Pixels: Using Cloud Functions as Accelerator for Elastic Data Analytics

Haoqiong Bian, Tiannan Sha, Anastasia Ailamaki

Cloud Resources for Query Processing: Cloud Function (CF) vs. Cloud Virtual Machine (VM)

CF is Popular for Query Processing in the Cloud
Features of CF:
+ Quick start-up: start-up hundreds of CFs in 1 second.
+ Easy to use: no hardware and software to manage (by users).
  - Limited size: e.g., up to 10GB memory and 6 vCPUs.
  - Limited lifetime: e.g., each instance lives up to 15 minutes.
  - No interconnection between instances.
  - No persistent local storage.

Query processing on CF is similar to that on MapReduce:

Insight 1: CF is More Expensive than VM
The unit prices of CF (e.g., Lambda) and VM (e.g., EC2 on-demand or spot) resources on AWS:

<table>
<thead>
<tr>
<th>Resource Type</th>
<th>Lambda</th>
<th>EC2 On-Demand</th>
<th>EC2 Spot</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU ($/core-h)</td>
<td>10</td>
<td>4.8</td>
<td>1.1</td>
</tr>
<tr>
<td>RAM ($/GB-h)</td>
<td>6</td>
<td>1.2</td>
<td>0.27</td>
</tr>
<tr>
<td>Network ($/Gbps-h)</td>
<td>85.71</td>
<td>15.36</td>
<td>3.53</td>
</tr>
</tbody>
</table>

Insight 2: CF is Less Scalable than VM
S3-based data shuffling and the lower scalability of S3 read bandwidth make CF-based query processing less scalable.

Insight 3: CF is More Elastic for Bursty Workload
CF provides shorter start-up time than VM (1 sec vs. 1-2 min). CF is suitable for processing unpredictable workload spikes.

Problem: How to improve cost-efficiency without compromising the elasticity.

Our Solution: Adaptive Query Processing on CF and VM

Our solution prioritizes processing queries in an auto-scaling VM cluster, and leverages CFs as accelerator (by sub-plan pushdown) to process the workload spikes if the VM cluster is unable to scale-out promptly.

Auto-scaling VM Cluster
We propose a framework that keeps the query concurrency stable by automatically adding or removing VMs to/from the VM cluster. It also handles the reclaim events of spot VMs.

Sub-plan Optimization and Execution
We propose a cost-based optimizer for the sub-plan in CF.
We propose sub-plan execution optimizations such as chain-joins:

Experimental Evaluation

Our solution (Hybrid) is more cost-efficient than the baselines without compromising elasticity (TPCH 1TB q12 + a real-world log analysis query).

EPFL

Pure VM-based query processing provides 1-2 orders of magnitudes higher cost-efficiency on sustained workload (TPC-H 1TB, 2TB RAM for each system except Athena).