BiDAi: Big Data Analyzer for Cluster Traces

Alkida Balliu, Dennis Olivetti, Ozalp Babaoglu, Moreno Marzolla, Alina Sirbu

Department of Computer Science and Engineering
University of Bologna, Italy
Talk Outline

- Motivation
- BiDAI
- Case Study
- Conclusions
Motivations

- Modern datacenters produce huge amounts of data in the form of event and error logs.
- Understanding the logs is essential to identify problems or improve efficiency.
  - Understand and exploit hidden patterns and correlations.
  - First step towards self-managing, self-healing datacenters.
Challenges

• Huge size of logs
  – A 2010 study [Thusoo et al, SIGMOD'10] reports that Facebook data centers produced 60TB of logging information daily

• Log analysis falls within the class of Big Data applications
  – Data sets are so large that conventional storage and analysis techniques are not appropriate to process them
BiDAI
Big Data Analyzer

• Java application (with GUI)
  – Proof-of-concept
• Typical workflow:
  – Instantiation of a storage backend
  – Data selection and aggregation
  – Data analysis
BiDAI
Big Data Analyzer

- Can import raw data in .CSV format
- Uses SQLite or Hadoop File System (HDFS) as storage backends
  - Additional storage types can be added
  - Although the current storage backends are based on the concept of “table”, other backends could be used too, e.g., HBase for <key, value> pairs
- Uses (a subset of) SQL as the query and data manipulation language
  - Translates SQL to the language understood by the storage backend – currently RSQLite or RHadoop
BiDAI
Big Data Analyzer

• Statistical computations can be performed using either R or Hadoop MapReduce
  – R commands are usually applied to the SQLite storage, while MapReduce commands are usually applied to the HDFS storage
  – BiDAI can transfer data automatically and transparently between the backends, to allow both languages to operate on both backends

• Computations can be concatenated
  – Usually, a data reduction is followed by the computation of some statistics
Data flow in BiDAI

CSV -> Import -> HDFS -> Transfer -> SQLite

R -> Execute -> RSQLite -> Convert

MapReduce -> Execute

SQL -> Convert -> RHadoop
Tables:

| Machine EVENTS | @/home/me/Desktop/Storages/Google Storage | SQL
| Machine Downtime | @/home/me/Desktop/Storages/Google Storage | SQL

Table preview:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>7</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>13</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>14</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>19</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>21</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>23</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>25</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Commands:

- export
- rutils
- spline @/home/me/Desktop
- exponential_distribution
- aggregate @/home/me/Desktop
- filter @/home/me/Desktop
- get column @/home/me/Desktop
- lognormal_distribution
- polynomial_regression
- spline.ecdf @/home/me/Desktop
- ecdf @/home/me/Desktop
- histogram @/home/me/Desktop
- spline function export
- spline.function.export

Current commands:

```r
ggplot(filename="temp.png");
plot(ecdf(t[1]));
dev.off();
```

Temp result:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
</tr>
</thead>
<tbody>
<tr>
<td>336</td>
<td>1</td>
</tr>
<tr>
<td>1011</td>
<td>0</td>
</tr>
<tr>
<td>136</td>
<td>0</td>
</tr>
<tr>
<td>594</td>
<td>0</td>
</tr>
<tr>
<td>5462</td>
<td>1</td>
</tr>
<tr>
<td>4957</td>
<td>0</td>
</tr>
<tr>
<td>7553</td>
<td>0</td>
</tr>
<tr>
<td>153</td>
<td>0</td>
</tr>
<tr>
<td>1032</td>
<td>0</td>
</tr>
<tr>
<td>397</td>
<td>0</td>
</tr>
</tbody>
</table>
Case Study: Google Traces

• The development of BiDAI was initially motivated by the need to analyze Google traces
  – https://code.google.com/p/googleclusterdata/

• Goal:
  – Extract workload parameter
  – Instantiate a simulation model of the Google cluster
  – Validate the simulation with respect to the observed data
Google traces

- Contain 29 days of information from May 2011, on a cluster of about 11k machines
  - Machine event (e.g., new machine is added to the pool, ...)
  - Machine attribute (e.g., OS is updated to a newer version, ...)
  - Jobs and Tasks (requirements, submit/completion time...)
  - Resource usage (sampled at some fixed intervals)
- Total size of the compressed trace is ~40GB
- https://code.google.com/p/googleclusterdata/
Entities of the Simulation Model

- Tasks and Jobs
- Arrival
  Process that generates new events (new job, new machine, machine removal...)
- Scheduler
  Decides where the tasks of a job can be executed
- Machines
  Execute tasks; notify the scheduler when a task terminates; send status updates to the scheduler
- Network
  Allows other entities to communicate
Trace-Driven Simulation Results

Number of Tasks Completed (15 min window, exponential smoothing)

#Tasks Successfully completed (Real)
#Tasks Successfully completed (Simulation)
Trace-Driven Simulation Results

Number of Tasks Evicted (15 min window, exponential smoothing)

#Tasks Evicted (Real)  
#Tasks Evicted (Simulation)
Workload Characterization

• We used BiDAI to extract workload parameters from the traces
  – Jobs Inter-arrival time distribution
  – Number of tasks per job
  – Distribution of execution times of different types of jobs (e.g., jobs that terminate successfully, jobs that are aborted by the user, …)
  – ...

Examples

Frequencies of the **amount of RAM** used by tasks (left) and **number of tasks per job** (right)
Examples

- Machine update events

Left
- Density and CDF with lines representing exponential fitting

Right
- Goodness of fit in Q-Q and P-P plots (straight lines denote perfect fit)
Examples

CDFs fitted by a sequence of splines: **CPU task requirements** (left) and **machine downtime** (right)
### Results using synthetic traces

<table>
<thead>
<tr>
<th></th>
<th>Real</th>
<th>Simulated</th>
<th>Rel. dif.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Running</td>
<td>124217</td>
<td>136037</td>
<td>0.09</td>
</tr>
<tr>
<td>Ready</td>
<td>5987</td>
<td>5726</td>
<td>0.04</td>
</tr>
<tr>
<td>Completed</td>
<td>3277</td>
<td>2317</td>
<td>0.29</td>
</tr>
<tr>
<td>Evicted</td>
<td>1057</td>
<td>2165</td>
<td>1.04</td>
</tr>
</tbody>
</table>

Can be explained by the high variance of real data
Conclusions and Future Works

- Big Data Analyzer (BiDAI) is a prototype data analysis tool that can handle large datasets
  - SQL, R, Hadoop/MapReduce
  - Extensible
- We used BiDAI to analyze the Google traces dataset
- Future works
  - Support additional storage backends
  - Include additional analysis algorithms (e.g., predictive algorithms, machine learning)
  - Live log analysis
Thanks for your attention!

http://www.cs.unibo.it/~sirbu/bidal.zip