BigDataBench:
a Benchmark Suite for Big Data Application

Wanling Gao
Institute of Computing Technology, Chinese Academy of Sciences

HVC tutorial
in conjunction with The 19th IEEE International Symposium on High Performance Computer Architecture (HPCA 2013)
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• A Use Case
• Future Work
Requirements for Big Data Benchmark

• To truly reflect
  – Use-cases & requirements

• To rapidly evolve
  – With new workloads and use cases

• To widely cover
  – Application domains, data types and use-case scenarios

<<Proposal for a Big Data Benchmark Repository>>--Andries Engelbrecht . WBDB2012
Benchmark for Big Data: State of Practice

• Sort Benchmark
  – Only one application: MinuteSort, JouleSort, TeraByte Sort
    • One-fits-all solution?
      • Sort: mainly integer comparison operation and I/O bound pattern
  – Fixed data scale
  – Fixed data format
    • 100-byte input records with a 10-byte random key to be sorted for test.

http://sortbenchmark.org/

? = Sort
Benchmark for Big Data: State of Practice

• GridMix, Hadoop microbenchmark
  – Generating data set randomly

• Cons
  – Ignoring the characteristics of real-world data
Challenges

• Current State--**Immature**
  – “We Don't Know Enough to Make a Big Data Benchmark Suite”
    
    *An Academia-Industry View, Yanpei Chen, UC Berkeley/Cloudera WBDB2012*

• Our incremental solution
  – To single out foundation application in the most important domain
  – Then to expand
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• A Use Case
• Future Work
Benchmark overview

• Methodology

• BigDataBench
Methodology Overview

Step 1
Investigate Application Domains

Step 2
Choose Typical Workloads

Step 3
Create Big Data Benchmarks
Step 1: Investigate Application Domains

So many application domains

Which application domain should I choose?
Step 1: Investigate Application Domain

92% of online adults use search engines to find information on the web.

(pew internet study)
Step 1 (Cont’)

• Search Engine is a key data center application
  – 40% of top20 websites

http://www.alexa.com/topsites/global;0
Step 1 (Cont’)

Step 1
Investigate Application Domain

Search Engine

Step 2
Choose Typical Workloads

Step 3
Create Big Data Benchmarks
Step 2: Choose Typical Workloads

- Representative workloads in the most important application domain
- Widely used in other domains
- Having representative features
Step 2: Choose Typical Workloads

- Sort
- Typical
- Grep
- Wordcount
- Naïve Bayes
### Step 2: Representative Workloads in Search Engine

<table>
<thead>
<tr>
<th>Workloads</th>
<th>Roles in the search engine</th>
</tr>
</thead>
</table>
| **Sort**    | ➢ URL sorting  
              ➢ Word frequency sorting  
              ➢ Other sorting               |
| **Wordcount** | ➢ Word frequency count                                                    |
| **Grep**    | ➢ Abstracting content from HTML  
              ➢ Abstracting content from TextFile  
              ➢ String replacement             |
| **Naïve Bayes** | ➢ Web page classification  
                        ➢ News classification       |
## Step 2: Widely Used in Other Domains

<table>
<thead>
<tr>
<th>Workloads</th>
<th>Application Scenarios in Industry</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort</td>
<td>Pages sorting by its ID <em>(Web storage in database)</em></td>
</tr>
<tr>
<td>Wordcount</td>
<td>Calculating the TF-IDF base information, such as term frequency. Obtain the user operations count to analysis their social behavior <em>(in Wolfram Alpha)</em></td>
</tr>
<tr>
<td>Grep</td>
<td>Log analysis</td>
</tr>
<tr>
<td></td>
<td>Web information extraction</td>
</tr>
<tr>
<td></td>
<td>Fuzzy search</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>Spam recognition <em>(Spam Filtering)</em></td>
</tr>
<tr>
<td></td>
<td>Bioinformatics <em>(Naïve Bayesian Classifier for Rapid Assignment of rRNA Sequences into the New Bacterial Taxonomy)</em></td>
</tr>
</tbody>
</table>
### Step 2: Having Representative Features

<table>
<thead>
<tr>
<th>Workloads</th>
<th>Resource Characteristic</th>
<th>Computing Complexity</th>
<th>Instructions</th>
<th>Read/Write</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sort</td>
<td>I/O bound</td>
<td>$O(n \cdot \log n)$</td>
<td>Integer comparison domination</td>
<td>0.75</td>
</tr>
<tr>
<td>Wordcount</td>
<td>CPU bound</td>
<td>$O(n)$</td>
<td>Integer comparison and calculation domination</td>
<td>1.15</td>
</tr>
<tr>
<td>Grep</td>
<td>Hybrid</td>
<td>$O(n)$</td>
<td>Integer comparison domination</td>
<td>1.50</td>
</tr>
<tr>
<td>Naïve Bayes</td>
<td>/</td>
<td>$O(m \cdot n)$</td>
<td>Floating-point computation domination</td>
<td>1.70</td>
</tr>
</tbody>
</table>
Step 2 (Cont’)

Step 1
Investigate Application Domain

Step 2
Choose Typical Workloads

Search Engine

Sort
Wordcount
Grep
Naïve Bayes

Step 3
Create Big Data Benchmarks
Step 3: Big Data Puzzle

- Confidential for company
- Difficult to download
- Easily get small-scale data
Step 3: Generating Big Data

• To preserve the characteristics of real-world data

- Characteristic Analysis
- Expand

Small-scale Data → Big Data

Semantic

Usually less than 100MB
Word frequency

Locality

Temporally
Word reuse distance

Spatially
Dependency of words
Step 3: Create Big Data Benchmarks

- **Data Generation Tool**: Generate from discretionary small-scale data
- **Big Data**: Use HDFS as the storage
- **Representative workloads in Search Engine**

- **Tools**:
  - Sort
  - Wordcount
  - Grep
  - Naïve Bayes
Step 1
Investigate Application Domain

Search Engine

Step 2
Choose Typical Workloads

Sort
Wordcount
Grep
Naïve Bayes

Step 3
Create Big Data Benchmarks

Generate big data from small-scale data
BigDataBench Overview

• Methodology

• BigDataBench
BigDataBench

Data Generation Tool

Text Data

10GB-1PB

Format conversion

Sequence file

Sort

Wordcount

Grep

Naïve Bayes

Characteristic Analysis

Expand

Seed Data 1

Seed Data 2

Data Set 1

Data Set 2

The Same Characteristics
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• A Use Case
• Future Work
Use Guide

• Prerequisite Software
  – Java (version 1.6.0_20 and later)
  – Hadoop (version 1.0.2 and later)
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• A Use Case
• Future Work
Analyzing the Seed Data

- \$HADOOP_HOME/bin/hadoop jar count.jar arg1 arg2
  - arg1: input file, which saves the seed data
  - arg2: output file, which saves the characteristics of seed data

- An example
  - \$HADOOP_HOME/bin/hadoop jar count.jar /gwl/seed test
Generating Data

- $\text{HADOOP\_HOME/bin/hadoop jar TextProduce.jar arg1 arg2 arg3 arg4 arg5}$
  - arg1: input file, also the output file of first program
  - arg2: output file, which will save the new data
  - arg3: the numbers of types in the seed data
  - arg4: the numbers of news need to be generated
  - arg5: the groups of each type of news will be divided
About arg5

- The arg5 is adjustable as to the reduce slots in the cluster
- Usually, the more the number of reduce slots, the bigger the arg5 should be set

<table>
<thead>
<tr>
<th>Data Size</th>
<th>100G</th>
<th>1T</th>
<th>10T</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reduce slots</td>
<td>100</td>
<td>180</td>
<td>280</td>
</tr>
<tr>
<td>arg5</td>
<td>5</td>
<td>9</td>
<td>14</td>
</tr>
</tbody>
</table>
An Execution Example

- `$HADOP_HOME/bin/hadoop jar TextProduce.jar bayes-input file-100G 20 75000000 5`

  - **Program Name:** The file which stores the characteristics of seed data
  - **Output File:** The filename of the output file
  - **Number of Types:** the numbers of types in the seed data
  - **News Need to Be Generated:** the numbers of news need to be generated. Generally, 75- million represents 100GB

Generally, 75- million represents 100GB
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• Use Cases
• Future Work
How to Convert the Data Format

- sort-transfer.sh arg1 arg2
  - arg1: the input file, which contains the original data
  - arg2: the output file, which saves the conversion data
[root@hw084 gw1]# ./sort-transfer.sh /user/root/baves-out-100G 
/user/root/seq-out-100G

Deleted hdfs://172.18.11.84:9000/user/root/seq-out-100G
13/02/05 17:34:11 WARN mapred.JobClient: No job jar file set. User classes 
may not be found. See JobConf(Class) or JobConf#setJar(String).
****hdfs://172.18.11.84:9000/user/root/baves-out-100G
13/02/05 17:34:11 INFO input.FileInputFormat: Total input paths to process : 
280
13/02/05 17:34:11 INFO util.NativeCodeLoader: Loaded the native-hadoop 
library
13/02/05 17:34:11 WARN snappy.LoadSnappy: Snappy native library not loaded
13/02/05 17:34:12 INFO mapred.JobClient: Running job: job_201302042231_0006
13/02/05 17:34:13 INFO mapred.JobClient: map 0% reduce 0%
13/02/05 17:34:30 INFO mapred.JobClient: map 1% reduce 0%
13/02/05 17:34:31 INFO mapred.JobClient: map 2% reduce 0%
13/02/05 17:34:32 INFO mapred.JobClient: map 3% reduce 0%
13/02/05 17:34:33 INFO mapred.JobClient: map 4% reduce 0%
13/02/05 17:34:35 INFO mapred.JobClient: map 6% reduce 0%
13/02/05 17:34:37 INFO mapred.JobClient: map 7% reduce 0%
13/02/05 17:34:39 INFO mapred.JobClient: map 8% reduce 0%
13/02/05 17:34:42 INFO mapred.JobClient: map 9% reduce 0%
13/02/05 17:34:44 INFO mapred.JobClient: map 10% reduce 0%
Choosing Among Four Workloads

- How to benchmark
  ① run-sort.sh arg
  ② run-wordcount.sh arg
  ③ run-grep.sh arg
  ④ run-bayes.sh arg
  (arg: the data scale)
[root@hw084 gwl]# ./run-sort.sh 100G
[root@hw084 gwl]# ./run-bayes.sh 100G
MAHOUT_LOCAL is not set; adding HADOOP_CONF_DIR to classpath.
Running on hadoop, using HADOOP_HOME=/opt/lzg/hadoop-1.0.2
HADOOP_CONF_DIR=/opt/lzg/hadoop-1.0.2/conf
MAHOUT-JOB:
/opt/lzg/mahout-distribution-0.6/mahout-examples-0.6-job.jar
13/02/03 15:09:27 WARN mapred.JobClient: Use GenericOptionsParser
for parsing the arguments. Applications should implement Tool
the same.
13/02/03 15:09:28 INFO util.NativeCodeLoader: Loaded the
native-hadoop library
13/02/03 15:09:28 WARN snappy.LoadSnappy: Snappy native librar
not loaded
13/02/03 15:09:28 INFO mapred.FileInputFormat: Total input pat
to process : 280
13/02/03 15:09:30 INFO mapred.JobClient: Running job: job_201302031457_0002
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• Use Cases
• Future Work
Use Case 1: Optimizing the Hadoop

I have 64 hadoop nodes to process 100TB data, but how to configure the parameters?
Use Case 1: Optimizing the Hadoop

Configuration 1
- Map slot
- Reduce slot
- Java_opts
- ...

BigData Bench under 100TB

V.S

Optimal configuration of Hadoop under 100TB

Result 1
- User observation metrics

Result 2

Configuration 2
Use Case 2: OS Performance

I want to build a brand new OS for big data!

So, which part of kernel should I optimize?

How to evaluate the OS kernel?
Use Case 2: OS Performance

Step 1: Big Data Selection
- Text data
- Webpage data
- Ranking data
- Trading data

Step 2: Increasing Data Set
- Text data
- Webpage data
- Ranking data
- Trading data

Step 3: Fetch OS Performance Metric
- I/O Capacity
- Latency
- Scheduling
- Bandwidth
- Interrupt Handler
Outline

• Motivation
• BigDataBench Overview
• Usage Guide
  – Generating Big Data
  – Configuring Workloads
• A Use Case
• Future Work
Future Work

• To include workloads in other important domains
  – Recommendation system
  – Graph computing
  – etc

• Release BigDataBench soon on http://prof.ict.ac.cn/ICTBench

• Please give us your email.
  – We will notify you the updates.
Thank you!

gaowanling@ict.ac.cn