Koala: Improved Scheduler Running on Eucalyptus

Will Dietz, Kevin Larson
{wdietz2, klarson5}@illinois.edu
University of Illinois at Urbana Champaign

Abstract

As cloud computing continues to gain momentum, the infrastructure supporting clouds continues to grow. Many modern solutions make heavy use of virtualization, not only to provide isolation between tasks, but also for dividing hardware resources, and balancing machine utilization. Eucalyptus is an infrastructure that has contributed a great deal of work to this area. It is one of the most common open source cloud computing systems, and uses two simple scheduling algorithms, a greedy scheduling algorithm, and a round robin scheduling algorithm. We feel both of these algorithms leave room for improvement, and we expect to demonstrate that with our scheduler.

1 Related

The baseline of our work is the default set of schedulers contained in Eucalyptus [12]. Eucalyptus provides an EC2/S3 [1, 2] compatible cloud, using virtualization technologies such as Xen [5] and KVM/QEMU [7], Eucalyptus ships with two main schedulers: greedy and round-robin. Greedy will simply use the first available node that can run a given job, and round-robin each time will find the *next* suitable node. They also implement a power-save scheduler based on the greedy scheduler that attempts to shut down idle nodes to help conserve power. These are all basic scheduling algorithms that are simple and generic. However we believe that especially with respect to a given workload much more effective schedulers can be useful and intend to explore that problem space. Additionally Eucalyptus's sechduling framework does not allow the live migration of VM's once they're allocated, preventing a more complicated scheduling algorithms. We hope to address this as well in our work.

Researchers from Microsoft built Quincy [9], which is a cloud scheduling framework for Dryad and MapReduce-like systems. By reducing the scheduling problem to a customizable min-cost flow problem they expose a number of knobs that they found were rather effective in making an effective scheduler for various workloads. In our project, we will build a similarly parameterizable scheduler, but targeting Eucalyptus which has fundamentally different requirements. Additionally our work focuses on building a tool that helps you find the optimal parameters, not just the scheduling framework itself.

Zaharia et al. developed their LATE scheduler for MapReduce to improve task speculation on heterogenous systems [15]. This contrasts our work because we target not just the service level (MapReduce) but the infrastructure itself (at the VM level), where speculation is not traditionally used. Similarly Mantri et al. [4] focus on the outliers, the stragglers, in a MapReduce system to improve performance. Mantri et al. focus on exploring the cause of an outlier through monitoring and other methods to take appropriate action and free those resources. Early detection of outliers is key and is effective for MapReduce workloads.

A problem related to optimizing scheduling is that of load-balanacing. Zhao, et. al [17] developed an adaptive load-balancing system algorithm for Eucalyptus called “COMPARE_AND_BALANCE”. This system focuses on load-balancing as the primary goal, while we hope to build a more general scheduler to fit the needs and success metrics unique to each cluster, which is a fundementally different problem. Additionally they focused on a distributed algorithm forbidding the use of a centralized scheduling node, which is not a limitation we impose in our system.

Begnum et all [6] have experimented with moving virtual machine policy inside the virtual machine itself. The benefits of this include improved portability and aptitude for live-migration. Additionally, this increases the ease at which virtual machines can move between non-homogeneous systems within a cloud. They worked with MLN (Manage Large Networks), an open source tool providing atomic operations to enable the management of large quantities of a variety of virtual machines [5, 3, 14]. It provides an interface in the form of a configuration language.

A number of researchers have applied genetic algorithms (GA) to the task of dynamically scheduling on the cloud [11, 18, 16]. In [11] they improved upon Eucalyptus’s scheduling algorithms with a focus on topology sensitivity using GA. They claim in their future work they would be interested in machine learning based systems, similar to our system. Additionally they focused on the data-intensive workload produced from MapReduce tasks. Zhong, et al. [18] built their Improved Genetic Algorithm (IGA) to improve the scheduling algorithms commonly used in open-source clouds.

Some scheduling algorithms focus on particular workloads. While some researchers only consider MapReduce-like loads, others focus on scientific workloads [10, 13, 8]. By gathering basic characteristics of the workload, Simmhan et al. [13] build a *blackbox* system to help schedule the workflow. This is similar to our work, although we hope to determine the characteristics of the workflow experimentally, and not require them as inputs from the user.

2 Design

For our project, we will modify the scheduler for eucalyptus to have adjustable inputs. Options such as prioritizing of vms, number of vms per server, resources per vm, migration thresholds and many other options will be modify-able. We will then execute benchmarks and log the events related to the scheduler. We will then run a variety of benchmarks, likely including cpu-limited, memory-limited, disk-limited, and network limited benchmarks. We will then analyze these event logs by utilizing data mining techniques. We will use this analysis to generate the scheduler. We will compare the performance of this scheduler to the unmodified scheduler of Eucalyptus. Given sufficient time, we will generate a system that logs evens during execution, which will adjust the inputs on the fly, while attempting to maximize performance.

3 Proposed Timeline

- March 7
  - Get a basic install of Eucalyptus up and running.
- March 14
  - Have strong understanding of how scheduling works in Eucalyptus
  - Begin work on the design and modifications to the scheduler
- March 21
  - Have a working version of the adjustable scheduler which is ready to generate data.
- March 28

  - Characterize various workloads and design experiments to simulate them
  - Gather data with our scheduler on these workloads
- April 18
  - Analyze results of experiment
  - Investigate how we can improve it with respect to data
  - Implement various improvements and evaluate result
  - Iterate and document the process
- May 2
  - Final Evaluation: (comparing original scheduler to our own)
  - (Optional) Explore an automated version of this process and integrate with Eucalyptus.

References

[1] Amazon ec2. `http://aws.amazon.com/ec2/`.

[2] Amazon s3. `http://aws.amazon.com/s3/`.

[3] User-mode linux. `http://user-mode-linux.sourceforge.net/`.

[4] G. Ananthanarayanan, S. Kandula, A. Greenberg, I. Stoica, Y. Lu, B. Saha, and E. Harris. Reining in the outliers in map-reduce clusters using mantri. In *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, OSDI'10, pages 1–16, Berkeley, CA, USA, 2010. USENIX Association.

[5] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *Proceedings of the nineteenth ACM symposium on Operating systems principles*, SOSP '03, pages 164–177, New York, NY, USA, 2003. ACM.

[6] K. Begnum, N. A. Lartey, and L. Xing. Cloud-oriented virtual machine management with mln. In *Proceedings of the 1st International Conference on Cloud Computing*, CloudCom '09, pages 266–277, Berlin, Heidelberg, 2009. Springer-Verlag.

[7] F. Bellard. Qemu, a fast and portable dynamic translator. In *Proceedings of the annual conference on USENIX Annual Technical Conference*, ATEC '05, pages 41–41, Berkeley, CA, USA, 2005. USENIX Association.

[8] C. Hoffa, G. Mehta, T. Freeman, E. Deelman, K. Keahey, B. Berriman, and J. Good. On the use of cloud computing for scientific workflows. In *Proceedings of the 2008 Fourth IEEE International Conference on eScience*, pages 640–645, Washington, DC, USA, 2008. IEEE Computer Society.

[9] M. Isard, V. Prabhakaran, J. Currey, U. Wieder, K. Talwar, and A. Goldberg. Quincy: fair scheduling for distributed computing clusters. In *Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, SOSP '09, pages 261–276, New York, NY, USA, 2009. ACM.

[10] G. Juve and E. Deelman. Scientific workflows and clouds. *Crossroads*, 16:14–18, March 2010.

[11] G. Lee, N. Tolia, P. Ranganathan, and R. H. Katz. Topology-aware resource allocation for data-intensive workloads. In *Proceedings of the first ACM asia-pacific workshop on Workshop on systems*, APSys '10, pages 1–6, New York, NY, USA, 2010. ACM.

[12] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov. The eucalyptus open-source cloud-computing system. In *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid*, CCGRID '09, pages 124–131, Washington, DC, USA, 2009. IEEE Computer Society.

[13] Y. Simmhan and L. Ramakrishnan. Comparison of resource platform selection approaches for scientific workflows. In *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*, HPDC '10, pages 445–450, New York, NY, USA, 2010. ACM.

[14] B. Walters. Vmware virtual platform. *Linux J.*, 1999, July 1999.

[15] M. Zaharia, A. Konwinski, A. D. Joseph, R. Katz, and I. Stoica. Improving mapreduce performance in heterogeneous environments. In *Proceedings of the 8th USENIX conference on Operating systems design and implementation*, OSDI'08, pages 29–42, Berkeley, CA, USA, 2008. USENIX Association.

[16] C. Zhao, S. Zhang, Q. Liu, J. Xie, and J. Hu. Independent tasks scheduling based on genetic algorithm in cloud computing. In *Proceedings of the 5th International Conference on Wireless communications, networking and mobile computing*, WiCOM'09, pages 5548–5551, Piscataway, NJ, USA, 2009. IEEE Press.

[17] Y. Zhao and W. Huang. Adaptive distributed load balancing algorithm based on live migration of virtual machines in cloud. In *Proceedings of the 2009 Fifth International Joint Conference on INC, IMS and IDC*, NCM '09, pages 170–175, Washington, DC, USA, 2009. IEEE Computer Society.

[18] H. Zhong, K. Tao, and X. Zhang. An approach to optimized resource scheduling algorithm for open-source cloud systems. In *Proceedings of the The Fifth Annual ChinaGrid Conference*, CHINAGRID '10, pages 124–129, Washington, DC, USA, 2010. IEEE Computer Society.

4 Business Plan

Modern computing is moving towards larger and more complex tasks. These tasks require more storage and computational power than ever before. Cloud Computing is a developing field, built around providing solutions to this problem. Cloud Computing is closely tied with distributed and grid computing, and shares technology and infrastructure with them. Many advancements have been made in these areas, and Cloud Computing is currently a very popular venture for computing start-ups.

Eucalyptus is an infrastructure provided by Eucalyptus Systems Inc, which allows for deployment across a large variety of systems and provides a consistent interface for access to the underlying network. The current version of Eucalyptus has a primitive scheduler, utilizing greedy and round robin algorithms. In order to fully utilize a large cluster of machines, properly delegating tasks and balancing resource utilization is key.

We propose an enhanced system, built on top of Eucalyptus, which uses a learning scheduler to maximize throughput and utilization of the machines in a cluster. Our algorithm will gather scheduling events during operation and use this data to adjust how the scheduler behaves, based of various metrics such as disk and memory utilization, network bandwidth, and resource contention. Our software would allow for increased utilization of machines in a cluster, reducing the hardware, energy, and management expenses associated with the operation of a large cluster of machines.

The primary innovation behind the design of our scheduler is the utilization of machine learning techniques to leverage the event logs generated while executing workloads. For example, if a cluster is network limited, the scheduler can assign higher priority to data locality for jobs which have high network utilization. The scheduler will be designed to adjust allocation priorities based on a variety of factors, such as the properties of the workloads or the current utilization of the cluster.

In addition to the automated improvements to efficiency and throughput, the scheduler can be influenced externally to account for additional factors. It can be instructed to minimize network bandwidth (for other tasks) or reduce power consumption to reduce operation costs. For example, if resources are not scarce, or a workload is not time critical, the scheduler can be instructed to prioritize efficiency, and not use a machine if it cannot sufficiently utilize that machine's resources.