Measured Characteristics of FutureGrid Clouds for Scalable Collaborative Sensor-Centric Grid Applications

Geoffrey C. Fox
School of Informatics and Computing and Community Grids Laboratory, Indiana University, Bloomington IN 47408 USA
gcf@indiana.edu

Alex Ho, Eddy Chan
Anabas, Inc., 580 California Street, Suite 1600, San Francisco, CA 94104, USA
{alex.ho, research.eddychan}@anabas.com

1. ABSTRACT

The emergence of cloud technology has raised a renewed emphasis on the issue of scalable on-demand computing. Cloud back-end support of small devices such as sensors and mobile phones is one important application. We report our preliminary study of measured characteristics of distributed cloud computing infrastructure for collaboration sensor-centric applications on the FutureGrid [1, 2]. We focus on understanding the characteristics of the underlying network and its impact on multipoint, distributed cloud scalability. We report our findings in areas of performance, scalability and reliability at the network level using standard network performance tools. We measure data at the message level using the NaradaBrokering system [3-8] by the Indiana University Community Grids Laboratory which supports a large number of practical communication protocols. Results are also presented at the collaboration and communication applications level using the Anabas sensor-centric grid framework [9], a message-based sensor service management and sensor-centric application development framework.

Geographically distributed and heterogeneous clouds in the FutureGrid are used because of their support for scalable simulations. Our preliminary data indicates that a heterogeneous cloud infrastructure like FutureGrid coupled with a flexible collaborative sensor-centric grid framework is suitable for the study and development of new, scalable, collaborative sensor-centric system software and applications.

2. INTRODUCTION

Cloud computing services promise infrastructure resources to support application scalability. There are ample studies with systematic evaluation of this emerging information technology infrastructure [10-19] but few are on collaboration applications in general. There is even fewer work on leveraging heterogeneous clouds for real-time, distributed, collaborative sensor-centric applications in particular.

Increased use of collaborative sensing in a wide range of social, environmental, commercial and military types of applications is being driven by the need for better information about the environment or operational picture of interest and the advancement of technology which provides smaller, inexpensive and more capable sensors. For instances, some collaborative sensor-centric applications could be found in the fields of environmental monitoring, security surveillance, or target tracking [20]. One example of an interesting application is the sharing of filtered, neighborhood parking meter sensor information regarding available parking spots via local street signs to smartphones [21].

In recent years, technology has enabled a noticeable shifting from using few expensive and feature-riched sensors to deploying a large number of small, inexpensive commodity sensors with some level of direct or indirect networking capability. This technology trend should continue for the foreseeable future. Therefore, there will be a growing demand for scalable support of collaborative sensor-centric applications that could utilize a wide variety of sensor types and a massive number of globally deployed sensors for timely and actionable decision-support scenarios.

Our preliminary study is focused on the understanding of the suitability of distributed clouds for scalable, real-time collaborative sensor-centric applications. In particular, we consider network and transport layer characteristics of

distributed clouds, and performance characteristics of messages at some specific middleware and application layers.

The rest of the paper is organized as follows. Firstly, we define some key terminologies used, and revisit a general methodology used for an earlier study [22] on the Amazon Elastic Cloud Computing (EC2) infrastructure which is also being used here. Secondly, we review the technology in the collaborative sensor-centric grid framework [9] and message broker [4, 7, 8] that we leverage for this study. Thirdly, we give an overview of the FutureGrid [1, 2], the underlying heterogeneous distributed cloud infrastructure that we conduct the experiments on. Then, we discuss the experimental setup and report performance measurements in several scenarios. Lastly, we present conclusions and future work.

3. TERMINOLOGY AND METHODOLGY

Some technical terms could have different meaning when used by researchers in different communities or applications. This is particularly evidential in inter-disciplinary and emerging fields. For clarity and consistency, we highlight and recap several key terminologies we use throughout this paper and some that we reference.

We define collaboration as the general sharing of digital objects, and a sensor broadly as source of a time-dependent stream of information. We consider the definition of real-time is application-specific. In the case of a VoIP application, for instance, a round-trip latency of less than 300 milliseconds is considered acceptable timeliness while other collaborative applications could have more stringent real-time requirements. Grids have been extensively discussed in the literature. We adopt the view that grids represent the system formed by the distributed collections of digital capabilities that are managed and coordinated to support some sort of enterprise [23]. Clouds are commercially supported data-center models competing with compute grids and general-purpose computing centers [24]. Clouds do not supplant data grids.

In our earlier study of collaborative applications [22] on the Amazon EC2 distributed clouds, we devised a methodology to study the characteristics of distributed cloud computing infrastructure at the network, transport messages, and message-based collaboration applications levels. We were able to measure performance at the network layer and modeled typical multipoint VoIP application-level traffic at the transport layer. We had access to two clouds only, those at the Amazon EC2 US-East and Europe-West.

We adopt the same methodology in this study on FutureGrid. However, several significant differences exist between this study on the FutureGrid and that on the Amazon EC2. In this study we are able to conduct performance measurements on the network, transport messages, and message-based collaboration applications levels. We also extend our experiments on a homogeneous, 2-point, EC2 clouds to a heterogeneous, 4-point, Nimbus and Eucalyptus clouds.

4. COLLABORATIVE SENSOR-CENTRIC GRID FRAMEWORK

In order to generate and measure collaborative sensor-centric grid application traffic on distributed clouds we first need tools to build a sensor-centric grid, and to deploy and manage sensors. Instead of developing new tools and technology for building a sensor-centric grid and deploying and managing sensors, we reuse some of those capabilities we developed for an earlier project, namely a collaborative sensor-centric grid framework [9]. The framework supports the integration of a sensor-centric grid with collaboration and other grids, and provides a sensor interface and sensor-centric application interface. The framework also includes GB, a grid builder tool, for building, deploying, discovering and managing grid services and local and remote sensors.

GB follows the idea of constructing grids of grids, which assembles a multitude of subgrids into a mission-specific grid application. GB is a sensor management module which provides services for (a) defining sensor properties, (b) deploying sensors according to defined properties, (c) monitoring deployment status of sensors, (d) remote management irrespective of the locations of deployed sensors, and (e) distributed management irrespective of the location of the operator/user. Sensor streams are being shared in real-time with any sensor-centric applications that are developed using the API provided by the framework. A deployed sensor-centric grid communicates with (a) deployed sensors irrespective of sensor locations, (b) deployed sensor-centric applications irrespective of application locations, and (c) Grid Builder to mediate the collaboration among these three modules. In this framework, a primary function of a sensor-centric grid is to manage and broker message flows for sensor data and controls.

A typical scenario of a collaborative sensor-centric application using the framework encompasses a global deployment of a large number of sensors of different

types. Each sensor (for examples, video, GPS, video/audio, sound, light, temperature, gyroscope, ultrasonic, or RFID) gathers data from its environment and publishes it in real-time to a sensor-centric grid via a sensor adapter architecture. Some types of sensors can subscribe to other sensors' published data in the sensor-centric grid and provide filtering services, the results of which are published to the sensor-centric grid like any other sensors. A collaborative sensor-centric application provides the application logic and user-interface to orchestrate and manage real-time collaboration among only those sensors of interest for timely decision-support.

A demonstrative illustration of a sensor-centric application over the public Internet for collaborative, real-time sensor control and video motion detection was described in [9]. The demonstrative scenario involved the deployment of sensors in California, Indiana and Hong Kong. We summarily depict the scenario in Figures 1 and 2. Figure 1 shows some sensors, including a Lego NXT Tribot, deployed in Hong Kong. Figure 2 shows a snapshot of real-time motion control of the Hong Kong-deployed Tribot by a California-deployed WiiMote (Wii remote control) sensor, superimposed with the live filtering of a video stream from a Hong Kong-deployed webcam sensor by an Indiana-deployed software-based video motion detection sensor, which draws a bounding box around the area motion is detected.

Figure 1. Sensors and robot with sensor payload deployed in Hong Kong for a collaborative sensor-centric application demo.

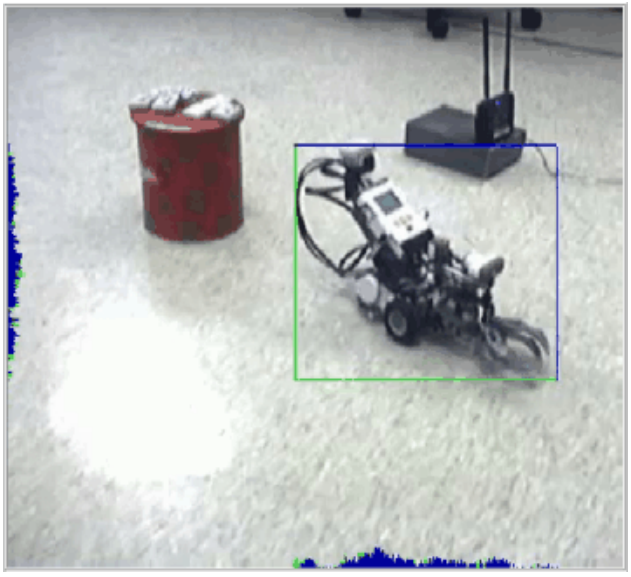

Figure 2. A real-time, collaborative control of a Tribot and sensing of motion in video stream.

For this study on FutureGrid, we develop another sensor-centric application using the framework. We also port GB to FutureGrid which enables us to build a sensor-centric grid, deploy sensors and sensor-centric applications to generate, measure and analyze specific application-level performance on FutureGrid distributed clouds.

5. FUTUREGRID

FutureGrid [2] is a part of the TeraGrid [25]. The aim of FutureGrid is to support the development of new system software and applications that can be simulated in order to accelerate the adoption of new technologies in scientific computing. The project has several computing clusters at different locations with a sophisticated virtual machine and workflow-based simulation environment to support research on cloud computing, multicore computing, new algorithms and software paradigms.

Unlike production cloud systems like the Amazon EC2, Microsoft Azure or Google App Engines for commercial applications, or TeraGrid for scientific computing, FutureGrid, by contrast, is oriented towards developing tools and technologies rather than providing production computational capacity [26].

FutureGrid is an infrastructure comprising currently approximately 4,000 cores at six sites - Indiana University (11 Teraflop IBM 1024 cores, 7 Teraflop Cray 684 cores, 5 Teraflop Disk Rich 512 cores), University of Chicago (7 Teraflop IBM 672 cores), University of California San Diego Supercomputing Center (7 Teraflop IBM 672 cores), University of Florida (3 Teraflop IBM

256 cores), Purdue University (4 Teraflop Dell 384 cores) and Texas Advanced Computing Center (8 Teraflop Dell 768 cores) - connected by a high-speed, network which is dedicated except for public link to Texas Advanced Computing Center. It is an experimental testbed that could support large-scale research on distributed and parallel systems, algorithms, middleware and applications. Figure 3 shows the connectivity of the six sites.

Figure 3. FutureGrid connectivity.

FutureGrid includes services accessible to users to run HPC (High Performance Computing) jobs such as MPI or OpenMP. It also supports several Grid and Cloud environments including the Eucalyptus and Nimbus Clouds.

Eucalyptus [27, 28] is an open source software platform that implements an Infrastructure-as-a-Service (IaaS)-style cloud computing. Eucalyptus provides an Amazon Web Services (AWS)-compliant, EC2-based web service interface for interacting with the cloud service. Additionally, Eucalyptus provides Walrus, an AWS storage-compliant service, and a user interface for managing users and images.

Nimbus is an open source toolkit that allows one to turn a cluster into an IaaS cloud [29]. Nimbus on FutureGrid allows users to run virtual machines on FutureGrid hardware. A Nimbus account user can easily upload custom-built virtual machine (VM) image or customize an image provided by FutureGrid. When a VM is booted, it is assigned a public IP address (and/or an optional private address). The VM is accessible by logging in as root via SSH. A user can then run services, perform computations, and configure the system as desired. After using and configuring the VM, the modified VM image can be saved to the Nimbus image repository.

6. EXPERIMENTAL SETUP

In our study we use up to four clouds on FutureGrid. The clouds we use are the Hotel (in University of Chicago running Nimbus), Foxtrot (in University of Florida running Nimbus), India (in Indiana University running Eucalyptus) and Sierra (in San Diego Supercomputing Center running Eucalyptus). The distributed clouds scenarios we setup either involve pairs of clouds or a group of four clouds. We choose m1.xlarge instances in the Eucalyptus cloud (each m1.xlarge instance is approximately equivalent to a 2-core Intel Xeon X5570 with 12 GB RAM) and 2 cores with 12 GB RAM in Nimbus. The selection of m1.xlarge VM in Eucalyptus is to ensure the Eucalyptus VMs we use for heterogeneous distributed clouds experiments have about the same level of computing resource as those in Nimbus.

To ensure acceptable precision of timing measurements in a distributed environment, we use the ntpdate command to synchronize the cloud instances we launch in our experiments with a time server in Chicago. In a Linux environment, which is the case in our experiments, the use of the NTP algorithm can usually maintain time synchronization to within 10 milliseconds over the public Internet.

For network-level measurement, we use the ping [30] and iperf [31] commands, both are commonly used by network administrators to monitor network characteristics. Ping is used to test the reachability of a host on an IP network and measure round-trip transmission time for ICMP echo request packets to and an ICMP response from the target host. In the process ping records any packet loss. Iperf is used to create TCP and UDP data streams, and measure network throughput.

For transport-level measurement, we use NaradaBrokering messages modeled after typical multipoint video conferencing traffic. NaradaBrokering servers work as an overlay transport layer to applications by taking care of all the communication among nodes composing the application using it NB is a middleware working as a glue connecting remote parts of a distributed application.

For application-level traffic generation and data gathering, we use our collaborative sensor-centric grid framework and the grid builder tool. In order to investigate scalability issues, it is not practical to deploy real sensors at large scale. Instead, we could deploy virtual sensors. The collaborative sensor-centric grid

framework supports development and deployment of real and/or virtual sensors. As an initial study on a multi-point distributed cloud, we deploy virtual GPS sensors only even though we have developed virtual sensors for RFID and WiiMote.

6.1. NETWORK-LEVEL MEASUREMENT

We run two types of experiments. They are (a) single-pair of cloud instances, one instance on each cloud, using iperf for measuring bi-directional throughput between all 2-combination distributed clouds of the set of four clouds selected (Hotel, Foxtrot, India, and Sierra); and (b) single-pair of cloud instances, one instance on each cloud, using the ping command together with the iperf command for measuring packet loss and round-trip latency under loaded and unloaded network between all 2-combination of the set of four clouds selected.

Figure 5 shows measured total bi-directional throughput using a range of one to sixty-four iperf connections for all 2-combination distributed clouds of the set of four selected clouds. The legend of Figure 5 shows all six combinations of 2-combination distributed clouds in our setup.

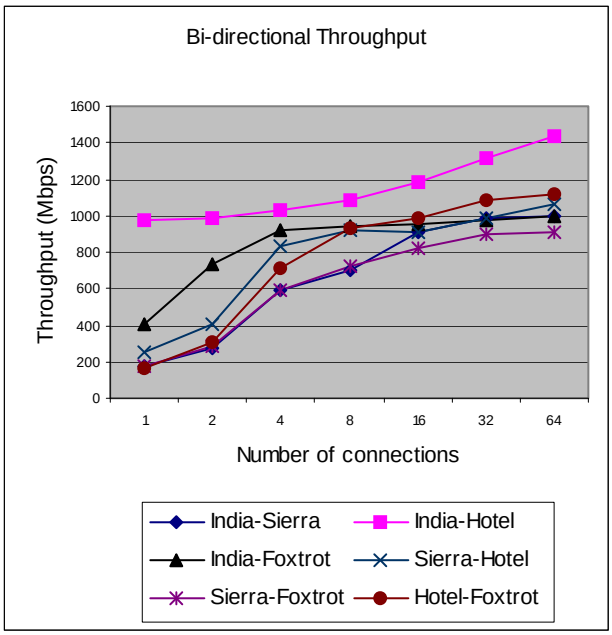

Figure 5. Throughput between distributed clouds.

While the maximum bi-directional throughput between any 2-combination ranges from 900 Mbps (on Sierra/Foxtrot pair) to 1,400 Mbps (on India/Hotel pair), we find the total iperf throughput in FutureGrid is over 800 Mbps when we connect any pair of cloud instances on distinct clouds with more than 16 connections in each direction.

As a comparison the Amazon EC2-US and EC2-EU distributed clouds sustained a throughput of 126 Mbps at 128 iperf connections [22]. However, we note that the maximum sustainable throughput had not been reached in the experiments reported in [22].

We use the ping tool to measure network latency and packet loss between two clouds. Figure 5 shows the throughput between any 2 clouds in our experiments either levels off or starts to level off at 32 iperf connections for all but the connection between India and Hotel.

For comprehensiveness the number of iperf connections should be increased up to the point the network is saturated to explore the elasticity of the current state of the FutureGrid network. We use iperf with 32 connections only to generate relatively heavy traffic of a loaded network for this initial study. We report measured network latency and packet loss in the connections between all 2-combination distributed clouds for both loaded and unloaded networks.

Our results (see Table 1) show ping packet loss rates in unloaded network for all the 2-combination of clouds were 0%; while the highest ping packet loss rate is 0.67% between the India/Hotel pair. The results indicate a highly reliable FutureGrid network under the experimental conditions.

Table 1: Inter-cloud ping packet loss rate

Instance Pair	Unloaded Packet Loss Rate	Loaded Packet Loss Rate
India-Sierra	0%	0.33%
India-Hotel	0%	0.67%
India-Foxtrot	0%	0%
Sierra-Hotel	0%	0.33%
Sierra-Foxtrot	0%	0%
Hotel-Foxtrot	0%	0.33%

For baseline information we measure ping round-trip latency between 2 cloud instances on Sierra for the unloaded case and loaded cases with 16 and 32 connections before we conduct the same experiment on distributed clouds. We find latencies for the unloaded and the two loaded cases between two virtual machines communicating on the same cloud no higher than 1.18 milliseconds. Thus, we could reasonably assume for the ping experiments on distributed clouds the measured round-trip latencies are mainly due to distance between clouds. Virtual machine overhead is negligible in these experiments.

Ping round-trip latency for all six combinations of pairs of clouds is measured. We find the lowest average round-trip latency of about 18 milliseconds between India and Hotel in a loaded condition (see Figure 6). India and Hotel has the shortest distance between any 2 of the four clouds; and thus, is expected to show the lowest round-trip latency here.

We observe the highest ping round-trip latency in a loaded network condition is about 145 milliseconds on the Sierra and Foxtrot connection (see Figure 7). Although the inter-cloud latency between Sierra and Foxtrot is the highest due to its longest distance between any two of the four selected clouds, we note that a round-trip latency below 300 milliseconds still meets a requirement for acceptable quality of service for collaboration applications with stringent network requirement like that of VoIP [32].

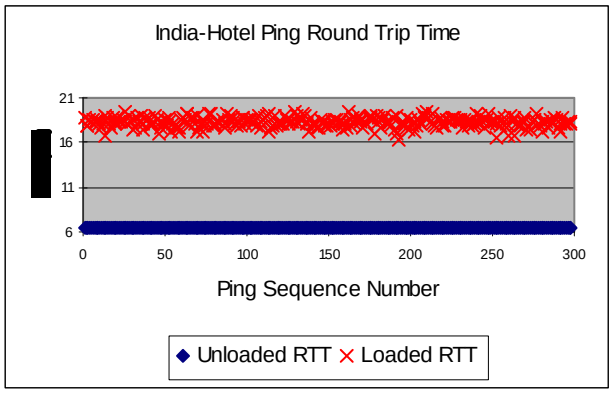

Figure 6. Ping round-trip latency between India and Hotel

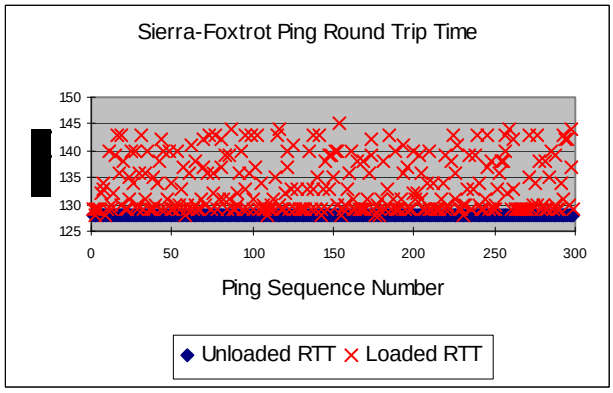

Figure 7. Ping round-trip latency between Sierra and Foxtrot

Overall, our limited initial results indicate that FutureGrid can sustain at least near 1 Gbps inter-cloud throughput and is a reliable network with low packet loss rate.

6.2. MESSAGE-LEVEL MEASUREMENT

In one set of experiments, extensive measurements are taken to evaluate the performance, stability and reliability characteristics for an increasingly larger collaboration session by injecting NaradaBrokering messages. We select the Foxtrot and Hotel, both running Nimbus environment, for our 2-cloud distributed experiments. A NaradaBrokering broker runs on Foxtrot. Simulated multiple meetings with groups of 20 participants run on Hotel.

Even though we have an actual multipoint video conferencing application in the Anabas Impromptu conferencing suite that could be used to generate real video traffic, it is easier and more practical to scale the number of users/participants at the message-level using NaradaBrokering clients than at the application level using real cameras and people for modeling a large-scale video session.

Figure 8 shows latency data on the inter-cloud connection between Foxtrot and Hotel. The average latency incurred in a single meeting with up to about 2,400 participants is below 50 milliseconds. Average latency jumps rapidly when the number of participants in a single meeting is more than 2,400. However, if a large meeting is divided into multiple smaller ones, we find that distributed clouds could sustain a higher aggregate total number of participants. In our experiments we find the average latency can be maintained below 50 milliseconds with 150 meetings, each of which has 20 participants; that is, a total of 3,000 participants.

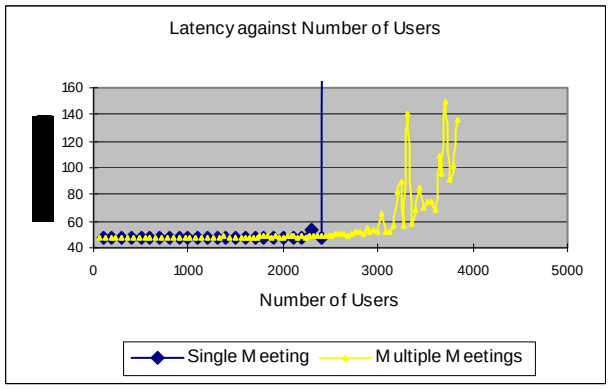

Figure 8. Average latencies of single and multiple video meetings.

The average latency result indicates that multiple smaller meetings balance the work of a NaradaBrokering broker better. Also reflected from the experiments is that there is message backlog on a single broker when there are more than 2,400 participants in a single meeting or 3,000 participants in multiple meetings. When there is message

backlog on a message broker, latency will increase rapidly. Of course NaradaBrokering can support multiple distributed brokers to control a collaboration or sensor network, so limits shown in Figure 8 represent limits of a single broker and not of the system. Clouds are attractive as they support the auto-scaling needed to add brokers on demand.

Overall, our limited initial results of message-based experiments indicate that FutureGrid can sustain a throughput close to its implemented capacity of 1 Gbps between Foxtrot and Hotel. The multiple meetings experiment also shows clouds can support publish-subscribe brokers effectively. Note the limit of around 3000 clients in Figure 8 was reported as 800 in earlier work [5] showing any degradation in server performance from using clouds is more than compensated by improved server performance.

6.3. APPLICATION-LEVEL MEASUREMENT

In this section, we discuss measurements of the scalability of multipoint distributed clouds on FutureGrid for collaborative sensor-centric applications. While a main objective of our research plan is to quantify the CPU, memory and communication requirements of a broad class of naturally distributed and highly scalable collaborative sensor-centric grid applications on the underlying distributed cloud architectures, we report our initial observations of one such applications, namely the collaborative sensor-centric grid framework [3], running on several distributed cloud scenarios on FutureGrid infrastructure as a starting point.

We develop and use virtual GPS sensors that we modeled after real GPS sensors. These are functional virtual sensors with reasonable design but their implementations at this stage are not optimized in any way. Each virtual GPS sensor streams information to the sensor-centric grid at a rate of 1 message per second. A sensor-centric grid application consumes all the sensor streams and computes message latency and jitter for a range of deployed sensors.

We first establish a performance baseline by deploying as many virtual GPS sensors as possible in one cloud instance without hitting any critical bottlenecks in CPU or RAM. When we deploy 100 virtual GPS sensors in an instance in India cloud, we observe the sensors continues running even though both idle CPU and unused RAM are at critically low level, with idle CPU at 7% and unused RAM at 1 GB. Since our primary focus is on distributed cloud communication characteristics for scalable collaborative real-time sensor-centric applications, we want to avoid running into CPU or RAM bottlenecks in our scalability experiment. When the number of deployed sensors in a single cloud instance is lowered to 60, we observe idle CPU at about the 35% level.

We conduct 2 different experiments. They are (a) establishing a baseline measurements in a single instance in one cloud only by deploying as many virtual sensors as possible; and (b) measurements of the communication characteristics by deploying up to 50 virtual GPS sensors in a single instance in each of the four selected clouds; that is, a total of up to 200 virtual GPS sensors are deployed in the experiment.

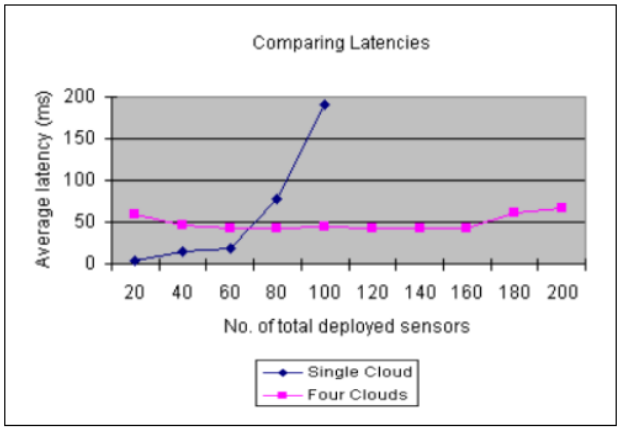

Figure 9. Comparing average latency of a single cloud and 4-point distributed cloud.

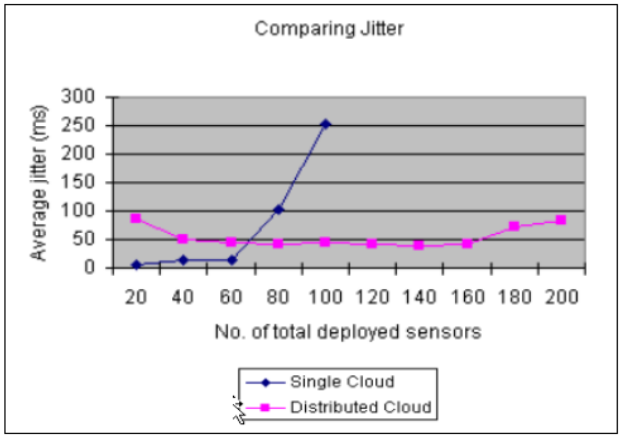

Figure 10. Comparing average jitter of a single cloud and 4-point distributed cloud.

There are three important observations related to scalability that could be made. Firstly, as shown in Figure 9, in the case of using a single instance in one cloud only for deploying sensors, the maximum number of virtual GPS sensors that could be stretched in a deployment is 100 but the instance shows a critically high CPU and RAM utilization. Such low levels of unused resources in an instance have a high risk of running out of resources and become unstable. In the case of running a single

instance in each of the four selected distributed clouds, it has a much lower level of resource utility, and will be more stable and suitable for long running simulations. Secondly, even though the case of using a single instance on a single cloud could be pushed to deploy 100 virtual GPS sensors, the average latency starts to grow rapidly after deploying 60 sensors. At the level of 80 deployed sensors, the average latency is higher than that of the case of the 4-point distributed cloud at the level of 200 deployed sensors. We notice that in the distributed case, the average latency is relatively constant and sufficiently low even for demanding network applications like VoIP [30], and with small variations only when sensor deployment is scaled up from 20 to 200. Thirdly, a similar pattern is observed in the comparison of the average jitter for the two cases (see Figure 10). In the case of sensor deployment in a single instance in one cloud only, average jitter is low until after deploying 60 sensors. At the level of 80 deployed sensors, the average jitter is already higher than that of the distributed case for 200 deployed sensors.

Overall, our limited initial results indicate distributed clouds has an encouraging potential to support scalable collaborative sensor-centric applications that have stringent throughput, latency, jitter and reliability requirements.

6.4. CONCLUSION

We conducted three types of experiments on FutureGrid to understand its performance characteristics in distributed clouds setting to support scalable collaborative sensor-centric applications. We ported the Grid Builder to FutureGrid and developed virtual GPS sensors for managing the scaling of application-level deployed sensors to a large number. We measured FutureGrid distributed clouds characteristics at the network, transport and application levels. Although this study is preliminary, we observe satisfactory performance characteristics for network, CPU and memory demanding simulations that are used as tools in our experiments. We conclude that coupling a flexible sensor-centric grid framework with a heterogeneous distributed clouds infrastructure like FutureGrid has the potential to effectively support the study of large-scale, collaborative sensor-centric applications that have stringent real-time and quality of service requirements.

Future work includes a better understanding of how to fully utilize the potential of a single instance to confidently simulate the optimal or near-optimal number of sensors possible without worrying about system abnormality due risks of running out of resources in an instance. Scalability in terms of using more instances per cloud should be incorporated to augment scalability in the number of distributed clouds.

KEYWORDS: distributed cloud, heterogeneous cloud, collaboration, sensor-centric applications, scalability, FutureGrid

6.5. ACKNOWLEGMENTS

We thank Bill McQuay of AFRL, Ryan Hartman of Indiana University and Gary Whitted of Ball Aerospace for their important support of the work. This material is based upon work supported in part by the National Science Foundation under Grant No. 0910812 to Indiana University for "FutureGrid: An Experimental, High-Performance Grid Test-bed." Other partners in the FutureGrid project include U. Chicago, U. Florida, San Diego Supercomputer Center - UC San Diego, U. Southern California, U. Texas at Austin, U. Tennessee at Knoxville, U. of Virginia.

7. BIOGRAPHY

GEOFFREY C. FOX received a Ph.D. in Theoretical Physics from Cambridge University and is now the Associate Dean for Research and Graduate Studies at the School of Informatics and Computing Indiana University Bloomington and professor of Computer Science, Informatics, and Physics at Indiana University where he is director of the Community Grids Laboratory. He previously held positions at Caltech, Syracuse University and Florida State University.

ALEX HO is the CEO of Anabas, Inc. He was a staff scientist with the IBM Research Division and the Caltech Concurrent Computation Program for over ten years. He was the founder and co-founder of several Silicon Valley startups in the areas of collaboration and Internet media technology.

EDDY CHAN is an R&D engineer. He has conducted extensive research in ad-hoc wireless network and Voice over IP, and is focusing on message-based collaboration technology.

8. REFERENCES

1. Geoffrey Fox. *FutureGrid Platform FGPlatform: Rationale and Possible Directions (White Paper)*. 2010 [accessed 2010 June 12]; Available from: http://grids.ucs.indiana.edu/ptliupages/publications/FGPlatform.docx.
2. *FutureGrid Homepage*. [accessed 2011 January 19]; Available from: http://www.futuregrid.org.

3. Wenjun Wu, Geoffrey Fox, Hasan Bulut, Ahmet Uyar, and Tao Huang, *Special Issue on Voice over IP edited by John Fox, P. Gburzynski: Service Oriented Architecture for VoIP conferencing* Theory and Practice of the International Journal of Communication Systems April 13, 2006. **19**(4): p. 445-461. DOI:http://dx.doi.org/10.1002/dac.803. http://grids.ucs.indiana.edu/ptliupages/publications/soa-voip-05.doc

4. Shrideep Pallickara, Hasan Bulut, Pete Burnap, Geoffrey Fox, Ahmet Uyar, and David Walker. *Support for High Performance Real-time Collaboration within the NaradaBrokering Substrate*. 2005 May [accessed 2011 March 11]; Available from: http://grids.ucs.indiana.edu/ptliupages/publications/NB-Collaboration_update.pdf.

5. Ahmet Uyar and Geoffrey Fox, *Investigating the Performance of Audio/Video Service Architecture I: Single Broker*, in *IEEE International Symposium on Collaborative Technologies and Systems CTS05*. May, 2005, IEEE. St. Louis Missouri, USA. pages. 120-127. http://grids.ucs.indiana.edu/ptliupages/publications/SingleBroker-cts05-submitted.PDF. DOI: http://doi.ieeecomputersociety.org/10.1109/ISCST.2005.1553303.

6. Ahmet Uyar and Geoffrey Fox, *Investigating the Performance of Audio/Video Service Architecture II: Broker Network*, in *International Symposium on Collaborative Technologies and Systems CTS05*. May, 2005, IEEE. St. Louis Missouri, USA. pages. 128-135. http://grids.ucs.indiana.edu/ptliupages/publications/BrokerNetwork-cts05-final.PDF. DOI: http://doi.ieeecomputersociety.org/10.1109/ISCST.2005.1553304.

7. NaradaBrokering. *Scalable Publish Subscribe System*. 2010 [accessed 2010 May]; Available from: http://www.naradabrokering.org/.

8. Pallickara, S. and G. Fox, *NaradaBrokering: a distributed middleware framework and architecture for enabling durable peer-to-peer grids*, in *ACM/IFIP/USENIX 2003 International Conference on Middleware*. 2003, Springer-Verlag New York, Inc. Rio de Janeiro, Brazil.

9. Geoffrey Fox, Alex Ho, Rui Wang, Edward Chu, and Isaac Kwan, *A Collaborative Sensor Grids Framework*, in *2008 International Symposium on Collaborative Technologies and Systems (CTS 2008)*. May 19-23, 2008. The Hyatt Regency Irvine, Irvine, California, USA. http://grids.ucs.indiana.edu/ptliupages/publications/CTS08_paper_final.pdf.

10. Keith R. Jackson, Lavanya Ramakrishnan, Karl J. Runge, and Rollin C. Thomas, *Seeking supernovae in the clouds: a performance study*, in *Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing*. 2010, ACM. Chicago, Illinois. pages. 421-429. http://dsl.cs.uchicago.edu/ScienceCloud2010/p07.pdf. DOI: 10.1145/1851476.1851538.

11. Keith R. Jackson, Lavanya Ramakrishnan, Krishna Muriki, Shane Canon, Shreyas Cholia, J. Shalf, Harvey J. Wasserman, and N.J. Wright, *Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud*, in *CloudCom*. 2010, IEEE. Indianapolis. http://www.nersc.gov/projects/reports/technical/CloudCom.pdf.

12. Wei Lu, Jared Jackson, Jaliya Ekanayake, Roger Barga, and Nelson Araujo, *Performing Large Science Experiments on Azure: Pitfalls and Solutions*, in *CloudCom*. 2010, IEEE. Indianapolis. pages. 209-219.

13. Wei Lu, Jared Jackson, and Roger Barga, *AzureBlast: A Case Study of Developing Science Applications on the Cloud*, in *ScienceCloud: 1st Workshop on Scientific Cloud Computing co-located with HPDC 2010 (High Performance Distributed Computing)*. June 21, 2010, ACM. Chicago, IL. http://dsl.cs.uchicago.edu/ScienceCloud2010/p06.pdf.

14. Jaliya Ekanayake and Geoffrey Fox, *High Performance Parallel Computing with Clouds and Cloud Technologies*, in *First International Conference CloudComp on Cloud Computing*. October 19 - 21, 2009. Munich, Germany. http://grids.ucs.indiana.edu/ptliupages/publications/cloudcomp_camera_ready.pdf.

15. Judy Qiu, Thilina Gunarathne, Jaliya Ekanayake, Jong Youl Choi, Seung-Hee Bae, Hui Li, Bingjing Zhang, Yang Ryan, Saliya Ekanayake, Tak-Lon Wu, Scott Beason, Adam Hughes, and Geoffrey Fox, *Hybrid Cloud and Cluster Computing Paradigms for Life Science Applications*, in *11th Annual Bioinformatics Open Source Conference BOSC 2010*. July 9-10, 2010. Boston. http://grids.ucs.indiana.edu/ptliupages/publications/HybridCloudandClusterComputingParadigmsforLifeScienceApplications.pdf.

16. Jaliya Ekanayake, Thilina Gunarathne, Judy Qiu, Geoffrey Fox, Scott Beason, Jong Youl Choi, Yang Ruan, Seung-Hee Bae, and Hui Li, *Applicability of DryadLINQ to Scientific Applications*. January 30, 2010, Community Grids Laboratory, Indiana University. http://grids.ucs.indiana.edu/ptliupages/publications/DryadReport.pdf.

17. Katarzyna Keahey, Mauricio Tsugawa, Andrea Matsunaga, and Jose Fortes, *Sky Computing*. Internet Computing, IEEE, 2009. **13**: p. 43-51. DOI:http://doi.ieeecomputersociety.org/10.1109/MIC.2009.94. http://www.nimbusproject.org/files/Sky_Computing.pdf

18. Mauricio Tsugawa and Jose Fortes. *ViNe: Managed virtual networks in Grids*. [accessed 2010 20 November]; Available from: http://vine.acis.ufl.edu/.
19. M. Tsugawa and J.A.B. Fortes, *A virtual network (ViNe) architecture for grid computing*, in *International Parallel & Distributed Processing Symposium*. 2006, IEEE. Rhodes Island, Greece. pages. 123.
20. Biswas, S., S. Gupta, F. Yu, and T. Wu, *A networked mobile sensor test-bed for collaborative multi-target tracking applications.* Wirel. Netw., 2010. **16**(5): p. 1329-1344. DOI:10.1007/s11276-009-0206-x
21. John Markoff. *Can't Find a Parking Spot? Check Smartphone*. 2008 July 21 [accessed 2011 March 12]; New York Times Available from: http://www.nytimes.com/2008/07/12/business/12newpark.html.
22. Geoffrey Fox, Alex Ho, Eddy Chan, and William Wang, *Measured Characteristics of Distributed Cloud Computing Infrastructure for Message-based Collaboration Applications*, in *International Symposium on Collaborative Technologies and Systems CTS 2009*. May 18-22, 2009, IEEE. The Westin Baltimore Washington International Airport Hotel Baltimore, Maryland, USA. pages. 465-467. http://grids.ucs.indiana.edu/ptliupages/publications/SensorClouds.pdf. DOI: 10.1109/cts.2009.5067515.
23. Fox, G., *Grids of Grids of Simple Services.* Computing in Science and Engg., 2004. **6**(4): p. 84-87. DOI:10.1109/mcse.2004.10. http://grids.ucs.indiana.edu/ptliupages/publications/Cisegridofgrids.pdf
24. Geoffrey Fox. *Cloud Computing for ADMI*. 2010 [accessed 2011 March 11]; ADMI Board Meeting and faculty workshop at Elizabeth City State University Available from: http://grids.ucs.indiana.edu/ptliupages/presentations/ECSU-Dec16-10.pptx.
25. *TeraGrid open scientific discovery computational infrastructure*. [accessed 2010 November 20]; Available from: https://www.teragrid.org/.
26. Geoffrey Fox. *Interview on FutureGrid*. 2009 September 29 [accessed 2011 March 11]; by Sander Olson Available from: http://nextbigfuture.com/2009/09/interview-of-geoffrey-fox-director-of.html.
27. Nurmi D., Wolski R., Grzegorczyk C., Obertelli G., Soman S., Youseff L., and Zagorodnov D., *The Eucalyptus Open-Source Cloud-Computing System*, in *9th IEEE/ACM International Symposium on Cluster Computing and the Grid. CCGRID '09*. 18-21 May, 2009. Shanghai. pages. 124-131. DOI: 10.1109/CCGRID.2009.93.
28. *Eucalyptus Open Source Cloud Software*. Available from: http://open.eucalyptus.com/.
29. *Nimbus Cloud Computing for Science*. [accessed 2011 March 11]; Available from: http://www.nimbusproject.org/.
30. *Ping computer network administration utility used to test the reachability of a host on an Internet Protocol (IP) network and to measure message round-trip time*. [accessed 2011 March 20]; Wikipedia Entry Available from: http://en.wikipedia.org/wiki/Ping.
31. *Iperf network testing tool that can create TCP and UDP data streams and measure the throughput of network carrying them*. [accessed 2011 March 20]; Wikipedia Entry Available from: http://en.wikipedia.org/wiki/Iperf.
32. Tim Szigeti and Christina Hattingh, *Quality of Service Design Overview*. 2004: Cisco Press. http://www.ciscopress.com/articles/article.asp?p=357102&seqNum=3