Resource and Performance Distribution Prediction for Large Scale Analytics Queries

Prof. Rajiv Ranjan, SMIEEE
School of Computing Science, Newcastle University, UK
Visiting Scientist, Data61, CSIRO, Australia
Guest Professor, Chinese University of Geosciences
Heterogeneous Programming Models running on Big data Cluster

- Resource provisioning
- Workload scheduling
- Admission control
Motivation

Template-7 (Q7) of TPC-H against 100GB database size.

The histograms for 30 instance queries based on Q7
Goal

• Resource and Performance *Distribution* Prediction For *Hive Queries*
Approach Overview

• To **predict** performance **distribution** of Hive workloads, we use knowledge of **Hive query execution** combined with **machine learning** techniques.
Hive: data warehousing application in Hadoop

- Query language is HQL, variant of SQL
- Tables stored on HDFS as flat files
- Developed by Facebook, now open source
Query Processing in Hive

- Hive looks similar to an SQL database
  - SQL specific operators (e.g. table scan, select) implemented in map and reduce functions
  - MapReduce specific tasks (e.g., read, spill, shuffle, write)
  - End-to-End execution time depends on the number of mappers and reducers and their runtime performance.

Hive components

```
TABLE customer
    (customer_id   BIGINT,
     gender      STRING,
     ...)
```

SELECT *
FROM customers
WHERE gender = 'M';
## Feature list for training the model

<table>
<thead>
<tr>
<th>Feature Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQL Operator No</td>
<td>Number of SQL operators (e.g. Table Scan) which appear in the HiveQL query plan.</td>
</tr>
<tr>
<td>SQL Operator Input Records</td>
<td>Total number of rows affected by each operator in the query plan (e.g., a query operator uses 1000 rows to answer the query)</td>
</tr>
<tr>
<td>SQL Operator Input Byte</td>
<td>Input Data Size to SQL operator.</td>
</tr>
<tr>
<td>MapReduce Operator No</td>
<td>Number of MapReduce operators (e.g. Reduce Output Operator), appear in the HiveQL query plan.</td>
</tr>
<tr>
<td>MapReduce Operator Input Records</td>
<td>Total number of rows processed by each mapper/reducer.</td>
</tr>
<tr>
<td>MapReduce Operator Input Byte</td>
<td>Input Data Size to the MapReduce specific workflow steps (e.g. reading, spilling, shuffling, writing)</td>
</tr>
</tbody>
</table>
Feature Selection → Model Selection → Training and Testing
Mixture Density Network

- **MDN** = Neural Network + Mixture Model
- MDN uses Gaussian mixture model with multilayer perceptron
- Neural Network: $x \rightarrow$ mixture model ($\mu$, $\sigma$, $\alpha$)
  - Returns the conditional distribution $p(t|x)$
Training and Testing: **Workload**

- The data set we used contains 995 queries that were generated based on TPC-H benchmark.
- TPC-H queries were executed on six scaling factors: 2, 5, 25, 50, 75, and 100 GB.
- We divided the workload randomly into training and testing datasets with 66% and 34% respectively.
- We use a Netlab toolbox which is designed for the simulation of neural network algorithms and related models, in particular MDN.
Experiment: Setup

- The models are evaluated on CSIRO Big Data cluster. The cluster comprises of 14 worker nodes.

- All experiments were run on top of HiveQL 0.13.1, and Hadoop 2.3.0 in Yarn mode on.

- The cluster comprises of 14 worker nodes connected with fast Infiniband network, each featuring 2 x Intel Xeon E5-2660 @ 2.20 GHz CPU (8 cores), 128 GB RAM and 12 x 2 TB NL-SAS HD making up the total disk space of 240 TB.
Experiment: \textbf{Error metrics}

- continuous ranked probability score (CRPS)
  \[
  CRPS(F,t) = \int_{-\infty}^{\infty} [F(x) - O(x,t)]^2 \, dx
  \]

- negative log predictive density (NLPD)
  \[
  NLPD = \frac{1}{n} \sum_{i=1}^{n} - \log(p(t_i | x_i))
  \]

- root mean-square error (RMSE)
  \[
  RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (t_i - m_i)^2}
  \]
Experiment: **State of the art techniques**

- Support Vector Machine (SVM)
- REPTree
- Multilayer Perceptron
## Experiment: Results

### Accuracy of the Model

<table>
<thead>
<tr>
<th></th>
<th>MDN</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Target</td>
<td>CRPS</td>
<td>RMSE</td>
</tr>
<tr>
<td>CPU Time</td>
<td>0.024</td>
<td>0.08</td>
</tr>
<tr>
<td>Response Time</td>
<td>0.017</td>
<td>0.073</td>
</tr>
</tbody>
</table>

- MDN accuracy as per distribution specific metric error
- MDN accuracy compared to competing SVM model
Experiment: Results

• Training time of the MDN Model
  • Training times in seconds with regard to different workload sizes for 500 iterations.

<table>
<thead>
<tr>
<th>Workload Size</th>
<th>1K</th>
<th>2K</th>
<th>4K</th>
<th>8K</th>
<th>16K</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elapsed Time (Sec)</td>
<td>1.47</td>
<td>1.9</td>
<td>2.63</td>
<td>3.84</td>
<td>7.83</td>
</tr>
</tbody>
</table>
Experiment: **Results Summary**

In summary, our approach *outperforms* the state of the art single point techniques in 2 out of 4 experiments conducted using SVM and REPTree.

This result is quite promising because it shows that our approach is *not only* able to predict the **full distribution** over targets accurately, it is *also* a **reliable single point estimator**.
Wrap up

We presented a novel approach of using

**Mixture Density Networks**

for **Performance Distribution Prediction**

of Hive Queries

For future work:

Distribution-based Admission Controller and
Query Scheduler
Thank You!

• Questions...