

RESEARCH

Open Access



Look-ahead energy efficient VM allocation approach for data centers

İlksen Çağlar* and Deniz Turgay Altılar

Abstract

Energy efficiency is an important issue for reducing environmental dissipation. Energy efficient resource provisioning in cloud environments is a challenging problem because of its dynamic nature and varied application workload characteristics. In the literature, live migration of virtual machines (VMs) among servers is commonly proposed to reduce energy consumption and to optimize resource usage, although it comes with essential drawbacks, such as migration cost and performance degradation. Energy efficient provisioning is addressed at the data center level in this research. A novel efficient resource management algorithm for virtualized data centers that optimizes the number of servers to meet the requirements of dynamic workloads without migration is proposed in this paper. The proposed approach, named Look-ahead Energy Efficient VM Allocation (LAA), contains a Holt Winters-based prediction module. Energy efficiency and performance are inversely proportional. The energy-performance trade-off relies on periodic comparisons of the predicted and active numbers of servers. To evaluate the proposed algorithm, experiments are conducted with real-world workload traces from Google Cluster. LAA is compared with the best approach provided by CloudSim based on VM migration called Local Regression-Minimum Migration Time (LR-MMT). The experimental results show that the proposed algorithm leads to a consumption reduction of up to 45% to complete one workload compared with the LR-MMT.

Keywords: Cloud computing, Resource management, Energy efficiency

Introduction

Cloud computing is a collection of computer system resources that are dynamically provisioned to provide services to users based on demand access. Service providers offer customers three services, Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS), through data centers. As customers' needs for the services offered by data centers increase, the amount of energy consumed by data centers increases linearly. Increases of 48% and 34% are estimated for total world energy consumption and CO₂ emissions, respectively, between 2010 and 2040 [1]. Service providers try to reduce the energy cost in data centers due to both laws and regulations and standards. In addition, reducing the cost of the services and increasing the profit

rate are other goals of providers. On the other hand, users also want to have the same service with acceptable quality and less cost which are defined through Service Level Agreement (SLA). For this reason, while aiming to reduce energy consumption and cost, the performance of the service offered to users should also be considered. However, it is not easy to address the needs of the users and the resources that will meet these requirements.

Improving the energy efficiency of data centers has received significant attention in recent years. When cloud data centers are running at low usage levels of computing capacity without optimization, it causes high energy inefficiency. Many existing studies [2–5] have proposed running servers' computational units at full capacity to increase energy efficiency, but this causes performance degradation. To solve performance degradation caused by running at full capacity, the static optimal utilization threshold is defined for each resource type, including CPU, RAM, bandwidth and so on [6–9]. However, a static threshold may lead to

*Correspondence: ilksenozcan@gmail.com
Department of Computer Engineering, Istanbul Technical University,
Istanbul, Turkey

machines being turned on or off unnecessarily since the resource demand in the future is not considered. Another approach in the literature is server consolidation [10–13], which reduces the number of active physical machines through VM migration or by collocating VMs to a small set of physical machines. However, VM migration and server consolidation techniques cause low throughput from the perspective of the service consumer as well as energy overheads from the perspective of the service provider.

To address performance and cost issues, we propose an energy efficient resource allocation approach that integrates the Holt Winters forecasting model for optimizing energy consumption. The proposed approach includes a forecasting module to take into account not only the current situation but also the potential customer resource needs in the future. The approach is based on an adaptive decision mechanism for turning servers on/off and detecting under/over utilization. This approach is designed to avoid performance degradation and improve energy efficiency. The basis of this approach is based on our previous work [14]. The forecasting module is added as promised in previous paper. Results have been analyzed more robust through sensitivity analysis and preferred comparison methods. Moreover, google cluster data [15] are used to evaluate the algorithm with a real-world workload. The algorithm is implemented and run on CloudSim [16], which is a commonly used Cloud simulator as in the previous study. CloudSim provides several VM allocation and migration policies and is mainly focused on IaaS-related operations. However, the provided allocation algorithms run with static workloads. The existing code of CloudSim is extended to meet dynamic workload requirements and make a fair comparison with the proposed algorithm.

The key contributions of this research are as follows:

- We propose an energy efficient resource allocation algorithm called Look-ahead Energy Efficient Resource Allocation (LAA). LAA facilitates adaptive allocation of the incoming user requests to computing resources. A single threshold is used for CPU overutilization detection, but it is not the only parameter used to decide whether to turn a new server on to host the newly arrived workload. The trend of the system is as important as the threshold during the allocation decision process. If the number of already active servers meets the forecasted future requirement and the current overutilization situation occurs temporarily, then the proposed algorithm makes the allocation decision for the newly arrived workload by considering the remaining time for already running workloads on active servers.
- The proposed algorithm minimizes energy consumption while preventing performance degradation. The algorithm is based on not only the current state of

the system but also future demand as determined through Holt Winters forecasting to make adaptive decisions. Google has published the trace data of their clusters. These data are used to evaluate the performance of the proposed algorithm. After data analysis and comparison with another time series analysis methodology called Auto Regressive Integrated Moving Average (ARIMA), Holt Winters gave the better result in terms of the minimum error rate.

The remainder of this paper is organized as follows. Related work is reviewed in [Related work](#). [Problem definition](#) describes the obtained problem and system model. The focus of [VM consolidation algorithm](#) is on the energy efficient VM consolidation approach. Performance evaluation evaluates our work through Google workloads by comparison with previous studies with migration. In [Conclusion](#), we summarize and provide concluding comments.

Related work

Khattar et al. [17] is a broad survey paper about energy efficiency in Cloud Computing. Optimization techniques are divided into two main categories: heuristics and metaheuristics. Although, it mainly focuses on the heuristics, it also provides the view of metaheuristics. According to the survey, LAA can be evaluated under the category of the probabilistic heuristics. Choosing the appropriate optimization technique to solve the problem is not always easy. The workload type and the time constraint can be considered to decide the appropriate solution for the problem under consideration. Workload types can be split into two categories: workflow and independent tasks. Contrary to independent tasks, in workflow category the workload is considered as a set of dependent tasks and the problem is formulated as a mapping of a set of tasks and resources. Metaheuristic methodologies are suitable to find global optimal mapping solution for workflows. Moreover, metaheuristic methodologies can be applied in resource scheduling step for reallocation of already allocated tasks. Heuristic methodologies are commonly preferred to find the local optimal solution of resource placement and task scheduling for independent tasks at a particular instant of time.

As a representative example in metaheuristic approach, a recent paper [18] proposes Particle Swarm Optimization (PSO) model for cost minimization and deadline constraint scheduling. Though they focus on the advantages of the cloud computing such as the use of unlimited resources, flexibility in terms of acquiring and releasing resources, heterogeneity, and pay per use model, the key challenges such as energy inefficiency and carbon emission are not considered [19] provides a solution combination of different techniques for resource allocation and task scheduling. Due to providing the response time improvement, the combination

of Bandwidth-aware task scheduling (BATS) and BAR is used to allocate tasks. Since that the system only considers availability of bandwidth in allocating resources with this step, preemption and divide and conquer steps are proceeds to execute tasks if the VM is overloaded with the assigned task in BATS+BAR step. However, additional steps to execute workloads because of ignoring step to check availability of resources causes energy overhead [20] that provides a comparison metaheuristic scheduling algorithms and analysis results, emphasizes the lack of applying metaheuristic techniques to solve energy inefficiency as an open issue. In another broad survey on task scheduling mechanisms [21], only few papers consider energy consumption while providing task scheduling via using metaheuristic techniques.

Consolidation is a commonly used technique during allocation and migration for saving energy by using minimum number of resources and increasing resource utilization rate. Consolidation can be achieved during both allocation and migration by using heuristic solutions. Allocation and placement step of migration are handled as a Bin Packing optimization which places items into the minimum number of PMs using heuristic techniques which are enough fast for large-scale system but not guaranteeing optimal results [22]. Bin packing approach is applied in different ways in the literature as shown in Table 1. Finding suitable host for both incoming workload and already running workload to be replaced is the basic step of the bin packing approach. Studies differ from each other according to methods used for finding suitable host such as First Fit, Worst Fit and incoming request types such as VM, task or container [23]. Other differentiating point of studies is meaning of bin capacity as optimal resource utilization rate of a resource or fully loading. Studies are also divided into two categories in terms of bin. In some studies, CPU is considered as a bin since [24] shows that CPU is the component with the highest proportion in terms of power consumption of a physical host. Moreover, the running machines at idle states or low utilized cause energy inefficiency. Power consumption of CPU in idle state is more than 50% of the fully loaded state [25–28]. However, [6, 13, 29] propose multi-dimensional bin packing solutions because CPU is a sufficient parameter alone for only CPU-bounded workload and based on the workload requirements, other resource types also should be considered.

Server consolidation through migrating workloads into a small set of servers and switching idle servers off is a commonly used technique to save energy. Migration is a process of replacement VMs from the source hosts to target hosts to

reduce the number of active servers via following 4 fundamental steps: (i) threshold definition, (ii) source host detection, (iii) VM selection and (iv) VM placement. Threshold is used to classify hosts as source and target. Threshold value can be set for single resource or multiple resources. It depends on the problem formulation defined as above (i.e. unidimensional and multidimensional). Threshold value can be set before the runtime statically or at the runtime dynamically. According to [8], optimal CPU utilization is set as 70% and if the usage exceeds the optimal value, it means the server is overloaded. Therefore, some selected VMs should be migrated out to reduce performance degradation and energy consumption [30–32] propose auto-adjustment of threshold by using statistical analysis of historical data such as Median Absolute Deviation, Interquartile Range, Local Regression belong VMs because of the motivation which fixed values of thresholds are unstable for unpredictable workloads. Underloading is also undesired state in terms of energy efficiency. Through migration all VMs from underutilized server to target hosts, underutilized server’s state is changed to idle, and it can be switched off to save energy. In [33], K-means clustering algorithm is used to define three threshold T_a , T_b , T_c values to classify hosts as little loaded, less loaded, normally loaded and overloaded. Underutilization is divided into two sub states as little loaded and less loaded. Besides overloaded hosts, little loaded hosts also are candidates of source hosts while target hosts are selected from less loaded hosts during VM placement step [34] uses power performance ratio called gear. There are four types of gears as preferred, best, underutilized and overutilized and 11 gears value corresponding to 11 distinct utilization rates from 0 to 100%. If the gear which the server is working at is higher than the preferred gears, it means the server is overutilized. Otherwise, it is underutilized. Unlike the previous studies, based on the characteristics of the computing node (i.e., 4 different computing nodes Fujitsu, Inspur, Dell, and IBM are evaluated), there may not be a utilization rate that means overloading. Threshold usage differences of VM allocation and migration source/target host detection are shown in Table 2.

The objective is to use fewer resources to run the same workload at low cost and energy efficiently through consolidation. However, when the migration is preferred for consolidation, it causes low throughput and extra energy consumption since it is a time- and resource-consuming process. Therefore, [35] avoids migration of VMs which would not recover the energy consumption during migration. VM selection and VM placement steps become

Table 1 Bin packing application differences

Problem		Bin Type		Bin Size	Item	Method	
VM Allocation	VM Placement	CPU	Multi-dimensional	Full capacity	Optimal utilization rate	Task/VM/ Container	Heuristic/Meta-heuristic

Table 2 Threshold usage differences of energy efficiency techniques

Technique	Threshold		Papers
Allocation	Up to fully loaded	Single resource	[6, 27, 28]
		Multiple resources	[29]
	Up to optimal utilization rate	Single resource	[26]
		Multiple resources	[2, 13]
Migration	Single threshold	Single resource	[25]
	Multiple thresholds	Single resource	[31, 33, 34]

important in terms of ensuring energy efficiency via migration. If the remaining length of the selected VM is long enough to run on target host and the target host is more energy efficient compared to the source host, then the migration decision is made [31] proposes minimum migration time policy which selects VM with minimum time requirement compared to other VMs on the same host for completion of migration process. Other selection method proposed in the same paper is based on selection VM with the highest correlation of CPU utilization with other VMs [36] considers VM on/off operation cost besides data transmission cost. The research calculates performance degradation considering CPU utilization of the selected VM and the network distance between source and target hosts [33] proposes two VM selection methods called MRCU and MPCU which select VMs loaded with more CPU or I/O intensive tasks. Because higher memory utilization causes more energy consumption of VM migration. VM placement method is based on both maximization of energy efficiency and minimization of SLA violation. Multi resources category is not existed for consolidation in Table 2 since other resource utilization rate beside CPU is only important at VM selection phase, not in source host decision.

Switching idle servers off after consolidation is a commonly used technique to save energy. However, decisions to switch servers off without any estimation of the future demand can cause additional energy consumption depending on the duration of the passive state. The duration of the passive state is not long enough to save more energy than is required to keep the server idle when an additional server is required to host incoming requests in this interval. When all active servers will be overutilized by hosting an incoming workload, switching a new server on is an option. However, if the energy consumption of switching a new server on is greater than the required energy for overutilization, then the incoming workload should be allocated to one of the already active servers.

Therefore, the prediction of future workloads becomes an important issue for the decision of turning servers on and off to save power. The idle and running time periods of servers and the workload execution time should be observed, and the resource usage pattern should be derived from this

knowledge with an ignorable error rate. A new scheduling algorithm [37] called PreAntPolicy has been proposed with a prediction model based on fractal mathematics that decides whether to turn servers on/off. However, in this algorithm, during the calculation of energy consumption, the energy requirements for both VM migration and turning servers on/off are not considered. The model is based on deadline and budget constraints, and the aim is to find the best node-workload pair. Another scheduling algorithm has been presented based on urgency analysis to schedule the real-time workload while defining two static thresholds to decide both urgency and whether to turn a server off [38]. However, these decisions are only based on the current status of the system without forecasting the future demand. The costs of VM migration and turning a server on are also not considered, although migration causes additional costs and prolongs the response time of the workload. Similarly, a heuristic workload scheduling algorithm without VM migration based on laxity has been proposed [39]. To minimize the number of servers, the method is based on postponing the execution of workloads according to the deadlines of workloads. Nevertheless, the method does not contain a prediction methodology to decide the required number of servers. In addition, an optimal resource utilization rate is not defined to avoid performance degradation. A scheduling algorithm [40] has been built named the Longest Loaded Interval First that considers VM migration and the energy requirement during the migration transition period. The authors of the algorithm emphasized that energy consumption can be reduced by applying a limited number of VM migrations. However, a static workload is preferred. A workload and prediction mechanism are not considered, but prediction methods can also help to decrease the number of VM migrations for dynamic workloads. The ECTC and MaxUtil [41] workload consolidation techniques have been compared with their other versions, including the migration mechanism. When the remaining execution time of workloads is short, the experimental results show that techniques using migration do not contribute to power savings. However, these works have not adopted an adaptive decision mechanism or forecasting technique.

Considering the drawbacks of previously proposed methods in the literature, an adaptive decision-making mechanism that evaluates not only the system's current state, but also seasonal and trend patterns is proposed. A novel energy efficient resource management approach with no migration is proposed in this research.

Problem definition

This section introduces the workload type, the energy model and the obtained problem. The objective of this paper is to propose an energy efficient resource allocation while considering the minimization of performance

degradation. The model deals with the optimum number of servers relying on the adaptive optimal utilization of hosts with respect to the consumed energy to meet dynamic workload requirements based on prediction. Many studies in the literature aim to run resources at the maximum utilization rate to provide energy efficiency. Note that the utilization of resources can be managed by introducing new processes to the servers running preemptive schedulers. However, under- or overutilization of servers causes performance degradation. Each resource type has its own optimum utilization rate. In addition, turning servers on and off unnecessarily causes performance degradation. Workloads can be classified according to resource needs. In this study, we address computationally intensive workloads. The CPU needs of this type of workload are more intense than those of other resource needs. The threshold for the optimal utilization rate of the CPU is set at 70% [8]. Through an adaptive decision mechanism, the unnecessary turning of servers on/off is avoided to increase energy efficiency.

System model

The proposed model consists of three functional modules: a monitoring module; a workload placement module, with physical and virtual servers; and a forecasting module.

The monitoring module is responsible for observing the CPU requirement (CR) and execution time (ET) of each incoming workload during the last 5 min. The processing requirement of each workload W_i is calculated by multiplying CR_i and ET_i . The total processing requirement (PR_{total}) is the sum of submitted workloads' processing requirements in that 5 min interval. The monitoring module transfers the obtained knowledge to the forecasting module and the workload placement module.

The forecasting module determines the required number of processing units (Nr) according to user demand. Previously, the forecasting module used PR_{total} time series to predict future demand. Since PR_{total} does not have seasonal patterns and trends, forecasting methodologies, such as Holt Winters, ARIMA, support vector regression and nonlinear regression, give results with a high error rate. The approximated total processing requirements (APR_{total}) are used to calculate Nr . APR_{total} uses the mean value of both the CPU requirement and execution time instead of exact values. The noise of the PR_{total} time series is filtered by using the mean value of the parameters. APR_{total} is calculated by multiplying the number of submitted workloads ($NoSW$), mean execution time of submitted workloads (MET) and mean CPU requirements of submitted workloads (MCR) in intervals. $NoSW$, MET , and MCR are forecasted separately through forecasting methodologies, namely, Holt Winters and ARIMA. Holt Winters gives significantly better results than ARMIMA in terms of the Mean Absolute

Percentage Error (MAPE) for each time series since the time series includes seasonality and trend. In addition, because the interval is short and the period being forecast is long, ARIMA is not the right choice. To use optimal system resources (SC_{total} indicates the total capacity of a server and Th indicates the optimal utilization rate), the following equations are used to forecast the required number of processing units (SC_{total} indicates the total capacity of a server and Th indicates the optimal utilization rate).

$$APR_{total} = NoSW * MCR * MET \tag{1}$$

$$Nr = \frac{APR_{total}}{SC_{total} * Th} \tag{2}$$

The forecasting module based on the Holt Winters forecasting methodology has an 8.85 error rate.

The workload placement module is responsible for the allocation of workloads to suitable VMs and the allocation of these VMs to suitable servers. According to the information received from the monitoring module and forecasting module, decisions about turning a server on/off and the placement of incoming workloads are made in this module. The designed working model is explained in [Energy model](#).

Energy model

The cumulative energy consumption of a server (E_{CS_i}) is equal to the sum of the energy required the switching the server off and on (E_{OO_i}) and the energy required for the actual service time (E_{Ser_i}). It is given by:

$$E_{CS_i} = E_{OO_i} + E_{Ser_i} \tag{3}$$

The sum of the energy consumption of servers in a data center is equal to the total energy consumption of the data center (E_{CDC}), as shown in Eq. 4, where m indicates the number of servers in the data center.

$$E_{CDC} = \sum_{i=0}^m E_{CS_i} \tag{4}$$

During the service time, the server may run at different utilization levels that consume different amounts of power. Power consumption figures are published by computer producers. Total energy consumption during service time is shown in Eq. 5, where t indicates the time spent at a particular level of processor usage given by $Power[j]$. Note that the power model of the server is based on the utilization level which is split into eleven rates from 0 to 100%. For example, with the placement of a newly incoming workload, the utilization level of the server can be increased from 10 to 30%. It means that the server has never run at 20% utilization level and t equals 0 for the Power [2]. The discrete power model is used to be compliant with the simulator.

$$E_{Ser_i} = \sum_{j=0}^{10} t * Power[j] \tag{5}$$

In addition, the migration cost is not an ignorable parameter when calculating the energy consumption of a system. The real energy consumption (ER) can be obtained by adding costs of migration and switching on to this result, as seen in Eq. 6.

$$ER = E_{CDC} + NoM * MigrationCost \tag{6}$$

PI_i is a special representation of $Power[0]$ of a Server i . t_{idle} shows the idle time of Server i . If the required energy for switching a server on/off is more than the required energy for running the server at idle state for a period, then keeping the server in an idle state during that period is more energy efficient. The monitoring window is decided based on Eq. 7. In the calculated window, keeping a server in the idle state is more energy efficient than switching a server on/off to meet the incoming request. According to the spec of the selected server in this research, the monitoring interval is 5 min.

$$PI_i * t_{idle} < E_{OO_j} \tag{7}$$

VM consolidation algorithm

The consolidation algorithm is based on a single threshold to decide whether the server is overutilized. In other words, if a server’s CPU utilization ratio exceeds the pre-defined threshold, then the server is overutilized; otherwise, the server is underutilized. If the utilization of the server equals the threshold, then the server is running at the optimal utilization rate. Unlike other studies, overloading detection does not trigger VM migration. Overloading is undesirable since it causes performance degradation, but it can be acceptable under some conditions. To decide the allocation of incoming workloads, this threshold is not a sufficient parameter. In addition to the threshold, future demands are also considered to be as important as the system’s current state. There are two essential trends in the VM Consolidation Algorithm: Shrinking and Enlarging. Since the algorithm starts with no active server, the default trend in which the system is going to enlarge to serve incoming workloads by introducing new servers is enlarging. The second trend is shrinking, in which the number of active servers will be reduced because the number of expected workloads can be completed with a smaller number of servers than the number of existing ones.

All the servers in the system are in the passive (i.e., available in the system to be activated upon request) list initially. When a request arrives, one of the passive servers is turned on to meet the requirement. When all the

allocated workloads on a server are completed, the server is added to the idle host list. Then, the server is turned off and moved from the idle host list to the passive host list. According to the load of the server when it is in an active state, it is either on the underutilized server list or overloaded server list.

The proposed algorithm also uses a new set of parameters, such as the remaining execution time of a workload, active number of servers (N_a), required number of servers (N_r) and efficient utilization threshold. The system can be unstable in two cases: (1) N_a is greater than N_r , which means that there are underutilized servers, causing energy inefficiency, or (2) N_r is greater than N_a , causing overutilized servers and performance degradation when new servers cannot be switched on.

The proposed model to find the most appropriate server in terms of energy efficiency, as shown in Algorithm 1, is based on the shrinking or enlarging trend of the system. In the shrinking trend, the system has more active servers than required for foreseen demand. There are three different types of servers from the perspective of utilization: idle, underutilized and utilized. If the number of idle servers equals the difference between the number of active servers and the number of required servers, then the servers will be switched off. Moreover, if the number of idle servers to be switched off is not sufficient for optimal solution, then a subgroup of underutilized servers, and even utilized ones for some extreme cases, will also be switched off as soon as their respective ongoing workloads are completed. To support the shrinking trend when a workload is submitted, the workload is assigned to the most suitable active server instead of activating a new server for the workload. The most suitable server selection algorithm starts with finding the longest remaining execution time of the running workloads among underutilized servers. Table 3 provides a list of main symbols and their meanings in the algorithms.

The selection of the server with the highest remaining time for the existing workloads (Algorithm 2) makes it possible to overlap the execution time of the newly arrived workload with it. Therefore, the newly arrived workload is expected to be completed after the completion of the existing workloads in the worst case. If there is more than one suitable server, then an additional parameter called minimum violation is used to decide the most appropriate one. The efficient utilization rate of the CPU is determined to be 70% for energy efficiency. Given that 70% is the optimal value, a range between 65 and 75%, whose mid-value is 70%, is accepted as the energy efficiency range. Minimum violation is the most approximate utilization rate to the efficient utilization rate when the incoming workload is assigned to them.

Algorithm 1: Look Ahead VM Allocation

Input: Set of Workloads W , underutilizedServerList, passiveServerList

Output: Set of Provisioned VMs and Allocated Workloads

```

1: for  $W_j \in W$  do
2:   if active server list is empty then
3:     select and switch on a server  $S_i$  from passiveServerList;
4:     create  $VM_1$  in  $S_i$ ;
5:     allocate  $W_j$  on  $VM_{i,1}$ ;
6:     update Server Lists according to utilization rate of  $S_i$ ;
7:     continue;
8:   if  $N_r > N_a$  then //Enlarging Trend
9:     if underutilized server list is not empty then
10:      select a server  $S_i$  from underutilizedServerList;
11:      create  $VM_k$  in  $S_i$ ;
12:      allocate  $W_j$  on  $VM_{i,k}$ ;
13:      update Server Lists according to utilization rate of  $S_i$ ;
14:      continue;
15:   else //Shrinking Trend
16:      $S_i \leftarrow$  call HRT
17:     if  $S_i$  is not null then
18:       create  $VM_k$  in  $S_i$ ;
19:       allocate  $W_j$  on  $VM_{i,k}$ ;
20:       update Server Lists according to utilization rate of  $S_i$ ;
21:       continue;
22:     else
23:        $S_i \leftarrow$  call LRT
24:       if  $S_i$  is not null then
25:         create  $VM_k$  in  $S_i$ ;
26:         allocate  $W_j$  on  $VM_{i,k}$ ;
27:         update Server Lists according to utilization rate of  $S_i$ ;
28:         continue;
29:   if  $W_j$  is not allocated on any VM then
30:     select and switch on a new server  $S_i$  from passiveServerList;
31:     add  $S_i$  to active server list;
32:     create  $VM_1$  in  $S_i$ ;
33:     allocate  $W_j$  on  $VM_{i,1}$ ;

```

```

Algorithm 2: HRT


---


Input: underutilizedServerList,  $W_{new}$ , RC, UR
Output: targetServer

1: threshold  $\leftarrow$  0.70;
2: minViolation  $\leftarrow$  0.70;
3: highestRT  $\leftarrow$  ET[ $W_{new}$ ];
4: targetServer  $\leftarrow$  null;
5: foreach server  $S_i$  in underutilizedServerList
6:   if RC[ $S_i$ ]  $\geq$  UR[ $W_{new}$ ] then
7:     TU[ $S_i$ ] = TU[ $S_i$ ] + UR[ $W_{new}$ ];
8:     violation = threshold - TU[ $S_i$ ];
9:     if minViolation > violation
10:      && max(RT $_{i,j}^W$ ) > highestRT then
11:        minViolation = violation;
12:        highestRT = max(RT $_{i,j}^W$ );
13:        targetServer =  $S_i$ ;
13: return targetServer;
    
```

In the example in Fig. 1, the number of active servers is more than the future demand; therefore, the number of active servers should be decreased. Assuming that these two servers from the active server set have been selected to be turned off, the server with the highest remaining time is selected for the incoming workload W_4 . If W_4 is allocated on Host 1, then it prolongs the time in standby of the server.

Table 3 List of the main symbols and their meanings

Symbol	Meaning
S	Set of servers where $S, S_i \in S, i = \{1, \dots, n\}, j \in N$
RC[S_i]	Remaining capacity of S_i
UR[W_{new}]	Utilization requirement of W_{new}
TU[S_i]	Total utilization of S_i
W_{ij}	$W_j \in W$ running on $S_{ij} = \{1, \dots, m\}, m \in N$
VM $_{i,k}$	VM $_k \in VM$ on $S_k = \{1, \dots, z\}, z \in N$
max(RT $_{i,j}^W$)	Maximum remaining time of existing workloads
min(RT $_{i,j}^W$)	Minimum remaining time of existing workloads
ET[W_{new}]	Execution time of W_{new}

Conversely, when W_4 is allocated on Host 0, it does not cause the same issue. The cost of running the same server with the same resource requirement is not considered in this study.

If there is not a suitable server to host the new workload according to the parameters of the highest remaining time and minimum violation, the search algorithm is conducted among servers and will be overloaded after the assignment. Running in the overloaded state causes performance degradation in terms of response time and throughput. In addition, switching a new server on to meet short-term needs can cause redundant energy consumption when the systems trend is considered. Therefore, the proposed approach is based on the selection of the most suitable server from the set of active servers instead of switching a new server on. The temporarily overloaded state is acceptable when the system has a number of active servers that is greater than the required number of servers for future demand, i.e., the system is in a shrinking trend. To reduce the running time of the selected server in the overloaded state, the workload with the shortest remaining execution time according to the

optimal utilization rate after assignment should be run, as described in Algorithm 3. The shortest remaining time of existing workloads is not the only parameter used. The utilization rate of the server with the shortest remaining time is another parameter to be considered. When the workload is finished, the utilization rate of the server should approximate the optimal utilization rate. In addition, the minViolation parameter is used to ensure that the server is up to 100% utilization.

compared to decide which server is the most appropriate in terms of the least performance degradation. However, in the example shown in Fig. 2, the shortest remaining times on Host 0 and Host 1 are equal. Therefore, the decision is made according to the utilization rate. For this case, looking at the further utilization rate when W_1 and W_3 are finished, Host 1 will be well utilized while Host 0 will be overutilized. Therefore, Host 1 is selected for the placement of W_4 .

Algorithm 3: LRT

Input: underutilizedServerList, W

Output: targetServer

```

1: threshold  $\leftarrow$  0.70;
2: minViolation  $\leftarrow$  0.30;
3: leastRT  $\leftarrow$  ET[Wnew];
4: targetServer  $\leftarrow$  null;
5: foreach server Si in underutilizedServerList
6:   if RC[Si]  $\geq$  UR[Wnew] then
7:     TU[Si] = TU[Si] + UR[Wnew];
8:     violation = TU[Si] - threshold;
9:     if minViolation > violation
10:      && min(RTi,jW) < leastRT then
11:        minViolation = violation;
12:        leastRemainingTime = min(RTi,jW);
13:        targetServer  $\leftarrow$  Si;
13: return targetServer;
    
```

In Fig. 2, W_0 and W_1 are running in Host 0, while W_2 and W_3 are running in Host 1. W_4 is newly arrived with a 30% utilization requirement, and rt_4 shows that the remaining time of W_4 is equal to its execution time. When the new workload is allocated on either Host 0 or Host 1, it causes performance degradation since the utilization rate will exceed the optimal utilization threshold. The least remaining time on Host 0 is rt_1 of W_1 , and rt_3 is the shortest remaining time of Host 1. The servers are

In the enlarging trend, if N_a is less than N_p , then a destination host is investigated from the underutilized server list to be well utilized when the server is allocated for the workload. If there is no suitable server among the underutilized servers, then the idle server list is checked to determine whether the list is empty. If the idle server list is not empty, then one of the idle servers is used to run the workload. Otherwise, a server from the passive server list is switched on.

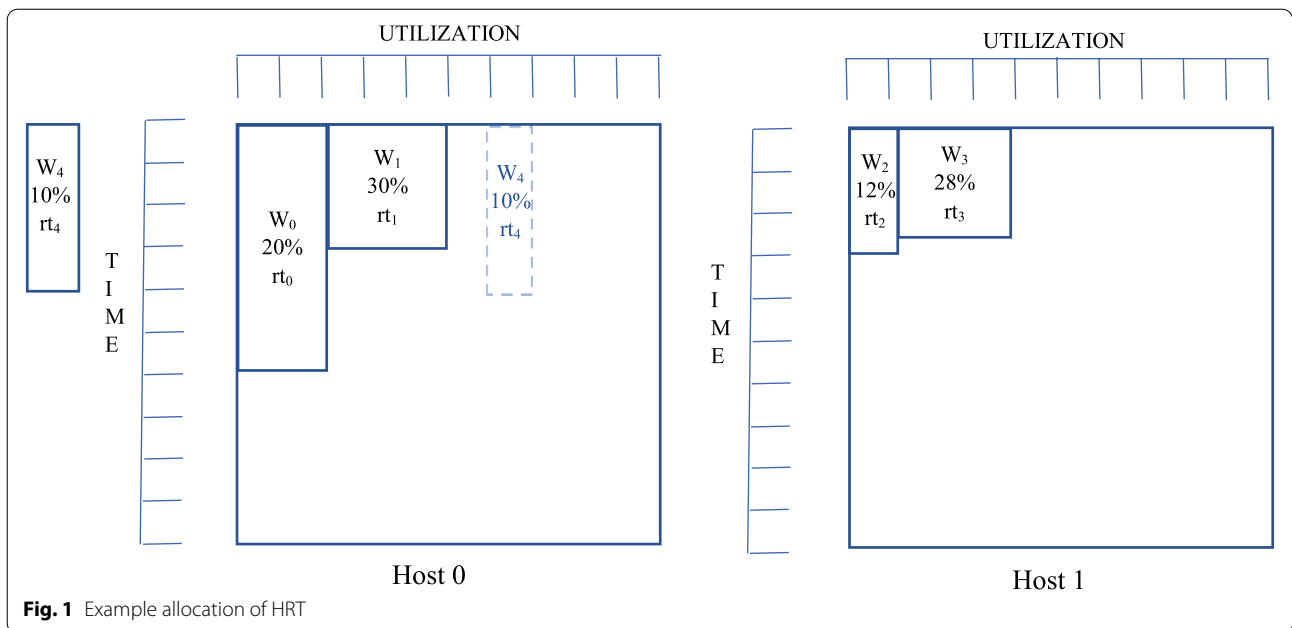


Fig. 1 Example allocation of HRT

The algorithm is based not only on the current state of the system but also on future demands, as described below. The number of incoming workloads is forecasted by using different methodologies: Holt Winters and ARIMA.

Performance evaluation

Experimental setup

The algorithm is implemented and evaluated in Cloud-Sim, which is commonly preferred in the literature since it provides a fair comparison between proposed

algorithms and previous approaches and is easy to adapt and implement. However, workloads enter the system in a static manner, and the usage rates of the works vary depending on time. Our algorithms provide dynamic submission. Therefore, to make a fair comparison, the benchmark code is modified to meet the dynamic requirements by working with Google Cluster Data via MongoDB integration.

It is assumed that the cloud data center consists of homogeneous servers. Initially, only the processor is

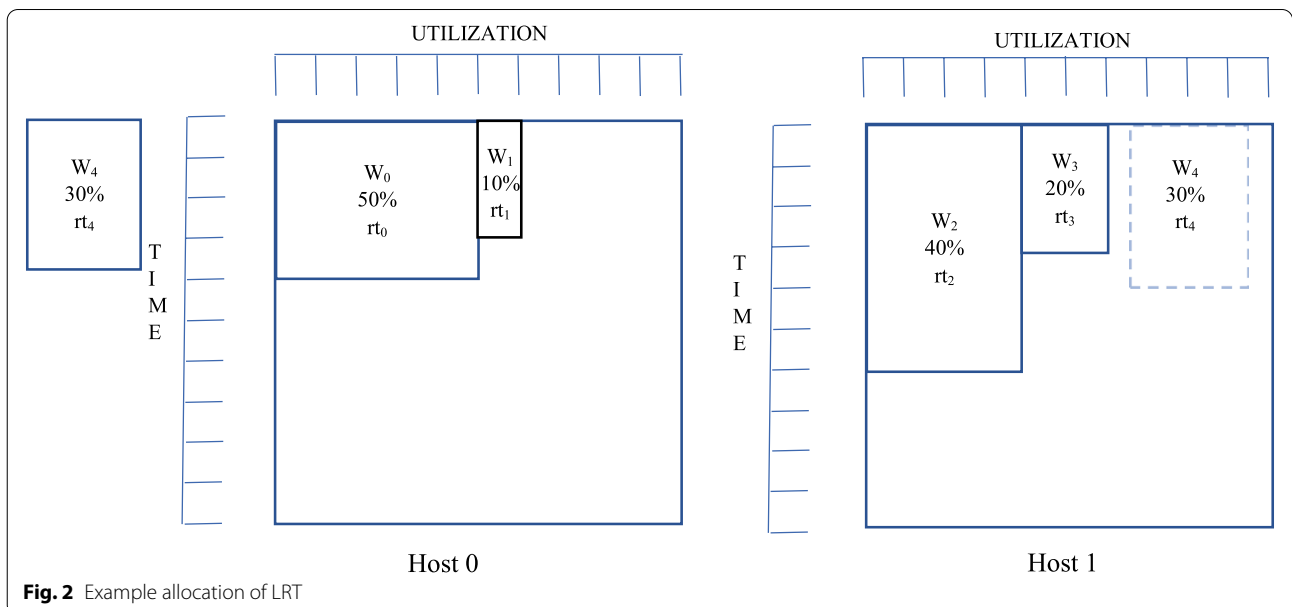
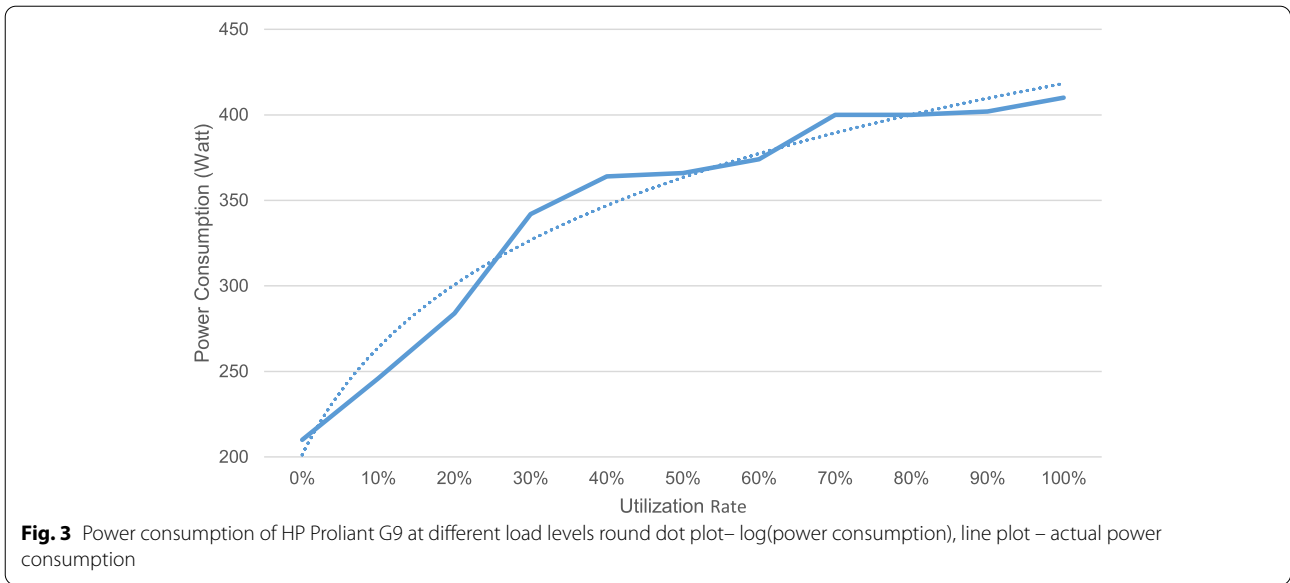


Fig. 2 Example allocation of LRT

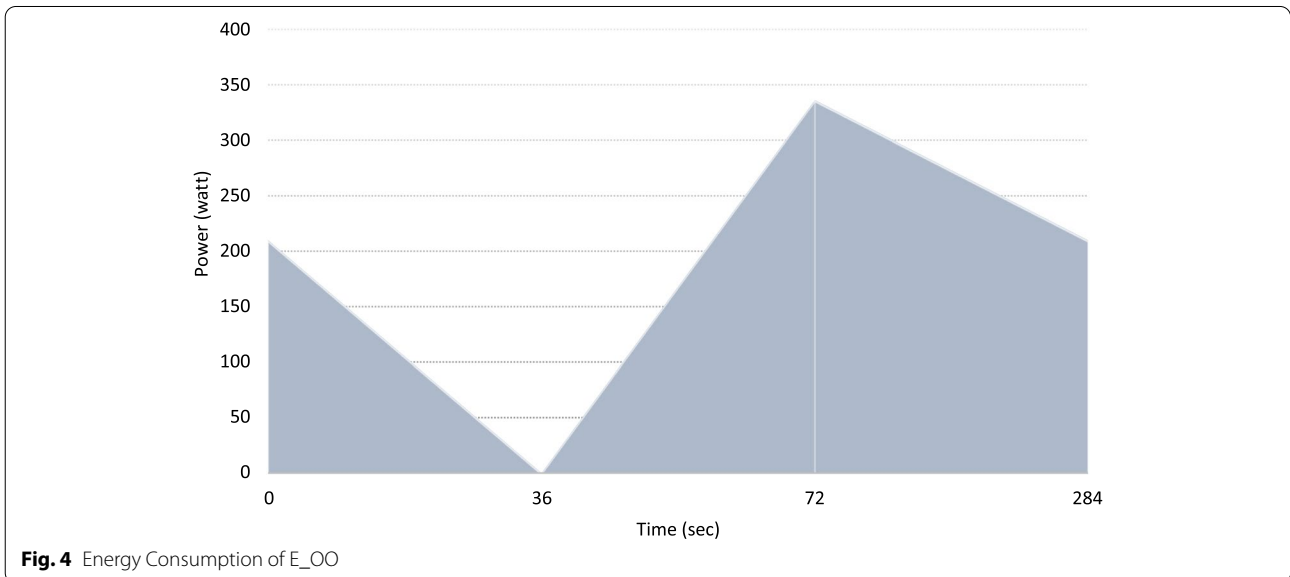


considered in terms of the resource type of energy consumption. The power consumption of selected server with respect to the utilization rate is shown in Fig. 3. HP Proliant XL170R G9 Xeon 2670 G4 includes 2 × 128 GB Ram, Intel Xeon 2670, 2 × 12 cores × 2300 MHz. The power spec of the server is shown in Fig. 3. In addition, based on the experiments conducted with TÜBİTAK B3 Laboratory, the energy consumption for switching the server off and on (E_{OO}) is measured as 0,019kwh. The result is obtained as an average of 5 times measurement as shown in Fig. 4. Shut down period takes 36 s and consumed 3600ws. The time it takes for the server to be ready to operate after the power has been turned on is 248 s and energy consumption for boot time is around

64000ws. As shown in Fig. 4, during boot time, power consumption is increased until 336 W which is almost the power at 30% utilization of the server according to the server’s spec and then it is decreased to the power at idle state of the server. The model presented in Eq. 8 can be used to calculate approximately value of E_{OO} for any server.

$$E_{OO} = (t_1 - t_0) * P_{idle} + (t_2 - t_1) * P_{20\%} + (t_3 - t_2) * (P_{20\%} + P_{idle}) / 2 \quad (8)$$

Each workload request is allocated on a virtual machine (VM). The VM types are Amazon EC2 instance types: Extra Large Instance (2000 MIPS, 3.75 GB); Small Instance (1000 MIPS, 1.7 GB); and Micro Instance (500 MIPS, 613 MB). Initially, the most appropriate VM is



allocated for the incoming workload with the resource requirements defined by the VM types. Then, VMs utilize less resources according to workloads [30].

Workload characteristics

Google published trace data on a cluster of approximately 12.5 k machines including 29 days of cell information from May 2011. A cell means a set of machines sharing a common cluster-management system. A job consists of one or more workloads. Google shared the trace data through six tables: machine events, machine attributes, job events, workload events, workload constraints, and resource usage. In the scope of this paper, the workload event table is used. The workload event table consists of timestamps, missing info, job ids, workload indexes, machine ids, event types, usernames, scheduling classes, resource requests for CPU cores, RAM and local disk space and different-machine constraints. Timestamps are in microseconds. Event types have different values, such as submit, schedule, finish and fail. The submitted workloads are used as inputs into the proposed scheduling algorithm. The interval of utilization measurements was decided as 5 min. The sum of incoming workloads in each 5 min was used.

The model that we used to compare with our proposed model is based on the VM migration technique [30–32]. This model does not use a static threshold to detect overloading. The threshold is changed during runtime according to historical data about the load of the host. The historical data are used in various ways. One of the ways is averaging the historical data according to the median absolute deviation (MAD). Other than the MAD, three methods are used: interquartile range (IQR), local regression (LR) and local regression robust (LRR). After the overloading detection step, the VM is selected to migrate. Several approaches have been suggested to select the VM, such as maximum correlation (MC), minimum migration time (MMT), minimum utilization (MU) and random selection (RS) [16]. According to experiments, LR-MMT provides the best results. Therefore, we compared our proposed model with LR-MMT. The proposed algorithms for LR-MMT are included in the power package part of CloudSim. The power package is based on static allocation. To compare the proposed model, the power package is extended to meet dynamic workload submission.

Performance metrics

The number of Hosts Shutdown (NoHS), Number of Received Workloads (NoRW), Energy Consumption (EC) and Number of Migrations (NoM) are used as key performance metrics in the migration model to which we compare our proposed approach. Our approach does not

contain a migration step. Therefore, the number of VM migrations becomes zero in this paper. The number of active servers is less than the migration model according to the experimental results. If there is already a place on an active server, then a new server will not be awakened. Moreover, energy consumption due to the number of active servers and the number of unnecessary migrations can be reduced through the proposed approach.

Simulation results and analysis

The forecasting module is based on Holt Winters, as described before. Therefore, the approach is named LAA-HW. If we knew the actual values instead of forecasted values, then the system gives the optimal result as LAA-O. The proposed model uses N_a and N_r parameters to decide whether the system has more active servers than required. If N_a is greater than N_r , then incoming workloads are allocated on already active servers. The mean CPU requirement of a day and the mean execution time of a day in the Google Trace logs are 3% and 1,13 min, respectively, leading to a small N_r value and causing less received workloads in LAA compared to LR-MMT. These requirements cause bottlenecks for workloads with short execution times and fewer CPU requirements. The number of migrations is zero in our approach. The energy consumption for switching on/off in our model is less than that in the migration model. As described in [System model](#), the energy consumption is the sum of required energy for computing the workload, migration and switching on/off decision. Each migration consumes 0,05 kwh [42]. Each E_{OO} consumes 0,019 kwh according to the lab experiments. The comparison results are shown in [Table 4](#). LR-MMT does not have a forecasting module. The host shutdown decision is made by considering only the system's current state. Therefore, the number of hosts shut down in LR-MMT is greater than in any type of LAA. LAA-O gives the best result in terms of energy consumption, and it also gives better results for NoRW than LAA-HW. This is because LAA-O knows the incoming CPU requirements of the workload, instead of forecasted the values belonging to the workloads.

To ensure the robustness of the proposed algorithm, sensitivity analysis is performed by increasing the execution time by multiplying by 10, 100 and 1000. According to the results, the experiment with the longest execution

Table 4 Comparison results

	NoRW	EC (kwh)	NoM	NoHS
LR-MMT	34,838	3344,26	6036	8563
LAA-HW	23,945	1300,92	-	3125
LAA-O	26,538	1200,59	-	2716

time gives the best result since a longer execution time requires a higher server open time and a higher number of active servers. The longest execution time overcomes the bottleneck of the small value of N_r .

LR-MMT gives a similar number of received workloads under different execution time values, unlike LAA-HW and LAA-O, as seen in Fig. 5. The numbers of received workloads by both LAA-HW and LAA-O are increased under extended execution time since longer execution times require a greater number of active servers.

In Fig. 6, the mean energy requirement to complete a workload is shown. The mean energy requirement is calculated by dividing the total energy consumption by the number of received workloads. LAA gives the optimal result with workloads of approximately 10 min since both high throughput and energy efficiency can be provided without turning a new server on. When the workloads remain as they are, the forecasted N_r value is small. Because of the lower number of active servers, the received number of workloads is low. On the other hand, the increase in the execution time of workloads causes an increase in the number of active servers. However, the proposed algorithms are based on the remaining time of already running workloads. A long execution time means a longer remaining time at any time. It is difficult to find the appropriate server from active servers with the least remaining time algorithm. Therefore, the only option is turning a new server on to meet the requirements, which gives rise to the need for an increment of the number of servers. With the increment of the number of active servers, the number of received workloads is also increased, but it causes significant energy consumption per workload since energy consumption and the number of active servers is directly proportional.

In addition, the experiment was conducted with first fit, best fit and worst fit algorithms to make comparisons in terms of the execution time for the same workloads. These algorithms are implemented as nonpower aware (NPA) algorithms. These algorithms consider CPU utilization requirements. First fit allocates the workload on the first hole that is large enough by scanning from the beginning. Best fit allocates the workload on the smallest hole that is large enough and produces the smallest leftover hole. Worst fit allocates the workload on the largest hole that is large enough and produces the largest leftover hole, which may be more useful than a leftover hole from a best fit approach. Figure 7 shows the mean execution time results in milliseconds belonging to the related algorithms to find appropriate place among search pool for single workload. It does not show the time the entire simulation takes to complete whole workloads. The first fit algorithm is the fastest algorithm among these three algorithms because it searches as little as possible. The best fit and worst fit algorithms show similar results, almost 25 times slower than the first fit algorithm. The NPA algorithms – first fit, best fit and worst fit and LR-MMT – started with 800 active hosts in the experiments. Then, according to the system’s trend, the number of active hosts was decreased by shutting idle hosts down. LR-MMT searches for the most appropriate host among a host list that does not contain overutilized servers. Therefore, the execution time of the LR-MMT algorithm is affected by both changes in workloads on servers and the number of active servers. If the number of active servers decreases, then the search pool decreases. The results of the sensitivity analysis show that when the execution time of workloads increases, the number of running workloads in parallel increases, which causes

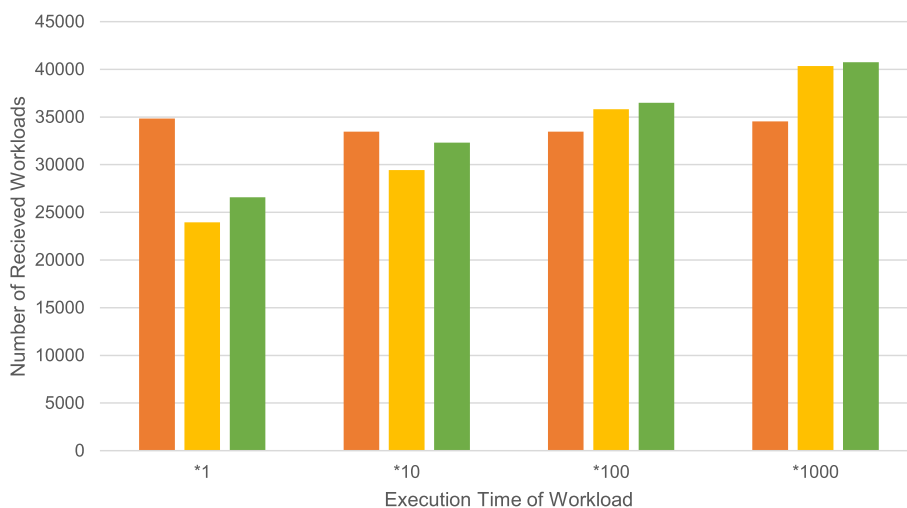
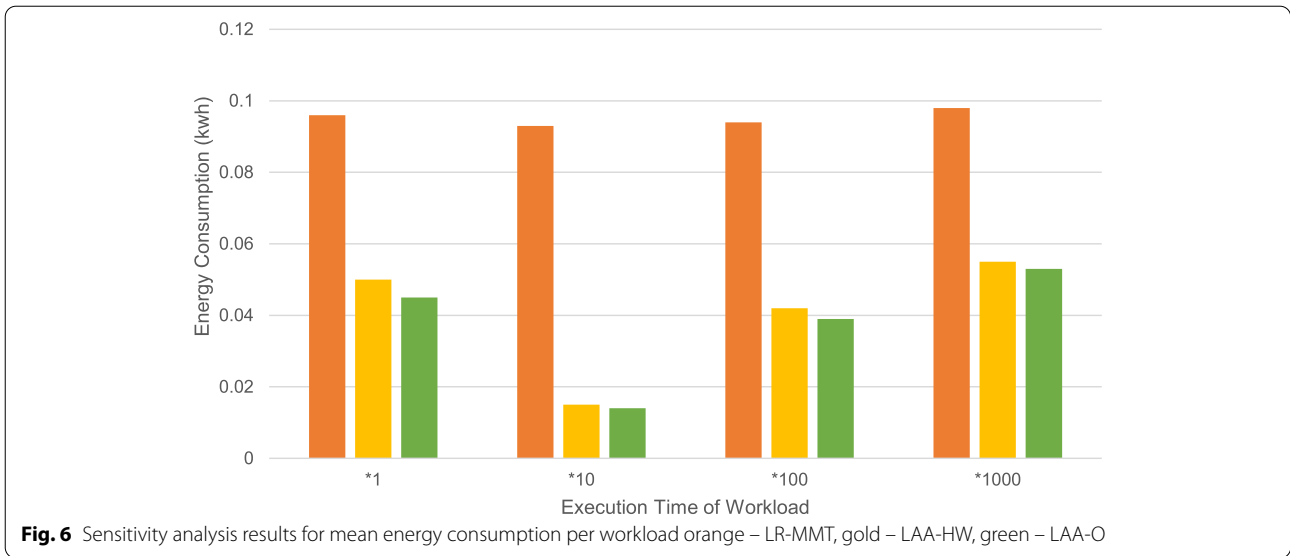
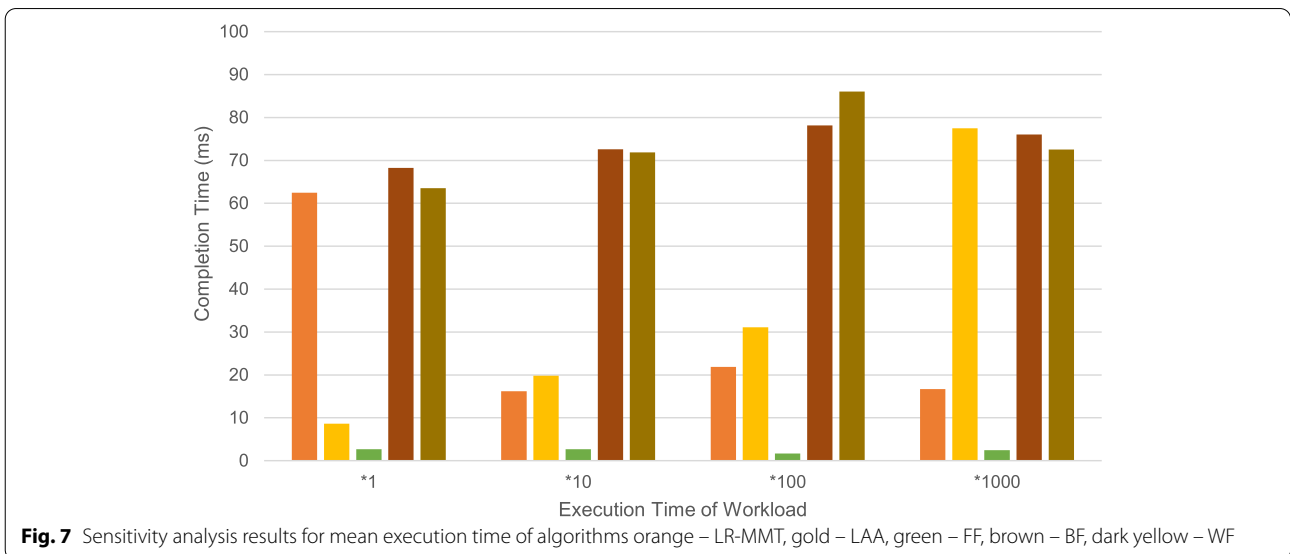


Fig. 5 Sensitivity analysis results for number of received workloads orange – LR-MMT, gold – LAA-HW, green – LAA-O



overutilization and decreases the size of the search area and execution time of the search algorithm. It starts with similar performance to the best fit and worst fit algorithms, and the execution time of the algorithm decreases to one-third as a result of the increased execution time of workloads with sensitivity analysis. The execution times of the NPA algorithms remain similar during the simulation period because the search area consists of all active servers. The LAA algorithm starts with 800 passive hosts and has discrete sets as the overloaded server list and the underutilized server list with a value of 0 at the beginning of the simulation. Based on the system's trend, shrinking, or enlarging, the search algorithm starts with the related server list or the passive or underutilized server list. The overloaded server list is only used for exclusion, which means that the search area becomes larger as the

number of underutilized servers' increases. The LAA starts with 4 times the execution time of the first fit algorithm, but with the increasing number of underutilized servers, it starts to give similar results to the best fit and worst fit algorithms. As explained in the problem definition section, the monitoring interval is selected as 5 min. The mean execution time of a day in Google tracelog is 1.13 min which is smaller than the monitoring window. It means existing workloads are completed until next monitoring time comes. Therefore, N_a and N_r has small values during the simulation time and the search pool consists of a small number of resources. If the lengths of the workloads are longer than the monitoring interval, then N_a and N_r are increased and so does the completion time of the algorithm is increased. Approximately 10 min lengths workloads which LAA gives the optimal results



of both energy efficiency and throughput with, also gives acceptable result in term of the completion time of the algorithm.

Conclusion

Energy efficiency in cloud environments has received significant attention in the past few years because of the increasing usage of system resources with developing technology and decreasing prices. To handle energy inefficiency, common resource provisioning and running servers at an optimal utilization rate through VM migration have been proposed to reduce energy consumption on cloud infrastructures. However, running at the optimal utilization rate may require turning a new server on to meet the requirement of incoming workloads, and it consumes more energy than is consumed due to the performance degradation from allocating the incoming workload on already active servers that are at optimal utilization rates. Moreover, VM migration also consumes energy and causes execution delays since time is needed for VM migration.

In this paper, the proposed model focuses on the open issues of the research in the literature. VM migration and the unnecessarily switching of servers on/off cause additional energy consumption. In addition, completing a workload with the migration overhead consumes more energy since it takes more time. Therefore, to avoid migration by optimizing the placement of new requests well and to avoid unnecessarily switching servers on/off, the proposed approach uses prediction methodology. Holt Winters is preferred as a forecasting technique because of its suitability to time series. Furthermore, most approaches in the literature propose using resources at the optimal utilization rate since over- and underutilization cause energy inefficiency. However, this means increasing the number of required resources, leading to more energy consumption. Based on this motivation, we propose an adaptive approach for VM placement without VM migration. The most important contribution is preventing VM migrations while considering the remaining time of running workloads. Optimum utilization is a significant factor in providing energy efficiency. However, even if its load will exceed the optimum utilization rate, allocation of a workload to an already active host instead of allocation of the workload to a new server should be preferred under some circumstances. We propose an adaptive decision-making approach to energy efficient allocation without migration. We consider not only history but also future demands and the remaining time of running workloads. To determine the systems behavior for workloads with longer execution times, sensitivity analysis is performed. The real execution times of workloads are extended by multiplying by 10, 100 and 1000 to perform the sensitivity

analysis. A short execution time means a lower processing requirement and a smaller value for N_r . However, it causes bottlenecks and fewer received workloads than with LR-MMT. When the execution times of workloads are increased, N_r is increased. Thus, the number of received workloads is increased compared to LR-MMT. In this manner, the energy consumption decreases with the proposed algorithm and VM migration overheads are avoided. In addition, the system performance in terms of energy efficiency and throughput is better than LR-MMT for longer execution times.

In this paper, we focus on CPU intensive workloads. We plan to extend the proposed approach to work with workloads based on other system resources apart from CPUs, such as networking components, I/O devices and storage.

Acknowledgements

The authors would like to thank the anonymous reviewers for their insightful comments and suggestions on improving this paper.

Authors' contributions

İlksen Çağlar reviewed the state of the art of the field, proposed the method, implemented algorithms and analyzed the results. Deniz Turgay Altılar supervised this research, lead and approved its scientific contribution, provided general input, reviewed the article and issued his approval for the final version. The author(s) read and approved the final manuscript.

Funding

The authors declare that they have no funder. It is aimed to find a funder through OA funding service.

Availability of data and materials

The datasets used during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interests

The authors declare that they have no competing interests.

Received: 4 June 2021 Accepted: 25 February 2022

Published online: 19 March 2022

References

1. Kaur T, Chana I (2018) GreenSched: An intelligent energy aware scheduling for deadline-and-budget constrained cloud tasks. *Simul Model Pract Theory* 82:55–83
2. Cao J, Wu Y, Li M (2012) Energy efficient allocation of virtual machines in cloud computing environments based on demand forecast. In *international conference on grid and pervasive computing*. Springer, Berlin, pp 137–151
3. Garg SK, Yeo CS, Anandasivam A, Buyya R (2011) Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers. *J Parallel Distrib Comput* 71(6):732–749
4. Stillwell M, Schanzenbach D, Vivien F, Casanova H (2009) Resource allocation using virtual clusters. In *2009 9th IEEE/ACM international symposium on cluster computing and the grid*. pp 260–267
5. Xu L, Seng S, Ye X (2012) Multi-objective optimization based virtual resource allocation strategy for cloud computing. In *2012 IEEE/ACIS 11th international conference on computer and information science*. pp 56–61

6. Li X, Qian Z, Chi R, Zhang B, Lu S (2012) Balancing resource utilization for continuous virtual machine requests in clouds. In *Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, 2012 Sixth International Conference on IEEE. pp 266–273
7. Galloway J, Smith K, Carver J (2012) An empirical study of power aware load balancing in local cloud architectures. In *Information Technology: New Generations (ITNG)*, 2012 Ninth International Conference on IEEE. pp 232–236
8. Dhiman G, Marchetti G, Rosing T (2010) Vgreen: A system for energy-efficient management of virtual machines. *ACM Transactions on Design Automation of Electronic Systems (TODAES)* 16(1):6
9. Geronimo GA, Werner J, Weingartner R, Westphall CB, Westphall CM (2014) Provisioning, resource allocation, and DVFS in green clouds. *International Journal on Advances in Networks and Services*, 7(1 & 2):108–117
10. Garg SK, Toosi AN, Gopalaiyengar SK, Buyya R (2014) SLA-based virtual machine management for heterogeneous workloads in a cloud data-center. *J Netw Comput Appl* 45:108–120
11. Verma A, Ahuya P, Neogi A (2008) pMapper: power and migration cost aware application placement in virtualized systems. *ACM/IFIP/USENIX international conference on distributed systems platforms and open distributed processing*. Springer, Berlin, Heidelberg, 243–264
12. Gmach D, Rolia J, Cherkasova L, Kemper A (2009) Resource pool management: Reactive versus proactive or let's be friends. *Comput Netw* 53(17):2905–2922
13. Srikantaiah S, Kansal A, Zhao F (2008) Energy aware consolidation for cloud computing. *Proceeding of the 2008 conference on power aware computing and systems*
14. Çağlar I, Altılar DT (2016) An energy efficient VM allocation approach for data centers. In *2016 IEEE 2nd International Conference on Big Data Security on Cloud (BigDataSecurity)*, *IEEE International Conference on High Performance and Smart Computing (HPSC)*, and *IEEE International Conference on Intelligent Data and Security (IDS)*. pp 240–244
15. Google cluster-usage traces: format and schema. <https://github.com/google/cluster-data>. Published May 6, 2013. Updated October 17, 2014. Accessed June, 2015
16. Calheiros RN, Ranjan R, Beloglazov A, De Rose CAF, Buyya R (2011) CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. *Soft Pract Exp* 41(1):23–50
17. Khattar N, Sidhu J, Singh J (2019) Toward energy-efficient cloud computing: a survey of dynamic power management and heuristics-based optimization techniques. *J Supercomput* 75(8):4750–4810
18. Rodriguez MA, Buyya R (2014) Deadline based resource provisioning and scheduling algorithm for scientific workflows on clouds. *IEEE Trans on Cloud Comput* 2(2):222–235
19. Gawali MB, Shinde SK (2018) Task scheduling and resource allocation in cloud computing using a heuristic approach. *J Cloud Comput* 7(1):1–16
20. Alboaneen DA, Tianfield H, Zhang Y (2016). Metaheuristic approaches to virtual machine placement in cloud computing: a review. *15th International Symposium on Parallel and Distributed Computing (ISPDC)* IEEE. United States : 214–221
21. Kalra M, Singh S (2015) A review of metaheuristic scheduling techniques in cloud computing. *Egypt Inform J* 16(3):275–295
22. Ferreto TC, Netto MA, Calheiros RN, De Rose CA (2011) Server consolidation with migration control for virtualized data centers. *Futur Gener Comput Syst* 27(8):1027–1034
23. Khan AA, Zakarya M, Khan R, Rahman IU, Khan M (2020) An energy, performance efficient resource consolidation scheme for heterogeneous cloud datacenters. *J Netw Comput Appl* 150:102497
24. Fan X, Weber WD, Barroso LA (2007) Power provisioning for a warehouse-sized computer. *ACM SIGARCH Computer Architecture News* 35(2):13–23
25. Hsu CH, Slagter KD, Chen SC, Chung YC (2014) Optimizing energy consumption with task consolidation in clouds. *Inf Sci* 258:452–462
26. Galloway JM, Smith KL, Vrbsky SS (2011) Power aware load balancing for cloud computing. In *proceedings of the world congress on engineering and computer science*, vol 1. pp 19–21
27. Lee YC, Zomaya AY (2012) Energy efficient utilization of resources in cloud computing systems. *J Supercomput* 60(2):268–280
28. Dabbagh M, Hamdaoui B, Guizani M, Rayes A (2015) Energy-efficient resource allocation and provisioning framework for cloud data centers. *IEEE Trans Netw Serv Manage* 12(3):377–391
29. Chen J, Du T, Xiao G (2021) A multi-objective optimization for resource allocation of emergent demands in cloud computing. *J Cloud Comput* 10(1):1–17
30. A Beloglazov R Buyya 2012 Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers *Practice and Experience*. Published online in Wiley InterScience *Concurrency and Computation* <https://doi.org/10.1002/cpe.1867;24:1397-1420>
31. Beloglazov A, Buyya R (2010) Energy Efficient Resource Management in Virtualized Cloud Data Centers. *10th IEEE/ACM International Conference on Cluster Cloud and Grid Computing*. Massachusetts Ave., NW Washington, DC United States.
32. Beloglazov A, Buyya R (2010) Energy Efficient Allocation of Virtual Machines in Cloud Data Centers. *10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*. Massachusetts Ave., NW Washington, DC United States
33. Zhou Z, Abawajy J, Chowdhury M, Hu Z, Li K, Cheng H, Li F (2018) Minimizing SLA violation and power consumption in Cloud data centers using adaptive energy-aware algorithms. *Futur Gener Comput Syst* 86:836–850
34. Ruan X, Chen H, Tian Y, Yin S (2019) Virtual machine allocation and migration based on performance-to-power ratio in energy-efficient clouds. *Futur Gener Comput Syst* 100:380–394
35. Zakarya M, Gillam L (2016) An energy aware cost recovery approach for virtual machine migration. *International conference on the economics of grids, clouds, systems, and services*. Springer, Cham, pp 175–190
36. Xu X, Zhang X, Khan M, Dou W, Xue S, Yu S (2020) A balanced virtual machine scheduling method for energy-performance trade-offs in cyber-physical cloud systems. *Futur Gener Comput Syst* 105:789–799
37. Duan H, Chen C, Min G, WU Y (2017) Energy-aware scheduling of virtual machines in heterogeneous cloud computing systems. *Futur Gener Comput Syst* 74:142–150
38. Chen H, Liu G, Yin S, Liu X, Qiu D (2018) Erect: energy-efficient reactive scheduling for real-time workloads in heterogeneous virtualized clouds. *J Comput Sci* 28:416–425
39. Zhang Y, Cheng X, Chen L, Shen H (2018) Energy-efficient workloads scheduling heuristics with multi-constraints in virtualized clouds. *J Grid Comput* 2018:1–17
40. Tian W, He M, Guo W et al (2018) (2018) On minimizing total energy consumption in the scheduling of virtual machine reservations. *J Netw Comput Appl* 113:64–74
41. Valentini GL, Khan SU, Bouvry P (2013) Energy-efficient resource utilization in cloud computing. *Large Scale Network-centric Computing Systems*, John Wiley & Sons, Hoboken
42. Voorsluys W, Broberg J, Venugopal S, Buyya R (2009) Cost of virtual machine live migration in clouds: A performance evaluation. In *IEEE international conference on cloud computing*. Springer, Berlin, pp 254–265

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Submit your manuscript to a SpringerOpen® journal and benefit from:

- Convenient online submission
- Rigorous peer review
- Open access: articles freely available online
- High visibility within the field
- Retaining the copyright to your article

Submit your next manuscript at ► [springeropen.com](https://www.springeropen.com)