Automatic Grouping on Performance Investigation

Francisco de Melo
May 5th, 2017

École Polytechnique de Montréal
Laboratoire DORSAL
1 Outline

- Problem description
- ECCT Model
- Methodology
- Analysis techniques
- Utilization
- Use Case:
  - Description
  - Method
  - Results
- ECCTView
- Status
Problem Description

Compare several executions of the same software.

The current tools are limited to compare several executions of the same software or require human analysis to find root cause problems.
Construction of tree using Ittng and other techniques

UST level
Methodology

Overall view:

- Instrument the code
- Create the tree
- Apply the classification methods
Techniques

Support Vector Machine
k-means algorithm

▷ Auto grouping mechanism
Auto grouping

SSE heuristic

SSE vs Number of Groups

Number of groups

Average Dispersion
Utilization

This **automatic technique** can be applied for the following situations:

- Comparing C/C++ library performances
- Performance Regression investigation
- User Space Trace: required instrumentation
OpenCV

In a regression case in OpenCV, a later version of the **HoughLines** decreased the performance in the new version.

Function: **HoughLines**

Versions: 3.1 vs 3.0
Real use case

HoughLines
Run several times the process on both versions

Record the information for each run

Apply the grouping techniques

Group Comparison
## Results

Results: Classification in 2 groups

<table>
<thead>
<tr>
<th></th>
<th>FAST Group</th>
<th>Slow Group</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution</td>
<td>6500 Cache misses</td>
<td>2400 Cache misses</td>
</tr>
</tbody>
</table>

![Graph showing classification in 2 groups with FAST Group having 6500 cache misses and Slow Group having 2400 cache misses.](graph.png)
Version 3.0 to 3.1

250 commits > 4 lines of difference
10 Conclusion

Result

Cache misses different found as performance cause by comparing two the means of each metric

Lines of code reduction to 4 lines

Unit test to find cache misses
Automate Grouping

Take the best number of groups

Apriori Algorithm for more complex cases

Cross over of the groups to find root causes

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>X</td>
<td>75%</td>
<td>100%</td>
</tr>
<tr>
<td>B</td>
<td></td>
<td>75%</td>
<td>X</td>
</tr>
<tr>
<td>C</td>
<td></td>
<td>100%</td>
<td>65%</td>
</tr>
</tbody>
</table>

Comparison of groups

Compare them using a specific algorithm
ECCT View

(i) Display the tree
(ii) Display groups mechanisms
13 Status

Implemented the Tree [done]
Application of Grouping methods [done]
Apriori Method [done]
Displaying the groups and associations [doing]
References


Questions

isnaldo-francisco.de-melo-jr@polymtl.ca

Any other info?
Obrigado