Hybrid Solution for Container Placement and Load Balancing based on ACO and Bin Packing

Oussama SMIMITE[1]
LabSIV, Department of Computer Science
Faculty of Science, Ibn Zohr University
BP 8106, 80000 Agadir, Morocco

Karim AFDEL[2]
LabSIV, Department of Computer Science
Faculty of Science, Ibn Zohr University
BP 8106, 80000 Agadir, Morocco

***Abstract*—Currently, data centers energy consumption in the cloud is attracting a lot of interest. One of the most approaches to optimize energy and cost in data centers is virtualization. Recently, a new type of container-based virtualization has appeared, containers are considered very light and modular virtual machines, they offer great flexibility and the possibility of migration from one environment to another, which allows optimizing applications for the cloud. Another approach to saving energy is to consolidate the workload, which is the amount of processing that the computer has to perform at any given time. In this article, we will study the container placement algorithm that takes into account the QoS requirements of different users in order to minimize energy consumption. Thus, we proposed a Hybrid approach for managing resources and workload based on ant colony optimization (ACO) and the first-fit decreasing (FFD) algorithm to avoid unnecessary power consumption. The results of the experiment indicate that using the first-fit decreasing algorithm (FFD) for container placement is better than ant colony optimization especially in a homogeneous systems. On the other hand the ant colony optimization shows very satisfying results in the case of workload management.**

***Keywords*—Cloud; virtualization; container; placement; Green IT; containerization**

I. Introduction

Recently, Cloud computing is considered as a new model that offered virtually immense resources. Customers can allocate resources as needed and pay as much as they have used. Thus, resources are managed by the cloud provider according to customer demand. In 2017, data centers in the United States consumed more than 90 billion kilowatt hours of electricity. Globally, data center power consumption was approximately 416 terawatts, or about 3% of all electricity produced on the planet.In a sense, the energy consumption of data centers worldwide was 40% more than all the energy consumed by the United Kingdom, an industrialized country with more than 65 million inhabitants. And this consumption will double every four years. [1, 2, 3, 4] From a business perspective, reducing energy consumption can lead to massive cost reductions. Moreover, in addition to the huge energy costs, heat dissipation inevitably increases with increasing energy consumption and doubles the probability of hardware failure [5, 6]. Therefore, reducing energy consumption not only saves a large amount of money and improves system reliability, but also helps protect our environment. According to [7], data centers emit CO2 like Argentina entirely, and their emissions are likely to exponentially increase in the coming years.

II. Background

There are different approaches to energy conservation. Besides the possibility of using more energy-efficient Hardware, reducing energy wasted due to the overuse of hardware is very important. The existing data center infrastructure is generally over-provisioned to maintain the availability of service during periods of high demand. However, the average use is low in datacenter due to tot he fort demand for resources in existing data centers.

Consequently, stopping or suspending unnecessary servers can impact a large part of the resources, which can influence performance constraints on clients.

Virtualization technologies such VMware, Xen, and Hyper V [8, 9, 10] are widely used in cloud datacenter due to their ease of use, flexibility of resource ,cost efficiency and the simplicity of enabling the high availability. More precisely, virtualization technologies offer the possibility of fine-tuning the resources allocation by associating processors, RAM, disk space and network bandwidth to a specific Virtual Machine [11, 12]. This approach has allowed the development of solutions such as Software as a Service (SaaS) and Platform as a Service (PaaS), on top of the usual Infrastructure as a Service (IaaS), where services providers can quickly make available virtual machines with the required resources to their customers almost in no time, and not burden them with the pain of infrastructure management.

Unlike traditional IT systems, Virtualization makes cloud computing more suitable for marketing, it provides promising approach to divide the resources of one or more physical servers into various parts and each part runs in an isolated environment [13]. To better manage resources, we can create isolated virtual machines (VMs) for each application, which allows us to parameterize the size of the resource such as the memory and the size of the processor according to the variable demand of the customers.

As virtualization allows us to create virtual instances, whether virtual machines (VM) or a container, the problem of virtual instances placement has become an important research subject in cloud computing. placement involves finding an optimal method for placing virtual instances on physical servers in order to efficiently use cloud resources [14, 15]. To maintain the servers in the data center, a lot of energy is consumed and the cost of cooling the installations is very high, which can translate to a very high cost [16]. Therefore, the goal of placement of virtual instances is to efficiently use physical resources to host virtual resources, in order to reduce the number of running physical servers. In a cloud environment, good placement of virtual instances means placing the VMs in a way that the service level agreement (SLA) is guaranteed without signaling losses at the provider level. Within a data center, it is possible to place VMs with the main objective of reducing energy consumption and cost. Several studies have been developed in this context. We will present them in the following.

Besides the importance of the placement of virtual instances in a cloud environment, load management is a crucial issue to be resolved in order to maintain the system stability and improve the reliability of the cloud environment. [17, 18, 19] Load balancing ensures that all system instance do roughly the same amount of work at all times.

The organization of the paper is as follows. in Section III ,Containerization technologies, the Placement problem ,and load balancing problem are discussed . Section IV covers the Proposed System Architecture and the two Proposed types of algorithms: First Fit Decreasing (FFD)algorithm as a classic algorithm and Ant colony optimization (ACO) algorithm as a metaheuristic algorithm. Section V presents a analysis of experimental results and evaluation.In the last section, conclusions and future work are discussed.

III. Formal Problem Definition

A. Containerization

Containerization, or virtualization that uses containers, is a technology that virtualizes hardware resources in a container and ships applications and their dependencies across multiple operating systems. at the time of migration, the containers guarantee that their content is identical, and that it is secure, thanks to the isolation.

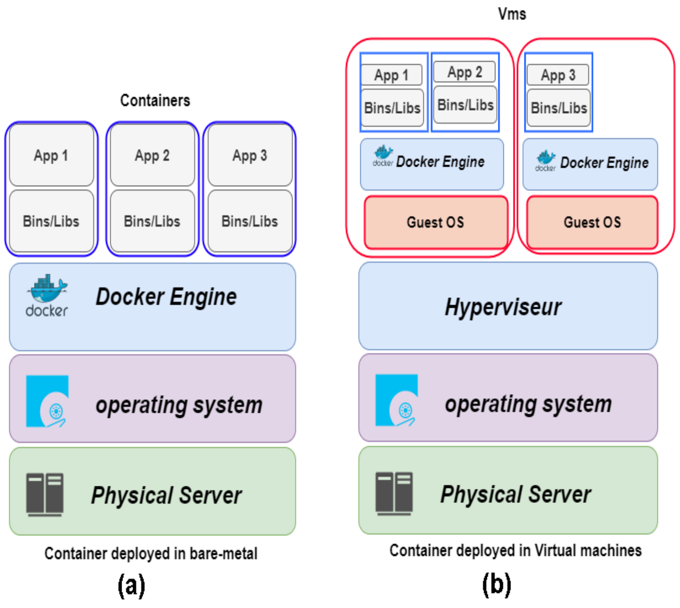

Fig. 1. Container Deployed in VMs VS Container Deployed in Bare-Metal.

1) Container deployed in Bare-Metal: Running containers on bare metal have many advantages, such as makes it possible to deploy applications with high performance in environments that can easily switch from the host server to another because there is no hardware emulation layer separating the containers from a host server [20, 21]. Furthermore, containerization allows for application isolation. Even if containers may not offer the same level of isolation as VMs, they set strict limits on the privileges and accessibility of resources associated with each container. But there are considerable difficulties that prevent the direct deployment of containers on the bare-metal host's server, such as the problem of updating physical servers. In fact, to replace a bare metal server, you must recreate the container environment from scratch while using virtualization makes it easy to migrate VMs to a new server. Another problem is that the containers depend on the type of the operating system, for example, Linux containers run in Linux hosts and Windows containers run on Windows hosts, also there are a few hosts that offer bare-metal solutions,most cloud platforms require VMs (see Fig. 1(a)).

2) Container deployed in virtual machines: deploying containers on Vms, offer advantages such as applications can be easily moved from one host to another by transferring images from one server to another. from a security point of view, applications that run in different VMs are isolated What makes management easier [22]. Also the possibility of grouping the same types of containers in a VM which allows creating a more coherent system. But virtual machines also have disadvantages such as under utilization of resources because of the pre-allocation of these resources even if not used. also, the VMs cannot directly access the physical hardware in order to unload it in the event of an overload (see Fig. 1(b)).

B. Container Placement Problem

Virtualization is a technique used to take better advantage of hardware resources, so it is useful for deploying more test environments thanks to the use of virtual machines. However, the appearance of containers has been evaluated as an improvement of virtualization. For this, we can consider the container placement as an improved version of virtual machine placement, to better managing resources in a cloud environment.The placement of containers is an important operation that has a direct effect on resource utilization, energy consumption, and Resource utilization cost. an efficient placement optimizes the use of material resources by minimizing the number of physical machines active in a data center, which allows both to minimize the cost of resources utilization and reduces energy consumption by stopping the inactive physical machine.

C. Load Balancing Problem

load balancing is an approach of distributing workloads across various computing resources to improve response time and resource utilization. load balancing is used to balance the load between the different resources of the system to avoid having idle resources and overused resources. [23] in a homogeneous environment where all resources are identical, the load must be distributed equally.but in heterogeneous environments, resources that have more capacity should be used more than other resources.

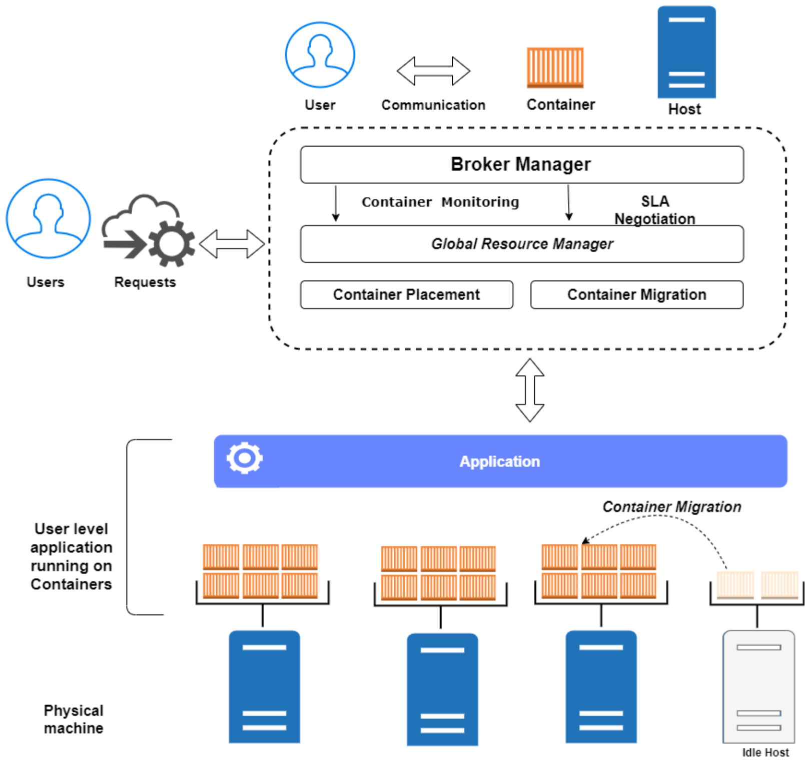

Fig. 2. Proposed System Architecture.

IV. Proposed System and Algorithms

A. Proposed System Architecture

Fig. 2 shows the proposed architecture of the system, which consists of:

1) The Brokers: The Broker is the main component of the System, it serves as an agent between service providers and customers. he may also sign SLA terms with cloud providers in place of the client. When the customer request has been sent, the broker initiates the process and submits those requests to the other modules. after that, based on information provided by the monitor module he makes the decision whether to approve or reject the request. It provides an interface for multiple clouds and sharing resources.

2) The Monitor of containers: This module's key role is to control resource state, RAM use, processor utilization, SLA violation, and power consumption. Once an exception in resource usage is triggered, The controller must send a warning to The Global Resources Manager to take necessary action.

3) The Global Resource Manager: The main function of this module is to analyze user requests and check the QoS requirements before choosing to accept or refuse the request. To guarantee that no SLA violation persists the module requests updated information from the Containers Monitor to reallocate resources efficiently.

4) The Physical Machines (PM): Also known as the ”bare-metal server” is the physical machine, which is the support of hardware to create and host Container. The PM can host several Containers depending on their capacity.

5) The Containers (CNT): Containers are the layer of virtualization that runs within the operating system. Therefore containers are relatively light and take only seconds to get started, unlike VM. The speed, flexibility, and portability of containers permit them to Help optimizing software development. Process of transferring Containers from one PM to another called Container migration.

B. Proposed Algorithms

C. First Fit Decreasing algorithm (FFD)

The bin packing In operational research is an algorithmic problem that involves storing objects with a minimum number of boxes. It can be applied in IT such as the storage of files on IT support.[24] To solve the bin packing problem, we often use simple algorithms like first-fit decreasing (FFD) which works as follows: we sort the list of articles in decreasing order of size, then we put each article in order. In first-fit, we put the current article in the first box that can contain it (see Fig. 3) . This algorithm allow to obtain very good results in practice. In our case, we used the FFD algorithm to allocate the containers in the hosts as the Container placement is considered a Bin Packing problem [25, 26, 27]. The bin represents the physical machine and the items are the containers to be assigned to the Bin. The containers are sorted first in descending order of their Ram memory capacity. The pseudo-code of containers placement is presented in algorithm 1.

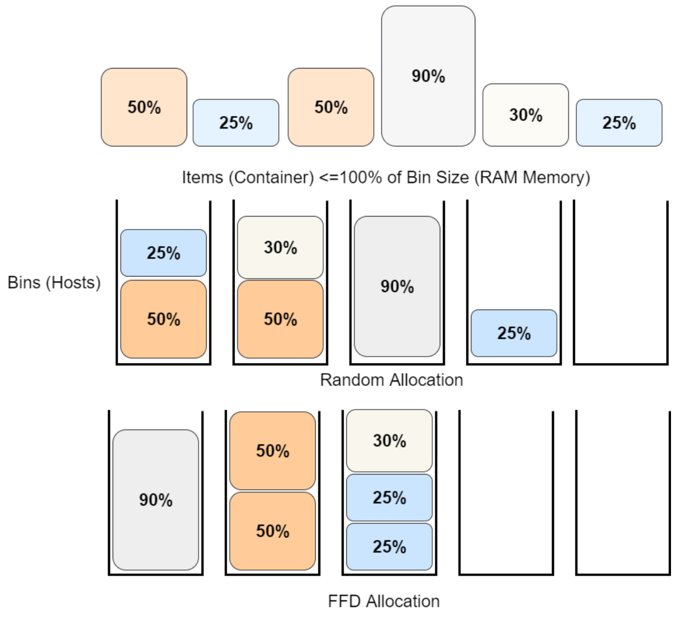

Fig. 3. FFD illustration example.

Algorithm 1 Container placement (FFD).

```
Require: ContainerList ,HostListOutput /: ContainerPlacement
  Sort the containers in descending order according to the RAM memory
  while container in containerList do
    if $container.Ram <$ container.Ram.next then
      Exchange (container, container.next)
    end if
  end while
  max=VmRAM *threshold
  while Host in HostList do
    while container in containerList do
      if $VmRamEstimate(Vm, container) <$ max then
        Allocate (Host ,Container)
      end if
    end while
  end while
```

D. Ant Colony Optimization algorithm (ACO)

Ant Colony Optimization (ACO) is a meta-heuristic inspired from the natural food-discovery behavior of real ants. due to the limited memory of the ants they have developed a system of communication based on chemical substance called pheromone, this last is used by each ant to mark their tracks .Other ants can smell the concentration of this substance and chose the paths probabilistically according to the quantity of the pheromone. after a while, the entire ant colony move towards the shortest path to the food source. at first, the algorithm was developed to solve the Traveling Salesman Problem (TSP). after that, it has been successfully adapted to solve many other complex combinatorial optimization problems.in our case, we try to use this algorithm to solve the placement problem, and also the load balancing issues [28]. The pseudo-codes of containers placement and the load balancing are presented consecutively in algorithm 2 and algorithm 3. The containers placement is represented as a graph g= (N, E) where N is the Set of Containers and the physical machines, and E represent the connections between Containers and physical machines as mentioned in Fig. 4(A) we can also represent the load balancing issue as graph G1= (N1, E1) where N1 is the set of Tasks and containers and E1 the connections between the task and containers as shown in 4(B)

1) ACO Container Placement: In virtualization, once the virtual machine starts, the RAM memory (R) allocated by the VM becomes unavailable for the physical machine.for that in our approach, we focus on Ram memory as being an important parameter for placing containers in the host.

In the proposed algorithm for containers placement, each ant receives all the containers and try to assign them to the host using the probabilistic decision (Prob) rule mentioned equation 1

$$Prob_{Host} = PH(Host)^{alpha} * H(Host)^{beta} \tag{1}$$

the probabilistic calculation (Prob) is based on the present concentration of pheromone (PH), and a heuristic (H) which help ants to chose the most optimizing hosts.Besides, two parameters $alpha, beta \geq 0$ are used to point out more the heuristic information or the pheromone.

for each container(CNT) we calculated the possible ram allocation (RA) for every host(PM) using the equation 2 we can represented the results as a graph $G1 = (C, ((H1, A1), (H2, A1), .., (Hn, An)))$ whereby, C is the container,H is the host and A is the possible allocation for every host.

$$RA_{CNT}^{PM} = CNT_R / PM_R \tag{2}$$

the heuristic (H) is calculated based on 2 as mentioned in equation 3 we can also represented that as a graph $G2 = ((H1, \sum A/A1), (H2, \sum A/A2), .., (Hn, \sum A/An))$ whereby, H is the host and $\sum$ A is a is the sum of the possible allocation of each container .

$$H_{CNT}^{PM} + = \sum RA(CNT)/RA(CNT)_{PM} \tag{3}$$

the pheromone (PH) concentration of the host is initialized by the RAM memory capacity of each host .after a host has been chosen by an ant the pheromone concentration is updated according to the equation 4

$$PH^{BestPM} = PH(BestPM)*(1-rho) + Q/RA(BestPM) \tag{4}$$

as, the constant rho,$0 \leqslant rho \leqslant 1$ is used to simulate pheromone evaporation.and Q is an adaptive parameter. in the end, we compare all ants allocation proposition for each container and choose the best solution.

2) ACO load balancing: As mentioned before, the objective of this approach is to distribute the workloads on the cloud resources in a balanced way.to do this we consider the workload as a list of cloudlets that must be run on a set of containers. In the proposed algorithm for load balancing, each ant receives all the cloudlets (Cl) the and try to execute them in the appropriate container(CNT) using the probabilistic (Prob) decision rule mentioned equation 5

$$Prob_{CNT} = PH(CNT)^{alpha} * ET(CNT)^{beta} \tag{5}$$

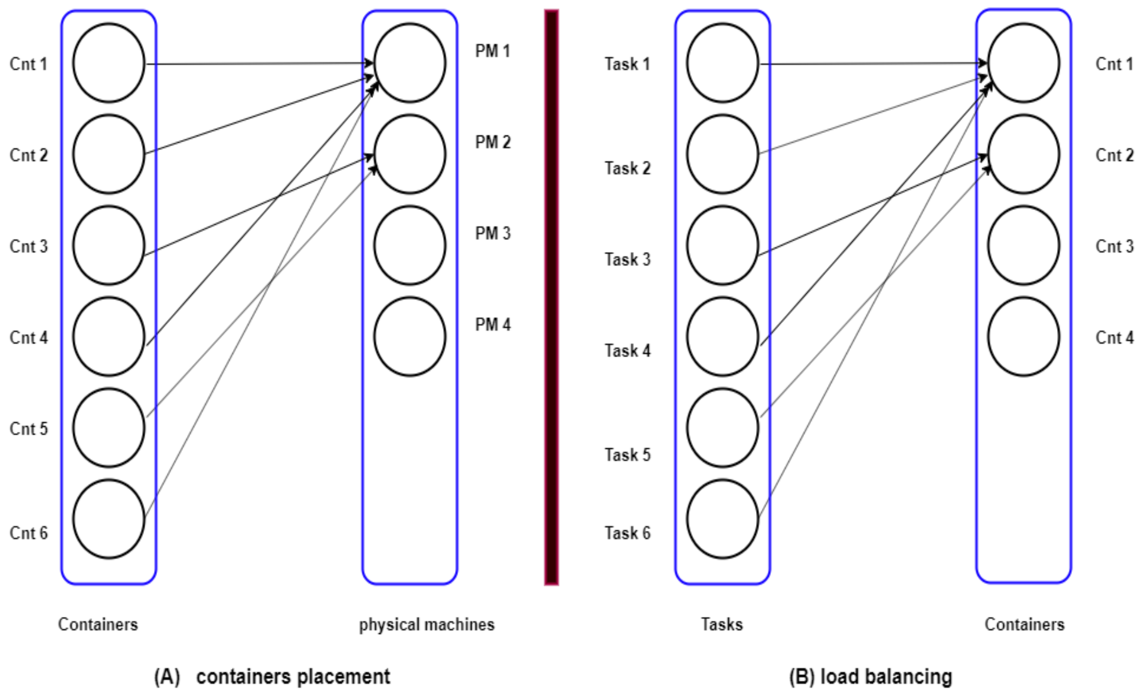

Fig. 4. Problem representation Based on ACO.

Algorithm 2 ACO-based Container placement

```
Require: ContainerList ,HostList Output /: ContainerPlacement
  Initialize parameters, Set pheromone value
  for all ContainerList do
    for all HostList do
      T= container.getRam()/Host.getRam()
    end for
    initializeRamAllocation(T);
  end for
  for all HostList do
    C = host.getRam();
    initializePheromone(C);
  end for
  for t=1 to t=tmax  do
    for all ContainerList do
      calculate allocation percentage of each container according to Eq 3
      for all HostList do
        Calculate probability according to Eq 1
        initializeProbability(P);
      end for
      Compare ants solutions and vote for best solution
      for i=1 to k do
        vote(Hosts,probab)
      end for
      for all HostList do
        if $MaxVote$ then
          Allocate (BestHost ,Container)
          UpdatePheromones according to Eq 4
        end if
      end for
    end for
  end for
```

the probabilistic calculation (Prob) is based on the present concentration of pheromone (PH), and a heuristic (ET) which help ants to chose the most optimizing container. Besides, two parameters $alpha, beta \geq 0$ are used to point out more the heuristic information or the pheromone.

for each cloudlet, we calculate the estimate Execution time for every container using the equation 6 we can represented the results as a graph $G3 = (Cl, ((C1, T1), (C2, T1), .., (Cn, Tn)))$ whereby, Cl is the cloudlet ,C is the container and T is the estimate Execution time for every container.

$$ET_{Cl}^{CNT} = Cl_{Length}/(CNT_{NbrPes} * CNT_{Mips} \tag{6}$$

the heuristic (H) is calculated based on 6 as mentioned in equation 7 we can also represented that as a graph $G4 = ((C1, \sum T/T1), (C2, \sum T/T2), .., (Cn, \sum T/Tn))$ C is the container and $\sum$ T is a is the sum of the estimate Execution time of each cloudlet.

$$H_{Cl}^{CNT} += \sum ET(Cl)/ET(Cl)_{CNT} \tag{7}$$

the pheromone concentration of the host is initialized by the Number of MIPS of each container. After a container has been chosen by an ant the pheromone concentration is updated according to the equation 8

$$PH^{BestCNT} = PH * (1 - rho) + Q/ET(BestCNT); \tag{8}$$

as, the constant rho,$0 \leqslant rho \leqslant 1$ is used to simulate pheromone evaporation, and Q is an adaptive parameter. in the end, we compare all ants proposition for each Cloudlet and choose the best solution.

Algorithm 3 ACO-based Load Balancing

```
Require: CloudletList ,HostList Output /: Load Balancing
  Initialize parameters, Set pheromone value
  for all CloudletList do
    for all ContainerList do
      T=Cloudlet.Length()/(Container.NbrPes()*Container.Mips())
    end for
    initializeExecTimes(T);
  end for
  for all ContainerList do
    C = Container.NbrPes()*Container.Mips()
    initializePheromone(C);
  end for
  for t=1 to t=tmax do
    for all CloudletList do
      calculate estimate Execution time of each Cloudlet according to Eq 6
      for all ContainerList do
        Calculate probability according to Eq 5
        initializeProbability(P);
      end forCompare ants solutions and vote for best solution
      for i=1 to k do
        vote(Container,probab)
      end for
      for all ContainerList do
        if $MaxVote$ then
          Allocate (BestContainer,Cloudlet)
          UpdatePheromones according to Eq 8
        end if
      end for
    end for
  end for
```

V. Experimental Setup and Results

To evaluate our proposed method, simulation experiments are implemented on CloudSim[29, 30] to study the effects.

We tried to apply the two proposed algorithms (ACO and FFD) on the Container placement and load balancing.

For the Container placement we adapt two approach , in The first we use a homogeneous system, where hosts has the same characteristic (CPU, RAM, Bandwidth), in the other approach we use hosts with different characteristics for the load balancing, in the heterogeneous system, we use a different types of containers, and in the other scenario we use many identical containers .

A. Containers Placement

1) Homogeneous system: In this experiment, our configuration consists in using 3 identical hosts, 90 containers as shown in the Table I.

a) Scenario 1: In this Scenario,we apply the FFD algorithm to assign all container in the hosts based on RAM memory. Container placement can be considered a Bin Packing problem,The bin represents the Host and the items being the containers to be assigned to the Bin. The containers are sorted first in descending order of their Ram memory capacity Before applying the FFD algorithm to the allocation problem,

TABLE I. Characteristics of hosts and containers in experiment 1.

	Number	MIPS	RAM	BW
Host -type 1-	3	37274/8	32768	1000000
CONTAINER -type 1-	21	2358	128	2500
CONTAINER -type 2-	23	4658	256	2500
CONTAINER -type 3-	23	9320	512	2500
CONTAINER -type 4-	23	18636	1024	2500

we should first make sure that the total memory capacity of the containers does not exceed the host's available memory capacity. in order to avoid a probable performance degradation because of the overloading of ram memory, we define a maximum threshold, 80% of host RAM memory, which can reduce the risk of Service Level Agreement (SLA) violation.

b) Scenario 2: In this Scenario, we repeat the previous scenario but we sorted first all hosts in descending order of their Ram memory capacity before applying the FFD algorithm.

c) Scenario 3: In this Scenario,we apply the ACO algorithm and inspected their efficiency by experimentation. The parameters (alpha, Beta,rho, tmax, m the number of ants and Q) considered here are those that affect directly or indirectly the computation of the algorithm Table II. showing the selected ACO parameter. in order to avoid a probable performance degradation because of the overloading of ram memory, we define a maximum threshold, 80% of host RAM memory, which can reduce the risk of SLA violation.

TABLE II. Selected Parameters of ACO

Parameter	alpha	beta	rho	Q	m	Tmax
Value	1	2	0.7	100	10	100

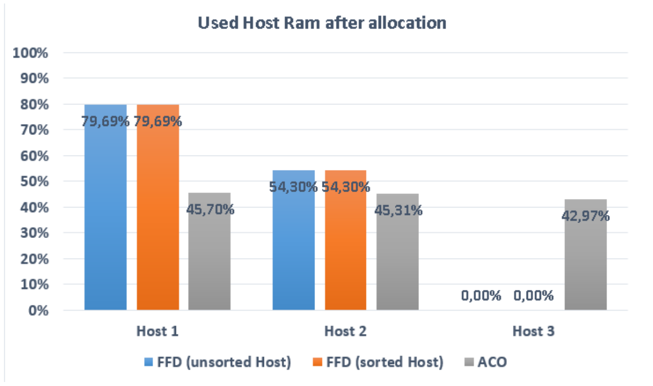

Fig. 5. Used Host Ram after Allocation (Homogeneous System).

As it is presented in Fig. 5 using the FFD algorithm allows us to use the totality of RAM memory of host 1 and reduce the total number of Hosts used. on the other hand, when using ACO algorithm the three hosts are used and the Ram memory used of each host does not exceed 46

Regarding the distribution of RAM memory of the containers on the hosts, we notice that in the case of using the FFD

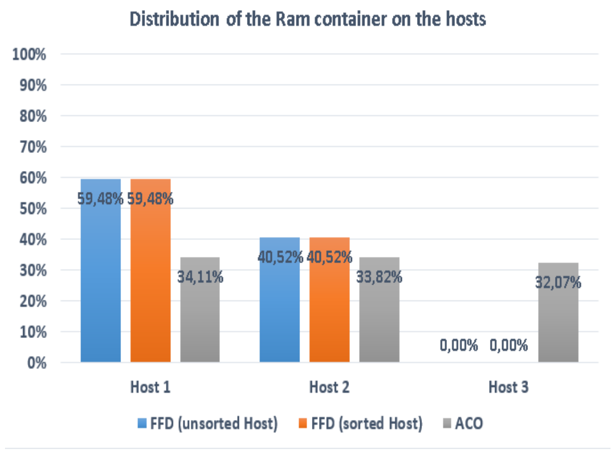

Fig. 6. Distribution of the Ram Container on the Hosts (Homogeneous System).

algorithm the first host has favored compared to the others hosts. on the other hand, when we use the ACO algorithm the distribution is done in a balanced way. (see Fig. 6)

2) Heterogeneous system: In this experiment, our configuration consists in using three different hosts, 80 containers as shown in Table III.

TABLE III. Characteristics of hosts and Containers in experiment 2 .

	Number	MIPS	RAM	BW
Host -type 1-	1	37274/8	32768	1000000
Host -type 2-	1	37274/4	16384	1000000
Host -type 3-	1	37274/2	8162	1000000
CONTAINER -type 1-	20	2358	128	2500
CONTAINER -type 2-	20	4658	256	2500
CONTAINER -type 3-	20	9320	512	2500
CONTAINER -type 4-	20	18636	1024	2500

In this approach, we repeat the three preceding scenarios (FFD sorted ,FFD unsorted and ACO) and compare the results.

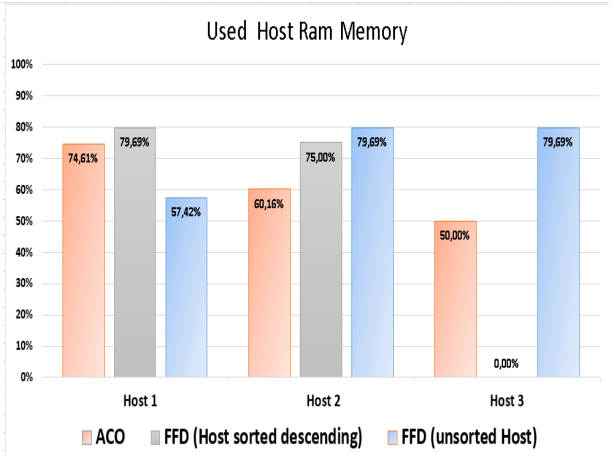

Fig. 7. Used Host Ram after Allocation (Heterogeneous System).

As it is shown in Fig. 7 when we use the FFD algorithm in heterogeneous unsorted system all hosts are used and the Ram memory used of Host 1 does not exceed 58%. but sorting host before applying the FFD algorithm allows us to use the totality of RAM memory of host 1 and reduce the total number of Hosts used. on the other hand, when using ACO algorithm the three hosts are used and the Ram memory used of each host depends on the characteristics of each host.

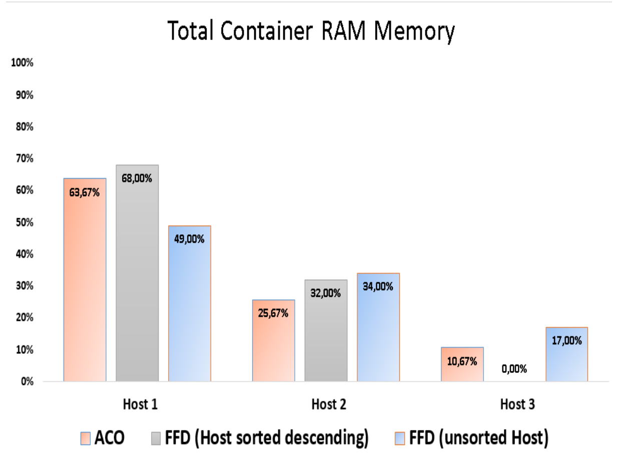

Fig. 8. Distribution of the Ram Container on the Hosts (Heterogeneous System).

Regarding the distribution of RAM memory of the containers on the hosts, we notice that in the case of using the FFD algorithm in an unsorted system the load which can handle by two hosts is distributed into three hosts, what is shown in the sorted system. on the other hand, when we use the ACO algorithm the distribution is done in a balanced way based on the percentage of each host in Ram's total memory (see Fig. 8).

To better evaluate the algorithms proposed or to propose a scenario where we combined between the two preceding systems ,we use several group of hosts as shown in Table IV.

TABLE IV. Characteristics of hosts and Containers in experiment 3 .

	Number	MIPS	RAM	BW
Host -type 1-	10	37274/8	32768	1000000
Host -type 2-	10	37274/4	16384	1000000
Host -type 3-	10	37274/2	8162	1000000
CONTAINER -type 1-	75	2358	128	2500
CONTAINER -type 2-	75	4658	256	2500
CONTAINER -type 3-	75	9320	512	2500
CONTAINER -type 4-	75	18636	1024	2500

The third experiment gives us a general idea of the usefulness of each algorithm. for container placement, the most important thing is to allocate all containers using the minimum host taking under consideration the maximum threshold for RAM usage Fig. 9 shows that the use of the FFD algorithm reduced the total number of hosts used from 30 hosts in the case of ACO to 17 hosts in an unsorted system and to 6 in a sorted system. which allowed us to optimize nearly 40% to 80% of the host used.

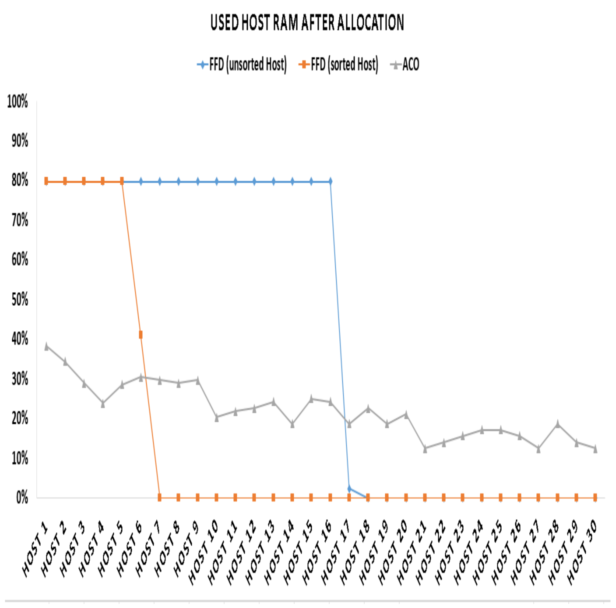

Fig. 9. Used Host Ram after Allocation.

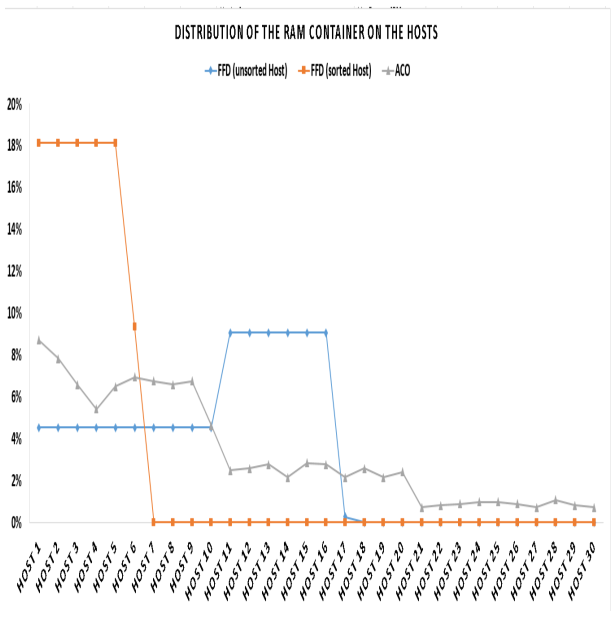

Fig. 10. Distribution of the Ram Container on the Hosts.

As shown in Fig. 10 the ACO curve can be divided into three parts according to the characteristics of the host. each part can be considered as a homogeneous system where the hosts are identical, this is why in each party the load is distributed in a balanced way.

To take advantage of the two algorithms, we propose a hybrid solution (ACO-FFD) where we apply first the FFD algorithm to optimize the number of hosts after that apply the ACO algorithm on the chosen hosts to balance the load between them. Fig. 11 and 12 show the results.

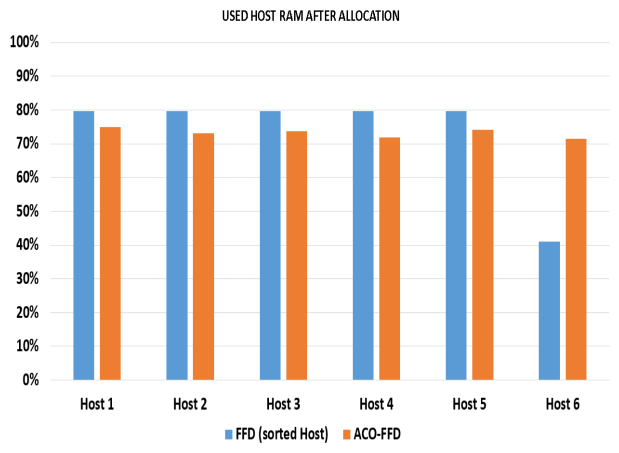

Fig. 11. Used Host Ram after Allocation (Hybrid Approach).

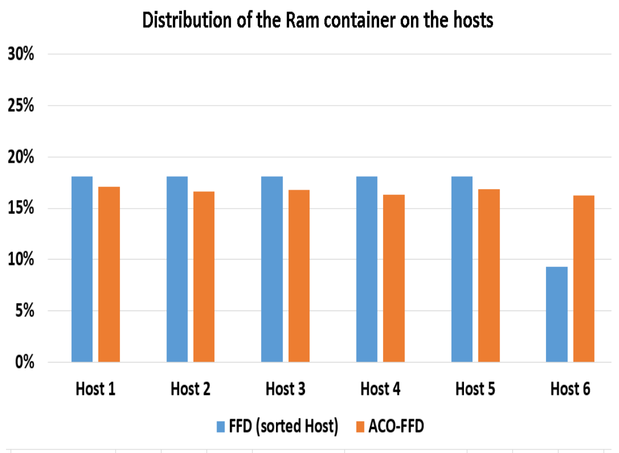

Fig. 12. Distribution of the Ram Container on the Hosts (Hybrid Approach).

B. Task Scheduling and Load Balancing

To assess the impact of our algorithm proposed on the task Scheduling and Load Balancing, we consider each task as cloudlet identified by its length must be performed in containers according to the MIPS number of each container.

1) Heterogeneous system: In this experiment, our configuration consists in using four different containers and 100 Cloudlet where their size varies between 100 and 400 mips as shown in Table V.

TABLE V. CHARACTERISTICS OF CONTAINERS AND CLOUDLETS IN EXPERIMENT 4.

	Number	MIPS	RAM	Lenght
CONTAINER -type 1-	1	2358	128	
CONTAINER -type 2-	1	4658	256	
CONTAINER -type 3-	1	9320	512	
CONTAINER -type 4-	1	18636	1024	
Cloudlet	100			100 /400

a) Scenario 1: In this Scenario, we apply the FFD algorithm to allocate all Cloudlet in Containers based on the length of each cloudlet and the Mips of each container. in order to avoid a probable performance degradation because of the overloading of container, we define a maximum threshold, 80% of container Mips, which can reduce the risk of degradation of the quality of service.

b) Scenario 2: In this Scenario, we apply the ACO algorithm and inspected their efficiency by experimentation. The parameters (alpha, Beta, rho, tmax, m the number of ants and Q) considered here are those that affect directly or indirectly the computation of the algorithm. Table VI showing the selected ACO parameter.

TABLE VI. SELECTED PARAMETERS OF ACO

Parameter	alpha	beta	rho	Q	m	Tmax
Value	1	2	0.7	100	10	100

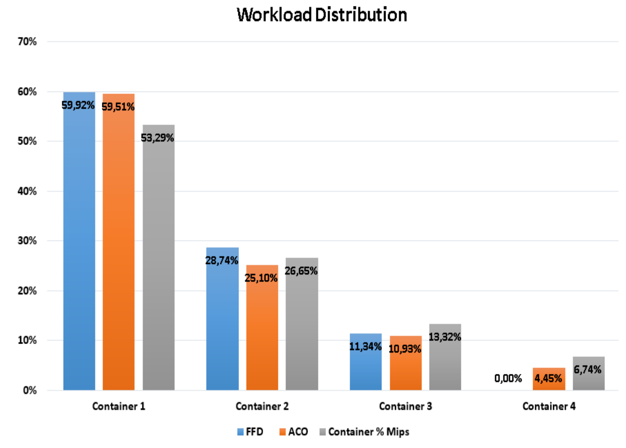

Fig. 13. Workload Distribution on Containers (Heterogeneous System).

Regarding the Workload Distribution on containers, we notice that in the case of using the FFD algorithm until we reach the maximum threshold, the first containers are favored compared to the others hosts and container 4 is not used. On the other hand, when we use the ACO algorithm the distribution is done in a balanced way based on the percentage of each host in MIPS's total (see Fig. 13).

2) Homogeneous system: In this experiment, our configuration consists using 100 Cloudlet where their size varies between 100 and 400 mips and four identical containers as shown in Table VII.

TABLE VII. CHARACTERISTICS OF HOSTS AND CONTAINERS IN EXPERIMENT 2 .

	Number	MIPS	RAM	Lenght
CONTAINER -type 3-	4	9320	512	
Cloudlet	100			100/400

In this approach, we repeat the two preceding scenarios (FFD and ACO) and compare the results.

As it is shown in Fig. 14 using the FFD algorithm we notice that almost 90% of the workload is managed by the first three containers and that just 11% is managed by container 4.

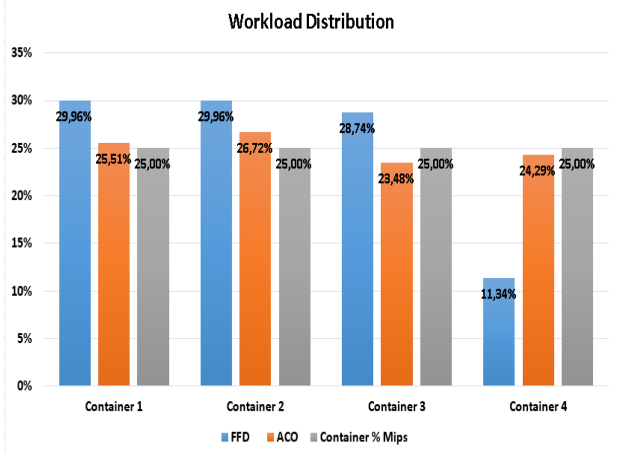

Fig. 14. Workload Distribution on Containers (Homogeneous System).

on the other hand, when we use the ACO algorithm the distribution is done in a balanced way. almost 25 % of workload for each container.

VI. CONCLUSION AND FUTURE WORK

In this work, we try to present Containerization technologies, the Placement problem, load balancing problem and there impact in a cloud environment. In addition, we provide a container allocation approach based on the ACO and FFD algorithm, taking into account QoS requirements and service level agreement. The use of the FFD algorithm has allowed us to better manage the placement of containers using a minimum number of hosts, which reduces power consumption. The use of ACO shows very acceptable results for a balanced workload management. At the end, a hybrid approach was proposed between the two methods in order to benefit from the advantages of each of these algorithms. As a perspective of our research work, we plan to track under-utilized Hosts by proposing solutions based on the Metaheuristic algorithm to optimize our architecture based on containerization.

REFERENCES

[1] A. Shehabi, S. Smith, D. Sartor, R. Brown, M. Herrlin, J. Koomey, E. Masanet, N. Horner, I. Azevedo, and W. Lintner, "United states data center energy usage report," Lawrence Berkeley National Lab.(LBNL), Berkeley, CA (United States), Tech. Rep., 2016.
[2] X. Zhang, T. Lindberg, N. Xiong, V. Vyatkin, and A. Mousavi, "Cooling energy consumption investigation of data center it room with vertical placed server," *Energy procedia*, vol. 105, pp. 2047–2052, 2017.
[3] W. Jiye, Z. Biyu, Z. Fa, S. Xiang, Z. Nan, and L. Zhiyong, "Data center energy consumption models and energy efficient algorithms," *Journal of Computer Research and Development*, vol. 56, no. 8, p. 1587, 2019.
[4] S. Pang, K. Xu, S. Wang, M. Wang, and S. Wang, "Energy-saving virtual machine placement method for user experience in cloud environment," *Mathematical Problems in Engineering*, vol. 2020, 2020.
[5] M. I. Green, "Cloud computing and its contribution to climate change," *Greenpeace International*, vol. 83, 2010.
[6] S. K. Mishra, B. Sahoo, and P. P. Parida, "Load balancing in cloud computing: a big picture," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 2, pp. 149–158, 2020.
[7] J. M. Kaplan, W. Forrest, and N. Kindler, "Revolutionizing data center energy efficiency," Technical report, McKinsey & Company, Tech. Rep., 2008.
[8] P. K. Das, "Comparative study on xen, kvm, vsphere, and hyper-v," in *Emerging Research Surrounding Power Consumption and Performance Issues in Utility Computing*. IGI Global, 2016, pp. 233–261.

[9] F. Rodríguez-Haro, F. Freitag, L. Navarro, E. Hernánchez-sánchez, N. Farías-Mendoza, J. A. Guerrero-Ibáñez, and A. González-Potes, "A summary of virtualization techniques," *Procedia Technology*, vol. 3, pp. 267–272, 2012.

[10] S. S. Kolahi, V. S. Hora, A. P. Singh, S. Bhatti, and S. R. Yeeda, "Performance comparison of cloud computing/iot virtualization software, hyper-v vs vsphere," in *2020 Advances in Science and Engineering Technology International Conferences (ASET)*. IEEE, 2020, pp. 1–6.

[11] S. J. Vaughan-Nichols, "New approach to virtualization is a lightweight," *Computer*, vol. 39, no. 11, pp. 12–14, 2006.

[12] R. Bachu, "A framework to migrate and replicate vmware virtual machines to amazon elastic compute cloud: Performance comparison between on premise and the migrated virtual machine," 2015.

[13] A. Abohamama and E. Hamouda, "A hybrid energy–aware virtual machine placement algorithm for cloud environments," *Expert Systems with Applications*, vol. 150, p. 113306, 2020.

[14] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *Journal of Computer and System Sciences*, vol. 79, no. 8, pp. 1230–1242, 2013.

[15] C. Vijaya and P. Srinivasan, "A hybrid technique for server consolidation in cloud computing environment," *Cybernetics and Information Technologies*, vol. 20, no. 1, pp. 36–52, 2020.

[16] M. Uddin, Y. Darabidarabkhani, A. Shah, and J. Memon, "Evaluating power efficient algorithms for efficiency and carbon emissions in cloud data centers: A review," *Renewable and Sustainable Energy Reviews*, vol. 51, pp. 1553–1563, 2015.

[17] A. Hota, S. Mohapatra, and S. Mohanty, "Survey of different load balancing approach-based algorithms in cloud computing: a comprehensive review," in *Computational Intelligence in Data Mining*. Springer, 2019, pp. 99–110.

[18] V. Priya, C. S. Kumar, and R. Kannan, "Resource scheduling algorithm with load balancing for cloud service provisioning," *Applied Soft Computing*, vol. 76, pp. 416–424, 2019.

[19] P. Kumar and R. Kumar, "Issues and challenges of load balancing techniques in cloud computing: a survey," *ACM Computing Surveys (CSUR)*, vol. 51, no. 6, pp. 1–35, 2019.

[20] I. Odun-Ayo, V. Geteloma, I. Eweoya, and R. Ahuja, "Virtualization, containerization, composition, and orchestration of cloud computing services," in *International Conference on Computational Science and Its Applications*. Springer, 2019, pp. 403–417.

[21] A. Abuabdo and Z. A. Al-Sharif, "Virtualization vs. containerization: Towards a multithreaded performance evaluation approach," in *2019 IEEE/ACS 16th International Conference on Computer Systems and Applications (AICCSA)*. IEEE, 2019, pp. 1–6.

[22] S. C. Mondesire, A. Angelopoulou, S. Sirigampola, and B. Goldiez, "Combining virtualization and containerization to support interactive games and simulations on the cloud," *Simulation Modelling Practice and Theory*, vol. 93, pp. 233–244, 2019.

[23] R. Mishra and A. Jaiswal, "Ant colony optimization: A solution of load balancing in cloud," *International Journal of Web & Semantic Technology*, vol. 3, no. 2, p. 33, 2012.

[24] L. Tadic, P. Afric, L. Sikic, A. S. Kurdija, V. Klemo, G. Delac, and M. Silic, "Analysis and comparison of exact and approximate bin packing algorithms," in *2019 42nd International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2019, pp. 919–924.

[25] X. Tang, Y. Li, R. Ren, and W. Cai, "On first fit bin packing for online cloud server allocation," in *2016 IEEE International Parallel and Distributed Processing Symposium (IPDPS)*. IEEE, 2016, pp. 323–332.

[26] H. I. Christensen, A. Khan, S. Pokutta, and P. Tetali, "Approximation and online algorithms for multidimensional bin packing: A survey," *Computer Science Review*, vol. 24, pp. 63–79, 2017.

[27] G. Dósa and L. Epstein, "The tight asymptotic approximation ratio of first fit for bin packing with cardinality constraints," *Journal of Computer and System Sciences*, vol. 96, pp. 33–49, 2018.

[28] E. Feller, L. Rilling, and C. Morin, "Energy-aware ant colony based workload placement in clouds," in *2011 IEEE/ACM 12th International Conference on Grid Computing*. IEEE, 2011, pp. 26–33.

[29] M. C. Silva Filho, R. L. Oliveira, C. C. Monteiro, P. R. Inácio, and M. M. Freire, "Cloudsim plus: a cloud computing simulation framework pursuing software engineering principles for improved modularity, extensibility and correctness," in *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*. IEEE, 2017, pp. 400–406.

[30] D. A. A. G. Singh, R. Priyadharshini, and E. J. Leavline, "Analysis of cloud environment using cloudsim," in *Artificial Intelligence and Evolutionary Computations in Engineering Systems*. Springer, 2018, pp. 325–333.