Subsetting Big Data Workloads from BigDataBench

Zhen Jia

Institute of Computing Technology, Chinese Academy of Sciences

BigDataBench Tutorial
MICRO 2014 Cambridge, UK
Challenges in Understanding Big Data Apps

- A huge number of representative workloads
  - Hard to thoroughly understand behaviors
  - Prohibitively expensive for simulation-based research

- Having many software stacks aggravates the challenge
Revisit BigDataBench 3.0

- Include multiple software stacks
- Too time-consuming to run and analyze them all

BDGS (Big Data Generator Suite)

Six Real-world Data Sets

- Wikipedia Entries
- Google Web Graph
- E-commerce Transaction
- Amazon Movie Reviews
- Facebook Social Network
- ProfSearch Person resumes

Search Engine
OLAP
Offline Analytics
Interactive analytics

Algorithms or Operations

77 workloads

Software Stacks

Hadoop RDMA
Shark
Impala
Hive
MySQL
Oracle
NoSql
DataMP1
MVAPICH
MP1
Spark Lightning Fast Cluster Comp

BigDataBench MICRO 2014

INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES
Why do we consider different software stacks?

- Software stacks have significant impact on workload behaviors—even greater than benchmark algorithms [1]
  - Deep software stacks
  - Integrated mechanisms

Easy to write a big data app
App code << software stack

Goals

- Find a way to downsize BigDataBench 3.0
  - BigDataBench suite contains 77 workloads
  - Should be shrunk to a manageable number

- Reduce the evaluation time of the Big Data research
  - Especially for architecture research using simulator
Subsetting Methodology

--From a view of microarchitecture

1. Microarchitectural Metric Selection
2. Correlation Removing
3. Clustering
4. Choosing Representative Ones
Metric Selection

- 45 total metrics, including:
  - Instruction Mix
  - Cache Behavior
  - TLB Behavior
  - Branch Execution
  - Pipeline Behavior
  - Offcore Requests and Snoop Responses
  - Parallelism
  - Operation Intensity

- PMCs accessed via *perf*

- Hard to analyze 77 workloads with 45 metrics
Correlation

- Many program characteristics (metrics) are correlated
  - e.g. long latency cache misses => pipeline stalls

- Correlated data can skew analysis
  - May overemphasize a particular property’s importance
PCA

- Use PCA (Principal Components Analysis) to eliminate correlated data.
- PCA: A statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated.

- Principal Components Analysis (PCA) computes Principal Component ($Y_i$):
- Yis that are linear combinations of original metrics $x_i$ called PCs (Principal Components):
  - $Y_i = a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{ip}x_p ; i=1..p$
PC properties

- PCs are derived in decreasing order of importance, and they are orthogonal
  - First principal component is the direction of greatest variability (covariance) in the data
  - Second is the next orthogonal (uncorrelated) direction of greatest variability
- Keep PCs with eigenvalues >=1
  - Data is ensured to be uncorrelated while capturing most of the original information (Kaiser criterion)
Clustering

- Use K-means algorithm to partition workloads into $K$ clusters.

- The problem: how to choose $K$?
Use Bayesian Information Criterion (BIC) to choose proper K value

- Measures how well the clustering fits the data set
- Larger BIC scores are better
- We choose the K with highest BIC scores

\[ BIC(D, K) = l(D|K) - \frac{p_j}{2} \log(R) \]

\[ l(D|K) = \sum_{i=1}^{K} \left( -\frac{R_i}{2} \log(2\pi) - \frac{R_i \cdot d}{2} \log(\sigma^2) \right) - \frac{R_i - K}{2} + R_i \log R_i - R_i \log R \]

\[ \sigma^2 = \frac{1}{R - K} \sum_i (x_i - \mu(i))^2 \]
Selecting Representative Workloads

- Select workloads near the cluster center

- Select workloads near the cluster boundary
## Architecture Subset Workloads

<table>
<thead>
<tr>
<th>No. of cluster</th>
<th>Workload name</th>
<th>Number of workloads in cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Cloud-OLTP-Read</td>
<td>10</td>
</tr>
<tr>
<td>2</td>
<td>Hive-Difference</td>
<td>9</td>
</tr>
<tr>
<td>3</td>
<td>Impala-SelectQuery</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>Hive-TPC-DS-query3</td>
<td>9</td>
</tr>
<tr>
<td>5</td>
<td>Spark-WordCount</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>Impala-Orderby</td>
<td>7</td>
</tr>
<tr>
<td>7</td>
<td>Hadoop-Grep</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>Shark-TPC-DS-query10</td>
<td>4</td>
</tr>
<tr>
<td>9</td>
<td>Shark-Project</td>
<td>4</td>
</tr>
<tr>
<td>10</td>
<td>Shark-Orderby</td>
<td>3</td>
</tr>
<tr>
<td>11</td>
<td>Spark-Kmeans</td>
<td>1</td>
</tr>
<tr>
<td>12</td>
<td>Shark-TPC-DS-query8</td>
<td>1</td>
</tr>
<tr>
<td>13</td>
<td>Spark-Pagerank</td>
<td>1</td>
</tr>
<tr>
<td>14</td>
<td>Spark-Grep</td>
<td>1</td>
</tr>
<tr>
<td>15</td>
<td>Hadoop-WordCount</td>
<td>1</td>
</tr>
<tr>
<td>16</td>
<td>Hadoop-NaiveBayes</td>
<td>1</td>
</tr>
<tr>
<td>17</td>
<td>Spark-Sort</td>
<td>1</td>
</tr>
</tbody>
</table>
What do those 17 workloads do?

- Offline analytics:
  - Sort, Grep, Word Count, Page Rank, K-means, Bayes

- No-SQL operation: Read

- TPC-DS queries:
  - Query 3, 8, 10

- Basic relational algebra operations:
  - Difference
  - Select query to filter data
  - Sorting
  - Project
Further Work

- Workloads in Big Data change frequently
  - New workloads may be introduced
  - Out-of-date workloads will be removed

- Subsetting is a continuing process
  - The subset may change over time.
Thank You!