Power-Aware Allocation of Virtual Machine-Based Real-Time Cloudlets in Cloud Data Centers

Eman Elbedewy
Computer Science dept.
Faculty of Computers and Information,
Menofia University, Egypt
eman.elbedawy@ ci.menofia.edu.eg

Anas Youssef
Computer Science dept.
Faculty of Computers and Information,
Menofia University, Egypt
anas.youssef@ci.menofia.edu.eg

Arabi Keshk
Computer Science dept.
Faculty of Computers and Information,
Menofia University, Egypt
arabikeshk@yahoo.com

***Abstract*— Due to the expanding utilization of cloud computing services, power consumption in cloud data centers has increased significantly. The number of active physical hosts impacts data center power usage, so the number of active physical hosts should be decreased. To achieve this goal, cloud data centers use virtualization technology to consolidate multiple virtual machines on a single physical server, using state-of-the-art virtual machine placement algorithms. Specifically, bin packing algorithms have been widely used to place a set of items, i.e., cloudlets and virtual machines, into a set of bins, i.e., virtual machines and physical hosts. However, a set of cloud services, i.e., cloudlets, are characterized as real-time and need to be provided within strict deadlines. In this paper, a cloud resource allocation framework is proposed to provide a compromise between two goals. The proposed framework uses the optimal physical host MIPS to achieve minimum possible power consumption while satisfying virtual machine-based cloudlets' deadline constraints. The proposed framework includes two modules, namely cloudlet allocator and virtual machine allocator. A set of widely used bin packing algorithms is exploited and compared in both modules. Firstly, the algorithms exploited in the cloudlet allocator module include first-fit, best-fit, and round-robin. The evaluation results showed that the round-robin algorithm provides the best outcomes in terms of real-time constraints. Round-robin could allocate an increasing number of cloudlets to virtual machines without scarifying the deadline constraints. Secondly, the algorithms used in the comparison in the virtual machine allocator module include first-fit, best-fit, next-fit, and worst-fit. The results showed that the best-fit algorithm reduces power consumption among all other algorithms under consideration. The results also suggest that setting the physical host CPU MIPS to optimal MIPS achieves the least consumed power.**

Keywords—Power-Aware Resource Allocation, Cloudlet Allocator, Virtual Machine Allocator, Bin Packing.

I. INTRODUCTION

Due to its advantageous qualities, such as on-demand self-service, extensive network access, resource pooling, rapid elasticity, and measured service, cloud computing has gained much attention in recent years[1]. These services are available through virtualization techniques that support power reduction in data centers through virtual machine (VM) consolidation on physical hosts.

Power-aware cloud computing aims to reduce the power consumed in cloud data centers[2]. The number of data centers is increasing as cloud computing becomes more popular. Consequently, the reduction in power consumption is crucial [3]. Physical servers located in cloud data centers are among the components that exhaust the highest portion of the power consumed [4]. Therefore, it is required to decrease the number of active servers to save power in cloud data centers. This can be done by consolidating the largest possible number of VMs on the least possible number of physical hosts so that idle hosts can be switched off since powered on idle servers still consume power [5].

Real-time services are characterized by providing their output within time-constrained deadlines [6]. There are two types of real-time services: hard and soft. A hard real-time service should be provided on or before the deadline; otherwise, a catastrophic consequence will occur. A soft real-time service provides a tolerable penalty if the execution time exceeds the deadline [7]. This paper defines a real-time VM as a VM that serves real-time tasks, i.e., cloudlets. The focus of this work is on hard real-time cloudlets. Handling soft real-time cloudlets is left for future work.

This paper proposes a resource allocation framework to compromise between reducing power consumption and achieving real-time constraints. The proposed framework consists of two modules. The first module, named cloudlet allocator, aims to allocate cloudlets to VMs to fulfill their deadlines efficiently. While, the second module, named VM allocator, aims to allocate VMs to physical hosts to achieve minimum possible data center power consumption while still not violating cloudlets deadlines.

In this paper, a set of bin packing algorithms [8]has been compared to decide which algorithm is suitable for each of the two modules. Using cloudsim simulator [9], bin backing algorithms have been compared in each module. In the cloudlet allocator module, first-fit, best-fit, and round-robin have been compared. The evaluation results showed that round-robin provides the best results in terms of satisfying cloudlets' deadlines. In the VM allocator module, first-fit, best-fit, and worst-fit have been compared. The best-fit algorithm was found to be the best algorithm in this module in terms of reducing power consumption. Furthermore, keeping the physical hosts' CPU million instructions per second (MIPS) at their optimal levels of utilization reduces power consumption significantly.

This paper is organized in the following manner: Related work is presented in section II. In section III, the proposed framework is described. The power model used to obtain the evaluation results is described in section IV. In section V, the simulated bin packing algorithms are briefly described. In

section VI, the simulation setup is presented. In section VII, the simulation results are shown and discussed. Finally, conclusions and future work are listed in section VIII.

II. RELATED WORK

In this section, a set of related works is discussed. Fig. 1 shows VM placement schemes in cloud data centers. VM placement can either be done offline or online [10]. In offline VM placement, VMs are allocated to physical hosts before they start running. At the same time, in online VMs placement, VMs are allocated to physical hosts while they are up and running.

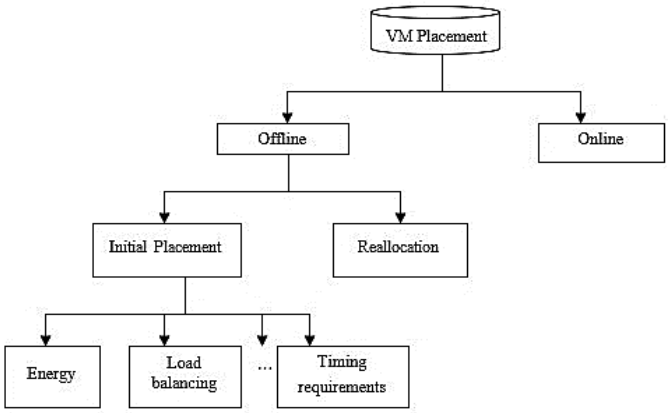

Fig. 1 Virtual Machine Placement Schemes

As shown in Fig. 1, offline VM placement on hosts can be subdivided into two steps[11], initial placement and reallocation. In this paper, the focus is on initial offline VM placement.

Initial VM placement means enrollment of new VM provisioning requests and placement of approved VMs on physical hosts [11]. In [12], The first-fit bin-packing algorithm is used to distribute VMs among hosts. However, it was only used to address the problem of power consumption.

In [13], the authors introduced a method of initial VM placement for setting redundant configurations to overcome host server failures.

VM reallocation means optimizing the placing of VMs using VM migration from hosts to other hosts[14]. In [15], cloud tasks are first distributed using the first-fit bin packing algorithm, then VMs are reallocated from underutilized hosts so that these hosts can be switched off for power saving. In [16], the authors used VM placement and reallocation to address the scalability in existing networks and data centers by locating large traffic chunks and then reducing the load at high-level switches. A method was proposed to reallocate VMs for load balancing among physical hosts in the cloud data center [17].

In [18], the authors proposed a VM selection policy that considers power consumption using the migration of VMs from underutilized hosts. In [19], the authors proposed a Power-Aware Next Fit Decreasing (PANFD) algorithm that selects a host to migrate VMs from other underutilized hosts so that such underutilized hosts can be turned off. All the work proposed in [14-19] used reallocation and migration of VMs among hosts. Reallocation and migration take time and are not suitable to satisfy tasks with hard real-time deadlines. Our proposed work is a combination of power and time restrictions which is quite different from the above-listed related work.

Bin Packing algorithms are widely used in the literature to support VM placement in cloud data centers. In[20], the VM placement problem is solved using the particle swarm optimization method with variable-sized bin packing for power optimization in the cloud data center. Finally, in [21], a VM placement method was proposed to save consumed power by minimizing CPU usage. The authors used a best-fit bin packing algorithm with particle swarm optimization to reduce switched-on physical machines.

In [20-21], the authors employed several VM placement algorithms to reduce power consumption. However, they did not target the objective of satisfying real-time deadlines of deployed tasks. On the contrary, as described previously, our proposed framework combines two modules to balance two conflicting objectives: reduction in hosts' power consumption and achieving cloudlet hard real-time deadlines.

III. PROPOSED FRAMEWORK

In this section, the proposed framework is presented. The proposed framework is shown in Fig. 2. The figure shows that the user first connects to a cloud platform to request a cloud task. Each cloud task is named a "cloudlet." Then, the broker invokes the cloudlet allocator module to allocate cloudlet(s) to an existing VM. When the cloudlet allocator accepts a certain cloudlet, it must be sure to complete all its instructions before its assigned real-time deadline. Thus, each cloudlet is characterized by two parameters: the length in a million instructions to be executed and the real-time deadline.

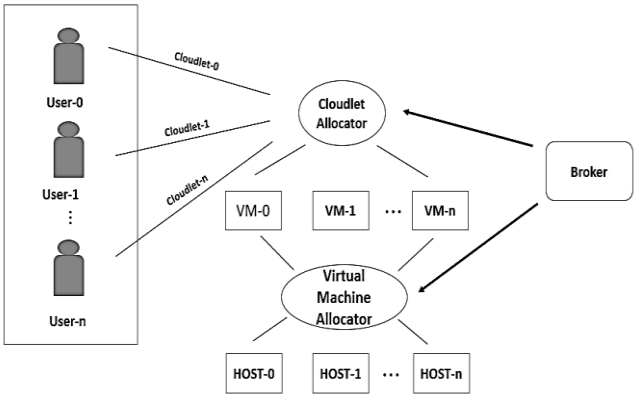

Fig. 2 Proposed Framework

As mentioned previously, the focus in this paper is on cloudlets whose real-time deadlines are hard. As described previously, cloudlets with hard deadlines must meet their deadlines; otherwise, a catastrophic cost will occur. Eq. (1) shows how the cloudlet allocator computes the MIPS needed for a cloudlet to finish at its deadline.

$$CloudletMIPS = \frac{Length}{Deadline} \quad (1)$$

where *CloudletMIPS* is the execution speed required by each cloudlet to run on a VM, the *length* is the number of instructions requested by the cloudlet, and the *deadline* is the time before a particular cloudlet must finish.

The broker then invokes the VM allocator module, which allocates the VMs obtained from the cloudlet allocator to

physical hosts. Each VM requires a particular CPU share of the physical host to complete its cloudlets. This share is translated to MIPS, known as VMMIPS. The following equation defines *VMMIPS* in terms of *CloudletMIPS* as defined in Eq. (2).

$$VMMIPS_{=} \sum_{i=0}^{n} CloudletMIPS_i \quad (2)$$

where VMMIPS equals the sum of *n* CloudletsMIPS that has been assigned to this VM. Each VM also owns a maximum MIPS value which represents the capacity of the VM in MIPS. The VM allocator ensures that each VM on its assigned host can run at its maximum MIPS.

A VM allocator looks for a host where each VM can get the MIPS it needs and finish the cloudlets given before the deadline. The aggregate MIPS of all VMs distributed to a host must not exceed the total CPU MIPS of the host.

The data center energy is reduced as much as possible in the proposed framework while still satisfying the *VMMIPS*. The VMs are consolidated on the least possible active hosts in the data center. To accomplish this, a set of bin packing algorithms are exploited and will be explained in detail later in the paper. A power model is defined since a set of bin packing algorithms in both the cloudlet allocator and the VM allocator modules is compared in terms of their consumed power. This power model is used to measure the power consumed by each bin-packing algorithm. The adopted power model is described in the next section. Any other power model can still be used in place of the adopted one.

IV. Power Model

We provide a complete description of a host's model of power consumption in this section. A host's total power is the summation of two different power values: static and dynamic power, as shown in Eq. (3).

$$P_{host} = P_{static} + P_{dynamic} \quad (3)$$

where the static power is the power used by a host when its CPU is idle. This power value remains constant as long as a host is turned on, whereas dynamic power consumption depends on CPU utilization, as described later.

The static power is defined by Eq. (4) as follows:

$$P_{static} = \alpha P_{max} \quad (4)$$

where P_{max} is the amount of power consumed when a host operates at its maximum CPU utilization, and α is the ratio of the static power of a host to its maximum power. It is a factor that ranges from 0 to 1 inclusive. During the duration that a host is turned on, this value remains constant. This factor depends on the host's specific physical characteristics. Many power models are suggested for calculating the host's dynamic power as a function of the host's CPU utilization [11, 12, and 22]. We utilized Eq. (5) in this paper to compute the dynamic power [12]. This power model is commonly used in the literature as well as in real-life systems[20-22].

In Eq. (5), as follows, the dynamic power that a host can utilize at a given time is a function in its CPU utilization. Therefore, in this paper, we utilized the CPU MIPS that provides the optimal power consumption.

$$P_{dynamic}(u) = (P_{max} - P_{static}) * u^2 \quad (5)$$

The total power consumed by a host is calculated as follows.

$$P_{host} = P_{max}\ [\alpha + (1-\alpha)u^2] \quad (6)$$

A host's energy consumption required to fulfill its particular amount of instructions is calculated as follows

$$E = [\alpha + (1-\alpha)u^2] \frac{P_{max} t_{max}}{u} \quad (7)$$

where t_{max} is the time for a host to run at its total processing capacity so as to finish a certain number of instructions. Eq.(7) indicates that the only variable that has an impact on total energy is the CPU utilization given a constant α value. By keeping α at a certain value, a specific CPU utilization will generate minimum energy if it is committed. For example, at α = 0.4, the optimal CPU utilization is 80% of the total CPU MIPS. By using the optimal CPU utilization u_{opt} [12] the overall consumed energy is decreased. Optimal CPU utilization is computed using the following equation:

$$u_{opt} = \sqrt{\alpha/(1-\alpha)} \quad (8)$$

Table 1 lists the amount of energy utilized when P_{max} =1000, t_{max}=1000, and α =0.2 are substituted in Eq. (7). When CPU utilization equals 50% of the total CPU MIPS, the minimum energy consumption is provided in the table. In this paper, each host's CPU MIPS is always kept at its optimal level. As a result of this constraint, less power is consumed.

Table 1: Energy consumption for each CPU utilization level at α =0.2

Utilization	Energy consumption (joule)
0.2	1160000
0.3	906666
0.4	628000
0.5	600000
0.6	621333
0.7	845714
0.8	890000

V. Simulated Bin Packing Algorithms

In this section, the simulated bin packing algorithms are described. The following is a definition of the bin packing problem. If n bins are given, i.e., VMs and hosts, with a fixed capacity, C, and a set of *m* items, i.e., cloudlets and VMs, each with a particular weight, w, where 0< w < C, it is required to pack all *m* items into *n* bins without exceeding the capacity of any bin [23]. Examples of bin packing algorithms are First-fit, best-fit, next-fit, and worst fit [21]. The ideas behind the four algorithms will be summarized as follows. First-fit attempts to pack all *m* items in the first active bin before going to the next bin. Best-fit selects a bin to leave the minimum possible empty bin capacity when the item is packed. Worst-fit selects the bin with the most remaining bin capacity after the item has been packed. Next-fit places an item in the same bin used for the previous item if it fits in the bin. Otherwise, a new bin is used. Finally, in the next-fit algorithm, the most recently selected bin is the only bin that will be used to pack a new item. The preceding described algorithms are used to implement both the cloudlet allocator and the VM allocator modules.

A. Cloudlet Allocator

The algorithms used to implement the cloudlet allocator module are first-fit, best-fit, and round-robin. The following

subsections elaborate on how to apply these bin packing algorithms to implement the cloudlet allocator module.

1) First-fit

A cloudlet is allocated to the *first* VM that fits its needs. First, *CloudletMips* is checked; if the *CloudletMips* is less than or equal to the remaining *VMMips* of a particular VM, this VM is used for allocation.

2) Best-fit

A cloudlet is allocated to the *best* VM that fits its needs such that this allocation leaves the least remaining *VMMips* after the cloudlet is allocated.

3) Round-robin

A cloudlet is allocated to the first available VM as long as it satisfies the cloudlet's needs. Afterward, the second cloudlet is allocated to the second available VM and so forth.

B. VM Allocator

The algorithms used to implement the VM allocator module are first-fit, best-fit, worst-fit, and next-fit, as explained in the following subsections.

1) First-fit VM allocator

Before moving on to the next active host, First-fit starts with the first host and tries to pack each VM into it., as shown in Fig. 3(a).

2) Best-fit VM allocator

Best-fit chooses a host such that minimum space will be left after the VM is packed into the host, as shown in Fig. 3(b).

3) Worst-fit VM allocator

Worst-fit chooses a host with the maximum space to be left after allocating the VM in that host, as shown in Fig. 3(c).

4) Next-fit VM allocator

If a VM fits in the same host as the previous VM, this VM is allocated to this host. Otherwise, a new host is used for the next allocation. A previously used host cannot be used for VM allocation. This is shown in Fig. 3(d).

VI. Simulation Setup

In this section, the simulation setup implemented to evaluate our proposed framework is presented. The proposed framework has been evaluated using a Cloudsim simulation environment [9]. Table 2 shows the parameters used in the simulation setup. Migration of cloudlets between VMs is not allowed. Also, migration of VMs between hosts is not allowed.

VII. Evaluation Results

A. Cloudlet Allocator

In this subsection, we first compare first-fit, best-fit, and round-robin bin packing algorithms in terms of the number of cloudlets that each algorithm can successfully allocate. Second, the host power consumption is compared when using each algorithm. All comparisons are made using selected α values and a different number of cloudlets

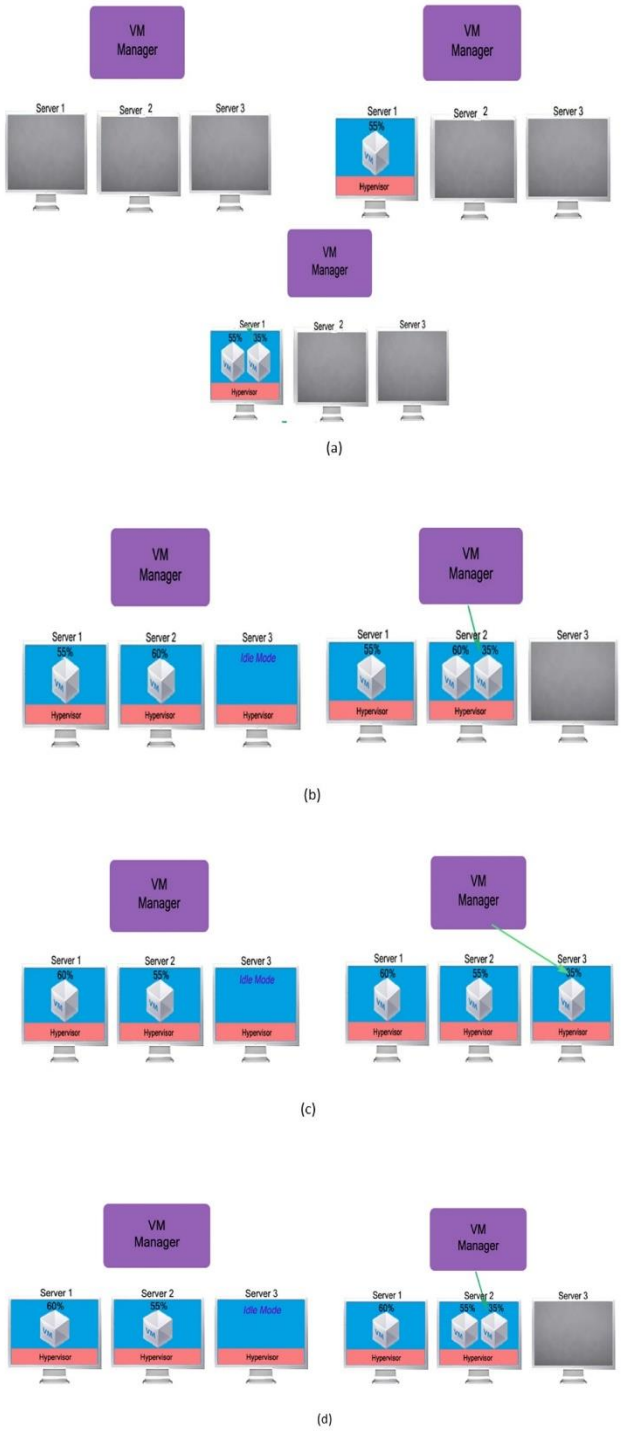

Fig. 3 An illustration of how to employ the exploited algorithms to pack VMs into hosts (a) First-fit (b) Best-fit (c) Worst-fit (d) Next-fit

1) *Number of allocated cloudlets*

In this section, the selected bin packing algorithms used to implement the cloudlet allocator module are compared in terms of the number of allocated cloudlets.

As shown in Fig. 4, each row represents the number of successfully allocated cloudlets by each algorithm. As shown in the figure, the first-fit algorithm fails to allocate all the needed cloudlets when the number of cloudlets that need to be allocated exceeds 200 at α = 0.3. The same case happens with the first-fit when the number of cloudlets that need to be

allocated exceeds 100 at both α = 0.5 and α = 0.7. In contrast, the best-fit and round-robin algorithms can allocate all needed cloudlets. In summary, the best algorithm to be used is either best-fit or round-robin when maximizing the number of allocated cloudlets is of much interest.

TABLE 2: SIMULATION SETUP PARAMETERS

Cloudlet Allocator Module Parameters	
Number of VMs	10
MIPS values	500, 1000, 2000
α	0.3, 0.5, 0.7
Number of cloudlets	50,100,200,300,400,500
VM Allocator Module Parameters	
P_{max}	1000
α	0, 0.2, 0.4, 0.8
Number of VMs	5,10,20,30,40
Number of hosts	2,4,6,8,10
Host Optimal Utilization Experiment	
Number of VMs	10
VMMIPS	500,1000, 2000
α	0.3,0.5, 0.7
Number of cloudlets	50,100,200,300

Number of Allocated Cloudlets

	α =0.3					
	50 cloudlets	100	200	300	400	500
FirstFit	50	100	200	207	222	177
BestFit	50	100	200	300	400	500
RoundRobin	50	100	200	300	400	500

	α =0.5					
	50 cloudlets	100	200	300	400	500
FirstFit	50	100	194	213	160	178
BestFit	50	100	200	300	400	500
RoundRobin	50	100	200	300	400	500

	α =0.7					
	50 cloudlets	100	200	300	400	500
FirstFit	50	100	189	237	201	191
BestFit	50	100	200	300	400	500
RoundRobin	50	100	200	300	400	500

Fig. 4 Number of allocated cloudlets using α = 0.3, α = 0.5 and α = 0.7

2) *Power Consumption*

This section compares the bin packing algorithms used to implement the cloudlet allocator module in terms of their power consumption. As shown in Fig. 5, in most cases, the best-fit algorithm gives the highest power consumption while the round-robin gives the minimal power consumption. The first-fit can only be used when the number of cloudlets that need to be allocated does not exceed 200 at α = 0.3 and does not exceed 100 at α = 0.5 and 0.7, as described previously.

B. *VM Allocator*

This section first compares first-fit,best-fit,worst-fit, and next-fit bin packing algorithms in terms of the number of required hosts needed to allocate a specific number of VMs. Second, host power consumption is compared for each algorithm. Again different α values and different numbers of VMs are used in the comparisons.

1) *Number of required hosts*

In this section, the bin packing algorithms used to implement the VM allocator module are compared in terms of the number of required hosts. Fig. 6 shows the number of hosts required for VM allocation by each of the bin packing algorithms used in the VM allocator module. The setup in this figure uses up to 10 available eight-core hosts. As shown in the figure, the best-fit algorithm used the smallest number of active hosts, while the first-fit used the highest number of active hosts. We would like to note that the first-fit algorithm was previously used in [12] and is here compared with our proposed work, as shown in Fig. 6.

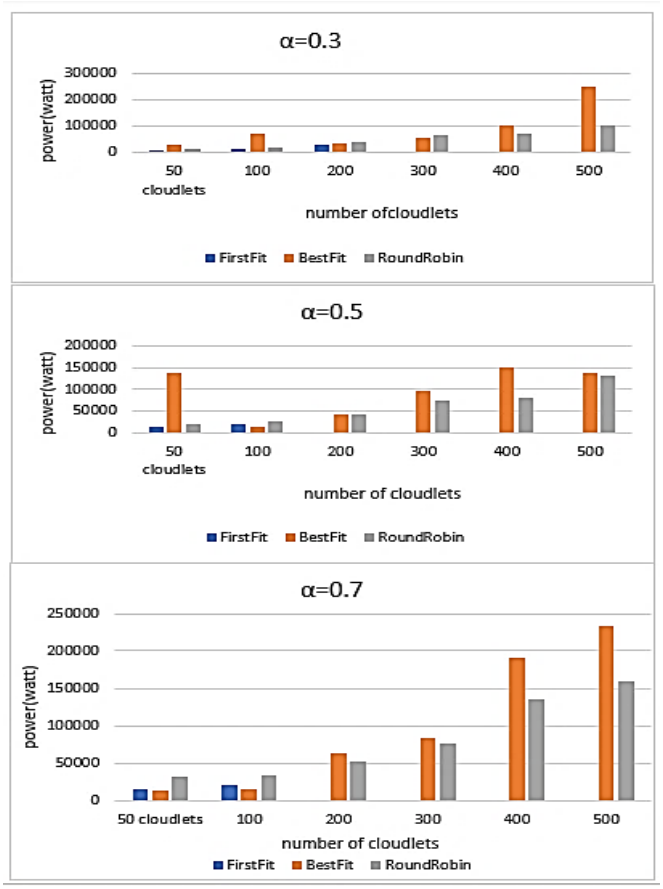

Fig. 5 Evaluation of power consumption for the cloudlet allocator algorithms

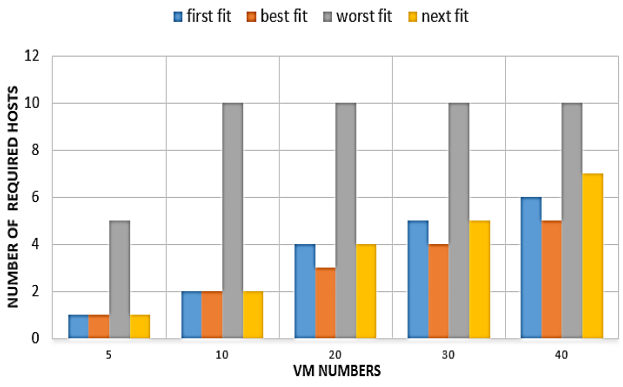

Fig. 6 Number of required hosts used to host VMs when the total number of available hosts is 10, with each host having eight cores

2) *Power Consumption*

Fig. 7 shows the evaluation results of the power consumption for the bin packing algorithms used in the VM allocator module. As shown in the figure, in all cases, the worst-fit algorithm consumes the highest power consumption

value. The next-fit algorithm consumes more power by increasing the number of VMs. The first-fit algorithm consumes roughly the same amount of power as the best-fit algorithm; however, it requires more hosts, as shown previously in Fig. 6. The first-fit algorithm consumes more power as the number of VMs is decreased. The best-fit algorithm consumes the least amount of power; even when the number of VMs grows, it provides a minor increase in power consumption.

Fig. 8 shows the power consumed by all VM allocator algorithms when α=0. As shown in the figure, the static power will be zero according to Eq. (4). Therefore, all algorithms should give the same power consumption when they operate at the same utilization level.

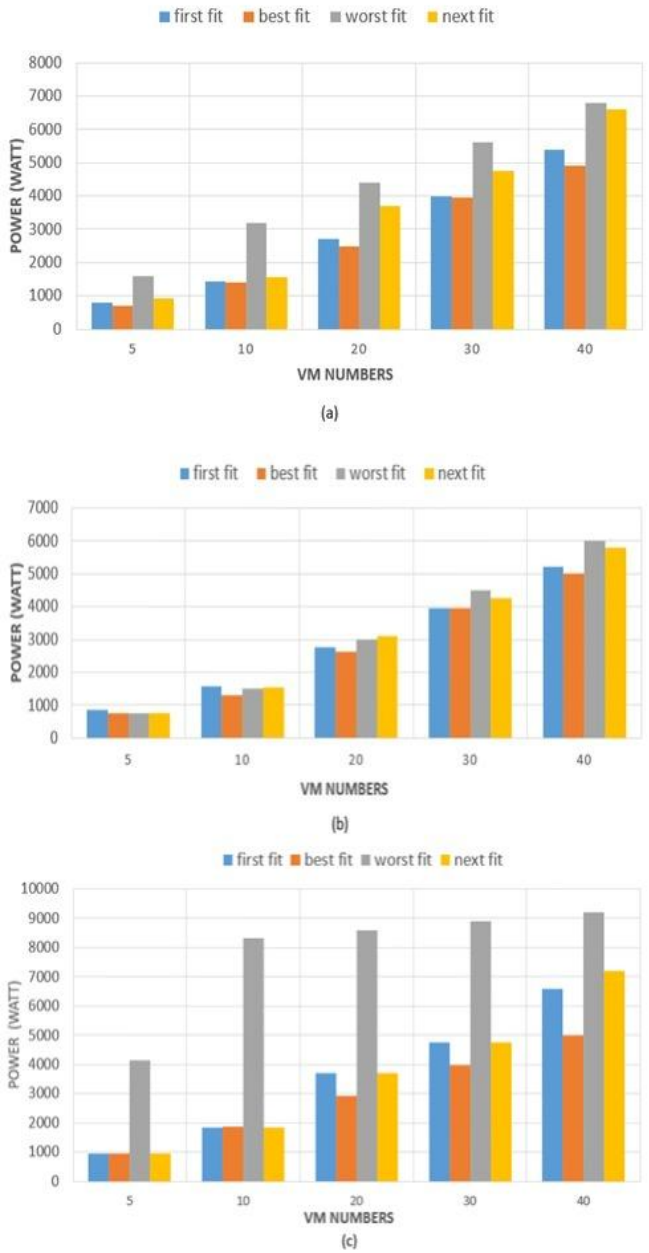

Fig. 7 Evaluating the power consumption of VM allocator bin packing algorithms

C. Proposed Framework General Steps

Fig. 9 depicts the proposed framework's general steps. Initially, Host MIPS is set to a certain u_{opt} generated by Eq. (8). Secondly, using the round-robin bin-packing algorithm, the cloudlets are allocated to VMs. Thirdly, the best-fit bin-packing algorithm is used to allocate VMs to hosts. Finally, the actual CPU utilization of the host is reviewed; if it is less than, u_{opt} , the VM allocator distributes the remaining MIPS among the VMs in a manner that maximizes performance. These steps maintain optimal CPU utilization levels for all hosts at all times.

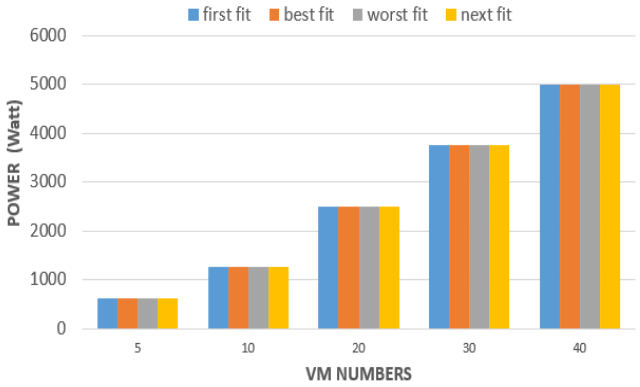

Fig. 8 VM allocator power consumption at α=0

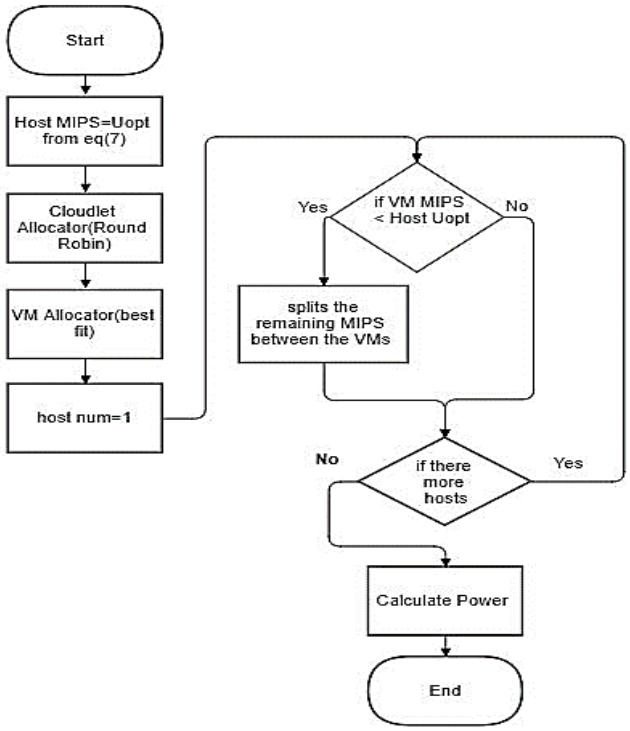

Fig. 9 Proposed Framework General Steps

As mentioned previously, a similar framework was proposed in [12]. In this work, all the steps shown in Fig. 9 were applied except for some differences. The first difference is that the VM allocator used the first-fit algorithm instead of the best-fit algorithm, which is used in our work. The second difference is, actually, the more important one where VMs

are allowed to utilize all host available host MIPS, and if VMs MIPS are found to be less than optimal host MIPs, the VM allocator distributes the remaining MIPS to other VMs. This represents a significant difference from our work where the host utilization MIPS is always set to the optimal host utilization and cannot by any means exceed this optimal host utilization value.

Fig.10 compares the power consumed when the work proposed in [12] is used versus the work proposed in this paper. The results confirm that operating hosts at their optimal CPU MIPS values consume less power in all α. values.

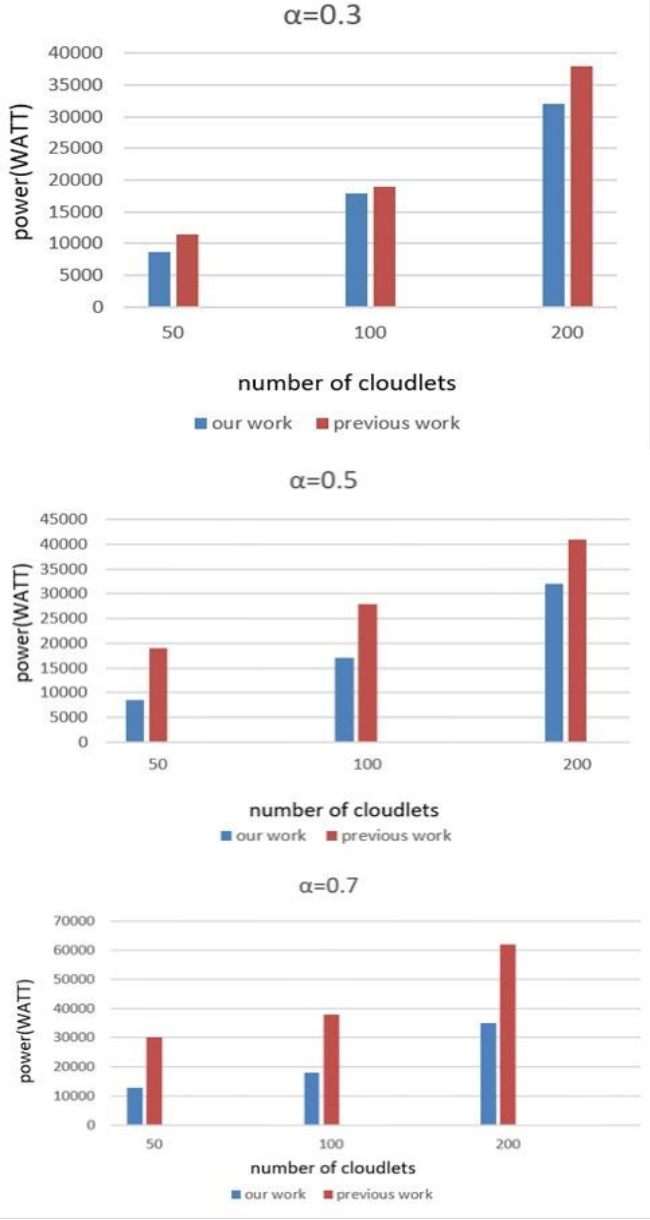

Fig. 10 Comparing power consumption with previous work in [12]

VIII. Conclusion and Future work

This paper proposes a framework to compromise between reducing power consumption and achieving real-time deadlines when allocating cloudlets to VMs and VMs to physical hosts. The proposed framework is composed of two modules, namely: cloudlet allocator and VM allocator. When comparing several bin packing algorithms, it was found that round-robin gives the best results in the cloudlet allocator module. At the same time, the best-fit gives the most reduction in power consumption in the VM allocator module. The least value for power consumption is generated by keeping the hosts' CPU utilization at their optimal levels.

Future work will apply soft real-time cloudlets where violation of the real-time deadlines of a few tasks can be allowed. Also, other host components like memory and hard disk utilization can be incorporated in addition to CPU utilization.

References

[1] N. Computer and S. Division, "The NIST Definition of Cloud Computing Recommendations of the National Institute of Standards and Technology," 2011.

[2] T. Kaur and I. Chana, "Energy aware scheduling of deadline-constrained tasks in cloud computing," *Cluster Comput.*, 2016.

[3] K. Khajehei, "Green Cloud and Virtual Machines Migration Challenges," *Indian J. Sci. Technol.*, vol. 9, no. 5, 2016.

[4] T. Case and E. Computing, "The Case for," *Computer (Long. Beach. Calif).*, vol. 40, no. 12, pp. 33–37, 2007.

[5] G. Chen *et al.*, "Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services," *NSD*, vol. 8, pp. 337–350, 2008.

[6] N. K. Jangid, "Real Time Cloud Computing," in *Data Management & Security*, 2011.

[7] K. H. Kim and A. Beloglazov, "Power-Aware Provisioning of Virtual Machines for Real-Time Cloud Services," *Concurr. Comput. Pract. Exp.*, vol. 23, no. 13, pp. 1491–1505, 2011.

[8] E. G. Co, M. R. Garey, and D. S. Johnson, "APPROXIMATION ALGORITHMS FOR BIN PACKING : A SURVEY," in *Approximation algorithms for NP-hard problems*, no. m, pp. 46–93, 1996.

[9] R. N. Calheiros, R. Ranjan, A. Beloglazov, and A. F. De Rose, "CloudSim : a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," in *4th IEEE international conference on advanced information networking and applications. IEEE*, pp. 23–50, 2010.

[10] K. Dubey, A. A. Nasr, S. C. Sharma, and N. El-bahnasawy, "Efficient VM Placement Policy for Data Centre in Cloud Environment Efficient VM Placement Policy for Data Centre in Cloud Environment," *Soft Comput. Theor. Appl.*, no. January, pp. 301--309, 2020.

[11] M. Masdari, S. S. Nabavi, and V. Ahmadi, "Author ' s Accepted Manuscript An Overview of Virtual Machine Placement Schemes In Cloud Computing Reference : An Overview of Virtual Machine Placement Schemes," *J. Netw. Comput. Appl.*, vol. 66, pp. 106–127, 2016.

[12] S. Hosseinimotlagh and F. Khunjush, "SEATS : smart energy-aware task scheduling in real-time cloud computing," *J. Supercomput.*, vol. 71, no. 1, pp. 45–66, 2015.

[13] F. C. S. Clusters, F. Machida, M. Kawato, and Y. Maeno, "Redundant Virtual Machine Placement for," in *2010 IEEE Network Operations and Management Symposium-NOMS.*, 2010, pp. 32–39.

[14] K. Braiki and H. Youssef, "virtual machine reallocation," *J. Supercomput.*, vol. 76, no. 1, pp. 427–454, 2020.

[15] A. Fayyaz and M. U. S. Khan, "Energy Efficient Resource Scheduling through VM Consolidation in Cloud Computing," in *13th International Conference on Frontiers of Information Technology (FIT). IEEE*, pp. 65–70, 2015.

[16] X. Meng, V. Pappas, and L. Zhang, "Improving the Scalability of Data Center Networks with Traffic-aware Virtual Machine Placement," in *Proceedings IEEE INFOCOM. IEEE*, pp. 1–9, 2010.

[17] S. Dhahbi, "Load balancing in cloud computing using worst-fit bin-stretching," in *Cluster Computing*, vol. 7, pp. 1–15, 2021.

[18] R. Mandal, M. K. Mondal, S. Banerjee, and U. Biswas, "An approach toward design and development of an energy-aware VM selection policy with improved SLA violation in the domain of green cloud computing," *J. Supercomput.*, vol. 76, no. 9, pp. 7374–7393, 2020.

[19] J. Sengupta, P. Singh, and P. K. Suri, *Energy Aware Next Fit Allocation Approach for Placement of VMs in Cloud Computing Environment*, vol. 1130 AISC. Springer International Publishing, 2020.

[20] A. F. Id, N. Javaid, T. Sultana, and W. Hussain, "Virtual Machine

Placement via Bin Packing in Cloud Data Centers," *Electronics*, vol. 7, no. 12, pp. 1–22, 2018.
[21] S. Wang, P. P. Huang, C. H. Wen, and L. Wang, "EQVMP : Energy-efficient and QoS-aware Virtual Machine Placement for Software Defined Datacenter Networks," in *The International Conference on Information Networking (ICOIN2014)*, pp. 220–225, 2014.
[22] S. Filiposka, A. Mishev, and C. Juiz, "Balancing Performances in Online VM Placement," in *International Conference on ICT Innovations*, pp. 153–162, 2016.
[23] E. Falkenauer, "A Hybrid Grouping Genetic Algorithm for Bin Packing," *J. heuristics*, vol. 2, no. 1, pp. 5–30, 1996.