Data-Intensive HPC Tasks Scheduling with SDN to Enable HPC-as-a-Service

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

***Abstract*— Advances in Cloud Computing attracted scientists to deploy their HPC applications to the cloud to benefit from the flexibility of the platform such as scalability and on-demand services. Nevertheless, HPC applications can face serious challenges on the cloud that could undermine the gained benefit, if care is not taken. This paper starts first comparing the performance of several HPC benchmarks on a commodity cluster and Amazon public cloud to illustrate the confronted challenges. To mitigate the problem, we have introduced a novel approach called ASETS, "A SDN Empowered Task Scheduling System", to schedule data-intensive High Performance Computing (HPC) tasks on a Cloud environment. In this paper, we focus on the implementation and performance analysis of ASETS and its first algorithm called SETSA, (SDN Empowered Task Scheduling Algorithm). ASETS uses the "bandwidth awareness" capability of SDN to better utilize network bandwidths when assigning data intensive tasks to virtual machines (workers) in the cloud. This novel approach aims to improve the performance of HPC applications on the cloud in order the platform could provide efficient HPC-as-a-Service (HPCaaS). The paper briefly describes the ASETS architecture and its SETSA algorithm, and then focuses on the details of the implementation and performance analysis of ASETS and SETSA. Preliminary results indicate that ASETS provides substantial performance improvement for HPCaaS as the degree of multi-tenancy in the cloud increases. This result is significant since it indicate both the users and the cloud service providers can benefit from ASETS.**

Keywords— Cloud Computing, Task Scheduling, HPCaaS, Software-Defined Networking

I. INTRODUCTION

Cloud Computing provides a set of unique benefits such as resource pooling, cost efficiency, availability, and broad network access. These features have attracted scientists and scholars worldwide, including the community of High Performance Computing (HPC) developers and users. Nevertheless, moving the HPC applications to the cloud can face several key challenges, primarily, the virtualization overhead, multi-tenancy and network latency. The solution tends to be the emerging trend establishing a new cloud-based service known as HPC-as-a-Service (HPCaaS). This paper describes the details of the challenges of using cloud resources for HPC applications after discussing about the initial motivations. To illustrate performance fluctuations we compare cases where HPC applications are executed on Amazon public cloud with the equivalent configuration on a commodity HPC cluster. Our studies indicate that the multi-tenant environment of the cloud causes instable link bandwidth and high latency. This volatility can further result in a low performance for network sensitive HPC applications as the available network bandwidth becomes not predictable in the cloud. We argue that the instability of the cloud network bandwidth comprise a primary challenge of the HPC applications on the cloud. To remedy such instability, we suggest using programmable networking.

Software-Defined Networking (SDN) [1] as an emerging technology in networking appears to have sufficient potentials to address some of the challenges of HPC applications in the cloud. The idea of SDN is to decouple the network intelligence from the data plane into an independent layer called controller. Therefore, the control layer will have an overview across the network components and their dynamic configurations during run time. This information will provide substantial benefits for developers to build network-aware applications.

In a recent paper [2], we proposed a novel scheme (ASETS) on task scheduling for data intensive HPC tasks that utilize the capability of SDN to smooth the cloud networking bandwidth. A task scheduling algorithm (SETSA: SDN Empowered Task Scheduling Algorithm) is also proposed to run on top of ASETS. The proposed architecture actively monitors the network configurations during the runtime and redirect data related to the tasks over the links with highest instant available bandwidth capacity. This feature is extremely important in a multi-tenant cloud environment where the bandwidths of the links are rapidly changing. This paper argues that as the number of tenants in the cloud increases, SETSA would be more capable to mitigate the networking problem of HPCaaS platform. Furthermore, this paper focuses on the implementation and performance analysis of ASETS and SETSA. We provide preliminary results of the system on Amazon Cloud. While the obtained results are significant, we trust we need more in-depth performance analysis of the innovative system.

The rest of the paper is as follows: Section II describes the related work and recent achievements in addressing HPCaaS. Section III discusses the motivations of adapting HPCaaS and describes the challenges as well through careful experiments. Section IV briefly describes SDN. Section V overviews ASETS and its task scheduling algorithm. Section VI details the implementation methodologies of the system. Section VII

illustrates performance analysis and Section VIII concludes the paper and provides future directions.

II. Related Work

Works related to this research include attempts to address the challenges of HPCaaS as well as studies of improving performance of the HPC applications on the cloud. This section is dedicated to describe a number of recent achievements in this area.

Gilad et al. [3] explored the notion of HPCaaS by identifying the ability of running HPC applications simultaneously on a single cluster as the primary motivation. They believed that HPCaaS needs a specific scheduling strategy to achieve a reasonable performance, nevertheless their idea was that this scheduling strategy is application specific. As an example they proposed a smart scheduling algorithm for a subset of bioscience applications. Their results showed that the smart application specific scheduling algorithm increased the system productivity and efficiency.

General Purpose Graphics Processing Units (GPGPUs) provide performance improvement for scientific and HPC applications. However, the performance of a virtual GPU on the cloud cannot compete with its physical one. Younge et al. [4] studied the role of GPGPUs in Cloud Computing by providing the GPU-enabled virtual machines (VMs) and evaluating its performance for HPC scientific applications. Their proposed GPU-enabled VMs use "Pass-through" technique in the hypervisor and a virtual machines will have a direct access to GPU but through the hypervisor. A portion of HPC applications utilize GPGPUs and this research provides a solution for them to benefit from the Cloud.

Thamarai et al. [5] proposed a framework called Cloud Resource Broker (CLOUDRB) for scheduling HPC applications on the cloud. The framework follows a Particle Swarm Optimization method for allocating resources. A Discrete Event Simulation of the framework on Matlab indicates that CLOUDRB minimizes makespan, cost and job rejection ratio.

Gupta et al. [6] consider Cloud Computing as an alternative to supercomputers for a subset of HPC applications. They comprehensively analyzed the performance of running HPC applications on the cloud by comparing it with a range of platforms from a supercomputer to a commodity cluster. Although their conclusion was that the current clouds cannot substitute supercomputers, they can effectively complement them. They proposed an application-aware dynamic scheduling heuristics that could improve the performance of HPC applications in terms of average turnaround time and throughput.

AbdelBaky et al in [7] introduce a prototype to transform supercomputer into a cloud that supports accessibility of HPC resources through IaaS, PaaS and SaaS abstractions. In their experiment, they could dynamically scale resource of a supercomputer for a typical HPC application from 640 to 22,016 processors, spanning two systems in different continents. The performance of the running HPC application was neither reduced nor improved but the provided abstraction layer was simpler and more powerful.

HPC jobs are often in form of workflows where the sequence of tasks matters and the output of a completed task would become the input for the next task. Traditionally, HPC applications run on a dedicated hardware in a batch mode with single workflow scheduling. Clouds make it possible for HPC workflows to run simultaneously in a multi-tenant environment. Jiang et al. [8] proposed a mechanism to schedule simultaneous HPC workflows in a cloud oriented datacenter. The primary idea in this workflow scheduling mechanism is to fill the gaps between tasks. Using this method they could not only schedule HPC workflows in the cloud, but also increase the performance up to 18%.

Perhaps the closest related work to our study would be the one that applies the features of Software-Defined Networking (SDN) in addressing the challenges of HPCaaS in the cloud. To our knowledge, ASETS is the first one exploring this area by proposing a scheduling scheme for assigning data-intensive HPC tasks with SDN to virtual machines in the cloud. Nevertheless, Hadoop and Big Data applications that are sensitive about the network can also benefit from Software-Defined Networking capabilities. Qin et al. in a recent study [9] proposed a bandwidth-aware scheduling with SDN (BASS) mechanism for Hadoop jobs that can improve the performance in terms of job tunraround time.

III. HPC as a Serive (HPCaaS)

High Performance Computing applications often require huge computational power with careful consideration of the network. Job scheduling on supercomputers are traditionally batch. However, recently with the advent of multi-tenant cloud oriented data centers, HPC is facing a revolution. Moving HPC to the cloud to provide resources, infrastructure, applications and platforms of HPC in form of a service is called HPCaaS. This section describes the motivations as well as the challenges of moving HPC applications to the cloud.

A. HPCaaS Benefits

Cloud with turning the utility computing into reality have recently gained the attention of both service providers and users. The HPC community is no exception as the resource pooling, availability, cost efficiency, flexibility, on demand broadband access and several other benefits of the cloud are attractive to them. This research identifies the followings as the primary benefits of HPCaaS.

1) Cost Efficiency: HPCaaS makes it possible for cloud providers to run simultanous HPC jobs on their infrastructure to not only fill the gaps in the workflows with other jobs [8], but also to provide customizable, scalable, and elastic virtual clusters for users. This will result in a more efficient service and a maximized benefit. Users as well can also reduce costs by turning the capital expenditure to operational by renting services instead of buying the required infrastructure. A comprehensive study and evaluation of HPC applications on the cloud [10] indicates the cost efficiency of the cloud for small scale HPC applications.

2) Resource Utilization: Cloud providers benefit from a multi-tenant environment where they can utilize their resources by having simultanous HPC jobs. Moreover, users

can scale up and down their resources even during run-time according to the application deman.

3) Maintenance and Administration: The scientific community that contributes as the majaroty of HPC users often have limited computer ccience background. HPCaaS eliminates all the hassles of setting up an HPC cluster and maintenance of a powerful supercoputer for the users.

B. HPCaaS Chellenges

Unique requirements of current HPC applications such as demand of batch scheduling and direct access to dedicated hardware, fast dedicated interconnects, and low latency of the network do not match well with current cloud technology. In order for the HPC applications to have a competitive performance on the cloud, either the current applications or the current cloud technology need to be revised. We conducted a comprehensive experiment by comparing the performance of an HPC benchmark on Amazon AWS cloud with a physical commodity cluster to identify the following shortcomings of HPCaaS.

1) Cloud Networking: Network bandwidth and latency play very important roles in the performance of HPC applications on the cloud. Scientific applications often need fast intercommunication between parallel jobs and/or high bandwidth to transfer large volumes of data. To evaluate how well existing cloud technology performs in term of networking for HPCaaS, we used iPerf networking benchmark [11] in our experiments on Amazon EC2 c3.8xlarge instances. We ran the benchmark in 8 number of experiments each for 15 times.

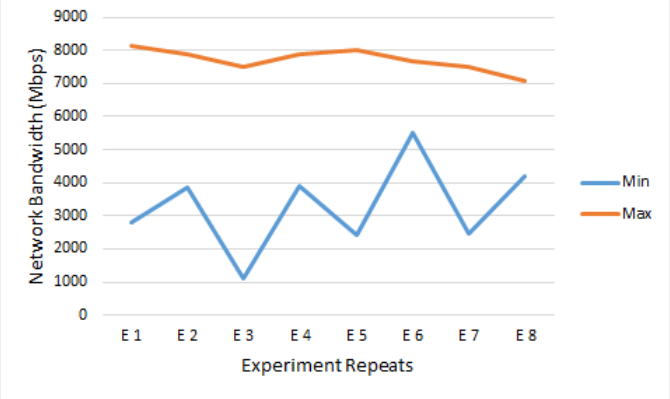

Figure 1. Network Bandwidth Performance on Public Cloud

Figure 1 shows a large variability in the performance of the cloud network in terms of bandwidth. This instable network bandwidth causes the HPC applications, in particular those with high network demand, to have an unpredictable performance. Due to the fact that the resources on the cloud are shared among many simultaneous running applications, sometimes the network links are extremely busy (e.g. the minimum measured bandwidth on E3 in Figure 1) and sometimes they are free (e.g. maximum measured bandwidth in E1 as shown in Figure 1). This effect will result in a zigzag behavior of the network bandwidth as illustrated in Figure 1. In another experiment we analyzed the network latency of the cloud and compared it with a commodity Rocks cluster. Figure 2 shows that HPCaaS can potentially suffer from high latency of the network.

Our experiments prove that existing networking methodologies in the cloud do not provide promising performance for HPCaaS and the multi-tenant environment of the cloud plays the most important role in the shortcoming of cloud networking. This research considers multi-tenancy of the cloud as a second challenge of HPCaaS.

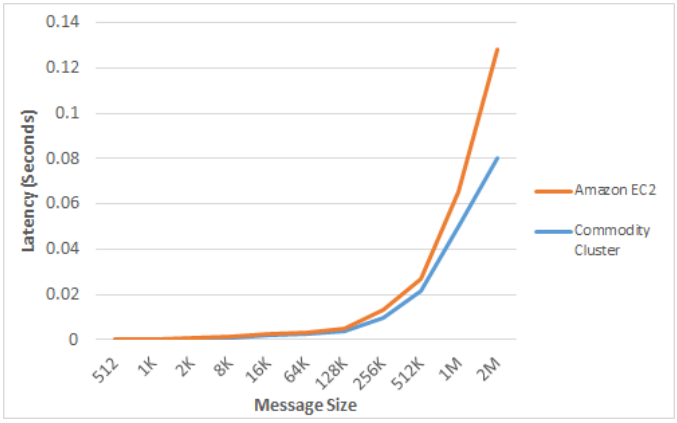

Figure 2. Network Latency of Public Cloud Compared to Commodity Cluster

2) Multi-tenancy: Although multi-tenancy is one of the motivations of adapting cloud computing technology, it is with great contrast with the requirements of HPC applications. Multi-tenancy enables cloud providers to share resources among multiple tenants to maximize profit. Nevertheless, HPC applications demand direct access to dedicated hardware using some sort of batch scheduling.

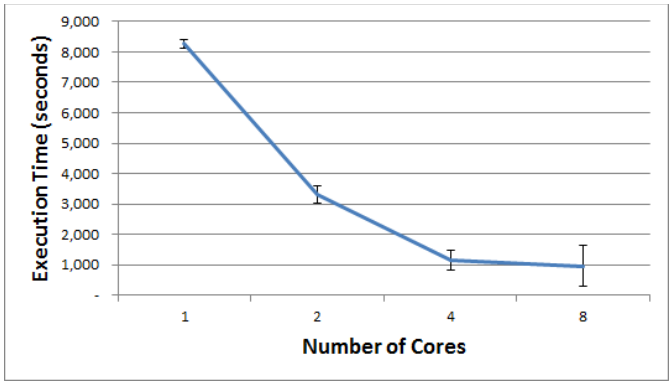

Figure 3. Speed-up for GROMACS benchmark running on Amazon EC2 instance

We conducted an experiment by running GROMACS benchmark [12] on a virtual instance of Amazon EC2 public cloud. GROMACS is a real-world scientific application used in molecular simulation. We repeated the experiment for 5 times and Figure 3 shows the result for the achieved speed-up. The error bars represent the standard deviation of the results and indicate that by increasing the number of cores, the diversity and variability of values we get in multiple experiments, increases. In other words, by scaling up the HPC applications on the cloud, the performance becomes more unpredictable. To support our results, in another experiment the performance of

Matrix Multiplication benchmark is evaluated on Amazon EC2. Figure 4 represents the efficiency achieved for the experiments and the error bars are again the standard deviations. These two experiments show how performance of HPC applications on the cloud is not predictable due to the shared resource and multi-tenant environment of the cloud. Moreover, the cloud clearly lowers the efficiency.

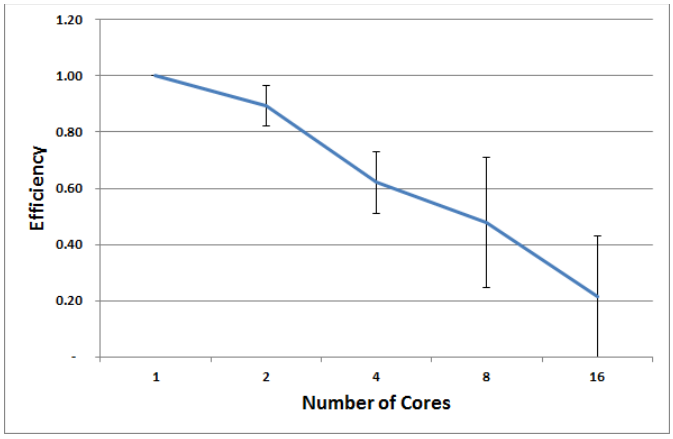

Figure 4. The efficiency achieved by running Matrix Multiplication benchmark on Amazon EC2 public cloud

There is a relatively huge gap between the average and best performance of running GROMACS or Matrix Multiplication benchmark on Amazon EC2. This gap is due to the fact that several tenants are using a shared resource and the performance of the application depends on the number of simultaneous running applications. It is worth mentioning that other experiments such as [10] confirm our findings and provide evidences that the multi-tenancy of the network is the major bottleneck and has the greatest influence in degrading the performance of HPC applications in the cloud.

3) Virtualization Overhead: Virtualization plays a key role in the cloud helping the cloud to have rapid elasticity, resource pooling, and flexibility. However, virtualization and in particular the hypervisor adds unwanted overhead by adding a software layer and preventing applications to have direct access to the hardware resources. This virtualization overhead is not the same for all types of hardware. For example, because of the hardware support, virtualization overhead for processors is significantly less than the overhead of network virtualization. For some hardware types such as GPUs, it is often more efficient to pass through GPUs than to have virtual GPUs [13].

IV. Software-Defined Networking (SDN)

Traditional networking provides great flexibility on the network edge for the developers to utilize various suits of protocols to enable the development of multi-purpose applications. This flexibility of the network at the edge can be considered as one of the primary reasons behind the success of the Internet. In contrast, the network in the core level is rather firm and inflexible. Any change or dynamic configuration is either practically difficult or highly expensive. With the rise of distributed applications and file systems, Big Data, Internet of Things, etc. the network traffic has switched from a mostly vertical pattern, to a more of a horizontal pattern [14]. In other words, modern datacenters tend to keep most of their data traffic within their internal distributed proximity rather than directing the traffic to the external network. As a result, there is enormous demand for more dynamic, flexible, and elastic network at the core. Programmable networks appear to provide excellent solutions compensating the inflexibility challenge of the core network and avoid expensive physical reconfiguration.

Software-Defined Networking (SDN) is an emerging technology that turns the notion of programmable networks into reality for the existing networking technologies [15]. The idea is to separate the controlling layer of the network from the data transfer layer and turn it into a programmable and dynamically configurable layer [16]. Figure 5 shows a conceptual architecture of a network managed by SDN. The network traffic is forwarded in the data plane based on the flow tables inside the switches. Records in the flow tables are assigned with commands issued by the SDN controller [17]. The most popular protocol for the SDN controller as of today is OpenFlow [18]. With a separate control plane in SDN architecture, the controller is capable of monitoring the whole network properties and can alter the configurations according to the users and applications demand during run-time using RESTful APIs.

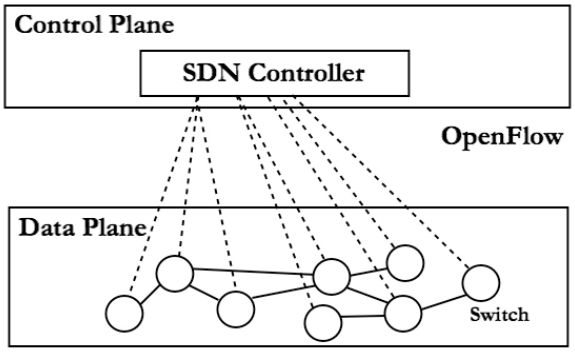

Figure 5. Conceptual Architecture in Software-Defined Networking

SDN plays an important role in cloud networking where the network is virtualized [19]. Each tenant becomes capable of having its private virtual network configuration and topology. ASETS aims to utilize the capabilities of SDN in assigning HPC tasks to virtual machines in the cloud. Accordingly, the SETSA algorithm benefits from the "bandwidth awareness" feature of the SDN controller to more efficiently schedule data-intensive tasks and hence dramatically improve the performance of the system.

V. A Reconfigurable Task Scheduler on the Cloud

Section III described the motivations of HPCaaS as well as its limitations and shortcomings. Our experiments identified the networking, multi-tenancy and virtualization overhead of the cloud as the primary challenges of HPCaaS. In a recent publication [2], we proposed a dynamic configurable scheme for scheduling HPC tasks on the cloud that utilizes the capabilities of Software-Defined Networking. The scheme called ASETS (A SDN Empowered Task Scheduling System) aims to mitigate the multi-tenancy of the cloud for

simultaneous HPC applications on the cloud. This section briefly describes the system and its primary scheduling algorithm named SETSA (SDN Empowered Task Scheduling Algorithm).

A. *ASETS: A SDN-Empowered Task Scheduling System*

ASETS consists of at least a queue of tasks, a task broker, a shared file system and a SDN controller for the virtual network. The tasks queue is populated by the HPC job scheduler and the input data needed for each task is stored in the file system. Workers are virtual machines that can be launched or terminated during run-time according to the demand. This elasticity of the system helps reducing the cost as well as efficiently utilizing the resources. Figure 6 illustrates a conceptual overview of the architecture and clearly shows that ASETS with the use of network virtualization and dynamic allocation of virtual machines on the cloud is highly scalable for HPC applications.

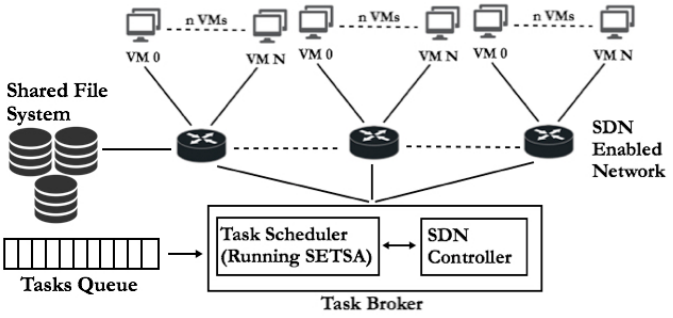

Figure 6. ASETS Conceptual Architecture

B. *SETSA: SDN-Empowered Task Scheduling Algorithm*

SETSA is the scheduling algorithm that runs in the task scheduler module of ASETS. Using APIs, SETSA benefits from a SDN controller that has an overview of the whole virtual network of ASETS during run time. The SDN controller monitors the network activities and provides the link bandwidth values for the task scheduler. This information will help SETSA schedule tasks with their corresponding data to the most suitable virtual machine. Our empirical analysis shows that as the number of simultaneous applications running on the shared cloud infrastructure (i.e. degree of the multi-tenancy) increases; SETSA will have a more influence in improving the performance of HPCaaS in term of job finishing time.

Figure 7 illustrates the result of our preliminary empirical analysis of the algorithm by comparing its performance with FIFO and Round-Robin as two popular scheduling algorithms. The results indicate that SETSA performs best when the cloud is under heavy utilization. As our experiments in section 3 showed, increasing the number of tenants accessing the cloud infrastructure simultaneously makes the performance of running HPC applications very unpredictable. This instability is primarily caused by the fluctuations of network bandwidths. SETSA attempts to make the task scheduling more compatible with the alternating network bandwidths by redirecting network traffic to more available links.

C. *Discussions*

1) SETSA Overhead: API calls and the communication with the SDN controller adds some unwanted overhead to the scheduling. Our experiments and performance analysis shows that this unwanted overhead is tolerable when there are enough multi-tenant applications running on the cloud. Nevertheless, in an underutilized cloud, the overhead of SETSA may make it a less desirable scheduling solution compared with other light scheduling algorithms such as FIFO.

2) SETSA Window: SETSA originally schedules one task at a time assigning it to a single virtual machine. However, the algorithm potentially can be parallel to improve the performance and lower the overhead. Currently, a parallel implementation of the algorithm (SETSAW: SETSA Window) is in the research and development phase.

3) Cloud Over-utilization: When resources in the cloud are over-utilized by several tenants, SETSA plays a more important role. A recent study [20] indicates that cloud providers may maximize benefit by oversubscribing cloud resources to the users. Nonetheless, this oversubscription lowers the performance of the service. Our investigation suggests that SETSA has the potential capability to stabilize the performance while the cloud service provider may increase the revenue by oversubscription of the resources.

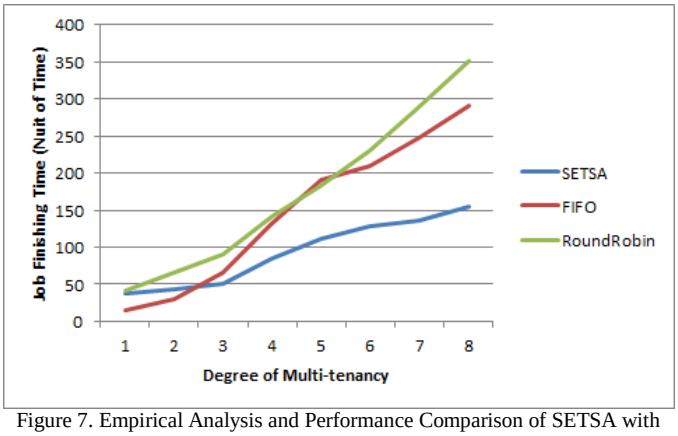

Figure 7. Empirical Analysis and Performance Comparison of SETSA with FIFO and Round-Robin

VI. Implementation Methods

This section describes our methodology to implement ASETS on both Amazon public cloud and a private OpenStack cloud. Depending on the infrastructure and platforms, there are several technologies that can be used to implement ASETS. In order to show the proof of the concept, we deployed an OpenStack cloud integrated with OpenDaylight [21] as the SDN enabler for the virtual network of the cloud. OpenDaylight is a community-led and industry-supported open source framework to accelerate adoption of SDN and Network Functions Virtualization (NFV).

A. *Private OpenStack Cloud*

We deployed a RedHat RDO [22] on our Dell commodity cluster of 6 compute nodes to have a private OpenStack cloud. The OpenStack manages cloud networking using a module named Neutron. In order to enable SDN on this cloud, we need

to configure Neutron to work with OpenDaylight using Open vSwitch and Modular Layer 2 (ML2) plug-in. Open VSwitch is multilayer virtual switch that enables SDN functionalities. And ML2 is a plug-in for Neutron to enable OpenStack benefit from layer 2 networking technologies. Figure 8 represents the conceptual overview of the integration of OpenStack with OpenDaylight to enable Software-Defined Networking for the private cloud.

One of the primary challenges of evaluating the performance of ASETS and SETSA on a private cloud is to build a multi-tenant environment. SETSA improves the performance of HPCaaS if the cloud resources are utilized enough by simultaneous tenants. In order to emulate such an environment for ASETS we set up several virtual cluster of 3 to 4 small scale compute nodes each running a Matrix Multiplication benchmark. This challenge only needs to be addressed in a private cloud setting as the Amazon public cloud resources are already fully utilized by real working tenants. Another shortcoming of the private OpenStack cloud for our experiments was the small scale of the implementation. The private cloud was built on top of a cluster of 6 compute nodes and over a total of 64 physical cores. Although the hardware configuration was enough to prove the concept, a larger scale of experiments was needed to show the elasticity and feasibility of ASETS on a real-world public cloud environment. To address such a problem, we implemented ASETS on Amazon AWS cloud as well.

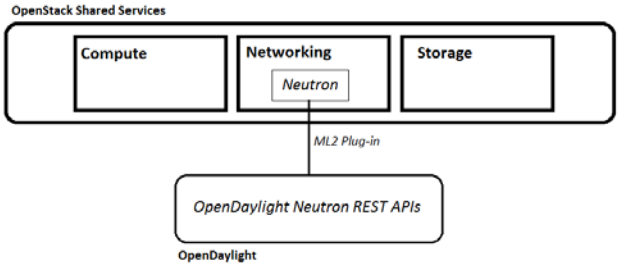

Figure 8. OpenStack and OpenDaylight Integration on a Private Cloud

B. *Public Amazon Cloud*

A real-world multi-tenant and dynamic public cloud is desired to evaluate the performance of ASETS and SETSA more accurately. Nevertheless, public cloud providers such as Amazon will consider a limited access in the infrastructure and hardware layer to the users, making it very difficult for us to deploy our SDN enabled cloud networking. Although, public cloud providers may utilize Software-Defined Networking capabilities for their networking infrastructures, such capabilities are blocked for public users for several reasons, primarily the security. To overcome this problem, we deployed a private virtual OpenStack cloud integrated with OpenDaylight on a virtual cluster on Amazon EC2. This will add another layer of virtualization to the system and therefore an unwanted overhead, yet makes it possible for us to utilize SDN capabilities for our own private cloud on top of a multi-tenant infrastructure to accurately evaluate performance of ASETS.

Figure 9 shows the conceptual architecture of our implementation of ASETS on Amazon public cloud. Amazon infrastructure provides a multi-tenant environment for us where we set up a private virtual OpenStack cloud with Software-Defined Networking enabled by OpenDaylight.

Amazon AWS provide cloud based services such as Amazon SQS (i.e. a queuing system) and powerful APIs besides typical virtual machines that make cloud-based developments a lot easier. In our implementation of ASETS on Amazon AWS, we utilized Amazon SQS for the tasks queue. Moreover, the Amazon EC2 Java APIs enable us to dynamically launch and terminate virtual machines on the cloud. Utilizing this capability, in order to make ASETS scale up and down according to the number of incoming tasks, we developed a module that actively monitors the size of the tasks queue. If the number of tasks in the queue exceeds a threshold, ASETS automatically launches new virtual machines to scale up. On the other hand if a virtual machine remains idle for a specific period of time, ASETS will terminate the virtual machine to save cost.

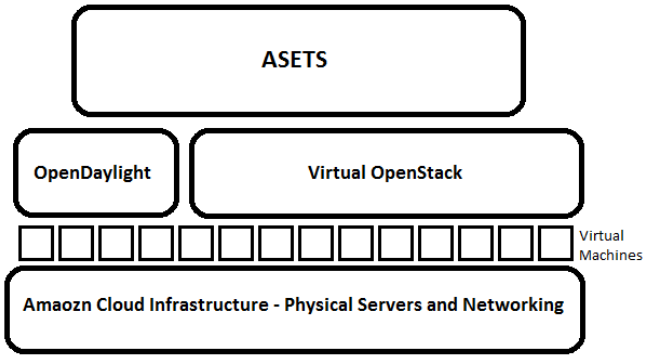

Figure 9. ASETS Conceptual Architecture on Amazon AWS

VII. Experiments Results and Analysis

We conducted the comprehensive performance analysis of ASETS and SETSA from three different perspectives; measuring the overhead of SETSA, performance evaluation of the system on a private cloud, and performance evaluation on Amazon public cloud. Our experiments indicate promising results for ASETS and its primary scheduling algorithm, SETSA. The proof of the concept implementation clearly indicates that ASETS is highly scalable and SETSA improves the performance of HPCaaS when the degree of multi-tenancy goes up.

A. *Measuring Overhead of SETSA*

Unlike regular scheduling algorithms like FIFO or RoundRobin, SETSA needs more calculations as it uses SDN APIs to monitor network bandwidths and make decisions accordingly. The extra calculations and process adds unwanted overhead that may influence the performance of the system. In order to measure the overhead of the system we compared the performance of ASETS when running SETSA with the time it is running a simple FIFO scheduling algorithm. The experiment was conducted on a private OpenStack cloud on a commodity cluster of six compute nodes running six virtual machines with zero multi-tenancy and repeated for 10 times. Data sizes and task granularities were randomly chosen for each repeat of the experiment.

Figure 10 compares the performance of SETSA in 10 numbers of executions with FIFO in a private cloud with no multi-tenancy. When there is not any simultaneous applications running on the cloud, network bandwidths remain stable and SETSA schedules HPC tasks the same as FIFO. The experiment shows that the undesired overhead of SETSA running on ASETS in such a case is approximately 5%. Further studies and experiments will indicate that this overhead is reasonably low and worthwhile.

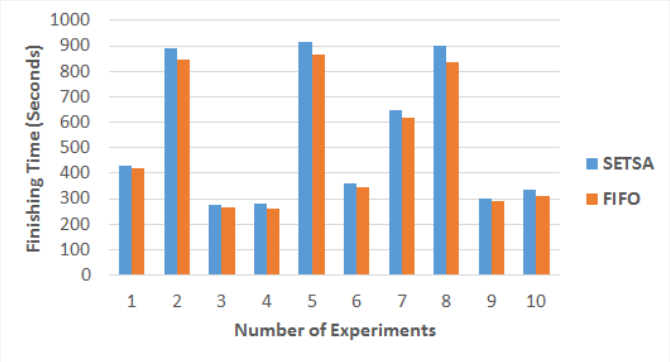

Figure 10. Comparing the Performance of SETSA with FIFO in Multiple Experiments

B. *ASETS on Private OpenStack Cloud*

Our empirical analysis of SETSA, previously, indicated that as the degree of multi-tenancy increases, SETSA performs better by mitigating the overhead of the multi-tenancy and improves the performance of HPCaaS. In order to evaluate SETSA in action, we artificially created a multi-tenant environment on our private OpenStack cloud by launching simultaneous virtual clusters. Each virtual cluster has 3 virtual machines and runs a Matrix Multiplication algorithm. Number of the virtual clusters running on our cloud indicates the degree of multi-tenancy.

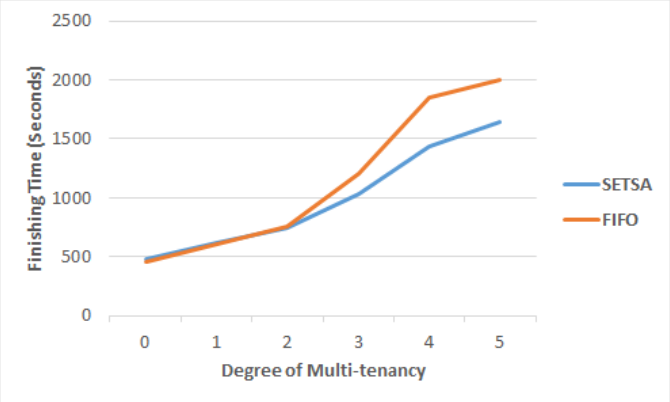

Figure 11. Performance of SETSA on Private OpenStack Cloud Based on the Degree of Multi-tenancy

Figure 11 confirms our empirical analysis. When the degree of multi-tenancy is low, SETSA performs almost the same as FIFO. However, as the number of simultaneous applications running on the cloud goes up, SETSA tends to mitigate the fluctuating available network bandwidths of the cloud and therefore increase the performance of HPCaaS in term of job finishing time. This performance improvement on a private OpenStack cloud running on a commodity cluster of 6 compute nodes is measured to be 18%. Nevertheless, in order to show how SETSA can improve performance in a real-world commercial HPCaaS environment, experiments in a larger scale are required. Therefore, we evaluated our implementation of ASETS running SETSA on Amazon public cloud.

C. *ASETS on Amazon Piblic Cloud*

Amazon AWS enables us to evaluate ASETS on a larger scale and on an inherently multi-tenant environment. Nevertheless, since access to the hardware and networking infrastructure of Amazon cloud is limited, we need to deploy our implementation of the cloud integrated with an SDN controller. Although this will result in an extra virtualization overhead, we will be able to evaluate the scalability of ASETS and SETSA.

In our experiment, we define the scale of the system by the number of virtual machines launched as the workers. SETSA is expected to perform better as the scale of the system goes up. Results confirm our assumption. Figure 12 shows how SETSA improves the performance of HPCaaS on public amazon cloud significantly up to 67%. As we scale up the system, number of network links and available bandwidths increase, letting SETSA to have a larger variety of choices to redirect data.

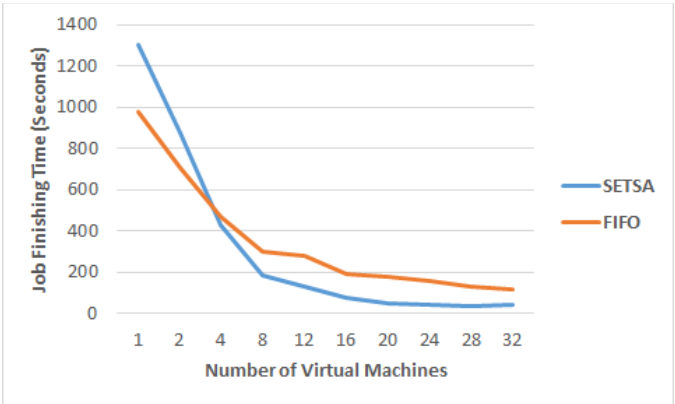

Figure 12. The performance of SETSA on Amazon Public Cloud

VIII. CONCLUSION AND FUTURE WORK

The paper comprehensively analyzed performance of the proposed task scheduling scheme for HPCaaS; ASETS (A SDN Empowered Task Scheduling System), alongside with its primary scheduling algorithm; SETSA (SDN Empowered Task Scheduling Algorithm). ASETS benefit from a Software-Defined Networking capabilities by leveraging a SDN controller in the architecture. SETSA utilizes the SDN controller to actively monitor the available network bandwidths in order to redirect data over the most suitable link. Our studies and experiments identified three primary challenges for HPCaas; cloud networking, cloud multi-tenancy, and the virtualization overhead. SETSA aims to improve the performance of scheduling data-intensive HPC tasks on the cloud in term of job finishing time by better utilizing the

network. Analyzing the performance of SETSA on two implementations of ASETS on a private OpenStack cloud and Amazon public cloud indicates that SETSA is capable of improving the performance of HPCaaS significantly up to 67%.

ASETS is a configurable, dynamic and scalable architecture capable of adapting other scheduling techniques for HPCaaS that may utilize Software-Defined Networking features as well. While SETSA showed a significant performance improvement for data-intensive HPC tasks, research and developments for other scheduling algorithms to utilize SDN capabilities are ongoing. Moreover, future works include the implementation and performance analysis of SETSA Window (SETSAW) which is a parallel version of SETSA as well. SETSAW aims to reduce the overhead of SETSA by assigning multiple tasks to workers (virtual machines) at a same time.

Furthermore, SETSA potentially is capable of considering the cost of virtual machines as well. Adding the cost model enables the scheduler to decide about the target virtual machines such that the ratio of performance/cost could be maximized. Also, while ASETS has a shared file system, another area of possible future studies includes expanding the idea to distributed file system architectures as well.

ACKNOWLEDGEMENT

The research was partially supported by David and Amy Fulton Endowed Professorship in Computer Science at Bowling Green State University.

REFERENCES

[1] Nick McKeown , "Software-defined networking," in *INFOCOM keynote talk 17, no. 2* , 2009.

[2] Saba Jamalian, Hassan Rajaei, "ASETS: A SDN Empowered Task Scheduling System for HPCaaS on the Cloud," in *2nd IEEE International Workshop on Software Defined Systems (SDS 2015) in conjunction with IEEE International Conference on Cloud Engineering (IC2E 2015)*, Tempe, AZ, USA, 2015.

[3] Shainer, G.; Tong Liu; Layton, J.; Mora, J., "Scheduling strategies for HPC as a service (HPCaaS)," in *IEEE International Conference on Cluster Computing and Workshops, 2009. CLUSTER '09.* , 2009.

[4] Andrew J. Younge, John Paul Walters, Stephen Crago, Geoffrey C. Fox, "Evaluating GPU Passthrough in Xen for HighPerformance Cloud Computing," in *IEEE 28th International Parallel & Distributed Processing Symposium Workshops*, 2014.

[5] Thamarai Selvi Somasundaram, Kannan Govindarajan, "CLOUDRB: A framework for scheduling and managing High-Performance Computing (HPC) applications in science cloud," *Future Generation Computer Systems*, vol. 34, pp. 47-65, 2014.

[6] Gupta, A. ; Faraboschi, P. ; Gioachin, F. ; Kale, L.V. ; Kaufmann, R. ; Lee, B.-S. ; March, V. ; Milojicic, D. ; Suen, C.H. , "Evaluating and Improving the Performance and Scheduling of HPC Applications in Cloud," *IEEE Transactions on Cloud Computing*, no. 99, 2014.

[7] AbdelBaky, M.; Parashar, M.; Hyunjoo Kim; Jordan, K.E.; Sachdeva, V.; Sexton, J.; Jamjoom, H.; Zon-Yin Shae; Pencheva, G.; Tavakoli, R.; Wheeler, M.F.,, "Enabling High-Performance Computing as a Service," *Computer*, vol. 45, no. 10, pp. 72-80, 2012.

[8] He-Jhan Jiang,Kuo-Chan Huang,Hsi-Ya Chang,Di-Syuan Gu,Po-Jen Shih, "Scheduling Concurrent Workflows in HPC Cloud through Exploiting Schedule Gaps," *Algorithms and Architectures for Parallel Processing*, vol. 7016, pp. 282-293, 2011.

[9] Peng Qin, Bin Dai, Benxiong Huang, Guan Xu, "Bandwidth-Aware Scheduling with SDN in Hadoop: A New Trend for Big Data," *arXiv:1403.2800v1*, 2014.

[10] Gupta, Abhishek, Laxmikant V. Kale, Filippo Gioachin, Verdi March, C. Suen, B. Lee, Paolo Faraboschi, Richard Kaufmann, and Dejan Milojicic, "The who, what, why and how of high performance computing applications in the cloud," in *Proceedings of the 5th IEEE International Conference on Cloud Computing Technology and Science*, 2013.

[11] Tirumala, Ajay, Feng Qin, Jon Dugan, Jim Ferguson, Kevin Gibbs, "Iperf: The TCP/UDP bandwidth measurement tool," [Online]. Available: http://dast.nlanr.net/Projects(2005).

[12] Lindahl, Erik, Berk Hess, and David Van Der Spoel, "GROMACS 3.0: a package for molecular simulation and trajectory analysis," *Molecular modeling annual*, vol. 7, no. 8, pp. 306-317, 2001.

[13] Wei-Shen Ou, Chao-Tung Yang, Yu-Tso Liu, Ching-Hsien Hsu, Hsien-Yi Wang, "On implementation of GPU virtualization using PCI pass-through," in *Proceedings of the 2012 IEEE 4th International Conference on Cloud Computing Technology and Science, CLOUDCOM '12*, 2012.

[14] Jim Theodoras, "The Huge Internet Change You Probably Missed," 6 November 2014. [Online]. Available: http://www.informationweek.com/cloud/infrastructure-as-a-service/the-huge-internet-change-you-probably-missed/a/d-id/1317190. [Accessed 14 February 2015].

[15] Nunes, B.A.A.; Mendonca, M.; Xuan-Nam Nguyen; Obraczka, K.; Turletti, T., "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1617 - 1634, 2014.

[16] Barath Raghavan, Martín Casado, Teemu Koponen, Sylvia Ratnasamy, Ali Ghodsi, Scott Shenker, "Software-defined internet architecture: decoupling architecture from infrastructure," in *11th ACM Workshop on Hot Topics in Networks (HotNets-XI). ACM*, New York, NY, USA, 2012.

[17] William Stallings, "Software-Defined Networks and OpenFlow," *The Internet Protocol Journal*, March 2013.

[18] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, Jonathan Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69-74, 2008.

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

[20] Rachel Householder, Scott Arnold, Robert Green, "On Cloud-based Oversubscription," *International Journal of Engineering Trends and Technology (IJETT)* , vol. 8, no. 8, 2014.

[21] "OpenDaylight," [Online]. Available: http://www.opendaylight.org/. [Accessed February 2015].

[22] "RedHat RDO," [Online]. Available: https://openstack.redhat.com/. [Accessed February 2015].