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
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

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.”

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

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

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

Function-as-a-Service (focus of this lecture)

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

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

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

**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

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

Table 1: Characteristics of function invocations.

A single driver can invoke 220–290 workers per second
Invoking 4096 workers takes 13–18 seconds
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]

Both optimizations lower the cost
With three-level exchange and write combining, the exchange cost becomes negligible compared to worker cost
Evaluation – Worker Configuration

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

Both Q1 and Q6 in TPCH only scan a single table
  • Q1 selects 98% of the input data
  • Q6 selects 2% of input data

Figure 9: Distribution of processing time.
Evaluation – Comparison with QaaS

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)

Figure 11: TPC-H queries on Lambada (M = 2 GiB).
Evaluation – Straggler in 2-level Exchange

Straggler effect is significant for large data set and large numbers of workers.
Starling: A Scalable Query Engine on Cloud Functions

Matthew Perron
MIT CSAIL
mperron@csail.mit.edu

David DeWitt
MIT CSAIL
david.dewitt@outlook.com

Raul Castro Fernandez
University of Chicago
raulcf@uchicago.edu

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:

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

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

Future of Serverless Computing

Opinion from Berkeley Report [1]
- 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

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?
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?
Submit review for

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