Clustering Performance Anomalies in Web Applications Based on Root Causes

Satoshi Iwata, Kenji Kono
Keio University
Performance Anomalies

- Unexpected, undesirable degradations in performance
  - Increases in response time
  - Decreases in server throughput

- Becoming a huge problem in web applications
  - Lead to violations of service level agreements (SLAs)
  - Lead to loss of potential customers due to lowered quality of services [Kohavi et al. ’07]

- Methods for analyzing root causes are required to recover systems

- Current methods assume single anomaly
  - Although many methods have been already proposed
Concurrent Anomalies

- Problems are more complicated
  - When more than one anomaly are detected at the same time

- We have to judge if their root causes are the same
  - As a first step toward hunting for root causes

- When R1 and R2 are detected as anomalous at the same time, we can consider two cases
  - C is anomalous
  - A and B are anomalous at the same time
Proposal

- We propose a method for clustering performance anomalies based on root causes
  - If R1 and R2 are clustered *together*
    - They are affected by the *same* root cause
  - If R1 and R2 are clustered *differently*
    - They are affected by the *different* root causes

- Although successively diagnosing root causes is required, this is beyond the scope of this work
The key insight

- Processing times affected by the same root cause deviate similarly

![Diagram showing CDFs for different causes with labels: Same Cause and Different Cause.](image)

CDFs from R1

CDFs from R2

CDFs from R3

: Increase of Processing Time
Three Steps for Clustering

1. Distill a *performance anomaly signature* from the measurements of each request type (R1 or R2)
   - The signature characterizes the deviation from the standard
   - We compare CDFs

2. Calculate a *similarity* of the signatures

3. Cluster request types based on the similarity
   - With group average method in hierarchical clustering
Anomaly Signature

- Our anomaly signature indicates how the distribution of the processing times has changed.
  1. Plot two inverse CDFs of the processing times.
  2. Simplify the inverse CDFs to ease calculation.
  3. Derive a performance anomaly signature from the simplified CDFs.
- Basically, signature similarity is overlap area of two signatures.
Case Study

- We determined root causes of performance anomalies with the help of clustering result
  1. Detected performance anomalies at the granularity of request types with the previous work [Iwata and Kono ’10]
      - 26 out of 27 were detected as anomalous
  2. Clustered 27 request types with our clustering method
  3. Hunted for root causes with the help of clustering result

- Our target was RUBiS [http://rubis.objectweb.org]
  - An auction prototype modeled after eBay.com
  - Has a typical three-tier architecture
  - Has 27 request types
      - E.g., Home, SearchItemsInCategory, PutBid
Clustering Result

- 27 request types were clustered into 2 clusters
  - Ca included 5 request types
  - Cb included 22 request types

We hunted for root causes with manual effort, guided by this clustering result.

Signatures typical for Ca

Signatures typical for Cb
Discovered Root Causes

- We could narrow down possible causes to each of Apache and JBoss
  - Request types
    - Ca
    - Cb
  - Apache
  - JBoss
  - MySQL

- Each default value of two settings were improper in our environment
  - KeepAliveTimeout in Apache
  - maxThreads in JBoss

<table>
<thead>
<tr>
<th>Request Types</th>
<th>Processing Times Before</th>
<th>Processing Times After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ca</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cb</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Processing Times
Related Work

- Methods for judging if two anomalies are similar to each other have been proposed
  - [Bodik et al. ‘10, Cohen et al. ’05, Yuan et al. ’06]
  - Those methods compare the current anomaly to a previous one
  - We must distinguish anomalies happening simultaneously

- Many methods have been already proposed for determining root causes of performance anomalies
  - [Bodik et al. ‘05, Aguilera et al. ’03, Chen et al. ‘04, Chanda et al. ’07]
  - Root causes of concurrent anomalies may be difficult to be determined
  - Our methods can be applied to the output of those methods
Summary

- We proposed a method which clusters performance anomalies based on root causes
  - Resultant cluster helps us determine root causes in the case of more than one anomaly are detected simultaneously
    - E.g., An operator can search for components exclusively used by each cluster

- The key insight behind our method
  - The measurements of anomalies that are negatively affected by the same root cause show a deviation similar to each other

- We are planning to extend our method to use other measurements
  - CPU loads, I/O rates