CSC2231 Project Proposal

SnowFlock-Aware Hadoop Implementation

(http://www.cs.toronto.edu/~mjulia/CSC2231Project/SnowFlockAwareHadoop.html)

Shahan Khatchadourian and Julia Rubin

MapReduce [1] is a framework for data-intensive distributed computing of batch jobs. It allows programmers to think in a *data-centric* fashion: they focus on applying transformations to sets of data records, and allow the details of distributed execution, network communication and fault tolerance to be handled by the MapReduce framework. The MapReduce framework became prevalent due to its simplicity as a cloud programming model.

Hadoop, a popular implementation of the MapReduce framework [2], is commonly installed on a shared hardware controlled by virtual machine monitors (*Cluster Setup* Hadoop installation [3]). Such installations require identification and configuration of all machines in the cluster upfront. Adding a new machine to the cluster involves additional installation steps performed by a cloud administrator – a process that might take significant time ("minutes", according to [4]). Additionally, a job's configuration needs to be updated and may require the job itself to be restarted.

While Hadoop does not provide a way to immediately provision additional machines as needed, it allows controlling cluster machines utilization by providing explicit configuration options which control the number of spawned map and reduce steps for each job. Enhancing Hadoop with an explicit control over the number of virtual machines that are available to each job can provide even better machines utilization and, at the same time, decreased processing time, without the need to modify Hadoop command that the users are experienced with. Towards this end, we propose to integrate Hadoop with SnowFlock [5] – a system that allows Xen virtual domains to be cloned into impromptu clusters in a matter of sub-seconds.

An application that has been designed to work in the SnowFlock environment can expand its processing footprint in sub-second time, and then reduce it again when the computation is finished. We plan to enhance the Hadoop MapReduce implementation with the ability to efficiently expand the computing footprint to the number of desired processors in a dynamic manner. Required HDFS support will be explored as well. We plan to evaluate our implementation by comparing the performance of the extended Hadoop system to the original one. We will also evaluate the processing time of the proposed dynamic allocation of virtual machines.

We propose the following tentative list of milestones:

M1. Install Hadoop on a SnowFlock-based virtualized environment.
M2. Implement a Java wrapper for the SnowFlock Python or C APIs.
M3. Identify modification required to extend the Apache Hadoop code with the ability to dynamically allocate virtual machines.
M4. Implement the identified extensions.
M5. Identify the applications to be used for performance measurements. We aim at benchmarking common MapReduce use-cases, such as tuple selection in the map step and aggregation in the reduce step.
M6. Execute identified tests and analyze the results.

Refrences:

[1] DEAN, J., AND GHEMAWAT, S. MapReduce: Simplified data processing on large clusters. In OSDI (2004).
[2] Apache Hadoop MapReduce Project. http://hadoop.apache.org/mapreduce
[3] Apache Hadoon Cluster Setup. http://hadoop.apache.org/common/docs/current/cluster_setup.html
[4] Amazon Elastic Compute Cloud Developers Guide. http://docs.amazonwebservices.com/AWSEC2/latest/DeveloperGuide/
[5] LAGAR-CAVILLA, H. A., WHITNEY, J. A., SCANNELL, A., PATCHIN, P., RUMBLE, S. M., DE LARA, E., BRUDNO, M., AND SATYANARAYANAN, M. SnowFlock: Rapid Virtual Machine Cloning for Cloud Computing. In Eurosys 2009.