Workload Balance in Cloud-replicated Services

Workload Balance in Cloud-replicated Services

César Angonese
angonese@utfpr.edu.br
Universidade Tecnológica Federal do Paraná (UTFPR)

Alcides Calsavara
alcides@ppgia.pucpr.br
Pontifícia Universidade Católica do Paraná (PUCPR)

Luiz Augusto de Paula Lima Jr.
laplima@ppgia.pucpr.br
Pontifícia Universidade Católica do Paraná (PUCPR)

Resumo

O problema do balanceamento da carga de trabalho entre as réplicas de um serviço que estão implantadas em um *data center* é discutido, incluindo o *design* de sistema principal e parâmetros que podem afetar a performance dos sistemas. Algumas questões críticas, ou seja, a abordagem do equilíbrio da carga de trabalho e a estratégia de replicação são avaliadas através da simulação. Os resultados preliminares mostram que a performance dos serviços replicados podem variar significativamente, dependendo da localização da réplica e do mecanismo de seleção da réplica

Palavras-chave: Balanceamento de carga. Computação em nuvem. Data centers.

Abstract

The problem of workload balance among the replicas of a service that is deployed in a data center is discussed, including the main system design and parameters that can affect systems performance. Some critical issues, namely workload balance approach and replication strategy, are evaluated through simulation. The preliminary results show that the performance of replicated services can vary significantly, depending on replica placement and replica selection mechanism.

Keywords: Load balance. Cloud computing. Data centers

1 Introduction

An important benefit from cloud computing platforms is the dynamic adjustment of system resources allocated for a given service according to the demand (P. MELL, T. GRANCE, 2011), (L. M. VAQUERO, 2009) and (C. TAURION, 2009). Typically, that can be achieved through service replication; replicas of a service can work in parallel in order to process its incoming requests. Such technique should provide for a stable service response time, independently of demand variation. However, a data center is a complex environment, including resource partitioning and sharing among virtual machines (B. MARCELO, 2006). Moreover, a data center can host many services simultaneously, and that can cause impact on performance. The great challenge is to keep the workload balanced between all machines of a data center, such that their idleness is minimized.

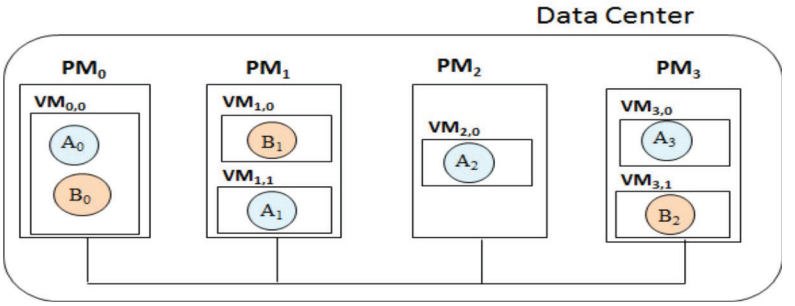
This paper is structured as follows. Section 2 describes the problem of workload balance in data centers. Section 3 discusses some system design issues and parameters, especially the alternative approaches to achieve workload balance and its relation with the replication strategy. Section 4 presents the main algorithms employed by data centers to achieve workload balance. Section 5 describes an experiment that has been carried out to verify the impact of some system design strategies and corresponding results. Finally, Section 6 presents some concluding remarks.

2 Description of the Problem

A data center is basically composed of physical and virtual machines and service providers as illustrated in Figure 1. Let Π denote the set of physical machines connected by a network and let ϒ be the set of virtual machines of a data center. Each $VM \in \Upsilon$ runs on top of some $PM \in \Pi$. Moreover, let ∑ be the set of services running on virtual machines in ϒ. The following properties hold:

- each physical machine $PM \in \Pi$ hosts any number of virtual machines $VM \in \Upsilon$;
- each service $S \in \sum$ may have an arbitrary number of replicas designated by $S_0, S_1, ..., S_k$ $(K \in \square^+)$;
- each service replica S_i runs on a single virtual machine in ϒ;
- a virtual machine hosts, at least, one service replica;
- a service request R_s is associated with service S, and not with a specific service replica S_i;
- a service request R_s is processed by a single replica S_i of that service;
- a service request R_s can be processed indistinctively by any replica S_i of service S, independently.

Figure 1 - An example of replicated services hosted by a data center



The cloud-replicated services workload balance problem consists of selecting which service replica will process each service request, such that the overall system performance is optimized, that is, the average response time is minimized by equally distributing the service work load among physical machine. Notice that the problem does not include replica consistency and that issue is out of the scope of this paper.

In the example of Figure 1, $\Pi = \{PM_0, PM_1, PM_2, PM_3\}$, $= \Upsilon \{VM_{0,0}, VM_{1,0}, VM_{1,1}, VM_{2,0}, VM_{3,0}, VM_{3,1}\}$ and $S = \{A. B\}$. The replicas of service A are Ai: $0 \leq i \leq 3$ and the replicas of service B are $B_j : 0 \leq j \leq 2$. Notice that, due to some system policy, replicas A_0 and B_0 are located in a single virtual machine $VM_{0,0}$, while the remaining virtual machines hold only one replica each. In this example, when a request for service A arrives at the data center, it can be processed by any of its replicas, and the same applies to a request for service B. For the sake of clarity, let us assume that service A is a query-based public service, such as the web site of the Instituto Brasileiro de Geografia e Estatística (IBGE), while service B is a transaction-based service, such as the Amazon e-commerce web site. Thus, in the example, there are four replicas of the IBGE server (A_o, A_1, A_2 and A_3) and three replicas of the Amazon server (B_o, B_1 and B_2). Thus, a client request for the IBGE service may be processed within any of the four physical machines, independently of the addressed virtual machine by the request; the server replica for the IBGE service that will process the request will be chosen by the system according to the physical machines workload. Moreover, the number of server replicas for each service should be determined according to some criteria, including client requests demand and costs. Their placement, on the other hand, should be determined according to network infrastructure, physical machine load, client proximity, legislation constraints, costs, etc. It should be noticed that the complex decision about the number and the placement of replicas relies on the assumption that the system provides mechanisms that guarantee a certain quality of service regarding response time to client requests. In the example, the decision about the number and the placement of replicas for the IBGE service may aim an expected response time of two seconds, while the decision about the number and placement of replicas for the Amazon service may aim an expected response time of five seconds. In addition, the virtual machines that host the servers should be dimensioned to fulfill such a requirement.

Clearly, the optimal average response time can be compromised if relatively too many service requests are sent to replicas that are placed in the same physical machine while replicas in other physical machines are idle. In other words, all these decisions regarding replica number and placement, and virtual machine dimensioning may become compromised

if, due to an inefficient workload balance between physical machines, a certain physical machine gets overloaded due to a relatively large number of requests and, consequently, stops providing the agreed quality of service regarding time response; an efficient workload balance mechanism will guarantee that the agreed time response will be broken only if all physical machines are fully busy.

3 Design Issues

Basically, the effectiveness of a solution to the workload balance problem will depend on several interrelated system design issues and parameters, as discussed in (X. ZHONG, H.RONG, 2009), (S. SHARMA et al, 2008) and (W. LEINBERGER et al, 2000), including the following:

- **The processing power of each physical machine.** Typically, the power of a physical machine is measured in millions of full word-size floating-point multiply operations that can be performed per second (MFLOPS). The set of physical machines can be either homogeneous or heterogeneous with respect to the processing capacity.
- **The number of cores per physical machine.** Each physical machine can have a different number of processing cores. Its power is given by the sum of the powers of its cores.
- **The power of each virtual machine.** The *absolute power* of a virtual machine corresponds to its processing power, measured in MFLOPS. A set of virtual machines can be either homogeneous or heterogeneous with respect to machine power. The *relative power* of a virtual machine corresponds to the ratio between that virtual machine power and the corresponding physical machine power.
- **The physical machine scheduling policy.** *Time-shared policy*: all virtual machines placed in a certain core run in parallel, thus sharing core power at any moment. *Space-shared policy*: the virtual machines placed in a certain core run one at a time, that is, for each core, only one virtual machine can run at a given moment.
- **The virtual machine scheduling policy.** *Time- shared policy*: service requests within a certain virtual machine are processed in parallel, thus sharing virtual machine power at any instant. *Space-shared policy*: service requests within a certain virtual machine are queued and processed one at a time, that is, only one request can be being processed at a given instant.
- **The service requests pattern.** *Size*: the size of a request is given by the amount of processing power required, typically measured in MFLOPS. *Frequency*: the number of requests per time unit gives the frequency of service requests. Both size and frequency can vary, depending on the service and seasonality.
- **The number of replicas of each service.** This number can be either static (defined by the system initial setup) or dynamic (replicas are created and destroyed as needed in order to adjust to service request patterns).
- **The placement of service replicas.** The placement of each service replica on a virtual machine can also be either static or dynamic. In the latter approach, a replica can migrate from one virtual machine to another when it needs more processing power, for instance. In the specific case where a virtual machine hosts only one service replica, the virtual machine itself along with its service replica can migrate. In any case, a migration can occur either between two distinct physical machines or between two distinct cores of a certain physical machine.
- **Service interdependency**. There may be stronger and weaker dependency relations among services. As a consequence, some services may need to communicate more or less frequently with other services. Therefore, it would be better, for the sake of performance, to place closely related services on the same virtual machine, or, at least, on the same physical machine, even at the expense of a relatively unbalanced load distribution.
- **The network infrastructure**. Messages are employed to deliver a service request to a service replica, and also to send a reply to the client. In addition, the mechanism for selecting a replica to process a request can be based on message exchange. Finally, service interdependency may cause network traffic. Hence, network infrastructure plays an important role in providing good service response time.
- **Failure recovery**. System failures, such as physical machine crash and message loss, can compromise service operation. A simple way to handle a service failure is to let the client decide whether the corresponding request should be reissued. Alternatively, a data center can implement mechanisms for fault tolerance so that a request is very likely to be successfully processed. For example, if the physical machine where a service request is placed for processing crashes, a recovery mechanism could trigger a new processing of that request on some available replica of the corresponding service. Anyway, a system failure will cause some delay, thus affecting the average response time.

Additionally, the approach taken to implement a workload balance mechanism can be centered on either service or physical machine. In a *service-centered approach*, the aim is to keep balance among all replicas of each particular service, while, in a *physical-machine-centered approach*, the aim is to keep balance among all physical machines in the data center.

As a simple example, let us consider a data center composed of three single-core physical machines, namely PM_0, PM_1 and PM_2, where they all have the same processing power. In addition, let us consider that such data center has to host three distinct services, namely *A*, *B* and *C*. A possible replication strategy is to keep a replica of each service on each physical machine, as illustrated by Figure 2 and 3. Here, such strategy is referred to as *full replication*. Also for simplicity, each replica is placed on an exclusive virtual machine, and all virtual machines have equal processing power. By assuming that all services present a similar request pattern (similar frequency and similar size), Figure 2 shows a possible distribution of a sequence of nine service requests composed of three requests per service among replicas (each request processing is represented by a *cloudlet*) in the case of a physical-machine-centered approach, while Figure 3 shows a possible distribution of the same sequence of service requests in the case of a service-centered approach. It can be noticed that, under full replication, physical-machine-centered workload balance does not imply service workload balance, since the number of cloudlets varies between the replicas of each particular service. Even though physical machines run three cloudlets each, it is unclear the impact on performance, that is, that impact on services response time caused by the relative unbalance between the virtual machines that are placed on each particular physical machine. Surely, that will depend on a combination of several system parameters, including the relative power of the virtual machines, physical machine scheduling policy and virtual machine scheduling policy. On the other hand, service-centered workload balance implies physical machine workload balance and, as well, balance between virtual machines placed on each physical machine. Such perfect balance suggests a scenario where the best possible performance is achieved, independently of virtual machine relative power or scheduling policies.

Alternatively, depending on environment conditions, such as costs constraints, each service can have a distinct number of replicas. As illustrated by Figure 4 and 5, service *A* is replicated on each physical machine, that is, three times, while service *B* is replicated only PM_0 and PM_1, and service *C* is not replicated at all; its single instance is placed on PM_2. Here, such strategy is referred to as *partial replication*. Figure 4 shows a possible distribution of a sequence of twelve service requests composed of four requests per service among replicas in the case of a physical-machine-centered approach, while Figure 5 shows a possible distribution of the same sequence of service requests in the case of a service-centered approach. It can be noticed that neither physical-machine-centered workload balance implies service workload balance nor service-centered workload balance implies physical machine workload balance. Although the data center is balanced with respect to the number of virtual machines per physical machine (there are two virtual machines per physical machine), both approaches imply unbalance between the virtual machines placed on each physical machine. Thus, in the case of partial replication, for both approaches, virtual machine relative power and scheduling policies should severely impact on services response time. Those should be even more relevant when a service-center approach is taken, since an unbalance between physical machines is observed too, in addition to the unbalance between the virtual machines placed on each physical machine.

A full replication scenario is very unlikely to be found in real-world data centers. Nevertheless, it can be useful for the purposes of benchmarking, since the performance results observed in a service-centered based solution are, in practice, an upper bound. Therefore, a difficult challenge in the cloud-replicated services workload balance problem is to find a solution that performs well in the general case, that is, under partial replication.

Figure 2 - Physical-machine-centered workload balance under full replication.

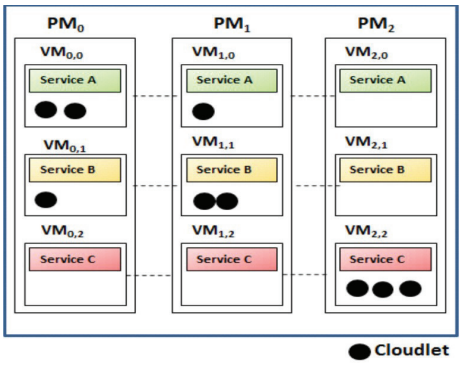


Figure 3 - Service-centered workload balance under full replication.

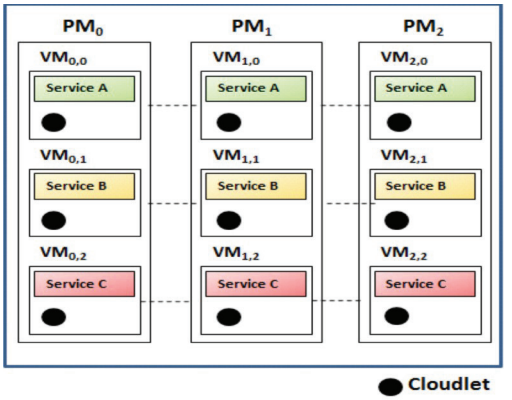


Figure 4 - Physical-machine-centered workload balance under partial replication.

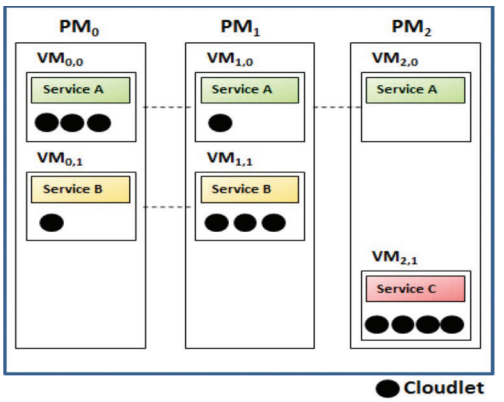
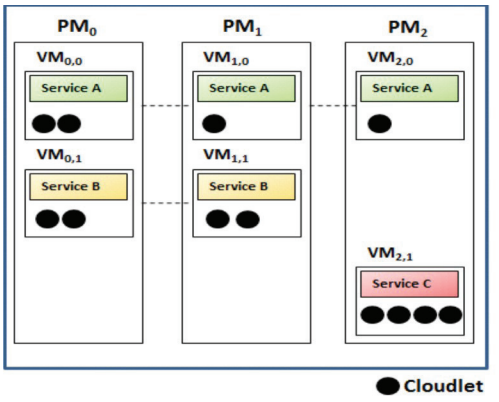


Figure 5 - Service-centered workload balance under partial replication.



4 Related Work

A load balancing algorithm aims at keeping all servers at the same workload at any given time (L. A. BARROSO, U. HOLZLE, 2009). The case where the servers are the service replicas corresponds to the service-centered approach, and it applies to both full and partial replication. The case where the servers are physical machines correspond to the physical-machine-centered approach. At first, an algorithm based on the physical-machine-centered approach applies only in the case of full replication. However, some algorithms can be modified to work under partial replication by mapping physical machine attributes to service replica attributes. For instance, the workload of a service replica can be identical to the workload of the corresponding physical machine for the purpose of load balancing between service replicas. Moreover,

both approaches service-centered and physical-machine centered are equivalent under full replication. According to (P. NUSRAT, A. AGARWAL, R. RASTOGI, 2014), (S. RAY, A. DE SARKAR, 2012) and (CITRIX, 2011), the load balancing algorithms typically employed in data centers are the following:

- In the Round Robin Algorithm, servers are organized in a circular queue, so that each client request is sent to the next server in the queue. Such server can be either a service replica or a physical machine.
- In the Throttled Algorithm, a maximum number of simultaneous requests is established for each server. While all servers are fully busy, client requests are kept in a queue.
- In the Active Monitoring Algorithm, a new client request is sent to the server with the lowest current workload.
- In the Least Connections Algorithm, a new client request is sent to the server with the fewest active connections.
- In the Least Response Time Algorithm, a new client request is sent to the server with the fewest active connections and the lowest average response time.
- In the Least Bandwidth Algorithm, a new client request is sent to the server that is currently serving the least amount of traffic.
- In the Least Packets Algorithm, a new client request is sent to the server that received the fewest packets in the last time interval, previously established.
- In the Randomized Algorithm, a probability for receiving new requests is assigned to each server, so that a new client request is sent to a server that is randomly chosen according to the assigned probabilities.
- In the Token Algorithm, a token is associated to each client request so that a knowledge base about the load of each server is built, thus permitting the choice of the most suitable server for each request.
- In the Central Queuing Algorithm, all client requests are stored in a queue as activities to be dispatched to the servers. Concurrently, the corresponding queue manager accepts requests for activities from servers. When a request for activity arrives, the first activity in the queue is dispatched to the corresponding server; while no activity is available, the server request is buffered.

5 Experiments

An experiment has been carried out to evaluate the impact of the workload balance approach and the replication strategy on systems performance. The experiment employed the CloudSim simulator (R. BUYYA et al, 2009) to evaluate a data center composed of eight single-core physical machines, denoted as PM_1 to PM_7, where four services, denoted as A, B, C and D, are deployed. In the case of partial replication, service A is deployed on PM_0, PM_2, PM_4 and PM_6, service B is deployed on PM_1, PM_3 and PM_5, service C is deployed on PM_0 and PM_7, and service D is deployed solely on PM_1. Thus, PM_0 and PM_1 host two replica services each, while the remaining physical machines host a single replica service. For simplicity, each virtual machine holds only one service replica. All the physical machines and all the virtual machines have equal processing power (2,660 MIPS), while the average size of each request corresponds to half of a machine power (1,330 MIPS) and presents a uniform distribution within 20% less and 20% more of that average. Such a configuration permits a physical machine's CPU to become fully occupied when processing service requests. All the services are required to process an equal number of requests (8,000) in the case of full replication, while they are required to process a number of requests that is proportional to the respective number of replicas in the case of partial replication. In all cases, the total simulated real time is 800s, such that the sequence of requests corresponding to each service is uniformly distributed during that period of time. Both, physical machine and virtual machine employ the time-shared scheduling policy. The performance metrics adopted is the idleness of each physical machine's CPU. Such idleness is evaluated according to the requests throughput, varying from one to ten requests per second per physical machine. The service-centered workload balance mechanism is simply implemented by a round robin policy; each set of replicas for a given service is structured as a circular queue. The implemented physical-machine-centered workload balance mechanism is based on the Active Monitoring Algorithm: it selects a replica to forward a request according to the current amount of pending work on each corresponding physical machine; the preference is for the replica placed on the least occupied physical machine.

The performance results for full replication are shown in Figure 6 and 7, while Figure 8 and 9 show the results for partial replication. When the service-centered approach is employed for full replication (Figure 6), all physical machines usage are very well balanced. As expected, the idleness comes to zero when the requests throughput equals the physical

machine power (2,660 MIPS or two requests per second). When the physical-machine-centered approach is employed for full replication (Figure 7), a certain degree of unbalance can be noticed, and also the average idleness is slightly higher, thus implying a worse average response time. When the service-centered approach is employed for partial replication (Figure 8), a very high unbalance and idleness average can be noticed. In real-world systems, virtual machine migration is usually employed in order to overcome that problem. When the physical-machine-centered approach is employed for partial replication (Figure 9), the perceived unbalance and idleness average are much lower. Actually, the unbalance in the physical-machine-centered approach for partial replication is not that far from the unbalance noticed in the case of service-centered approach for full replication.

Figure 6 - Service-centered approach performance for full replication.

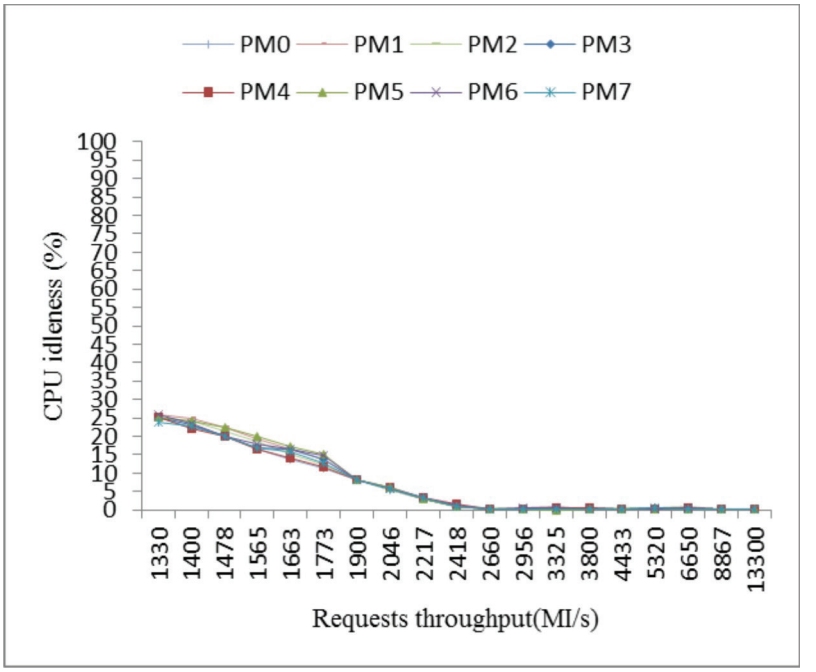


Figure 7 - Physical-machine-centered approach performance for full replication.

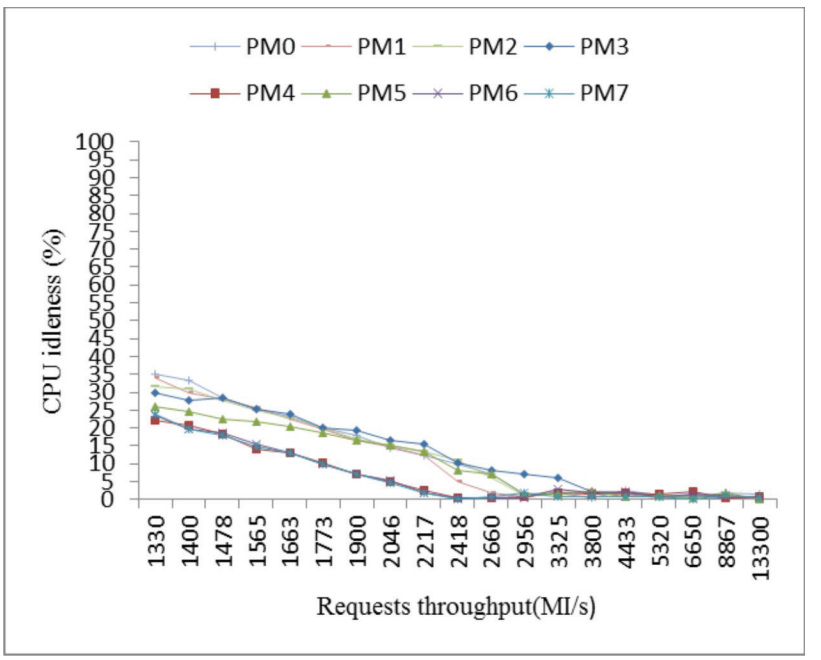


Figure 8 - Service-centered approach performance for partial replication.

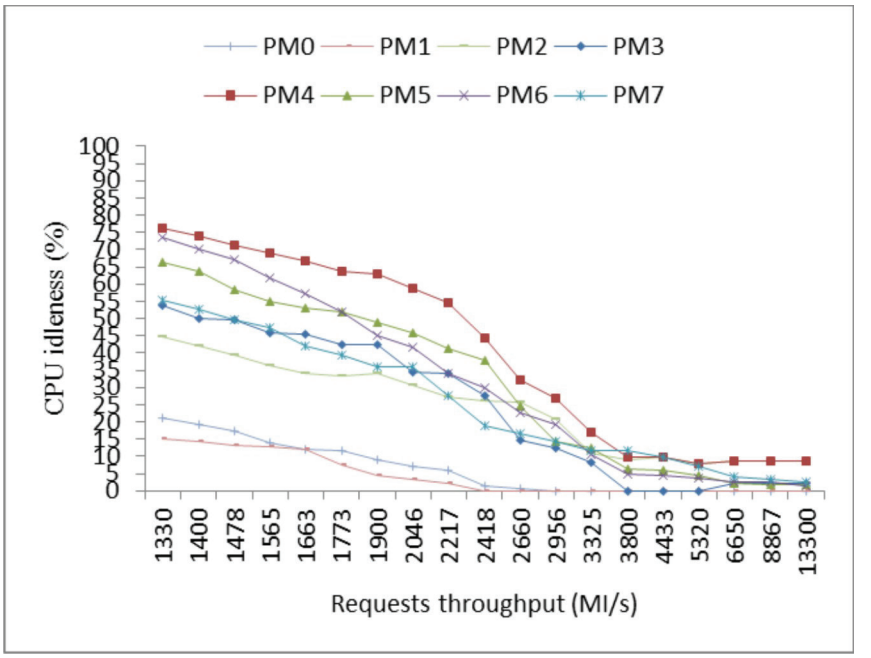
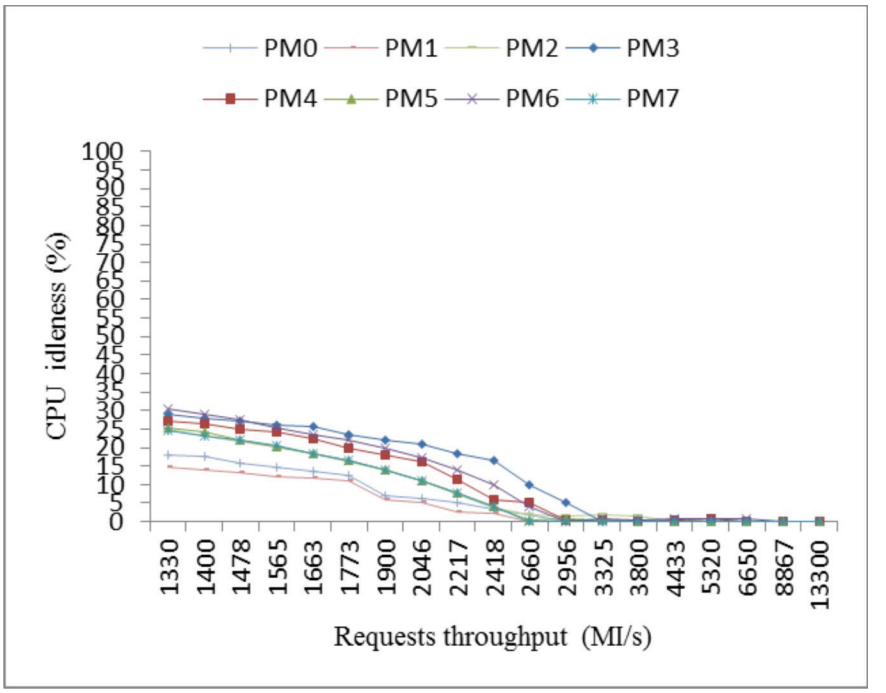


Figure 9 - Physical-machine-centered approach performance for partial replication.



Therefore, the experiment proved that the service-centered approach combined with full replication provides the best possible performance results. In addition, it points out that the physical-machine approach is much more appropriate for real-world service replication, that is, for the case of partial replication.

6 Conclusion

The problem of workload balance in cloud-replicated services is very complex because an efficient solution should consider many interrelated system design issues and parameters, including the replication strategy and the replica

selection approach. The experiments can be further extended to accomplish for other algorithms and the influence of other parameters. Also, they can be improved to detail the average response time per service type, including service-specific request patterns. Finally, a load balance mechanism based on the concept of Virtual Magnetic Fields, proposed in (L. LIMA, A. CALSAVARA, 2010) and (A. CALSAVARA, L. LIMA) can be exploited as an alternative implementation of the Active Monitoring Algorithm.

References

BARROSO, L. A.; HOLZLE, U. *The datacenter as a computer: an introduction to the design of warehouse-scale machines*: synthesis lectures on computer architecture. 2.ed. Wisconsin: C. Morgan & Claypool Publishers, 2009. 156p.

BUYYA, R. *et al.* , Cloudsim: a novel framework for modeling and simulation of cloud computing infrastructures and services, Technical Report, GRIDS-TR-2*009-1, Grid Computing and Distributed Systems Laboratory, the University of Melbourne*, *Australia,* v.1, n.1*, p. 1-9,* mar, 2009.

CALSAVARA, A.; lima, Autonomic application-level message delivery using virtual magnetic fields. J*ournal of network and systems management,* New York, v. 18, n.4*,* p. 97-116, mar, 2010.

CALSAVARA, A.; LIMA JR, L. A. Routing based on message attraction. In: Sixth International Symposium on Frontiers of Information Systems and Network Applications - FINA, 2010, Perth, Australia. Proceedings of the IEEE 24th International Conference on Advanced Information Networking and Applications Workshops - AINA 2010, 2010. p. 189-194.

CITRIX. *Load Balancing Methods*. Available: <http://community.citrix.com/display/cdn/Load+Balancing+Methods>. Accessed in: 04 out. 2011.

LEINBERGER, W.; KARYPIS, G.; KUMA, V. Load balancing across near-homogeneous multi-resource servers, Heterogeneous Computing Workshop, 2000. *(HCW 2000) Proceedings. 9th* , Cancun: IEEE, p. 60-71, 2000. doi: 10.1109/HCW.2000.843733

MARCEL, B. Data Centers: nova norma define hierarquia e orienta usuários, *Cabling News*, São Paulo, v.1, n. 31, p.20-20, jan. 2006.

MELL, P.; GRANCE, T. The nist definition of cloud computing. *Computer Security*. Gaithersburg: NIST - National Institute of Standards and Technology, Special Publication 800-145, p. 1-7, set., 2011.

NUSRAT, P.; AGARWAL, A.; RASTOGI, R. Round robin approach for vm load balancing algorithm in cloud computing environment*, Internation journal of advanced research in computer science and software enginnering,* India, n. 5, v. 4, p. 34-39, mayo, 2014.

RAY, S.; SARKAR, A. Execution analysis of load balancing algorithms in cloud computing environment, *International journal on cloud computing:* services and architecture, Atlanta*,* v.2, n.5, p. 1-13*,* oct, 2012.

SHARMA, S.; SINGH, S.; SHARMA, M. Performance analysis of load balancing algorithms, W*ord academy of science, engineering and technology*, Riverside, v. 38, n. 1, p. 269-272, 2008.

TAURION, C. *Cloud computing: computação em nuvem: transformando o mundo da tecnologia da informação*. Rio de Janeiro: Brasport,, 2009.

VAQUERO, L. M. et al. A break in the clouds: towards a cloud definition, *SIGCOMM Comput. Commun,* Waterloo, v. 39, n.1, p. 51-55, 2009.

ZHONG, X.; RONG, H. Performance study of load balancing algorithms in distributed web server systems. *CS213 Parallel And Distributed Processing Project Report,* Riverside , v. 14, n. 1, p. 1-17, 2009.

Sobre os autores

César Angonese

Ciência da computação pela Universidade do Vale do Rio dos Sinos(1990) – UNISINOS. Mestrando na Pontifícia Universidade Católica do Paraná (PUCPR), Professor da Universidade tecnológica Federal do Paraná (UTFPR).

Alcides Calsavara

Ciência da Computação pela Universidade Estadual de Campinas (1985) e doutorado em Computer Science pela University of Newcastle upon Tyne, Grã-Bretanha (1996). Atualmente é professor titular da Pontifícia Universidade Católica do Paraná e membro do Programa de Pós-Graduação em Informática.

Luiz Augusto de Paula Lima Jr.

Ciência da Computação pela Universidade de São Paulo, mestrado em Ciência da Computação pela Universidade Estadual de Campinas (1994) e doutorado em Ciência da Computação - Université D'Evry - Val D'Essonne/Institut National des Télécommunications. Atualmente é professor titular da Pontifícia Universidade Católica do Paraná e membro do Programa de Pós-Graduação em Informática.