Distributed Adaptive VoIP Load Balancing in Hybrid Clouds*

Jorge M. Cortés-Mendoza [1], Andrei Tchernykh [1†], Alexander Yu. Drozdov [2], Pascal Bouvry[3], Ana-Maria Simionovici[3], Arutyun Avetisyan [4]

CICESE Research Center[1], Moscow Institute of Physics and Technology[2], University of Luxembourg[3], ISP RAS[4]

Cloud computing as a powerful economic stimulus widely being adopted by many companies. However, the management of cloud infrastructure is a challenging task. Reliability, security, quality of service, and cost-efficiency are important issues in these systems. They require resource optimization at multiple layers of the infrastructure and applications. The complexity of cloud computing systems makes infeasible the optimal resource allocation, especially in presence of uncertainty of very dynamic and unpredictable environment. Hence, load balancing algorithms are a fundamental part of the research in cloud computing. We formulate the problem of load balancing in distributed computer environments and review several algorithms. The goal is to understand the main characteristics of dynamic load balancing algorithms and how they can be adapted for the domain of VoIP computations on hybrid clouds. We conclude by showing how none of these works directly addresses the problem space of the considered problem, but do provide a valuable basis for our work.

1. Introduction

Cloud computing is a pay-per-use model for enabling on-demand computing resources. It is defined as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [1].

In on-demand computing, resources are made available to the user as needed and they provide easy way to keep and retrieve data and files. A generic structure and components of clouds is commonly described as a three-layered concept, each layer treated as a service.

Infrastructure as a Service (IaaS) involves offering computational resources that include processing, disk storage, network and other computational resources, and users have control over the software, storage and processing capacity. Platform as a Service (PaaS) involves offering a development platform on the cloud, the user deploys applications and possible configuration settings for the application. Software as a Service (SaaS) includes the software offered by the cloud. The user rents software applications running on a demand infrastructure and can only change the application configuration settings.

Cloud computing considers four deployment models based on locations of the clouds: Public, Private, Community, and Hybrid. Public Clouds are managed by their providers. The infrastructure is shared between organizations, users grant access to the resources through subscriptions. Private Clouds can be accessed only by the provider of the resources; the cloud is fully owned by a single company who has total control over the applications running on the infrastructure. Hybrid Cloud combines private and public cloud, user applications run on a private or public infrastructure (applications with relative importance are scheduled in private clouds). Community Clouds allow sharing infrastructure between organizations with common concerns or similar polices.

We address five important characteristics of cloud computing: cost, performance, scalability, mobility, and virtualization. (Cost) In cloud computing, the resources do not belong to users, and the users do not have to buy or maintain them, the initial investment is not needed. (Performance) The goal is to improve the processing power or maximize storage capacity by consolidation of CPUs, memory,

* This work is partially supported by the Ministry of Education and Science of Russian Federation under contract No02.G25.31.0061 12/02/2013 (Government Regulation No 218 from 09/04/2010).
† Corresponding author

and storage to deploy services. (Scalability) The user can increase or decrease resources (storage, CPUs, memory, etc.) at any time, he only pays for what he is really used. (Mobility) The data can be accessed anytime anywhere, the user only needs a device with internet connection (laptop, smartphone, etc.). (Virtualization) It is a technology used to share resources. It makes a single physical resource appear as many individually separate virtual resources. It allows the use of the server capacity effectively, reducing unused CPU cycles, and minimizing wasted energy.

The management of the large-scale cloud infrastructure is a challenge. Resource management for clouds has been subject to research and development for many years. Here, we discuss different aspects related with our study.

Load balancing is a mechanism to improve the overall performance of the system by distribution of the workload between the nodes. By reducing idle times, providers can improve the profit and achieve a higher user satisfaction. Proper load balancing can help on utilizing the available resources, thereby, minimizing the resource consumption [2]. It also helps in enabling scalability, avoiding bottlenecks and over-provisioning, reducing response time, and energy consumption.

Quality of service (QoS). In cloud computing, providers need to ensure that sufficient amount of resources are provisioned to ensure that QoS requirements of cloud service consumers such as deadline, response time, and budget constraints are met [3, 19]. Service Level Agreements (SLAs) are binding contracts between a service provider and the user.

These SLAs contain the list of services, metrics, responsibilities of the provider and auditing mechanisms. Any violation will lead to a penalty. Several ways exist to provide QoS: scheduling, admission control, traffic control, dynamic resource provisioning, etc.

Energy management. Energy consumption is determined by hardware efficiency, resource management system deployed on the infrastructure, and the efficiency of applications running on the system. The efficiency is very important due to its impacts to users in terms of resource usage costs, which are typically determined by the total cost of ownership incurred by a resource provider [4]. The goal is to avoid utilization of more resources than is required by the application. One solution to this problem is to migrate virtual machines for one node to another and shut down the idle nodes. Such a Dynamic Component Deactivation (DCD) policy [5, 18] switches off parts of the computer system that are not utilized.

The paper is structured as follow. The next section presents several important issues of dynamic load balancing algorithms. Section 3 provides details of the Internet telephony and voice over IP telephony (VoIP). Section 4 describes VoIP provider model and quality of service (QoS). Section 5 presents more formal definition (jobs, cloud infrastructure and criteria). Section 5 gives a brief overview of the load balancing algorithms in cloud computer environments. Section 6 presents our load balancing algorithm. Section 7 concludes the paper.

2. Load balancing

Load balancing is a job distribution decision-making process used in many production systems and computing. It is widely known as a technique for the efficient utilization of resources, and it can be implemented with hardware and software support. Jobs arrival rate, communication delay, the variability of the job parameters, and other factors affect the performance of the systems, to deal with such complex factors it is essential to design efficient and scalable load balancing algorithms. It helps in implementing fail-over, scalability, avoiding bottlenecks, over-provisioning, reducing response time, reducing energy consumption, etc. [6]

Load balancing of services, computational jobs, virtual machines, virtual storages, database requests, and VoIP traffic on the network are identified as a major concern for the efficient use of cloud computing.

The development of an effective dynamic load balancing algorithm involves many important issues: load estimation, load levels comparison, performance indices, system stability, amount of information exchanged among nodes, job resource requirements estimation, job selection for transfer, remote nodes selection, etc. [7]. Important aspects of the problem are distribution of the nodes, storage replications, and virtual machine migrations.

Distribution of nodes. Some algorithms are efficient only if the nodes are closely located and the communication delays are negligible. However, it is necessary to consider communication delay in the cloud infrastructure.

Storage replications. Full replication of data increases the storage and communication overhead. Partial replication saves information in different nodes (with overlap), so that, the utilization, fault tolerance and availability of data are increased.

Virtual machine migrations. A heavily loaded node can migrate its virtual machines (VM) to reduce the overload, but determinate which VMs have to be moved, the destination node, and the profit of the migration are questions difficult to answer.

3. Internet telephony

The Internet telephony (VoIP–Voice over Internet Protocol) refers to the provisioning of communication services over the Internet, rather than via the traditional telephone network. VoIP services significantly reduce calling rates, leading VoIP vendors continue offering extraordinary service using modern cloud technology. Selection of a cloud based VoIP further reduces resource costs, adds new features and capabilities, provides easier implementations, uniform deployments, and integrates services that are dynamically scalable.

To deploy and manage effective telephony tools via clouds a variety of factors need to be improved. The most important one is the utilization of the infrastructure.

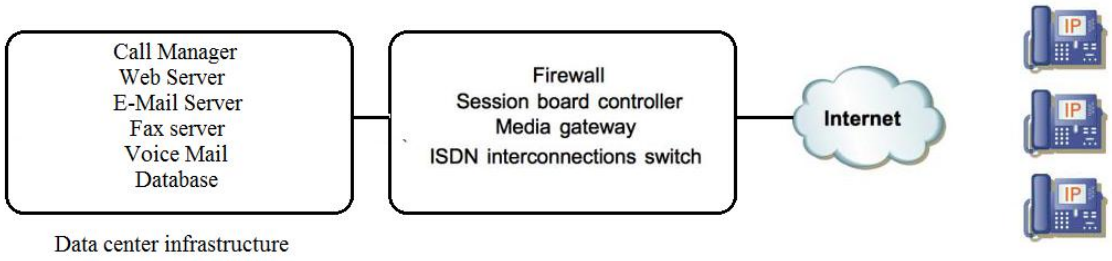


Fig. 1. VoIP architecture.

Fig. 1 shows the general VoIP architecture. The elements of communication infrastructure connect phones remotely through the Internet. The servers use software to emulate a telephone exchange. A drawback of this architecture arises when the hardware reaches its maximum amount of connections.

Traditional VoIP solutions are not very scalable. It is necessary to duplicate infrastructure or replace existing physical hardware.

Fig. 2 shows a cloud based solution of the problem. The voice nodes are virtual machines that execute a variety of services (call transfer, voice mail, music on hold, etc.). The advantage of this architecture is the increasing scalability.

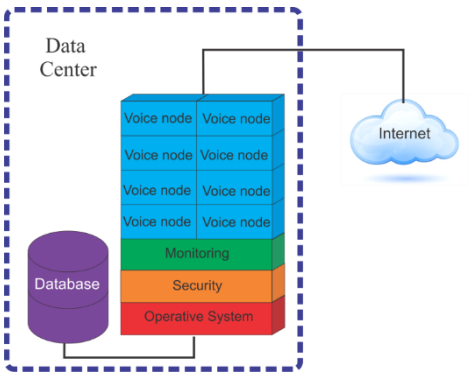
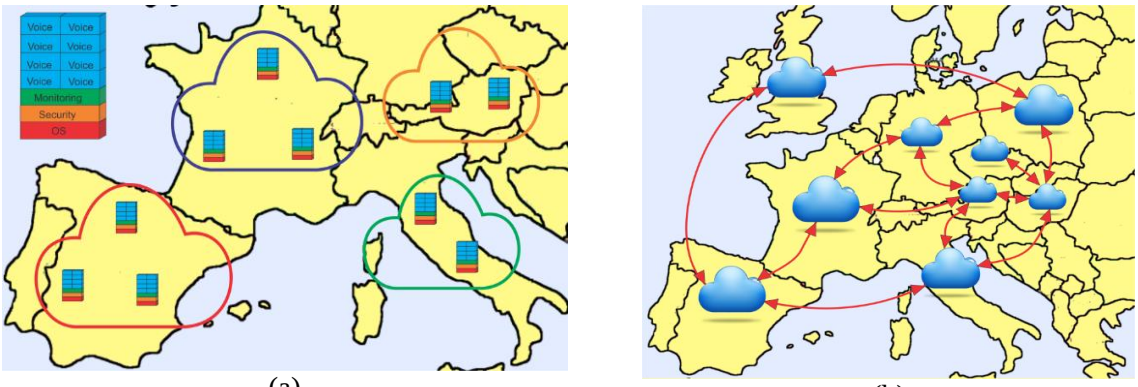


Fig. 2. Single cloud VoiP architecture.

Fig. 3 shows next step in distributed cloud based VoIP architecture, where voice nodes are grouped in data centers geographically distributed.



(a) (b)

Fig. 3. Data centers in clouds (a) and the cloud federation (b).

However, it has several unsolved problems. To optimize the overall system performance, a processor load of the voice signal processing over IP (jobs) should be balanced. The overload of a processor reduces the quality of the call. A similar problem could happen with the network capability. Further, the processor idle time increases the useless expenses of the provider.

Load-balancing maximizes VoIP performance by keeping processor idle time and interprocessor communication overhead as low as possible. To minimize the overall computation time, all processors should contain the same amount of computational work.

It is necessary to design a multi-level distributed VoIP load balancer to improve the local load imbalance in data centers, and new techniques to scale on federation of data centers (Figure 4).

The most important cause of load imbalance in VoIP is the dynamic nature of the problem over time (in both computational and communication costs). Other causes may include the interference from other users in time-sharing mode, the migration process, the time arrival, variability on the utilization process, etc.

Most load balancing algorithms focus on deterministic environments assuming knowledge of the user jobs and system parameters. In general, it is impossible to get exact knowledge about the system. Parameters like processor speed, number of available processors, and actual bandwidth are changing over the time. However, load balancing algorithms should search how to improve resources and ensure Quality of Service (QoS).

4. Cloud provider model

An important factor of VoIP on cloud computing is the rental time of the infrastructure. The VoIP providers rent the resources during a time interval, this factor affects the incoming of the providers and even can increase the cost of the service, for this reason is important to consider when a new resources are rented and the rented time interval.

Fig. 4 shows an example of VMs that provide VoIP services. This example consider the load balancing of the utilization on the VMs. In Figure 5a, the VoIP provider needs three VMs to deal with the load of time during the day. VM_2 starts its execution when VM_1 cannot process all calls in the system. During the rental time of VM_2, its utilization is low, in this case the provider has to paid for the rent of three VMs. Figure 5b shows the same example, but in this case the load balancer distributes the arrival calls between VM_1 and VM_2. When VM_1 rental time is almost completed, a consolidation technique is used to reduce VM_1 utilization (number of call running on VM_1). This approach helps providers to reduce the number of VMs for calls processing.

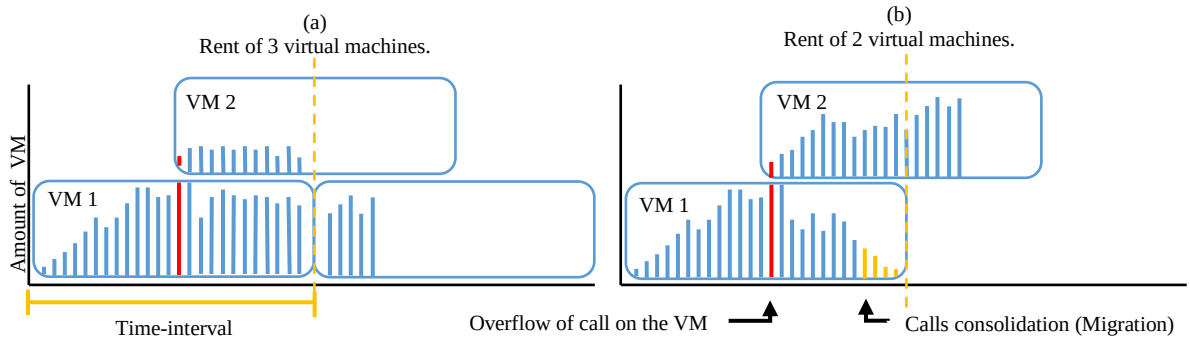


Fig. 4. VoIP load balancing.

VoIP providers also have to guarantee QoS to the users. Quality of service requirements for VoIP are very important but several factor can affect the quality of calls. The quality degradation is determined by the transit of the packets across the Internet, queuing delays at the routers, packet travel time from source to destination, jitter (deviations of the packet inter-arrivals), packet loss, call set-up time, and call tear-down time, etc. Processor utilization of 100% provides the best expected performance. However, with increasing number of call, hence utilization, CPU cannot be able to handle the stress anymore and jitters and broken audio symptoms will appear (Fig. 5).

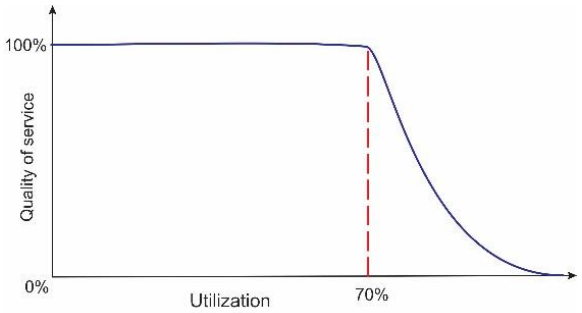


Fig. 5. Voice quality versus processor load (utilization).

Fig. 6 shows an example of VM rental to provide VoIP services with QoS. In Figure 6a, the VoIP provider needs three VMs to deal with the load during a day. In this case, the QoS is not considered as independent optimization criteria. The utilization of VM_1 is above the utilization threshold. Figure 6b shows the same example. However, load balancing guarantees the QoS by maintaining the VMs utilization under the utilization threshold. Moreover, it reduces the amount of VMs to process calls.

This model allows providers to deploy services in different countries by renting infrastructure in public or/and private clouds providers.

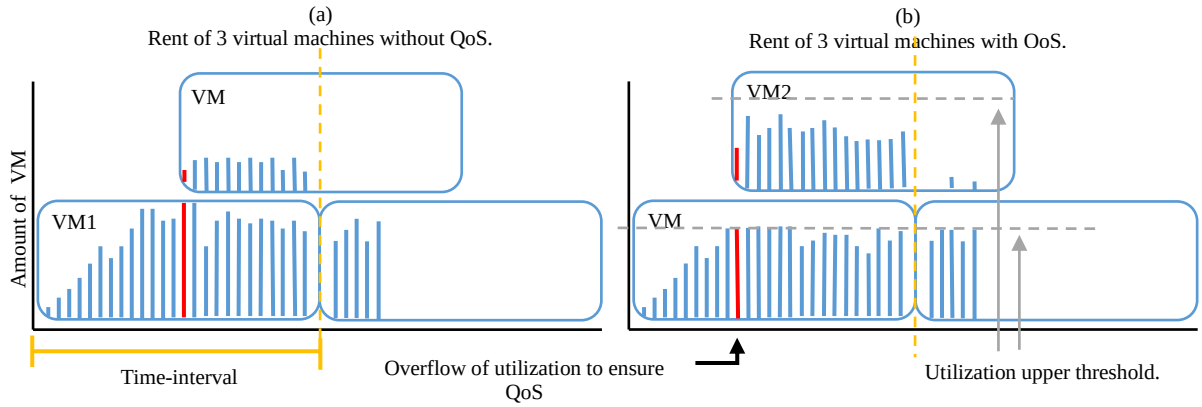


Fig. 6. VoIP with Quality of service.

5. Formal definition

We address load balancing problem in the hierarchical federated cloud environment, where clouds of different providers collaborate to be able to fulfill requests during peak demands. We assume heterogeneous clouds and data centers with different number of cores, execution speed, energy efficiency, amount of memory, bandwidth, etc.

5.1 Infrastructure model

Let us consider cloud C that consists of m nodes (data centers, sites) $D_1, D_2, \dots, D_m$. Each node D_i, for all $\mathrm{i} = 1..\mathrm{m}$, consists of b_i servers (blades, boards) and p_i processors per board. We assume that processors in the data center are identical and have the same number of cores. Let m_i be the number of identical cores of one processor in D_i.

We denote the total number of cores belonging to the data center D_i by $\bar{m}_i = b_i * p_i * m_i$, and belonging to all data centers of the cloud C by $\bar{\mathrm{m}} = \sum_{\mathrm{i}=1}^{m} \bar{m}_i$.

The processor of data center D_i is described by a tuple $\{\mathrm{m_i},\ \mathrm{s_i},\ \mathrm{mem_i},\ \mathrm{band_i},\ \mathrm{eff_i}\ \}$, where $\mathrm{s_i}$ is a measure of instruction execution speed (MIPS), $\mathrm{mem_i}$ is the amount of memory (MB), $\mathrm{band_i}$ is the available bandwidth (Mbps), and $\mathrm{eff_i}$ is energy efficiency (MIPS per watt).We assume that data centers have enough resources to execute any job but their resources are limited.

A data center contains a set of routers and switches that transport traffic between the servers and to the outside world. They are characterized by the amount of traffic flowing through it (Mbps). A switch connects a redistribution point or computational nodes. The connections of the processors are static but their utilization change, overload can occur due to a large amount of I/O being pushed through it. The interconnection network architecture is Three-tier data center architectures that include: access, aggregation, and core layers. The interconnection between clouds will be done through public Internet.

In addition, to satisfy requests during the peak demands that exceed the capacity of the cloud C, it collaborates with k external independent clouds (sites) $\mathrm{C}_1, \mathrm{C}_2, \dots, \mathrm{C}_\mathrm{k}$.

5.2 Job model

We consider n independent jobs $J_1, J_2, \dots, J_n$ that must be scheduled on federation of clouds. The job (call) J_j is described by a tuple $\{\ r_j,\ p_j, u_j\ \}$ that consist of: its release time $r_j \geq 0$, p_j the duration of the job and the contribution to the processor utilization of the job j. The release time of a job is not available before the job is submitted, the duration of the job is unknown until the job has completed its execution, and contribution is a constant for a given job. Due to the virtualization technique and resource sharing, the resources are constantly changing. A job can be allocated to one cloud only. Jobs submitted to the one cloud can be migrated to another one.

The admissible set of data centers for a job J_j is defined as a set of indexes $\{\ a_1^j, \dots, a_l^j\ \}$ of data centers that can be used for migration of the job.

5.3 Optimization criteria

In this paper, two criteria are considered for the model: the billing hours for VMs to provide a service, and their utilization to increase quality of service.

In order to optimize VoIP cloud solutions, we use metrics that are useful for systems with VMs. They have to allow the provider to measure the cost of the system in terms of number of demanded VMs and time of their using. These metrics enable us to compare different load balancing policies: total VM rented (VM_{rented}) (number of billing hours) and utilization of VM (VM_{util}).

VM_{rented} allows providers to measure the cost of the system in terms of parameters that helps him to establish utility margins.

In general, QoS standards for VoIP traffic are set for voice. One of the possible generalizations of the voice quality is processor utilization. Each codec provides a certain quality of speech only if processor utilization is low enough in order to ensure QoS. If the number of calls is increased, and utiliza-

tion is close to 100%, CPU cannot be able to handle the stress anymore and reduced audio quality will appear. Hence, we use VM_{util} as a way of measure the user satisfaction for the service.

6. Related work

Previously, we briefly overview the load balancing algorithms in different computer environments [8]. Now, we consider the advances in the field of cloud computing. Table I presents the main characteristics of the algorithms, and metrics used to study their quality.

HBB -LB - Honeybee Foraging Behavior [9]. It is a dynamic load balancing algorithm for scheduling of tasks in cloud computing environment. The proposed algorithm balances the priority of tasks on the machines to minimize the waiting time of the tasks in the queue. The tasks (honey bees) are removed from the over loaded VM, they update the number of priority tasks and load of VMs, this will help tasks to choose a VM based on load and priorities. Whenever a high priority task has to be submitted to other VMs, it should consider the VM that has minimum number of high priority tasks so that the particular task will be executed at the earliest.

DT - PALB - Power Aware Load Balancing [10]. A new version of the algorithm Power Aware Load Balancing named Double Threshold PALB uses the migration of VMs to minimize the energy consumption in the system. When the utilization of a node is under 25% (lower threshold), the load balancer migrates workload (VMs) of the node to reduce its utilization to zero and shut down the node.

TBSLB-PSO - Particle Swarm Optimization [11]. It is a load balancing mechanism for cloud computing. This algorithm uses Central Task Scheduler (CTS) to transfer extra tasks from an overloaded VM to a new similar VM by applying the information on a blackboard; the blackboard contains all cloud schedulers about VM features, executing tasks and Quality of Service (QoS). The migration process considers the amount of data, memory, bandwidth and numbers of CPU of VM. Idle Physical Machines (PM) will not be chosen as the new PM hosts, this allows to decrease energy consumption.

LBS-BF - Load Balance Scheduling Based on firefly [12]. It is a load balancing mechanism for cloud computing. The load balancer computes a load index for resources shared and it will be initiated to effectively use the resources dynamically. The fireflies attraction is linked to objective function and monotonic decay of the attractiveness with distance, it (in the firefly algorithm) helps to generate scheduling index and the distance calculation serves to find the closely associated nodes in the cloud network. The technic proposed three parameters for the algorithm: attraction between the nodes and the request, the scheduling index, and the distance between nodes. This parameters consider the CPU rate, memory rate, processing time and the loads to the nodes.

A2LB - Autonomous Agent Based Load Balancing Algorithm [13]. It is a dynamic load balancing algorithm for cloud environment. The load balancer mechanism comprises of three agents: Load agent (LA), Channel agent (CA) and Migration agent (MA). LA is a static agent and it controls information policy and maintains all detail of a datacenter, it most important job is to calculate the load on every available VM. CA is a static agent, it controls the transfer policy, selection policy and location policy. Finally, MA is an ant (special category of mobile agents), it will move to other datacenters and communicate to enquire the status of VMs.

GA – Genetic Algorithm based on load balancing [14]. It is a load balancing algorithm for cloud computing. It uses a binary representation for the chromosomes, a random single point crossover, and a mutation with probability of 0.05. This algorithm considers an estimate of penalty (delay cost), the amount of money that cloud service provider needs to pay to customer when job finishing time being more than the deadline advertised by the service provider

Table 1. Load balancing algorithms.

	Characteristics												**Metrics**								
	Centralized	Decentralized	Hierarchical	On line	Off line	Non-clairvoyant	QoS	Migration cost	Static	Dynamic	Replication	Migration	Utilization	Performance	Overhead	Response time	Scalability	Fault tolerant	Throughput	Energy consump-	Cost

HBB-LB	•			•					•		Task		•		•			•		
DT-PALB	•			•					•		VM	•							•	
TBSLB-PSO	•				•			•	•		Task	•							•	
LBS-BF									•			•			•					
A2LB			•	•					•						•					
GA	•			•			•		•			•			•					•
FFA-DLB	•			•					•		Task	•								•

FFA-DLB - Fuzzy-based Firefly Algorithm for Dynamic Load Balancing in Cloud Computing [15]. It is a dynamic load balancing in Cloud computing environment and it is a combination of Firefly algorithm with the fuzzy logic, this algorithm separate the cloud based on the frequent node allocation to balance the load across the variety of partitions. The goal is to separate the hotspots and least loaded nodes, then classify the nodes into groups (like lightly loaded, normal, and heavily loaded). The set of tasks enter into the load balancer after the partition of the cloud. This algorithm consider a balancing factor based on the parameters of the VM and the files to be processed from input. The fuzzy inference engine determine to assign the tasks, with a condition that already assigned tasks are migrated only when a high necessity arises.

7. Adaptive load balancing algorithm

Campos and Scherson proposed a dynamic distributed load balancing algorithm named Rate of Change (RoC-LB) [16]. The balancer (Bal) makes job distribution decisions at runtime, locally and asynchronously. Each Bal considers its own load; migration does not depend on the load of other Bals. The migration decision depends on current load, load changes in the time interval (Rate of Change), and current load balancing parameters.

To define WHEN load balancing should be started the algorithm considers three thresholds: Ub upper bound, Lb lower bound, and Cb critical bound. If the load is larger than Ub then the Bal is considered as a source of the load and can satisfy job requests. If the predicted load is less than Lb the Bal is considered as a sink. When the current load is between these two bounds, the Bal is in the neutral state. However, if the load or the predicted load fall below Cb, the Bal immediately initiates a request for a load.

We extend this algorithm and design a VM-Aware Adaptive Rate of Change (VMA-AdRoC) algorithm that is based on an adaptive decision policy and virtual machine utilization optimization. Adaptability is essential for the efficient use of cloud infrastructure. Clouds differ from previous computing environments in the way that they introduce a continuous uncertainty into the computational process. The uncertainty becomes the main feature of the cloud computing and the principal difficulty of the efficient resource management [17].

There are several major sources of uncertainty: dynamic elasticity, dynamic performance changing, virtualization, loosely coupling application to the infrastructure, among many others. A workload in such an environment is not predictable and can be changed dramatically. It is impossible to get exact knowledge about the system. Parameters such as an effective processor speed, number of available processors, and actual bandwidth are changing over the time.

VMA-AdRoC takes into account these uncertainties. The accuracy of each balancing decision depends on the actual cloud characteristics at the moment of balancing. Cloud parameters are changing over time and balancing parameters should be adapted to these changes. This dynamic and adaptive approach can cope with different workloads, and cloud properties. To adjust Ub, Lb, and Cb, the past information within a given time interval can be analyzed to determine an appropriate parameters. This interval should be set according to the dynamics of the system.

Let $u_i(t)$ be the utilization of ith Bal at time t. Let $\Delta_i(t) = (u_i(t) - u_i(t - si))/si$ be the utilization change during sample interval si=$[t - si, t]$. We named it utilization change speed or utilization consumption speed. The sampling time interval si is an adaptive parameter; finer sampling allows detecting the need to balance the system faster, but it generates a larger communication overhead.

Bal uses $\Delta_i(t)$ as a predictor of the future utilization. It can be also used to estimate the number of sampling intervals to reach an idle state. Let $rd_i(t)$ be the response delay of at time t. It is an adaptive

parameter, and it is defined as the time it takes between the initiation of a load request and the reception of load. If the time to reach idle state is less than $rd_i(t)$, then Bal must initiate a migration request. Let us note that si and $u_i(t)$ are independent from others Bals.

Fig. 7 shows possible load balancing scenarios. Solid line shows real utilization, dashed lines are predicted utilization.

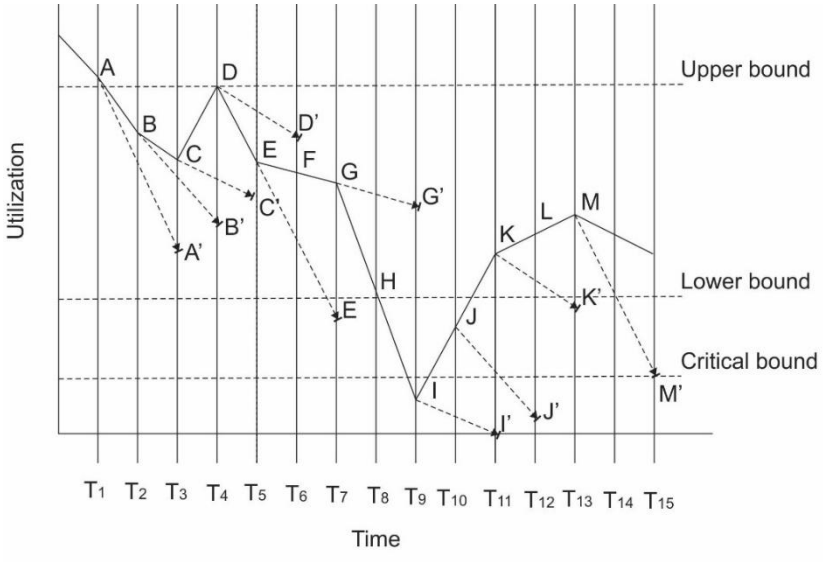


Fig. 7. Dynamic load balancing scenario.

At time T_9 the *Bal* immediately initiates a request for load regardless of the predicted future utilization based upon the estimation $\Delta_i(t)$ value or it initializes migration of the load to other VM. Estimation J' on T_{11} is under Cb but *Bal* cannot initiates a request because the request at T_i is not arrived, a new request only can be generate at T_k. In VMA-AdRoC, unlike RoC-LB, if the utilization is above than Ub then the *Bal* sends jobs to the sinks.

To define WHERE a load is requested from or send to, each Bal keeps two lists. The sink list records Bals that previously needed jobs, and source list enrolls Bals that previously offered jobs. Bal that initiates a request is considered to be a sink. A sink selects a Bal from its source list for a load request, and sends a requesting message. The source can accept the request or broadcast the request to other Bals from its own source list. Bal does not send several load requests at the same time. It has to wait an answer for the first request until it sends another message. The result of this message is the load coming from other *Bal* or the request comes back as unfulfilled.

In our algorithm, each element of the source and sink lists includes not only IP address like in RoC-LB, but utilization $u_i(t)$, of the corresponding Bal. This information is not accurate and updated dynamically. In our case, choosing the sink/source node is a two-parameter problem. In the future work, we consider also requested load $ru_i(t)$, $\Delta_i(t)$, $rd_i(t)$, answer time of the request $rt_i(t)$, admissibility of Bals, etc. The goal is to choose the most adequate compromise solution.

8. Conclusions

In this paper, we formulate and discuss load balancing problem addressing VoIP in cloud computing federation. We define models for VoIP load balancing. We overview the last advances of load balancing in distributed computer environments to understand the main characteristics and requirements of load balancing algorithms. We show that none of these works directly addresses the problem space of the considered problem, but do provide a valuable basis for our study.

In real clouds, the load balancing bounds can be dynamically adjusted to cope with the dynamic workload situation. To this end, the past workload must be analyzed for a certain time interval to determine an appropriate lower and upper bounds. The time interval should be set according to workload characteristics, communication delays, and cloud configurations.

While the scope of this work is the introduction of the adaptive dynamic distributed load balancing algorithm, in future work, we also intend to evaluate the practical performance of the proposed strategy and their derivatives to assess its actual efficiency and effectiveness. To this end, we plan simulations using real VoIP traces and corresponding VoIP cloud configurations. Further, we will compare our approach with other existing strategies which are typically based on round robin approach.

References

1 Mell, P., and Grance, T. (2011). The NIST definition of cloud computing (draft).NIST special publication, 800 (145), 7.

2 García, J. L. G., Yahyapour, R., and Tchernykh, A. (2013). "Load Balancing for Parallel Computations with the Finite Element Method". Computación Y Sistemas, 17(3), 299–316.

3 Buyya, R., Calheiros, R. N., and Li, X. (2012). "Autonomic Cloud computing: Open challenges and architectural elements". In Emerging Applications of Information Technology (EAIT), 2012 Third International Conference on, 3-10. IEEE.

4 Beloglazov, A., Abawajy, J., and Buyya, R. (2012). "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing". Future Generation Computer Systems, 28(5), 755-768.

5 Tchernykh, A., Pecero, J. E., Barrondo, A., and Schaeffer, E. (2014). "Adaptive energy efficient scheduling in Peer-to-Peer desktop grids". Future Generation Computer Systems. Elsevier Science, Volume 36, July 2014, Pages 209–220. DOI: 10.1016/j.future.2013.07.011,

6 Kansal, N. J., and Chana, I. (2012). "Cloud Load Balancing Techniques: A Step Towards Green Computing". IJCSI International Journal of Computer Science Issues, 9(1), 238-246.

7 Alakeel, A. M. (2010). "A guide to dynamic load balancing in distributed computer systems". International Journal of Computer Science and Information Security, 10(6), 153-160.

8 Tchernykh, A., Cortés-Mendoza, J. M., Pecero, J. E., Bouvry, P., and Kliazovich, D. (2014). Adaptive energy efficient distributed VoIP load balancing in federated cloud infrastructure. In Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference on (pp. 27-32). IEEE.

9 Krishna, P. Venkata (2013). "Honey bee behavior inspired load balancing of tasks in cloud computing environments." Applied Soft Computing 13.5: 2292-2303.

10 Adhikari, J., and Patil, S. (2013, July). Double threshold energy aware load balancing in cloud computing. In Computing, Communications and Networking Technologies (ICCCNT), 2013 Fourth International Conference on (pp. 1-6). IEEE.

11 Ramezani, F., Lu, J., and Hussain, F. K. (2014). Task-based system load balancing in cloud computing using particle swarm optimization. International Journal of Parallel Programming, 42(5), 739-754.

12 Florence P., and Shanthi V., (2014). A load balancing model using firefly algorithm in cloud computing. Journal of Computer Science, 10(7), 1156-1165.

13 Singh, A., Juneja, D., and Malhotra, M. (2015). Autonomous Agent Based Load Balancing Algorithm in Cloud Computing. Procedia Computer Science, 45, 832-841.

14 Dasgupta, K., Mandal, B., Dutta, P., Mandal, J. K., and Dam, S. (2013). A genetic algorithm (ga) based load balancing strategy for cloud computing. Procedia Technology, 10, 340-347.

15 Susila, N., Chandramathi, S., and Kishore, R. (2014). A Fuzzy-based Firefly Algorithm for Dynamic Load Balancing in Cloud Computing Environment. Journal of Emerging Technologies in Web Intelligence, 6(4), 435-440.

16 Campos, L. M., and Scherson, I. D. (2000). Rate of change load balancing in distributed and parallel systems. Parallel Computing, 26(9), 1213-1230.

17 Tchernykh, A., Schwiegelsohn, U., Alexandrov, V., and Talbi, E. G. (2015). Towards Understanding Uncertainty in Cloud Computing Resource Provisioning. SPU'2015 - Solving Problems with Uncertainties (3rd Workshop). In conjunction with The 15th International Conference on Computational Science (ICCS 2015), Reykjavík, Iceland, June 1- 3, 2015. Procedia Computer Science, Elsevier, Volume 51, Pages 1772–1781.

18 Tchernykh, A., Lozano, L., Schwiegelshohn, U., Bouvry, P., Pecero, J. E., and Nesmachnow, S., Alexander Yu. Drozdov (2015). Online Bi-Objective Scheduling for IaaS Clouds with Ensuring

Quality of Service. Journal of Grid Computing, Springer-Verlag. DOI 10.1007/s10723-015-9340-0

19 Schwiegelshohn, U., Tchernykh, A.: Online Scheduling for Cloud Computing and Different Service Levels, 26th Int. Parallel and Distributed Processing Symposium, Los Alamitos, CA, pp. 1067–1074 (2012)