Aachen Institute for Advanced Study in Computational Engineering Science

Preprint: AICES-2009-5

20/February/2009

Can Cloud Computing Reach the TOP500?

P Bientinesi and J Napper

RWTH AACHEN UNIVERSITY

AICES

Financial support from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111 is gratefully acknowledged.

List of AICES technical reports: http://www.aices.rwth-aachen.de/preprints

Can Cloud Computing Reach The TOP500?

Jeffrey Napper*
Vrije Universiteit,
Amsterdam, The Netherlands

Paolo Bientinesi†
RWTH Aachen University, AICES,
Aachen, Germany

Abstract

Computing as a utility has reached the mainstream. Scientists can now rent time on large commercial clusters through several vendors. The cloud computing model provides flexible support for "pay as you go" systems. In addition to no upfront investment in large clusters or supercomputers, such systems incur no maintenance costs. Furthermore, they can be expanded and reduced on-demand in real-time.

Current cloud computing performance falls short of systems specifically designed for scientific applications. Scientific computing needs are quite different from those of web applications—composed primarily of database queries—that have been the focus of cloud computing vendors.

In this paper we investigate the use of cloud computing for high-performance numerical applications. In particular, we assume unlimited monetary resources to answer the question, "How high can a cloud computing service get in the TOP500 list?" We show results for the Linpack benchmark on different allocations on Amazon EC2.

*Supported by the XtreemOS project, which is partially funded by the European Commission under contract #FP6-033576.

†Financial support from the Deutsche Forschungsgemeinschaft (German Research Association) through grant GSC 111 is gratefully acknowledged.

1 Introduction

Computing as a utility has reached the mainstream: vendors now rent all or portions of physical machines for hourly periods for web services [11, 8, 5]. The *cloud computing* model emphasizes the ability to scale compute resources on demand. The advantages for users are numerous. Unlike conventional cluster systems, there is no upfront investment in infrastructure or people and ongoing expenses are simplified. Total cost can be close to zero when resources are not in use. The cloud user can pay costs directly proportional to need rather than allocating resources according to average or peak load.

This paper explores whether cloud computing services are suitable for high-performance computing (HPC) workloads. In contrast, web service workloads that often have little intra-cluster communication are the primary users of current cloud computing services. However, cloud nodes are typically configured to run user-provided software so that cloud computing nodes can just as easily run scientific applications. The ability to quickly create and scale-up a custom compute cluster is a boon to individual scientists whose computing needs can be sporadic. Cloud computing services can also be used to extend existing clusters for larger problem sizes [6]. Although cloud providers currently place small bounds on dynamically allocated resources, trends point toward increasing bounds on these resources over time.

To run scientific applications efficiently, cloud computing needs to provide resources comparable to current HPC systems. These systems are ranked by the TOP500 list [13] that lists the fastest supercomputers worldwide. Expectations are low that cloud systems built for web service workloads (that is, without extremely fast interconnects) can compare favorably with purpose-built HPC supercomputers even though cloud systems may ultimately provide more resources. To determine the feasibility of this new platform for high-performance numerical applications, we benchmark clusters of up to 128 compute cores using Amazon EC2 web services [11].

Specifically, we use the High-Performance Linpack (HPL) implementation [10] of the LINPACK benchmark to evaluate cloud performance on dense linear algebra workloads. The LINPACK benchmark is based on the solution of a random dense system of linear equations. The solution is computed by an LU decomposition with partial pivoting followed by backsubstitution. In early versions of the benchmark, the size of the system was fixed, initially to 100×100, and subsequently to 1000×1000. Such a constraint was subsequently released to generate more accurate estimates of the peak performance

for a target computer. We chose the LINPACK benchmark for several reasons: 1) dense linear algebra is prevalent in scientific applications, 2) the performance of the benchmark scales linearly with the size of the cluster [2], 3) due to compute-intensive algorithms (n^3 computations over n^2 data), LINPACK provides a good upper bound on the expected performance of scientific workloads, and 4) LINPACK is similarly used by the TOP500 [13] list.

Can the sheer size of a cloud computing cluster help attain a rank on the TOP500 list at any price? Our results show that the performance of single nodes available on EC2 is as good as nodes found in current HPC systems [13]. On the other hand, the available memory and network performance are insufficient to maintain high performance when scaling up the cluster. It appears that regardless of how many nodes are used, a cluster built out of current EC2 nodes cannot attain a spot on the TOP500 list. While the high-performance interconnects used in supercomputing systems will protect their standing in the TOP500 list, the number of nodes available in a cloud system may allow those without access to the biggest supercomputers to solve larger problem sizes. However, these solutions come at the significantly reduced efficiency (as measured in GFLOP/sec) available in cloud systems.

In addition to standard metrics such as GFLOP/sec and efficiency used in HPC, we introduce GFLOP/$ (billions of floating point operations per dollar) and $/size (dollars to solve a linear system of given input matrix size) to analyze in depth the pros and cons of clouds. Cloud computing makes the total cost of computation explicit—there is no need to add maintenance and administration costs. The GFLOP/$ metric allows users similarly to estimate straightforward costs for different applications with respect to computational efficiency while the $/size metric estimates costs with respect to problem size.

2 Experiment

We perform our experiments on the Amazon Elastic Compute Cloud (EC2) service [11]. Although there are competing cloud offerings that were publicly available at the time [8, 5], Amazon's service provides complete control over a node so that all processors that share memory in the same system are allocated to the user. Some other services provide processors with sizeable memory allocations, but do not guarantee that another processor with access to the same memory bus is not allocated to a different user.

Nodes allocated through EC2 are called *instances.* Instances are allo-

cated from Amazon's data centers according to unpublished scheduling algorithms.[1] Data centers are combined into entities known as an *availability zone*, Amazon's smallest logical geographic entity for allocation. These zones are further combined into regions, which consist of only the US and Europe at the moment.

After allocation, each instance automatically loads a user-specified *image* containing the proper operating system (in our case Linux) and user software (described below). Images are loaded automatically by Amazon services onto one or more virtualized processors using the Xen virtual machine (VM) [1]. Each processor is itself multi-core, resulting in a total of 4 to 8 virtual cores for the instances we reserved.

Tools written to Amazon's public APIs provide the abilities to allocate extra nodes on demand, release unused nodes, and create and destroy images to be loaded onto allocated instances. Using these tools and developing our own, we built images with the latest compilers provided by the hardware CPU vendors AMD and Intel. We use HPL 2.0 [10] from the University of Tennessee, compiled with GotoBLAS 1.26 [4] from the Texas Advanced Computing Center (TACC), and MPICH2 1.0.8 [7] from the Argonne National Laboratory. Using our tools we can allocate and configure variable size clusters in EC2 automatically, including support for MPI applications.

Although we developed tools to automatically manage and configure EC2 nodes for our applications, there are also other publicly available tools for running scientific applications on cloud platforms (including EC2) [12, 9]. Further, as the cloud computing platform matures, we expect much more development for specific applications such as high-performance computing to reduce or eliminate much of the initial learning curve for deploying scientific applications on cloud platforms. Already, for example, public images are available on EC2 supporting MPICH [3].

Finally, Edward Walker has previously compared EC2 nodes to current HPC systems [14]. Our results are similar to his for the small clusters of 4 nodes that he used.

[1] Allocations are initially limited to 20 total instances, but this restriction can be lifted upon request.

2.1 Setup

Our evaluations were carried out using *extra-large* nodes of both the *standard* and *high-CPU* categories of the Amazon EC2 cloud in January 2009. Both node types cost $0.80 per hour to allocate.[2] The extra-large node of the high-CPU class consists of 2 Intel Xeon quad-core processors operating at a frequency of 2.3 GHz with a total memory of 7 GB. Each core is capable of executing 4 floating-point operations per clock cycle, leading to a theoretical peak performance of 74.56 GFLOP/sec per node. The biggest nodes of the standard class contain two AMD Opteron dual-core processors at 2.60 GHz with 15 GB of local memory. Each core performs 2 floating-point operation per second, resulting in a theoretical peak performance of 20.8 GFLOP/sec per node.

In regards to multithreaded parallelism provided by the multi-core processors, extensive testing delivered the best performance when we set the BLAS to use as many threads as available cores per processor—4 and 2, for the Xeon and the Opteron, respectively. With these settings and using the platform-specific libraries and compilers, we reached 76% and 68% of theoretical peak performance (as measured in GFLOP/sec) for the Xeon and Opteron, respectively, for single node performance. We thus believe the configuration and execution of LINPACK in HPL on the high-CPU and standard instances is efficient enough to use as an exemplar of compute-intensive applications for the purposes of our evaluation.

The specific interconnect used by Amazon is unspecified [14] and multiple instances might even share a single hardware network card. Therefore, the entire throughput might not be available to any particular instance. In order to reduce the number of hops between nodes, we run all experiments with cluster nodes allocated in the same availability zone.

2.2 Results

Figure 1 provides the percentage of theoretical peak GFLOP/sec for the given cluster size achieved using both the AMD and Intel CPUs. The problem sizes were increased as the number of nodes allocated to the cluster increased. Note that the nodes are configured without swap space. The problem size was chosen to keep the memory allocation per processor constant—for every doubling of cluster size, the problem size is scaled by a factor of $\sqrt{2}$. We

[2]There are additional costs for bandwidth used into and out from Amazon's network.

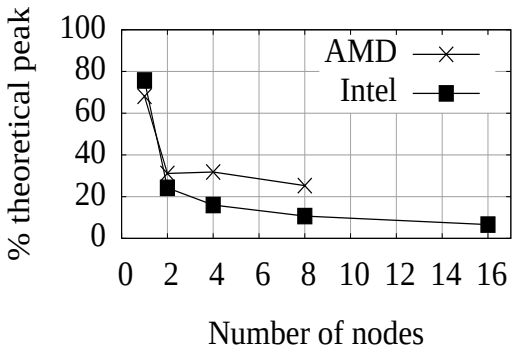
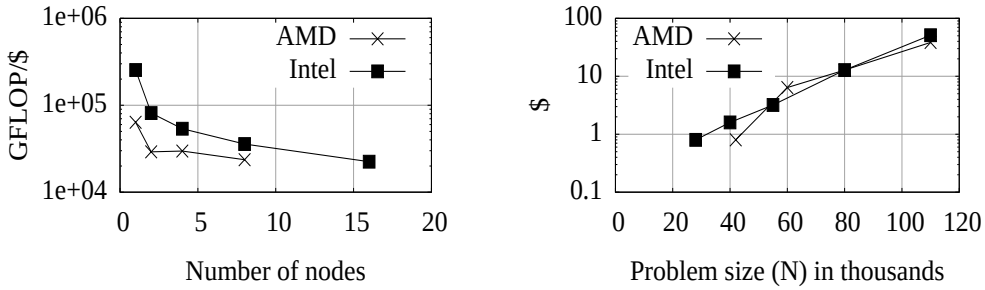


Figure 1: Efficiency as percentage of optimal GFLOP/sec on clusters of AMD or Intel EC2 instances.

did not allocate clusters of 16 nodes for the AMD instances because their larger physical memories allowed them to solve problem sizes with 8 nodes (120 GB RAM) that required 16 Intel nodes (112 GB RAM). Furthermore, the performance trends are already clear.

The data sets for both processor types display a severe loss in performance stepping up from one to two nodes although performance of LINPACK should scale linearly [2]. We ran many different configurations to find the best HPL settings (number of rows and columns, partitioning block size, panel factorization algorithm, broadcast algorithm, etc.) and report the peak for each cluster size individually. Settings are typically different across cluster sizes. Without disk activity, the exponential degradation in performance for two EC2 nodes is easily attributable to the slow interconnect. The Xeon cluster obtains the same order of magnitude of GFLOP/sec for different clusters sizes while the Opteron instances appear to increase GFLOP/sec as the cluster size increases. However, for the Xeon, the problem of a slow network is aggravated by the limited memory available: less than 1GB/core (7GB every two quad-cores). In fact, the Xeon does not surpass single-node performance (that is, 75 GFLOP/sec) in our experiments until 16 nodes are used (80 GFLOP/sec). Given that the AMD instance scales significantly better using (presumably) the same interconnect, we conclude that the small memory provision represents a limiting factor for the Xeon—especially in the presence of a slower interconnect.

To evaluate cloud computing as an alternative on-demand cluster, we present results with respect to costs. One advantage of cloud computing is the "pay as you go" aspect wherein instances can be created and deleted



(a) By cluster size.

(b) By problem size.

Figure 2: Performance for money spent on EC2 using AMD and Intel instances by cluster and problem sizes. The y-axis is log scale.

on demand. Figure 2a demonstrates the GFLOP/sec per dollar (GFLOP/$) obtained for different cluster sizes and instance types. The values are measured as the average floating point operations obtained over large problem sizes divided by the average compute time used on the cluster per dollar (note: instances are charged by the hour—we did not quantize the costs for this figure). Figure 2a clearly shows the exponential decrease in performance with respect to dollar cost of the clusters. Hence, as we double the cluster size, we receive less GFLOP/sec in return for money spent. We adjusted for cluster size so that a perfectly scaled cluster should appear as a straight line. Although spending extra money on EC2 might still help reach the solution faster, the reverse can also be true—for a fixed sum, smaller clusters can attain higher performance.

In our experiments GFLOP/sec increase only marginally with cluster size. Instead of improved performance, larger clusters can also be used to solve larger problems without disk I/O. Figure 2b provides the actual costs for solving linear systems of different sizes. The data represents the actual costs for the cluster of compute nodes to complete one specific run of HPL for the peak GFLOP/sec configuration. The costs of larger problem sizes also increases exponentially, as we expect from Figure 2a. Users must balance the problem size solveable by different size clusters with the significant cost pressure to keep clusters small for efficiency of GFLOP/$. Our experiments show that although the AMD instances can solve larger problems sizes due to the better provisioned memory, the costs are equivalent to Intel instances that use more nodes to solve the same problem size faster. Of course, this

tradeoff could change as the cluster sizes increase further.

3 Conclusions

While cloud computing provides an extensible and powerful computing environment for web services, our experiments indicate that the cloud (or Amazon's EC2, at least) is not yet mature enough for HPC computations. We observe that the GFLOP/sec obtained per dollar spent decrease exponentially with increasing computing cores and correspondingly, the cost for solving a linear system increases exponentially with the problem size—very much in constrast to existing scalable HPC systems.

We do see a future for cloud systems in HPC. The effort to create customized images will shrink over time as tools improve. At the moment, clouds can already be used to run private clusters or extend current HPC systems. However, we have shown that these clusters are very inefficent compared to current HPC systems. If cloud computing vendors are serious about targeting the HPC market, different cost models must be explored. An obvious first step would be to offer better interconnects or nodes provisioned with more physical memory to overcome the slower network.

References

[1] Paul T. Barham, Boris Dragovic, Keir Fraser, Steven Hand, Timothy L. Harris, Alex Ho, Rolf Neugebauer, Ian Pratt, and Andrew Warfield. Xen And The Art Of Virtualization. In *Symposium on Operating Systems Principles*, pages 164-177, Bolton Landing, New York, October 2003.

[2] Jack Dongarra, Robert van de Geijn, and David Walker. Scalability Issues Affecting The Design Of A Dense Linear Algebra Library. In *Journal of Parallel and Distributed Computing*, 22(3):523–537, September 1994.

[3] Chris Gemignani and Peter Skomoroch. Elasticwulf: Beowulf Cluster Run On Amazon ec2. Available on the WWW, December 2008. http://code.google.com/p/elasticwulf/.

[4] Kazushige Goto. Gotoblas. Available on the WWW, May 2008. http://www.tacc.utexas.edu/resources/software/#blas.

[5] ServePath Dedicated Hosting. gogrid Cloud Hosting. Available on the WWW, 2009. http://gogrid.com.

[6] K. Keahey, T. Freeman, J. Lauret, and D. Olson. Virtual Workspaces For Scientific Applications. In *SciDAC 2007 Conference*, June 2007.

[7] Argonne National Laboratory. Mpich2: High-performance And Widely Portable mpi. Available on the WWW, October 2008. http://www.mcs.anl.gov/research/projects/mpich2/.

[8] xcalibre communications ltd. flexiscale Cloud Computing. Available on the WWW, 2009. http://www.flexiscale.com.

[9] Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. The Eucalyptus Open-source Cloud-computing System. In *Proceedings of Cloud Computing and Its Applications [online]*, October 2008.

[10] A. Petitet, R. C. Whaley, J. Dongarra, and A. Cleary. hpl - A Portable Implementation Of The High-performance Linpack Benchmark For Distributed-memory Computers. Available on the WWW, September 2008. http://www.netlib.org/benchmark/hpl/.

[11] Amazon Web Services. Amazon Elastic Compute Cloud (ec2). Available on the WWW, 2009. http://aws.amazon.com/ec2.

[12] Amazon Web Services. Nimbus Science Clouds. Available on the WWW, 2009. http://workspace.globus.org/.

[13] TOP500.Org. Top 500 Supercomputer Sites. Available on the WWW, November 2008. http://www.top500.org/.

[14] Edward Walker. Benchmarking Amazon ec2. In *;LOGIN:* page 18–23, October 2008.