Sanity: The Less Server Architecture for Cloud functions

Shripad J Nadgowda, Nilton Bila, Canturk Isci

IBM T J Watson Research Center
Agenda

• Background (check)
• Motivation (for Change)
• Sanity Architecture (What, Why and How?)
• Validation by Evaluation
• Take away Conclusion
• Open for Discussion
Getting on the same page

Event/Data Sources

Data Store

Compute Platform

Data Read/Write
Function trigger

f

D1

D2

R1

R2
Getting on the same page

Event/Data Sources

Compute Platform

Data Read/Write

Function trigger

D1 ~ D1’
D2 ~ D2’
Eureka Moment!

What if

Data from the input set are Equivalent And

associated functions Deterministic (or idempotent)

Then,

can we Avoid execution of functions, and still

De-duplicate the output results?
Sincere tribute

**Insanity:**
Doing same thing over and over again, and expecting different results

**Sanity:**
De-duplicate execution of cloud functions for equivalent data events
Validation: Equivalent data

• Common data sources for Cloud functions:
  – IoT/Sensor data (e.g. weather), social media (e.g. tweets), user-activity (e.g. click stream), system monitor data (e.g. Prometheus)

• Bounded range of values
  – E.g. temperature data to be (-20C to 50C)

• Temporal duplication
  – E.g. data from a fixed sensors, system monitors

• Spatial duplication
  – E.g. data from geo-distributed sensors

• Semantically equivalent data
Validation: Deterministic functions

An idempotent operation completes no more than one time.

Action semantics
The following sections describe details about Cloud Functions Actions.

Statelessness
Action implementations are stateless, or idempotent. While the system does not enforce this property, it is not guaranteed that any state maintained by an Action is available across invocations.

• Delivery of function invocations is not currently guaranteed. As the Cloud Firestore and Cloud Functions integration improves, we plan to guarantee "at least once" delivery. However, this may not always be the case during beta. This may also result in multiple invocations for a single event, so for the highest quality functions ensure that the functions are written to be idempotent.
Sanity: Less-server Architecture

Event/Data Sources

Data Store

Compute Platform

D1 ~ D1'  
D2 ~ D2'

R1'

R2'

Sanity Deduplication  Data Read/Write  Function trigger
Sanity: Less-server Architecture

Sanity Deduplication → Data Read/Write → Function trigger

Extending Sanity to Sequence of Cloud functions
Sanity Use case: Vulnerability Analysis

Container Data

- metadata {namespace...}
- file "/etc"
- {atime,mtime, ...}
- file "/var"
- {atime,mtime, ...}
- os "linux" {...}
- config "/etc/groups" {...}

......
Sanity: Mind the Gap...

- Considering ONLY storage-closed loop functions
  - reads data from storage
  - writes result back to the storage

- External stimuli are avoided
  - stimulate external events like sending email, slack, SMS etc.
Sanity Use case: Architecture

Data curation

Filtering → PoV Annotation → Checksum → Indexing → Serverless controller → Storage System

Sanity Controller

New data event

$f$
Sanity Use case: PoV based de-duplication

Data curation

Filtering → PoV Annotation → Checksum

Sanity Controller

Indexing → Short-circuiting → Serverless controller

Storage System

New data event

D
Sanity Use case: PoV based de-duplication

Original Data

```
{
  metadata:
    namespace: "dev/mysql",
    crawl-time: "2017-03-11T17:04:42"
  },
  file:
    name: "/etc/hosts",
    atime: "1459243509",
    mtime: "1459243509",
  packages:
    name: "coreutils",
    version: "0.5.8-2.1ubuntu2",
}
```

Filtered Data

```
{
  metadata:
    namespace: "dev/mysql",
    crawl-time: "2017-03-11T17:04:42"
  },
  packages:
    name: "coreutils",
    version: "0.5.8-2.1ubuntu2",
}
```

PoV annotated Data

```
{
  metadata:
    namespace: "$name$",
    crawl-time: "$crawl-time$
  },
  packages:
    name: "coreutils",
    version: "0.5.8-2.1ubuntu2",
}
```

MD5SUM

```
Original Data
Filtered Data
PoV annotated Data
```
Sanity Use case: Controller

Data curation

Filtering → PoV Annotation → Checksum → Sanit Control

New data event

Storage System

Sanity Control

Indexing

Short-circuiting

Serverless controller
Sanity Use case: Controller

2 GB Memory for 40K unique data entries
## Sanity Use case: Evaluation

<table>
<thead>
<tr>
<th>Computational task</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>PoV annotation and filtering</td>
<td>4.7 ms</td>
</tr>
<tr>
<td>SHA1 computation</td>
<td>0.05 ms</td>
</tr>
<tr>
<td>Indexing and output de-duplication</td>
<td>0.02 ms</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>5 ms</strong></td>
</tr>
</tbody>
</table>

Sanity Deduplication Overhead

<table>
<thead>
<tr>
<th>Computational task</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input data retrieval</td>
<td>772 ms</td>
</tr>
<tr>
<td>Function computation</td>
<td>333 ms</td>
</tr>
<tr>
<td>Output data storage</td>
<td>59 ms</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td><strong>1165 ms</strong></td>
</tr>
</tbody>
</table>

Function execution stats
Conclusion

Disaggregation with Cloud functions
• Data and Compute are managed independently

Data events are largely semantically equivalent
• Presenting an opportunity to de-duplicate data

Cloud functions are commonly deterministic
• Presenting an opportunity to de-duplicate data

• Cloud functions can be efficiently de-duplicated avoiding their redundant execution
• Scale serverless platform by requiring less-server
Thank You

Contact : nadgowda@us.ibm.com