Future Generation Computer Systems 56 (2016) 595–604

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Network health and e-Science in commercial clouds

Ryan Chard, Kris Bubendorfer*, Bryan Ng

School of Engineering and Computer Science, Victoria University of Wellington, New Zealand

HIGHLIGHTS

- We identify and characterize network performance in commercial clouds.
- An overall health system is constructed using tomographic probes to establish and compare an instance's network performance.
- We deploy the health system over a testbed of 100 AWS instances and explore its ability to scale.
- We apply the health system to a medical imaging e-Science application and demonstrate performance benefits.

ARTICLE INFO

Article history:
Received 2 March 2015
Received in revised form
27 May 2015
Accepted 2 June 2015
Available online 23 June 2015

Keywords:
Network tomography
Network health
Cloud computing

ABSTRACT

This paper explores the potential for improving the performance of e-Science applications on commercial clouds through the detailed examination, and characterization, of the underlying cloud network using network tomography. Commercial cloud providers are increasingly offering high performance and GPU-enabled resources that are ideal for many e-Science applications. However, the opacity of the cloud's internal network, while a necessity for elasticity, limits the options for e-Science programmers to build efficient and high performance codes. We introduce health indicators, markers, metrics, and score as part of a network health system that provides a model for describing the overall network *health* of an e-Science application. We then explore the suitability of a range of tomographic techniques to act as health indicators using two testbeds—the second of which spanned one hundred AWS instances. Finally, we evaluate our work using a real-world medical image reconstruction application.

1. Introduction

Cloud computing provides convenient, self-serviceable and cost-effective infrastructure services to users. A key restriction for the adoption of the cloud as an e-Science platform has been the performance of the network connecting provisioned instances. He et al. [1] demonstrate that the cloud's computational resources are capable of executing e-Science applications and even state that applications with low inter-process communication suffer little to no performance degradation when compared to dedicated HPC clusters. In addition, the majority of existing research into the execution of e-Science applications on commercial clouds took place prior to the inclusion of specialized compute instances by commercial cloud providers [2]. However, research suggests workloads with significant degrees of communication are more suited for dedicated HPC infrastructure [3,4].

Although dedicated infrastructure is still the platform of choice for data- and compute-intensive e-Science applications, many research and education projects are finding success with commercial cloud resources [5]. The Magellan initiative [6] explores the cloud model for the purpose of scientific and data-intensive applications. The authors find cloud environments useful for applications that require customizable software stacks and have minimal communication and I/O characteristics. Lifka et al. [7] survey uses of commercial clouds and identify many projects from over 25 scientific domains, ranging from engineering to the arts and humanities, that benefit from the cost-effective computing platform. In another example, the first author of this paper investigated the ability to create a scalable, on-demand medical image reconstruction service for proton computed tomography (pCT) on Amazon Web Services (AWS) [8]. That work compares the cloud service against a dedicated HPC cluster and found that the data distribution phase in the cloud took significantly longer than on the dedicated HPC infrastructure. The network performance of the cloud solution was identified as a key contributor to the performance discrepancies between the cloud and HPC infrastructure solutions.

This paper investigates the use of network tomography in commercial clouds to improve the performance of e-Science

* Corresponding author.
E-mail addresses: ryan@ecs.vuw.ac.nz (R. Chard), kris@ecs.vuw.ac.nz (K. Bubendorfer), bryan.ng@ecs.vuw.ac.nz (B. Ng).

http://dx.doi.org/10.1016/j.future.2015.06.001
0167-739X/© 2015 Elsevier B.V. All rights reserved.

applications. Network tomography is the process of deriving internal network information by sending and monitoring packets as they travel between two end points. End-to-end probes can be used to infer the condition of a network at a fine grain level, identifying bottlenecks, Round-Trip-Time (RTT), and loss [9]. By measuring the delay incurred by a probe as it travels between two end points, congested links that cause long queuing delays can be detected [10]. We use a set of network tomography techniques, referred to as health indicators, to capture the performance of network connections between cloud instances and then apply this information to an e-Science application.

Our work uses network tomography indicators to understand opaquely provisioned cloud networks and infer both the network load and relative proximity of instances. With this information, we introduce and formulate health markers to trigger alerts of degrading network performance, and health metrics and scores to compare the network performance of instances. We employ two testbeds of up to 100 AWS instances to refine the marker trigger points and evaluate the aggregation of health metrics to compute a health score. The pCT project is used as a test case to evaluate the potential benefit of utilizing sub-clusters selected on the basis of health score. In the pCT project test case, instances participating in a pCT image reconstruction are selected based on their locality with the shared file system. This work can be used to improve resource management in many cloud-based services by guiding the allocation of workloads. For example, based on this work we are currently building the health service into a set of on-demand scientific gateways that operate on Amazon EC2. This will enable workloads to be deployed to compute resources based on the network health that is observed between it and the gateway. Utilizing resources with the highest network performance can reduce data transfer times and improve overall application performance.

This paper is organized as follows: we discuss related work in Section 2 and then give an overview of our network health diagnostics and metrics in Section 3. Section 4 introduces our two AWS testbeds and presents our baseline measurements from which our health diagnostics were derived. Section 5 discusses the health diagnostics and markers in depth. Health metrics are then presented and analyzed in Section 6. The health diagnostic system is then applied to a real-world e-Science medical imaging application (pCT) in Section 7, followed by a discussion of our results in Section 8. Finally, we present our conclusions in Section 9.

2. Related work

Many diagnostic tools, such as traceroute, rely on the cooperation of link-layer components [11], yet commercial cloud providers rarely expose network information, rendering such tools ineffective or disabling them entirely. In 2004, Tsang et al. [12] presented a novel tomographic technique, called Network Radar that was based on RTT measurements that did not require the cooperation of the receiver. The work did not require multicast routing capabilities, synchronized clocks, or the capability to capture measurements at both the sending and receiving hosts. Instead, Tsang's work used RTT measurements captured from TCP SYN and SYN–ACK segments to determine the delay variance of a shared network.

Tomographic methods can be broadly classified as either loss- or delay-based. Loss-based methods are focused on identifying congested links within a network by observing packet loss. Duffield et al. [13] and Coates and Nowak [14] presented tomography techniques that were employed to identify lossy links using unicast probes between two end points. However, loss-based mechanisms have become less effective due to the reliability of modern connections, especially with light loads. Coates et al. [15] have shown that thousands of probes must be measured before a one percent loss rate can significantly effect the end-to-end performance of a link. Coates et al. also presented a novel probing scheme, called Sandwich probing. This probing scheme was designed to measure path delay without the requirement of a synchronized clock. Sandwich probing works by sending and recording the delay of two smaller packets which are separated by a single large packet. The delay caused by the large packet can be captured by measuring the difference in the RTT of the smaller packets.

More recently, work to explore a commercial cloud network was conducted by Battré et al. [16]. Their work investigated methods to infer the network topology within opaque cloud infrastructure. The authors used a testbed of 64 Xen VMs to explore and evaluate both loss- and delay-based, tomography techniques. The results indicated loss to be a less effective measure, and found, when using the Robinson–Foulds metric, that RTT outperformed Sandwich probing. Our work in this paper directly extends Battré et al.'s research by evaluating a number of additional network health indicators and investigating their performance with a real-world e-Science application.

Another recent example of research that utilizes network information within a cloud is Cloud MPI (CMPI). CMPI [17] was a network-aware implementation of MPI designed for cloud environments. The research investigated optimizing MPI's collective communication algorithms to utilize network performance information. The authors found that Amazon EC2 had significant network performance unevenness, where performance was not symmetrical between virtual machine pairs. The work used simplified latency and bandwidth matrices to evaluate network performance, and then used the derived information to optimize the Broadcast and Reduce, and Gather and Scatter operations. The work resulted in optimization of between 13% and 38%, compared to that of MPICH2 [18]. CMPI exemplifies the potential opportunities that are available within the cloud when applying inferred network information in the deployment of distributed computations.

3. High level view of our work

We have adopted and extended the concept of health metrics from the second author's prior work [19,20], in which the overall health of a service container was characterized in order to make service deployment decisions. In this paper we apply the health metric concept to the commercial cloud domain and extend the concept to employ network tomography—to infer the properties and characteristics of provisioned cloud instances. Calculating health metrics based on the network performance that an instance is experiencing provides a mechanism to evaluate and compare instances and inform decisions regarding cloud workload deployment.

To evaluate the network health between two instances we devised a set of health indicators. In this context, a health indicator is a tomographic method of measuring the network performance between two instances. Although Amazon has an integrated health service for EC2 instances, its capabilities are limited to identifying instances becoming unresponsive. We utilize a range of health indicators to thoroughly observe the network and identify performance properties. The health indicators use multiple network protocols and customizable attributes, such as varying payload size and probe frequency. This technique of deriving health indicators through probes can be traced back to [10,21] and remains a preferred method to monitor the state of the network [22,23].

To use the information gathered from health indicators, we formulate a set of health markers. A health marker is a binary, lightweight yet easily computable diagnostic, used to detect a significant change in network performance across probing cycles and trigger an alert. We have formulated a marker for each of the health indicators. Each marker is used to quickly establish the degree to which the network performance between two instances changes over a period of time and prompt the recalculation of the overall health score for an instance when necessary.

A health metric is a normalized aggregation of health indicator measurements. The set of health metrics is weighted and combined to compute an overall health score for a target instance, providing a high-level diagnostic on the network performance of the target instance with respect to its peers. A health score is a single value used to represent the overall health of an instance and allow instances to be compared to one another. When computed, a health score gives a perspective of the load the target instance, or the network connecting the two instances, is experiencing and can facilitate the selection of healthy clusters.

We combine each of these concepts into a single tool, known as the health diagnostic system. Fig. 1 depicts an overview of the health diagnostic system. The health indicators collect network performance measurements from other instances, and feed information to the health markers, which can trigger the recomputation of the overall health score if sufficient variation is detected. In addition to this, an overall health score is periodically computed for a target instance by combining health metric values, enabling their comparison and facilitating the selection of healthy clusters.

4. Testbeds and cloud performance baselines

We have provisioned and monitored the network health of two testbeds of AWS instances to investigate the properties of commercial cloud networks, refine the triggering points of health markers, and evaluate health metric aggregation weights.

In our earlier work [24], we provisioned a small set of six AWS instances which we now refer to in this extended paper as “Testbed I”. Testbed I was used to observe baseline AWS cloud performance characteristics, evaluate various tomographic techniques, and develop the health diagnostic system. In addition to this, we have since implemented an entirely new software harness that enables us to experiment with much larger numbers of AWS instances. Testbed II consists of one hundred AWS instances and represents a large-scale environment from which we can identify additional performance characteristics and evaluate the health diagnostic system as it scales across many instances. The following describes the testbeds, data collection methods, and the performance characteristics we have established.

4.1. Testbed I

Testbed I initially consisted of three t1.micro and three m3.medium type instances run over one week[1] and was intended to determine baseline AWS cloud performance characteristics. A series of tests involving various tomographic probes was conducted using the testbed. The testbed utilizes two availability zones in order to document the perceived effect of data traversing the network. The tests were run in a round-robin process from each instance, where every five minutes each instance would communicate with every other instance in the testbed. The probing schedules were deliberately offset in an effort to reduce the interference caused by multiple hosts concurrently recording measurements with a specific host. Testbed I was later scaled out to include 20 and then later 50 instances, however the software harness used to collect network information suffered from scaling issues above this number of instances, leading to the development of Testbed II, see Section 4.2.

[1] The tests were run over a period of one week to examine how recurring events, such as how the time of day across the globe, influenced the network health measurements.

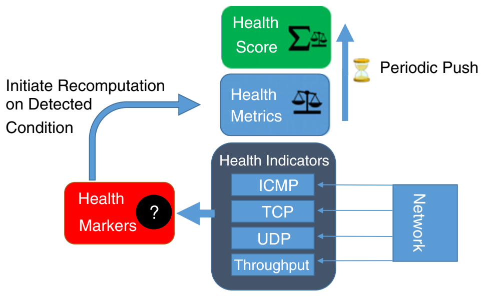


Fig. 1. An overview of the health diagnostic system.

ICMP echo requests are a typical method used to measure the latency between hosts. The jitter, or variance in latency, within a link can be established by collecting RTT probes over a sufficient period of time. In order to establish the influence packet size has on network characteristics, we have used echo requests with different payload sizes.

Figs. 2(a)–(c) show the variation we measured in ICMP RTT probes over different periods of time. These results show that the distribution of probes contains a significant number of high RTT values, over each period of time—demonstrating the high degree of network RTT volatility experienced by a regular AWS instance.

Over longer measurement intervals, the variation in ICMP RTT occurs over sufficiently long periods of time to have an impact on the performance of an application—or in other words, the performance of a link can deteriorate (or improve) for periods of hours, rather than in short intermittent bursts. One example of this is shown in Fig. 3, where the average hourly RTT between two instances, for various ICMP packet sizes, is collected and displayed for an individual day. Over this period, the 4096 and 512 byte ICMP packets have higher RTTs at the beginning of the day and then gradually improve. The performance of the 64 byte ICMP packets is reasonably consistent over the same period. While having different shapes, similar trends are observable in many other periods, and across many other pairs of instances that we examined. The fact that these situations occur for meaningful lengths of time, reinforces our position that monitoring the changes in network performance and utilizing it for scheduling and deployment within the cloud is a worthwhile strategy.

In order to further understand the network performance between various instance types and across availability zones we conducted a series of bandwidth measurements. The measurements involved transferring as much data as possible between pairs of instances within a ten second time span. The results found significant bursting characteristics for TCP transfers. Fig. 4 shows the average throughput over the ten second time span between various instance types, where links across availability zones are denoted by-*AZ*. The figure depicts the presence of substantial throttling and probable profiling of data transfer within the AWS network. The throttling differs between instance types, where t1.micro instances achieve a relatively high throughput of approximately 200 Mb/s for the first four seconds of a transfer before being throttled to approximately 80 Mb/s. Similarly, the m3.medium instances achieved an initial throughput of almost 1000 Mb/s for approximately one second before being throttled to slightly over 200 Mb/s. From this data it appears that the instances are granted a burst throughput rate for approximately the first 1000 Mb of data being transferred. Additional tests with cluster compute, cg1.4 $\times$ large, instances demonstrated a sustained throughput of approximately 8000 Mb/s within an availability zone, and a sustained 2000 Mb/s connection across zones. The cluster compute instance types did not appear to be subjected to the same throttling techniques and were most likely achieving their maximum available throughput.

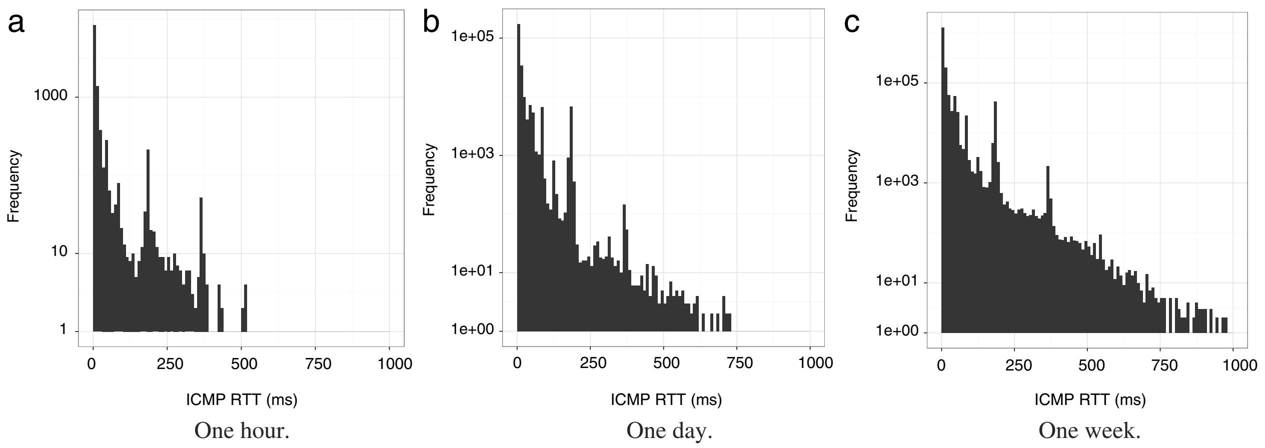


Fig. 2. The frequency (log) of ICMP packet RTTs over different periods of time.

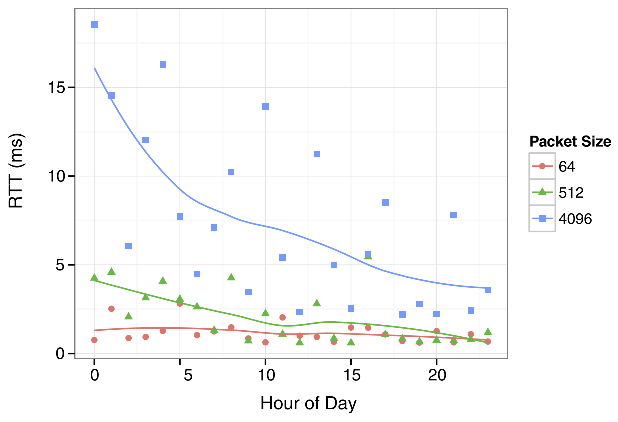


Fig. 3. The hourly average RTT for different packet sizes between two hosts.

These findings demonstrate two interesting properties that support our goal of utilizing network information for e-Science application deployment. Firstly, network performance exhibited by instances is significantly volatile and can easily change in excess of 50% over a short period of time, affecting the performance of communication between instances. The performance fluctuations also persist for sufficiently long periods of time to make action regarding them meaningful. If the network variance was only observable for a short period of time (for example, on the order of seconds), the volatility of the network would render any deployment optimizations ineffective as the network performance could change many times during execution. However, our results indicate that this is not the case. In a number of examples, the degraded performance of an instance can be observed from multiple hosts for periods of hours.

4.2. Testbed II

e-Science applications often require large numbers of compute nodes and may be adversely influenced by invasive probing schemes. For this reason, we have developed a second testbed, referred to as "Testbed II", in order to better evaluate performance characteristics as well as examine the health diagnostic system as it scales over large sets of instances. Testbed II consists of 100 m3.medium instances distributed across three availability zones in the US-East region. Due to gradually scaling the testbed size, more instances have been acquired in the first availability zone than the others, with 35 instances on us-east-1a, 33 instances on us-east-1b, and 32 instances on us-east-1c.

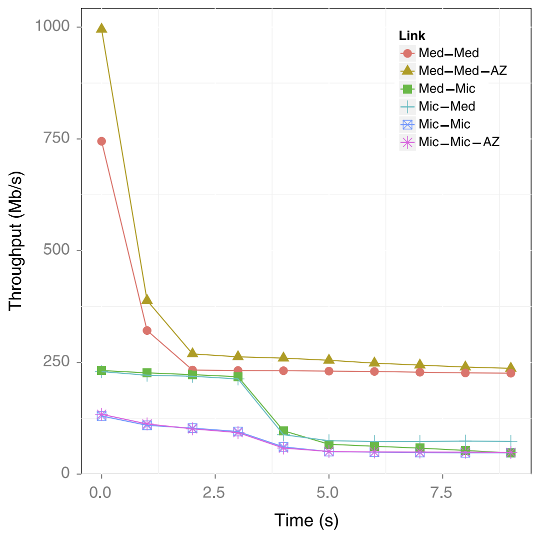


Fig. 4. The average throughput between medium and micro instances within and across an availability zone.

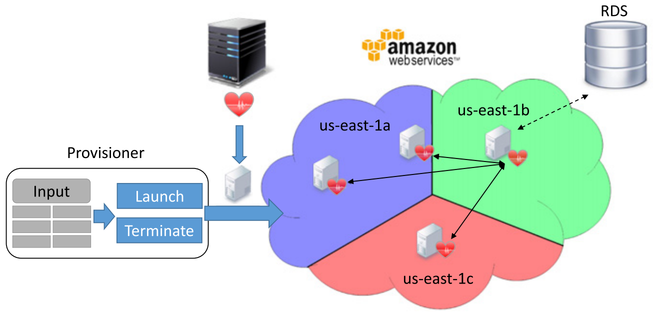


Fig. 5. An overview of the method used to deploy and monitor the health of the network.

To improve the reliability of launching and monitoring many nodes concurrently, a general provisioning system has been constructed. Fig. 5 represents an overview of the provisioning system used to deploy, monitor, and collect data on the network performance in Testbed II. The provisioning system creates spot requests for the desired instance type and attempts to evenly distribute them across the specified availability zones. AWS presents two methods for establishing provisioned instances: through the use of a predefined Amazon Machine Image (AMI), or by dynamically contextualizing the instance. We opt to dynamically contextualize each instance using CloudInit as it provides additional flexibility in the deployment of the health system. During contextualization the health diagnostic system is downloaded to the instance and deployed. Once the health system begins operating on each instance, it contacts the shared Relational Database Service (RDS) instance, to publish its address for others to monitor. The health system then begins periodically communicating with other health systems that are in the testbed. The results that are gathered from communicating with other instances are reported to a shared RDS instance for analysis.

It takes approximately twelve seconds for an instance to probe another instance and collect measurements for each of the health indicators, primarily due to throughput measurements which are taken at one second intervals over a ten second period. Meaning, to probe each of the 99 other instances in Testbed II requires almost 20 min. To reduce the risk of collisions, which could potentially distort results, we have reduced the rate at which instances are probed in Testbed II. We employ a 60 min probing rate, where each instance begins collecting health data for every other instance once an hour. To further minimize collisions, the shared database determines the order in which probes are made between instances. The health diagnostic system probes hourly in a round-robin fashion, beginning with instances that have joined the testbed since itself (as recorded in the database), before iterating through the remaining instances.

5. Network health

Deploying the health indicators over the testbeds for a prolonged period of time enables health markers to be defined and used. A diagnostic health marker is used to identify symptoms of an instance with degrading network performance. Each marker is an easily computable binary test to recognize performance changes and prompt the reassessment of the overall health scores for instances in the environment.

5.1. Health indicators

The performance of an instance can vary over time due to the network load both itself and surrounding resources are experiencing. A set of health indicators have been selected and are later evaluated with respect to their ability to reliably observe performance fluctuations, and influence the health score of a target instance. Although Amazon provides a health service, its role is to identify when instances become unresponsive. Our health indicators are capable of monitoring fine-grained latency and throughput variations as well as capturing timeout occurrences.

The delay-based tomographic indicators utilize ICMP, UDP and TCP, with the goal of establishing load by observing variations in RTT and measuring jitter in the network. A range of packet sizes and varying intervals between sending packets have been used to identify the effect of queuing in the network.

Spot instances are often used to reduce the cost of application deployment on AWS and are prone to becoming unresponsive. When a bid for a spot instance is exceeded, the resource is reallocated to another user. For this reason, timeouts have been incorporated as indicators as it is critical to identify unresponsive instances.

Throughput indicators and Sandwich probing, first presented by Coates et al. [15], have also been employed. Sandwich probing measures the delay incurred by a small packet traversing a network when preceded by a large packet. This is accomplished by sending two small packets separated by a time d with a larger packet immediately preceding the second packet. By measuring the time between the arrival of the two packets, d', the difference between d and d' can be obtained to infer the delay caused by the large packet.

Consideration to Sandwich probing packet sizes is required when implementing the probing scheme. The maximum transmission unit (MTU) for t1.micro and m3.medium instances is 1500 bytes. However, AWS supports jumbo frames for cluster compute instances types, allowing packet sizes of up to 9001 bytes to be used when probing cg1.4 $\times$ large instances. We adapt the Sandwich probing large packet size to reflect the MTU, while maintaining a small packet size of 64 bytes regardless of instance type.

5.2. Health markers

Health markers have been employed to set lightweight diagnostic values which are used to alert when degrading network performance is identified. A health marker is a binary indicator (trigger/no-trigger) of the presence of a problematic state in the network (analogous to tumor markers in medicine). Five markers have been selected and implemented in order to alert the diagnostic system of significant variations in network performance. These markers each represent a quantifiable threshold value related to a specific goal gathered from a protocol between two hosts. The following describes the implementation details of each health marker. The decisions regarding the point at which a marker is trigged, and an alert is raised, have been established through experimental analysis in order to identify targets which are only triggered when a significant degree of volatility or degradation is detected in the network connection.

- **Timeout** A timeout marker is used to raise a notification when an instance becomes unresponsive. This health marker requires consensus from more than one indicator on the unresponsiveness, or lack of response within five seconds, of another instance.
- **ICMP** An ICMP health marker combines each of the three packet sized RTT measurements and compares them with the standard deviation from the previous round of probes. The marker is triggered when 20% of the current round's measurements exceed the standard deviation of the previous round.
- **UDP** The UDP based health marker employs UDP RTT and Sandwich probing. The RTT is gathered from 1024 and 64 byte probes and the standard deviation from the previous round is used to infer degradation. Sandwich probing measurements are used to trigger a notification when a 50% increase in delay is observed, implying the effect of queuing in the network is significant.
- **TCP** The TCP health marker monitors the time required to establish a TCP connection between two instances as well as the RTT achieved through the connection. The standard deviation from the previous round is used to establish threshold times, which when exceeded triggers the marker to raise a diagnostic notification.
- **Throughput** The available throughput between two instances is measured over a ten second period. A health marker has been established to identify when the total throughput over the ten second period drops below a longer term threshold.

In the next section we introduce the concept of health metrics, which are concrete measures used to inform selection of instances for improved performance. Health markers are not used in computing the health metrics but both utilize the same information (health indicators) collected by the tomographic probes, as shown in Fig. 1.

6. Health metrics

Health metrics provide a normalized mechanism to evaluate instances against one another. A health metric has been formulated

Table 1
Forward step analysis of health metrics for two sets of trace data.

Step	Health metric	Data Set I		Data Set II	
		April 2014		January 2015	
		AIC	p-value	AIC	p-value
4	$H - TP_{ij}$	216.311	0.04126	106.686	0.02454
3	$H - ICMP_{ij}$	98.227	0.00587	99.114	0.01399
2	$H - UDP_{ij}$	90.221	0.00752	103.098	0.00605
1	$H - TCP_{ij}$	106.842	0.00609	118.212	0.00465

for each individual network protocol utilized by the indicators and typically aggregates the information collected from multiple indicators into a single value. Each health metric computed for an instance is normalized with the other instances a host is monitoring, providing a relative health for each network protocol.

An overall health score is computed by combining each of the individual health metrics through weighted aggregation. The weights associated with each health metric have been selected through a statistical evaluation in order to give each marker meaningful influence on the overall health score. The overall health score, $H - All$, of an instance gives the host a mechanism to directly compare each instance in the environment and select healthy nodes to perform workloads.

The ICMP health metric (denoted by $H - ICMP_{ij}$) prioritizes packet size from largest to smallest, with heavier weights given to the larger payload measurements. The health metric computes the ICMP score by averaging the RTT of each packet size, and normalizes it against the average RTT of its respective size for each instance the host has probed during a round. Eq. (1) shows the calculation of the ICMP health score where $S = \{64, 512, 4096\}$ is the set of packet sizes being used as probes and P_{ij_s} represents a set of probes from host i to j of size s where $s \in S$. A_{i_s} denotes the set of the average ICMP probe measurements sent by host i, of size s, to every other host in the environment. Finally a weight (denoted by ω_s) is associated with each packet size, such that larger packets are given more influence than smaller packets.

$$H - ICMP_{ij} = \sum_{s \in S} \frac{\text{avg}(P_{ij_s}) - \min(A_{i_s})}{\max(A_{i_s}) - \min(A_{i_s})} \times \omega_s. \quad (1)$$

The UDP health metric (denoted by $H - UDP_{ij}$) normalizes the average RTT values and the average delay measured from Sandwich probing to evaluate the link between instances i and j. The two RTT values and the Sandwich probing delay are measured by the UDP health indicator and are combined with weights giving more influence to larger packets. The Sandwich delay is given an equal weighting to the RTT measurements to give influence to the delaying properties of the network.

The TCP health metric (denoted by $H - TCP_{ij}$) is computed in a similar fashion and normalizes the average connection time and RTT through the connection during a measurement period. Each value is then combined with equal weighting to provide a relative health score of the connection ij.

The throughput health metric (denoted by $H - TP_{ij}$) incorporates the total amount of data transferred over the ten second measurement period with the variance in throughput during each one second interval. These values are individually normalized and then combined with equal weighting to construct the throughput health score.

We use two data sets, Data Set I and Data Set II, which have been collected from Testbed I and Testbed II, respectively, to analyze the role and influence of each health metric. An associated weight is required for each health metric in order to aggregate the metrics to compute an overall health score. For each health metric we apply the variable selection technique to a linear regression model over the complete data set and rank the metrics with the strongest

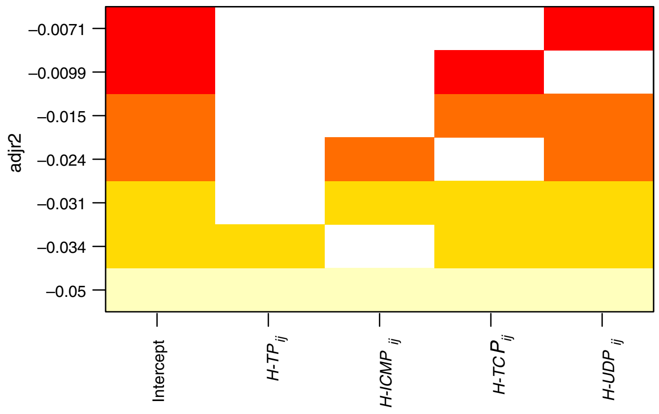


Fig. 6. The heat map depicting the merit of the health metrics. The red bars indicate the lowest error range (higher merit) while beige bars indicate the highest error range (lower merit). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

influence on the aggregate health score [25,26]. Based on trace data collected from AWS from 29 April 2014 (10:45:16) to 6 May 2014 (14:24:27) we use forward step analysis on Eq. (2) to rank the influence of the individual health metrics on the overall health score. Starting from Step 4 (Column Data Set I) in Table 1, we add one health metric per step and calculate the corresponding Akaike Information Criteria (AIC) measure and p-values.

$$H - ALL_{ij} \sim H - TP_{ij} + H - ICMP_{ij} \\ + H - UDP_{ij} + H - TCP_{ij}. \quad (2)$$

Lower values of the AIC signify stronger influence of the health metric in the regression while the p-values indicate confidence in health metric influence on the health score. In Table 1 for example, using the sole metric $H - TCP_{ij}$, the step analysis at Step 1 yields an AIC of 106.842 and corresponding p-value of 0.00609, adding the $H - UDP_{ij}$ metric reduces the AIC to 90.221 but increases the resulting p-value to 0.00752. The reduced confidence in the regression at Step 2 is expected due to collinearities in metrics $H - TCP_{ij}$ and $H - UDP_{ij}$. We reason that the predictive power of the throughput metric is captured by both $H - TCP_{ij}$ and $H - UDP_{ij}$ thus weakening the $H - TCP_{ij}$ metric in the forward step analysis. Upon terminating the forward step analysis, we select all four health metrics because the p-value is greater than 0.05, which indicates that the health indicators are significant and hence all four health metrics influence the network performance prediction.

Another set of trace data, referred to as "Data Set II", was collected from AWS from 28 January 2015 (16:59:15) to 28 January 2015 (22:09:31). The forward variable selection analysis on this newly collected data is presented in Table 1. The observed AIC trend is consistent with the results from Data Set I and therefore reinforces our earlier findings on the merits of the four selected health metrics on predicting the health score. For both data sets, the reverse step (elimination) analysis yields final AIC values identical to the forward analysis.

The adjusted-R^2 (adjr2) measure for different subsets of network health metrics are shown in Fig. 6. The adjr2 compares the errors in the regression that is adjusted to the different numbers of health metrics. For example, in the plot of Fig. 6, using a single health metric $H - TP_{ij}$ yields an adjr2 value of −0.034. This value is calculated by taking into account the fact that only a single health metric is used. For the case of using three health metrics ($H - TP_{ij}$, $H - UDP_{ij}$ and $H - TCP_{ij}$) a smaller adjr2 value of −0.0099 is obtained, this value is a fair comparison with −0.034 because it has been adjusted for three health metrics. The minimum value of the adjusted adjr2 is −0.0071 when all four health metrics are used and this is marked with red bars denoting the lowest

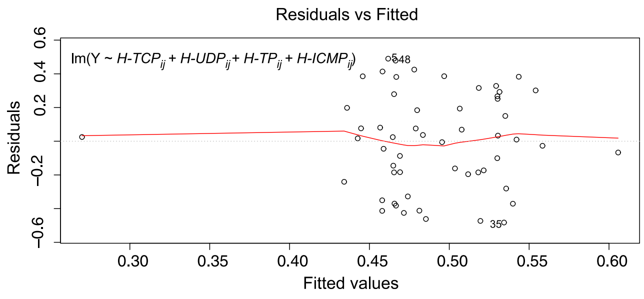


Fig. 7. The residuals for linear regression.

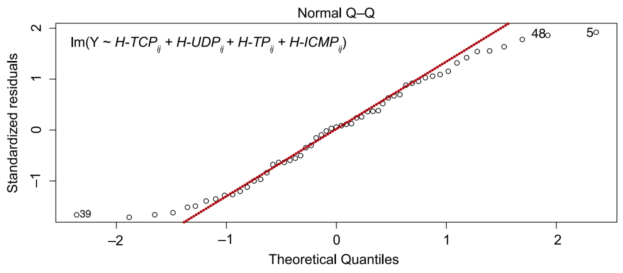


Fig. 8. The Q–Q plot of the standardized residuals from the linear regression (y-axis) vs. theoretical (Normal) quantiles (x-axis).

error range. The maximum value of adjr2 is −0.050 and it occurs when a single metric is used ($H - TP_{ij}$). This is marked with beige bars (highest error range) in Fig. 6. These observations from the heat map suggests that all the selected health metrics have merit in predicting network health, some more than others, and this conclusion agrees with the conclusion drawn from the forward step analysis.

6.1. Health metric diagnostics

The health metric selection diagnostics are used after performing variable selection to check if the linear regression and its assumptions are consistent with the observed data. The basic indicator for the diagnostic is the residual. A residual or fitting error, is an observable estimate of the statistical error. If the linear regression does not give a set of residuals that appear to be reasonable, then the choice of the health metrics (one or more) may be called into doubt or perhaps linear regression was not the best model to fit the data. In this paper, we adopt visual inspection to analyze the merit of the health metric used to predict network health.

The evenly distributed residuals (with respect to Residuals = 0) in the plot of Fig. 7 shows that the residuals (errors) and the fitted values of the health metrics are uncorrelated. This validates the choice of health metrics as appropriate measures for predicting network performance.

The quantile–quantile or Q–Q plot is a graphical diagnostic used to check the validity of a distributional assumption [27] for the linear regression model used in Eq. (2). In the linear regression, the residuals are assumed to have a normal distribution and thus the Q–Q plot for the standardized residuals will be close to a straight if this assumption is valid. Fig. 8 shows that the distribution of the model residuals are balanced with respect to both the upper and lower quantiles. Moreover, the curve tracks the straight line quite well for theoretical quantiles between −1.5 and +1.5 which is what is expected of normally distributed residuals.

We have shown the relative significance of each health metric and their respective statistical interpretations, however, this analysis must be framed within a networking perspective. In the following section, we introduce an aggregate measure called the health score, which summarizes the four health metrics. We discuss the effect of each health metric on the health score and how the health score aids decision making in selecting cloud instances.

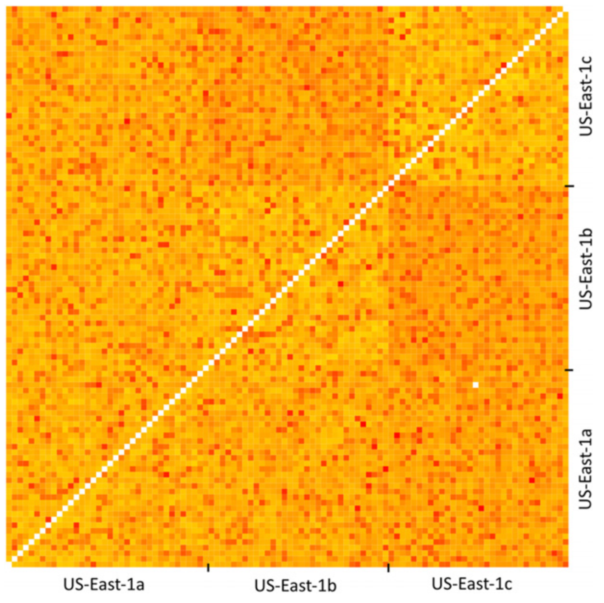


Fig. 9. A heat map of the health scores computed between the 100 instances monitored in Data Set II. Each square represents the health score computed between two instances, where red indicates a lower, or less healthy score, and lighter values depict healthier connections. The cost of communicating across availability zones is depicted by noticeable variations in color between US-East-1b and US-East-1c instances. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

6.2. Health score

The timeout of an instance is critical to identify and nullify additional health metrics. Therefore, the overall health metric incorporates timeout values by computing a health score of zero, or $H - ALL_{ij} = 0$. However, if no timeouts are identified, and an instance is considered operational, the overall health metric is computed as seen in Eq. (3). From the variable selection analysis, we noted that the marker $H - TP_{ij}$ is weaker than the remaining markers. Moreover, throughput is one of the key metrics in service level agreements in AWS specifications. Thus, the weight of 0.4 is chosen for marker $H - TP_{ij}$ to prioritize throughput over latency. Each of the individual metrics are normalized against the other connections in the system and aggregated together with weights.

$$\begin{aligned} H - ALL_{ij} &= 0.4 \times H - TP_{ij} + 0.2 \times H - ICMP_{ij} \\ &\quad + 0.2 \times H - UDP_{ij} + 0.2 \times H - TCP_{ij}. \end{aligned} \tag{3}$$

The health score has been computed between each instance in Testbed II and is represented as a heat map in Fig. 9. The instances are ordered by availability zone, with the first 35 residing in us-east-1a, 33 in us-east-1b, and 32 in us-east-1c. While the heat map demonstrates the similarities between availability zones, it clearly depicts the boundaries of the us-east-1b and us-east-1c availability zones, showing the deprecation in network performance across them. The heat map also clearly demonstrates the volatility in network performance that can be experienced within a single availability zone.

The collection and computation of health scores between each instance in the environment requires a significant amount of time. When employing the health system for an e-Science application, the profile of the application should be considered. For example, the pCT application utilizes a centralized file system to distributes workloads, which is pivotal to the performance of the data distribution phase of a reconstruction, while little communication between instances is necessary. Due to the data-intensive nature of the pCT workload, it is essential to base execution in proximity to the data, whereas the locality, or health, of instances to one another does not impact performance. Therefore frequent monitoring and computation of health scores between instances is unnecessary and could negatively influence the overall performance of the application.

7. Proton computed tomography

The pCT project is a real-world e-Science scenario that we have employed to evaluate the effectiveness of the health information that can be inferred from a commercial cloud. pCT is a medical imaging modality and was developed to acquire high accuracy images for proton therapy applications [28]. The pCT modality is based on tracking the change in trajectory of protons as they pass through a target. Because protons passing through different mediums travel in non-straight paths, optimization techniques that are often applied to other imaging modalities cannot be applied to reduce data. This causes pCT image reconstruction to be both data- and compute-intensive, and can require up to 100 GB of data, or two billion proton histories, to be processed. A pCT reconstruction is primarily comprised of four stages, data distribution, computing cuts and margins, most likely path (MLP) calculation and an iterative linear solver.

Karonis et al. [29] have developed parallel MPI codes that enable large, two billion proton history, images to be reconstructed within ten minutes on a dedicated HPC cluster. A detailed explanation of each pCT reconstruction phase is also presented their work. The first author's previous work re-purposed these codes to construct a scalable, on-demand, pCT reconstruction service that operates over AWS [8]. The cloud-based pCT reconstruction solution leverages a shared Gluster [30] file system to distribute the data to each working process. The previous work found the data distribution phase of pCT reconstruction to take significantly longer on the cloud service when compared to dedicated HPC infrastructure. For small 131 million history reconstruction over 20 instances, the data distribution phase took 25.8% of the total execution time. When deployed over 120 instances, the data distribution phase accounted for 38% of the total execution time for a two billion history reconstruction. The data distribution phase scaled accordingly to the number of instances being used [8].

To explore the potential of network inference and our health metrics, in this paper we have deployed the cloud-based pCT reconstruction software and a Gluster file system in conjunction with our network health diagnostic system. The health-aware pCT reconstruction experiment was deployed over a new testbed consisting of fifteen GPU enhanced cluster compute instances, known as the cg1.4 × large instance type. The instances were provisioned from two separate availability zones within the US-East region. The pCT codes were used to reconstruct a 131 million proton history image over eight instances, utilizing two processes per instance to match each available GPU.

Three clusters of instances have been selected to investigate the usefulness of the health information. These clusters consist of instances with the highest, lowest and a random set of health scores, and have been used to compute pCT reconstructions. Because the pCT codes rely on a centralized, shared, file system, the health metrics have been calculated with respect to the instance maintaining the GlusterFS brick. Fig. 10 depicts the inferred distance of each instance from the shared file system, using health scores to weight edge lengths. The topological distribution of each instance is not considered in this figure.

The average result of multiple pCT reconstructions over each group has been computed and represented in Fig. 11. The figure demonstrates a distinct advantage to leveraging the proximity of instances when deploying the application. Improvements in performance can be seen during the data distribution phase and during execution of the linear solver. The inclusion of health information resulted in the data distribution phase taking, on average less than half as long as that of the least healthy group of instances. Similarly, the cuts and margins phase has been reduced as it requires data to be communicated between the instances, whereas the time required to compute the MLP is consistent between clusters as little data is transferred. Due to the small number of instances being used to pCT reconstruct the 131 million history data set, the computationally intensive linear solver accounts for the majority of the reconstruction time. Over eight nodes, the linear solver accounts for between 80% and 85% of the reconstruction time. Where as over the larger cluster sizes utilized in our prior work, the linear solver accounted for less than 55% of the execution.

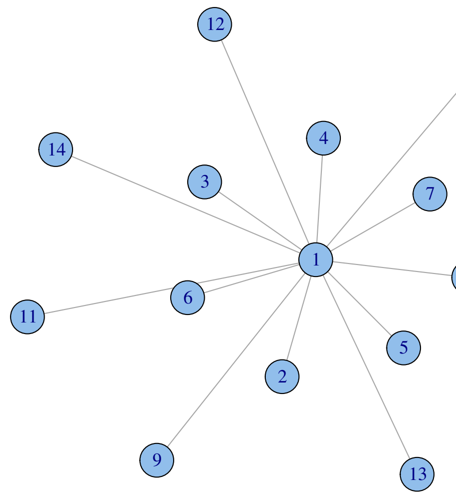


Fig. 10. The proximity of instances used for pCT reconstruction where edge length is determined by health score.

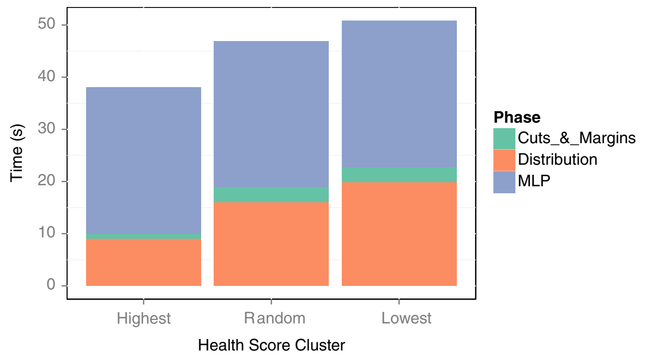


Fig. 11. The average time required for phases of pCT reconstruction by various health score clusters.

8. Discussion

The performance observed from the Testbed I and Testbed II has demonstrated that each of the health indicators is capable of identifying network fluctuations. The volatile nature exhibited by the network connecting instances provoked significant variations in all of the health indicators that we measured. Deploying the health system over a large set of 100 instances in Testbed II has highlighted the volatility in cloud network performance. The results, discussed in Section 6.2, show unexpectedly high levels of variation within an availability zone. These results support the findings of Gong et al. [17], who describe the unevenness and unsymmetrical nature of network performance they encountered when developing CMPI.

An evaluation of the variance observed by the health indicator measurements were used to rank the influence of health metrics and contributed to the weights used when computing the overall health score. Counterintuitively, the throughput health metric was identified as less influential than other metrics in the forward step regression analysis. However, when considering the cluster compute environment, throughput variations were less significant than regular instance tests. In order to reflect the priorities of data-intensive e-Science applications, throughput was assigned a larger weighting than other indicators when computing health metrics.

Diagnostic health markers have been established from these variations in order to prompt reassessment of the overall network health. Due to the noisy nature of the regular instance test bed, an initial set of threshold-based markers frequently responded to network fluctuations. However, the cluster compute 10-Gigabit Ethernet connection is far less volatile and resulted in fewer notifications being raised by the health markers. In order to operate more effectively, the health markers need to adapt to the environment in which they are executing. Lower tolerances and the inclusion of more historic data are needed to fine tune the markers over various platforms.

The health metrics provide an effective method to compare and evaluate instances against one another. The metrics operate successfully in all of the monitored environments, and have demonstrated the ability to improve the data distribution performance of pCT reconstructions. Although the difference in cluster compute instance health scores is most apparent between availability zones, the health metrics were accurate enough to consistently identify low performance cluster compute instances within a single zone as well.

Although the advantages of using the health information are significant when considering small scale executions of the pCT application, the results are unlikely to scale linearly with the application performance for larger reconstructions. The previous work to initially construct the cloud-based pCT reconstruction service identified additional overheads, in part responsible for the performance deterioration as the application scaled [8]. Thus, we do not believe the improvement to two billion history reconstructions will be as significant as the 131 million history reconstructions that have been examined. Further investigation is required to establish the effect of health information on the application as it scales.

The deployment and evaluation of Testbed II has also identified limitations of the health diagnostic system to scaling. The risk of interference and the time required to establish health scores for each instance grew sharply as more instances joined the testbed. Establishing health scores for the 100 nodes in Testbed II required approximately 20 min. This restricts the ability for the health system to identify network variations in a timely manner, and also limits the viability of applying it to larger e-Science applications.

The throughput indicators account for the majority of the time required to establish health, and are most likely to negatively impact the performance of an instance. However, we have established that the throughput metrics have little significance on the overall health score an instance is given, meaning we can either reduce, or eliminate the throughput measurements. We also aim to investigate the ability to apply application profiles to the health service, such that health measurements are only collected between hosts of significance. For example, in the case of the pCT application, the health service was only applied between worker nodes and the shared file system, as the health between workers has little effect on overall application performance due to low coupling.

9. Conclusion

Commercial clouds are opaque and limit the ability to exploit data locality. Our work aimed to improve the viability of performing compute- and data-intensive scientific research over commercial cloud resources. To this end, we have investigated and evaluated the ability of various tomographic tools to infer network properties and establish the performance an instance is currently experiencing. Our work has identified a number of properties of commercial clouds, such as the variability in network performance, and the sustained nature of performance fluctuations that an instance can experience. A health system has been constructed to monitor the network health between a set of instances. Health markers have been established to trigger alerts of significant changes in network performance, and health metrics have been formulated to calculate a comparable health score for each instance, that is indicative of their current network performance. We have used two testbeds, with up to 100 AWS instances, to explore the cloud with tomographic indicators and evaluate the health system. Finally, we have utilized the real-world e-Science medical imagining application, pCT. We have deployed the pCT work over various subsets of instances, determined by health scores, and found considerable advantages to employing health information during application deployment.

Our future work aims to further investigate the potential of network health to further facilitate e-Science in the cloud. Our immediate goal is to utilize our understanding of cloud networks and apply the network health system to a group of cloud-based scientific gateways. We are currently working to monitor the network health between gateways and provisioned worker instances to improve the overall management of resources and increase the performance of scientific research on the cloud. We also plan to evaluate additional tomographic techniques, as well as establish our own in order to further identify the features of the network between two commercial cloud instances. Additional research is required to explore the effect health information can have on larger scale pCT reconstruction.

References

[1] Q. He, S. Zhou, B. Kobler, D. Duffy, T. McGlynn, Case study for running HPC applications in public clouds, in: Proceedings of the 19th ACM International Symposium on High Performance Distributed Computing, 2010, pp. 395–401.

[2] P. Mehrotra, J. Djomehri, S. Heistand, R. Hood, H. Jin, A. Lazanoff, S. Saini, R. Biswas, Performance evaluation of Amazon EC2 for NASA HPC applications, in: Proceedings of the 3rd Workshop on Scientific Cloud Computing, 2012, pp. 41–50.

[3] Y. Zhai, M. Liu, J. Zhai, X. Ma, W. Chen, Cloud versus in-house cluster: evaluating Amazon cluster compute instances for running MPI applications, in: State of the Practice Reports, 2011, pp. 11:1–11:10.

[4] K. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. Wasserman, N. Wright, Performance analysis of high performance computing applications on the Amazon Web Services cloud, in: Second International Conference on Cloud Computing Technology and Science, CloudCom, 2010, pp. 159–168.

[5] R.K. Madduri, K. Chard, R. Chard, L. Lacinski, A. Rodriguez, D. Sulakhe, D. Kelly, U. Dave, I. Foster, The globus galaxies platform: Delivering science gateways as a service, Concurr. Comput.: Pract. Exp. (2015) advance online publication. http://dx.doi.org/10.1002/cpe.3486.

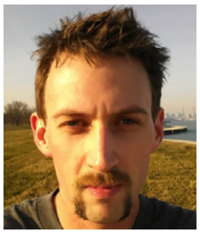
[6] L. Ramakrishnan, P.T. Zbiegel, S. Campbell, R. Bradshaw, R.S. Canon, S. Coghlan, I. Sakrejda, N. Desai, T. Declerck, A. Liu, Magellan: Experiences from a science cloud, in: Proceedings of the 2nd International Workshop on Scientific Cloud Computing, 2011, pp. 49–58.

[7] D. Lifka, I. Foster, S. Mehringer, M. Parashar, P. Redfern, C. Stewart, S. Tuecke, XSEDE cloud survey report, Tech. Rep. 20130919-XSEDE-Reports-CloudSurvey-v1.0, XSEDE, 2013.

[8] R. Chard, R.K. Madduri, N.T. Karonis, K. Chard, K.L. Duffin, C.E. Ordoñez, T.D. Uram, J. Fleischauer, I.T. Foster, M.E. Papka, J. Winans, Scalable pCT image reconstruction delivered as a cloud service, IEEE Trans. Cloud Comput. (2014) submitted for publication.

[9] A. Bestavros, J.W. Byers, K.A. Harfoush, Inference and labeling of metric-induced network topologies, IEEE Trans. Parallel Distrib. Syst. 16 (10) (2005) 1053–1065.

[10] Y. Tsang, M. Coates, R.D. Nowak, Network delay tomography, IEEE Trans. Signal Process. 51 (8) (2003) 2125–2136.
[11] B. Yao, R. Viswanathan, F. Chang, D. Waddington, Topology inference in the presence of anonymous routers, in: INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications, vol. 1, 2003, pp. 353–363.
[12] Y. Tsang, M. Yildiz, P. Barford, R. Nowak, Network radar: tomography from round trip time measurements, in: Proceedings of the 4th ACM SIGCOMM Conference on Internet Measurement, 2004, pp. 175–180.
[13] N.G. Duffield, F. Lo Presti, V. Paxson, D. Towsley, Inferring link loss using striped unicast probes, in: INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, vol. 2, 2001, pp. 915–923.
[14] M. Coates, R. Nowak, Network loss inference using unicast end-to-end measurement, in: Proceedings of the ITC Conference on IP Traffic, Modelling and Management, 2000, pp. 28.1–9.
[15] M. Coates, R. Castro, R. Nowak, M. Gadhiok, R. King, Y. Tsang, Maximum likelihood network topology identification from edge-based unicast measurements, in: Proceedings of the 2002 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems, 2002, pp. 11–20.
[16] D. Battré, N. Frejnik, S. Goel, O. Kao, D. Warneke, Evaluation of network topology inference in opaque compute clouds through end-to-end measurements, in: IEEE International Conference on Cloud Computing, CLOUD, 2011, pp. 17–24.
[17] Y. Gong, B. He, J. Zhong, Network performance aware MPI collective communication operations in the cloud, IEEE Trans. Parallel Distrib. Syst. 99 (1) (2013) 11.
[18] W. Gropp, MPICH2: A new start for MPI implementations, in: Recent Advances in Parallel Virtual Machine and Message Passing Interface, Springer, 2002, page 7.
[19] C. Reich, K. Bubendorfer, M. Banholzer, R. Buyya, A SLA-oriented management of containers for hosting stateful web services, in: Proceedings of the 3rd IEEE International Conference on e-Science and Grid Computing, 2007, pp. 85–92.
[20] C. Reich, M. Banholzer, R. Buyya, K. Bubendorfer, Engineering an autonomic container for WSRF-based web services, in: International Conference on Advanced Computing and Communications, ADCOM, 2007, pp. 277–282.
[21] Y. Gu, G. Jiang, V. Singh, Y. Zhang, Optimal probing for unicast network delay tomography, in: Proceedings of INFOCOM, IEEE, 2010, pp. 1–9.
[22] P. Qin, B. Dai, B. Huang, G. Xu, K. Wu, A survey on network tomography with network coding, IEEE Commun. Surv. Tutor. 16 (4) (2014) 1981–1995.
[23] L. Bai, S. Roy, A two-stage approach for network monitoring, J. Netw. Syst. Manage. 21 (2) (2013) 238–263.
[24] R. Chard, K. Bubendorfer, B. Ng, Network health and e-Science in public clouds, in: 10th International Conference on e-Science (e-Science), Vol. 1, IEEE, 2014, pp. 309–316.
[25] K.P. Burnham, D.R. Anderson, Model Selection and Multimodel Inference: A Practical Information-Theoretic Approach, Springer, 2002.
[26] T.W. Arnold, Uninformative parameters and model selection using Akaike's information criterion, J. Wildl. Manage. 74 (6) (2010) 1175–1178.
[27] K.-Y. Liang, S.L. Zeger, Longitudinal data analysis using generalized linear models, Biometrika 73 (1) (1986) 13–22.
[28] R. Schulte, V. Bashkirov, T. Li, Z. Liang, K. Mueller, J. Heimann, L. Johnson, B. Keeney, H.F.-W. Sadrozinski, A. Seiden, D. Williams, L. Zhang, Z. Li, S. Peggs, T. Satogata, C. Woody, Conceptual design of a proton computed tomography system for applications in proton radiation therapy, IEEE Trans. Nucl. Sci. 51 (3) (2004) 866–872.
[29] N.T. Karonis, K.L. Duffin, C.E. Ordoñez, B. Erdelyi, T.D. Uram, E.C. Olson, G. Coutrakon, M.E. Papka, Distributed and hardware accelerated computing for clinical medical imaging using proton computed tomography (pCT), J. Parallel Distrib. Comput. 73 (12) (2013) 1605–1612.
[30] The gluster web site. http://www.gluster.org/ (Accessed on May 2015).


Ryan Chard is a Ph.D. student at the School of Engineering and Computer Science, Victoria University of Wellington. His research interests include distributed systems, social computing, cloud computing and reputation. He actively teaches undergraduate computer science courses.

Kris Bubendorfer is a Senior Lecturer in Network Engineering at Victoria University of Wellington. He received his Ph.D. in Computer Science, on a mobile agent middleware, from the Victoria University of Wellington in 2002. His current research interests include distributed computing, social computing, digital provenance and reputation. He teaches courses in networking, operating systems, security and distributed systems.

Bryan Ng studied Network Engineering and Statistics at Universiti Malaya. He is a lecturer at Victoria University. His research interests include mathematical modeling and performance analysis of networks and protocols. Prior to joining Victoria, he worked in Orange Labs, INRIA and AC Nielsen consulting.