Impressionism in Cloud Computing[*]
A Position Paper on Capacity Planning in Cloud Computing Environments

Iván Carrera Izurieta and Cláudio Resin Geyer
Institute of Informatics, Federal University of Rio Grande do Sul - UFRGS, Porto Alegre, Brazil

Keywords: Cloud Computing, Capacity Planning, Virtualization, Performance Evaluation.

Abstract: Cloud computing is a model that relies on virtualization and can lower costs to the user by charging only for the computational resources used by the application. There is a way to use the advantages of cloud computing in data-intensive applications like MapReduce and it is by using a virtual machine (VM) cluster in the cloud. An interesting challenge with VM clusters is determining the size of the VMs that will compose the cluster, because with an appropriate cluster and VM size, users will be able to take a full advantage of resources, i.e., reducing costs by using idle resources and gaining performance. This position paper is intended to bring to consideration the necessity for accurate capacity planning at user level, in order to take fully advantage of cloud resources and will focus specially for data-intensive applications users.

1 INTRODUCTION

Cloud computing, as defined by the U. S. National Institute of Standards and Technology NIST, is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (Mell and Grance, 2011). The cloud infrastructure can be understood as an elastic shared pool of configurable computing resources, so users can define the features of their virtual machine characteristics to meet application requirements, and increase or decrease said features when the requirements change.

Cloud computing nowadays is a paradigm that is becoming more present in many environments and applications. Works like (Boutaba et al., 2012) agree that Cloud Computing is by far the most cost-effective technology for hosting Internet-scale services and applications, and also say that dealing with workload and cluster resources heterogeneity is a very important challenge.

Virtualization is a key technology for Cloud Computing because it introduces an abstraction layer that partitions a computer system into virtual machines as if they were physically isolated (Carissimi, 2008). Managing computers as virtual machines that can be modified and therefore optimized for a certain application is a key advantage for Cloud Computing.

Distributed applications, when moved to cloud environments require a cost-effective and time-effective resource planning for their clusters, formed now by virtual machines. An optimal cluster size should save money by optimizing resources (Boutaba et al., 2012), but this optimization cannot be generalized for all applications (Herodotou et al., 2011).

In this work we discuss the introduction of a methodology that can help a cloud computing application user to take advantage of the inherent elasticity of a cloud infrastructure to determine the size of the virtual machine cluster running in the cloud that can have a predictible performance.

The remainder of this work will be structured as follows: Section 2 discusses the motivation for this work and the relevance of Capacity Planning in cloud environments, Section 3 addresses a methodology for determining general purpose Capacity Planning in cloud environments, Section 4 exposes an ex-

[*] **A Brief Explanation about the Title**
Impressionism was a 19th-century art movement originated by a group of Paris-based artists. Those artists developed a technique in which the representation of objects was made by relatively small, thin, yet visible brush strokes. In Impressionism, no objects were drawn but only represented by these brush strokes whose size was determined by each artist's technique.

In a Cloud Computing environment, virtual machines with variable size compose a cluster that behaves like a single machine as the brush strokes did for the objects in the Impressionism. In our case, the skill for the capacity planner will be to find the exact size of the virtual machines that could achieve a desired performance.

Carrera Izurieta I. and Resin Geyer C..
Impressionism in Cloud Computing - A Position Paper on Capacity Planning in Cloud Computing Environments.
DOI: 10.5220/0004567803330338
In *Proceedings of the 15th International Conference on Enterprise Information Systems* (ICEIS-2013), pages 333-338
ISBN: 978-989-8565-60-0

ample for using the methodology explained in Section 3, and finally Section 5 explains future work for this approach, and finally the Conclusions of this work are presented.

2 MOTIVATION

According to the Magellan Report, a document presented by the U.S. Department of Energy DOE in 2011 (Yelick et al., 2011), a cloud computing model can bring to the user a big gain in terms of scalability and usability of a computing infrastructure; but a cloud computing model still lacks a good management of security and performance.

In the opinion of the authors of this work, a cloud performance management model should guide the cloud user in the capacity planning process of its virtual machine cluster in order to have a time and cost-effective performance for its application.

In a private cloud environment, parameters of the virtual machines such as: number of cores of the processor, amount of RAM, bandwidth capacity and some other features can be freely managed according to the administrator definitions; on the other hand, in a public cloud environment this ability is restricted by the provider's instance types, which are specifications of the virtual hardware that will be provided for the virtual machines. For a named user, a modification to the parameters of the virtual machines will depend of the application, in other words, there is no single way to tune a virtual machine (Herodotou et al., 2011) in a cloud environment; but we believe that for a specific application this could be accomplished and generalized for similar applications.

Works like (Herodotou et al., 2011) and (Boutaba et al., 2012) say that an optimal cluster size should save money by optimizing infrastructure cloud resources for the cloud user and provider. Also, in (Mietzner and Leymann, 2008) the vision of a Software as a Service that requires Infrastucture as a Service level management is described as very important research field and an interesting trend in cloud computing. This vision relates to the resource provisioning based on the application performance. This optimization can let users to take a better advantage of the resources they are using in the cloud.

This way of thinking has been tested, for example in (Wang et al., 2010), where authors evaluate the ratio of the number of virtual machines and the number of cores of a physical computer, and how that ratio affects the performance of a distributed application (a MapReduce application). The overall suggestion of Wang's work is that in a computer with *n* cores there should be instantiated no more than *n* virtual machines. But, as we have said, there are more parameters that can be optimized for a given application in a cloud computing environment.

Another interesting work is (Ibrahim et al., 2009) where virtual machines are tuned for data-intensive applications, but the master node is shown as a single point of failure (SPoF) in a MapReduce application. In that case, the master node is recommended to be instantiated on a physical machine rather than a virtual machine. Ibrahim's work does not consider a cloud computing middleware, as it works directly with XEN based virtual machines. In a cloud computing environment this way of thinking is valid, but will not necessarily present the same results, and likely the experiments will not be the same either.

So, by citing these works we show the relevance for this particular issue of determining *a-priori* the size of a virtual machine cluster for distributed applications to be run in the cloud, with the objective of having a time- and/or cost-effective performance.

The approach intended in this work is a way to present a methodology that can guide users finding a capacity planning model for cloud computing applications by modeling the performance of an specific application.

3 PROPOSED METHODOLOGY

In this section it will be explained how a capacity planning model should be done for any specific cloud computing application.

3.1 Define the Parameters that Affect the Performance

For any application, it should be determined which characteristics of the cluster affect its performance. As it has been said, virtual machines in a private cloud can be configured by customizing the number of cores, amount of RAM, storage capacity and some other features depending of the cloud computing middleware. All of these parameters could affect the performance in some way or another, the key will be to determine for an specific application whether or not this performance variations are significant.

In a public cloud environment these factors are not so easily modifiable, because of the instance types each cloud platform defines. So, the instance type is *per-se* a unique factor and should be treated as such.

Also, some other application-specific parameters affect the performance of an application, and they could be as diverse as the kind of applications any user could want to run in a cloud computing environment. For distributed applications, which is our study field, the workload is certainly an important factor that affects the performance of the cluster. In MapReduce applications it is not only important to know the amount of workload that a certain application wants to process, but it is also important to know the type of workload it is. Works like (Abad et al., 2012) study the kinds of workloads a MapReduce application can have, and other works like (Sangroya et al., 2012) even characterize the workloads on several groups, according to their types.

Another very important parameter is the number of virtual machines that would compose the cluster. Depending on their escalability and degree of parallelism, some applications could benefit from running in bigger or smaller clusters.

Defining these parameters will help users to focus on what is actually affecting the performance.

3.2 Perform Observations

After the parameters that affect the performance are defined, observations of the behavior of the application have to be performed. Observations generate metrics, e.g. execution time, throughput, and time between errors (Jain, 1991), and are done with probes inserted into a normal implementation of the application. The selection of these metrics depends on the nature of the application and what the user defines as performance. By varying independently each factor, a user could observe which parameters are responsible for how much of the variation on the performance. This study is known as general full factorial experimental design, which is a generalization of the $2^k r$ experimental design proposed in (Jain, 1991).

This approach has its disadvantages because it could be very extensive to test each and every combination of parameters. For example, if a public cloud user wants to test its application [2] with:

- 3 types of workload,
- 4 sizes of each workload,
- 4 instance types for the virtual machines,
- and 3 different cluster sizes,

this will make a total of:

```
3 x 4 x 4 x 3 = 144 experiments
```

and each experiment run 30 times in order to have statisticly significant results.

```
144 x 30 = 4320 iterations in total.
```

4320 iterations could be impractical to perform.

So, some of the cases should be cut out of the experimental design, using the fractional factorial experimental design explained in (Jain, 1991), mainly because some of the combinations will be more important than others.

Observations done at this step will provide the information to continue with the following mathematical modeling.

3.3 Propose a Model

From the observations done in the previous step, some measures must be taken. If the goal of the model is to improve the performance, perhaps the execution time should be the main metric to be measured, but also some other metrics could be important to model the application performance like: energy consumption, throughput, resource utilization, time between errors, and some others.

After enough metrics are taken, depending on the nature of the observations, using regression models and statistical tools, a mathematical model can be proposed (Jain, 1991). There are some general ways that can help the user to propose a model, for example, the number of instances are inversely proportional to the execution time (the more machines compose the cluster, lesser the time the application will need to be performed, until it reaches a limit); some other metrics could be directly proportional, like the amount of workload (the more workload it is to be processed, the more time the application will need to be performed).

So, some previous criteria should help the user to propose a model that will relate the parameters specified in section *3.1* to the metrics taken from the observations.

3.4 Assess the Model

Once the user has got a model that specifies a mathematical relation for the parameters of the cluster and the application with the performance metrics, this model will have to be proven by experimentation.

The experiments done at this step should confirm (or not) the mathematical model with a certain confidence interval. So, values of the parameters should be given and, with these values, the metrics should be calculated *a-priori* (before the experiments) and then the experiments should be performed in order to confirm the calculations.

This is where things get interesting, because if the model is not quite correct, the results in this step should give feedback to correct and tune the mathematical model.

[2] If it were in a private cloud, there would be even more variations for instance types

For some help to the user that is modeling the performance of the application, some limits for the parameters should also be given, like saying that some model is useful if the parameters are between some limits specified by the application. This could help making the model easier to define, but at the same time this would take credibility from the model. The strength of a model is its generalization.

In figure 1 we present a simple flowchart that sums up the steps described in this section:

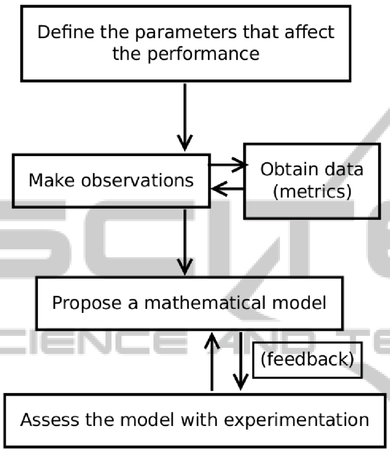
Define the parameters that affect the performance
Make observations
Obtain data (metrics)
Propose a mathematical model
(feedback)
Assess the model with experimentation

Figure 1: Flowchart for developing a model for capacity planning.

4 A PRACTICAL EXAMPLE

We now present an example on how the methodology presented in this work should be put into practice. We performed some experiments to determine which components of the virtual machines in the cluster have an incidence on the performance of the Map Reduce application.

4.1 Description of the Experiments

For this experiments, the System Under Testing, *SUT*, was a cluster of four virtual machines inside a cloud environment. Inside of this *SUT* the component studied was the cluster as a whole. Understanding the cluster as a whole gave us the idea that the cluster behaves as a single thing, and not as a collection of several components, which was better for improving the cluster performance.

The specified system ran a MapReduce application, particularly a Word Count application. And in the experimentation, the errors in execution were not considered, so all outcomes were considered as correct.

4.1.1 Parameters

The factor to study in these experiments was one: the amount of RAM of each virtual machine. The workload was fixed to a 512MB file of plain-text data for the Word count application.

Three values of RAM were tested: 512MB, 1024MB and 1536MB. The experiments were performed 40 times.

4.1.2 Observations

The speed of the system was considered as the main and only criteria for these experiments. The execution time was taken from the log files provided by the Map Reduce application.

For these experiments, the following configuration was used:

- For the Cloud infrastructure:
  - Cloud middleware: Eucalyptus 3.1.2
  - Virtual machine hypervisor: KVM
  - 4 physical cloud nodes: 1 master, 3 slaves
  - Node: Dual-core, Intel-VT, 4GB RAM, 1GbEthernet, 180GB HDD
- For the Map Reduce application:
  - Virtual machine cluster of 4 virtual nodes
  - Apache Hadoop 1.1.1 (released in Nov. 2012)
  - Example Wordcount application, included in the
  - Hadoop 1.1.1 package
  - Workload: 512MB of plain-text data
  - Metrics: execution time obtained from log files

4.1.3 Mathematical Model

Two regression models were applied to the taken data from the experiments.
For a linear model like:

$$time = \beta_0 + \beta_1 * RAM \quad (1)$$

Using the linear regression tool in R software [3] The two constants were calculated to:

$$time = 279,59 - 0,05 * RAM \quad (2)$$

This linear model has a coefficient of determination[4] R^2 = 0,8087.

[3] The R Project for Statistical Computing http://www.r-project.org/

[4] The coefficient of determination is a value that measures the accuracy of a model. A model is better when R^2 gets closer to 1 (Jain, 1991).

And, for a quadratic model like:

$$time = \beta_0 + \beta_1 * RAM + \beta_2 * RAM^2 \quad (3)$$

The three constants were calculated to:

$$time = 259 - 1,044 * 10^{-2} * RAM - 2,344 * 10^{-5} * RAM^2 \quad (4)$$

This quadratic model has an R^2 = 0,8187

4.1.4 Assessing the Model

Both values of R^2 cannot help to say if the relation between RAM and the execution time is actually linear or quadratic, so, more tests will be needed. And visually, no tendencies can be seen when the results are plotted:

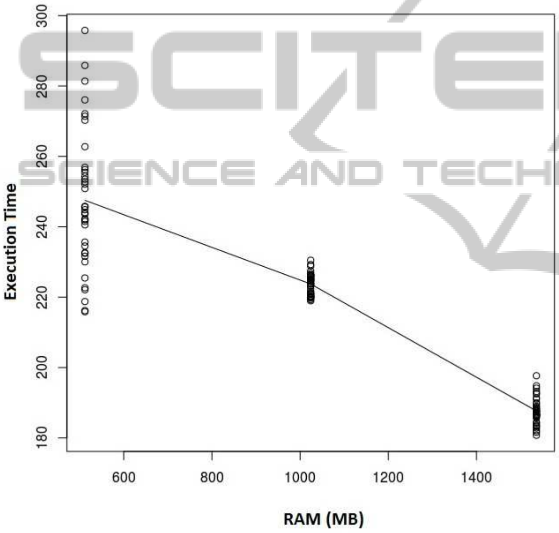

Figure 2: Execution time as a function of RAM of each VM.

Also, we can see that the results obtained with 512MB of RAM had a much bigger dispersion than the results with 1024 or 1536MB. This shows that 512MB of RAM was not an efficient size of memory for the application. Some further work should include more values of RAM for evaluating the performance, so the relation between RAM and the execution time can be modeled in a more precise way.

So, the conclusions for these example experiments can guide for new experiments and give feedback.

5 FUTURE WORK AND CHALLENGES

The objective of this work is to present a model that can help users to find a capacity planning model for cloud computing applications.

This work was motivated by a research that is currently being performed with MapReduce applications, so, the future work is to apply this capacity planning experimental model to MapReduce applications running in a cloud computing environment.

With MapReduce applications there are also some more parameters to consider, one of them is the heterogeneity of the cluster. There are some works that address heterogeneity issues, like (Anjos et al., 2012) and (Zaharia et al., 2008), which stablish that a virtual machine cluster should be homogeneous in order to have a better performance, but they also say that in a cloud computing environment it is hard to have homogeneity in terms of disk and network I/O because of the co-location of virtual machines. It means that sometimes other virtual machines running in the same physical server could affect the performance of virtual machines making harder to model the behavior of the cluster.

So, the I/O heterogeneity is an important challenge that should be assessed when modeling capacity planning for MapReduce applications. In this case, testing high performance storage platforms like the ones optimized for CDNs (Content Delivery Networks) is strongly recommended because will present different performance models, and also an interesting challenge.

When it comes to cloud computing environments, an important issue to assess is the costs that using a public cloud generates, so another interesting future work could be determining the price that could take a certain amount of workload to be processed and which specific parameters of a virtual machine cluster would provide the most cost-effective combination. This would have to take into account that not always the most time-effective combination will also be the most cost-effective.

Also, the methodology presented in this work should help modeling other cloud computing distributed applications, so we encourage other researchers to try to confirm the methodology with other parallel application problems running in a cloud computing environment like graphic computation, distributed filesystems, data processing, content delivery, search engines, and others.

6 CONCLUSIONS

As we have said in this work, this simple but general methodology could come handy when modeling the behavior of a parallel application in a cloud computing environment.

Determining a cluster size for a cloud computing application will help users to improve the performance and the time they pay for using the cloud infrastructure. Also, by optimizing the use of an infrastructure we will be offering more resources for other users.

The reason why we present this methodology as a new contribution is that it helps cloud computing users take advantage of the exclusive feature of the cloud, like elasticity, freedom for sizing the virtual machines, the shorter time that would take for a system administrator review his cluster sizing and perform changes on it.

This methodology could be used in other distributed environments, but it is in the cloud that a user can really play with several configurations of virtual machines.

We want to present this methodology for applications running in the cloud as general as we can, so that any user can take advantage of it. The future work indicated in the previous section will confirm (or not) if this methodology is general enough to have the strength to help defining a mathematical model for MapReduce applications, and hopefully some other cloud applications as well.

ACKNOWLEDGEMENTS

This work was made with the support of the Programa Estudantes-Convênio de Pós-Graduação PEC-PG, of CAPES/CNPq - Brazil.
This position paper was partially funded by Edital FAPERGS-CNPq n. 008/2009 Research Project: GREEN-GRID: Sustainable High Performance Computing, Edital MCT/CNPq No 70/2009 PGAESTMCT/CNPq

Also, we would like to thank Professor Arnaud Legrand from the CNRS, *Centre National de la Recherche Scientifique*, and the University of Grenoble in France for his help and guidance in simplifying the methodology presented in this work.

And a final acknowledgement to the special person who inspired this vision of Impressionism in Cloud Computing a few years ago by explaining the art movement.

REFERENCES

Abad, C. L., Roberts, N., Lu, Y., and Campbell, R. H. (2012). A storage-centric analysis of mapreduce workloads: File popularity, temporal locality and arrival patterns. In *Workload Characterization (IISWC), 2012 IEEE International Symposium on*, pages 100–109. IEEE.

Anjos, J., Kolberg, W., Geyer, C. R., and Arantes, L. B. (2012). Addressing data-intensive computing problems with the use of mapreduce on heterogeneous environments as desktop grid on slow links. In *Computer Systems (WSCAD-SSC), 2012 13th Symposium on*, pages 148–155. IEEE.

Boutaba, R., Cheng, L., and Zhang, Q. (2012). On cloud computational models and the heterogeneity challenge. *Journal of Internet Services and Applications*, 3(1):77–86.

Carissimi, A. (2008). Virtualização: da teoria a soluções. *Minicursos do Simpósio Brasileiro de Redes de Computadores–SBRC*, 2008:173–207.

Herodotou, H., Dong, F., and Babu, S. (2011). No one (cluster) size fits all: automatic cluster sizing for data-intensive analytics. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 18. ACM.

Ibrahim, S., Jin, H., Lu, L., Qi, L., Wu, S., and Shi, X. (2009). Evaluating mapreduce on virtual machines: The hadoop case. In *Cloud Computing*, pages 519–528. Springer.

Jain, R. (1991). *The art of computer systems performance analysis*, volume 182. John Wiley & Sons Chichester.

Mell, P. and Grance, T. (2011). The nist definition of cloud computing (draft). *NIST special publication*, 800:145.

Mietzner, R. and Leymann, F. (2008). Towards provisioning the cloud: On the usage of multi-granularity flows and services to realize a unified provisioning infrastructure for saas applications. In *Services-Part I, 2008. IEEE Congress on*, pages 3–10. IEEE.

Sangroya, A., Serrano, D., and Bouchenak, S. (2012). Benchmarking dependability of mapreduce systems. In *Reliable Distributed Systems (SRDS), 2012 IEEE 31st Symposium on*, pages 21–30. IEEE.

Wang, P., Huang, W., and Varela, C. A. (2010). Impact of virtual machine granularity on cloud computing workloads performance. In *Grid Computing (GRID), 2010 11th IEEE/ACM International Conference on*, pages 393–400. IEEE.

Yelick, K., Coghlan, S., Draney, B., Canon, R. S., et al. (2011). The magellan report on cloud computing for science. *US Department of Energy Office of Science, Office of Advanced Scientific Computing Research (ASCR) December*.

Zaharia, M., Konwinski, A., Joseph, A. D., Katz, R., and Stoica, I. (2008). Improving mapreduce performance in heterogeneous environments. In *Proceedings of the 8th USENIX conference on Operating systems design and implementation*, pages 29–42.