Flexible Cloud Computing through Swift Coasters

[Extended Abstract]

Ketan Maheshwari
MCS, Argonne National Lab
ketan@mcs.anl.gov

Justin M. Wozniak
MCS, Argonne National Lab
wozniak@mcs.anl.gov

Mihael Hategan
MCS, Argonne National Lab
hategan@mcs.anl.gov

Allan Espinosa
CS, University of Chicago
aespinosa@cs.uchicago.edu

Daniel Katz
CI, University of Chicago
dsk@ci.uchicago.edu

Michael Wilde
MCS, Argonne National Lab
wilde@mcs.anl.gov

ABSTRACT

In this short paper we describe the benefits—rapid prototyping and flexible computing—of using the Swift parallel scripting tool and its Coasters mechanism on a collection of diverse large scale and high-performance computing infrastructures. Coasters facilitates a pilot-job-based multi-level scheduling mechanism for provisioning and aggregating resources. We illustrate the benefits in a scientific experiment of the CyberShake application.

Keywords

clouds, scripting, coasters

1. INTRODUCTION

Rapid prototyping plays a crucial role for computationally intensive scientific applications. Furthermore, porting such applications on diverse computing infrastructures is not trivial. In today's collaborative scientific endeavors, diverse infrastructures are available to a typical scientific application user. Leveraging from such infrastructure is a significant challenge, however, since each has a different model of computation. Availability of a unifying and flexible enabler for scientific applications is crucial in such a scenario.

In this short paper, we describe and illustrate our efforts in achieving a flexible, multi-infrastructure enabling scientific applications through the development of tools capable of rapid prototyping and porting. We collectively call this multi-infrastructural setup a "metacloud".

2. SWIFT AND COASTERS

Swift [5] is a scripting language designed for composing application programs into parallel applications that can be executed on multicore processors, clusters, grids, clouds, and supercomputers. Scripting through Swift enables rapid prototyping of parallel applications through highly portable and compact representation. Coasters [3] in Swift facilitates use of a mechanism that enables job submission to a metacloud based on persistent, reusable placeholder for jobs, similar to pilot-based jobs or Condor's glide-ins [1]. These placeholders can be reused by a job submission system, enabling a prepared collection of resources for the bulk of computation. The power of Coasters lies in the fact that it offers the different modes of computing required for diverse applications and infrastructures. For instance, Coasters enables many-task distributed computing on local clusters and clouds through ssh and tcp-based tunneling, and on Grids and Supercomputing infrastructures, Coasters can talk to the batched job schedulers. Coasters operates on a *service-worker* mechanism where services receive jobs from Swift-scripted applications and pass the jobs in the form of work to remotely deployed workers. Coaster services also handle data staging based on the file-system configuration (shared or distributed). Especially in a dynamic metacloud Coasters is capable of the following:

- Dynamically growing and shrinking the resource pool, based on user policy and demand from the workflow.
- Keeping a workflow running even as some resources fail.
- Handling data transfer over the same connections as job management, and avoiding the use of and need for a shared filesystem (which is troublesome in a cloud environment).
- Managing multiple Clouds as separate pools, and integrating Cloud, Grid and Supercomputer resources in the same run.

Applications with many small tasks can readily take advantage of these mechanisms and avoid idling on a waiting mode in congested queues.

3. THE CYBERSHAKE APPLICATION

The CyberShake computation platform implemented at the Southern California Earthquake Center is a simulation facility with the objective of predicting earthquakes in the area by using stochastic methods such as probabilistic seismic hazard analysis [2]. For a single geographic site, it takes 17,000 CPU-hours to generate its hazard curve. Data

produced throughout this computation totals 750 GB. The tasks involved to complete the curve involves tightly coupled and embarrassingly parallel jobs. The current production system computes curves within 24 to 48 hours on a single NSF TeraGrid resource. Runs were made on 800 CPU reservations. The speedup time obtained is *O(1000)*.

4. INFRASTRUCTURES

Diverse infrastructures exist in today's scientific domain capable of seamless operation for applications. Each class of infrastructure, however, has its own model of computation owing to the suitability to its hardware architecture. In what follows, we give an overview of key computing infrastructures and their features crucial for porting scientific applications on them.

Clusters. Closely connected compute and data nodes on a shared filesystem and simple tcp-based connection facilities. Fast but limited in size and computational power. Examples include NFS-enabled LANs and dedicated clusters such as PADS (*http://pads.ci.uchicago.edu*).

Clouds and Grids. Loosely connected collection of clusters offering a middleware, batched job submission-based mechanism wherein the jobs are submitted to a compute node via a (meta)schedular. Schedular queues often get congested and cause serious application performance bottlenecks. Examples include OSG [4], TeraGrid and the Bionimbus (*http://www.bionimbus.org*) cloud.

Supercomputers. Production-grade supercomputing infrastructures offering enormous computing power with high-performance computing capabilities. Beagle (*https://beagle.ci.uchicago.edu*) is one example of such an infrastructure. Beagle is a 151-teraflops, Cray XE6 system supporting computation, simulation, and data analysis for biomedical research.

Together, these resources offer tremendous power and capabilities for scientific experiments, some of which were inconcievable until recently. Harnessing such power efficiently, however, requires highly sophisticated tools and techniques.

5. SETUP AND RESULTS

Shown in Fig. 1 is one such experiment setup where Coasters enables a "reservation" on a resource-competent Open Science Grid infrastructure. Upto 2500+ workers were acquired with this mechanism for the CyberShake experiment (see Fig. 2).

6. CONCLUSIONS

Operating through myriad diverse computing resources can be a daunting task for a scientific user. Tools such as scripting and Coasters not only facilitate rapid prototyping, but also effectively harness such resources with relative ease and flexibility.

Acknowledgments

This work was supported by the U.S. Department of Energy under Contract DE-AC02-06CH11357.

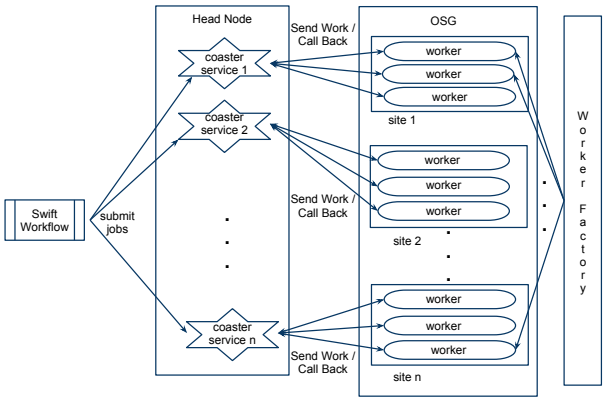

Figure 1: Swift Coasters setup for the CyberShake application experiment on the Open Science Grid

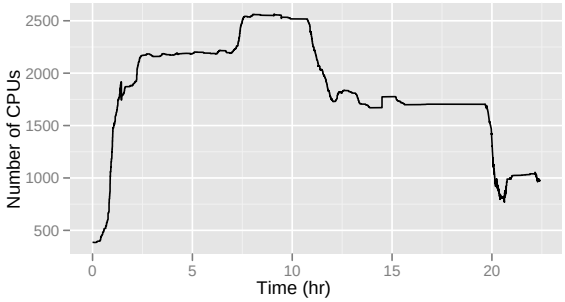

Figure 2: Peak "reservation" of as many as 2500 workers, achieved through Coasters

7. REFERENCES

[1] Frey J. et al. Condor-G: A computation management agent for multi-institutional grids. *Cluster Computing*, 5(3), 2002.

[2] R. Graves, T. Jordan, S. Callaghan, E. Deelman, E. Field, G. Juve, C. Kesselman, P. Maechling, G. Mehta, K. Milner, D. Okaya, P. Small, and K. Vahi. Cybershake: A physics-based seismic hazard model for southern california. *Pure and Applied Geophysics*, 168:367–381, 2011. 10.1007/s00024-010-0161-6.

[3] M. Hategan. urlhttp://wiki.cogkit.org/wiki/Coasters.

[4] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, F. Wurthwein, I. Foster, R. Gardner, M. Wilde, A. Blatecky, J. McGee, and R. Quick. The open science grid. *Journal of Physics: Conference Series*, 78(1):012057, 2007.

[5] M. Wilde, I. Foster, K. Iskra, P. Beckman, Z. Zhang, A. Espinosa, M. Hategan, B. Clifford, and I. Raicu. Parallel scripting for applications at the petascale and beyond. *Computer*, 42(11):50–60, 2009.