Selection of Virtual Machines Based on Classification of MapReduce Jobs

Adam Pasqua Blaisse, Zachary Andrew Wagner, and Jie Wu

Department of Computer and Information Sciences, Temple University
Cloud Computing

• Large number of physical machines (PM)
• Strongly networked together
• Resources sold on an hourly basis as virtual machines (VM)
• Eucalyptus
• Amazon EC2
Motivation

• Find the minimal virtual machine that will run a Map Reduce job as fast as possible
Map Reduce

- Programming Paradigm for distributed computing
- Two phases
  - Map Phase
  - Reduce Phase
- Apache Hadoop
  - Open source implementation used
Hadoop Implementation of Map Reduce

- **Map**
  - Many small Map tasks
  - Each task takes a small chunk of data
  - Turn the data into Key value pair (i.e. `<the,1>`)
  - Number of Map tasks varies based on input data size
  - When all Map task are finished data is Pasted to the Reduce Phase

- **Reduce**
  - Very few set number of Reduce tasks
  - Combine all the input key value pairs from the maps
  - Also takes care of shuffling data from Map Locations to Reduce Locations
Hadoop Implementation of Map Reduce

- Reduce
  - All Mapping must finish before Reducing can start
  - Shuffling can start before Mapping ends
Issues when Used Together

• Some jobs run better on different configurations of virtual machines
• Different configurations of virtual machines have different costs
• Some jobs may need more CPU’s while others may need I/O
• I.E Generating Data is I/O intense, and would be best run on Memory rich system
Our Approach

• Attempt to classify tasks into two types
  • CPU Bound Jobs
    • Jobs spent more time doing CPU work then I/O
    • Jobs need more CPUS’s and less I/O
    • Smaller more numerous machines
  • I/O Bound Jobs
    • Jobs spent more time doing I/O work then CPU
    • Jobs need more I/O and less CPU
    • Less Larger Machines
Mapping to machines

• If a job is classified as
  • CPU Bound Job
    • Many virtual machines
    • Little memory per virtual machine
  • I/O Bound Job
    • Fewer virtual machines
    • Each virtual machine has larger amounts of memory
Why?

• If a job is I/O bound
  • Would like to keep job running in memory rather than hit HDD
  • I/O more important than number of cores

• If a job is CPU bound
  • More important to have many cores running the maps
  • Less likely to hit HDD while running
TCloud (Virtual Cluster)

- **Hardware**
  - 12 Dell Power Edge R614 Servers
  - 96 conventional CPU Cores
  - 4-Way redundant 10 GB Ethernet
  - 2-Way redundant InfiniBand

- **Software**
  - Eucalyptus 3.3 (Amazon EC2 compatible)

- Public cloud used to create virtual machine clusters
Net Cloud (Physical Cluster)

- **Hardware**
  - 32 Dell PowerEdge R210 servers
  - Each server has
    - 4 GB of RAM Memory
    - 500 GB HDD

- **Software**
  - Hadoop version 1.2.1
  - CentOS 6.6

- Physical machine cluster used for prediction
Net Cloud (continued)

- Networking
  - Tree like structure
  - 4 machines to 1 group switch
  - 4 group switches to 1 rack switch
  - 2 rack switches connected to 1 Top Switch
How to classify

• Metrics
  • Shuffle_bytes
  • CPU_time

• (Shuffle_bytes/CPU_time)
  • Take the average of the map tasks
  • If value is over 1, then job is I/O Bound
  • Else CPU Bound
Results from Physical Machine runs

### Shuffle Bytes per CPU (ms)

<table>
<thead>
<tr>
<th>Task</th>
<th>HDFS Bytes Written</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word Count</td>
<td>133.4174082</td>
</tr>
<tr>
<td>Pi</td>
<td>0.109941888</td>
</tr>
<tr>
<td>Pentomino</td>
<td>0.019261066</td>
</tr>
<tr>
<td>TeraGen</td>
<td>9328.415925</td>
</tr>
<tr>
<td>TeraSort</td>
<td>0</td>
</tr>
<tr>
<td>Grep</td>
<td>0.036108882</td>
</tr>
<tr>
<td>MRBench</td>
<td>0.008636364</td>
</tr>
<tr>
<td>DFCIOTest Read</td>
<td>0.042364532</td>
</tr>
<tr>
<td>DFCIOTest Write</td>
<td>0.038053097</td>
</tr>
</tbody>
</table>
Results on the Virtual Clusters

<table>
<thead>
<tr>
<th>Job</th>
<th>I/O Bound System (S)</th>
<th>CPU Bound System (S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word Count</td>
<td>257.2338</td>
<td>235.2299</td>
</tr>
<tr>
<td>PI</td>
<td>473.3364</td>
<td>419.88242</td>
</tr>
<tr>
<td>Pentomino</td>
<td>408.1599</td>
<td>355.0055</td>
</tr>
<tr>
<td>TeraSort</td>
<td>603.9358</td>
<td>183.1389</td>
</tr>
<tr>
<td>TeraGen</td>
<td>89.2324</td>
<td>116.62483</td>
</tr>
<tr>
<td>Grep</td>
<td>217.8305</td>
<td>188.0857</td>
</tr>
<tr>
<td>MRBench</td>
<td>21.0116</td>
<td>18.6668</td>
</tr>
<tr>
<td>DFSCIOTest read</td>
<td>24.5882</td>
<td>19.5072</td>
</tr>
<tr>
<td>DFSCIOTest write</td>
<td>25.2971</td>
<td>20.2712</td>
</tr>
</tbody>
</table>
Conclusion

• Presented a method for selecting virtual machines
• Showed the intuition behind the selection process
• Tested the method on two test beds at Temple

Future works
• Finding a good constant to multiply by to for a cluster
• Including more types of virtual machines
• Including more metrics for prediction
Questions?

• Contact
  • Adam.blaisse@temple.edu
  • Astro.temple.edu/~tuc47904