Exploring the Performance and Mapping of HPC Applications to Platforms in the Cloud

Abhishek Gupta,
Laxmikant V. Kalé
University of Illinois at
Urbana-Champaign, Urbana,
IL, USA
(gupta59,kale)@illinois.edu

Filippo Gioachin,
Verdi March,
Chun Hui Suen,
Bu-Sung Lee
HP Labs, Singapore
(gioachin, verdi.march,
chun-hui.suen,
francis.lee)@hp.com

Paolo Faraboschi,
Richard Kaufmann,
Dejan Milojicic
HP Labs, Palo Alto, CA, USA
(paolo.faraboschi,
richard.kaufmann,
dejan.milojicic)@hp.com

ABSTRACT

This paper presents a scheme to optimize the mapping of HPC applications to a set of *hybrid* dedicated and cloud resources. First, we characterize application performance on dedicated clusters and cloud to obtain application signatures. Then, we propose an algorithm to match these signatures to resources such that performance is maximized and cost is minimized. Finally, we show simulation results revealing that in a concrete scenario our proposed scheme reduces the cost by 60% at only 10-15% performance penalty vs. a non optimized configuration. We also find that the execution overhead in cloud can be minimized to a negligible level using thin hypervisors or OS-level containers.

Categories and Subject Descriptors

D.1.3 [**Concurrent Programming**]: Parallel Programming;
K.6.4 [**System Management**]: Centralization/decentralization

Keywords

High Performance Computing, Clouds, Resource Scheduling

1. INTRODUCTION

A recent study reaffirmed that dedicated supercomputers are still more cost-effective than cloud for large-scale HPC applications [2]. This is largely due to the high overhead of virtualization on I/O latency which hinders the adoption of cloud for large-scale HPC applications [2, 6]. However, our preliminary study indicated that cloud resources could be cost-effective for small and medium-scale HPC applications [5]. As such, resource allocation should be aware of application and resource characteristics to maximize application performance yet minimizing cost.

This paper describes a proposed scheme to intelligently map an HPC application to a set of *hybrid* resources consisting of a mix of dedicated and cloud resources. Section 2 begins with our in-depth performance characterization for HPC applications on various dedicated clusters and cloud. We discover that the cloud overhead can be minimized to a negligible level using thin hypervisors or OS-level containers. Then, we propose an algorithm that leverages the performance characteristics to map an application to resources. In Section 3 we present our simulation results showing that our scheme reduces the cost by 60% compared to a non-optimized configuration, while the performance penalty is kept at 10-15%. Finally, Section 4 discusses the lessons learned and potential implications of our study.

Table 1: Experimental Test-bed

Platform	Ranger	Taub	Open Cirrus	Eucacloud	HPLS
Cores/ node	16 @2.3GHz	12 @2.67GHz	4 @3.00GHz	2 @2.67GHz	12 @2.67GHz
Network	Infiniband	Infiniband	10 GigaE	1 GigaE	1 GigaE

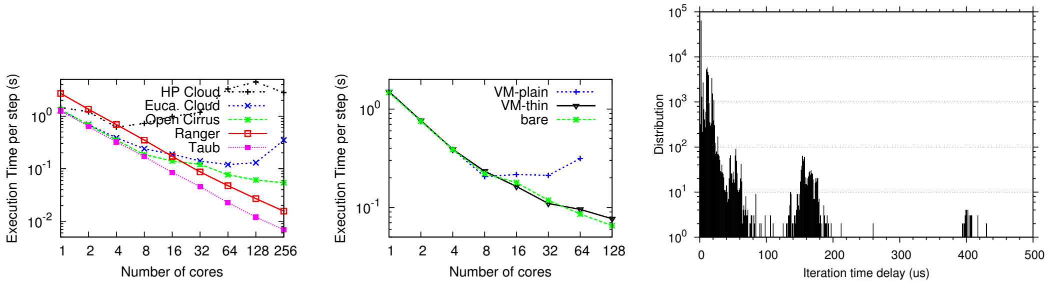
2. APPROACH

We benchmarked a variety of platforms spanning different architecture (see Table 1). Ranger and Taub are supercomputers, while Open Cirrus is a dedicated cluster with slower interconnect. HPLS and Eucalyptus are typical cloud environment. We also compare lightweight virtualization using dedicated network (thin VM) and Linux containers (LXC) using NAMD [4], a highly scalable molecular dynamics application with the ApoA1 input (92k atoms).

Figure 1 shows the scaling behavior of our testbeds for (a) different platforms and (b) for different virtualization techniques applied to a typical cloud node. Due to superior network performance on the supercomputers (Taub, Ranger), NAMD scales well over the test range, while we observe scaling problems on Open cirrus and even more on cloud (Eucalyptus, HPLS) due to inferior network performance, which we verified by measuring the time spent in communication. Networking on cloud is further impacted by the I/O virtualization overhead, although through a more in-depth study we show alternative techniques (b) that can partially mitigate the overhead. thin VM assigns a dedicated network interface to each VM via an IOMMU pass-through, and achieves near native performance ('bare'). We also show that the slowdown incurred by CPU virtualization is minimal, compared to conventional network virtualization ('plain VM'). Interference from the OS and hypervisor causes additional slowdown on VMs. Figure 1(c) shows the distribution of execution slowdown from the ideal 1000μs execution step measured on a virtualized node.

Based on these findings and our previous work [5], we developed a mapper tool shown in Figure 2. Starting from an HPC application, through characterization we extract a

HPDC'12, June 18–22, 2012, Delft, The Netherlands.
ACM 978-1-4503-0805-2/12/06.



(a) NAMD on different platforms (b) NAMD using different VM (c) Noise

Figure 1: (a,b) Execution time vs. Number of cores for NAMD (c) Noise Benchmark on a VM

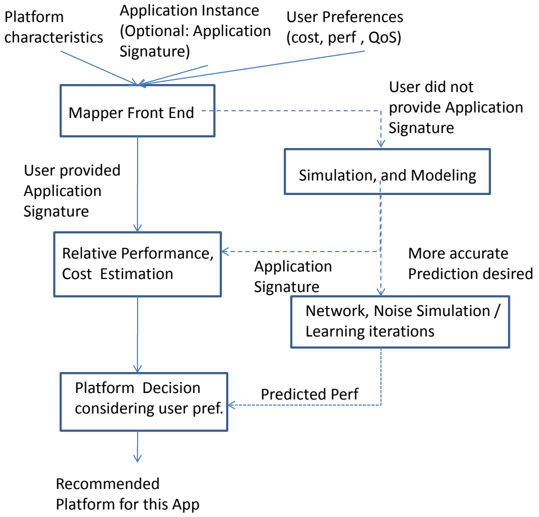

Platform characteristics
Application Instance (Optional: Application Signature)
User Preferences (cost, perf , QoS)
Mapper Front End
User did not provide Application Signature
Simulation, and Modeling
User provided Application Signature
Relative Performance, Cost Estimation
Application Signature
More accurate Prediction desired
Network, Noise Simulation / Learning iterations
Platform Decision considering user pref.
Predicted Perf
Recommended Platform for this App

Figure 2: Mapper Approach

signature capturing the most important dimensions: number and size of messages, computational grain size (FLOPS), overlap percentage of computation and communication, presence of synchronization barriers and load balancing. Subsequently, given a set of applications to execute and a set of target platforms, we define heuristics to map the applications to the platforms that optimize *parallel efficiency*. In doing so, we consider several target platforms spanning a variety of processor configurations, interconnection networks, and virtualization environments. Platform characteristics, such as CPU frequency, interconnect latency and bandwidth, platform costs (using a pay-per-use charging rate based model) and user preferences are considered. The output of the tool are platform recommendations to optimize practical scenarios such as best performance within a constrained budget, or cost minimization with performance guarantees.

3. RESULTS

We evaluated the results obtained by our mapper and studied the benefits using it to map a set of application to supercomputer (Ranger) and Eucalyptus cloud. Figure 3 shows the significant cost savings achieved while meeting performance guarantees using our intelligent mapper. Embarrassingly parallel (EP) and Integer sort (IS) benchmarks are part of NPB Class B benchmark suite [1] and Jacobi2D is a kernel which performs 5-point stencil computation to average values in a 2-D grid. The application suffix is the number of processors; for Jacobi, we consider multiple problem sizes, that is input matrix dimensions (e.g. size 1k × 1k). For this application set, our scheme reduces the cost on average by 60% compared to a non-optimized configuration, while the performance penalty is kept at 10-15%. Further details can be found in the technical report on this work [3].

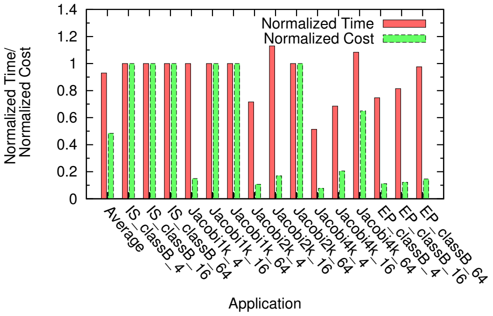


Figure 3: Normalized Performance and Cost (intelligent mapping vs execution on supercomputer)

4. LESSONS LEARNED, CONCLUSIONS

We have shown that the adoption of intelligent mapping techniques is pivotal to the success of hybrid platform environments that combine supercomputer and typical hypervisor-based clouds. In some cases, a hybrid cloud-supercomputer platform environment can outperform its individual constituents. We learned that application characterization in HPC-Cloud space is a challenging problem, but the benefits are substantial. Finally, we demonstrated that lightweight virtualization is important to remove “friction” from HPC in cloud.

We described the concept and initial implementation of a static tool to automate the mapping, using a combination of application characteristics, platform parameters, and user preferences. In the future, we plan to extend the mapping tool to also perform a dynamic adjustment of the static mapping through run-time monitoring.

5. REFERENCES

[1] NPB. `http://www.nas.nasa.gov/Resources/Software/npb.html`.

[2] Magellan Final Report. Technical report, U.S. Department of Energy (DOE), 2011.

[3] Exploring the Performance and Mapping of HPC Applications to Platforms in the Cloud. Technical report, HP Labs, 2012.

[4] A. Bhatele et al. Overcoming Scaling Challenges in Biomolecular Simulations across Multiple Platforms. In *IPDPS 2008*.

[5] A. Gupta and D. Milojicic. Evaluation of HPC Applications on Cloud. In *Best Student Paper, Open Cirrus Summit*, 2011.

[6] E. Walker. Benchmarking Amazon EC2 for high-performance scientific computing. *LOGIN*, 2008.