Evaluating Cost of Cloud Execution in a Data Repository

Zhiwu Xie, Yinlin Chen, Julie Speer, and Tyler Walters
University Libraries
Virginia Polytechnic Institute and State University
Blacksburg, VA, USA
{zhiwuxie, ylchen, jspeer, tyler.walters}@vt.edu

ABSTRACT

In this paper, we utilize a set of controlled experiments to benchmark the cost associated with the cloud execution of typical repository functions such as ingestion, fixity checking, and heavy data processing. We focus on the repository service pattern where content is explicitly stored away from where it is processed. We measured the processing speed and unit cost of each scenario using a large sensor dataset and Amazon Web Services (AWS). The initial results reveal three distinct cost patterns: 1) spend more to buy up to proportionally faster services; 2) more money does not necessarily buy better performance; and 3) spend less, but faster. Further investigations into these performance and cost patterns will help repositories to form a more effective operation strategy.

CCS Concepts

• **Information systems** □ **Digital libraries and archives** • **Networks** □ **Cloud computing** • *Applied computing* □ *Digital libraries and archives*

Keywords

Institutional repository; Big data; Cloud computing; Cost analysis.

1. EXPERIMENT DESIGN

We designed three controlled experiments to execute typical repository tasks in AWS: 1) data ingestion, where File Information Tool Set (FITS) was used to characterize the data files and create associated metadata to the Fedora Objects to be ingested, 2) fixity checking, where new file digests were calculated from the ingested data then compared with their current digest values; and 3) heavy data processing, where multiple Fast Fourier Transformation (FFT) operations were performed against the ingested sensor data. To run these experiments we first installed a Fedora 4 based data repository using a m4.xlarge Elastic Compute Cloud (EC2) instance. This repository instance had a large EBS storage volume attached to it and all data deposited to the repository would be considered locally stored. The cost of this instance was not counted towards the execution costs. The data used for experiments were vibration signals collected from 214 accelerometers mounted in Virginia Tech's Goodwin Hall [1-4], an engineering building and a highly instrumented smart infrastructure laboratory facility. The data were written into one-minute interval zlib-compressed chunked HDF5 files. The experiments made use of three full days of data collected from the accelerometers totaling approximately 223GB. Data was stored at a temporary holding area in a Simple Storage Service (S3) bucket. We then allocated *n* EC2 instances, either in type t2.medium or m4.large, where *n=1, 2, ...9,* to perform the processing. The S3, EBS storages and EC2 nodes were provisioned from the AWS US East Region, such that data movements among them were fast and free of charge.

JCDL '16, June 19–23, 2016, Newark, NJ, USA.
ACM 978-1-4503-4229-2/16/06.
http://dx.doi.org/10.1145/2910896.2925454

2. RESULTS AND ANALYSIS

2.1 Speedup

Figure 1 shows the speedup results of the three experiments. For the ingestion experiment, a linear speedup was consistently observed when using faster m4.large instances. This may be attributed to the vastly parallelizable workload. Because each execution is largely independent from the others in terms of resources needed, doubling the resources cuts the time in half. Situations were markedly different when using smaller, cheaper virtual instances. A superlinear speedup was on display when $n < 5$, then drifted to the linear or slightly sublinear region with larger n. Typically, superlinearity may be achieved when multiple resources can be interleaved. The ingestion process requires chaining three different types of resources: 1) the temporary storage at S3; 2) the processing node using EC2 instances; and 3) the repository node using EC2. Their interleaving is indeed a plausible cause, with the superlinearity slowly disappearing due to the interleaving benefits been sufficiently exploited. However, it is not clear why slower processing nodes can in turn achieve faster ingestion rates, as clearly illustrated when $n > 6$.

For the fixity checking experiments, close to linear speedup was observed at lower *n* for both faster and slower processing nodes, then hit a roof. This indicated a bottleneck was reached, possibly at the repository read/write speed limit, at around 400GB/hour. Faster machines tend to reach this bottleneck faster.

In the heavy data processing case, the bottleneck observed at the previous case is far from being reached, with highest processing speed less than 1/10 previously observed. This allows the expected linear speedup pattern to sustain all experiments, and faster machines yield a slightly faster speedup.

2.2 Cost

We calculate the unit cost by dividing the hourly rate of the aggregated processing instances by the processing speed. This is slightly different from the actual cost, since the Amazon charge rounds up the last partial hour into a full hour.

As shown in Figure 2, the speedup characteristics of the three different workload result in three drastically different cost patterns. The heavy data processing use case illustrates the

expected pattern associated with linearity, where using more processing nodes can process data faster, but at the same or slightly higher unit cost. This pattern is further supplemented by the fixity checking use case, where the predicted cost pattern takes a sharp turn up when a bottleneck is reached. Beyond this point, investing in more resources becomes wasteful. The rather surprising cost pattern is illustrated by the ingestion example, where throwing in more resources to some extent can save money and get the work done faster at the same time. When the data volume grows higher, searching for this optimal combination will be of particular interest to repositories.

In all three sets of experiments, using cheaper, slower instances tends to be more cost effective than using the faster ones if processing speed is not a concern. In some use cases, more expensive instances may be required in order to achieve higher processing speed to match the rate of data generation or collection. More money, however, cannot buy arbitrarily high speed.

3. SUMMARY

This paper describes a few interesting performance and cost patterns encountered in leveraging cloud computing for repository operations. Although the use cases under investigation are representative, we caution too literal a reading into the results. The speed and unit cost numbers are indicative, but are also specific to our cases. However, the trends and patterns illustrated in these cases may be useful in repository cost and service planning.

4. ACKNOWLEDGMENTS

This research is partially supported by Amazon AWS Research Grants.

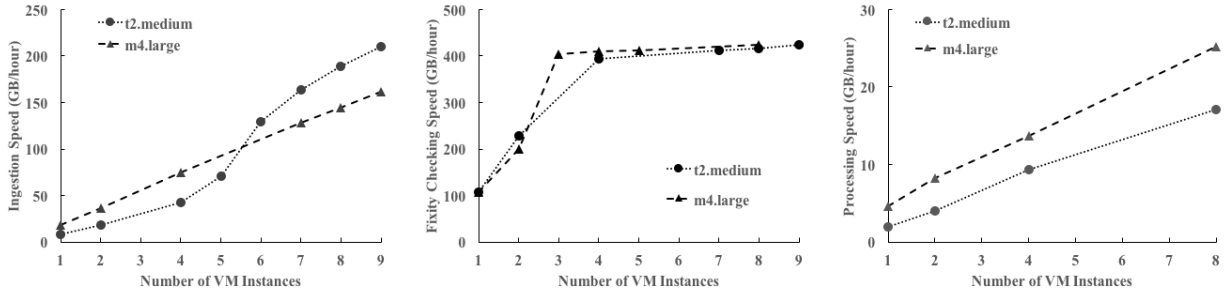

Figure 1. Speedup Using Multiple EC2 Instances.

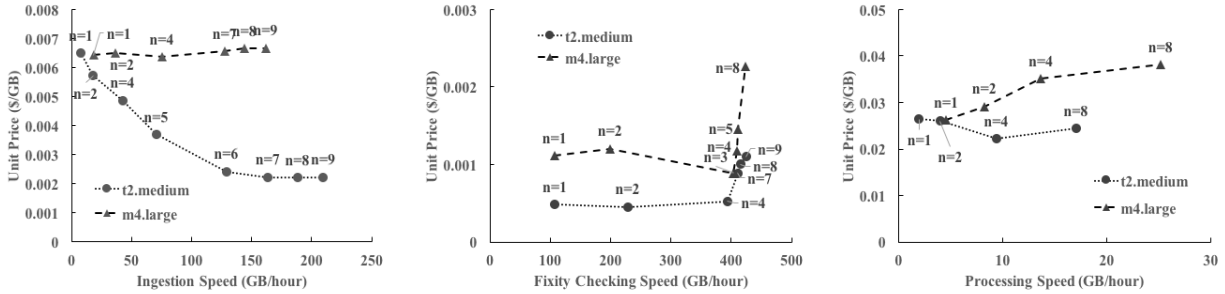

Figure 2. Unit Cost Using Multiple EC2 Instances.

5. REFERENCES

[1] Hamilton, J.M., Joyce, B.S., Kasarda, M.E. and Tarazaga, P.A. 2014. Characterization of Human Motion Through Floor Vibration. *Dynamics of Civil Structures, Volume 4*. F.N. Catbas, ed. Springer International Publishing. 163–170.

[2] Schloemann, J., Malladi, V.V.N.S., Woolard, A.G., Hamilton, J.M., Buehrer, R.M. and Tarazaga, P.A. 2015. Vibration Event Localization in an Instrumented Building. *Experimental Techniques, Rotating Machinery, and Acoustics, Volume 8*. J.D. Clerck, ed. Springer International Publishing. 265–271.

[3] Xie, Z., Chen, Y., Jiang, T., Speer, J., Walters, T., Tarazaga, P.A. and Kasarda, M. 2015. On-Demand Big Data Analysis in Digital Repositories: A Lightweight Approach. *Digital Libraries: Providing Quality Information*. R.B. Allen, J. Hunter, and M.L. Zeng, eds. Springer International Publishing. 274–277.

[4] Xie, Z., Chen, Y., Speer, J., Walters, T., Tarazaga, P.A. and Kasarda, M. 2015. Towards Use And Reuse Driven Big Data Management. *Proceedings of the 15th ACM/IEEE-CS Joint Conference on Digital Libraries* (New York, NY, USA, 2015), 65–74.