CloudCluster

Unearthing the Functional Structure of a Cloud Service

Weiwu Pang (Google/USC)
Sourav Panda (Google/UCR)
Jehangir Amjad (Google)
Christophe Diot (Google)
Ramesh Govindan (Google/USC)
Cloud providers struggle to provide customers with insights on performance and reliability of cloud services
Cloud customers organize their services into *projects*
- e.g., front-end, ads subsystem, analytics backend

Each project consists of a number of virtual machines (VMs)
- Projects can be large (*tens of thousands* of VMs)
Providers divide cloud infrastructure **regions** and **zones**

Customers specify which region and zone each VM must be placed

Image from https://cloud.google.com/about/locations
Limited Visibility Across Customer-Provider Boundary

What the provider knows
VM location
VM name
VM-to-VM traffic (opt-in)

What the customer knows
Location of regions and zones
Makes it Hard to Diagnose Cloud Projects

- Are my VMs properly provisioned?
- Why did my client traffic shift to another zone?
- Can I reduce cost by reconfiguring my project?
Today: Monitoring at Different Levels of Abstraction

Introduction

Approach

Design

Evaluation

VM

Zone/Region

Services Accessed

Screen capture from Google GCP Platform
A New Abstraction: The Functional Structure

Introduction
Approach
Design
Evaluation

Functional Structure

Zone/Region

VM

Services Accessed

Screen capture from Google GCP Platform
A New Abstraction: The Functional Structure

**Introduction**

**Approach**

**Design**

**Evaluation**

**Architecture**

- Load Balancer Layer
  - Processing Layer

**Deployment**

- **West**
  - Load Balancer VMs
  - Processing VMs

- **East**
  - Load Balancer VMs
  - Processing VMs

**Functional Structure**
What do we mean by Functional Structure?

 grouping by location captures communication locality

 Large groups by function due to scale-out, replication, sharding etc.

 Communication graph of VMs grouped by function and location
Can it extract the functional structure using information it has?*

What the provider knows
VM location
VM name
VM-to-VM traffic (opt-in)

* With customer consent
Extracting the Functional Structure

VMs in cloud project

VMs grouped by function and location

Functional Structure

Problem: How do you determine if two VMs are functionally similar?
Hypothesis

From traffic logs, sampled and aggregated over an hour

Traffic Similarity $\Rightarrow$ Functional Similarity

VM-to-VM Traffic Matrix*

* From traffic logs, sampled and aggregated over an hour
Hypothesis

Traffic Similarity $\Rightarrow$ Functional Similarity

VM-to-VM Traffic Matrix*

Distance metric $d(\vec{u}, \vec{v})$ is small

* From traffic logs, sampled and aggregated over an hour
Approach: Use Clustering!

Cluster rows of VM-to-VM traffic matrix

Functional groupings should just emerge
CloudCluster: Challenges

- Inter-VM traffic can vary significantly
  By 5-6 orders of magnitude

- Cloud projects can be large
  Some exceed 10,000 VMs

- Number of clusters not known \textit{a priori}

Feature Scaling

- Dimensionality Reduction

Hierarchical Clustering
CloudCluster: Challenges

- Inter-VM traffic can vary significantly by 5-6 orders of magnitude.
- Cloud projects can be large, some exceeding 10,000 VMs.
- Number of clusters not known a priori.

Feature Scaling
Dimensionality Reduction
Hierarchical Clustering
The Problem

Cloud projects can have thousands of VMs

Distance measures on sparse high dimensional data can be noisy
The curse of dimensionality

Computational complexity increases exponentially with size of project

<table>
<thead>
<tr>
<th>Project</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10000+</td>
</tr>
<tr>
<td>B</td>
<td>5000+</td>
</tr>
<tr>
<td>C</td>
<td>500+</td>
</tr>
<tr>
<td>D</td>
<td>500+</td>
</tr>
<tr>
<td>E</td>
<td>5000+</td>
</tr>
<tr>
<td>F</td>
<td>5000+</td>
</tr>
<tr>
<td>G</td>
<td>5000+</td>
</tr>
<tr>
<td>H</td>
<td>1000+</td>
</tr>
</tbody>
</table>
Insight: Exploit Redundancy in VM-to-VM Traffic

Skewed singular value spectrum suggests that a low-rank approximation of the traffic matrix might suffice.
Dimensionality Reduction

Original TM

$n \times m$ $
\rightarrow$

Low-rank approximation

$n \times r$

$r \ll m$

Cluster rows of this matrix
CloudCluster: Challenges

- Inter-VM traffic can vary significantly
  By 5-6 orders of magnitude

- Cloud projects can be large
  Some exceed 10,000 VMs

- Number of clusters not known \textit{a priori}

- Feature Scaling
- Dimensionality Reduction
- Hierarchical Clustering
The Problem

Don’t know **number** of clusters *a priori*
Cannot use *k*-means

Density-based approaches (e.g., DBSCAN, OPTICS)
Require thresholds that depend on **domain knowledge**

Iterative methods (MeanShift, AffinityPropagation)
Can have **high computational complexity**
Agglomerative Clustering

Bottom-up approach
Recursively merge clusters

Results viewed as a dendrogram

Introduction  Approach  Design  Evaluation
Generating the Clusters: Where to cut the dendrogram?

Two Clusters
Generating the Clusters: Where to cut the dendrogram?

Four Clusters
Inconsistency: measure of difference between cluster and its sub-clusters
   If small, then can select the cluster

Inconsistency has a skewed distribution
   Cut at the “knee” of the distribution
Evaluating Clustering Performance

Ground-truth: Projects with known VM Functions

Metric: **Homogeneity**
Fraction of VMs in a cluster with the same label

Metric: **Completeness**
Fraction of ground-truth labels that fall into the same cluster

Real value metric with range [0, 1]
Analogous to **Precision** and **Recall** in classification

High **Homogeneity** and High **Completeness** → Good Agreement with Ground truth
CloudCluster: Excellent Agreement with Ground Truth

<table>
<thead>
<tr>
<th>Project</th>
<th>Size</th>
<th># Clusters</th>
<th>Homogeneity</th>
<th>Completeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10000+</td>
<td>72</td>
<td>0.984</td>
<td>0.966</td>
</tr>
<tr>
<td>B</td>
<td>5000+</td>
<td>206</td>
<td>0.919</td>
<td>0.950</td>
</tr>
<tr>
<td>C</td>
<td>500+</td>
<td>4</td>
<td>0.996</td>
<td>0.963</td>
</tr>
<tr>
<td>D</td>
<td>500+</td>
<td>3</td>
<td>1.000</td>
<td>0.938</td>
</tr>
<tr>
<td>E</td>
<td>5000+</td>
<td>60</td>
<td>0.965</td>
<td>0.940</td>
</tr>
<tr>
<td>F</td>
<td>5000+</td>
<td>177</td>
<td>0.937</td>
<td>0.949</td>
</tr>
<tr>
<td>G</td>
<td>5000+</td>
<td>8</td>
<td>0.996</td>
<td>0.971</td>
</tr>
<tr>
<td>H</td>
<td>1000+</td>
<td>6</td>
<td>0.996</td>
<td>0.996</td>
</tr>
</tbody>
</table>
What is the Functional Structure Useful For?

- Are my VMs properly provisioned?
- Why did my traffic shift to another zone?
- Can I reduce cost by reconfiguring my project?
Visualization

Load Balancers

Processing Nodes

Storage backend
Detecting Traffic Shifts with CloudCluster

Introduction  Approach  Design  Evaluation

Cloud Region 1  Cloud Region 2

External IPs
Detecting Traffic Shifts with CloudCluster
Detecting Structural Changes

Region 1

Region 2

VM Migration

Zone 2
Dataset: 25 known traffic shifts or structural changes
   Based on internal anomaly detector or trouble tickets

Result: Able to detect all events
Other Uses

Reducing cost
Up to 40% by reconfiguring VM placements to minimize inter-region traffic

Detecting mis-provisioned VMs
1% of VMs perform the same task with different VM capabilities

Detects mis-labeled VMs
7% of VMs likely incorrectly labeled
Cloud monitoring is hard
  Providers need novel ways to deliver insights

Provider can unearth functional structure
  ... by analyzing traffic patterns

Surprisingly strong agreement with ground truth

Functional structure can provide useful insights
Thank you