Towards Efficient Mapping, Scheduling, and Execution of HPC Applications on Platforms in Cloud

Abhishek Gupta (4th year Ph.D. student) and Laxmikant V Kalé
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA
(gupta59, kale)@illinois.edu

***Abstract*—The advantages of pay-as-you-go model, elasticity, and the flexibility and customization offered by virtualization make cloud computing an attractive option for meeting the needs of some High Performance Computing (HPC) users. However, there is a mismatch between cloud environments and HPC requirements. The poor interconnect and I/O performance in cloud, HPC-agnostic cloud schedulers, and the inherent heterogeneity and multi-tenancy in cloud are some bottlenecks for effective HPC in cloud.**

Our primary thesis is that cloud is suitable for *some* HPC applications *not all* applications, and for those applications, cloud can be more cost-effective compared to typical dedicated HPC platforms using intelligent application-to-platform *mapping*, HPC-aware cloud *schedulers*, and cloud-aware HPC *execution* and parallel runtime system. To address the challenges, and to exploit the opportunities offered by HPC-clouds, we make OpenStack Nova scheduler HPC-aware and Charm++ parallel runtime system cloud-aware. We demonstrate that our techniques result in significant improvement in cost (up to 60%), performance (up to 45%), and throughput (up to 32%) for HPC in cloud; helping cloud users gain confidence in the capabilities of cloud for HPC, and cloud providers run a more profitable business.

***Keywords*-Cloud; High Performance Computing; Scheduling; Placement; Runtime system; Virtual machines**

I. Introduction and Related Work

Cloud computing has recently emerged as a cost effective alternative to dedicated infrastructure for HPC applications. Running an application in cloud avoids the long lead time, high capital expenditure, and large operational costs associated with a dedicated HPC infrastructure [1]. In addition, the ability to provision HPC resources on-demand with high *elasticity*, reduces the risks caused by under-provisioning, and reduces the underutilization of resources caused by over-provisioning. Finally, the built-in virtualization support in the cloud offers an alternative way to support flexibility, customization, security, and resource control to the HPC community.

However, despite these benefits, there is a mismatch between the requirements of HPC and the characteristics of current cloud environments [1–4]. Most HPC applications consist of tightly-coupled parallel processes which perform frequent communication and synchronizations. Dominant challenges for HPC in cloud are shown in Figure 1, and include the following: the absence of low-latency and high-bandwidth interconnect in clouds, network and I/O virtualization overhead, hardware heterogeneity, cross-application interference arising from multi-tenancy, and the HPC-agnostic cloud schedulers [1–4].

While the outcome of these studies paints a rather pessimistic view of HPC clouds, recently there have been efforts towards HPC-optimized clouds (such as Amazon Cluster Compute [5] and DoE Magellan project [1,3,6]), HPC-aware cloud schedulers [7, 8] and topology-aware mapping of application virtual machines (VMs) to physical topology [9]. These efforts point to a promising direction to overcome some of the fundamental inhibitors. However, much work remains to be done, and today only embarrassingly parallel or small scale HPC applications can be efficiently run in cloud [1–4].

In this thesis, outlined in Figure 1, we take a more holistic approach unlike past research: First, besides addressing the challenges of running HPC applications in cloud, we also explore the opportunities offered by cloud for HPC. Secondly, our research is aimed at improving HPC performance, resource utilization, and cost when running in cloud and hence it is beneficial to both – users and cloud providers. Finally, with the objective of providing a set of techniques to bridge the gap between HPC and clouds, we adopt a threefold complementary approach:

- *Mapping* applications to platforms in cloud intelligently: Through comprehensive performance evaluation and analysis, we identify what application and platform characteristics are crucial for the selection of a platform for a particular application. We conclude that a hybrid supercomputer-cloud approach can be more cost-effective compared to running all applications on a dedicated supercomputer or all in cloud [4, 10]. (§II)
- Making cloud *schedulers* and VM placement HPC-aware: We propose and demonstrate techniques for application-aware consolidation and placement of VMs on physical machines. Through topology-awareness, heterogeneity-awareness, cross-VM interference accounting, and careful co-location of application VMs of complementary execution profiles, we achieve significant improvement in performance and resource utilization [11, 12]. (§III)
- Making HPC *execution* and runtime cloud-aware: We address the challenges of heterogeneity and multi-tenancy

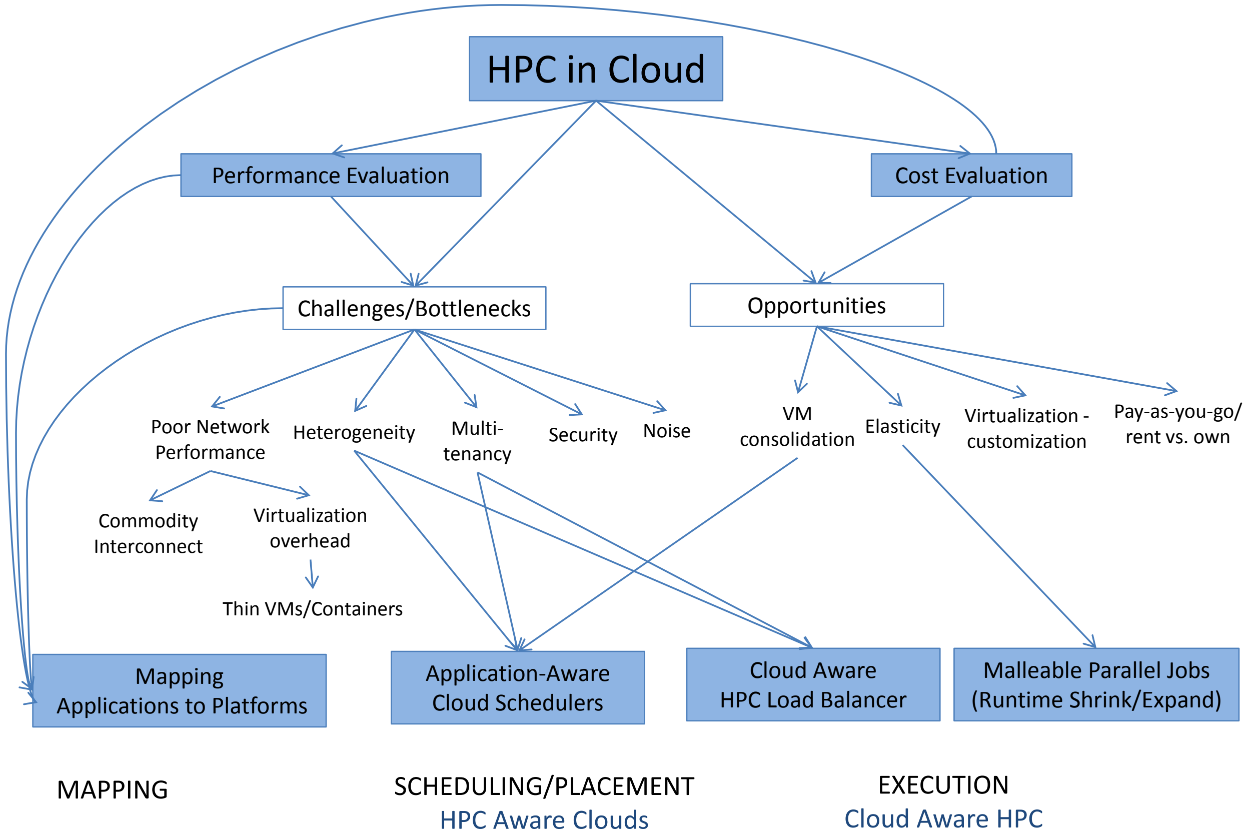


Fig. 1: Thesis overview

in cloud through dynamic redistribution of parallel tasks (Charm++ [13, 14] objects or AMPI [14] threads) to VMs [15, 16]. We also explore the use of malleable jobs to benefit from the inherent *elasticity* in cloud. (§IV)

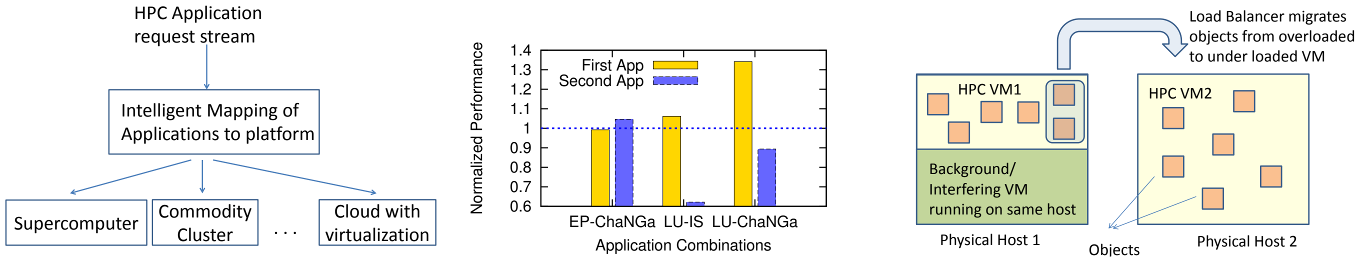
List of relevant publications

- A. Gupta and D. Milojicic, "Evaluation of HPC Applications on Cloud," in *Open Cirrus Summit Best Student Paper*, Atlanta, GA, Oct. 2011, pp. 22 –26
- A. Gupta et al., "Exploring the Performance and Mapping of HPC Applications to Platforms in the cloud," in *HPDC '12*. New York, NY, USA: ACM, 2012, pp. 121–122
- A. Gupta, D. Milojicic, and L. Kale, "Optimizing VM Placement for HPC in Cloud," in *Workshop on Cloud Services, Federation and the 8th Open Cirrus Summit*, San Jose, CA, 2012
- A. Gupta, L. Kale, D. Milojicic, P. Faraboschi, and S. Balle, "HPC-Aware VM Placement in Infrastructure Clouds ," in *accepted at IEEE Intl. Conf. on Cloud Engineering IC2E '13*
- O. Sarood, A. Gupta, and L. V. Kale, "Cloud Friendly Load Balancing for HPC Applications: Preliminary Work," in *Parallel Processing Workshops (ICPPW), 2012 41st Intl. Conf. on*, sept. 2012, pp. 200 –205
- A. Gupta, O. Sarood, L. Kale, and D. Milojicic, "Improving HPC Application Performance in Cloud through Dynamic Load Balancing," in *accepted at IEEE/ACM CCGRID '13*
- A. Gupta et al., "The Who, What, Why and How of High Performance Computing Applications in the Cloud," HP Labs, Tech. Rep., 2013
- A. Gupta and L. V. Kale, "Towards Efficient Mapping, Scheduling, and Execution of HPC Applications on Platforms in Cloud," in *accepted at 27th International Parallel and Distributed Processing Symposium(IPDPS) Ph.D Forum'13*

The **contributions** of this thesis are summarized as follows. These have been taken verbatim from above publications.

Evaluation and Mapping

- We analyze the performance of HPC applications on a range of platforms varying from supercomputer to cloud, study performance bottlenecks and identify *what* applications are suitable for cloud.
- We analyze the impact of virtualization on HPC applications and propose techniques, specifically thin hypervisors, OS-level containers, and hypervisor and application-level CPU affinity, to mitigate performance overhead and noise, addressing – *how* to use cloud for HPC.
- We investigate the economic aspects of running in cloud vs. supercomputer and discuss *why* it is challenging to make a profitable business for cloud providers for HPC compared to traditional cloud applications. We also show that small/medium-scale HPC users are the most likely



(a) Mapping Concept

(b) Co-locating VMs from different applications on a physical node: application-aware scheduling

(c) Load balancer to mitigate the effect of static and dynamic heterogeneity on application performance

Fig. 2: Methodology

candidates *who* can benefit from an HPC-cloud.

- We demonstrate that rather than running all the applications on a single platform (in-house or cloud), a more cost-effective approach is to leverage multiple platforms (dedicated and in the cloud) and use a smart application-aware mapping of applications to platforms, addressing – *how* to use cloud for HPC.

HPC-Aware Clouds

- We identify the opportunities and challenges of VM consolidation for HPC in cloud. In addition, we develop scheduling algorithms which optimize resource allocation while being HPC-aware. We achieve this by applying Multi-dimensional Online Bin Packing (MDOBP) heuristics while ensuring that cross-application interference is kept within bounds.
- We optimize the performance for HPC in cloud through intelligent HPC-aware VM placement – specifically topology awareness and homogeneity, showing performance gains up to 25% compared to HPC-agnostic scheduling.
- We implement the proposed algorithm in OpenStack Nova scheduler to enable intelligent application-aware VM scheduling. Through experimental measurements, we show that compared to dedicated execution, our techniques can result in up to 45% better performance while limiting jitter to 8%.
- We modify CloudSim [19] to make it suitable for simulation of HPC in cloud. To our knowledge, our work is the first effort towards simulation of HPC job scheduling algorithms in cloud. Simulation results show that our techniques can result in up to 32% increased throughput compared to default scheduling algorithms.

Cloud-Aware HPC

- We propose dynamic load balancing for efficient execution of tightly-coupled iterative HPC applications in heterogeneous and dynamic cloud environment. The main idea is periodic refinement of task distribution using measured CPU loads, task loads and idle times.
- We implement these techniques in Charm++ and evaluate their performance and scalability on a real cloud setup on Open Cirrus testbed [20]. We achieve 45% reduction in execution time compared to no load balancing.
- We analyze the impact of load balancing frequency, grain size, and problem size on achieved performance.

II. Performance and Mapping of HPC in Cloud

The primary research challenge that we address here is that rather than running all the applications on a single platform (in-house or cloud), will it be more cost-effective approach to leverage multiple platforms (dedicated and in the cloud) and if so, how? To answer this question, we evaluated the performance and cost of running a set of HPC benchmarks (NPB benchmarks [21]) and some real world applications, such as NAMD [22], ChaNGa [23], and Sweep3D [24], on a range of platforms – supercomputer, HPC-optimized cluster, private cloud, and public cloud. These platforms have different interconnects, operating systems, and virtualization. Our results, presented in [4], show that cloud can be cost-effective compared to supercomputers at small scale or for applications which are less communication-intensive.

Based on this observation, we proposed a tool for mapping application to platforms in cloud using application characteristics such as communication intensiveness and sensitivity to noise. Instead of considering cloud as a substitute for supercomputer, we investigated the co-existence of supercomputer and cloud (See Figure 2a). We follow a two step methodology – 1) Characterize an application using theoretical models, previous instrumentation, or simulation to generate an application signature that captures application's communication profile, grain size, and problem size and 2) use heuristics to select a suitable platform from a given set for an application based on application signature, platform characteristics, and user preferences. In [10], we provided a proof-of-concept of this approach, and evaluated the associated benefits of a smart mapping tool. Through simulation using simple regular applications, we showed that in a concrete scenario with a supercomputer (Ranger [25]) and a Eucalyptus based cloud as two available platforms, our scheme reduces the cost by 60% while limiting the performance penalty to 10-15% vs. a non optimized configuration.

Characterizing an HPC application and predicting its performance is challenging and has been extensively researched. Run-time instrumentation, event-tracing and curve-

TABLE I: Summary of planned work

Project	Area	Duration	Description
Mapping strategies	Mapping	2 months	Design and evaluation using simulation (CloudSim) of strategies for mapping application from an incoming stream to a set of platforms
Mix-match HPC,non-HPC	HPC-Aware Cloud	2 months	Evaluation of co-execution of web and HPC workloads ??
Malleable Jobs	Cloud-Aware HPC	4 months	Design and implementation of parallel runtime support for malleable jobs in Charm++, Evaluation using job simulation
Load Balancer	Cloud-Aware HPC	1 months	Extend load balancer to meta-balancer instead of periodic load balancing
Performance Simulation/Emulation	Cloud-Aware HPC	2 months	Larger scale results using simulation or emulation of cloud environment
Thesis Writing	Dissertation	2 months	Completing the dissertation document for final defense

fitting based performance-modeling approaches have been explored [26–28]. Our objective in this thesis is not to perform extensive applications characterization but to discover the most important dimensions for the purpose of mapping applications to platforms. Through our research, we have demonstrated that there are significant benefits which can be achieved by using an intelligent tool and a combination of multiple platform, compared to a single platform or naive mapping. We believe that our approach can be extended to complex applications such as those with irregular communication patterns and multiple phases.

III. HPC-Aware Cloud Scheduler

The second method which we adopt to bridge the gap between HPC and cloud is to focus on cloud schedulers and explore opportunities to a) improve HPC performance in cloud and b) reduce HPC cost when running in cloud. Current strategies for placement of VMs to physical machines are mostly HPC-agnostics, that is they do not consider the intrinsic nature of HPC applications. An HPC application consists of n processes which communicate and synchronize frequently with each other during the execution. However, in cloud physical machines can be heterogeneous, and achieved network performance between two physical nodes (also referred to as *hosts*) can vary significantly depending on the physical position of nodes in the network topology. Hence, to obtain better performance, we modified OpenStack [29] Nova scheduler to make at HPC-aware. OpenStack is a popular cloud management system. We evaluated the modified OpenStack Nova scheduler by setting up a cloud on Open Cirrus test-bed [20] using KVM [30] as hypervisor. In [11], we demonstrated performance improvement up to 20% through topology- and hardware-awareness.

We extended this HPC-aware scheduler to accomplish the second goal – to make HPC execution more economical in cloud. To this end, we explored the opportunities and challenges of VM consolidation for HPC. Figure 2b illustrates this with a simple experiment, where we use two multi-core physical nodes (4-core, 8GB, 3 GHz each) of Open Cirrus testbed at HP Labs site. We use VMs with 1-vcpu, 2GB memory, and KVM as hypervisor. The applications used in this experiment are NPB [21] (EP = Embarrisingly Parallel, LU = LU factorization, IS = Integer Sort) problem size class B and ChaNGa [23] = Cosmology. We first ran each application using all 4 cores of a node (dedicated mode), and then ran them in shared mode, where each node is shared by the two applications – 2 VMs of each application run on a node, 4 VMs total per application. Figure 2b shows the performance for both applications in shared execution normalized with respect to the dedicated execution for different application combinations. Here, the x-label represent the application combination and the first (similarly second) bar corresponds to the first (second) application in x-label. It is clear from Figure 2b that some application combinations achieve normalized performance close to one (EP-ChaNGa), some co-locations results in significant detrimental impact on performance of one application (e.g. ChaNGa-IS because IS is communication-intensive, hence locating all 4 VMs on same node reduces communication time), whereas in case of LU-ChaNGa, the interference actually results in performance improvement. Investigation revealed that this is due to the large working set size of LU and small working set size of EP, which means that the shared last level cache is better utilized when the applications are run in the shared mode [12].

We demonstrated that there are significant benefits of using a common pool of resources for applications with different characteristics (such as HPC vs. non-HPC, communication, synchronization, cache intensiveness) but cross-application interference is a major impediment to effective resource based packing of HPC applications. To address this problem, we adopt the following approach - 1) Characterize application along two dimensions – tightly coupledness and use of shared resources (such as cache) on a shared physical node and 2) match applications whose execution profiles well complement each other and place them on same node to improve resource utilization. We implemented this approach on top of existing OpenStack Nova scheduler and evaluated it in the same setup and above. Our results in [12] show that our techniques achieve 45% better performance while limiting jitter to 8% through cross-VM interference accounting. We also modified a popular cloud simulator – CloudSim [19] to make it HPC-aware. Simulation results using CloudSim showed that our application-aware consolidation technique can result in 32% increase in throughput compared to default scheduling techniques.

IV. Cloud-Aware HPC Runtime

The final approach that we follow is to adapt HPC runtime to meet the needs of cloud environments. Our hypothesis is that the parallel runtime system should be able to adapt to the the dynamic variations in the cloud execution environment, resulting in improved performance. In addition, by providing runtime support for dynamically expanding/shrinking parallel jobs, significant gains in terms of higher resource utilization and cost savings can be achieved by leveraging cloud features such as variable pricing.

To validate our hypothesis, we investigate the adaptation of Charm++ [13, 14] parallel runtime system to virtualized environment. HPC applications and runtime are typically designed to be run in a homogeneous and dedicated environment whereas in case of cloud, there is inherent static hardware heterogeneity, and multi-tenancy can result in dynamic heterogeneity (e.g. other application VMs entering and leaving a physical node). Heterogeneity – both static and dynamic, significantly degrades performance of parallel applications especially those which are iterative and bulk synchronous. To minimize the impact of these factors on application performance, we designed and implemented a cloud-aware load balancer for HPC applications on top of existing Charm++ load balancing framework. Our approach is based on decomposing the workload into medium grained tasks called objects, which can be easily migratable by the runtime across processors (virtual cores in our case). The load balancing framework instruments the application execution, and measures object and processor loads. Idle times on VMs are also measured. It is assumed that there is very small variation in object loads across iterations – sometimes referred to as *principle of persistence*. Hence, based on the measured statistics from previous iterations, we migrate loads away from overloaded VMs to underloaded VMs. Figure 2c illustrates a situation with two physical hosts (nodes). There is a VM from another application running on the first host. Without load balancing, each host would be distributed equal number of objects. Since, one of the HPC VM has to time-share the CPU with the interfering VM, the load is imbalanced, and whole application will slow down. Our load balancer detects this condition, and migrates objects from overloaded to underloaded VMs based on average load calculation. Details of the algorithm can be found in [16]. We evaluated our techniques on a real cloud setup up to 64 VMs. Our results shown in [15,16] demonstrate performance benefits up to 45% for scientific benchmarks and a real world molecular dynamics application.

In future, we plan to evaluate this load balancer on a larger scale, and to explore runtime support for malleable jobs. Adaptive MPI (AMPI) [14] can be used to obtain these benefits of our dynamic runtime system for MPI [31] applications.

V. Conclusions

Since clouds have traditionally been designed for business and web applications with the goal of increasing the utilization of underutilized resources through consolidation and multi-tenancy, there is a mismatch between current cloud offerings and HPC requirements. This thesis aims to bridge that gap through effective mapping, VM placement and scheduling, and execution of HPC applications on a range of platforms in cloud. Using a complementary approach of making clouds HPC-aware and HPC cloud-aware, we have demonstrated that HPC performance-cost tradeoffs in cloud can be significantly improved.

References

[1] "Magellan Final Report," U.S. Department of Energy (DOE), Tech. Rep., 2011, http://science.energy.gov/~/media/ascr/pdf/program-documents/docs/Magellan_Final_Report.pdf.

[2] A. Iosup et al., "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 22, no. 6, pp. 931 –945, june 2011.

[3] K. R. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. J. Wright, "Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud," in *CloudCom'10*, 2010.

[4] A. Gupta and D. Milojicic, "Evaluation of HPC Applications on Cloud," in *Open Cirrus Summit Best Student Paper*, Atlanta, GA, Oct. 2011, pp. 22 –26.

[5] "High Performance Computing (HPC) on AWS," http://aws.amazon.com/hpc-applications.

[6] "Magellan - Argonne's DoE Cloud Computing," http://magellan.alcf.anl.gov.

[7] "Nova Scheduling Adaptations," http://xlcloud.org/bin/download/Download/Presentations/Workshop_26072012_Scheduler.pdf.

[8] "HeterogeneousArchitectureScheduler," http://wiki.openstack.org/HeterogeneousArchitectureScheduler.

[9] P. Fan, Z. Chen, J. Wang, Z. Zheng, and M. R. Lyu, "Topology-Aware Deployment of Scientific Applications in Cloud Computing," *Cloud Computing, IEEE International Conference on*, vol. 0, 2012.

[10] A. Gupta et al., "Exploring the Performance and Mapping of HPC Applications to Platforms in the cloud," in *HPDC '12*. New York, NY, USA: ACM, 2012, pp. 121–122.

[11] A. Gupta, D. Milojicic, and L. Kale, "Optimizing VM Placement for HPC in Cloud," in *Workshop on Cloud Services, Federation and the 8th Open Cirrus Summit*, San Jose, CA, 2012.

[12] A. Gupta, L. Kale, D. Milojicic, P. Faraboschi, and S. Balle, "HPC-Aware VM Placement in Infrastructure Clouds ," in *accepted at IEEE Intl. Conf. on Cloud Engineering IC2E '13*.

[13] L. Kale and S. Krishnan, "Charm++: A Portable Concurrent Object Oriented System Based on C++," in *OOPSLA*, September 1993.

[14] L. V. Kale and G. Zheng, "Charm++ and AMPI: Adaptive Runtime Strategies via Migratable Objects," in *Advanced Computational Infrastructures for Parallel and Distributed Applications*, M. Parashar, Ed. Wiley-Interscience, 2009, pp. 265–282.

[15] O. Sarood, A. Gupta, and L. V. Kale, "Cloud Friendly Load Balancing for HPC Applications: Preliminary Work," in *Parallel Processing Workshops (ICPPW), 2012 41st Intl. Conf. on*, sept. 2012, pp. 200 –205.

[16] A. Gupta, O. Sarood, L. Kale, and D. Milojicic, "Improving HPC Application Performance in Cloud through Dynamic Load Balancing," in *accepted at IEEE/ACM CCGRID '13*.

[17] A. Gupta et al., "The Who, What, Why and How of High Performance Computing Applications in the Cloud," HP Labs, Tech. Rep., 2013.

[18] A. Gupta and L. V. Kale, "Towards Efficient Mapping, Scheduling, and Execution of HPC Applications on Platforms in Cloud," in *accepted at 27th International Parallel and Distributed Processing Symposium(IPDPS) Ph.D Forum'13*.

[19] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A Toolkit for Modeling and Simulation of Cloud Computing Environments and Evaluation of Resource Provisioning algorithms," *Softw. Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Jan. 2011.

[20] A. I. Avetisyan and et al., "Open Cirrus: A Global Cloud Computing Testbed," *Computer*, vol. 43, pp. 35–43, April 2010.

[21] "NAS Parallel Benchmarks," http://www.nas.nasa.gov/Resources/Software/npb.html.

[22] A. Bhatele, S. Kumar, C. Mei, J. C. Phillips, G. Zheng, and L. V. Kale, "Overcoming Scaling Challenges in Biomolecular Simulations across Multiple Platforms," in *IPDPS 2008*, April 2008, pp. 1–12.

[23] P. Jetley, F. Gioachin, C. Mendes, L. V. Kale, and T. R. Quinn, "Massively Parallel Cosmological Simulations with ChaNGa," in *IDPPS*, 2008, pp. 1–12.

[24] "The ASCII Sweep3D code," http://wwwc3.lanl.gov/pal/software/sweep3d.

[25] "Ranger User Guide," http://services.tacc.utexas.edu/index.php/ranger-user-guide.

[26] C. da Lu and D. Reed, "Compact Application Signatures for Parallel and Distributed Scientific Codes," in *Supercomputing, ACM/IEEE 2002*.

[27] J. S. Vetter, N. Bhatia, E. M. Grobelny, P. C. Roth, and G. R. Joubert, "Capturing Petascale Application Characteristics with the Sequoia Toolkit," in *In Proceedings of Parallel Computing 2005. Malaga*, 2005.

[28] D. H. Bailey and A. Snavely, "Performance Modeling: Understanding the Past and Predicting the Future," in *in Euro-Par 2005*, p. 185.

[29] "Open Stack Open Source Cloud Computing Software," http://www.openstack.org/.

[30] "kvm – Kernel-based Virtual Machine," Redhat, Inc., Tech. Rep., 2009.

[31] "MPI: A Message Passing Interface Standard," in *M. P. I. Forum*, 1994.