Locality-Aware Dynamic VM Reconfiguration on MapReduce Clouds

Jongse Park, Daewoo Lee, Bokyeong Kim, Jaehyuk Huh, Seungryoul Maeng
Virtual Clusters on Cloud

- Private cluster on public cloud
  - Distributed computing platforms
    - MapReduce [OSDI '04], Hadoop, Dryad [Eurosys '07]
    - New York Times used 100 nodes on Amazon EC2 [OSDI '08]
  - Each VM in a virtual cluster has static configuration

```
virtual cluster
```

```
e.g. Amazon EC2 VM instance types

<table>
<thead>
<tr>
<th>Instance types</th>
<th>Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Small</td>
<td>1 virtual core, 1.7GB memory</td>
</tr>
<tr>
<td>Large</td>
<td>2 virtual cores, 7.5GB memory</td>
</tr>
<tr>
<td>Extra Large</td>
<td>4 virtual cores, 15GB memory</td>
</tr>
</tbody>
</table>
```
Resource Utilization Management

- **Physical cluster**
  - Load balancing is the only mechanism for higher utilization

- **Virtual cluster**
  - Dynamic resource management is also possible
    - With using resource *hot-plug* technique
    - Possible resource types: *core* and *memory*

We focus on *core* hot-plugging in this work
Dynamic Resource Management

physical cluster

virtual cluster

Physical Machine

Virtual Machine

virtual machine
Management by Whom?

- **Requirements**
  1. Current resource utilization monitoring
  2. Platform-level information
  3. Privileged permission to hot-plug resource
  4. Support management for multiple users

- **Resource management as Platform-as-a-Service (PaaS) service**
  - Provider offers platform with dynamic resource management for various users
  - e.g. Amazon Elastic MapReduce
MapReduce
Disadvantages from low data locality

1. Network performance degradation because of network bottleneck
2. Under-utilization of computing resource
Hadoop Fair Scheduler

- Hadoop
  - Open source implementation of MapReduce
- Hadoop Fair Scheduler
  - Generally used scheduler
  - Guarantee fairness between submitted jobs on Hadoop

<table>
<thead>
<tr>
<th>Job</th>
<th># Running tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>job A</td>
<td>2</td>
</tr>
<tr>
<td>job B</td>
<td>0</td>
</tr>
</tbody>
</table>

Diagram:
- Master
- Slave 1
- Task 1
- Slave 2
- Task 1
- Job A
- Job B
Main Idea

- Approach
  - Move available resource to a node satisfying data locality and assign a task to the node
Dynamic Resource Reconfiguration

1. A node (source node) does not satisfy data locality
2. Master schedule to another node (target node) satisfying data locality
3. Reconfigure both source and target nodes
Dynamic Resource Reconfiguration

- **Resource hot-plugging**
  - **De-allocation**
    - Giving up and giving back resource to provider
    - Always possible
  - **Allocation**
    - Taking new resource from provider
    - Not always possible

- **Two solutions**
  - Synchronous DRR
  - Queue-based DRR
Synchronous DRR

- **Headroom**
  - Remained by provider
  - Idle and available resource on each physical machine
  - Shared by all VMs on a physical machine
Queue-based DRR

1. Reconfiguration from \textit{vm A} to \textit{vm C}
2. Reconfiguration from \textit{vm D} to \textit{vm B}
1. Reconfiguration from \textit{vm A} to \textit{vm C}
2. Reconfiguration from \textit{vm D} to \textit{vm B}
3. Reconfigure (\textit{vm A}, \textit{vm B}) and (\textit{vm C}, \textit{vm D})
Synchronous vs. Queue-based DRR

- **Synchronous DRR**
  - No waiting time until reconfiguration
  - Synchronously executed allocation and deallocation
  - Overall resource under-utilization because of headroom

- **Queue-based DRR**
  - Realistic and industry-applicable mechanism
  - Performance degradation if queuing delay is large
Evaluation

- Environment
  - EC2 cluster: 100 VM instances
    - 8 virtual cores, 7 GB memory (High-CPU Extra Large Instance)
    - Synchronous DRR only
  - Private cluster: 30 VMs on 6 physical machines
    - 6 cores, 16GB memory
    - Synchronous DRR + Queue-based DRR

- Workloads
  - Hive performance benchmark
    - grep, select, join, aggregation, inverted index

- Job schedule
  - Randomly generated schedule based on the trace of the industry [Eurosys'10]
Large-scale Evaluation

- Overall speedup: 15%

Graphs showing locality (%) and speed-up for different workloads and number of map tasks, comparing Original Hadoop and Synchronous DRR.
Evaluation on the Private Cluster

**Locality (%)**

- Original Hadoop
- Synchronous DRR
- Queue-based DRR

**Speed-up**

- Synchronous DRR
- Queue-based DRR

**Workloads, # of map tasks**

- (Grep.1)
- (Select.2)
- (Grep.5)
- (Select.10)
- (Grep.20)
- (Aggre.50)
- (Select.100)
- (Join.200)

- Overall speedup
  - Synchronous DRR: 41%
  - Queue-based DRR: 35%
Conclusion

- Propose a dynamic VM reconfiguration mechanism for distributed data-intensive platforms on virtualized cloud environment.

- Improve the input data locality of a virtual MapReduce cluster, by temporarily increasing cores to VMs to run local tasks, and it is called Dynamic Resource Reconfiguration (DRR).