ASETS: A SDN Empowered Task Scheduling System for HPCaaS on the Cloud

Saba Jamalian, Hassan Rajaei
Department of Computer Science
Bowling Green State University
Bowling Green, OH
{sabaj, rajaei}@bgsu.edu

***Abstract*—With increasing demands for High Performance Computing (HPC), new ideas and methods are emerged to utilize computing resources more efficiently. Cloud Computing appears to provide benefits such as resource pooling, broad network access and cost efficiency for the HPC applications. However, moving the HPC applications to the cloud can face several key challenges, primarily, the virtualization overhead, multi-tenancy and network latency. Software-Defined Networking (SDN) as an emerging technology appears to pave the road and provide dynamic manipulation of cloud networking such as topology, routing, and bandwidth allocation. This paper presents a new scheme called ASETS which targets dynamic configuration and monitoring of cloud networking using SDN to improve the performance of HPC applications and in particular task scheduling for HPC as a service on the cloud (HPCaaS). Further, SETSA, (SDN-Empowered Task Scheduler Algorithm) is proposed as a novel task scheduling algorithm for the offered ASETS architecture. SETSA monitors the network bandwidth to take advantage of its changes when submitting tasks to the virtual machines. Empirical analysis of the algorithm in different case scenarios show that SETSA has significant potentials to improve the performance of HPCaaS platforms by increasing the bandwidth efficiency and decreasing task turnaround time. In addition, SETSAW, (SETSA Window) is proposed as an improvement of the SETSA algorithm.**

Keywords—component; Cloud Computing, HPC, HPCaaS, SDN, Task Scheduling, Virtual Machines, Cloud Networking

I. INTRODUCTION

Advances in Cloud Computing (CC) attracted scientist and industries to deploy their applications on the cloud. High Performance Computing (HPC) programs also have targeted the Cloud as a base infrastructure. Nevertheless, HPC tends to have a unique set of requirements that might not fit directly into standard clouds. While the heart of the cloud heavily utilizes the high performance computing concept, direct provisioning of the HPC applications to the cloud can encounter numerous performance degradation issues. However, essential characteristics [1] of cloud computing persuade HPC users to consider using the cloud infrastructure for their application. To move these applications to the cloud, the infrastructure needs to be fine-tuned for the HPC tasks in order to reduce for example the overhead of virtual machines, provide improved network bandwidth, solve the performance shortcomings caused by multi-tenancy, and offer scalable number of virtual machines to test e.g. the scalability of an HPC algorithm. Moving High Performance Computing to the cloud has been introduced as HPC as a Service (HPCaaS) [2]. Cloud enables the HPC users to have access to supercomputing resources on demand and in a cost efficient manner. The providers of HPCaaS, often own the service platform, administrate, and maintain the virtual resources. They can either own the hardware or rent it from a cloud service provider. As a result, the platform can have updated resources with the latest technology while the users can benefit the service and need not worry about the underlying mechanisms.

Software-Defined Networking (SDN) [3] has established its name as a new trend in networking. The basic concept of SDN is to decouple the control plane of the network from the data plane with an Application Programming Interface (API) [4]. This technique enables the network administrators and the application developers to dynamically manage and alter network parameters in runtime and according to the current demand [5]. Unique characteristics of SDN have made it an appropriate choice for virtualized network and in particular suited for cloud networking [6].

An important capability of SDN is the fact that it enables applications to be aware of the network bandwidth for all of the links [6]. In this paper, we use this feature of SDN to design and implement an innovative task scheduling system called ASETS; A SDN Empowered Task Scheduling System for HPC as a Service. In addition, we propose a novel algorithm for this system called SETSA (SDN Empowered Task Scheduling Algorithm). This algorithm takes advantage of the SDN capabilities and schedules the tasks to the available Virtual Machines (VM) such that the virtualized network bandwidth is maximized on an elastic HPCaaS architecture with a shared file system and on-demand number of workers. To improve the performance of SETSA, we also introduce a second algorithm called SETSAW which utilizes a time window to parallelize assigning tasks to the VMs. Finally, several case studies of the algorithm using empirical modeling are presented to show the potential of the algorithm to improve performance of HPCaaS in terms of bandwidth efficiency and turnaround time.

The rest of this paper is as follows. Section II introduces the related works. Section III describes advances in HPCaaS and its architecture. Section IV describes SDN and its application in Cloud Networking. Section V states the problem whereas Section VI describes ASETS in details, the architecture, and its two algorithms SETSA & SETSAW. Section VII provides empirical case studies to evaluate performance. Discussions, future works, and concluding remarks are presented in Sections VIII, IX, and X respectively.

II. Related Work

This section briefly describes the efforts made by other researchers to improve the performance of HPCaaS systems using various methods, and the attempts made to use SDN in the HPCaaS platforms.

AbdelBaky et al in [2] introduce a prototype to transform a supercomputer into a cloud that supports accessibility of HPC resources through IaaS, PaaS and SaaS abstractions. It is discussed in the paper that efforts to provide HPC resources for scientific applications in forms of "HPC in the cloud", "HPC plus clouds", and "HPC as a cloud" have not reached to a complete success due to limited capabilities of the underlying commodity hardware, lack of high-speed interconnects, the physical distance between servers, and virtualization overhead. The study identifies ease of use, elasticity and accessibility of the cloud as the primary benefits of HPCaaS.

Taifi et al. [7] proposed architecture to build HPC clusters on top of a private cloud. High throughput connections using Infiniband switches are used between the compute VMs whereas an Ethernet is connecting the compute nodes to the Cloud controller. The high bandwidth interconnects play an important role in improving the performance of the network which is very critical for HPC applications. This study identifies three major benefits for HPCaaS: flexibility of the resources, resource efficiency, and cost reduction. To achieve these benefits, three important challenges need to be addressed: virtualization overhead, administrative complexity, and programming model. To address the administrative complexity of HPCaaS, they designed and implemented a virtual cluster administration tool called HPCFY. To evaluate the private HPC cloud both data-intensive and compute-intensive benchmarks were used in the experiments. Their results indicate that the primary challenges such as gaining performance and reliability will also grow with the scale of the HPC-cloud. Despite these weaknesses, the authors predict bright future for cloud-based HPC.

Task scheduling problem in a cloud is NP-Complete since both cloud resources and user application requirements can change rapidly. The Cloud Resource Broker (CLOUDRB) [8] is a framework to schedule HPC applications on the cloud for scientific purposes. This framework uses a Particle Swarm Optimization (PSO)-based Resource Allocation method to address the job scheduling problem. The purpose is to minimize job finishing time, cost and job rejection ratio. Job rejection ratio is the rate of jobs that cannot be accepted by the cloud due to the unavailability of resources. On the other hand, the framework aims to maximize number of jobs finished within a deadline. The performance is evaluated by Matlab simulation as well as implementation on a private cloud with real-world HPC applications. The results showed clear performance improvement as the framework meets the objectives.

The increasing demand for scientific computing in Biology and others has resulted in several approaches such as Galaxy [9] to address cloud-based services for computation intensive algorithms in this field. Galaxy is a scientific workflow engine for computational biology with a web-based interface that makes it easy for biologists to run their applications on the cloud. Researchers at University of Chicago have used Galaxy to deploy bio-informatics workflow across local and Amazon EC2 cloud [10]. A recent study [11] developed a proof of the concept for deploying HPC applications as a service on the cloud. The platform enables scientists and in particular Biology and Medicine specialists who have no computer science background to run their scientific HPC case studies easily and with promising performance. Kawai in [12] states that SDN has great potentials to help the HPC systems since these applications need very resource-intensive computation.

In addition, High Throughput Computing (HTC) is also capable of utilizing SDN services to improve performance of the data-intensive computing. Examples include Big Data applications and Hadoop which need careful considerations of network configuration. Qin et al [13] proposed a Bandwidth–Aware scheduling algorithm with SDN for Hadoop named BASS. Hadoop scheduler assigns tasks based on data locality. However, BASS takes the instant bandwidth of the links into account when assigning tasks. This capability enables the scheduler to move data from one node to another when necessary for better scheduling.

III. High Performance Computing as a Service

Desirable characteristics of the Cloud such as on-demand access, resource pooling and cost efficiency [14], have tempted industries and academia to embrace this technology into their businesses, including the HPC users. Nevertheless, standard clouds do not satisfy certain unique requirements of HPC such as batch processing, direct access to hardware, bypassing OS kernel, and high performance execution [15].

To receive the benefits of cloud for heavy scientific computing, the needs for "HPC as a service" on the cloud has recently become a sharp demand. A study [16] identifies the lower cost of Cloud as a primary motivation for scientists to move their HPC applications to the cloud. Dynamic allocation of resources based on application demand plays the most important role for cost efficiency. Accordingly, the systems which need HPCaaS tend to be elastic in term of resource provisioning. In addition, such a service eliminates troubleshooting, maintenance, and administration of local HPC resources which is often a challenge for scientists who normally do not want to get involved with such complex tasks. Nevertheless, existing implementation of HPCaaS have numerous shortcomings which need to resolve. Studies such as [16] [17] [18] identify "Virtualization Overhead" and "Multi-tenancy" of the cloud as primary source of challenges for HPCaaS. Multi-tenancy of the network can result in unstable cloud network bandwidth and high communication latency. Such turbulence will have crucial negative effect on the performance of HPC applications [19].

Earlier attempts such as [16] and [20] to implement HPCaaS suffer from the use of identical methods for the architecture, technology, and scheduling on the cloud the same as the traditional supercomputers. However, newer efforts such as [2] and [21] utilize cloud-specific characteristics (e.g. elasticity) and cloud based services (e.g. queuing service, database and storage). Results are much more promising for the architectures that utilize cloud-specific resources. Job and task schedulers for the HPCaaS systems differ from the

traditional ones on a private data center. Scientific and HPC applications often need to execute a workflow of jobs and tasks. In a typical workflow, the output of a task becomes the input for another task. HPCaaS schedulers need to take care of such workflows. Furthermore, traditional HPC applications needed to specify number of processors needed to execute the jobs. On the other hand, recently with moldable job scheduling [22] in HPCaaS, this requirement may no longer be needed.

IV. Software-Defined Networking

Software-Defined Networking (SDN) is a new paradigm in networking in which the network control and physical forwarding are separated [3]. In SDN, the network is directly programmable, agile, and manageable by a controller software which is open standard and vendor neutral network solution.

SDN architecture transforms the scattered network configuration into two parts: a simple light weight fast forwarding hardware and the decision-making network controller(s) [23]. By decoupling the control plane from the data plane in a software-defined network, the intelligence of the network is centralized in an SDN controller. This enables network administrators and application developers to monitor and modify network properties during execution time [4].

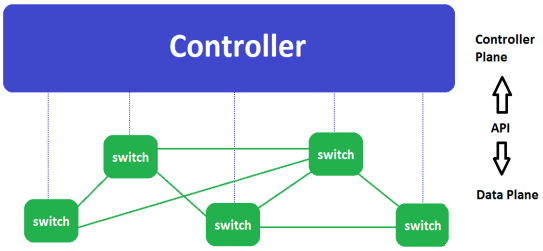

Figure 1: SDN Logical Architecture

Figure 1 shows the logical architecture of a software-defined network. Network controller has the full control over the network. Data plane is only used for data forwarding. The data plane is fully managed by the controller. Communication between the controller and data plane (switches) is through RESTful APIs and SDN protocol. As of today, the most popular protocol used for SDN is OpenFlow [24].

Software-Defined Networking has great potentials to play an important role in cloud networking. In a virtualized network, each tenant is able to have its own SDN enabled controller rather than having a separate physical switch [5]. In this paper, the capabilities of network monitoring in SDN are used to improve the performance of an HPCaaS task scheduler. We present a new HPCaaS architecture (ASETS) with a task scheduling algorithm (SETSA) that benefits from SDN capabilities to monitor network properties. We will describe the details in section VI.

V. Problem Statement

Our investigations suggest that there are three major issues that should be addressed by HPCaaS platform developers: 1) Virtual Machines overhead; 2) Cloud Networking; 3) Multi-tenancy bursts. Ian Foster et al. in [25] when comparing Clouds with Grids, stated that "*The one exception that will likely be hard to achieve in Cloud Computing (but has had much success in Grids) are HPC applications that require fast and low latency network interconnects for efficient scaling to many processors*". Multi-tenant network in the cloud suffer from high latency, since unpredictable network bursts make the networking in the cloud significantly unstable.

This research focuses on the networking issues of HPCaaS by using the capabilities of Software-Defined Networking. The ideal solution to the cloud networking bursts issue advocates removal of the multi-tenancy of the cloud or decreasing the degree of multi-tenancy for HPCaaS platforms. Nevertheless, this will result in reducing resource efficiency which is one of the chief reasons to utilize cloud computing in the first place. A better solution advocates by ASETS which promotes continuous monitoring of the network bandwidths and taking prompt actions accordingly.

VI. ASETS: A SDN Empowered Task Scheduling System

ASETS adds a SDN-controller into an elastic configuration of HPCaaS platform to monitor network bandwidths during the task scheduling phase. Information provided by this SDN-controller enables the task scheduler to take the network state into the account when assigning tasks to the VMs. We propose a novel algorithm called SETSA (SDN-Empowered Task Scheduling Algorithm) to perform the task scheduling by the ASETS system. The proposed algorithm enables ASETS to utilize the cloud network bandwidth more efficiently and thus it will increase the performance of the HPCaaS platform in term of turnaround time of the submitted jobs.

A. The Architecture

Figure 2 shows the architecture of ASETS. In this figure, the HPC job scheduler divides jobs into multiple tasks and delivers them to the task scheduler queue. The tasks data are stored in a shared file system. The task scheduler uses the SETSA algorithm to assign tasks to the appropriate workers (VMs). Further, it sends the data from the shared file system to the designated VMs. When assigning tasks to the workers, the data transfer time from the shared file system to respective VM is also considered by the SETSA algorithm.

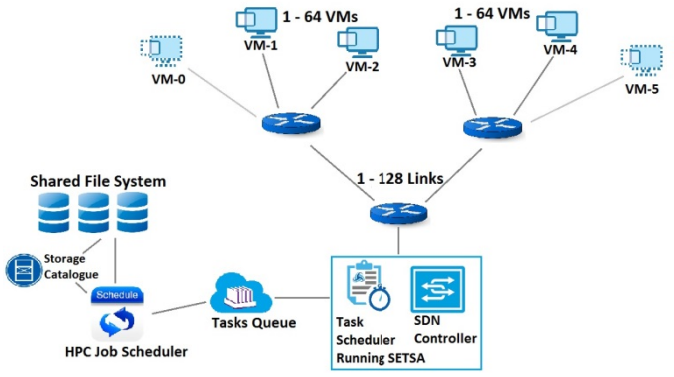

Figure 2: The ASETS Architecture overview

Scalability: In the conceptual architecture of ASETS shown in Figure 2, limited numbers of virtual machines are shown for simplicity. However, based on the computing power of the shared file system, tasks queue, and the schedulers of the

system, ASETS can exploit any additional numbers of virtual machines as needed. The scalability of ASETS seems to be depended on the centralized scheduler, the file system, and tasks queue. Nevertheless, highly available cloud-based file systems and queues (such as Amazon DynamoDB and SQS) can benefit the scalability of the system.

Elasticity: Based on the number of tasks submitted to the system, ASETS can dynamically add or remove VMs when needed. For example, virtual machines 0 and 6 in Figure 2 are not launched in the system yet but will be launched if needed. The dynamic allocation and de-allocation of virtual machines in the system is one of the primary benefits of the proposed HPCaaS platform. This efficient utilization method of the cloud resources will reduce the cost of the service.

B. SETSA (SDN-Empowered Task Scheduling Algorithm)

Definitions: We define the following definitions for the proposed algorithm:

D_j	Size of data for task$_j$ (Bytes)
A_i	Next available idle time for VM$_i$
L_i	Time to Launch VM$_i$
DT_{ij}	Transfer time for data of task$_j$ to VM$_i$
T_{ij}	Time to execute task$_j$ on VM$_i$
$Time_j$	Time that task$_j$ is finished
N	Maximum number of virtual machines
M	Number of tasks

Utilizing the SDN-Controller APIs, the bandwidths of all of the links becomes measurable. We define BW_{ij} as the maximum available bandwidth to transfer data of task$_j$ from the shared file system to VM$_i$.

$$\forall i=1 \text{ to N}, \ \forall j=1 \text{ to M}; \quad DT_{ij} = D_j / BW_{ij} \qquad (1)$$

If a virtual machine is not in *idle* state (i.e. running a task), then its next available time (A_i) will be the time when its running task is finished. For the virtual machines that are not launched yet, the next available time would be the time needed for the virtual machine to launch and become ready.

The task scheduler analysis all the information and calculates the finishing time of the task on all available virtual machines. Thus:

$$\forall i=1 \text{ to N};$$
$$\forall j=1 \text{ to M};$$
$$Time_j = \text{MIN} (A_i + DT_{ij} + T_{ij}) \qquad (2)$$

The SETSA algorithm is shown in Figure 3 which calculates Equations 2 for all the tasks in the queue and will assign them accordingly to the appropriate virtual machines.

The SETSA algorithm delivers the following benefits for the ASETS platform:

- Elastic task scheduling for HPCaaS
- Improved utilization of Network Bandwidth
- Reducing turnaround time of tasks in HPCaaS

```
While (true):
  For all tasks in tasks queue (j = 1 to M):
     j = pop_task_from_queue()

     Let N be the maximum number of VMs
     //Set Ai for deactivate VMs:

     for i = 1 to N:
          if VMi is not launched
               Ai = Li
     //Compute DTij for all links to all VMs
     for i = 1 to N:
          //Use SDN APIs to get the bandwidth of all links
          BWij = SDN_API_GET_BANDWIDTH(File_System, VMi)
          DTij = Dj / BWij

     //Find the most appropriate VM:
     Min = INFINITY
     for i = 1 to N:
          Timej = Ai + DTij + Tij
          if Timej < Min:
               Min = Timej
               vm_to_assign = i

     //Launch a VM if needed:
     if vm_to_assign is not luached:
          Launch vm_to_assign

     Assign task j to vm_to_assign
```

Figure 3: The SETSA algorithm

VII. CASE STUDIES

In this section we empirically model the system in a small scale of 3 workers for demonstration purpose. Examples are made to illustrate and compare two case scenarios for the algorithm. For simplicity, we assume that the workers are homogenous; therefore the time needed to finish a particular task is equal for all the workers. In both case scenarios a group of 8 tasks with different data sizes are assumed.

The two scenarios differ only in term of networking. In the first scenario, network bandwidth for the links connecting the shared file system to the workers, are assumed to be very close to each other. While in the second case, there is a significant difference between the links due to severe multi-tenancy of the system. We compare SETSA with FIFO and Round Robin for both scenarios in term of turnaround time.

Table 1: List of Tasks in the queue for both case scenarios

Task ID	Data Size (Mb)	Needed Time	Task ID	Data Size (Mb)	Needed Time
1	200	7	5	1500	25
2	800	15	6	1800	35
3	100	2	7	500	14
4	1200	19	8	700	18

A. Case Scenario I

In this example, network bandwidths for the links connecting the shared file system to virtual machines 1 to 3 are assumed to be 40Mbps, 50Mbps and 60Mbps. This is considered a small difference in the scale of a multi-tenant network of the cloud.

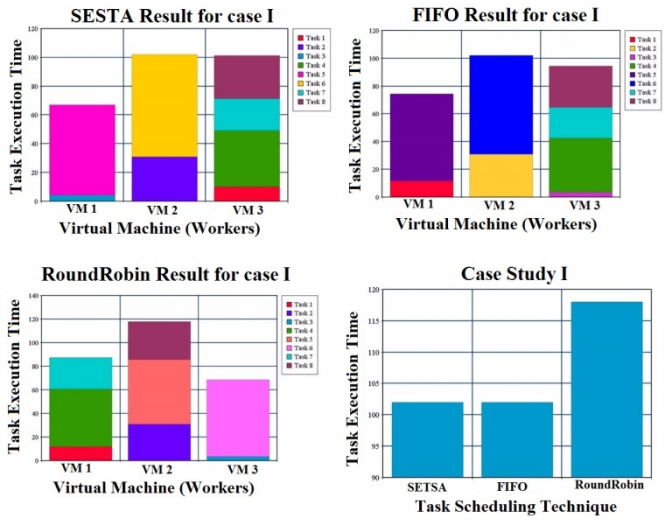

Figure 4: Case Scenario I Results

Figure 4 illustrates the result for the 3 different task scheduling algorithms. SETSA shows same result with FIFO and a slight improvement over RoundRobin. When we scale up the system to hundreds of workers, this improvement becomes considerable. However, we expect SETSA to show a much more significant performance improvement when the degree of multi-tenancy in the cloud increases. Case scenario II investigates the effect of increasing multi-tenancy.

B. *Case Scenario II*

In this example, the network bandwidths for the links are assumed to be 5 Mbps, 100Mbps, and 20Mbps. These values are selected based on the fact that the cloud is fully utilized by many tenants. Results in Figure 5 are considerably promising.

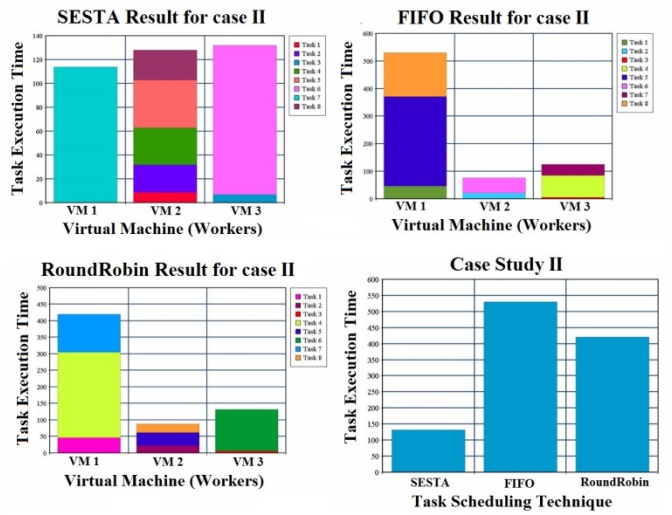

Figure 5: Case Scenario II Results

When the degree of multi-tenancy increases the SETSA algorithm improves the performance of HPCaaS in terms of task turnaround time and network utilization up to 75%. Our studies show that when increasing numbers of tenants are using a shared network and thus the cloud network becomes further unstable, SETSA will even play a more significant role in performance improvement.

VIII. Discussions

A. *Overhead of SETSA*

SETSA in comparison with traditional scheduling techniques such as FIFO or Round Robin, adds more overhead as it needs repetitive API calls to SDN controller with O(n) computation complexity for finding the best VM per task (where n is the number of virtual machines). Furthermore, SETSA shows promising results when the network is unstable (i.e. has more burst traffic) and have an effect on the scheduling. In a network with stable bandwidths and latency, SETSA may not expect to show promising results. Nevertheless, if the degree of multi-tenancy in ASETS increases, the network will start to show instability in bandwidth and latency. Therefore, SETSA will show the performance improvements in the ASETS architecture.

This observation shows that there is always a threshold in the degree of multi-tenancy after which SETSA improves the performance of ASETS by increasing network throughput efficiency and decreasing task turnaround time. A recent study [26] indicates that cloud service providers tend to offer services to the users more than their actual capacity by using oversubscription in order to increase revenue. However, this oversubscription lowers the performance. Oversubscription in other words means increasing the degree of multi-tenancy. Therefore, SETSA can play a critical role in such systems in order to stabilize the performance while the cloud service provider may increase the revenue by oversubscription.

B. *SETSAW: SETSA Window*

In each generation of SETSA, only one task is being assigned to one virtual machine. We can improve the performance of SETSA by assigning multiple tasks to several virtual machines per iteration. Thus, we propose SETSAW (SETSA Window) in which a number of tasks (equal to window size) are assigned to the appropriate VMs at a same time. This improves the performance of SETSA and reduces its overhead. The details of the SETSAW algorithm and its performance evaluations thereafter will be described in forthcoming papers.

IX. Future Works

The research continues by implementing the proposed system and its algorithms on a cloud infrastructure and a cloud simulator. The goal is to provide proof of the concept as well as performance evaluation of the system. Running real world HPC applications in a practical ASETS test-bed will make the performance improvement of SETSA much clearer.

SETSA has the potential to consider the cost of VMs as well. Adding the cost model can help the scheduler to decide about the target virtual machines such that the ratio of performance/cost could maximize. Currently, SETSA is designed only for shared file system architectures where data is sent to worker nodes from a centralized file system. The

ASETS architecture also follows that. Possible future works include expanding the idea to distributed file system architectures. We need to estimate the overhead cost of our proposed system along with the cost of virtualization compared with a traditional HPC platform.

X. Concluding Remarks

Our studies confirm previous findings regarding the current challenges for HPCaaS. We identified virtualization overhead, cloud networking, and multi-tenancy as the primary shortcomings of HPCaaS that need to be mitigated. This research addresses the problem of multi-tenancy of the network by proposing A SDN-Empowered Task Scheduling System (ASETS) for HPCaaS in the cloud as well as a novel task scheduling algorithm (SETSA) that utilizes SDN APIs to monitor network properties in the cloud for better scheduling. Ideas for improving the performance of SETSA are also proposed by introducing SETSAW (SETSA Window).

Previous task scheduling algorithms have not considered the network bandwidth property when assigning tasks to the virtual machines as workers. This is very crucial in the multi-tenant environment of the cloud for HPCaaS where network bandwidths are subject to unpredictable instability. Our empirical analysis indicates that SETSA has the potential to significantly improve HPCaaS in term of turnaround time up to 75% by better utilizing the network bandwidth.

Acknowledgement

This research was in part supported by the David and Amy Fulton Endowed Professorship in Computer Science at Bowling Green State University.

References

[1] Peter Mell, Tim Grance, "The NIST definition of cloud computing." 2011.

[2] Moustafa AbdelBaky, Manish Parashar, Kirk Jordan, Hyunjoo Kim, Hani Jamjoom, A Zon-Yin Shae, Gergina Pencheva, Vipin Sachdeva, James Sexton, Mary Wheeler, Mary F. Wheeler, "Enabling High-Performance Computing as a Service," Computer, vol. 45, no. 10, pp. 72-80, 2012.

[3] Nick McKeown, "Software-defined networking," in INFOCOM keynote talk, 2009.

[4] William Stallings, "Software-Defined Networks and OpenFlow," The Internet Protocol Journal, vol. 16, no. 1, 2013.

[5] Keith Kirkpatrick, "Software-Defined Networking," ACM Communication, vol. 56, no. 9, pp. 16-19, 2013.

[6] S. Azodolmolky, P. Wieder, R. Yahyapour, "Cloud computing networking: challenges and opportunities for innovations," IEEE Communications Magazine, vol. 51, no. 7, pp. 54-62, 2013.

[7] Moussa Taifi, Abdallah Khreishah, Justin Y. Shi, "Building a Private HPC Cloud for Compute and Data-Intensive Applications," International Journal on Cloud Computing: Services and Architecture (IJCCSA), vol. 3, no. 2, 2013

[8] Thamarai Selvi Somasundaram, Kannan Govindarajan, "CLOUDRB: A framework for scheduling and managing High-Performance Computing (HPC) applications in science cloud,"

[9] Jeremy Goecks, Anton Nekrutenko, James Taylor, The Galaxy Team, "Galaxy: a comprehensive approach for supporting accessible, reproducible, and transparent computational research in the life sciences," Genome Biology, vol. 11, 2010

[10] B. Liu, B. Sotomayor, R. Madduri, K. Chard, I. Foster, "Deploying bioinformatics workflows on clouds with galaxy and globus provision," in Third International Workshop on Data Intensive Computing in the Clouds, Salt Lake City, 2012.

[11] Philip Church, Andrzej Goscinski, Christophe Lefevre, "Exposing HPC and sequential applications as services through the development and deployment of a SaaS cloud," Future Generation Computer Systems, vol. 43, no. 44, pp. 24-37, 2015

[12] Eiji Kawai, "Can SDN Help HPC?," 2012 IEEE/IPSJ 12th International Symposium on Applications and the Internet, p. 210, (SAINT), 2012 Future Generation Computer Systems , vol. 34, pp. 47-65, 2014.

[13] Qin, Peng, Bin Dai, Benxiong Huang, and Guan Xu. "Bandwidth-Aware Scheduling with SDN in Hadoop: A New Trend for Big Data." arXiv preprint arXiv:1403.2800 (2014).

[14] D. Kondo, B. Javadi, P. Malecot, F. Cappello, "Cost-benefit analysis of Cloud Computing versus desktop grids," in IEEE International Symposium on Parallel & Distributed Processing (IPDPS), Rome, 2009.

[15] Douglas Eadline, "Admin Magazine, Moving HPC to the Cloud," [Online].Available:http://www.admin-magazine.com/HPC/Articles/Moving-HPC-to-the-Cloud.[Accessed November 2014].

[16] Piyush Mehrotra, Jahed Djomehri, Steve Heistand, Robert Hood, Haoqiang Jin, Arthur Lazanoff, Subhash Saini, Rupak Biswas, "Performance evaluation of Amazon EC2 for NASA HPC applications," in ACM 3rd workshop on Scientific Cloud Computing Date (ScienceCloud '12), New York, NY, USA, 2012.

[17] Roberto R. Expósito, Guillermo L. Taboada, Sabela Ramos, Juan Touriño, Ramón Doallo, "Performance analysis of HPC applications in the cloud," Future Generation Computer Systems, vol. 29, no. 1, pp. 218-229, 2013.

[18] Iman Sadooghi, Ioan Raicu. "Understanding the Cost of the Cloud for Scientific Applications." 2nd Greater Chicago Area System Research Workshop (GCASR) 2013

[19] Guohui Wang, TS Eugene Ng. "The impact of virtualization on network performance of amazon ec2 data center." INFOCOM, 2010 Proceedings IEEE. IEEE, 2010.

[20] Qiming He, Shujia Zhou, Ben Kobler, Dan Duffy, and Tom McGlynn. "Case study for running HPC applications in public clouds." In Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, pp. 395-401. ACM, 2010.

[21] Iman Sadooghi, Ioan Raicu, "CloudKon: a Cloud enabled Distributed task executiON framework." Illinois Institute of Technology, Department of Computer Science, PhD Oral Qualifier (2013).

[22] Huang, Kuo-Chan, Tse-Chi Huang, Mu-Jung Tsai, Hsi-Ya Chang. "Moldable Job Scheduling for HPC as a Service." In Future Information Technology, pp. 43-48. Springer Berlin Heidelberg, 2014.

[23] B. Nunes, Marc Mendonca, X. Nguyen, Katia Obraczka, Thierry Turletti. "A survey of software-defined networking: Past, present, and future of programmable networks." (2014): 1-18.

[24] N. McKewon, T. Anderson, H. Balakrishnan, G. Parulkar, L/ Peterson, J. Rexford, S. Shenker, J. Turner, "OpenFlow: Enabling Innovation in Campus Networks", ACM SIGCOMM Computer Communication Review, 38(2):69-74, 2008

[25] Foster, I.; Yong Zhao; Raicu, I.; Shiyong Lu, "Cloud Computing and Grid Computing 360-Degree Compared," Grid Computing Environments Workshop, GCE '08 , vol., no., pp.1,10, 12-16 Nov. 2008

[26] Rachel Householder, Scott Arnold, Robert Green, "On Cloud-based Oversubscription". arXiv:1402.4758