Acceleration of Big Data Applications with Cloud Computing Platform

Yan Luo* Yu Cao
Dept. of ECE Dept. of CS
University of Massachusetts Lowell
*Corresponding Author, Yan_Luo@uml.edu

Data analytics applications have been playing important roles in processing massive data in finance, biomedical, health care, social and political domains. Our research focuses on accelerating such data analytics applications with parallel computing platforms of clustered computers and GPUs. We have built a tool called Sparking for identification of task skew and speculative partition of data for spark applications. We are currently working on HeteroSpark, a heterogeneous computing framework for data intensive and deep learning applications. While being used in our research for scalability experiments, the current commercial cloud computing platforms such as Amazon EC2 have serious limitations. We expect the NSFCloud will be architected and implemented to address our research needs.

Ongoing Research Projects

1. Sparkling: Spark Application Analysis and Optimization

Apache Spark [1] has demonstrated its advantages over Hadoop's MapReduce in terms of both the runtime performance and the broader range of computation workloads that it can handle. Our contributions in the Sparkling project [2] include (a) we build a web tool called Sparkling with better execution visualization and statistical analysis to identify the task skew problem in a number of biomedical multimedia analytics applications; and (b) we propose two methods to address the unbalanced task execution problem: application-aware data partitioning and greedy (speculative) task scheduling. Our methods improve the execution time of biomedical multimedia analytics applications by up to 20%.

Sparkling enhances Spark application development in three ways: a) modify the original metrics system on Spark so that more useful analytical information can be obtained; b) visualize detailed activities on each task or node and demonstrate the overall performance with statistics; c) provide developers insight into how much better a program execution can achieve by avoiding "data skew" and "small tasks". We also propose two methods to mitigate data skews in Spark applications: (a) partition the input data into slices with decremented workload since the start time of the longest task often determines whether there is any idle time between stages (b) with the profiling information from input data profiler, we are able to partition the input based on domain knowledge such that "tough" data will be grouped into smaller workload slices.

2. HeteroSpark: CPU/GPU Heterogeneous Architecture for Big Data Processing

Heterogeneous systems combine both a CPU and GPU to gain advantages in performance and energy efficiency for suitable classes of applications, however, increase the complexity in programming such large-scale systems. To address these issues, we are working on HeteroSpark, a GPU-enabled heterogeneous cluster computing system integrated with Spark. In our design, we implement a wrapper library plugin on Spark slave node for transparent GPU invocation, so that user applications can benefit from GPUs without requiring explicit GPU programming. We

propose scheduling algorithm for schedule workloads dynamically to CPUs and GPUs based on the data dependencies and utilization of the processing elements. When a client submits a Spark job whose tasks can run on both CPU cores and GPU devices, the master job scheduler assigns the tasks onto CPU cores and GPU devices in order to minimize the overall job execution time by using profiling data collected from run-time tasks behavior. Worker nodes execute the scheduled map tasks on CPU cores or GPU devices within the framework designed for CPU-to-GPU task migration.

Problems with Current Experimentation Facilities

Most of our design and experiments were implemented on Amazon EC2 platform. Apart from the significant monetary cost of using it, however, we face severe problems with Amazon EC2, which stagnates the process of our research:

1. *Limited options in virtual machine provisioning*: VMs are allocated either within the same region or across regions by Amazon, however, it is difficult for a user to select the regions. Cross-region nodes cannot satisfy low network latency desired for certain applications.
2. *Resource sharing issue*: Since all allocated machines are virtual machines, users do not have the privilege to occupy the resource exclusively on a single machine. Thus, all resources are shared among different users. It is hard or nearly impossible for users to design their application based on accurate system profiling data.
3. *Reliability issues*: Since machines are on a sharing basis, it's often the case that nodes are down for unpredictable reasons. This introduces huge reliability issues when repeating scientific experiments.
4. *Limited GPU resources on a single machine*: The HeteroSpark project requires the testing of multiple GPUs on a single machine, and network-connected GPU nodes. However, current Amazon EC2 supports only up to one GPU per node.
5. *Slow system boot time*: Due to VM provision and initialization, the boot time of a large cluster on EC2 is extremely slow - about 20 to 30 minutes for a 64-node cluster. Some machines may fail to start.

Expectations on NSFCloud

1. Flexible usage models: While the facility is shared by design, we expect to be able to acquire a complete physical machine and boot it into images we plan to experiment with. This usage model is necessary for conduct profiling experiments for cloud workloads.
2. Reliable: We expect the platform to be highly reliable to ensure the execution of long-running tasks and repeatability.
3. Resource heterogeneity: GPUs are important resources we expect to access in addition to CPU based platforms. We expect to use nodes with multiple GPUs, and networked GPU nodes.
4. Free. We expect using NSFCloud is free of charge.

Reference

[1] Apache Spark. https://spark.apache.org/
[2] P. Li et al. Sparkling, Identification of Task Skew and Speculative Partition of Data for Spark Applications,
http://spark-summit.org/2014/talk/sparkling-identification-of-task-skew-and-speculative-partition-of-data-for-spark-applications