UNIVERSITY OF WISCONSIN–MADISON

CS 764: Topics in Database Management Systems

Lecture 23: Lambda functions

Xiangyao Yu

11/23/2020

Announcements

Please sign up for the presentation slots following the email

Today's Paper

Lambada: Interactive Data Analytics on Cold Data Using Serverless Cloud Infrastructure

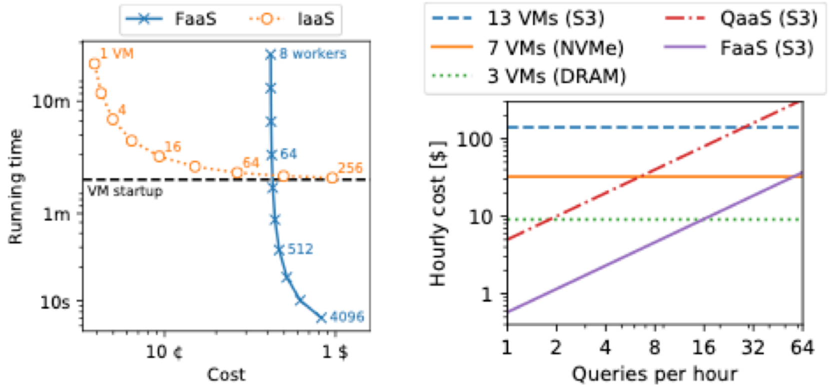
Ingo Müller
ingo.mueller@inf.ethz.ch
ETH Zurich

Renato Marroquín
marenato@inf.ethz.ch
ETH Zurich

Gustavo Alonso
alonso@inf.ethz.ch
ETH Zurich

ABSTRACT

Serverless computing has recently attracted a lot of attention from research and industry due to its promise of ultimate elasticity and operational simplicity. However, there is no consensus yet on whether or not the approach is suitable for data processing. In this paper, we present Lambada, a serverless distributed data processing framework designed to explore how to perform data analytics on serverless computing. In our analysis, supported with extensive experiments, we show in which scenarios serverless makes sense from an economic and performance perspective. We address several important technical questions that need to be solved to support data analytics and



(a) Job-scoped resources. (b) Always-on resources.

Figure 1: Comparison of cloud architectures.

SIGMOD 2020

What is Serverless Computing?

"**Serverless computing** is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources."

What is Serverless Computing?

"**Serverless computing** is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources."

Wikipedia
The Free Encyclopedia

No need to install, operate, and manage a server (infrastructure)

What is Serverless Computing?

WIKIPEDIA
The Free Encyclopedia

"**Serverless computing** is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources."

No need to install, operate, and manage a server (infrastructure)

According to a Berkeley TechReport [1]

Serverless computing = FaaS + BaaS → Backend-as-a-Service (e.g., BigQuery, Athena)

[1] E. Jonas, et al. *Cloud Programming Simplified: A Berkeley View on Serverless Computing*, Berkeley TR 2019

What is Serverless Computing?

"**Serverless computing** is a cloud computing execution model in which the cloud provider runs the server, and dynamically manages the allocation of machine resources."

No need to install, operate, and manage a server (infrastructure)

According to a Berkeley TechReport [1]

Serverless computing = FaaS + BaaS

BaaS → Backend-as-a-Service (e.g., BigQuery, Athena)

FaaS → Function-as-a-Service (**focus of this lecture**)

[1] E. Jonas, et al. *Cloud Programming Simplified: A Berkeley View on Serverless Computing*, Berkeley TR 2019

Function-as-a-Service

FaaS offerings

- **AWS Lambda**
- Google Cloud Functions
- Microsoft Azure Functions
- IBM/Apache's OpenWhisk (open source)
- Oracle Cloud Fn (open source)

AWS Lambda

Features

- Function starts execution (within a container) within sub-second
- Charged at 100ms granularity that the container runs
- Can run thousands/millions of small invocations in parallel
- Works well for source code compilation and video encoding

AWS Lambda

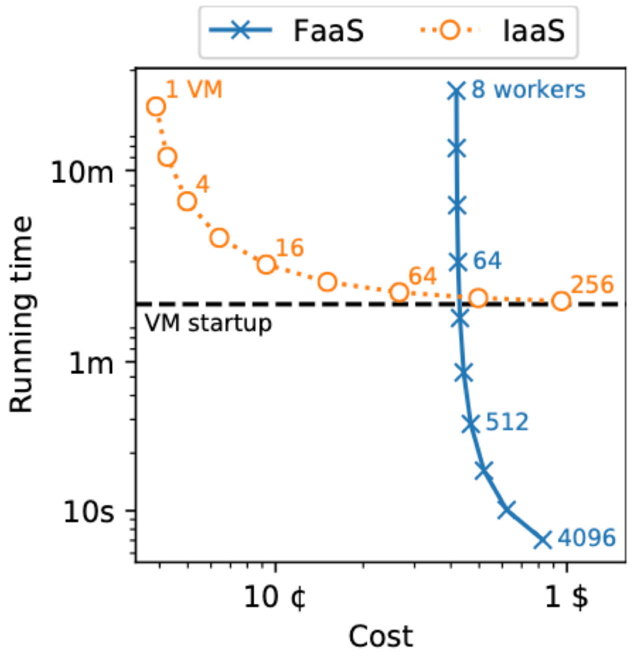
Features

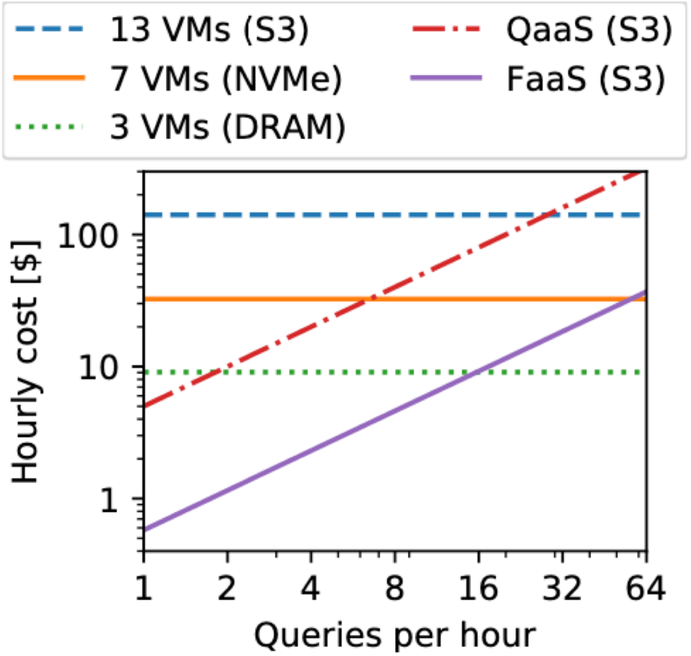
- Function starts execution (within a container) within sub-second
- Charged at 100ms granularity that the container runs
- Can run thousands/millions of small invocations in parallel
- Works well for source code compilation and video encoding

Limitations

- Limited runtime: 15 min
- Limited resources: 1~2 cores, 3 GB main memory
- No direct communication between functions

Serverless Database via FaaS?

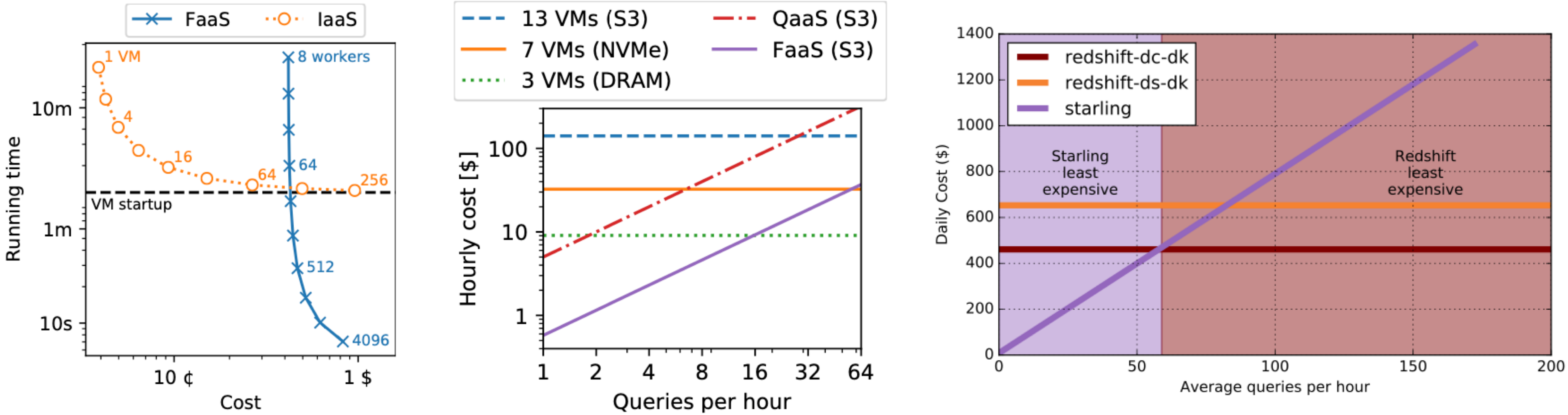




FaaS is attractive for **interactive** queries (i.e., low latency) on **cold** data (i.e., infrequently accessed)

- Fast startup time
- Pay-as-you-go

Serverless Database via FaaS?

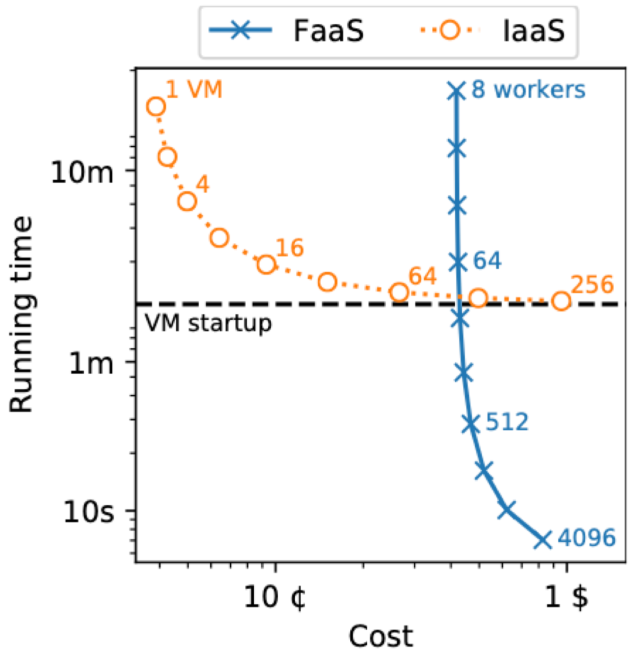


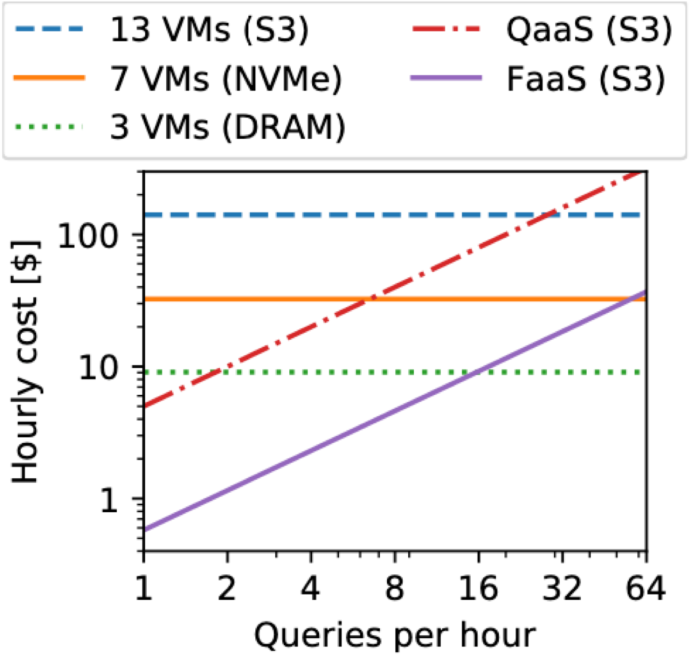
FaaS is attractive for **interactive** queries (i.e., low latency) on **cold** data (i.e., infrequently accessed)

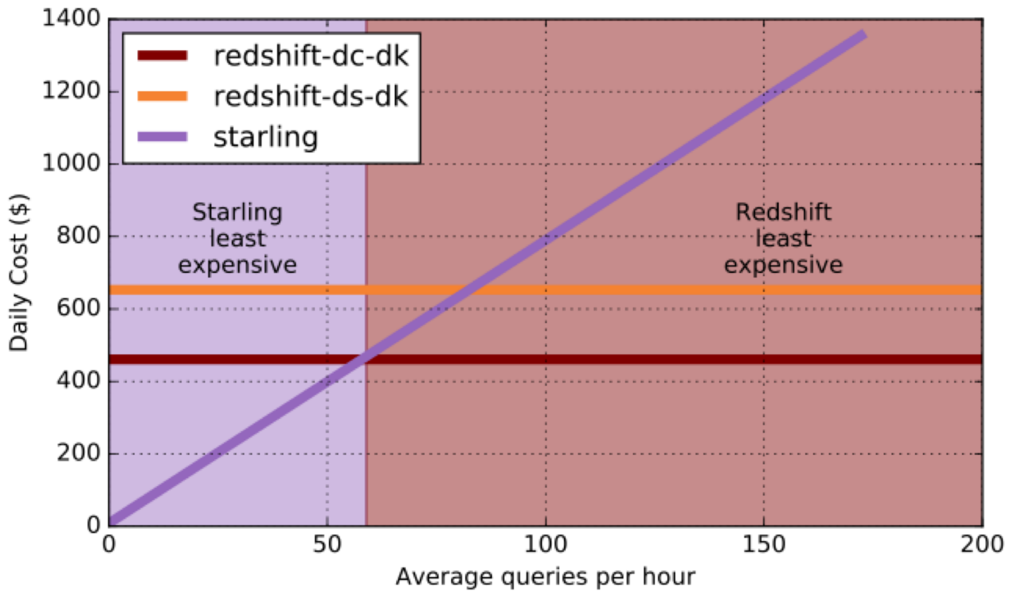
- Fast startup time
- Pay-as-you-go

[2] Perron, Matthew, et al. "Starling: A Scalable Query Engine on Cloud Functions." SIGMOD 2020

Serverless Database via FaaS?







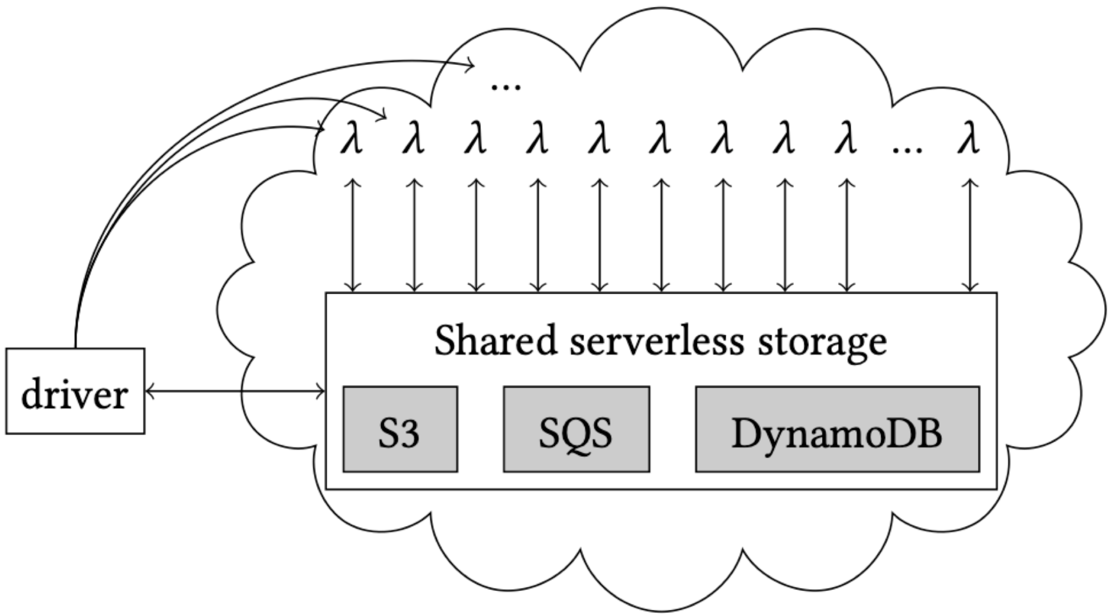
FaaS is attractive for **interactive** queries (i.e., low latency) on **cold** data (i.e., infrequently accessed)

- Fast startup time
- Pay-as-you-go

What about QaaS? (e.g., Big Query, Athena, etc) => will discuss later

[2] Perron, Matthew, et al. "Starling: A Scalable Query Engine on Cloud Functions." SIGMOD 2020

Architecture Overview of Lambada



Driver: Invokes serverless workers; runs on the local development machine

λ: Lambda function

S3: AWS cloud storage

DynamoDB: NoSQL key-value store

Simple Queue Service (SQS): Message queuing service for communication over the Internet

System Components of Lambada

Part 1: Batch start massive numbers of serverless workers

Part 2: Scan operator for efficiently reading query input

Part 3: S3-based exchange operator

Batch Invocation

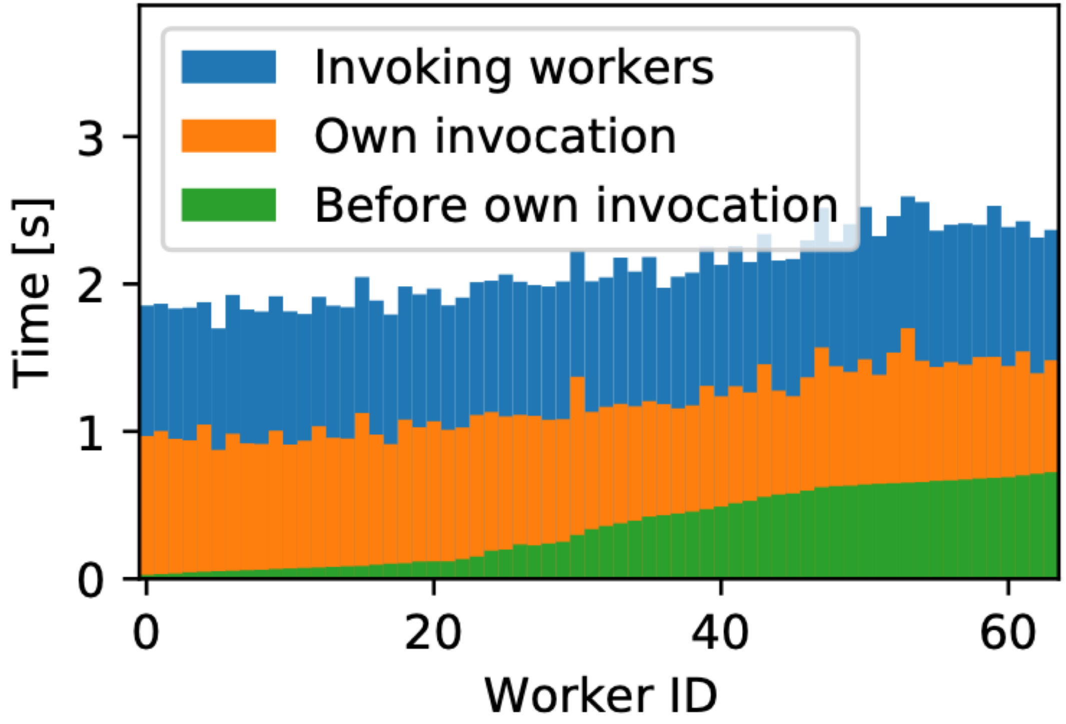
Metric	Region			
	eu	us	sa	ap
Single invocation time [ms]	36	363	474	536
Concurrent inv. rate [inv./s]	294	276	243	222
Intra-region rate [inv./s]	81	79	84	81

Table 1: Characteristics of function invocations.

A single driver can invocate 220–290 workers per second

Invoking 4096 workers takes 13–18 seconds

Lambada Two-Level Invocation

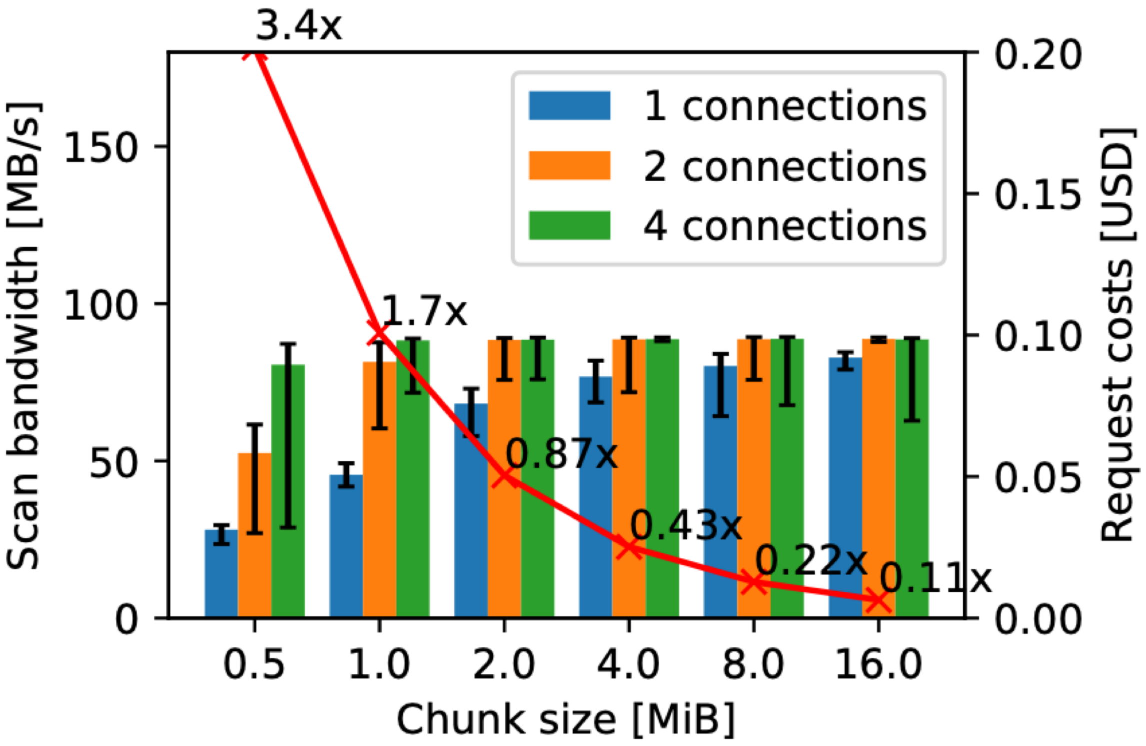


Driver invocates first-generation workers which invokes second-generation workers in parallel

Invoking 4096 workers takes 2.5 seconds (vs. 13–18 sec before)

- 64 first-generation workers, each invoking 64 second-generation workers

Lambada Cloud-Native Scan

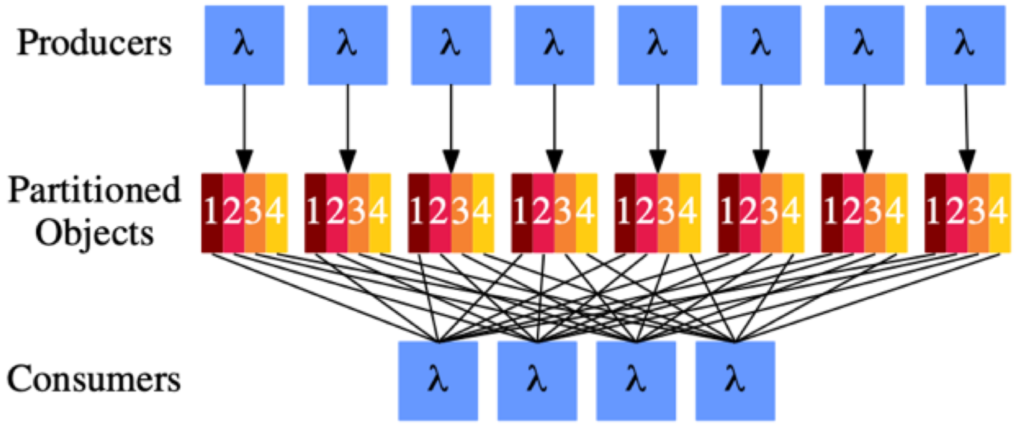


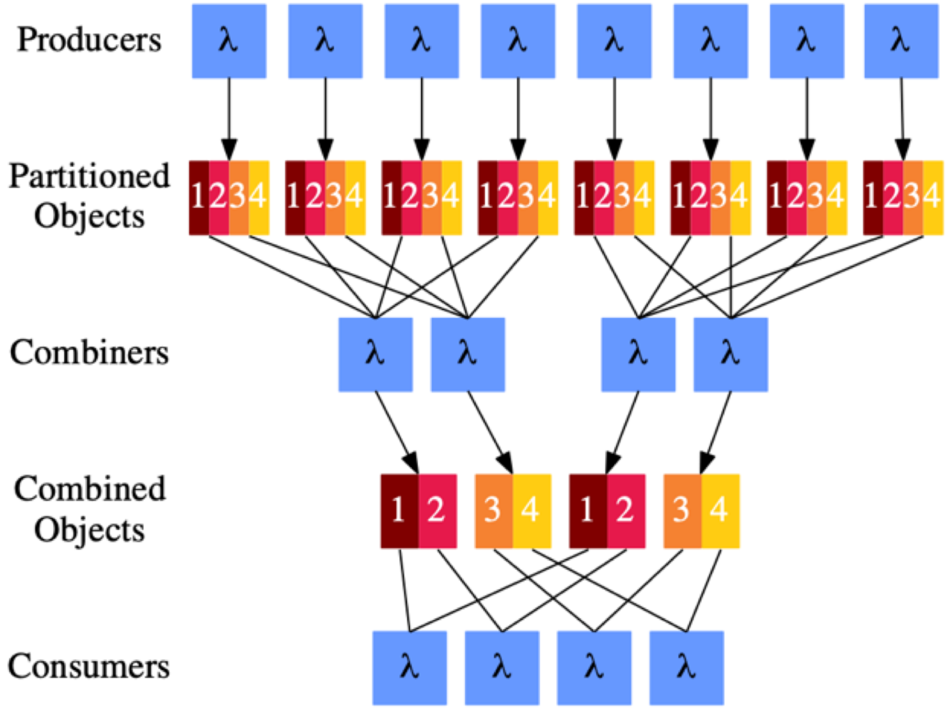
Scan bandwidth saturates at about 90MB/s per worker

Easier to saturate bandwidth with larger chunks and more connectors

Large chunk sizes reduce request cost

Exchange Operator





Optimization 1: multi-level exchange

Optimization 2: write combining

- Lambada assumes the number of workers does not change while Starling [2] assumes it can change

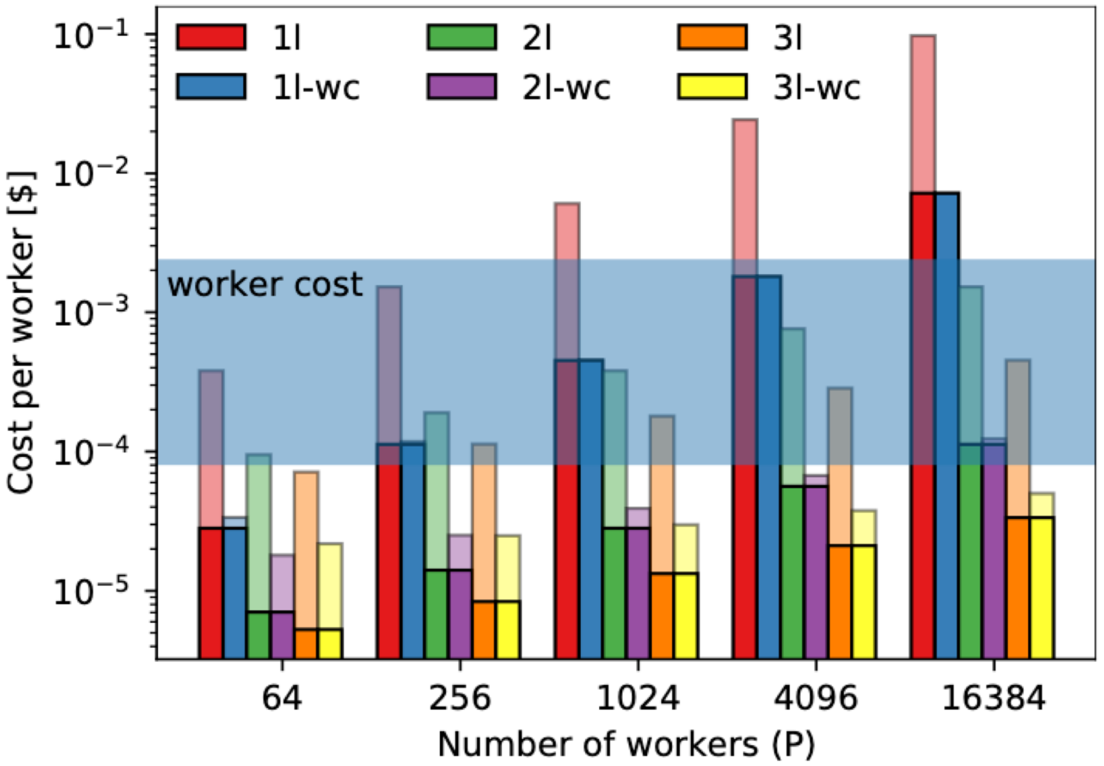
* figures copied from [2]

[2] Perron, Matthew, et al. "Starling: A Scalable Query Engine on Cloud Functions." SIGMOD 2020

Exchange Operator Costs

Table 2: Cost models of S3-based exchange algorithms.

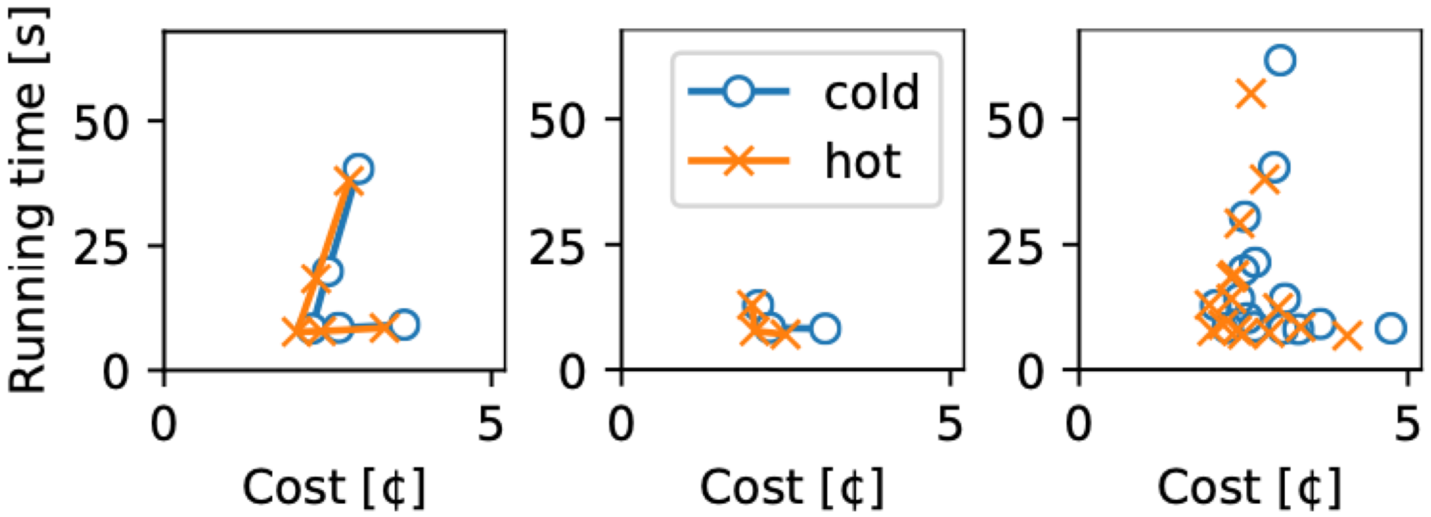
Algorithm	#reads	#writes	#lists	#scans
1l	P^2	P^2	$\mathcal{O}(P)$	1
1l-wc	P^2	P	$\mathcal{O}(P)$	1
2l	$2P\sqrt{P}$	$2P\sqrt{P}$	$\mathcal{O}(P)$	2
2l-wc	$2P\sqrt{P}$	$2P$	$\mathcal{O}(P)$	2
3l	$3P\sqrt[3]{P}$	$3P\sqrt[3]{P}$	$\mathcal{O}(P)$	3
3l-wc	$3P\sqrt[3]{P}$	$3P$	$\mathcal{O}(P)$	3



Both optimizations lower the cost

With three-level exchange and write combining, the exchange cost becomes negligible compared to worker cost

Evaluation – Worker Configuration



(a) $F = 1$, **varying** M. **(b)** $M = 1792$ MiB, **varying** F. **(c) Varying** M **and** F.

Figure 8: TPC-H Query 1 with varying memory (M**) and number of files (**F**) per worker.**

Range of M: 512, 1024, 1796, 2048, and 3008 MB
Range of F: 1, 2, 4

Evaluation – Min/Max Filtering

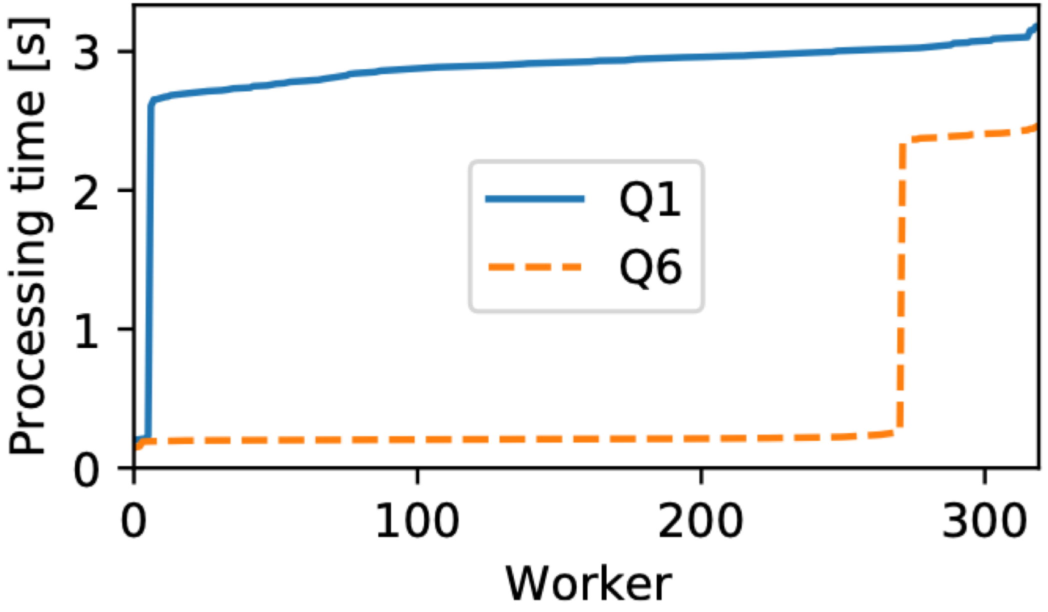
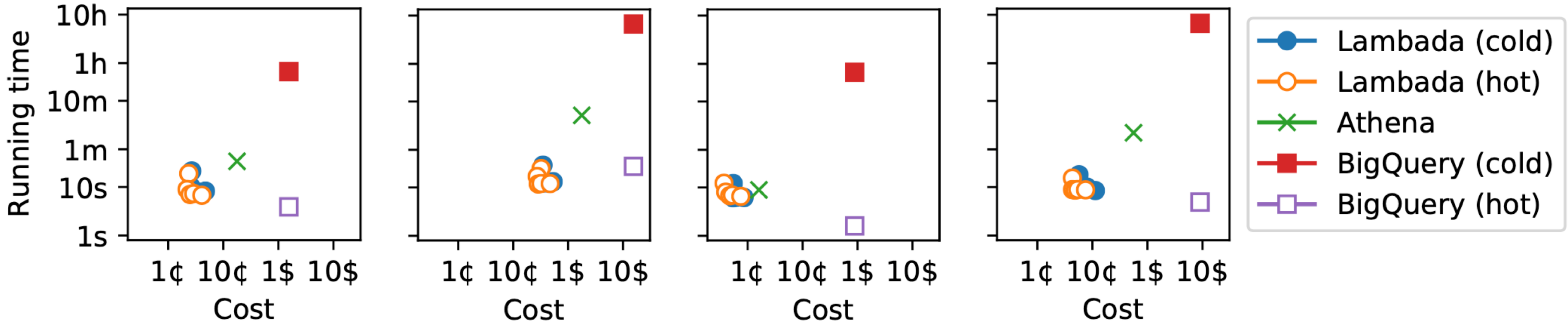


Figure 9: Distribution of processing time.

Both Q1 and Q6 in TPCH only scan a single table

- Q1 selects 98% of the input data
- Q6 selects 2% of input data

Evaluation – Comparison with QaaS



(a) Q1, SF 1 k. **(b) Q1, SF** 10 k. **(c) Q6, SF** 1 k. **(d) Q6, SF** 10 k.

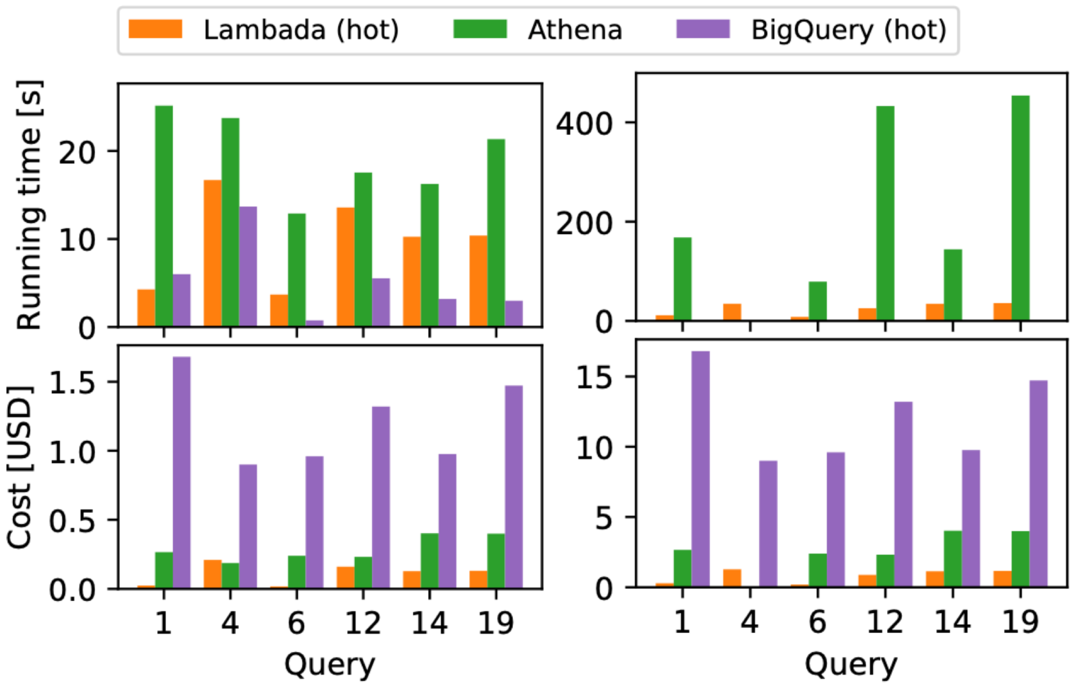
AWS Athena and Google BigQuery

- Similar cost model: $5 per 1TB scanned
- Athena considers only the selected columns while BigQuery considers all columns

Why is Lambada cheaper than QaaS? Is it because of the pricing model or is there a more fundamental reason?

Evaluation – Comparison with QaaS

Performance and cost benefits of Lambada reduce for complex queries (e.g., those with joins)

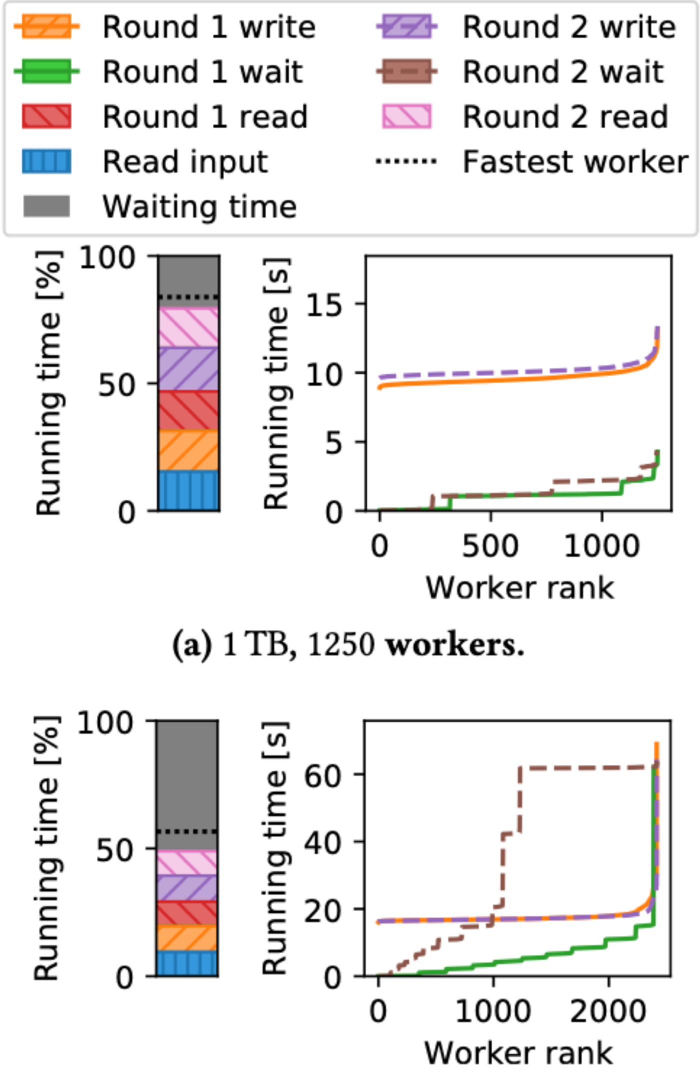


(a) **SF** 1 k, $W = 512$, $F = 2$. (b) **SF** 10 k, $W = 1280$, $F = 8$.

Figure 11: TPC-H queries on Lambada ($M = 2\,\text{GiB}$).

Evaluation – Straggler in 2-level Exchange

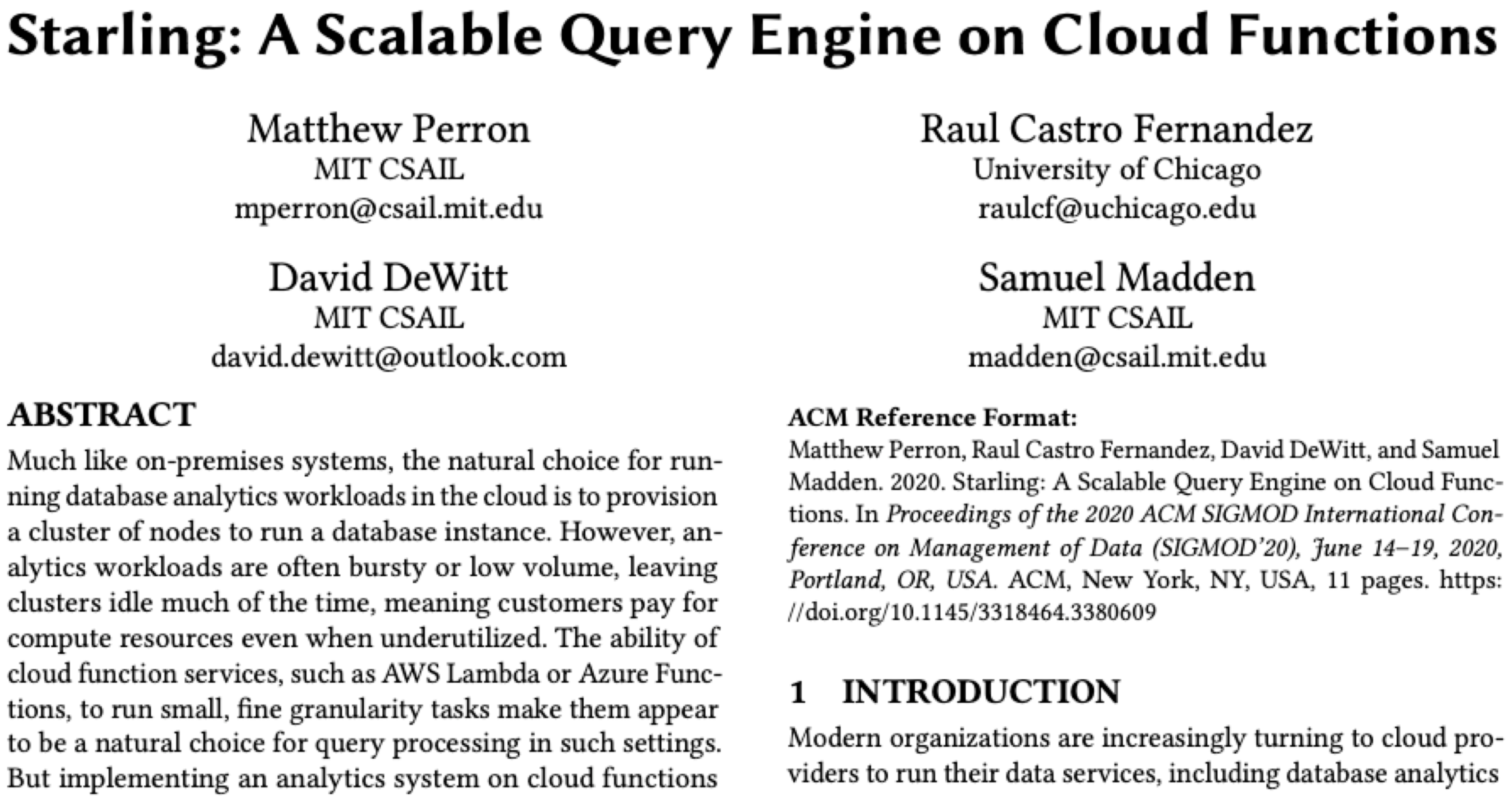
Straggler effect is significant for large data set and large numbers of workers



(a) 1 TB, 1250 **workers.**

(b) 3 TB, 2500 **workers.**

Another Relevant Paper


Starling: A Scalable Query Engine on Cloud Functions

Matthew Perron
MIT CSAIL
mperron@csail.mit.edu

Raul Castro Fernandez
University of Chicago
raulcf@uchicago.edu

David DeWitt
MIT CSAIL
david.dewitt@outlook.com

Samuel Madden
MIT CSAIL
madden@csail.mit.edu

ABSTRACT

Much like on-premises systems, the natural choice for running database analytics workloads in the cloud is to provision a cluster of nodes to run a database instance. However, analytics workloads are often bursty or low volume, leaving clusters idle much of the time, meaning customers pay for compute resources even when underutilized. The ability of cloud function services, such as AWS Lambda or Azure Functions, to run small, fine granularity tasks make them appear to be a natural choice for query processing in such settings. But implementing an analytics system on cloud functions

ACM Reference Format:

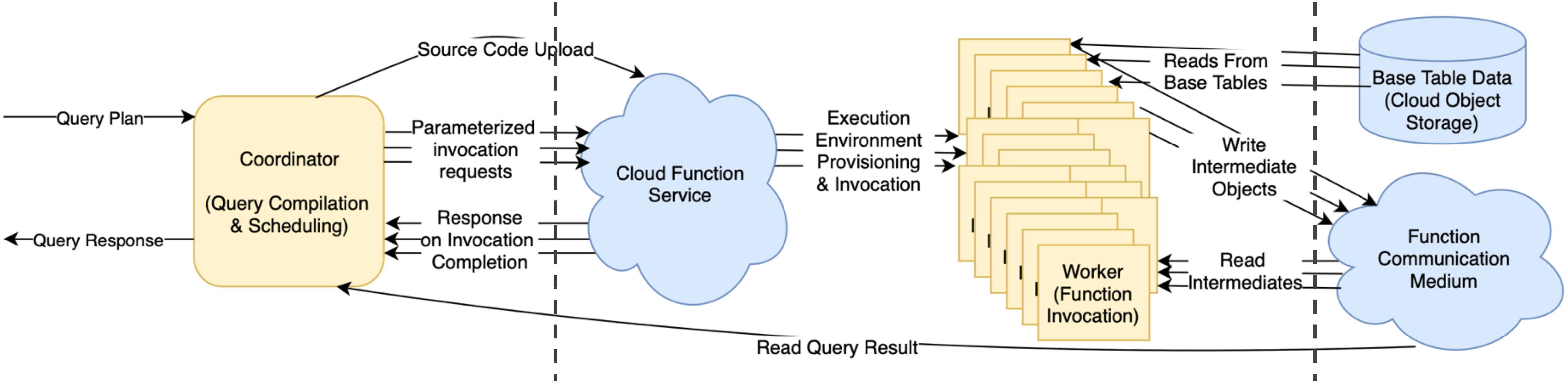
Matthew Perron, Raul Castro Fernandez, David DeWitt, and Samuel Madden. 2020. Starling: A Scalable Query Engine on Cloud Functions. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data (SIGMOD'20), June 14–19, 2020, Portland, OR, USA.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3318464.3380609

1 INTRODUCTION

Modern organizations are increasingly turning to cloud providers to run their data services, including database analytics

SIGMOD 2020

Starling Architecture

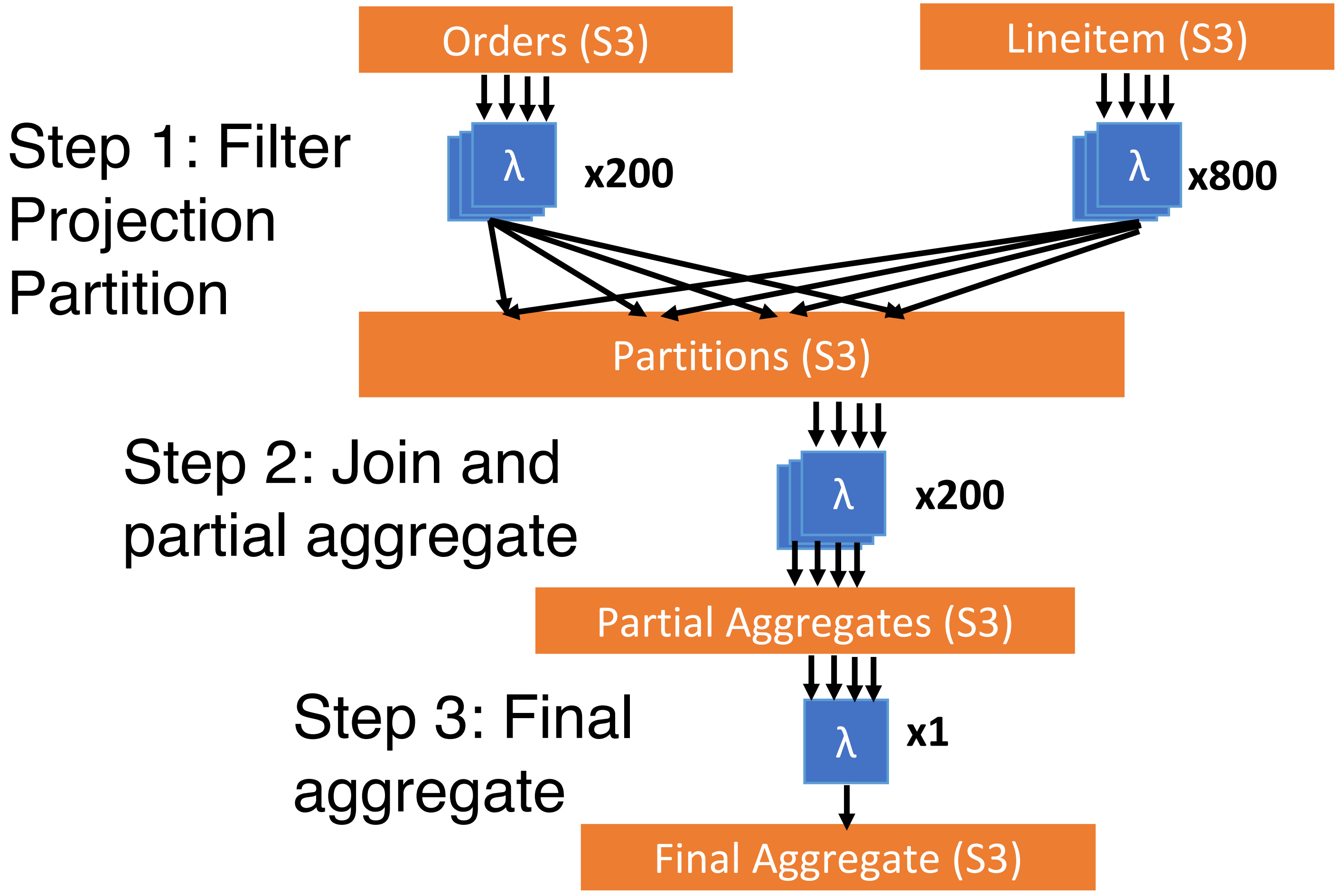


Coordinator
- Query compilation
- Initiate workers

Workers
- Query execution

Storage
- Input data
- Communication

Example Query Execution (TPC-H Q12)

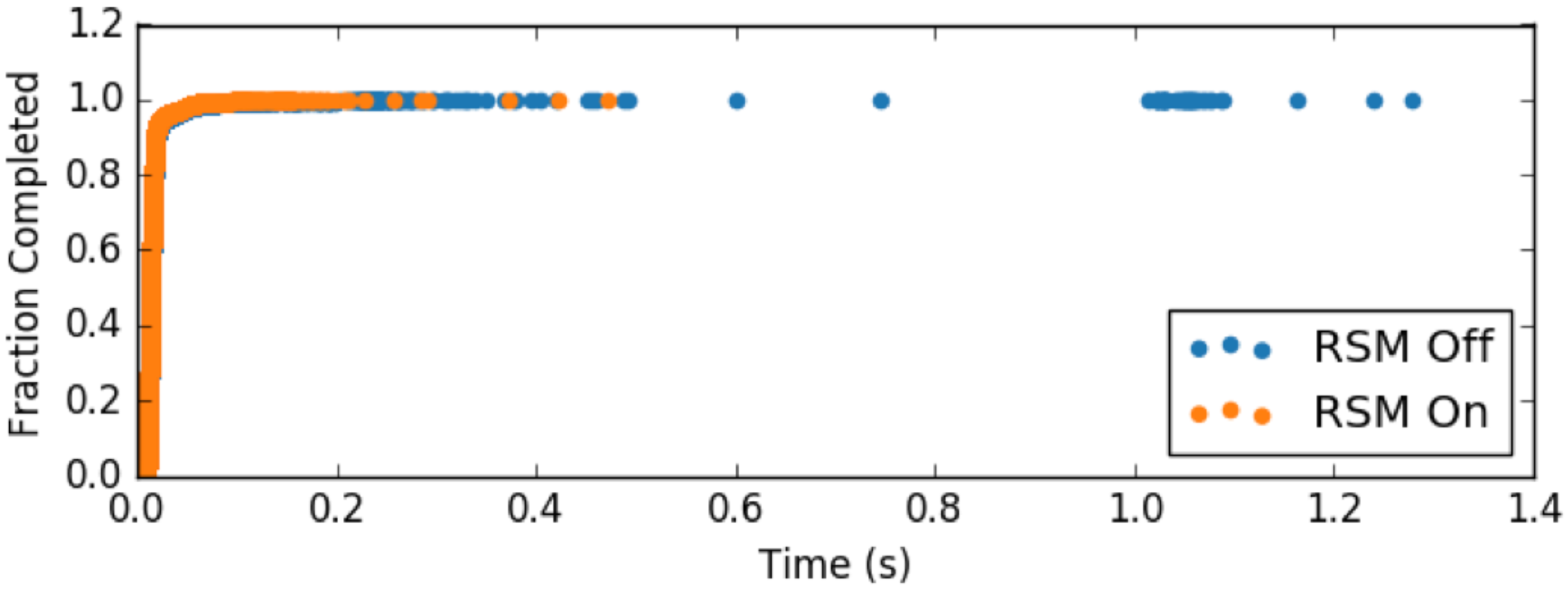


Different stages invoke different numbers of lambda workers

Optimizations

Read straggler mitigation (RSM)

- If a read request times out, send duplicate request

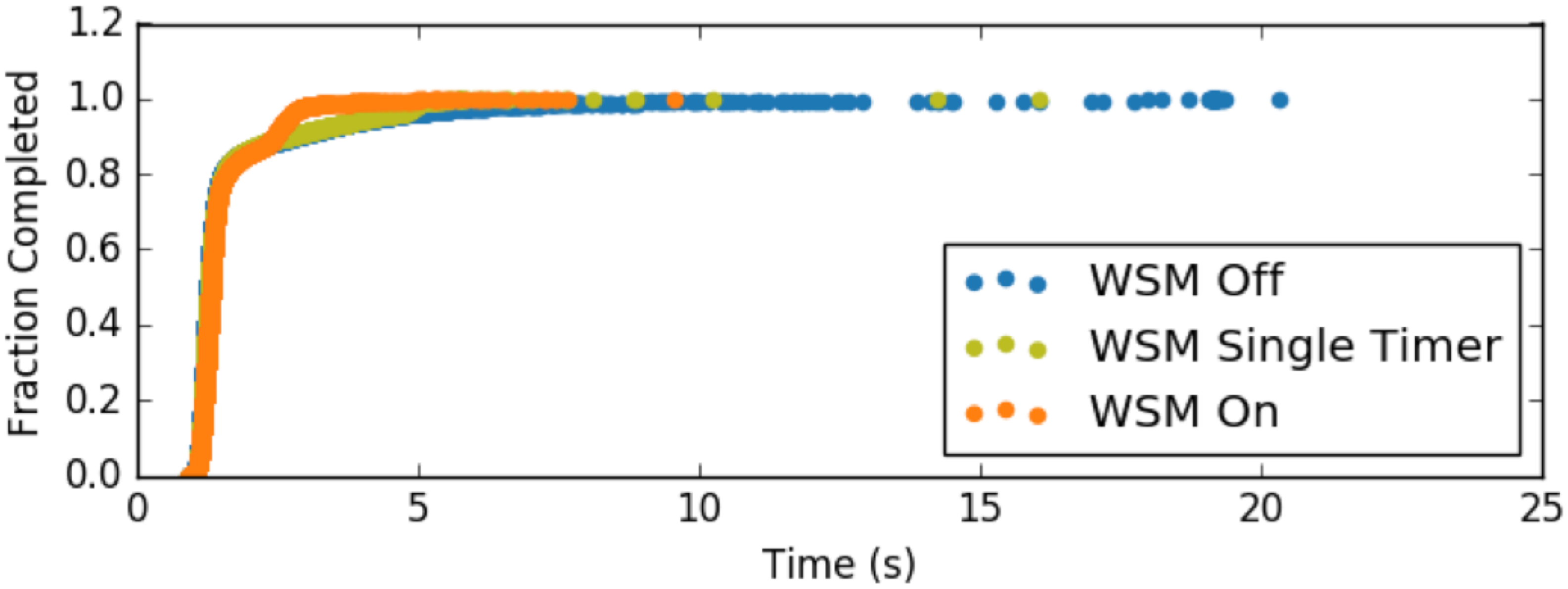


Optimizations

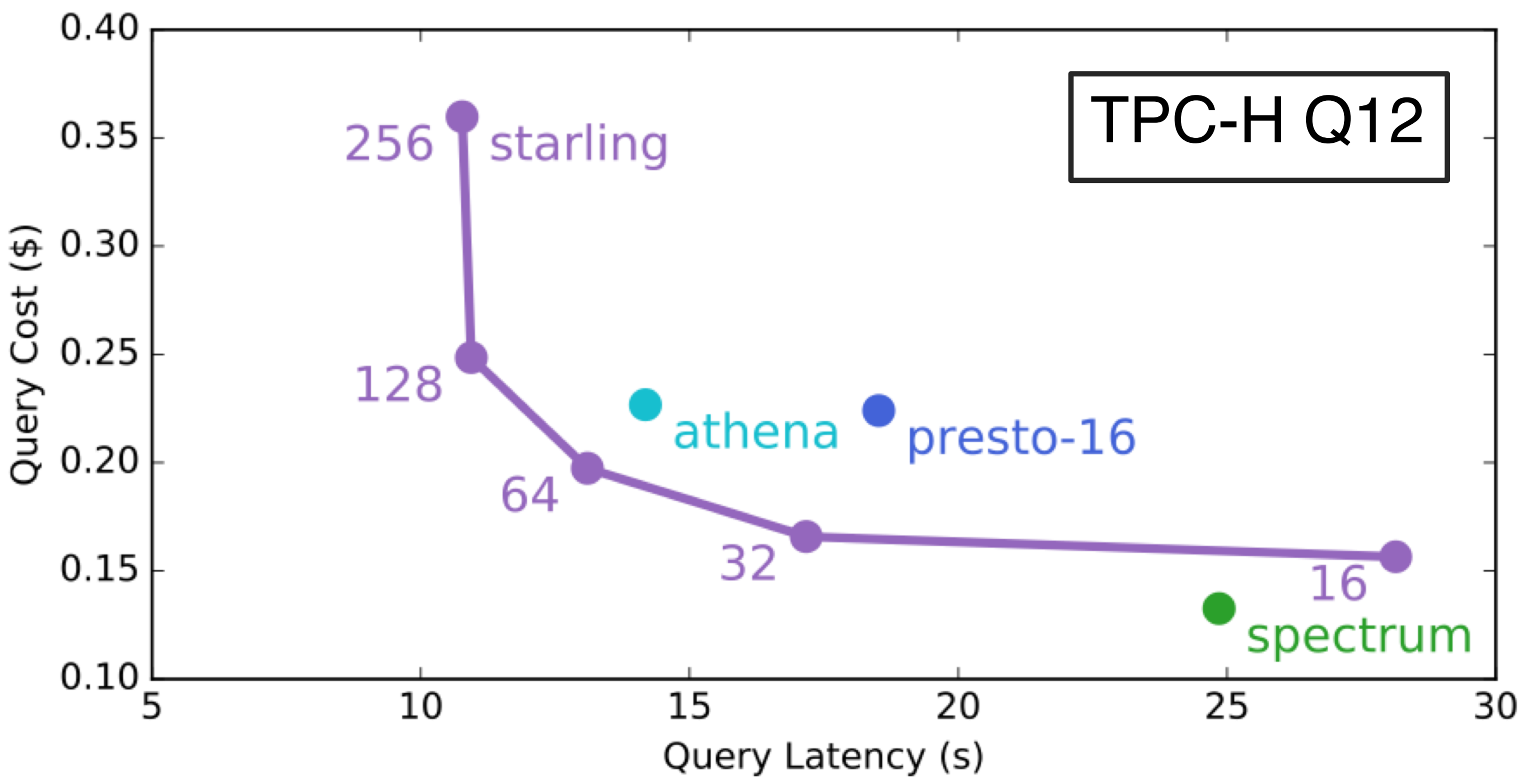
Read straggler mitigation (RSM)

Write straggler mitigation (WSM)

- If a write request times out, send duplicate request
- Single Timer: allow only single time out

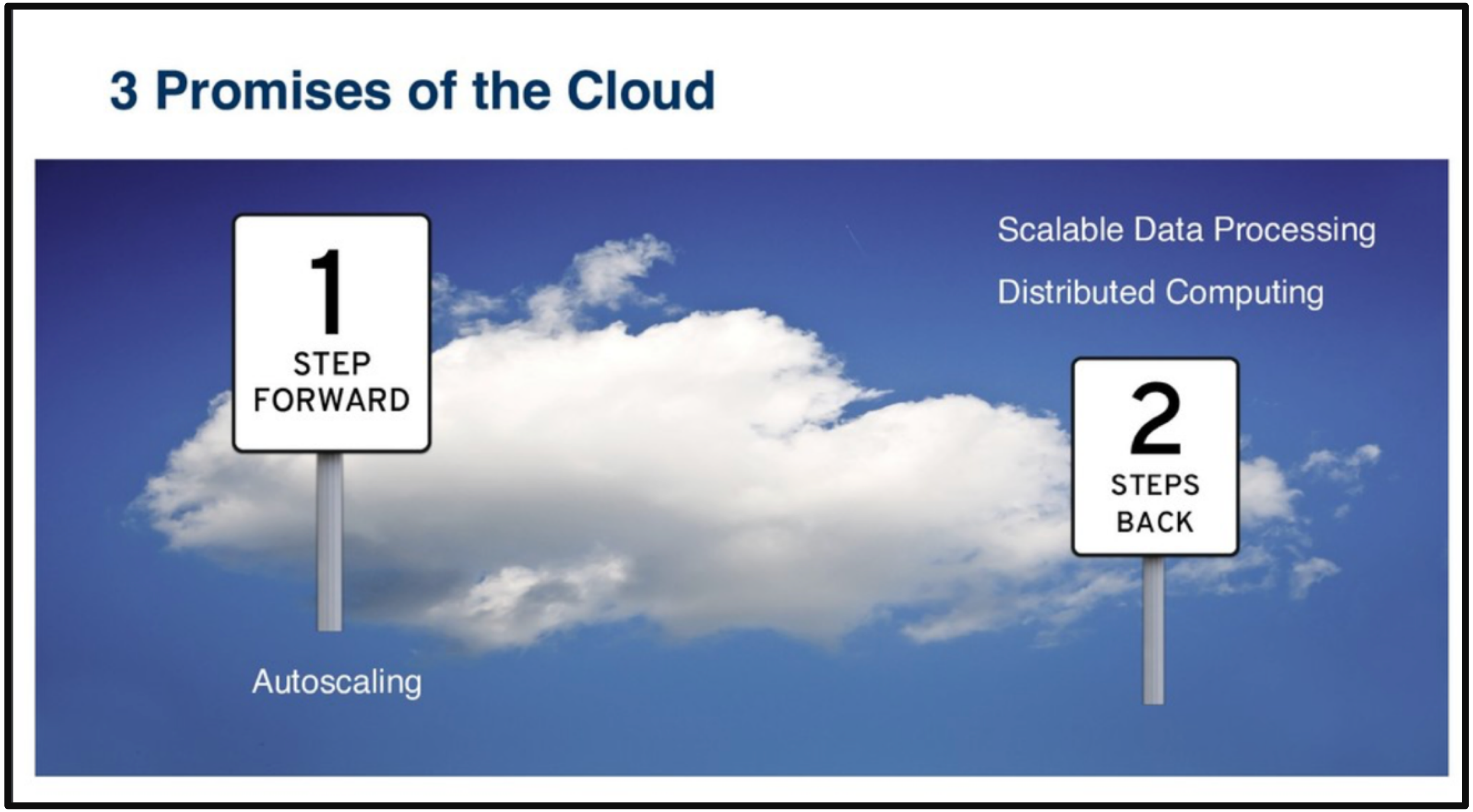


Evaluation of Starling



Easy to tune performance by changing the number of tasks

Other Opinions of Serverless Database [3]



- Cloud storage is 1—2 orders of magnitude slower than SSD
- No inter-function communication
- Paper gave suggestions for future work

[3] Hellerstein, Joseph M., et al. "Serverless computing: One step forward, two steps back." *arXiv preprint arXiv:1812.03651*(2018).

Other Opinions of Serverless Database [1]

> However in our final example, Serverless SQLite, we identify a use case that maps so poorly to FaaS that we conclude that databases and other state-heavy applications will remain as BaaS"

Serverless database as BaaS

- Examples: Snowflake, Athena, BigQuery

[1] E. Jonas, et al. *Cloud Programming Simplified: A Berkeley View on Serverless Computing*, Berkeley TR 2019

Future of Serverless Computing

Opinion from Berkeley Report [1]

- Challenges: Abstraction, System, Networking, Security, Architecture
- Predictions: new BaaS, heterogeneous hardware, easy to program securely, cheaper, DB in BaaS, serverless replacing serverful

Opinion from a CIDR'19 Paper [2]

- Fluid Code and Data Placement
- Heterogeneous Hardware Support
- Long-Running, Addressable Virtual Agents
- Disorderly programming
- Flexible Programming, Common IR
- Service-level objectives & guarantees
- Security concerns

[1] E. Jonas, et al. *Cloud Programming Simplified: A Berkeley View on Serverless Computing*, Berkeley TR 2019

[3] Hellerstein, Joseph M., et al. "Serverless computing: One step forward, two steps back." *arXiv preprint arXiv:1812.03651*(2018).

Lambda Functions – Q/A

Replace S3 with a key-value store like Redis?

Solution limited to OLAP workload?

Apply Snowflake’s incremental clustering to serverless?

Cost models are specific to current serverless architecture.

Folding the optimizations into a query processing layer on S3?

Group Discussion

What are the fundamental advantages of implementing serverless databases using FaaS (i.e., Lambda functions) over BaaS?

- Are there fundamental reasons why an FaaS database should be cheaper than a BaaS database?

Next Lecture

Submit review for

- Yihe Huang, et al. Opportunities for Optimism in Contended Main-Memory Multicore Transactions, VLDB 2020