CS 839: Topics in Database Management Systems

Lecture 15: Serverless-4

Xiangyao Yu

10/25/2023

Group Discussion from Lecture #14

What are the key limitations of cloud functions today? What would it take, in your opinion, to resolve each of these limitations? (e.g., better industry implementations or fundamental research breakthroughs?)

Ideally, a cloud database should be truly serverless. Existing systems fall short realizing this goal. What do you think are the primary obstacles that must be overcome to realize this vision?

Group Discussion from Lecture #14

Limitations of existing cloud functions

- High data access latency -> data locality
- State persistence -> use external service (e.g., storage or database), or local medium of communication
- Low startup time -> Amazon Firecracker
- Short lifetime -> allow longer execution time
- Limited CPU and memory
- No inter-function communication
- Transaction support
- No support for special hardware
- Inefficient function composition

Group Discussion from Lecture #14

A new abstraction layer on top of cloud functions

- A new programming language
- Specify the requirements and dependency of data
- Optimize for data placement
- Better coordination in invoking cloud functions

Group Discussion from Lecture #14

Obstacle for an ideal serverless database

[Cloud function specific challenges]

- Hard to build a transactional database from cloud functions
- Cloud functions have low throughput
- Cannot implement locking using cloud functions. Need a locking service
- Data consistency and integrity while resources are ephemeral

[More general challenges]

- Resharding database based on the load and data
- Security and policy control (e.g., GDPR)
- A truly serverless database might be expensive

Serverless-4– Q/A

What makes a system not serverless? E.g., users choosing hardware? specifying cluster size?

When customers do not use functions, do cloud providers provision any resources?

PolarDB architecture is complex

Are cloud functions the right way to use specialized hardware like GPUs?

Discussion Questions

Today we program the cloud through a service abstraction—developers call APIs of different services (e.g., DB, storage, cloud function, containers, spark, etc.). This approach may not scale as the number of services increase in the future.

- How would developers program the cloud differently from today in ten years?
- What new abstraction are we going to create on top of existing services?
- Does it make sense to build an OS for the entire cloud?