R, S
WHERE R.id = S.id
AND R.a = 3.14 AND S.a <> 2012;
```





The goal is to reduce the amount of memory that needs to be allocated in computing nodes by using a common buffer cache in DRAM available through the network via RDMA

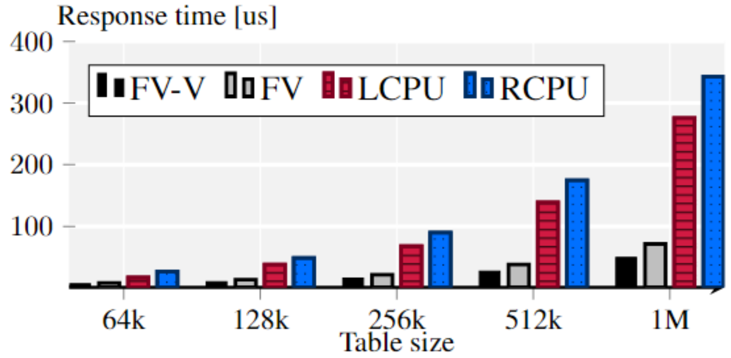
Farview: overhead



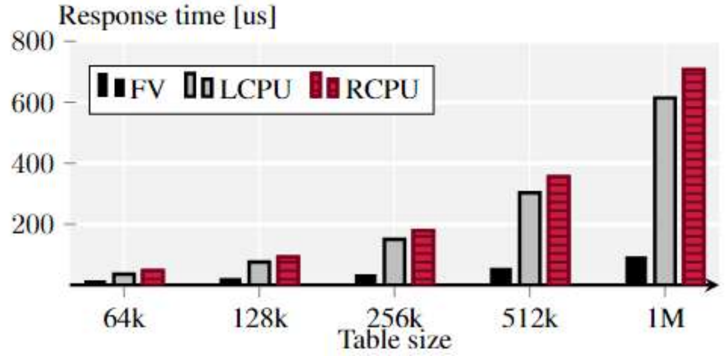
Evaluation

- Farview compared to two baselines
  - Buffer cache implemented in local memory with processing on the local CPU
  - Remote buffer cache, without FPGA, implemented in a remote machine accessed through commercial Mellanox NIC via RDMA

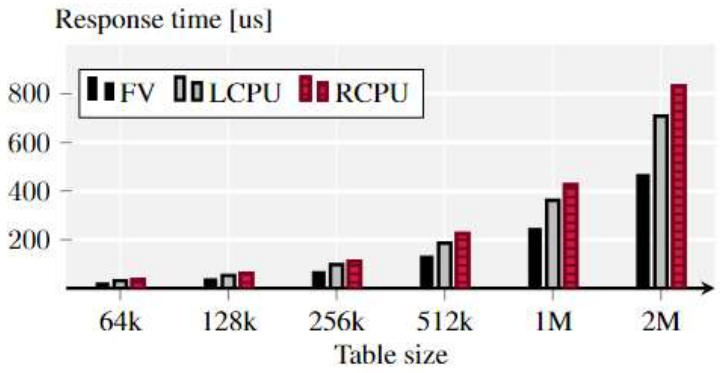
Response times for selection queries with 50% selectivity:



Response time comparisons for a group by query with aggregation:



Response time comparisons for a distinct query with 6 concurrent clients:



Generalizing the idea

- The same can be done on:
  - Remote storage (cloud storage layer)
  - Local disks
  - Local memory
  - Remote memory through a smart NIC
  - Caching layer similar to AQUA
- One can even think of a common interface for all these systems to make the operator offloading completely transparent

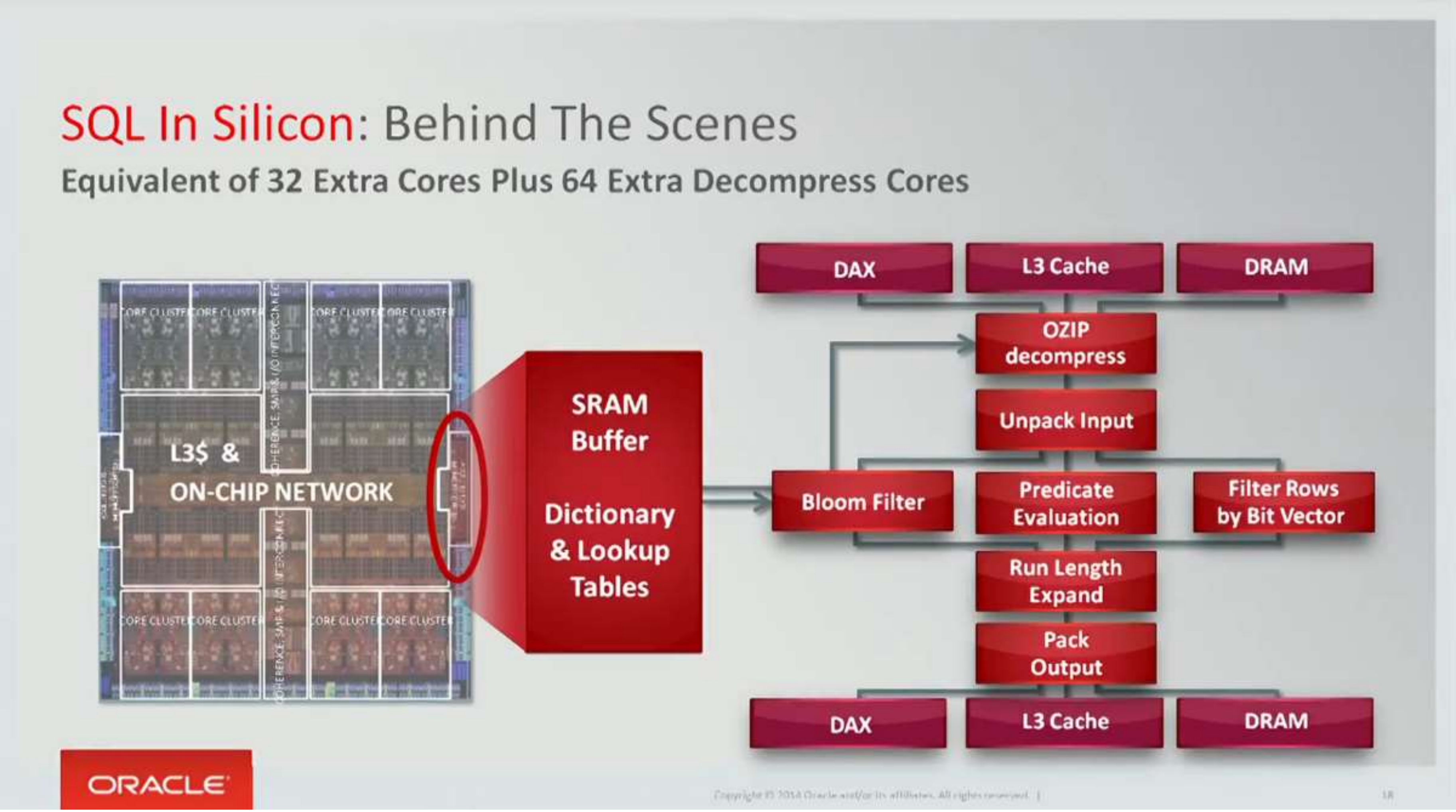
Streaming

Near Data Processing

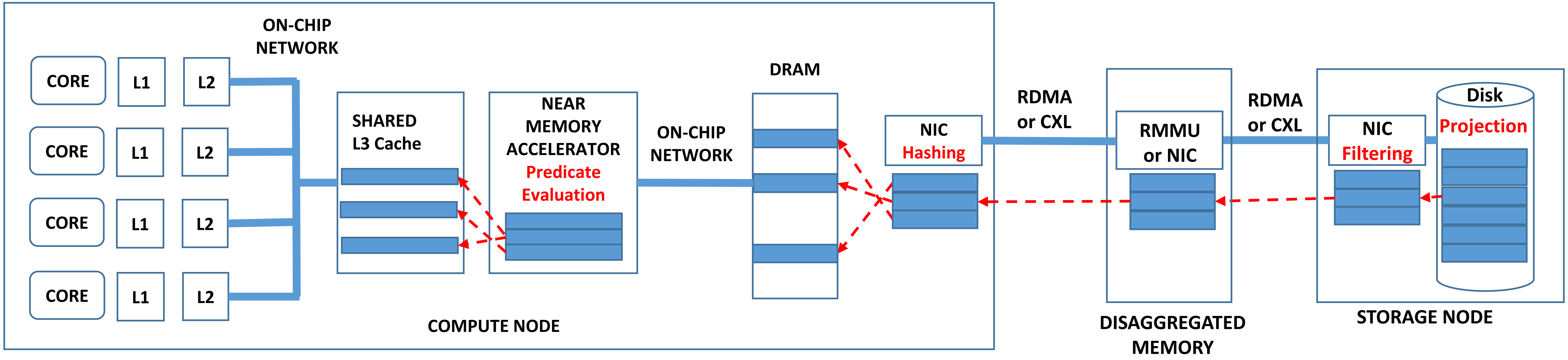
- Given the high cost of data movement, storage must be:
  - Distributed – for capacity reasons
  - Smart – to minimize data movement
  - Efficient – avoid unnecessary overheads

- Ideally, storage of any kind (DRAM, local disk, network attached, object repositories, storage layers, data lakes, etc.) should enable processing of the data at or near the storage itself
- This has to be done in a streaming fashion to avoid adding latency

Processing on memory streams



Active pipelines on the data path



Joint work with Alberto Lerner (Data Flow Architectures for Data Processing on Modern Hardware, ICDE 2024)

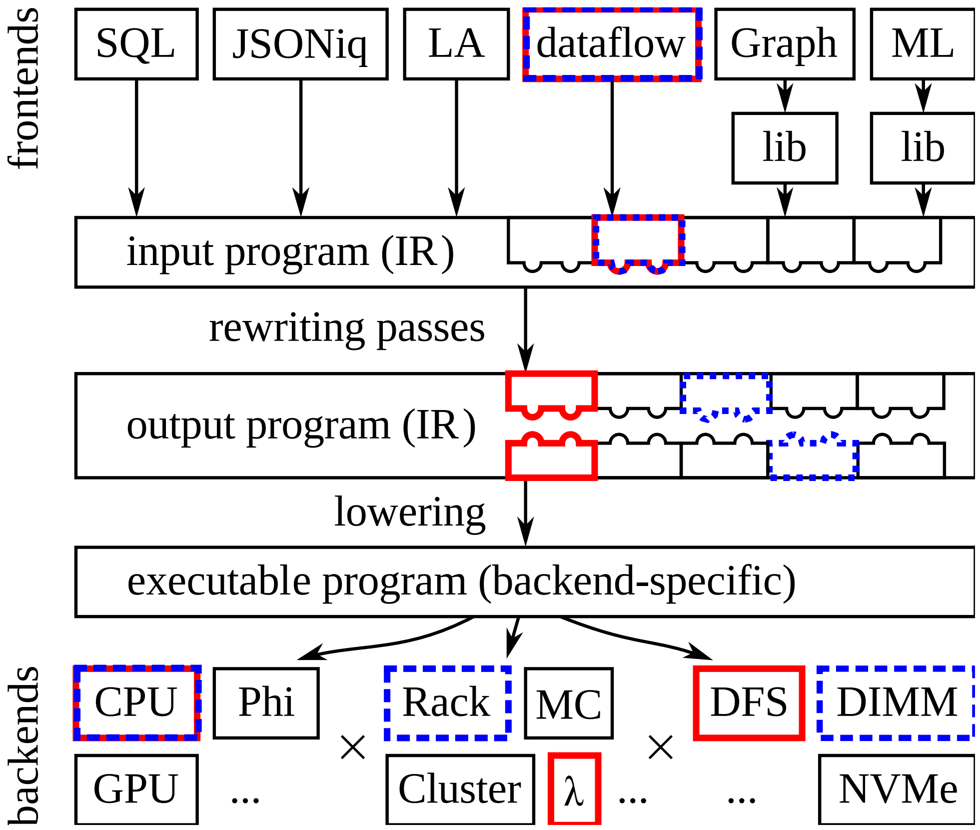
Maximus

Modularis: Modular Relational Analytics over Heterogeneous Distributed Platforms. VLDB 2021

Program once, run everywhere

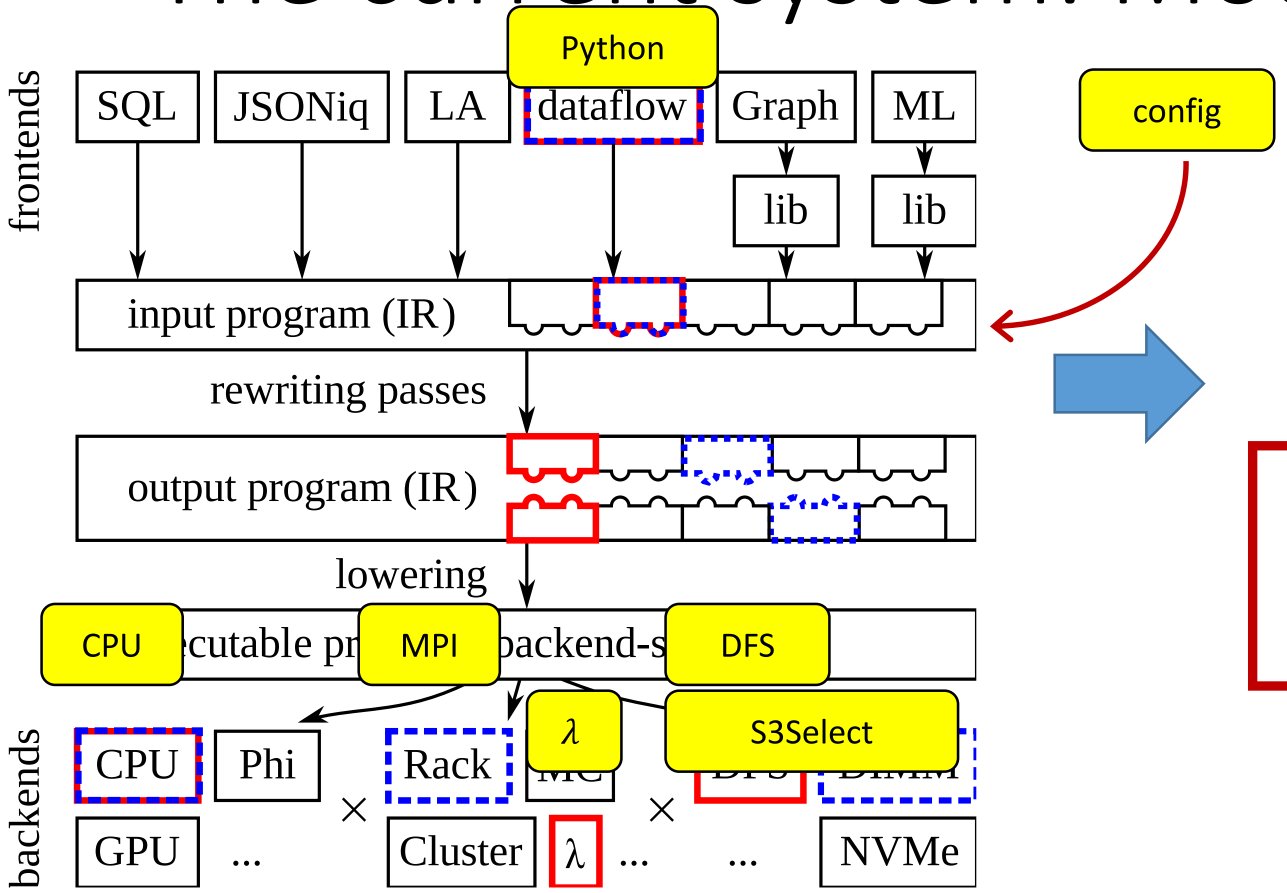
- Database engines and data processing systems tend to be very monolithic
  - Difficult to evolve
  - Expensive to maintain and change
  - Lots of legacy
- Is there a way to build systems that makes them as independent as possible of the underlying computing platform?
  - Make the choice of hardware part of the optimization process
  - Organize the system around its functionality, not its implementation

The initial idea



Müller, I., Marroquín, R., Koutsoukos, D., Wawrzoniak, M., Akhadov, S., & Alonso, G. (2020, June). The collection Virtual Machine: an abstraction for multi-frontend multi-backend data analysis. In *Proceedings of the 16th International Workshop on Data Management on New Hardware* (pp. 1-10).

The current system: Modularis

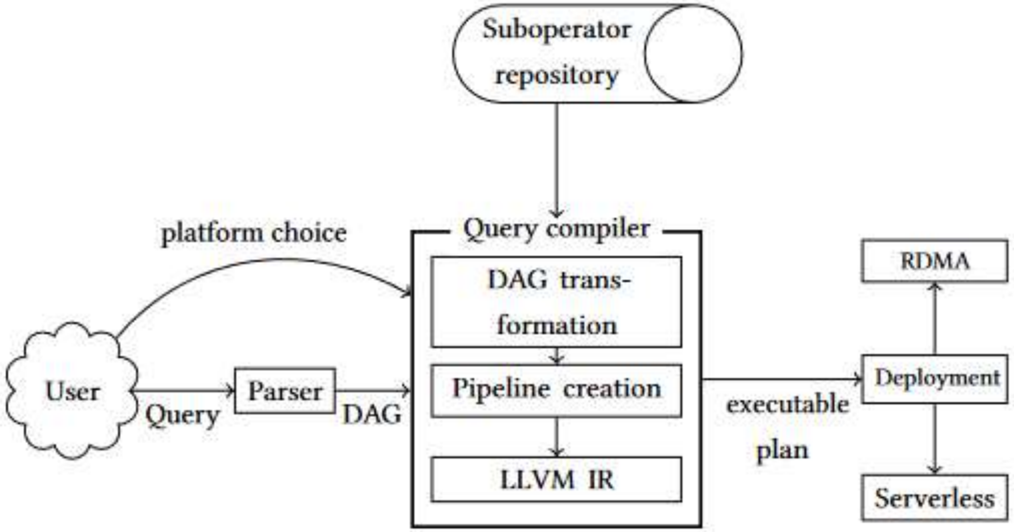


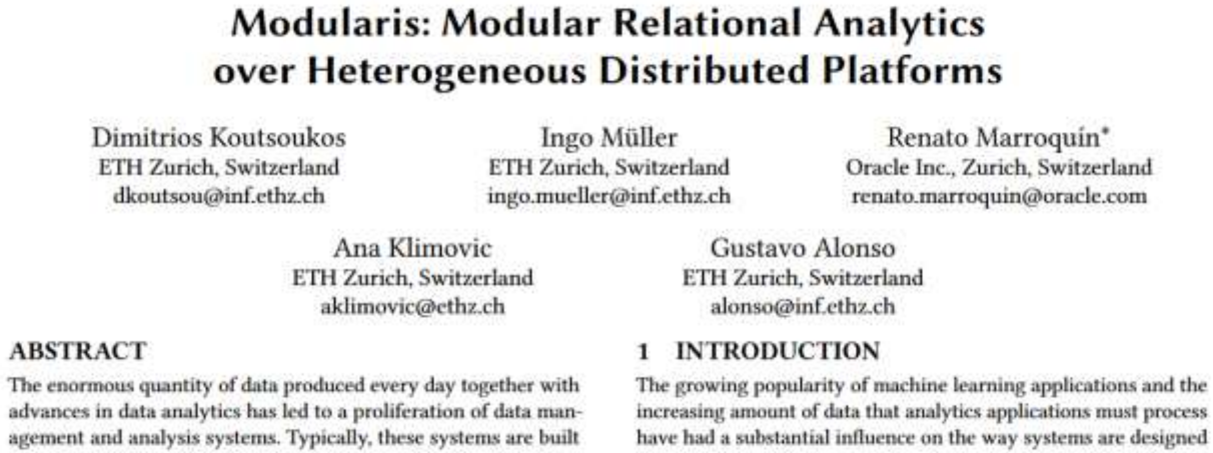
Category	Operators
Orchestration operators	Parameter Lookup, NestedMap
Data processing operators	(Parametrized) Map, Projection, Cartesian Product, Filter, Reduce (By Key), GroupBy, Zip, Local Histogram, Build and Probe, Partition, Semi-join, Sort, Top-K
MPI-specific operators	MPI Executor, MPI Histogram, MPI Exchange
Lambda-specific operators	Lambda Executor, Lambda Exchange
Smart storage-specific operators	S3Select Scan
Materialize and scan operators	Local Partitioning (AVX-based), Partition, Row Scan, Column Scan, Parquet Scan, Materialize Row Vector, Arrow table to collection

Müller, I., Marroquín, R., Koutsoukos, D., Wawrzoniak, M., Akhadov, S., & Alonso, G. (2020, June). The collection Virtual Machine: an abstraction for multi-frontend multi-backend data analysis. In *Proceedings of the 16th International Workshop on Data Management on New Hardware* (pp. 1-10).

Koutsoukos, D., Müller, I., Marroquín, R., Klimovic, A., & Alonso, G. (2020). Modularis: modular relational analytics over heterogeneous distributed platforms.

Modular data processing (Modularis)





Modularis: Modular Relational Analytics over Heterogeneous Distributed Platforms

Dimitrios Koutsoukos
ETH Zurich, Switzerland
dkoutsou@inf.ethz.ch

Ingo Müller
ETH Zurich, Switzerland
ingo.mueller@inf.ethz.ch

Renato Marroquín*
Oracle Inc., Zurich, Switzerland
renato.marroquin@oracle.com

Ana Klimovic
ETH Zurich, Switzerland
aklimovic@ethz.ch

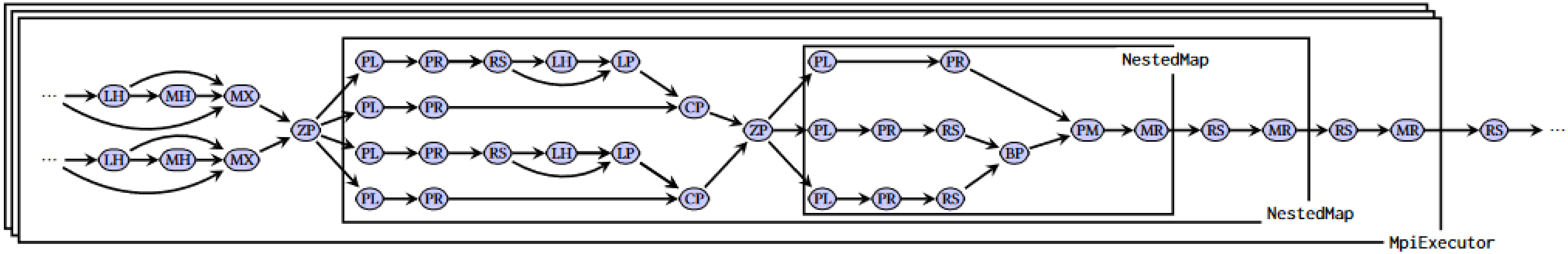
Gustavo Alonso
ETH Zurich, Switzerland
alonso@inf.ethz.ch

ABSTRACT

The enormous quantity of data produced every day together with advances in data analytics has led to a proliferation of data management and analysis systems. Typically, these systems are built

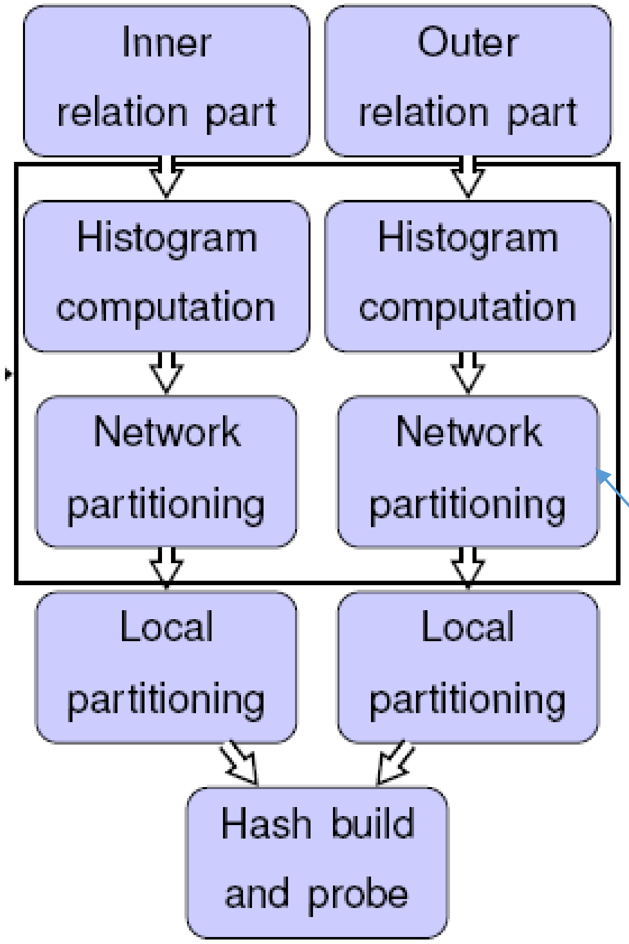
1 INTRODUCTION

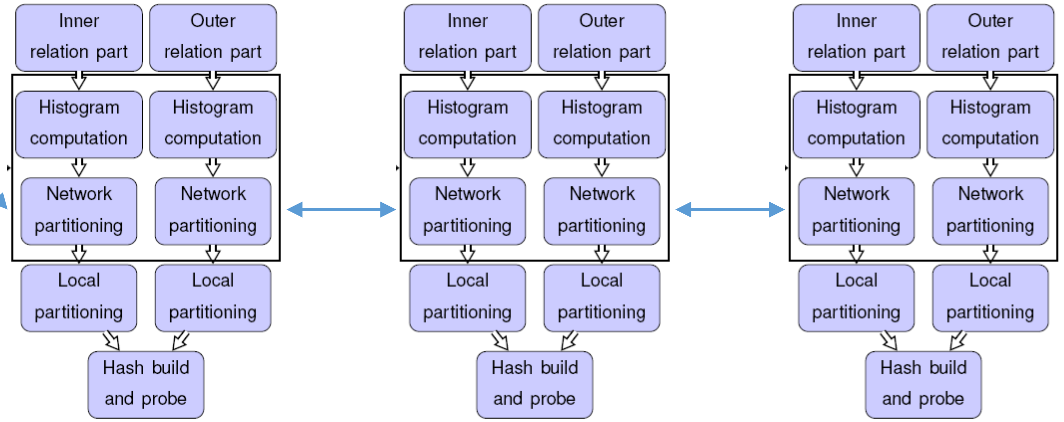
The growing popularity of machine learning applications and the increasing amount of data that analytics applications must process have had a substantial influence on the way systems are designed



Example of sub-operators

Partitioned hash join





Category	Operators
Orchestration operators	Parameter Lookup, NestedMap
Data processing operators	(Parametrized) Map, Projection, Cartesian Product, Filter, Reduce (By Key), GroupBy, Zip, Local Histogram, Build and Probe, Partition, Semi-join, Sort, Top-K
MPI-specific operators	MPI Executor, MPI Histogram, MPI Exchange
Lambda-specific operators	Lambda Executor, Lambda Exchange
Smart storage-specific operators	S3Select Scan
Materialize and scan operators	Local Partitioning (AVX-based), Partition, Row Scan, Column Scan, Parquet Scan, Materialize Row Vector, Arrow table to collection

A distributed join

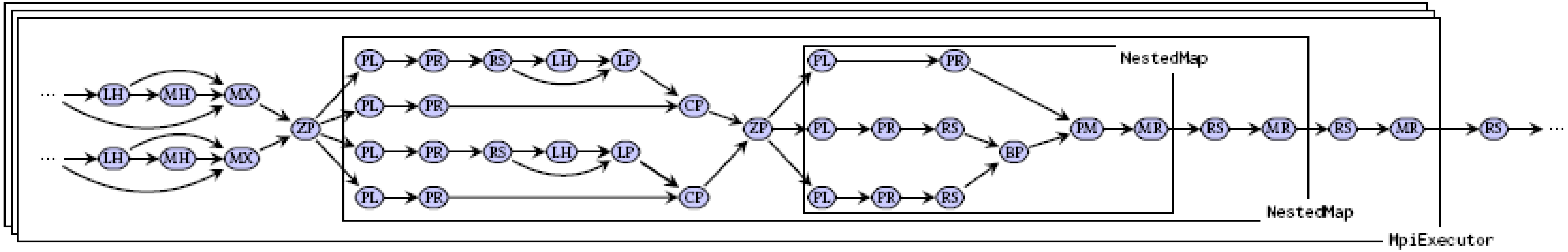


Figure 3: Plan that runs the distributed hash join with modular operators across many nodes

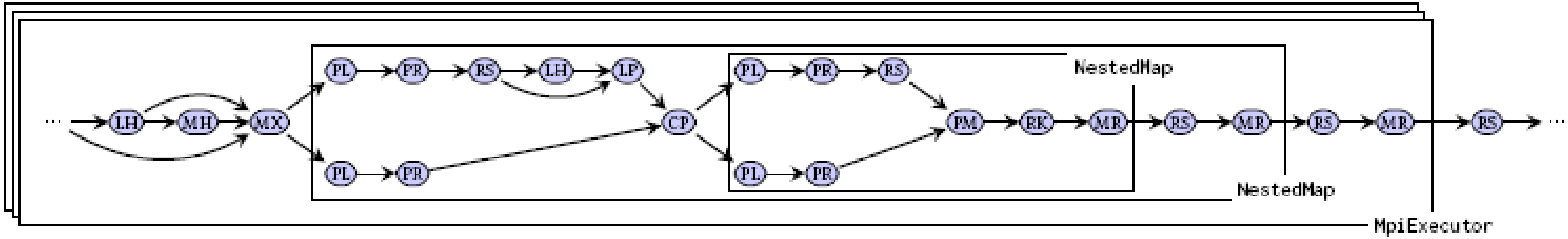


Figure 5: Plan that runs the distributed GROUP BY with modular operators across many nodes

Full queries

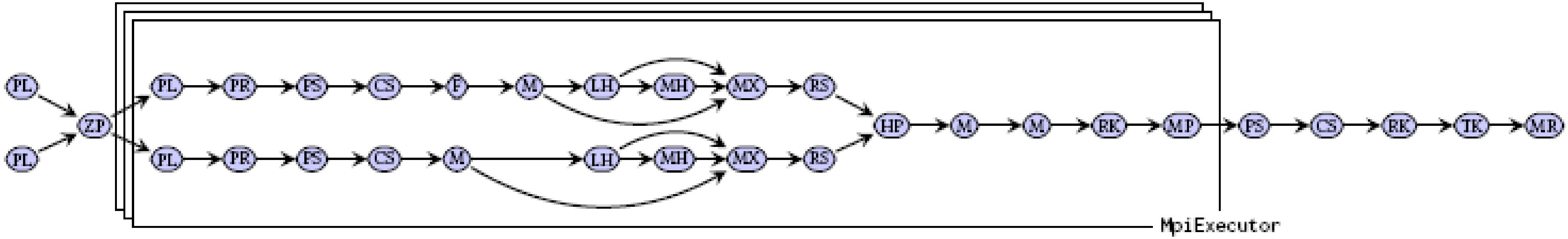


Figure 6: Modularis plan for TPC-H Q12 on RDMA

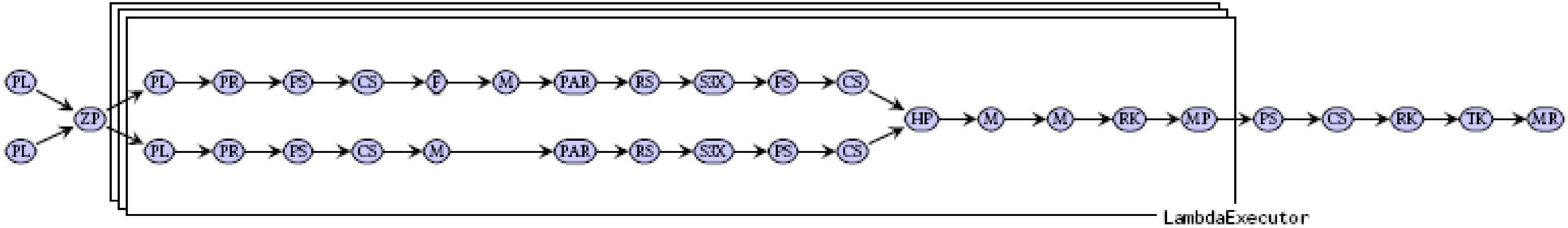
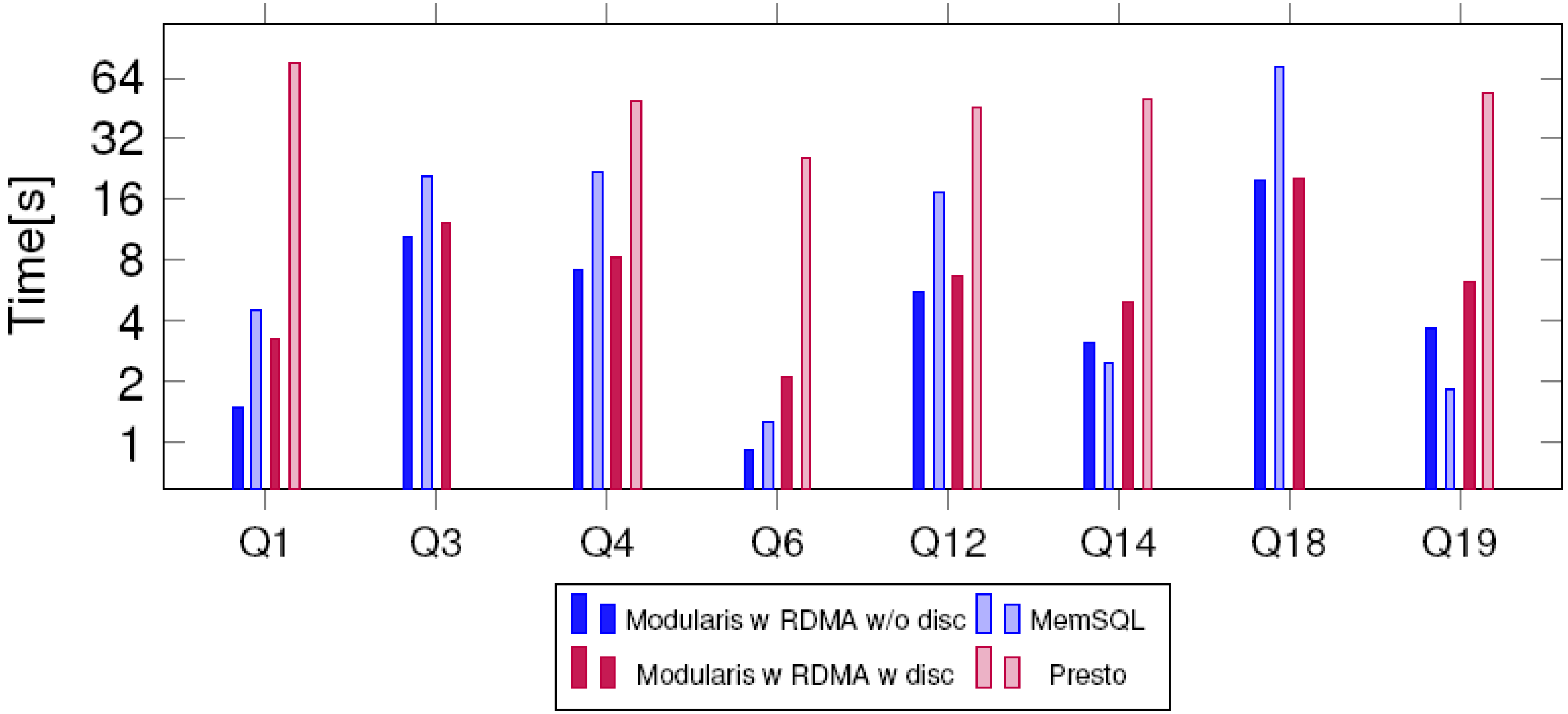


Figure 7: Modularis plan for TPC-H Q12 on Serverless

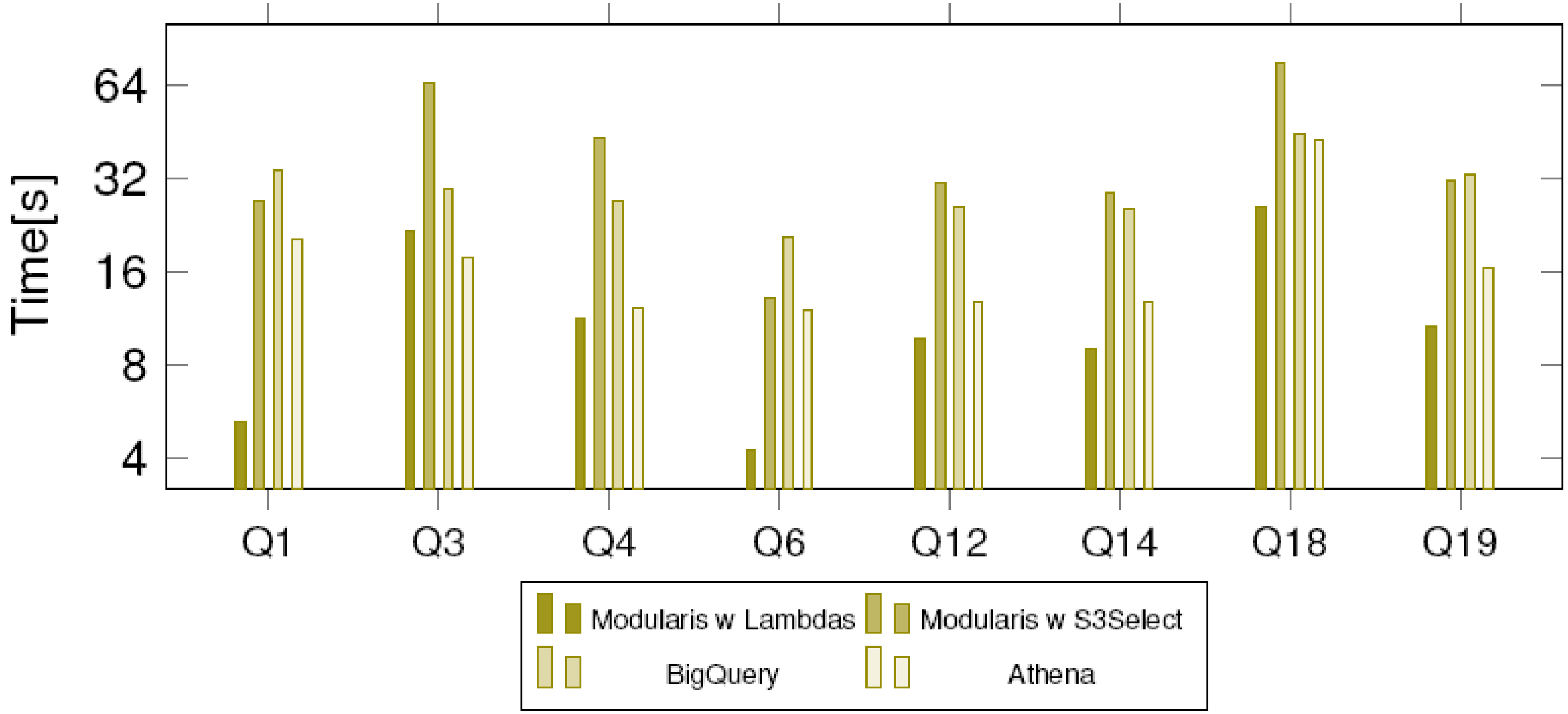
Deployment engine

- The deployment engine takes the query plan represented in LLVM IR and instantiates the sub-operators that are architecture dependent to produce an executable plan
- Current possible deployments:
  - RDMA based clusters (using MPI as communication library and for some operators)
  - Serverless computing (Amazon Lambdas) on S3
  - Serverless computing (Amazon Lambdas) and smart storage
  - Working on deployments over disaggregated memory and using hardware accelerators

TPC-H SF-500 (RDMA)



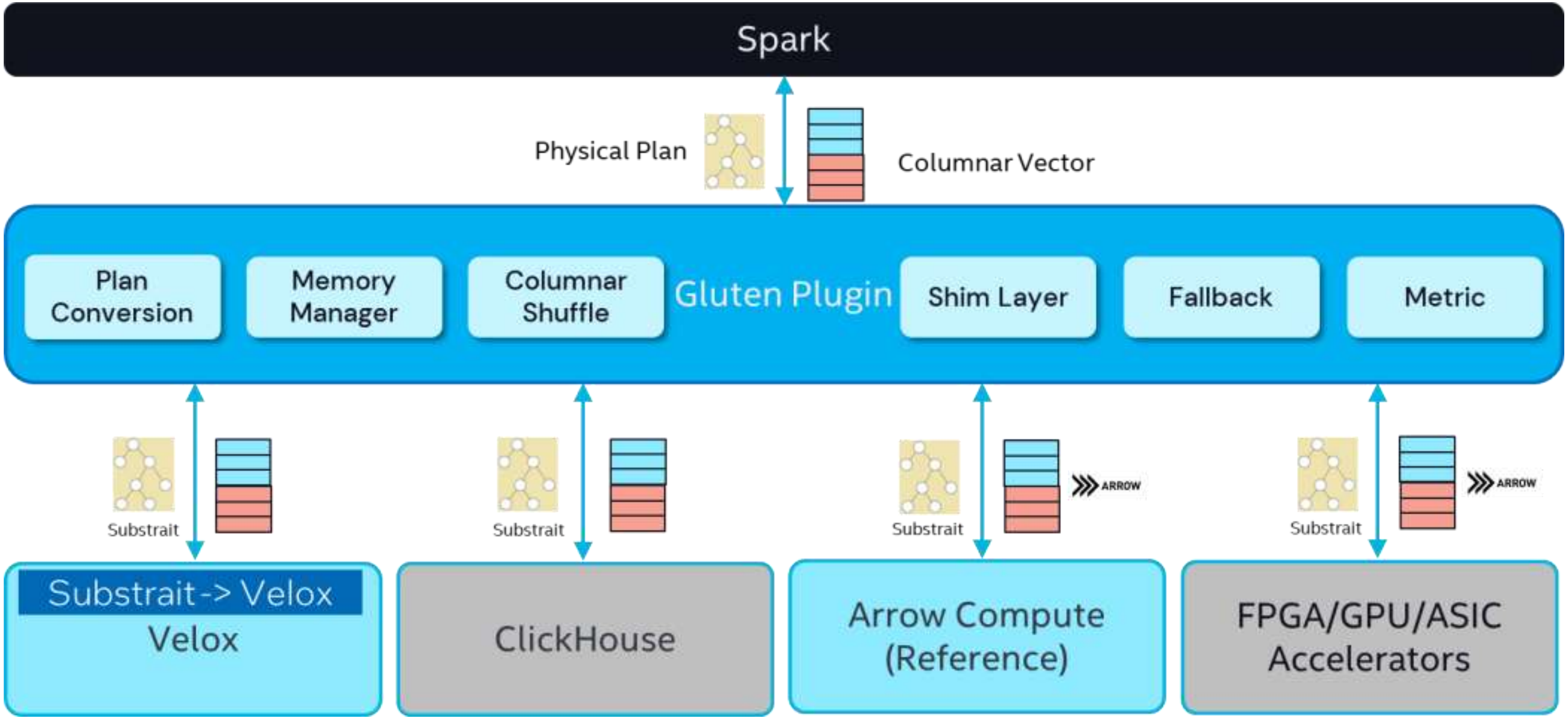
TPC-H SF-500 (Serverless)



The advantage?

- The effort goes into implementing sub-operators using a standardized interface
- Query optimization, tailoring to the underlying architecture, and using features of the hardware are all done automatically but are orthogonal to each other
- Where a query runs is just one more parameter passed to the system at query compilation time.

Impact: similar systems starting to appear

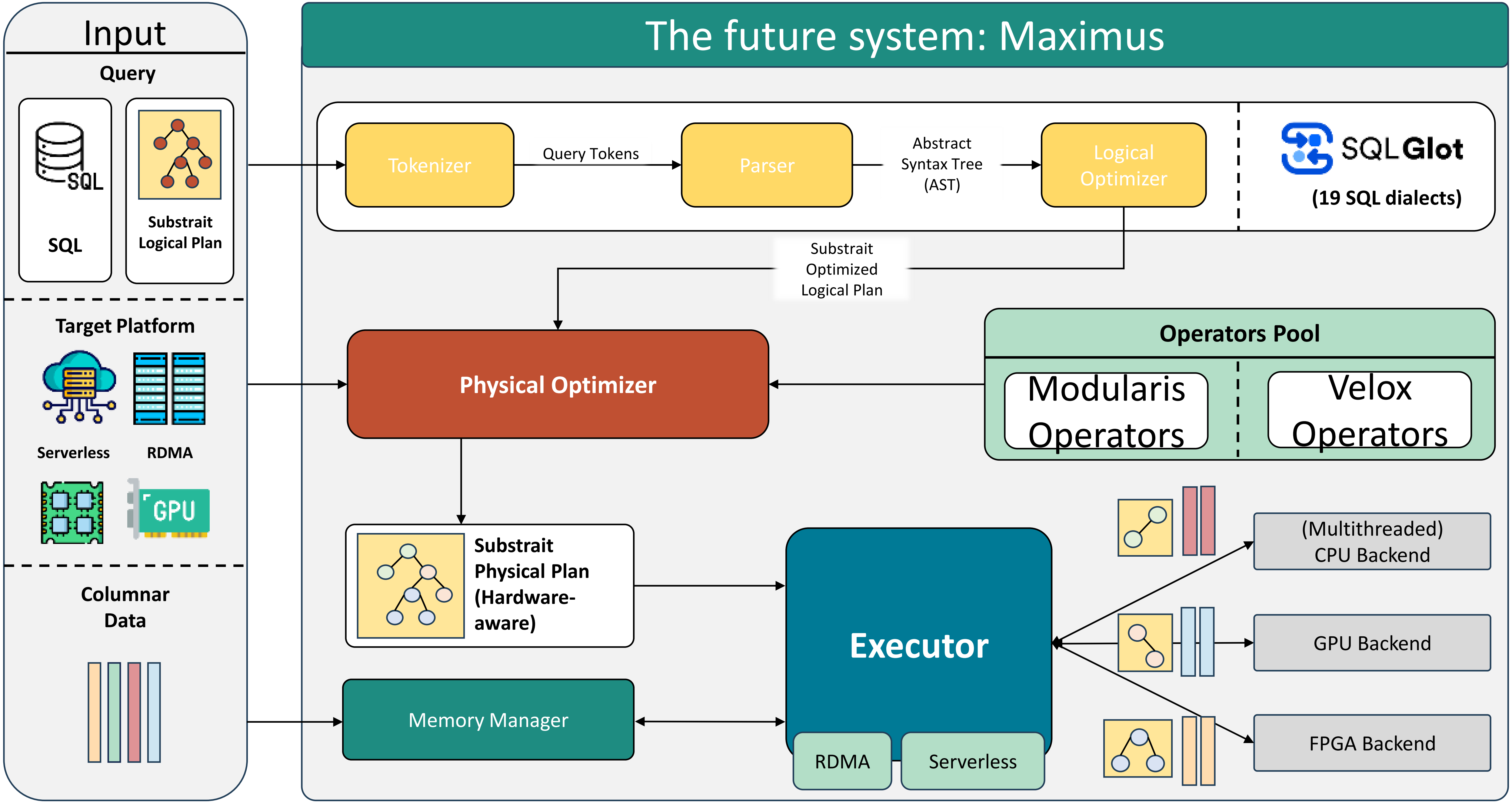


Synergies

- Modularis has attracted a lot of attention
- We are exploring synergies with several efforts:
  - Velox (Meta)
  - Gluten (Intel)
  - Use of GPU and DPUs (Microsoft)
  - Heterogeneous and disaggregated systems (Amazon)
- Agreement that an approach such as that of Modularis is the correct one to address all these challenges.
- Now engaged in a complete system redesign => Maximus

From Modularis to Maximus

- We have started a complete rewrite of Modularis into a new system
- Maximus:
  - Push execution model (vectorized)
  - Standard interfaces (compatible with Velox library)
  - Written in C++, high performance first
  - Integrating GPUs and other accelerators (smartNICs)
  - Porting relational operators to GPUs
  - Disaggregated storage and memory
  - Beyond relational (ML, vectors, embeddings, etc.)
  - Reuse of componets already available



Input
Query
SQL
SQL
Substrait Logical Plan
Target Platform
Serverless
RDMA
GPU
Columnar Data
The future system: Maximus
Tokenizer
Query Tokens
Parser
Abstract Syntax Tree (AST)
Logical Optimizer
SQLGlot
(19 SQL dialects)
Substrait Optimized Logical Plan
Physical Optimizer
Operators Pool
Modularis Operators
Velox Operators
Substrait Physical Plan (Hardware-aware)
Executor
RDMA
Serverless
Memory Manager
(Multithreaded) CPU Backend
GPU Backend
FPGA Backend

Future Work

- Continue system development with regular releases
- Develop algorithms on CPU, GPU, DPU and compare across the implementations and use cases
- Interfaces and execution models for heterogeneous architectures
- Derive a cost model and heuristics for choosing the best type of hardware to execute a given operator
- Query optimization in heterogeneous and distributed settings
- Generalization and interaction with other systems

Conclusions

Conclusions

- Data is growing
- Performance demands are growing
- But there are many options that new hardware provides
- The challenge is to redesign data processing so that it can evolve with the platforms and hardware