


Mobile Agents-Based Framework for Dynamic Resource Allocation in Cloud Computing

Safia Rabaoui¹ ^a, H ela Hachicha^{1,2} and Ezzeddine Zagrouba¹ ^b

¹University of Tunis El Manar, Higher Institute of Computer Science, Laboratory of Informatics, Modeling and Information and Knowledge Processing (LIMTIC), 2 Rue Abou Rayhane Bayrouni, Ariana 2080, Tunisia

²College of Computer Science and Engineering, University of Jeddah, Saudi Arabia


Keywords: Cloud Computing, Multi Agent-System, Mobile Agent, Dynamic Resource Allocation, Cost, Makespan, Task Scheduling.


Abstract: Nowadays, cloud computing is becoming the more popular technology for various companies and consumers, who benefit from its increased efficiency, cost optimization, data security, unlimited storage capacity, etc. One of the biggest challenges of cloud computing is resource allocation. Its efficiency directly influences the performance of the whole cloud environment. Finding an effective method to address these critical issues and increase cloud performance was necessary. This paper proposes a mobile agents-based framework for dynamic resource allocation in cloud computing to minimize the cost of virtual machines and the makespan. Furthermore, its impact on the best response time and task rejection rate has been studied. The simulation shows that our method gave better results than the former ones.

1 INTRODUCTION

The cloud is a term used to refer to a network of remote servers that are accessed over the Internet to store, manage, and process data instead of relying on a local server or personal computer (Yahya et al., 2020). It provides on-demand access to a shared pool of computing resources that can be rapidly provisioned and scaled to meet users' needs, including storage, processing power, and applications. The cloud enables users to store and access data and applications from anywhere with an internet connection, eliminating the need for local storage and computing infrastructure. It offers flexibility, scalability, and cost-effectiveness, as users can pay for the resources they use rather than invest in and manage their own hardware and software infrastructure. One of the main challenges in cloud computing is resource allocation due to the dynamic nature of resources (Kumar et al., 2018). In fact, with companies and consumers expanding their requirements, the need for efficient resource allocation is also emerging. Also, the cost of cloud resources represents a serious concern among many companies and consumers. This is why cloud resource allocation

is a topic of discussion, and this is to meet the current demand in the best response time and reduce cloud services costs. This issue is a challenge in cloud computing systems. Computing services must be highly dependable, scalable, and autonomous to support ubiquitous access, dynamic discovery, and composability (Buyya et al., 2008). Resource allocation in cloud computing faces challenges, including cost efficiency, response time, computational performance, and scheduling tasks (Belgacem, 2022). Users of cloud computing services target to accomplish tasks with the lowest costs possible. Research in multi-agent systems (MAS) and mobile agents has evolved considerably, and several distributed systems have been deployed using these technologies. Indeed, MAS is based on autonomous and intelligent agents sharing a typical environment. These agents cooperate and interact with each other to achieve a global goal. While there are differences between cloud computing and multi-agent systems, they are both technologies designed for distributed models. Thanks to their advantages and features, many issues in different fields can be solved by integrating or combining cloud and multi-agent systems, as in (Qasim et al., 2020). Also, in (Bei et al., 2022), the authors proposed a fair and efficient multi-resource allocation for cloud computing. In addition, research has been per-

^a  <https://orcid.org/0000-0001-5978-5613>

^b  <https://orcid.org/0000-0002-2574-9080>

formed in recent years to provide architectures for using the cloud in MAS, such as (Venkateshwaran et al., 2015), which presented a framework for agent negotiation while using the resources provided by the cloud environment. Other research focuses on using agent technologies to address cloud computing difficulties. Among these, we mention the work of (Mahavidyalaya and Nadu, 2021), (Barkat et al., 2021), and (Yahya et al., 2020), who recommended employing MAS to address cloud challenges. The authors of (FAREH, 2015) presented an architecture based on self-organizing agents to deal with the difficulty of cloud service composition. Current researchers handle these issues dependently without achieving an optimal solution. Therefore, this work focuses on dynamic resource scheduling for efficient resource allocation. Synchronously, we must maintain customer service quality by minimizing the makespan and cost of virtual machines. The contributions of our research are:

- Create a dynamic resource allocation model to increase resource utilization and user satisfaction by minimizing task rejection rate.
- Propose a new Dynamic Resource Allocation approach with Mobile Agents DRAMA to minimize the cost and makespan.

This paper is structured as follows. Section 2 is a list of related works. Section 3 shows the problem statement and formulation. Section 4 describes our proposed solution. Experiment results and the implementation of a proposed prototype are reported in Section 5. Finally, Section 6 concludes the paper by identifying our plans.

2 RELATED WORKS

This section will discuss some of the most relevant frameworks that focus on using multi-agent systems and mobile agents for resource allocation in cloud computing. In (Soltane et al., 2018), authors have proposed a cloud architecture based on a multi-agent system exhibiting a self-adaptive behavior to address dynamic resource allocation DRA. The principal focus of this work is to enhance energy consumption while satisfying the quality of service QoS demanded by users. This architecture consists of four agents: an analyzer agent who identifies the resources and services required by users and builds specific queries. The scheduling agent is responsible for allocating resources users need and making the final decision about resource allocation. The controller agents track

the status of resources in the data center. The coordinator agent supervises the whole process.

Wang et al. (Wang et al., 2016) have defined a decentralized multi-agent-based Virtual Machine (VM) allocation approach. The approach aims to allocate VMs to Physical Machines (PMs) while minimizing system energy costs. This approach allows dispatching a cooperative agent to each PM to assist the PM in managing resources. Another solution based on agent technology, called low-level resource distribution, is proposed by (Bajo et al., 2016). This approach allows the distribution of computational resources throughout the entire cloud computing infrastructure, considering its complexity and associated computational costs. In this system, agents are distributed over the infrastructure. Each physical server in the cloud environment contains a set of stationary agents in charge of monitoring and making decisions that involve assigning or removing nodes for a particular service. Each service offered to the cloud users is associated with two agents, one for monitoring and the other for control; both are responsible for ensuring compliance with the previously established SLA agreement. Other agents also ensure the proper operation of the cloud computing system. On the other hand, some approaches have been proposed using mobile agents. In (Singh et al., 2017), a new mechanism was proposed that deploys various intelligent agents to reduce the cost of virtual machines and resource allocation complexity. This system defines four stationary agents and one mobile agent, which searches the resources from the available resource instances of a current data center. The mobile agents can manage resource allocation.

Further, Aarti Singh et al. (Singh and Malhotra, 2015) proposed using mobile agents for resource allocation in cloud computing, focusing on cost optimization. The end users send the resource request to the cloud data center, where all the resources are available. Every cloud is associated with a cloud Mobile Agent (MAc). Every MAc is responsible for all information on resources and their status, whether free or allocated. Initially, a service request arrives at an MAc and checks the available free resources to decide whether the request can be served. After that, the resource manager agent (RMA) will determine how it would be allocated, i.e., which technique should be applied to resources so that they will be adequately distributed and the cost of VMs minimized. Finally, resources are distributed to the user.

Belgacem Ali et al. (Belgacem et al., 2020) focus on dynamic resource allocation. The authors present a multi-objective search algorithm called the Spacing Multi-Objective Antlion algorithm (S-MOAL) to

minimize the makespan and the cost of virtual machines. However, they did not consider the system's energy consumption and fault tolerance issues. In the work (Abdullah and Surputheen, 2022), the authors present the Cooperative Agents Dynamic Resource Allocation and Monitoring in Cloud Computing CADRAM system. They include more than one agent to manage and observe resources provided by the service provider. The solution increases resource utilization and decreases power consumption while avoiding SLA violations. He presented an algorithm to select a virtual machine called the node failure discovery. Hence, VMs are allocated to the user based on the type of the job.

3 PROBLEM STATEMENT AND FORMULATION

This research focuses on the dynamic resource allocation in Infrastructure as a Service (IaaS). In the IaaS model, computing resources are provisioned and delivered over the Internet in a virtualized environment. IaaS provides users with virtualized infrastructure components such as virtual machines, storage, and networking capabilities. The cloud resources provider makes virtual machines available to customers when it sends requests to determine their resource requirements via the graphical user interface GUI. Each virtual machine is configured with its own processors and memory resources, which gives an overview of a resource pool. These resources are dynamically reserved and released. The first aim of our research is to develop a dynamic resource allocation system that acts as an intermediary between cloud providers and users. This system allows us to connect these two partners: cloud users who want to choose the appropriate resources that meet their needs and cloud providers who seek to maximize their benefits and allocate their resources efficiently. We decided to use mobile agents and multi-agent systems that are widely used to design and develop complex and distributed systems. The cloud computing system has j Datacenters:

$$DC = \{DC_1, DC_2, \dots, DC_j\} \quad (1)$$

Each data center comprises a set of physical nodes (PNs):

$$PN = \{PN_1, PN_2, \dots, PN_p\} \quad (2)$$

Where p signified the number of physical nodes in the data center, and many virtual machines (VMs) reside on each physical node. Each VM is a resource modeled as a set:

$$VM = \{VM_1, VM_2, \dots, VM_m\} \quad (3)$$

Where m signified the number of virtual machines running on PN at a given time.

To handle this problem, we consider that the input is a set of tasks and a set of VMs, where:

$$T = \{T_1, T_2, \dots, T_n\} \quad (4)$$

Represent a set of tasks. Output obtaining a better mapping of T_n to VM_m , (T_n, VM_m) in order to reduce the cost and the makespan. We consider a parallel-machine scheduling problem with n Tasks and m machines: The makespan is the time taken to complete all the Tasks when optimally allocated across the machines. The equation can be represented as:

$$M(t) = \max(m_1, m_2, \dots, m_n) \quad (5)$$

Where m_1, m_2, \dots, m_n represent the completion times of each machine.

$$m(n) = \sum(p_i) \quad (6)$$

Where p_i represents the processing time of each Task n assigned to machine m . The cost of VM utilization is denoted by C is the sum prices of running all the tasks on the VM, as shown in (7).

$$C(t) = \sum_{i=0}^n C_i(t) \quad (7)$$

$$C_i(t) = Cost_{CPU} + Cost_{ram} + Cost_s + Cost_{bandwidth} \quad (8)$$

4 PROPOSED FRAMEWORK

The proposed dynamic resource allocation with mobile agents (DRAMA) model permits to handle with the main issues: minimizing the makespan, the cost, and the rejection rate of tasks. As depicted in Fig. 1, our architecture is structured into three layers, each with specific roles and responsibilities within the architecture.

1. User Layer: it provides an interface to access the cloud services through which users specify their resource allocation needs.
2. Resource Allocation Layer represents an inter-layer between the upper layer (users) and the lower layer (data center infrastructure).
3. Data Center Infrastructure Layer: it provides resources as services. It consists of a physical layer (physical machines, hosts) and a virtual layer (Virtual Machines, VMs).

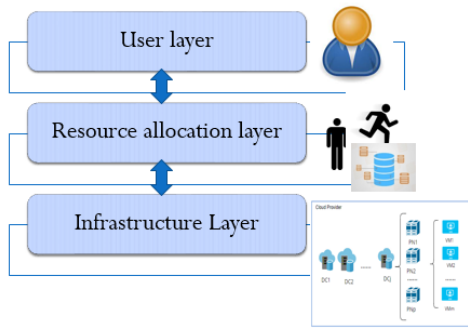


Figure 1: The Architecture's Layers.

The proposed architecture shown in Fig. 2 involves a set of stationary and mobile agents interacting and collaborating to achieve a common goal. In this architecture, we have integrated the concept of cloud providers. Within the same provider, it is possible to have many data centers which are connected through a network.

For this reason, we have defined a new class of mobile agents (Data Center Management Agents) that has a mission to migrate from one data center to another related to one cloud provider. Also, we have assigned each user request to one mobile agent (Allocation Agent) that migrates from one provider to another to make allocation decisions and define the allocation plan. In the following, we describe each agent's roles and internal architecture in our architecture.

Request Agent RA: is a stationary agent that acts as an intermediary between the user and the system. RA is responsible for retrieving the data sent from the user interface and formulating the user's request. After receiving the requests, RA organizes the requests (tasks) based on a task scheduling algorithm to use the available resources correctly. Every task is scheduled by a scheduling algorithm in such a way that each task is forwarded to the allocation agent. It is also accountable for building specific queries (Request Q). The request Q delivered by the request agent is composed of three elements, as follows:

$$Q = (R, Cp, Qos) \quad (9)$$

Where:

$$R = (r_1, r_2, \dots, r_n) \quad (10)$$

Equation (10) represents resources and services the user needs (such as data storage and virtual machines).

$$Cp = (cp_1, cp_2, \dots, cp_n) \quad (11)$$

Cp is the capacity corresponding to each resource;

$$Qos = (Qos_1, Qos_2, \dots, Qos_n) \quad (12)$$

Qos represents the Qos for each resource. We represent the combination of resources and their

corresponding requested capacities and Qos to build the suitable VM_r as:

$$\begin{pmatrix} r1 & cp1 & Qos1 \\ r2 & cp2 & Qos2 \\ r3 & cp3 & Qos3 \end{pmatrix} \quad (13)$$

The Request Agent uses the Task Scheduling algorithm 1 to assign the tasks to allocation agents. The priorities for each task are assigned based on the job size and inter-arrival time. Then, the tasks are converted into priority-based tasks represented by:

$$TQ = (TQ_1, TQ_2, \dots, TQ_s) \quad (14)$$

Then, they will search for the priority queue.

```

Input : Tasks T={t1,t2,,,,,tn};
Output : connections by the Allocation Agent;
for all t1,t2,,,,,tn in Task scheduler do
    Consider for all Task length Sx;
    if the current task length = next task length
    then
        Sort the request according to their
        arrival time At;
        Add Task T to Qe based on At;
        EnQueue Qe ;
    else
        Sort the request according to their
        length;
        EnQueue Qe;
    end
    while Qe!= Null do
        Create Allocation Agent AA ;
        Add Task TQ to Allocation Agent AA;
    end
end
    
```

Algorithm 1: Scheduling Algorithm.

Input : Tasks T={t1,t2,,,,,tn};
Output : best offer (available resources) ;

1. AA migrates to CSA (provider representative) to select the appropriate resources for the Task;
2. Receives responses from each CSA (sets of available resources);
3. It filters and selects the best offer;

Algorithm 2: Allocation Agent Algorithm.

Allocation Agent AA: is a mobile agent that aims to find the best offer appropriate to the user's request in the data-center infrastructure. It receives the task and migrates to all cloud providers to select the appropriate resources for the received request. It participates

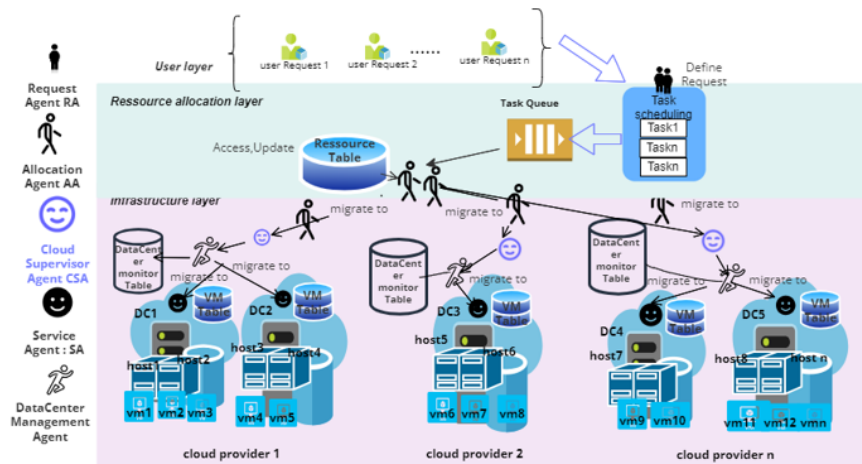


Figure 2: The Architecture's Cloud Providers and Agents.

in the system and aims at obtaining maximum benefit for the user (optimization of the cost-performance ratio, minimization of the cost).

Cloud Supervisor Agent CSA: It represents the provider in the cloud environment. This agent is responsible for cooperating with the allocation agents and assigning the data-center management agent the mission to find the virtual machine corresponding to the user's needs. The cloud supervisor agent offers an offer (list of resources to be allocated, prices, etc.) to respond to a given task. To do this, it cooperates with the Data-Center Management Agent in order to exploit the resources available in the different data centers.

Input : needed resources;

Output : available resources ;

1. CSA sends the request to DCMA;
2. It receives all available resources;
3. It proposes an offer (list of resources to be allocated);

Algorithm 3: Cloud Supervisor Agent Algorithm.

Data Center Management Agent DCMA: It discovers resource capacities on physical nodes and manages virtual machines in DC. This agent has visibility into all resources in DC from the same provider. It maintains the current state of physical machines and tracks resource status. It is responsible for the coordination between the different service agents. When it receives a Task from the cloud supervisor agent, it migrates to all service agents to discover the resource capabilities of the different virtual machines, then prepares the best answer with the minimum cost and transfers it to the Cloud Supervisor Agent.

Input : CPU, RAM, storage, $cost_{cpu}$, $cost_{ram}$, $cost_{storage}$, Bandwidth, $cost_{bandwidth}$;

Output : TotalCost;

for all VM1, VM2, ..., VMm **do**

$TotalCost_{CPU} = CPU * cost_{cpu}$;

$TotalCost_{ram} = ram * cost_{ram}$;

$TotalCost_{storage} = storage *$

$TotalCost_{bandwidth} = Bandwidth *$

$cost_{bandwidth}$;

$TotalCost = total_{cost}_{cpu} + total_{cost}_{ram} +$

$total_{cost}_{storage} + total_{cost}_{bandwidth}$

end

Algorithm 4: TotalCost Algorithm.

Service Agent SA: is responsible for maintaining the capacity of physical nodes and handling virtual machines in the data center. It is responsible for discovering the current state of each machine (capacity used, free capacity, state). It is also responsible for maintaining information on the status of resources and characteristics. It also has the mission to manage resources (allocation, freeing). For each physical machine exists one Service Agent.

5 SIMULATION AND RESULTS

This section presents the performance and result of the proposed framework throughout the implementation of three prototypes. We have used the CloudSim simulator platform (Calheiros et al., 2011) to simulate the cloud infrastructure and the JADE platform to implement our multi-agent system. The cloud data center is created by using a series of PMs. Each data center with multiple numbers of hosts and their corresponding VMs is initiated. In the first prototype,

we implement an allocation system without a multi-agent system. In the second prototype, we implement our proposed framework (Rabaoui et al., 2021).

In this architecture, we have only one mobile agent that migrates from one data center to another without considering the cloud provider. In the third prototype, we implement our proposed architecture using two mobile agents: allocation agents that migrate from one cloud provider to another and data center management agents that migrate from one service agent to another within the same cloud provider.

Table 1 provides the cloud sim specification of the proposed system. Different metrics, such as makespan, execution time, Refusal Rate and cost are used to analyze the simulation results. The proposed approach (DRAMA) improves system efficiency and is compared with some existing methods in the literature, including the First-Come First-Served (FCFS) algorithm and the Preference Based Task Scheduling (PBTS) algorithm.

Table 1: Simulation Specification.

Parameter	Value
The number of Hosts	4
The number of Tasks	30-300
Length of task	1600-3400
Memory	540
Bandwidth	25,00,00
Storage	500 GB
MIPS/PE	500

We use our proposed method. At $t=0$, a set of tasks arrive simultaneously. Then, jobs are prioritized by the task scheduling concept employed in our framework. The prioritized tasks are given to the allocation agents. AA will migrate and send the same received job to all cloud supervisor agents (CSA1, CSA2, etc.) listed in its directory. CSA forwards the task to one of the Data Center Management agents (DCMA) that it holds.

DCMA migrates to Service Agents (SA) to retrieve virtual machine states. The service agent sends the current status of each virtual machine to the data center management agent. DCMA compares the responses from the service agents to select the best solution that meets the user’s requirements and informs the cloud supervisor agent. The CSA receives the best-selected response and sends its offer to the allocation agent.

AA receives the offers of the CSA, filters them, and sorts them according to the price. It then selects the best suggestion that meets the user’s needs. It sends an allocation request to the CSA if it receives an allocation confirmation and sends the answer to

the RA. The AA gets an error message from the selected CSA if it is not confirmed. The AA returns an allocation request to the next CSA in its sorted list of offers. It enters a loop until it receives an allocation confirmation from a CSA, or it determines its circle and fails to allocate a resource, in which case it sends an error message to RA.

Fig.3 shows the makespan according to the number of tasks. This measurement reveals to us what time is needed to receive a response from the system. The simulation result gives a shorter Makespan when compared with the other two taken algorithms (FCFS and PBTS).

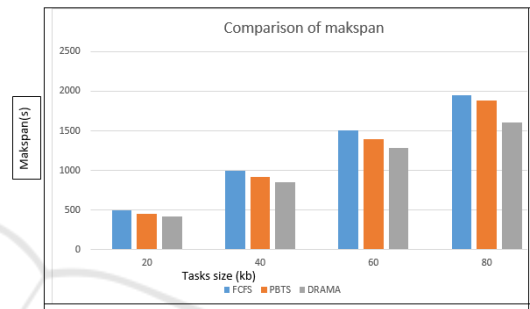


Figure 3: Comparison of makespan.

The examination of the proposed approach in terms of cost is presented in Fig. 4. It is apparent from the comparison of the three scheduling algorithms that the proposed method, DRAMA, decreases the value of the cost due to the completion of each task in an appropriate virtual machine that ensures a minimum price. Therefore, the proposed method allows a better match between the tasks and the virtual machines.

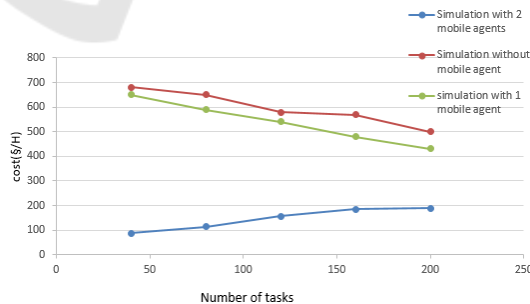


Figure 4: Costs for different numbers of tasks.

A comparative graph shown in Fig. 5 shows our proposed method’s lesser response time than existing methods.

The examination of the proposed approach in terms of the Refusal Rate is presented in Fig. 6.

$$RefusalRate = Total_{Trj} / Total_{Tsb} \quad (15)$$

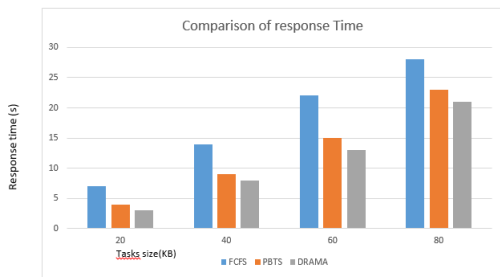


Figure 5: Comparison of response time.

Where: $Total_{Trj}$ is the total number of tasks rejected, and $Total_{Tsb}$ is the total number of tasks submitted. It indicates how many tasks are dismissed to the total number of submitted jobs and how many are accepted. It shows that our proposed methods have significantly lower refusal rates, indicating that they get more tasks than rejected ones.

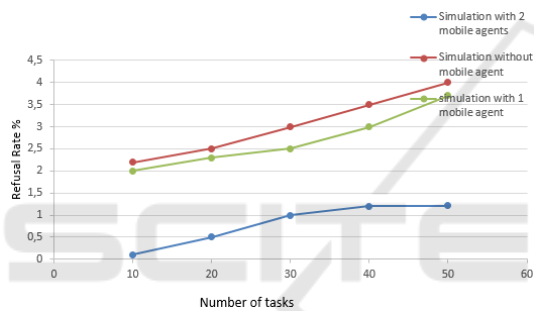


Figure 6: Refusal Rate.

6 CONCLUSIONS

In this paper, we proposed a Mobile agents-based framework for a Dynamic Resource Allocation approach with Improved task scheduling in cloud computing environments. Our system has two classes of mobile agents and three classes of stationary agents: mobile agents play a crucial role in choosing the most suitable offer for user demand. This agent ensures a secure exchange and greatly minimizes traffic on the network. The stationary agents manage local information. The performance evaluation results can significantly improve the makespan, the response time, and service quality (minimizing VM cost). This method combines the dynamic change of the cloud system state and the task scheduling to solve various issues, which makes it effective.

Three experiments were used to evaluate the performance of the proposed method, DRAMA. The first shows good results regarding the tasks' makespan, cost, and rejection rate. However, there is still much

work to be done. For our future work, we plan to present a formal model of our approach defining the query for resource allocation and user preferences. We also plan to describe in detail the scheduling algorithm and give additional experiments to evaluate other performance aspects and the security for the resource allocation.

REFERENCES

- Abdullah, M. and Surputheen, M. M. (2022). Cadram-cooperative agents dynamic resource allocation and monitoring in cloud computing. *IJCSNS*, 22(3):95.
- Bajo, J., De la Prieta, F., Corchado, J. M., and Rodríguez, S. (2016). A low-level resource allocation in an agent-based cloud computing platform. *Applied Soft Computing*, 48:716–728.
- Barkat, A., Kazar, O., and Seddiki, I. (2021). Framework for web service composition based on qos in the multi cloud environment. *International Journal of Information Technology*, 13:459–467.
- Bei, X., Li, Z., and Luo, J. (2022). Fair and efficient multi-resource allocation for cloud computing. In *International Conference on Web and Internet Economics*, pages 169–186. Springer.
- Belgacem, A. (2022). Dynamic resource allocation in cloud computing: analysis and taxonomies. *Computing*, 104(3):681–710.
- Belgacem, A., Beghdad-Bey, K., Nacer, H., and Bouznad, S. (2020). Efficient dynamic resource allocation method for cloud computing environment. *Cluster Computing*, 23(4):2871–2889.
- Buyya, R., Yeo, C. S., and Venugopal, S. (2008). Market-oriented cloud computing: Vision, hype, and reality for delivering it services as computing utilities. In *2008 10th IEEE international conference on high performance computing and communications*, pages 5–13. Ieee.
- Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A., and Buyya, R. (2011). Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Software: Practice and experience*, 41(1):23–50.
- FAREH, M. E.-k. (2015). *Une approche basée agents pour l'allocation des ressources dans le Cloud Computing*. PhD thesis, Université Mohamed Khider-Biskra.
- Kumar, D., Mehrotra, D., and Bansal, R. (2018). Meta-heuristic policies for discovery task programming matters in cloud computing. In *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, pages 1–5. IEEE.
- Mahavidyalaya, K. and Nadu, T. (2021). A novel multi-agent approach to control service level agreement violations in cloud computing. *Turkish Journal of Computer and Mathematics Education*, 12(12):1432–1438.

- Qasim, A., Amin, H., Aziz, Z., and Khalid, A. (2020). Efficient performative actions for e-commerce agents. *Applied Computer Systems*, 25(1).
- Rabaoui, S., Hachicha, H., and Zagrouba, E. (2021). Multi-agent-based framework for resource allocation in cloud computing. In *Agents and Multi-Agent Systems: Technologies and Applications 2021: Proceedings of 15th KES International Conference, KES-AMSTA 2021, June 2021*, pages 427–437. Springer.
- Singh, A., Juneja, D., and Malhotra, M. (2017). A novel agent based autonomous and service composition framework for cost optimization of resource provisioning in cloud computing. *Journal of King Saud University-Computer and Information Sciences*, 29(1):19–28.
- Singh, A. and Malhotra, M. (2015). Agent based resource allocation mechanism focusing cost optimization in cloud computing. *International Journal of Cloud Applications and Computing (IJCAC)*, 5(3):53–61.
- Soltane, M., Cardinale, Y., Angarita, R., Rosse, P., Rukoz, M., Makhlouf, D., and Okba, K. (2018). A self-adaptive agent-based system for cloud platforms. In *2018 3rd International Conference on Pattern Analysis and Intelligent Systems (PAIS)*, pages 1–8. IEEE.
- Venkateshwaran, K., Malviya, A., Dikshit, U., and Venkatesan, S. P. (2015). Security framework for agent-based cloud computing. *IJIMAI*, 3(3):35–42.
- Wang, W., Jiang, Y., and Wu, W. (2016). Multiagent-based resource allocation for energy minimization in cloud computing systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47(2):205–220.
- Yahya, Z. B., Ktata, F. B., and Ghedira, K. (2020). Cloud ma-morbac: A cloud distributed access control model based on mobile agents. *Int. J. Web Appl.*, 12(1):1–15.