2014 IEEE 10th International Conference on eScience

Network Health and e-Science in Public Clouds

Ryan Chard and Kris Bubendorfer and Bryan Ng
School of Engineering and
Computer Science
Victoria University of Wellington
Wellington, New Zealand
Email: {ryan, kris, bryan.ng}@ecs.vuw.ac.nz

***Abstract*—Commercial cloud providers are increasingly offering high performance and GPU-enabled resources capable of facilitating e-Science applications. However, the limitations of a public cloud's internal network performance are well documented and can lead to the decision to use dedicated infrastructure over cloud resources for scientific applications. This paper explores the potential for improvement in the performance of e-Science applications on public clouds through the examination of the network in more detail. We introduce health indicators and evaluate various tomographic techniques for their ability to infer information regarding the network connection between instances. We also propose and formulate a set of health markers and health metrics to efficiently assess the network over time in order to make informed deployment decisions. Finally, we evaluate our work through a real-world medical image reconstruction application.**

I. 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 public clouds took place prior to the inclusion of specialized compute instances by commercial cloud providers [2]. However, most research still 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. The Magellan initiative [5] 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. [6] survey uses of public 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) [7]. 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 explores the ability for network tomography to identify network properties and characteristics of a public cloud, and then apply the inferred information to the deployment of e-Science applications. Network tomography is the processes 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 [8]. By measuring the delay incurred by a probe between two end points, congested links that incur long queuing delays, can be detected [9].

Our work investigates network tomography techniques to better 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 and health metrics to compare the network performance of instances. We use the pCT project 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 paper is organized as follows: we discuss related work in Section II and then introduce an overview of our network health diagnostics and metrics in Section III. Section IV introduces our AWS testbed and presents our baseline measurements from which our health diagnostics were derived. We then apply our system to a real-world e-Science medical imaging application (pCT) in Section VI, followed by a discussion of our results in Section VII. Finally, we present our conclusions in Section VIII.

II. Related Work

Many diagnostic tools, such as traceroute, rely on the cooperation of link-layer components [10], yet commercial cloud providers often obfuscate their network, rendering such

978-1-4799-4287-9/14 $31.00 © 2014 IEEE
DOI 10.1109/eScience.2014.36

tools ineffective or disabling them entirely. In 2004 Tsang et al. [11] presented Network Radar, a novel tomographic technique based on RTT that does not require the cooperation of receivers. The work does not require multicast routing capabilities, synchronized clocks, or the capability to capture measurements at both the sending and receiving hosts. Instead, Tsang's work uses 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. [12] and Coates and Nowak [13] present tomography techniques employed to identify lossy links using unicast probes between two end points. However, loss-based mechanisms are becoming less effective due to the reliability of modern connections, especially with light loads. Thousands of probes must be measured before a one percent loss rate can significantly effect end-to-end performance of a link [14].

Coates et al. [14] present a novel probing scheme which extends delay-based unicast, end-to-end measurements, called Sandwich probing. This probing scheme is 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 induced by the large packet can be captured through the difference in RTT of the smaller packets.

Previous work to explore a public cloud network has been conducted by Battre et al. [15]. Their work investigates methods to infer the network topology within opaque cloud infrastructure. The authors use a testbed of 64 Xen VMs to explore and evaluate both loss-, and delay-based, tomography techniques. The results indicate loss to be less effective, and even find that when evaluating using the Robson-Foulds metric, RTT outperforms Sandwich probing. Our work extends this research by evaluating a number of additional network health indicators and investigating their performance with a real-world e-Science application.

An example of research that utilizes network information within a cloud is CMPI. CMPI, presented by Gong et al. [16], is a network-aware implementation of MPI designed for cloud environments. The research investigates optimizing MPI's collective communication algorithms, such that they utilize network performance information. The authors find Amazon's EC2 to have significant network performance unevenness, where performance is not symmetrical between virtual machine pairs. The work proposes using simplified latency and bandwidth matrices to evaluate network performance, and use the derived information to optimize the Broadcast and Reduce, and Gather and Scatter operations. The work results in optimizations of between 13% and 38%, compared to that of MPICH2. CMPI exemplifies the potential opportunities that are available within the cloud when applying network information to applications.

III. High Level View of Our Work

We have adopted and extended the concept of health metrics from the second author's prior work, Reich et al. [17], [18], 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 public cloud domain and extend the concept with health markers and indicators along with network tomography – in order to infer the properties and characteristics of provisioned cloud instances. Calculating health metrics based on the network performance 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 devise a set of health indicators. In this context, a health indicator is a method of gathering information regarding 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 devise 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.

To use the information gathered from health indicators, we formulate a set of health markers. A health marker is a lightweight and easily computable diagnostic, used to gauge the change in performance of the network across probing cycles. Markers are used to quickly establish the degree to which the network performance changes over a period of time and when necessary, prompt the reassessment of the overall health score for an instance.

A health metric is an aggregation of selected health indicator measurements. A set of health metrics are weighted and combined to compute an overall health score for an instance, providing a high-level opinion on the performance of an instance with respect to its peers. A health score gives a basis to compare instances against one another by normalizing health indicator measurements across each instance that the host is probing. When computed, a health score gives a local perspective of the load every other instance in the environment is experiencing, and facilitates the selection of healthy clusters.

In order to investigate the properties of the network and formulate health markers and health metric weights, we have deployed the health system over a testbed of regular AWS instances.

IV. Testbed and Cloud Performance Baselines

We deployed a testbed of regular AWS on-demand instances, of types t1.micro and m3.medium, to observe 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

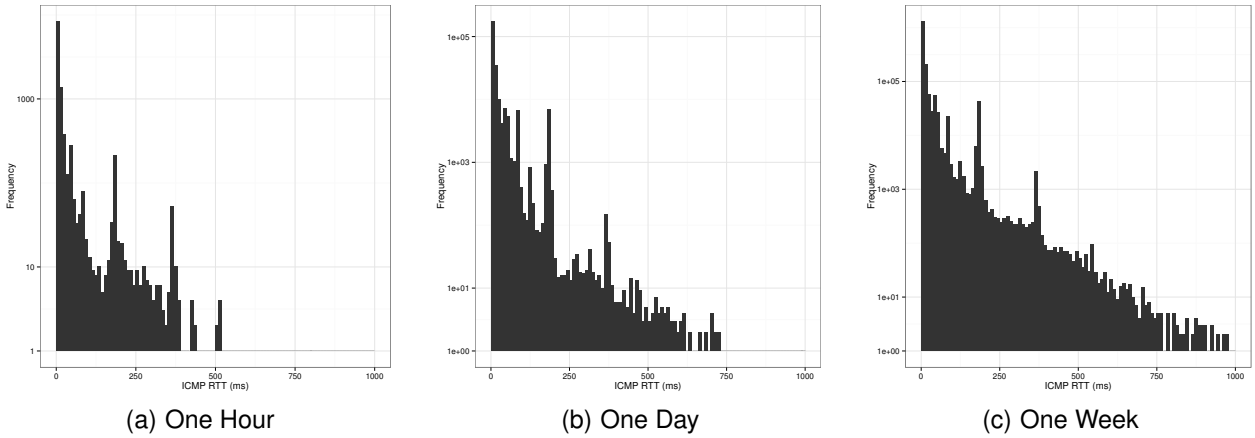

Fig. 1. The frequency of ICMP packet RTTs over different periods of time on a logarithmic scale.

communicate with every other instance in the environment. The probing schedules were deliberately offset in an attempt to reduce the interference caused by multiple hosts concurrently recording measurements on specific hosts. The test was run over the course of one week to thoroughly examine properties relating to recurring events, such as how work hours across the globe influence perceived network health.

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.

Figure 1 shows the volatility of the network by depicting the variation in ICMP RTT probes over different periods of time. Each image contains the frequency of RTT probes with payloads of size 64, 512 and 4096 bytes on a logarithmic scale. The figure shows the distribution of probes contains a significant number of high RTT values, over each period of time. This demonstrates the high degree of volatility that a regular instance exhibits.

Over a one day time period it can be noticed that the variation in performance occurs over sufficiently long periods of time to effect the performance of an application. The findings show that the performance of a link can deteriorate for periods of hours, rather than in short intermittent bursts. An example of this is shown in Figure 2, where the average hourly RTT between two instances, for various ICMP packet sizes, is collected and displayed for an individual day. The figure demonstrates that the performance the of 4096 and 512 byte packets are relatively slow during the beginning of the day and gradually improve to a lower latency. These results are consistent with other measurements granularities and are reflected by the observations from multiple instances. These findings support our objective of monitoring the change in network performance and reacting to it, as they persist for meaningful lengths of time.

In order to further understand the network performance be-

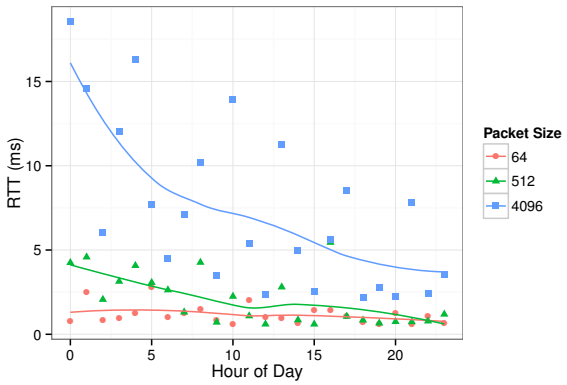

Fig. 2. The hourly average RTT for different packet sizes between two hosts.

tween various instance types and across availability zones we conducted a series of bandwidth measurements. The measurements involve transferring as much data as possible between two instances within a ten second time span. The results found significant bursting characteristics for TCP transfers. Figure 3 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 200Mb/s for the first four seconds of a transfer before being throttled to approximately 80Mb/s. Similarly, the m3.medium instances achieved an initial throughput of almost 1000Mb/s for approximately one second before being throttled to slightly over 200Mb/s. From this data it appears that the instances are granted a burst throughput rate for approximately the first 1000Mb of data being transferred. Additional tests with cluster compute, cg1.4xlarge, instances demonstrated a sustained throughput of approximately 8000Mb/s within an availability zone, and

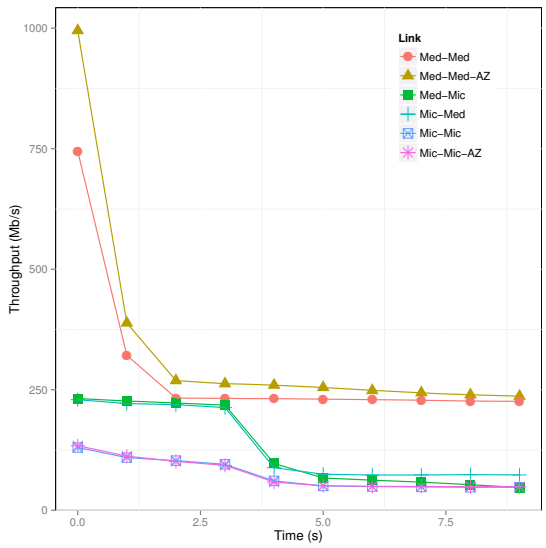

Fig. 3. The average throughput between medium and micro instances within and across an availability zone.

a sustained 2000Mb/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.

These findings demonstrate two interesting properties that support our goal of utilizing network information for e-Science application deployment. Firstly, the 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 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 some examples, the degraded performance of an instance can be observed from multiple hosts for periods of hours.

V. Network Health

A diagnostic system has been implemented to deploy the health indicators over a set of instances. The system utilizes AWS APIs to provision and load instances with a specific Amazon Machine Image (AMI) that is capable of performing the diagnostic tests. A client and server are automatically deployed on each instance, and can connect to a shared database. The client iterates through each known host and conducts a series of indicator measurements. An Amazon Relational Database Service (RDS) instance maintains the list of active instances and the data acquired from each probe.

Deploying the health indicators over a set of instances 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 test to recognize performance changes and prompt the reassessment of the overall health scores for instances in the environment.

Health metrics provide a normalized mechanism to evaluate instances against one another. Health metrics are formulated through an evaluation of health indicators and their ability to detect network load. Each indicator is taken into account and normalized with the other connections being monitored before being aggregated with predefined weights. The aggregate of the health metrics gives a health score for an instance and enables the direct comparison of instances and therefore the selection of healthy clusters.

A. 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 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 frequencies of sending 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. [14], have also been selected. 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 being 1500 bytes, the large packet has been set to 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 cg1.4xlarge instances. Thus we have selected a small packet size of 64 bytes, and large packet size of 1500 bytes for the AWS regular instance implementation.

B. Health Markers

Health markers have been employed to set lightweight diagnostic values which are used to identify degrading network performance. Five health markers have been selected and implemented in order to notify the diagnostic system of significant variations in network performance. These health markers each represent a quantifiable 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 has 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.

C. Health Metrics

A health metric has been formulated for each individual network protocol and typically aggregates the information gathered into a normalized 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 to 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 it's 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.

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

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.

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 employ the variable selection technique to a linear regression model from the complete data set and rank metrics with the strongest influence on the aggregate health score [19], [20]. Based on trace data collected from AWS from 29 April 2014 (10:45:16) to 6 May 2014 (14:24:27) we use the forward step analysis on Eq. (2) to rank the influence of health metrics on the health score. Starting from Step 4 in Table I, 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 I 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 marker $H - UDP_{ij}$ 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.

Step	Health metric	AIC	p-value
4	$H - TP_{ij}$	216.311	0.04126
3	$H - ICMP_{ij}$	98.227	0.00587
2	$H - UDP_{ij}$	90.221	0.00752
1	$H - TCP_{ij}$	106.842	0.00609

TABLE I
FORWARD STEP ANALYSIS OF HEALTH METRICS.

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. We discuss the effect of each health metric on the health score and how the health score aids decision making in selecting cloud instances.

D. Health Score

The timeout of an instance is critical to identify and makes other health metrics irrelevant. Therefore, the overall health metric incorporates timeout values by setting the 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.

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

This work can be extended to establish a global perspective of the network health as a whole. However, when considering the pCT application as an example e-Science workload, the deployment location of the shared file system is pivotal to the performance of the data distribution phase of reconstruction. Due to the data-intensive nature of the pCT workload, it is essential to base execution in proximity to the data.

VI. 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 public cloud. pCT is a medical imaging modality and was developed to acquire high accuracy images for proton therapy applications [21]. 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 100GB 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. [22] 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 [7]. The cloud-based pCT reconstruction solution leverages a shared Gluster [23] 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 [7].

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 fifteen GPU enhanced cluster compute instances, known as the cg1.4xlarge 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.

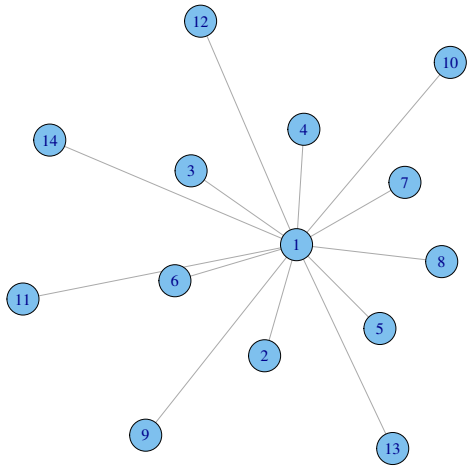

Fig. 4. The proximity of instances used for pCT reconstruction where edge length is determined by health score.

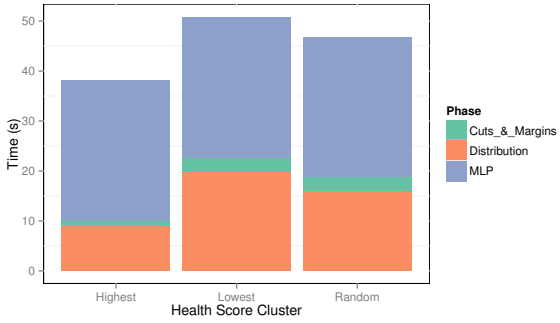

Fig. 5. The average time required for phases of pCT reconstruction by various health score clusters.

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. Figure 4 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 Figure 5. 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 dataset, 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.

VII. Discussion

The performance observed from the regular instance test bed demonstrated that each of the health indicators are capable of identifying network fluctuations. The volatile nature exhibited by the network connecting regular instances provoked significant variations in all of the health indicators that we measured. 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 more drastic than regular instance tests. In order to reflect the requirements of data-intensive e-Science applications, throughput was assigned a higher 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 less 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 is needed to fine tune the markers over various platforms.

The health metrics provided an effective method to compare and evaluate instances against one another. The metrics operated successfully in both 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 zone.

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 [7]. 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.

VIII. Conclusion

Public 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 public 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 public clouds, such as the variability in network performance, and sustained nature of performance fluctuations that an instance undergoes. A diagnostic system has been constructed to monitor the network health between a set of instances. Diagnostic health markers have been established to efficiently report significant changes within the network, and health metrics have been formulated to calculate a comparable health score for each instance, that is indicative of their current network performance. 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. We plan to evaluate additional tomographic techniques, as well as establish our own in order to further identify the features of the network between two public cloud instances. Additional research is required to explore the effect health information can have on larger scale pCT reconstruction. We also aim to investigate methods of visualizing the information that is inferred from the tomographic measurements. Finally, we aim to incorporate the inferred network information into a cloud scheduling service and explore whether further benefits can be applied to real-world e-Science applications.

References

[1] Q. He, S. Zhou, B. Kobler, D. Duffy, and 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, and 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, and 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, and 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] L. Ramakrishnan, P. T. Zbiegel, S. Campbell, R. Bradshaw, R. S. Canon, S. Coghlan, I. Sakrejda, N. Desai, T. Declerck, and A. Liu, "Magellan: experiences from a science cloud," in *Proceedings of the 2nd international workshop on Scientific cloud computing*, 2011, pp. 49–58.

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

[7] 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, and J. Winans, "Scalable pCT image reconstruction delivered as a cloud service," submitted to IEEE Transactions on Cloud Computing. 2014.

[8] A. Bestavros, J. W. Byers, and K. A. Harfoush, "Inference and labeling of metric-induced network topologies," *IEEE Transactions on Parallel and Distributed Systems*, vol. 16, no. 10, pp. 1053–1065, 2005.

[9] Y. Tsang, M. Coates, and R. D. Nowak, "Network delay tomography," *IEEE Transactions on Signal Processing*, vol. 51, no. 8, pp. 2125–2136, 2003.

[10] B. Yao, R. Viswanathan, F. Chang, and 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.

[11] Y. Tsang, M. Yildiz, P. Barford, and 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.

[12] N. G. Duffield, F. Lo Presti, V. Paxson, and 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.

[13] M. Coates and 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.

[14] M. Coates, R. Castro, R. Nowak, M. Gadhiok, R. King, and 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.

[15] D. Battre, N. Frejnik, S. Goel, O. Kao, and 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.

[16] Y. Gong, B. He, and J. Zhong, "Network performance aware MPI collective communication operations in the cloud," *IEEE Transactions on Parallel and Distributed Systems*, vol. 99, no. 1, p. 1, 2013.

[17] C. Reich, K. Bubendorfer, and 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.

[18] C. Reich, M. Banholzer, R. Buyya, and K. Bubendorfer, "Engineering an autonomic container for WSRF-based web services," in *International Conference on Advanced Computing and Communications, (ADCOM)*, 2007, pp. 277–282.

[19] K. P. Burnham and D. R. Anderson, *Model selection and multimodel inference: a practical information-theoretic approach*. Springer, 2002.

[20] T. W. Arnold, "Uninformative parameters and model selection using akaike's information criterion," *The Journal of Wildlife Management*, vol. 74, no. 6, pp. 1175–1178, 2010.

[21] 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, and C. Woody, "Conceptual design of a proton computed tomography system for applications in proton radiation therapy," *IEEE Transactions on Nuclear Science*, vol. 51, no. 3, pp. 866–872, 2004.

[22] N. T. Karonis, K. L. Duffin, C. E. Ordoñez, B. Erdelyi, T. D. Uram, E. C. Olson, G. Coutrakon, and M. E. Papka, "Distributed and hardware accelerated computing for clinical medical imaging using proton computed tomography (pCT)," *Journal of Parallel and Distributed Computing*, vol. 73, no. 12, pp. 1605–1612, 2013.

[23] "The gluster web site," http://www.gluster.org/, Apr 2014.