A New Design for Smart and Decentralized Approach for Resource Allocation in Cloud Computing using CSP Model

Almutawakel Abdallah[1], Kazar Okba[2], Bali Mouadh[3]
[1, 2] LINFI laboratory, Department of Computer Science
Biskra University, Biskra, Algeria
aboud.aboud2012@gmail.com, kazarokba@yahoo.fr

[3]LIMED Laboratory, Bejaia
University, Algeria, El Oued University
Algeria
bali-mouadh@univeloued.dz

Journal of Digital Information Management

ABSTRACT: *Cloud computing is the latest technology in both hardware and software. The cloud computing architecture offers a number of services including the SAAS, PAAS and IAAS. When customers specify their technical and economic needs, cloud computing helps them satisfy their request at the lowest possible cost and the best quality of resources. This paper presents a new approach for resource allocation (RA) in cloud computing. Its architecture is based on multi-agent system (MAS) and distributed constraint satisfaction problems (DCSP) where variables and constraints are distributed among multiple agents. In MAS, each agent makes his choice via a distributed negotiation to satisfy the global objective which is looking for the best solution to the users' needs. The present work highlights a new approach that is based on hybridizing (DCSP) and (MAS). This decentralized approach is designed to solve a number of problems such as resource allocation and planning cloud-computing systems.*

Subject Categories and Descriptors: [I.2.11 Distributed Artificial Intelligence]; Multi-agent systems: **[K.6.2 Installation Management];** Resource Allocation

General Terms: Multi-agent system, Cloud Computing, Resource Allocation.

Keywords: Cloud Computing, Resource, Multi-Agent System, Problem to Satisfaction of Contraines (CSP), Resource Allocation

Received:23 October 2018, Revised 10 February 2019, Accepted 26 February 2019

Review Metrics: Review Scale- 0/6, Review Score-4.47, Inter-reviewer Consisteny- 71.5%

DOI: 10.6025/jdim/2019/17/4/201-213

1. Introduction

Cloud computing is a type of parallel and distributed system. It consists of a collection of interconnected and virtualized computers which are dynamically presented as one or more unified computing resources [1]. The cloud provides the following services: infrastructure as a service (**IaaS**), platform as a service (**PaaS**) or software as a service (**SaaS**) [2] [3] [4].

In cloud computing environments, the assignment of computational resources is known as resource allocation (RA). Its main goal is to maximize the benefits of the provider to meet the user needs. To obtain an optimized solution for RA is a complex task as it increases exponentially with the increase of the number of exciting resources and services offered [5].

This work aims to introduce an agent paradigm [6] and constraints satisfaction problems (CSA) [7]. The multi-agent based negotiation distributes autonomous problem solvers that can act and collaborate flexibly and self-interestedly among each other [8], [9], [10]. Also, the distributed CSPs (DCSPs) are CSPs in which variables and constraints are distributed between agents [11].

The present work is divided into five main sections. The first section is an introductory section. It defines the cloud and its services and the RA problem and the agent problem. Previous research works as related to the subject of this paper are briefly covered in the second section. Section three is devoted to present our approach, which is followed by experimental results in section four. The final section presents the concluding remarks and future perspectives.

2. Related Works

The importance of software agents as significant tools for recourses autonomous management in the cloud is reviewed in [12]. In the process of achieving service-level agreements (SLAs), drawbacks occur in the selection of suitable CSPs when the proper decision is left to the customers, providers, and proper negotiations. On the other hand, mobility in the cloud is reviewed in [13]. Further the authors propose mobile agents for middleware framework. This sounds fine but it fails to address the same less migration of resources in cloud environment.

Nurmi et al [14] propose a Eucalyptus which is an open-source cloud computing framework for the creation of a virtual machine (VM). The drawback of this work lies in developing resource control in a hierarchical manner.

In [15], Farahnakian et al suggested an architecture based on multi-agent systems for dynamic VM consolidation. The authors split the problem of dynamic consolidation into sub-problems namely host status detection and VM placement optimization. The advantage of this work is to use the reinforcement learning (RL) approach in the optimisation process. Another agent, known as a global agent, has a supervisory role. It receives information from another local agent and takes the decisions for the migration of VMs.

In [16], Beloglazov and Buyya introduced an adaptive heuristic for dynamic consolidations of VMS based on adaptive threshold mechanism.

In [17], authors presented a three-way admission control and schedule mechanism. It involves Cloud users, a SaaS provider and public IaaS to minimize the number of providers' expenditure and improves users experience in utilizing cloud resources.

In [18] Farahnakian et al, offer a VM management framework which is based on multi-agent system (MAS). Their work has made a positive result in reducing Service Level Agreement (SLA) violations and power consumption, but it has a negative result in having agents arranged in hierarchical architecture.

In [19], Garg et al, proposed optimal policies that consider a number of energy efficiency factors which change from the data center to another depending upon location, architectural design, and management system.

The work of You et al. [20] suggested techniques that make use of virtualization space for RA. They introduce the resource allocation strategy based on market (RAS-M) framework. Although the model is based on the market economy, it seeks a redistribution of resources as a function of market prices.

The works of Sriram et al. [21] and Chandak [22] combined the planning of virtual resource with forecasting the use. They use techniques which are based only on instantiating, stopping and migrating virtual nodes. Both works do not exploit the latest possibilities offered by virtual technology such as the modification of virtual machine resources in execution time.

Table 1 sums up the advantages and disadvantages of resource allocation related works. In light of the results shown in the table, we conclude that these works have two common problems: the underutilization and fragmentation of resources. Although authors have proposed certain mechanisms and methods to solve these problems, their works still have some weaknesses (Table 1).

To solve these problems, our approach (in section 3) for RA is based on extending and distributing the resources allocation process at the micro-level in the cloud system and makes hybridization between more techniques to gain more advantages.

3. Our Approach for RA

In this section, we present the design of our approach and give its behaviour.

3.1 System Objectives

In this work, we focus on three challenges of cloud resource allocation:

• **Submission of Customer Cloudlets in VMs:** This process consists of a selection of the Virtual machines (VMs) that allow submitting (run) cloudlets according to their resource requirements (ram, storage, processor, bandwidth, …). We notice that VM could submit more than one cloudlet.

• **Hosting (placing) of VMs on hosts:** Once the Cloudlets have submitted by selected VMs, we have to place every free VM (which has not hosted yet) in the host that provides the least price, low load and available resources.

• **Optimization of Hosted VMs:** Preserves free resources and prevents the overload of the hosts by migrating (replacing) some hosted VMs from the overloaded hosts to under load hosts.

3.2 Resource Allocation System Architecture

The approach of our architecture is shown in Figure 1.

The functions of various layers are:

No	Advantages	Disadvantages	REF
1	Autonomous management of recourses.	Customers and providers select the suitable CSPs and make proper decision.	[12]
2	Mobility in the cloud.	Migration of resources in cloud environment.	[13]
3	Open source platform.	Hierarchical resource control.	[14]
4	Use AI methods such as reinforcement learning (RL) approach.	Some agents need critical information from other local agents.	[15]
5	Positive in reducing Service Level Agreement (SLA) violations and power consumption.	Hierarchical architecture.	[16]
6	Dynamic consolidations of VMS based on Adaptive heuristic.	Add adaptive threshold.	[17]
7	Minimizing of providers expenditure.	Improve users experience in utilizing cloud resources.	[18]
8	Change from data center to another depending upon location.	Hierarchical architecture.	[19]
9	The model was based on market economy.	It seeks a redistribution of resources as function of market prices.	[20]
10	It combines the planning of virtual resource with forecasting the use.	Does not exploit the modification of VM resources in execution time.	[21] ,[22]

Table 1. Summary for the advantage and disadvantages RA works

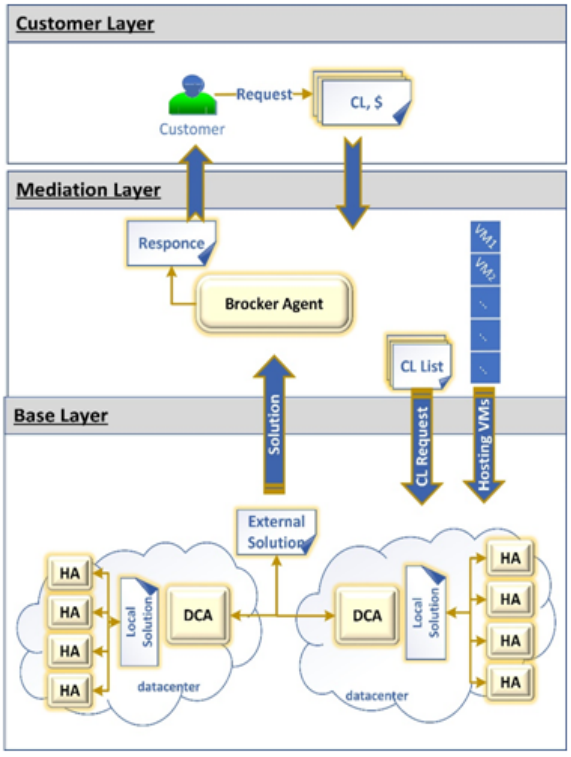

Figure 1. Resource allocation system architecture

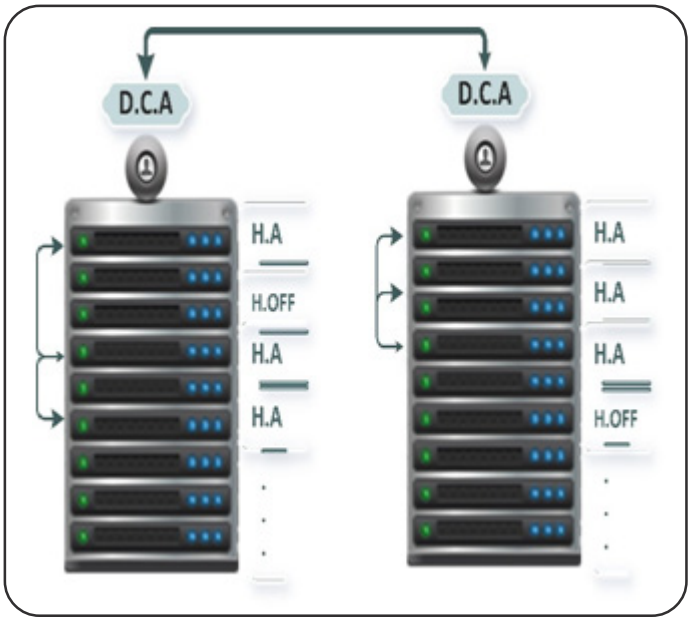

Figure 2. Relationships between DCA and HA

Customer Layer:
In this layer, the system focus to cloud customers and their requests, which are cloudlet and its budget (CL, $).The response can be one of three choices: ***Yes***, ***No*** or ***Negotiable*** (the cost solution is more than the budget). In the third choice, the system negotiates with the client to increase his cloudlet's budget.

Mediation Layer:
This layer contains a free VMs list. The broker agent (BA) manages the use, performance and delivery of cloud resources. The layer functions as a mediation between Baselayer and cloud consumers.

Base Layer:
It contains data center agents (DCA), Host agents (HA) and agents(HOffA: for any host in-state OFF). In this layer, the main processes are Ordonnance and Optimization.

The ordonnance operation is looking for the best resource allocation for the customer request in two levels:

Local (between HA agents in the same datacenter) and External (between DCA agents of the cloud). Further, the optimization operation aims at optimizing the VMs hosting in physical hosts to reduce the runtime and power consumption.

The relationship between DCA and HA agent is shown in Figure 2:

3.3 DCSP Modulation

In this section, the DCSP problem is formulated. Variables and constraints are distributed to multiple agents. Each agent in MAS makes its proposal plan (solution) by using distributed negotiation and satisfying its constraints. Details on the main ideas of the approach are discussed in the next sections.

The various variables and constraints are identified. After that, Scenario of computing is painted accordingly.

3.3.1 Define Variables

Variable	Definition	Domain
D	Number of Datacenters	Naturel Number
H	Number of the hosts in Datacenter	Naturel Number
P	Number of Processors in the host	Naturel Number
V	Number of virtual machines	Naturel Number
V_j	set of VMs hosted in the host *J*.	$\{V_1,\ldots,V_j,\ldots V_v\}$
SV_j	Sub set of VMs hosted (in the host *J)* selected to execute a set of cloudlets.	$\{SV_1,\ldots,SV_j,\ldots SV_v\}$
$MigV_{jj'}$	Sub set of VMs hosted (in the host *J)* selected to be migrated to the host *J'*.	$\{V_1,\ldots,V_j,\ldots V_v\}$

C	Number of cloudlets	Naturel Number
DC	Datacenter	$\{dc_1, \dots, dc_i, \dots dc_d\}$
Host	Host_j in DC_i	$\{Host_{11}, \dots, Host_{ij}, \dots Host_{dh}\}$
PE	Processor in the host	$\{Pe_{11}, \dots, pe_{jk}, \dots pe_{hp}\}$
VM	Virtual machine	$\{vm_1, \dots, vm_l, \dots vm_v\}$
PEV	Processor in Virtual machine	$\{Pev_{11}, \dots, pev_{lk}, \dots pev_{vp}\}$
CL	Cloudlet	$\{cl_1, \dots, cl_m, \dots cl_c\}$
Host (ram)	Size of ram of the host	Naturel Number
Host(bw)	Bandwidth of the host	Real number
Host(Storage)	Size off the hard disc of the host	Naturel Number
Host(nbr_pe)	Number of Processors in the host	Naturel Number
Host(state)	state of the host	{ON, OFF}
Host(mips)	Sum of Capacities of Processors of the host	Real number
Host(used_mip)	Sum of Capacities of the Processors used by virtual machines hosted in the host	Real number
Host(mips_load)	The energy of the host, the Capacity of Processors used in accordance to the total capacity of Processors of the host.	Real number (%)
Host(Cost_ON)	Cost of initial exploitation of the host	Naturel Number
Host(Cost)	Cost of the host: Cost total of the Capacities of the Processors used in the host.	Naturel Number
Host(oldCost)	Cost of the host before the optimization	Naturel Number
Host(newCost)	Cost of the host after the optimization	Naturel Number
Host(mips_price)	Unit price of mips in the host	Real number, obtained from proposed model for every host
Pe(mips)	Capacity of the Processor	Real number
VM(size)	V? hard disc Size	Naturel Number
VM(ram)	Size of the ram	Naturel Number
VM(bw)	Bandwidth of the VM	Real number
VM(nbr_pe)	Number of Processors in VM	Naturel Number
VM(mips)	Sum of Capacities of the Processors of the VM	Real number
CL(length)	Size of the program of CL	Real number

CL(fithe size)	Total size of files of CL	Real number
CL(output size)	Size of the result of the execution of CL	Real number
CL(nbr_pe)	Max number of Processors of CL	Naturel Number
CL(mips)	Capacity of the Processors of Cl	Real number
Dem(mips)	Sum of customer request's mips	Real number
Cost	The Cost of utilization of the capacity of the Processors in the host (mips)	Real number
Cl(Cost)	The Cost of the capacity of the Processors in the host (mips) à used by cloudlet	Real number (DA)
Cl(budget)	The max Cost of cloudlet.	Real number (DA)
Sol(Cost)	The total Cost of the solution and all the set of cloudlets	Real number (DA)
minLoad	The Min value of the energy in the optimized host	%
maxLoad	The Max value of the energy in the optimized host	%
E	Shared space E to facilitate the exchange of the resources between the host agents in the same datacenter	Matrix [CX3]
$E(Cl_m, Host)$	The Host that is going to execute the cloudlet m	@IP, nom de domain, …
$E(Cl_m, mipsprice)$	The proposed prix by the cloudlet m	Real number (DA)
$E\ (Cl_m, Counter)$	The Number of the hosts that has been validated the solution $E(Cl_m, Host)$	Naturel Number

Table 2. Defining the set of the variables

3.3.2 Constraints Identification

➢ **Constraint 1 :** For a virtual machine accepts to run a set of M cloudlets:

$$\sum_{m=1}^{M}(Cl_m(length)+Cl_m(outputsize)) \le VM_l(ram) \quad (1)$$

$$\sum_{m=1}^{M} Cl_m(filesize \le VM_l(storage) \quad (2)$$

$$\sum_{m=1}^{M} Cl_m(bw) \le VM_l(bw) \quad (3)$$

$$\sum_{m=1}^{M} Cl_m(mips) \le VM_l(mips) \quad (4)$$

Where:

$$Cl_m(mips) = CL(length) * CL(nbr\_pe) \quad (5)$$

$$VM_l(mips) = \sum_{k=1}^{p} PEV_{lk}[mips] \quad (6)$$

➢ **Constraint 2:** A virtual machine is already running a set of Mcloudlets. To allow run a new cloudlet M':

$$Cl_{m'}(length) + \sum_{m=1}^{M} Cl_m(length) + Cl_m(outputsize) \le VM_l(ram) \quad (7)$$

$$Cl_{m'}(filesize) + \sum_{m=1}^{M} Cl_m(filesize) \le VM_l(storage) \quad (8)$$

$$Cl_{m'}(bw) + \sum_{m=1}^{M} Cl_m(bw) \le VM_l(bw) \quad (9)$$

$$Cl_{m'}(mips) + \sum_{m=1}^{M} Cl_m(mips) \le VM_l(mips) \quad (10)$$

and $\boldsymbol{m}' \neg \in [1, \boldsymbol{M}]$.

➢ **Constraint 3 :** For a host, J accepts to host a set of $\boldsymbol{V}$ virtual machines:

$$\sum_{l=1}^{V} VM_l(ram) \le Host_j(ram) \quad (11)$$

$$\sum_{l=1}^{V} VM_l(storage) \le Host_j(storage) \quad (12)$$

$$\sum_{l=1}^{V} VM_l(bw) \le Host_j(bw) \quad (13)$$

$$\sum_{l=1}^{V} VM_l(mips) \leq Host_j(mips) \quad (14)$$

where

$$Host_j(mips) = \sum_{i=1}^{P} PE_i(mips) \quad (15)$$

➢ **Constraint 4:** A host ***J*** already hosts a set of V machines. To accept the hosting of new virtual machine ***l'*** (free VM or VM to migrate):

$$VM_{l'}(ram) + \sum_{l=1}^{V} VM_l(ram) \leq Host_j(ram) \quad (16)$$

$$VM_{l'}(storage) + \sum_{l=1}^{V} VM_l(storage) \leq Host_j(storage) \quad (17)$$

$$VM_{l'}(bw) + \sum_{l=1}^{V} VM_l(bw) \leq Host_j(bw) \quad (18)$$

$$VM_{l'}(mips) + \sum_{l=1}^{V} VM_l(mips) \leq Host_j(mips) \quad \text{and } l' \neg \in [1, V]. \quad (19)$$

➢ **Constraint 5:** We use a shared space ***E*** to facilitate the exchange of resources between the ***HA*** and ensure the best cost for each cloudlet. So, the constraint of selecting the HA_j as the best host to run the cloudlet ***m***:

$$E(Cl_{m'}, Host) \equiv Host_j \quad (20)$$

$$E(Cl_{m'}, mipsprice \equiv Host_j(mipsprice) \quad (21)$$

$$\forall i \in [1, C], E(Cl_{i'}, counter) \equiv H \quad (22)$$

➢ **Constraint 6**: After selecting a host J at a minimal cost, we reduce the number of VMs chosen SV_j to run the set of cloudlets. So, the best choice for host J is:

$$\forall SV_j, |SV_j| \leq |V_j| \quad (23)$$

$$bestsolution = min(SV_j) \quad (24)$$

➢ **Constraint 7:** Optimizing the load of a host: We say a host ***j*** (which hosts a set of ***N*** virtual machines) is optimized if it satisfies the following constraint:

$$minload \leq Host_j(mipsLoad) \leq maxload \quad (25)$$

➢ **Constraint 8:** To optimize a host ***j*** by consolidation we use another $host_{j'}$:

$$Host_j(mipsLoad) \leq minload \quad (26)$$

$$Host_j(mipsLoad) + Host_{j'}(mipsLoad) \leq maxload \quad (27)$$

Note: We launch the optimization process in the case of a change of the state of a host to ON, or the case of release of a virtual machine (ie: end of execution of all cloudlets of this VM).

➢ **Constraint 9 :** To optimize a host ***j*** by balancing the load using another host j':

$$Host_j(mipsLoad) \geq maxload \quad (28)$$

$$(mipsLoad) + MigV_{jj'}(mips) \leq maxload \quad (29)$$

$$Host_j(newCost) + Host_{j'}(newCost) \leq Host_j(oldCost) + Host_{j'}(oldCost) \quad (30)$$

➢ **Constraint 10**: For a solution to be valid for a cloudlet, his Cost must not exceed the budget proposed by the client:

$$Sol(cost) \leq CL(budget) \quad (31)$$

Scenario of Computing

❖ **BA** broadcasts the customer request to **DCA** agents.

❖ Every **DCA** forwards the request to any **HA** who is in **ON** state.

❖ **HA** agents negotiate and collaborate in order to establish the best local solution in datacenter level.

❖ **Initial Step:** We create the shared space between **HA** agents in the same datacenter

$$E(Cl_{m'}, mipsPrice) = MaxDouble$$

$$E(Cl_{m'}, Host) = null$$

$$E(Cl_{m'}, Counter) = 0$$

❖ Every Agent Host J looking for the possibility of executing each cloudlet, by the procedure **Ordonnance (*Host*$_j$)**

```
Procedure Inform (Host_j, Host_j' , Cl_m)
Host_j send a message « replacement of Cl_m»
to Host_j'
for all Cl_i confirmed by Host_j' do
CancelConfirmation(Host_j, Cl_i)
end
Ordonnance (Hostj')
end.

Procedure Confirm(Host_j, Cl_i)
E(Cl_m, mipsPrice) ++
end.

Procedure CancelConfirmation(Host_j, Cl_i)
E(Cl_m,mipsPrice) --
end.
```

• In the end of the procedure **Ordonnance**, any **HA** that verifies the constraint **C5'** send his solution proposals to **DCA.**

```
Procedure Ordonnance (Host_j)

for all Cl_m dans l'espace E do
if ∃ VM ∈ Host_j that satisfies C2 and
Host_j(mipsPrice)< E(Cl_m, mipsPrice) then
ifE(Cl_m, Host) '" null  then
Host_j' = E(Cl_m,Host)

Informe (Host_j , Host_j' , Cl_m)
end
E(Cl_m, Host) = Host_j
E(Cl_m, mipsPrice) = Host_j(mipsPrice)
end

Confirm(Host_j , Cl_m)
end.
```

Example:

H = 3 hosts

	Host	mipsPrice	Counter	NB
Cl_1	null	MaxDouble	0	
Cl_2	Null	MaxDouble	0	
Cl_3	Null	MaxDouble	0	
Cl_4	Null	MaxDouble	0	

H1 (mipsPrice = 100, cl1, cl2)

	Host	mipsPrice	Counter	NB
Cl_1	H1	100	1	
Cl_2	H1	100	1	
Cl_3	Null	MaxDouble	1	
Cl_4	Null	MaxDouble	1	

H2 (mipsPrice=150, *c*l1, *c*l2, *c*l4)

	Host	mipsPrice	Counter	Note
Cl_1	H1	100	2	
Cl_2	H1	100	2	
Cl_3	Null	MaxDouble	2	
Cl_4	H2	150	2	

H3 (mipsPrice = 50, cl1, cl4)

	Host	mipsPrice	Counter	Note3
Cl_1	H3	50	1	Info H1 about Cl1
Cl_2	H1	100	1	
Cl_3	Null	MaxDouble	1	
Cl_4	H3	50	1	Info H2 about Cl4

Info H3, H1, Cl1➔H1 (mipsPrice = 100, cl3, cl4)

	Host	mipsPrice	Counter	Note
Cl_1	H3	50	2	
Cl_2	H1	100	2	
Cl_3	H1	100	2	
Cl_4	H3	50	2	

Info H3, H2, Cl4➔ H1 (mipsPrice = 100, cl3)

	Host	mipsPrice	Counter	Note
Cl_1	H3	50	3	
Cl_2	H1	100	3	
Cl_3	H1	100	3	
Cl_4	H3	50	3	

	Host	mipsPrice	Counter	Note
Cl_1	H3	50	3	C5 verf
Cl_2	H1	100	3	C5 verf
Cl_3	H1	100	3	C5 verf
Cl_4	H3	50	3	C5 verf

• In order to establish the best solution in cloud level, **DCA** agents share their local solutions (given from local **HA** agents) and negotiate between them to choose the best solution (best cost) for each cloudlet.

• After the negotiation, **DCA** agents send their chosen solutions to **BA** and confirm their **HA** solution proposals.

• Each **HA** receives the confirmation and verifies the constraint C**6.**

• **BA** receives the response from **DCA**.

• If there is no solution for any cloudlet of the demand then **BA** selects a set of new **VM** and affect the cloudlets to them and start the procedure **HostingNewVMs.**

• else, **BA** executes the affected cloudlets according to the solution, and broadcasts the rest of cloudlets (not affected) to **DCA** agents as new request.

	Cl1	Cl2	Cl3	Cl4	Cl5	Cl6	Cl7	Selected VMs
SV1J	VM1	VM2	VM3	VM4	VM5	VM6		6 VMs
SV2J	VM2		VM6	VM5			VM6	3VMs
SV3J	VM5				VM3			2 VMs
the min(SVj) ≡ SV3J the number of selected VMs is 2								

Table 3. Selected VMs

Procedure Hosting New VMs

❖ **BA** broadcasts a request of hosting a set of new **VM** to **DCA** agents**.**

❖ Every **DCA** forward the request to any **HA** who is in **ON** state.

❖ **HA** agents negotiate and collaborate in order to establish the best solution and satisfy the constraint **C4**.

❖ **HA** agents send their solution proposals to **DCA.**

❖ In order to establish the best solution in cloud level, **DCA** agents share their local solutions (given from local **HA** agents) and negotiate between them to choose the best solution (best cost) for each VM.

❖ After the negotiation, **DCA** agents send their chosen solutions to **BA**.

❖ If there are some **VMs** that don't have solution then, **DCA** creates **HOffA** agents and broadcasts to them a request of **VM Hosting**.

❖ **HOffA agents negotiate and collaborate in order to establish the best solution and satisfy the constraint C3**.

❖ **HOffA** agents send their solution proposals to **DCA.**

❖ **DCA** agents share their local solutions and negotiate between them to choose the best solution and send it to **BA**.

❖ **BA** launch**processing ()** procedure checks of the constraint **C10** with every cloudlet.

4. Simulation Experiments and Results

We use Cloudsim platform (Rodrigo N., Rajiv, Anton, Rose, & Rajkumar, 2011) as simulation tool for our experiment. In this experiment, the applied algorithms are First Fit algorithm (FF Algo) and Election Algorithm (Elec Algo) which is based on our approach that requires SMA and DCSP Implementation. As shown in results, we made comparisons between (FF Algo and Elec Algo) taken into consideration the following characteristics: average System energy, Cost, Cost gain and use Constraints.

```
Procedure Processing ()
for all CLdo
|   if CL ∉ solution then
|   |  send NO to the customer
|   else
|   |  if solution satisfies C10 then
|||send YES to the customer
||  else
|||
|   |  |  send to the customer a Negotia-
tion request to change
|   |  |  the budget, in order to satisfy
C10
|   |   end
| end
end
BA apply the solution and confirm the DCA
agents by his decision.
End.
```

```
Procedure Optimization ()
for all Host_j doesn't satisfy C7 do
|   if ∃ Host_j' satisfies  C8 then
||Migrate all VMs from Host_j to Host_j'
  |   else
  |   |  ifHost_j(mipsLoad) ≥ maxLoad then
|||set Host_j marked
  |||      while ∃  Host_j' has not been
marked do
||||set Host_j' marked
||||    if ∃  MigV_jj' doesn't satisfy C9
then
|| ||| Migrate all VMs in MigV_jj' from
Host_j to Host_j'
  |    |   |     |  end
  |    |   |   end
|||   cancel the marking of all Host_j
  |    |   end
| end
end.
```

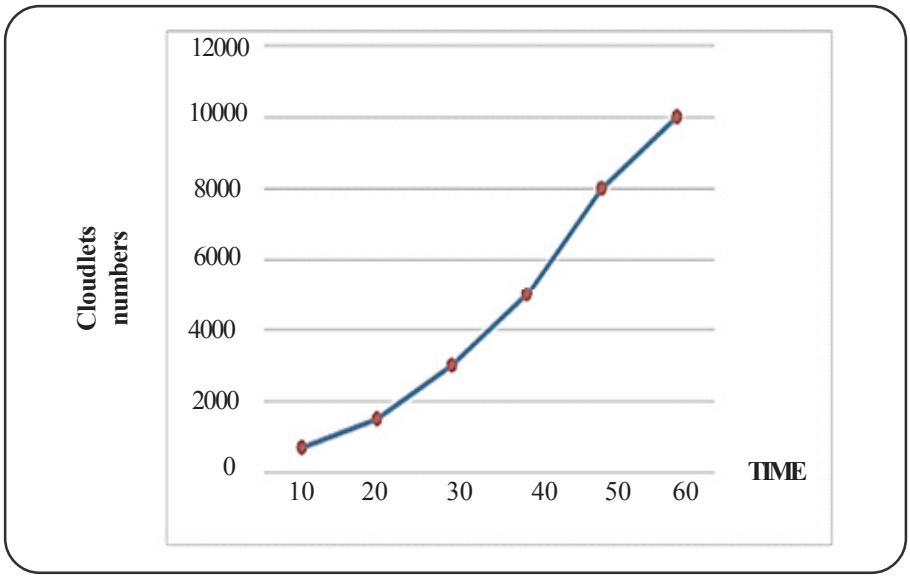

Figure 3. Number of Requested cleoudlets by time

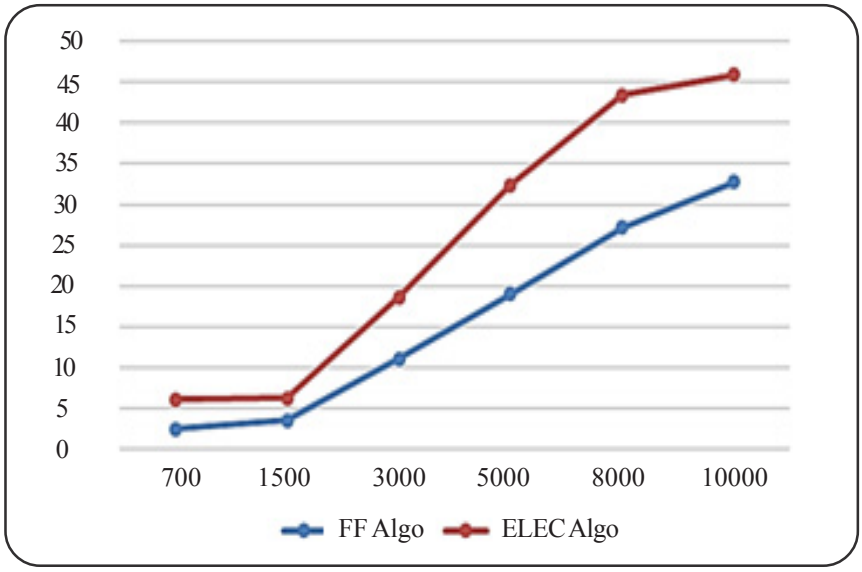

Figure 4. Average of energy consumption by number of requested cloudlets for each algorithm

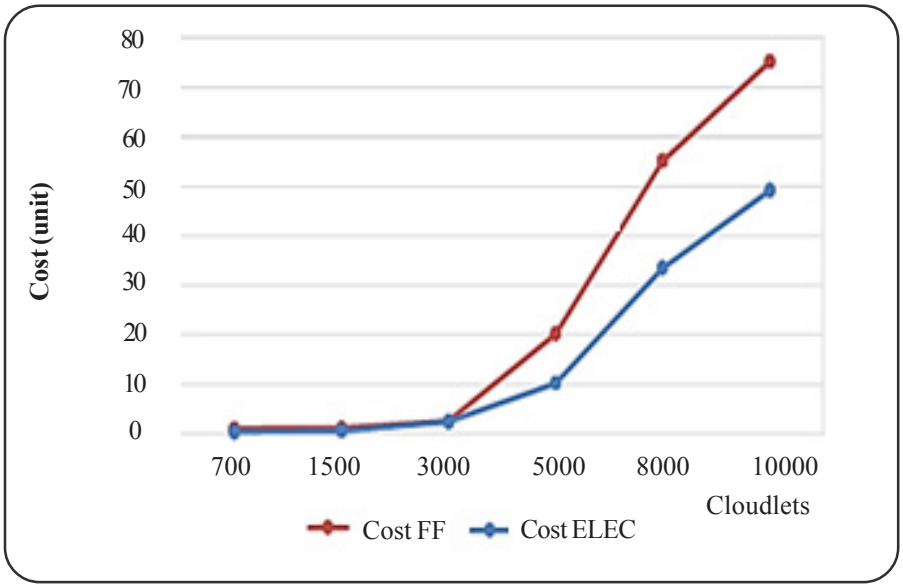

Figure 5. The cost for each algorithm

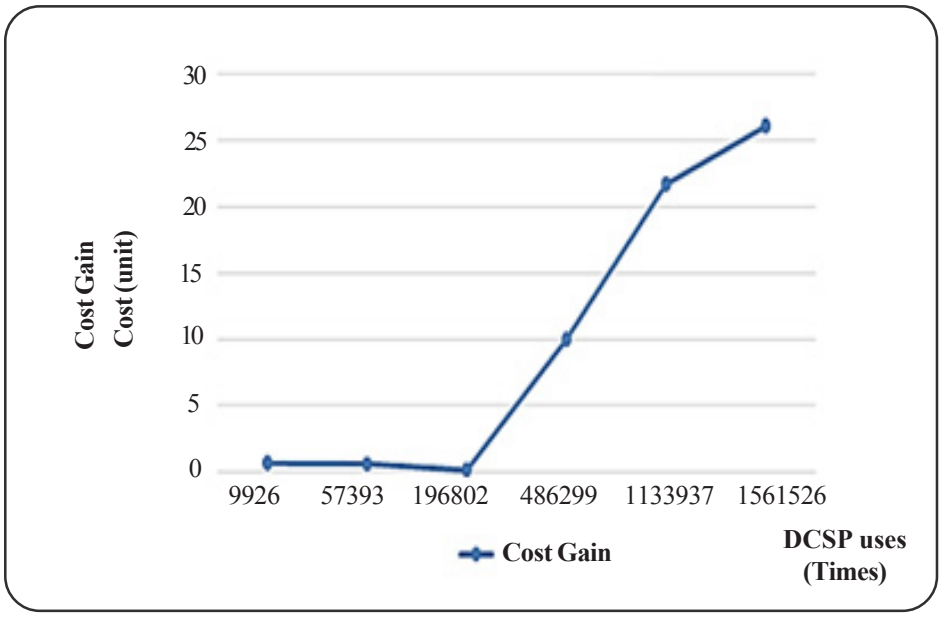

Figure 6. Evolution of cost Gain by DCSP using

The main objective of our algorithm is to find solutions with low cost and low energy consumption in data centers. Figure 4 showed that the average energy consumption by our algorithm is lower than FF Algorithm. This interprets the reduction of the cost of our Algorithm in Figure 5. Also, The figure 6 illustrates the benefit of using DCSP and its positive impact on the cost gain.

5. Conclusion

Results show that the possibility of including new paradigms in RA mechanisms without affecting its performance has become possible. Furthermore, experiments show that the use of DCSP beside MAS creates a new paradigm that can be used efficiently in solving the problems related not only to RA but also to plan the system of cloud computing.

For future works, we are looking forward to studying more cloud computing problems like scalability and extending this approach to other paradigms in scheduling and optimizing other procedures such as search algorithms, artificial intelligence and machine learning processes.

References

[1] Buyya, R., Yeo, C. S., Venugopal, S. (2008). 'Market-oriented cloud computing: Vision, hype, and reality for delivering IT services as computing utilities', *In*: Proceedings - 10th IEEE International Conference on High Performance Computing and Communications, HPCC 2008, 5–13.

[2] Al, M., et al. (2017). 'Distributed Resource Allocation in Cloud Computing Using Multi-Agent Systems, 9 (2), 110–115. Available

[3] Gamal Eldin I. Selim, Mohamed A. El-Rashidy, N. A. E.-F. (2016). An Efficient Resource Utilization Technique for Consolidation of Virtual Machines in Cloud Computing Environment', in NATIONAL RADIO SCIENCE CONFERENCE II . RELATED WORK B . *Energy Consumpt*, 316–324.

[4] Anuradha, V. P., Sumathi, D. (2014). A survey on resource allocation strategies in cloud computing', *International Conference on Information Communication and Embedded Systems* (ICICES2014), 3 (6) p 1–7.

[5] Bajo, J., De la Prieta, F., Corchado, J. M., Rodríguez, S. (2016. A low-level resource allocation in an agent-based Cloud Computing platform', *Applied Soft Computing Journa*l, 48, p. 716–728.

[6] Gutierrez-Garcia, J. O., Sim, K. M. (2010). Self-organizing agents for service composition in cloud computing', *In*: Proceedings - 2nd IEEE International Conference on Cloud Computing Technology and Science, *CloudCom* 2010, 59–66.

[7] Jing Liu, Weicai Zhong and Licheng Jiao (2006). A multiagent evolutionary algorithm for constraint satisfaction problems, *IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics)*, 36 (1) 54–73.

[8] Chen, J., Han, X., Jiang, G. (2014). A Negotiation Model Based on Multi-agent System under Cloud Computing, *In*: The Ninth International Multi-Conference on Computing in the Global Information Technology, 157–164.

[9] Son, S., Sim, K. M. (2012). A Price- and-Time-Slot-Negotiation Mechanism for Cloud Service Reservations', *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42 (3) 713–728.

[10] Gutierrez-Garcia, J. O., Sim, K. M. (2011). Agents for cloud resource allocation: An amazon EC2 case study', Communications in Computer and Information Science,

261 CCIS, p 544–553.

[11] Yokoo, M., Suzuki, K., Hirayama, K. (2005). Secure distributed constraint satisfaction: Reaching agreement without revealing private information, *Artificial Intelligence*, 161(1–2) 229–245.

[12] Sim, K. M. (2012). Agent-based cloud computing', *IEEE Transactions on Services Computing*, 5 (4) 564–577.

[13] Li, X., Zhang, H., Zhang, Y. (2009) 'Deploying mobile computation in cloud service', Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 5931 LNCS(200850731369), p. 301–311.

[14] Nurmi, D., Wolski, R., Grzegorczyk, C., Obertelli, G., Soman, S., Youseff, L., Zagorodnov, D. (2009). The eucalyptus open-source cloud-computing system', in 2009 9th *IEEE/ACM International Symposium on Cluster Computing and the Grid*, CCGRID 2009, 124–131.

[15] Farahnakian, F., Pahikkala, T., Liljeberg, P., Plosila, J. and Tenhunen, H. (2014) Multi-agent based architecture for dynamic VM consolidation in cloud data centers, *In*: Proceedings - 40th Euromicro Conference Series on Software Engineering and Advanced Applications, SEAA 2014.

[16] Beloglazov, A., Buyya, R. (2012). Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers, *In: Concurrency Computation Practice and Experience*, 1397–1420.

[17] Wu, L., Kumar Garg, S., Buyya, R. (2012).'SLA-based admission control for a Software-as-a-Service provider in Cloud computing environments, *Journal of Computer and System Sciences. Elsevier Inc*., 78 (5) 1280–1299.

[18] Farahnakian, F., Liljeberg, P., Pahikkala, T., Plosila, J. and Tenhunen, H. (2014). Hierarchical VM management architecture for cloud data centers, *In*: Proceedings of the International Conference on Cloud Computing Technology and Science, *CloudCom*, 306–311.

[19] Srikantaiah, S., Kansal, A., Zhao, F. (2008). Energy Aware Consolidation for Cloud Computing, *In*: Proceedings of the 2008 conference on Power aware computing and systems, p. 1–5.

[20] You, X., Xu, X., Wan, J., Yu, D. (2009). RAS-M:Resource allocation strategy based on market mechanism in cloud computing, *In*: Fourth ChinaGrid Annual Conference, *ChinaGrid,* 2009, 256–263.

[21] Sriram, V., Ravimaran, S. (2014). Dynamic Resource Parallel Processing and Scheduling by Using Virtual Machine in the Cloud Environment, *International Journal of Engineering Research & Technology*, 3 (4) 1265–1269.

[22] Chandak, A. (2012). Dynamic Load Balancing of Virtual Machines using QEMU-KVM, *International Journal of computer Applications*, 46 (6) 10–14.