Distributed, Heterogeneous Cloud Computing Infrastructure 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 focused 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 measured 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 were used because of their support for scalable simulations. Our preliminary observed data indicates that a heterogeneous cloud infrastructure like the FutureGrid is a 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, and there are studies with systematic evaluation of this emerging information technology infrastructure [10-19], but few publications on collaboration applications in general or leveraging multiple heterogeneous clouds for real-time, geographically distributed, collaboration 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, collaborative sensor-centric applications could be found in the fields of environmental monitoring, security surveillance, or target tracking[20]. For example, one 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 and so there will be a growing demand for scalable support of collaborative sensor-centric applications. One could imagine, for instance, collaborative sensor-centric applications relying on a wide variety of sensor types and a massive number of sensors, deployed globally, streaming in real-time an enormous amount of information at a wide range of time scales to support decision-making for disaster preparation, assistance and recovery.

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, revisit a general methodology we devised for an earlier study [22] on the Amazon Elastic Cloud Computing (EC2) infrastructure which is also being used here and discuss the differences in particular. Secondly, we review the technology in the collaborative sensor-centric grid framework [9] and message broker [4, 7, 8] that we used for this study. Thirdly, we give an overview of the FutureGrid [1, 2], the underlying heterogeneous distributed cloud infrastructure that we used to conduct the experiments. Then, we present 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 referenced.

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

4. COLLABORATIVE SENSOR-CENTRIC GRID FRAMEWORK

We needed some tools to build a sensor grid, deploy and manage sensors, and generate and measure collaborative sensor-centric grid application traffic on distributed clouds of the FutureGrid. Some technologies we had designed and developed for a collaborative sensor-centric grid framework that supports the integration of a sensor-centric grid with collaboration and other grids [9] were used. The framework includes a grid builder tool for building, deploying, discovering and managing grid services and local and remote, distributed sensors.

The Grid Builder tool follows the idea of constructing grids of grids, which assembles a multitude of subgrids into a mission-specific grid application. The Grid Builder 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 developed using the API provided by the framework. A deployed sensor 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 grid is to manage and broker message flows for sensor data and controls.

A sample scenario of a collaborative sensor-centric application using the framework reported in [9] is summarized and illustrated in Figures 1 and 2. There were different types of sensors deployed globally. Each sensor (GPS, Video/Audio, RFID, etc) gathered information from the environment and published it in real-time. A sensor adapter retrieved data from a connected sensor and communicated to the sensor grid. A sensor application subscribed to sensor data of interest, and collaborated in real-time with other applications or users by publishing to the sensor grid the data topics that it had subscribed to.

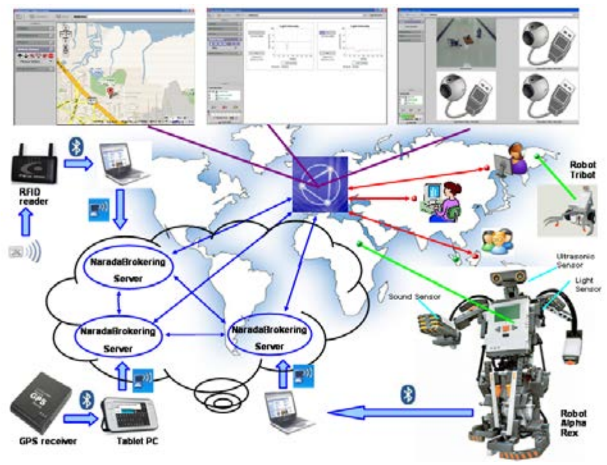


Figure 1. A Lego NXT Mindstorm robot-based collaboaritve sensor-centric application.

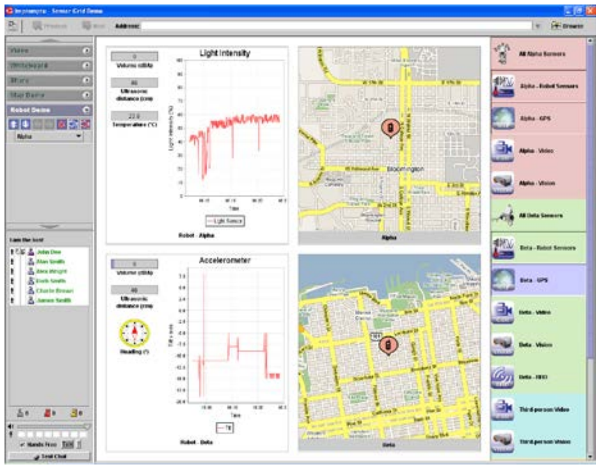


Figure 2. A collaborative sensor-centric application shows and shares eight real-time sensor streams from two NXT robots with their respective GPS locations; one in San Francisco and the other Bloomington, Indiana.

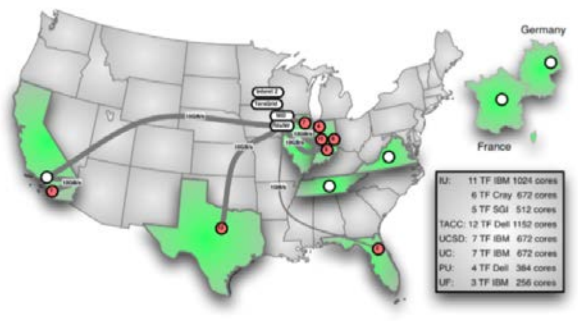
For the study, we developed a sensor-centric application using the framework, and ported the Grid Builder tool to the FutureGrid which enables us to build a sensor grid, deploy sensors and sensor-centric applications to generate, measure and analyze specific application-level performance on distributed 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, University of Chicago, San Diego Supercomputing Center, University of Florida, Purdue University and Texas Advanced Computing Center (TACC) - connected by a high-speed, network which is dedicated except for public link to TACC. 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 and computing resources at the six sites.



IU:	11 TF	IBM	1024 cores
	6 TF	Cray	672 cores
	5 TF	SGI	512 cores
TACC:	12 TF	Dell	1152 cores
UCSD:	7 TF	IBM	672 cores
UC:	7 TF	IBM	672 cores
PU:	4 TF	Dell	384 cores
UF:	3 TF	IBM	256 cores

Figure 3. FutureGrid hardware resources.

FutureGrid includes services accessible to users to run HPC jobs, and many environments including a Eucalyptus and Nimbus Cloud.

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

Nimbus is an open source toolkit that allows one to turn a cluster into an Infrastructure-as-a-Service (IaaS) cloud [29]. Nimbus in FutureGrid allows users to run virtual machines on FutureGrid hardware. A Nimbus account user can easily upload custom-built 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 used up to four clouds in FutureGrid. The clouds we used are named 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). We used the clouds either in pairs or as a group of four clouds. We used m1.xlarge instances in Eucalyptus cloud and 2 cores with 12 GB RAM in Nimbus.

For the purpose of ensuring precision of timing measurements in a distributed environment, we synchronized the cloud instances used in our experiments with a time server in Chicago using the ntpdate command. In a Linux environment, which was the case in our experiments, the NTP algorithm used can usually maintain time to within 10 milliseconds over the public Internet.

For network-level measurement, we used both ping and Iperf, both are commonly used network administration and performance tools. 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 used 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. It is a middleware working as a glue connecting remote parts of a distributed application.

For application-level measurement, we used our collaborative sensor-centric grid framework. In order to investigate scalability issues, it was not practical to deploy real sensors at large scale and instead, we deployed virtual sensors. The collaborative sensor-centric grid framework supports real and/or virtual sensors that implement its sensor API. As an initial study on a multi-point distributed cloud, we deployed virtual GPS sensors only even though we have developed virtual sensors for RFID and WiiMote.

6.1. NETWORK-LEVEL MEASUREMENT

We ran two types of experiments. They were (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 Ping together with Iperf for measuring packet loss and round-trip latency under loaded and unloaded network between all 2-combination of the set of four clouds selected.

Single-Pair of Cloud Instances

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. There were six combinations as shown in the legend of Figure 5.

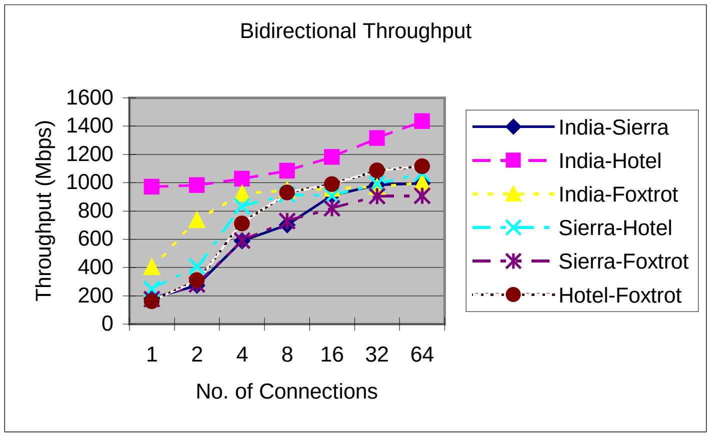


Figure 5. Throughput between 2 clouds

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 found that the total Iperf throughput in FutureGrid was over 800 mbps when we connected any pair of cloud instances on distinct clouds with more than 16 connections in each direction.

As a comparison with commercial cloud Amazon EC2, we did a test between EC2-US and EC2-Europe [22]. The EC2-US and EC2-EU distributed clouds sustained a throughput of 126 mbps at 128 Iperf connections. The maximum sustainable throughput had not been reached. We note that this is not an equitable comparison due to the difference in the state of deployed continental and trans-continental links.

Network Latency and Packet Loss

We used the Ping tool to measure network latency and packet loss between two clouds. We used Iperf with 32 connections to generate heavy traffic of a loaded network. 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 was 0.67% between the India/Hotel pair. The results indicate a highly-reliable FutureGrid network.

Table 1: Inter-cloud Ping loss rate

Instance Pair	Unloaded Loss Rate	Loaded 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%

We measured ping round-trip latency for all six combinations of pairs of clouds. We found the lowest average round-trip latency in a loaded condition of about 18 milliseconds between India and Hotel (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 observed the highest ping round-trip latency in a loaded network condition of 145 milliseconds between Sierra and Foxtrot (see Figure 7). While it is reasonable that the inter-cloud latency between Sierra and Foxtrot is highest due to the longest distance between any two of the four selected clouds, we note that a round-trip latency below 300 milliseconds will meet a requirement for acceptable quality of service for collaboration applications with stringent network requirement like VoIP [30].

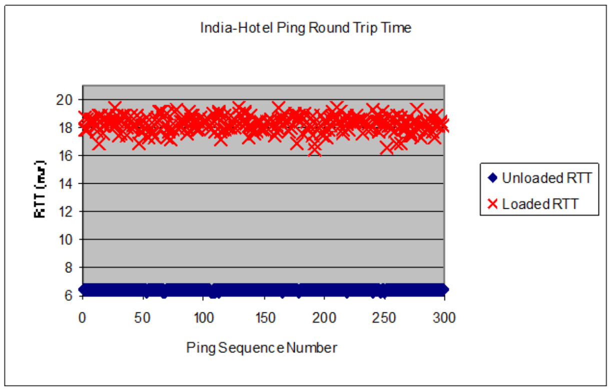


Figure 6. India-Hotel ping round-trip latency

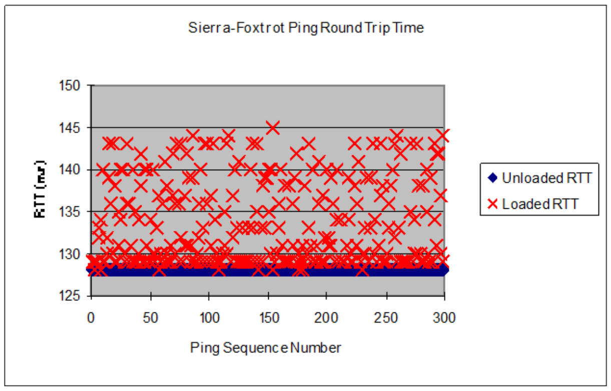


Figure 7. Ping round-trip latency between Sierra and Foxtrot

Overall, our results show that the FutureGrid has a high performance and reliable network.

6.2. MESSAGE-LEVEL MEASUREMENT

In one set of experiments, extensive measurements have been made to evaluate performance, stability and reliability characteristics for an increasingly larger collaboration session by using NaradaBrokering messages. We selected the Foxtrot and Hotel, both running Nimbus environment, for our 2-cloud distributed experiments, with a NaradaBrokering broker running on Foxtrot and simulated multiple meetings with groups of 20 participants 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 a large-scale video session.

An important result of these tests was the fact that as seen in Figure 8, the inter-cloud connection between Foxtrot and Hotel incurred an average latency below 50 milliseconds for the cases of a single meeting with up to 2,400 participants and up to 150 smaller meetings, each with 20 participants. It reflects the fact that multiple smaller meetings balance the work of a NaradaBrokering broker better. Also reflected in Figure 8 is that there was message backlog on the single broker when the total number of participants passed 2,400 in the single meeting and 3,000 in multiple meetings. When there is message backlog on the broker, latency will increase rapidly. Of course NaradaBrokering can support multiple distributed brokers to control a collaboration or sensor network, so limits 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.

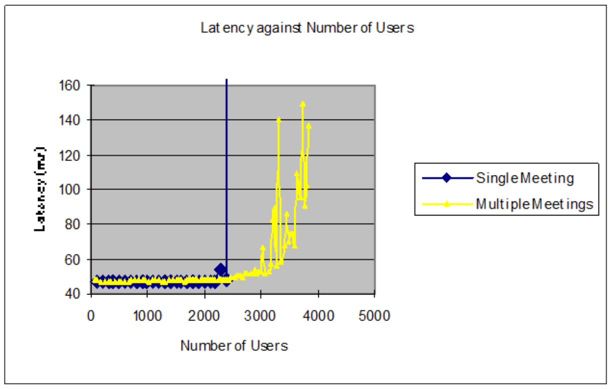


Figure 8. Average latencies of single and multiple video meetings.

Overall, our results show that the FutureGrid has a high performance and reliable network and that 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 in FutureGrid for collaborative sensor-centric applications. While a main objective of our research plan is to study the behavior of real world, highly distributed applications running on different experimental and production distributed cloud systems, and to quantify the CPU, memory and communication requirements of these naturally distributed and highly scalable applications on the underlying distributed cloud architectures as a way to understand how well the current cloud systems support such requirements, we report our initial observations of one specific, highly distributed application, namely the collaborative sensor grid framework [3], running on different combinations of distributed cloud scenarios within the flexible FutureGrid infrastructure as a starting point.

We developed and used 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 streamed information to the sensor grid at a rate of 1 message per second. A sensor-centric grid application consumed all the sensor streams and computed message latency and jitter for a range of deployed sensors.

We first established 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 deployed 100 virtual GPS sensors in an instance in India cloud, we observed the sensors continued running even though both idle CPU and unused RAM were at critically low level, with idle CPU at 7% and unused RAM at 1 GB. Since our focus is primarily on distributed cloud communication characteristics for scalable collaborative real-time sensor-centric applications, we wanted to avoid running into CPU or RAM bottlenecks in our scalability experiment. When we lowered the number of deployed sensors in a single cloud instance to 60, we observed idle CPU at about the 35% level.

We conducted 2 different experiments. They were (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 were deployed in the experiment.

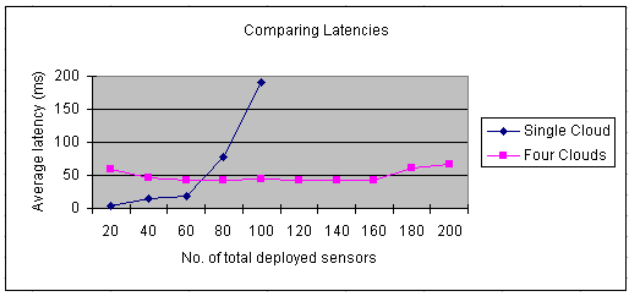


Figure 9. Comparing average latencies of a single cloud and 4-point distributed cloud.

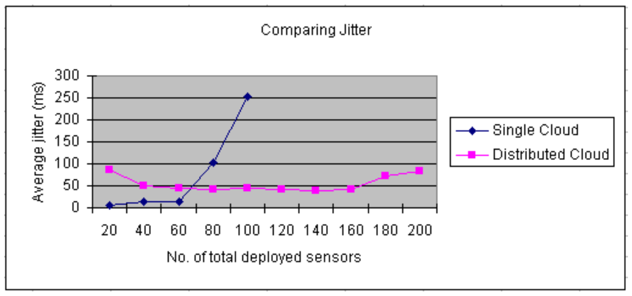


Figure 10. Comparing average jitters of a single cloud and 4-point distributed cloud.

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 for deployment was 100 but the instance was at a critically high CPU and RAM utilization. Such low levels of un-used resources in an instance has 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, even though the number of deployed virtual GPS sensors was 200 only because we chose to deploy 50 sensors in each instance, it had a much lower resource utility, and would 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 started to grow very fast after 60 sensors. At the level of 80 deployed sensors, the average latency was already 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 was relatively constant, sufficiently low and with small variations only when sensor deployment was scaled up from 20 to 200. Thirdly, a similar pattern occurred 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 was low until after deploying 60 sensors. At the level of 80 deployed sensors, the average jitter was already higher than that of the distributed case for 200 sensors.

Overall, distributed cloud scenarios exhibited an encouraging potential to support scalable collaborative sensor-centric applications that have stringent latency, jitter and reliability requirements.

6.4. CONCLUSION

We conducted three types of experiments in FutureGrid to understand its performance characteristics in distributed cloud scenarios to support scalable collaborative sensor-centric applications. We ported the Grid Builder to FutureGrid and developed virtual GPS sensors for manageable scaling of application-level deployed sensors to a large number. We measured FutureGrid distributed cloud characteristics at the network, transport and application levels. Although this study is preliminary, we observed satisfactory performance characteristics at each level. We concluded that a heterogeneous distributed cloud infrastructure like the 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*. 2009 [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. http://grids.ucs.indiana.edu/ptliupages/publications/SensorClouds.pdf.
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. Tim Szigeti and Christina Hattingh, *Quality of Service Design Overview*. 2004: Cisco Press. http://www.ciscopress.com/articles/article.asp?p=357102&seqNum=3