Decision Science Letters 11 (2022) 113–126

Contents lists available at GrowingScience

Decision Science Letters

homepage: www.GrowingScience.com/dsl

Hybrid and adaptive harmony search algorithm for optimizing energy efficiency in VMP problem in cloud environment

Amol C. Adamuthe[a*] and Smita M. Kagwade[b]

[a]*Department of Computer Science & Information Technology, Rajarambapu Institute of Technology, India*
[b]*Department of Computer Science & Engineering, Rajarambapu Institute of Technology, India*

C H R O N I C L E

Article history:
Received August 10, 2021
Received in revised format:
November 10, 2021
Accepted December 26 2021
Available online
December 26, 2021

Keywords:
Harmony search Algorithm
Virtual Machine Placement
Server consolidation technology
Datacenter
Optimization

A B S T R A C T

Data Center energy usage has risen dramatically because of the rapid growth and demand for cloud computing. This excessive energy usage is a challenge from an economic and environmental point. Virtual Machine Placement (VMP) along with virtualization technologies is widely used to manage power utilization in data centers. The assignment of virtual machines to physical machines affects energy consumption. VMP is a process of mapping VMs onto a set of PMs in a data center to minimize total power consumption and maximize resource utilization. The VMP is an NP-hard problem due to its constraints and huge combinations. In this paper, we formulated the problem as a single objective optimization problem in which the objective is to minimize the energy consumption in cloud data centers. The main contribution of this paper is hybrid and adaptive harmony search algorithm for optimal placements of VMs to PMs. HSA with adaptive PAR settings, simulated annealing and local search strategy aims at minimizing energy consumption in cloud data centers with satisfying given constraints. Experiments are conducted to validate the performance of these variations. Results show that these hybrid HSA variations produce better results than basic HSA and adaptive HSA. Hybrid HS with simulated annealing, and local search strategy gives better results than other variants for 80 percent datasets.

1. Introduction

Cloud computing is the on-demand delivery of computing resources, such as software, data storage, computing power, networking, and database. The different types of cloud services available such as infrastructure as a service (IaaS), software as a service (SaaS), and platform as a service (PaaS). There are many benefits for both data center providers and the end-user. In general, a cloud provider is able to sell computer resources to a large number of consumers. On the other hand, consumers are able to buy computing resources for lower costs, as compared to the costs of keeping each of these resources and private infrastructure (software, hardware). Basically, a cloud provider aims at maximizing profits and that implies reducing the deployed computer resources as much as possible.

Data centers are the main infrastructure of cloud computing that mainly contains IT equipment for data storage, data processing, and communications. To meet the user's demand these data centers operate 24/7 with thousands of active hosts or servers, networking equipment, storage devices (Shuja et al., 2016) and these data centers use large amounts of energy in their various operations. A report related to the workload of the global data center by cisco states that this workload will

* Corresponding author.
E-mail address: amol.admuthe@gmail.com (A. C. Adamuthe)

doi: 10.5267/dsl.2022.1.001

be more than twice between 2016 and 2021 (Networking, 2016). Due to this rapid growth in demands of cloud computing services the energy consumption in data centers is increasing. Data centers (DCs) have much greater operational costs than capital costs, and they are the single-largest power-consuming structures, accounting for roughly 2% of worldwide power consumption (Gartner Estimates, 2007) In addition, the number of data centers worldwide was estimated to reach at 8.6 million in 2017, with this figure expected to rise quickly in the future.

Much research on data center power consumption has been undertaken at various levels. The collaboration with the United States has resulted in several studies. The yearly energy consumption in data centers was 91 billion kWh in 2010, and it was predicted to rise to 140 billion kWh by 2020 (Alharbi et al., 2019). According to another estimate, data centers consume about 1.1-1.3 percent of total global energy consumption, with this figure expected to rise to 8% by 2020. Due to CO2 emissions from data centers, such a large growth in energy consumption in data centers is becoming not only a major economic issue, but also a critical environmental concern. It is estimated that running data center servers contribute 0.5 percent of global CO2 emissions. As a result, lowering data center power usage and virtual machine placement in cloud infrastructure research problems is a growing research subject. Many approaches have been studied by researchers to reduce energy consumption in data centers. Virtualization technology is a key to easily manage servers or physical machines (PMs). A number of virtual machines (VMs) are mapped on the top of PMs and then they share the physical machines (PMs) resources, such as CPU, memory, bandwidth, and storage. This placement of virtual machines on the physical machine in the data center is the process which is called virtual machine placement also called server consolidation. The VMP is considered an approach to efficiently manage the usage of physical machines to reduce the total number of active physical machines (PMs) in the data centers (Speitkamp & Bichler, 2010). The initial idea was to map virtual machines (VMs) onto a smaller number of energy-efficient active servers and then switch off inactive or unused hosts (Tang & Pan, 2015). The benefit of this switching of inactive PMs is a reduction in energy consumption and this helps to save up to 66% of the total energy consumption (Chen et al., 2008). However, with VMP there are some challenges and need to be addressed. One of the major challenges is how to find an optimal allocation or placement of VMs to PMs such that the total energy consumption in a data center is minimized. VMP algorithms aim to obtain the optimal allocation of VMs to PMs with the design objectives. With the VMP algorithm, many design objectives have been addressed in the literature such as improving resource utilization, optimizing the power consumption, optimal VMs to PM placement, etc. The VMP problem is formulated as the bin-packing problem, and that is an NP-hard optimization problem. The VMP algorithms attempt to balance the usage of multidimensional resources among active physical machines in data centers. The algorithms may generate amounts of residual resources for each resource type of each active PM. With the anticipation of future requests, the residual of each resource left on each PM should be balanced along the different dimensions. Otherwise, the uneven leftover resources may preclude any more VM placement, consuming computer resources. A virtual machine placement problem shows a trade-off between energy savings and overall system performance. If a placement algorithm's objective is only energy savings then it is likely that physical machines become overloaded more often and their customers may experience some performance issues caused by the virtual machine placement. Within this algorithm, only the minimally necessary physical machines are kept active all the time and others are kept idle. Paper (Panigrahy et al., 2011; Xia & Tan, 2010) tried to improve that solution by shuffling or reordering the virtual machine request queue according to some criteria before actual placement and they are referred to as First Fit Decreasing Algorithms. Some investigations are based on the Multiple Knapsack Problem and Best Fit Algorithm (Fidanova, 2021; Song et al., 2008; Singh et al., 2008; Mohamadi et al., 2011). Physical machines are less crowded here, and many of them are being used to deploy multiple virtual machines at the same time. As a result, fewer installations are anticipated at the expense of increased energy use.

In this paper, a single-dimensional knapsack approach to the virtual machine placement problem is explored using the harmony search algorithm. The Harmony search algorithm is evaluated with several experiments on various data sets. Objectives of the paper are,

- Design harmony search algorithm for solving virtual machine problems for minimizing energy consumption in cloud data centers.
- Compare performance of harmony search algorithms with static and adaptive PAR.
- Improve the performance of the harmony search algorithm by hybridizing with simulated annealing and local search operators.

One contribution of this paper is the proposed adaptive and hybrid harmony search algorithm for solving VMP. Proposed HSA variations efficiently solves the problem and helps to optimize the energy consumption by the physical machines. Performance of proposed algorithms is validated through several experiments.

Our research article consists of the following sections. In section 2, we point out some of the literature reviews that we used for references. After that problem formulation and methodology is presented in section 3. This section describes in detail all of the methods used in the paper. After that, section 4 reported the simulation details, experimental details and results. Finally, in section 5, we conclude our research.

2. Literature Review

VMP is one of the cloud computing challenges and affects many aspects of cloud environments. And therefore, many researchers studied this problem to optimize the VM placement among the available physical servers. Various objectives have been considered across different studies such as network traffic minimization, power consumption minimization, economical revenue maximization, quality of service maximization, utilization rate maximization, etc. Many methods have been proposed to address this VMP problem including deterministic algorithms (Chaisiri et al., 2009; Alicherry & Lakshman, 2013; Dang & Hermenier, 2013) and more intelligent metaheuristic algorithms (Gao et al., 2013; Abdel-Basset et al., 2019)

Exact allocation and migration algorithms are used in cluster computing based on constraint programming for solving VMP problems having strengths in reducing the number of active physical machines (APM's) and helping to reduce the migration cost, but the prolonged search time is a major drawback in constraint-based algorithms (Ghribi et al., 2013) This approach considers CPUs as only a resource and their objective functions are for optimality. Min-cut hierarchical clustering algorithms are based on constraint programming and use CPU, memory, and bandwidth for MLU optimization and VM reuse. But they have more VM migration costs (Dong et al., 2015). Group packing algorithms based on stochastic bin packing are used in the area of solving virtual machine placement problems and their aspect is using random variables to predict future bandwidth and helping to reduce APM's in data centers (Wang et al., 2011). The VMP problem was also addressed as a linear programming (LP) problem (Speitkamp et al., 2010) proposed linear programming formulations to address issues with server consolidation problems. They focused on minimizing both the energy and the hardware costs. This work is suitable for data centers with up to 600 physical machines and applicable for larger problems with more than 600 servers.

Power management has been used at the data center level in one of the first works done by (Pinheiro et al., 2001). The authors describe a method for reducing the power consumption of a heterogeneous cluster of nodes that serves numerous web apps in their paper. The main technique applied is balancing the workload of physical machines and switching idle nodes off. The algorithm periodically monitors the load on the resources (disk storage, CPU, and network interface) and decides to switch nodes on/off to optimize the overall power consumption and provide the expected performance. Heuristic algorithms First-Fit Decreasing (FFD) and its other variations such as Best-Fit Decreasing (BFD) were employed to address the VMP problem. In these variations, the VMP problem has been formulated as a classic bin packing problem which is Np-Hard. These are simple algorithms and have low complexity compared to optimization algorithms. But the solutions obtained from these algorithms are low quality because they do not consider the different objectives that also need to be optimized. The results with this are not able to achieve the balanced usage of the different resources. Metaheuristics such as simulated annealing (Hyser et al., 2007) and ant colony optimization (Gao et al., 2013) are suggested for solving VMP. Some authors suggested heuristics. Some of them are simple greedy algorithms (Wood et al., 2009; Salehi et al., 2012) or straightforward selection policies. Chase et al. (2001) have addressed the problem of energy-efficiency management of homogeneous resources in cloud hosting centers. The main challenge is to identify the resource demand of each application at its current request load level and properly allocate resources in the most efficient way. To deal with this problem they have applied an economic framework: services ''bid'' for resources in terms of volume and quality. Energy consumption is reduced by switching off idle servers. They have addressed the VMP problem by applying the statistical ''flip-flop'' filter, which helps to reduce the number of reallocations and can lead to a stable and efficient control.

The Genetic Algorithm Based Approach (GABA) is used widely to solve VMP problems and deals with multiobjective optimization (Mi et al., 2010). Adamuthe & Patil (2018) presented multi-objective virtual machine placement in a cloud environment focusing on different objectives. Profit maximization, load balance, and resource waste minimization are the goals. The paper compared the performance of three algorithms namely GA, NSGA, and NSGA-II.

Later several metaheuristic algorithms have been studied and utilized to address the problem. Evolutionary algorithms such as the genetic algorithm (GA) have been studied in many works. (Adamuthe et al., 2013) added a non-dominated sorting genetic algorithm-II (NSGA-II) to solve the virtual machine placement problem, which was formulated as a multiobjective optimization problem. The work aimed at maximizing profits and the load balance among the physical machines and minimizing the resource wastage However, the work was evaluated for small-sized problems with up to 60 VMs.

A detailed survey on the VMP problem is presented in the paper (Usmani & Singh, 2016) Classified VMP problems concerning different approaches depending on the goal of placement. With the recent survey allocation of virtual machines in cloud data centers (Mann, 2015) the problem is classified in three ways.

Resource Types

Most of the works and studies focus on the CPU as their main and critical resource (Batista et al., 2007; Breitgand & Epstein, 2011) and characterize the physical machines in terms of their CPU loads. Some works make the problem multidimensional

by considering other resources such as memory and bandwidth (Van et al., 2019; Zhu et al., 2017)

Considered V.M. set

Most studies consider the placement of all virtual machines in the data center at once (Beloglazov et al., 2012; Biran et al., 2012) However, some of them consider the placement of a single virtual machine or a set of virtual machines belonging to the same application (Breitgand & Epstein, 2011; Jayasinghe et al., 2011)

Objectives

Optimizing energy consumption is the main objective in most works (Ghribi et al., 2013; Beloglazov et al., 2012) However, there are differences in the studies that model energy consumption objectives. Several works consider the number of active physical machines as the main factor of energy consumption (Beloglazov & Buyya, 2010). Aside from energy usage, another goal in some projects is to reduce the number of overloaded physical equipment due to the performance loss that overloads cause (Beloglazov & Buyya, 2012). The cost of virtual machine migration was also included in several research (Wood et al., 2009).

3. Problem Formulation and Methodology

3.1 Virtual Machine Placement Problem

VMP is a process of mapping VMs onto a set of PMs in a data center to minimize total power consumption and maximize resource utilization. The VMP is an NP-hard problem due to its complexity and capacity constraints and the multiple candidate assignments of VMs to PMs. The VMP problem can be formulated as an optimization problem to find optimal solutions. There can be many objectives with the study of VMP. In this paper, the optimization goal is to minimize energy consumption by satisfying the given constraints. The VMP problem is formulated as a bin packing problem. The general bin problem is summarized as follows, given a set of indivisible items. Each item has certain weights and a set of bins with variable sizes. The objective is to pack all items into many bins such that the number of used bins is minimized. In this paper, the VMP problem is addressed as a bin packing problem and the problem formulation is like the problem presented in (Abohamama & Hamouda, 2020).

In this research, physical machines are represented as a resource. Consider that there are '*M'* VMs and *'N'* PMs. It is assumed that the total demand of all the VMs is less than the total capacity of the PMs. The main objective is to minimize the energy consumption of PMs. The constraints are as follows,

- Each virtual machine must be assigned to exactly one physical machine.
- Each physical machine must have enough resources for the assigned virtual machine.

The VMP problem can be formulated as follows,

$$minimize\ f(x) = \sum_{j=1}^{N} y_j \times \left(\left(P_j^{busy} - P_j^{idle} \right) \times ut_j^{cpu} + P_j^{idle} \right) \quad (1)$$

where,
i ∈{1, 2, ... , M}, and j ∈ {1, 2, ... , N}.

$f(x)$ is the total power consumption of the used physical machines.
y_j is a binary variable that indicates if PH_j contains virtual machines or not.
P_j^{busy} is the maximum power consumption of physical machine PH_j.
P_j^{idle} is the minimum power consumption of physical machine PH_j.
(as suggested in (Beloglazov et al., 2012), $P_j^{idle} \approx 0.6 * P_j^{busy}$).
ut_j^{cpu} is the CPU utilization ratio of physical machine PH_j.
ut_j^{cpu} is formulated as follows:

$$ut_j^{cpu} = \sum_{i=1}^{M} x_{ij} \times \frac{V_{cpu_i}}{P_{cpu_j}} \quad (2)$$

x_{ij} is a binary variable that indicates if VM_i is assigned to PH_j or not,
V_{cpu_i} is the CPU demands of virtual machine VM_i
P_{cpuj} is the CPU capacity of physical machine PH_j

3.2 Harmony Search Algorithm for VMP

Harmony search (HS) is a meta-heuristic search algorithm that tries to mimic the improvisation process of musicians in finding a pleasing harmony. In recent years, due to its advantages, HS has received significant attention by researchers HSA is simple to use, quickly converges to the best solution, and produces a good enough solution in a reasonable amount of time. The merits of the HS algorithm have led to its application to optimization problems of different engineering areas (Lee et al., 2005; Fesanghary et al., 2009). In literature, HS has shown promising performance in solving difficult optimization problems and different versions of this algorithm have been developed (Mahdavi et al., 2007; Omran & Mahdavi, 2008). In this section, the basic HS algorithms and proposed variations for solving VMP are presented.
Harmony search algorithm steps are as follows,

- Harmony search uses some parameters to develop from dissonance music to harmony. These parameters comprise HMCR and PAR (Yun, 2021). The algorithm initializes algorithm specific parameters- Harmony Memory Size, Harmony Memory Considering Rate (HMCR) and Pitch Adjusting Rate (PAR).

Harmony Memory Considering Rate (HMCR) - which is the probability factor and range between $0 \leq HMCR \leq 1$
Pitch Adjusting Rate PAR which is also the probability factor and range between $(0 \leq PAR \leq 1)$

- Generate random vectors or harmonies as many as Harmony Memory Size (HMS) and save them in the Harmony Memory (HM).
- Generate new harmonies by considering HMCR and PAR and check if the new HM is better than the saved one and update the HM.
- Repeat this process until the stopping condition (Like the maximum number of iterations) satisfies.

The sample solution representation used for solving the virtual machine placement problem is shown in figure 1. The symbols "*VM"* and *"PM"* are virtual machine and physical machine respectively. One-dimensional array represents the solution. Size of the solution depends on the number of virtual machines.

VM_1	VM_2	VM_3	VM_4	VM_5	VM_6	VM_7	VM_8
PM_1	PM_3	PM_1	PM_3	PM_3	PM_2	PM_1	PM_3

Fig. 1. Solution representation

Algorithm 1: Pseudocode for the HSA with static PAR

```
Input: no_vm, no_pm ,vm_cpu, pm_cpu, PAR, max _itr, HMS, HMCR
Output: Minimum fitness value(energy)
Initialize
Initialize vm, pm instance
Initialize harmony memory HM with HMS random solutions
Procedure
Begin
  while current_iteration <max _itr  do
    while current_hm ≤ HMS do
      while current_vm ≤ number_of_vm do
        if (rand() <HMCR)
            d =current_hm;
          if (rand() < par)
            while i < number_of_vm
              temp_HM[i] = HM[d][i]
            od
            temp_HM[current_vm] = rand() % number_of_pm;
            fitness= calculate_fitness(temp_HM)
              if fitness < calculate_fitness(HM[current_vm])
                HM[current_vm] = temp_HM[current_vm]
            fi
        fi
       fi
        get_min_fitness(temp_HM[current_vm])
      od
    od
  od
end
```

3.2.1 HSA with Static PAR values

Performance of many heuristic algorithms are influenced by algorithmic specific parameters. Exploration and exploitation strongly influenced by algorithmic specific parameters. For harmony search algorithms, HMCR and PAR values are crucial for fine-tuning the results and modifying the algorithms convergence to achieve the best answer. PAR is most significant to fine tune HSA. The probability factor PAR is a number that ranges from 0 to 1. In the basic experimental work, different PAR values are tested to analyze the performance of HSA. PAR values of 0.9, 0.7, 0.5, 0.3, and 0.1 are used in execution. The same initialized PAR value is used till the termination criteria is satisfied.

3.2.2 HSA with Adaptive PAR

HM diversity influences the value of PAR. In HSA with adaptive PAR, the PAR values are modified after each iteration. This study presents four methods for calculating adaptive PAR values. Three adaptive PAR strategies are taken from literature. The details of adaptive PAR strategies are as given below. Fig. 2 shows the changes of PAR values with respect to iterations.

Strategy 1: Eq. (3) is used to modify the value of PAR (Mahdavi et al., 2007). PAR strategy is increasing. The parameters PARmax and PARmin are set to 1.0 and 0.45, respectively.

$$PAR = PAR + \left(\frac{PAR_{max} - PAR_{min}}{NI}\right) \times gn \tag{3}$$

Strategy 2: Eq. (4) is used to modify the value of PAR (Mahdavi et al., 2007). It is increasing PAR strategy. The parameters PARmax and PARmin are set to 0.5 and 0.1, respectively.

$$PAR = PAR + \left(PAR_{min} + \frac{PAR_{max} - PAR_{min}}{NI} \times gn\ \right) \tag{4}$$

Strategy 3: Eq. (5) is used to modify the value of PAR. It is decreasing PAR strategy. The parameters PARmax and PARmin are set to 1.0 and 0.85, respectively.

$$PAR = \left(2 \times PAR_{max} - \left(\frac{PAR_{max} - PAR_{min}}{NI} \times gn\right)\right) \div 2 \tag{5}$$

Strategy 4: Eq. (6) is used to modify the value of PAR (Adamuthe & Nitave, 2020). It is decreasing PAR strategy. The parameters PARmax and PARmin are set to 1.0 and 0.85, respectively.

$$PAR = PAR - \left(PAR_{min} - \frac{PAR_{max} - PAR_{min}}{NI} \times gn\right) \tag{6}$$

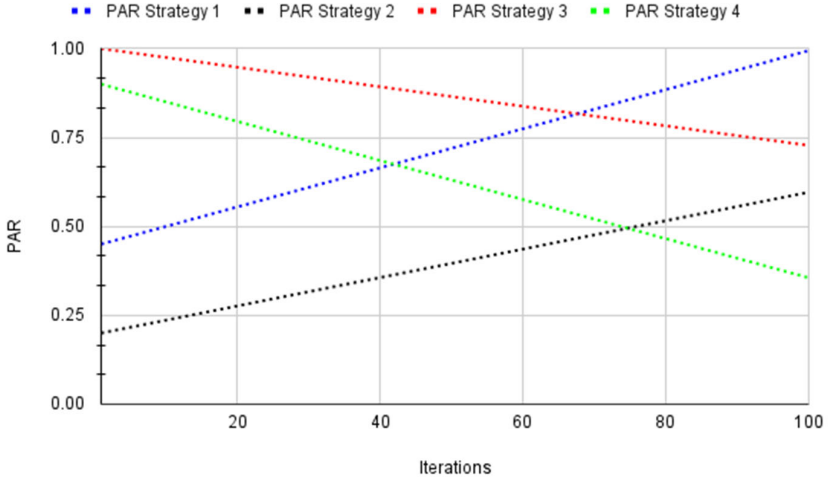

Fig. 2. Adaptive PAR strategies- change in PAR values

3.2.3 Hybrid Harmony Search Algorithms

In literature, local search algorithms are investigated for solving different optimization problems. Local search can be used to solve issues involving maximizing a criterion among a set of potential solutions. In this paper, we are using a local search approach to improve the exploitation ability of harmony search algorithms. Harmony search is hybridized with simulated annealing (SA) which is a probabilistic method for estimating a function's global optimum. The algorithm's name originates from metallurgy's annealing process, which involves heating and cooling a material to increase the size of its crystals and remove flaws.

Algorithm 2. Pseudocode for the hybrid HSA with simulated annealing and local search

```
Input: no_vm, no_pm ,vm_cpu, pm_cpu, PAR, max _itr, HMS, HMCR
Output: Minimum fitness value(energy)
Initialize
Initialize vm, pm instance
Initialize harmony memory HM with HMS random solutions
Initialize PAR with best adaptive PAR strategy
Initialize the Temperature (T) for the calculation
Procedure
Begin
  while current_iteration <max _itr  do
    while current_hm ≤ HMS do
      while current_vm ≤ number_of_vm do
        if (rand() <HMCR)
            d =current_hm;
             if (rand() < par)
              while i < number_of_vm
                    temp_HM[i] = HM[d][i]
               od
              while i< number_of_vm
                   temp_HM[current_vm] = rand() % number_of_pm;
                   fitness= calculate_fitness(temp_HM)
               if fitness < calculate_fitness(HM[current_vm])
                     HM[current_vm] = temp_HM[current_vm]
               else
                  Δ=-(HM[current_vm]-fitness)
                  result=e^(Δ/T)
                   if(rand1<result)
                  HM[current_vm] = temp_HM[current_vm]
                   update Tempreture (T)
                   fi
              fi
            od
          fi
        fi
          get_min_fitness(temp_HM[current_vm])
      od
    od
  od
end
```

4. Results and Analysis

4.1 Simulation details

This section presents the simulation details for the investigation and different data sets of virtual machine placement problems. The evaluation of our proposed algorithm is performed through different sets of experiments in a simulated environment. Datasets generation programs are written to generate the datasets for the experimental evaluation. We developed a python program that can randomly generate a data set for the problems of different characteristics. The datasets have been simulated with heterogeneous types of PMs and VMs. The maximum number of VMs and PMs examined is 600 and 450, respectively. In this study, both virtual machines and physical machines are represented by the most representative

resources- the CPU. Total 15 datasets are generated with normal distribution by considering VMs CPU values between 50 and 150. We have categories them as small, medium, and large data sets to the number of virtual and physical machines. The VM resources are generated by satisfying the constraint of having lower values of v_{cpu} than the available capacities of $P_{cpu.}$ The first data generator program created inputs by random distribution and generated random CPU values of virtual machines in a specified range. The second data generator program created inputs by a normal distribution and distributed virtual machines required CPU values normally. The values of a few parameters are kept fixed as per standards. Table 1 presents the parameter values for dataset generation. Table 2 presents type of method of generation of 15 dataset instances.

Table 1
Specifications for VM and PM

PM CPU	1000 (MIPS)
VM CPU range	50-150 (In MIPS)
PM CPU at busy state	250 (MIPS)

Table 2
Datasets

Dataset No.	PM:VM	Method of generation	Small/medium/large
1	35:50	Random	Small
2	40:100	Random	Small
3	50:120	Random	Small
4	70:150	Random	Small
5	100:200	Random	Medium
6	150:200	Normal	Medium
7	150:300	Normal	Medium
8	200:300	Normal	Medium
9	250:300	Normal	Medium
10	290:400	Normal	Medium
11	300:450	Normal	Medium
12	350:500	Normal	Large
13	400:500	Normal	Large
14	450:600	Normal	Large
15	500:700	Normal	Large

For experimentation, we assumed homogeneous physical machines and heterogeneous virtual machines. Number of physical machines, number of virtual machines, HMCR, PAR, number of iterations, harmony memory size, physical machines CPU value at busy state are passed as input to the programs. The input file for each dataset contains the following information: Harmony memory size, HMCR and PAR values, number of iterations considered, number of physical and virtual machines and their required CPU values, and P_{busy} CPU.

Sample Input file

```
300  - Harmony Memory
0.9    HMCR
0.3    PAR
100    Number of iterations

50     Number of virtual machine
…      CPU values of 50 virtual machine

35    Number of the physical machine
…..  CPU values of 35 physical machine
…… Pbusy values of the physical machine
```

The proposed harmony search algorithm variations are implemented using the 'C++' programming language. The programs are run on a computer with an Intel Core i5-A515-51G processor running at 3.4 GHz and 8 GB of RAM.

4.2. Comparison of HSA with static PAR values

We examined performance of harmony search algorithm with five different PAR values. The PAR values tested from the sets {0.9, 0.7, 0.5, 03, 01}. We examined each static PAR value on 15 datasets.

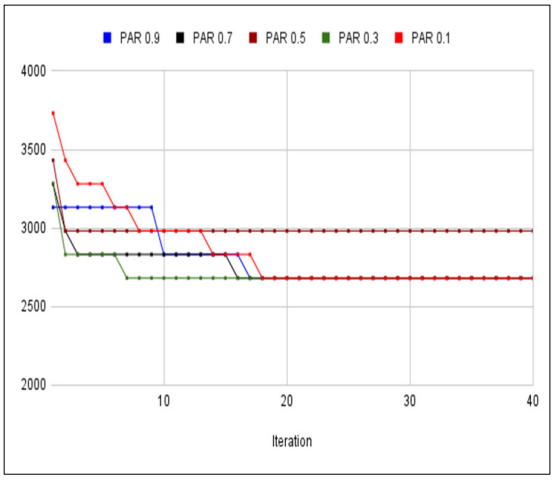

Fig. 3. Instance with 35 PM and 50 VM

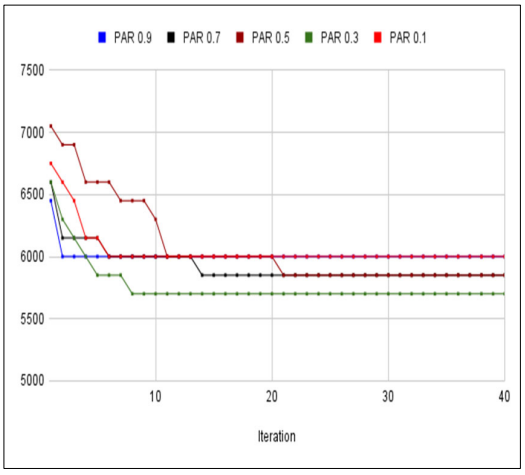

Fig. 4. Instance with 50 PM and 120 VM

Fig. 3 shows that the performance of harmony search algorithms for all tested PAR values except 0.5 is similar for virtual machine placement instances with 35 PM and 50 PM. HSA with a 0.3 PAR value shows rapid convergence towards the optimal solution. Fig. 4 shows that for VMP instances with 50 PM and 120 VM, 0.3 PAR gives the best results and has a faster convergence rate than the others. The HSA with 0.5 and 0.7 converged to the same fitness value. HSA with PAR 0.9 and PAR 0.1 converged to the same value as well.

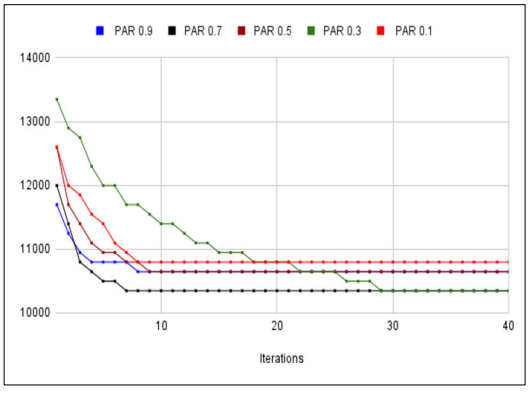

Fig. 5. Instance with 100 PM and 200 VM

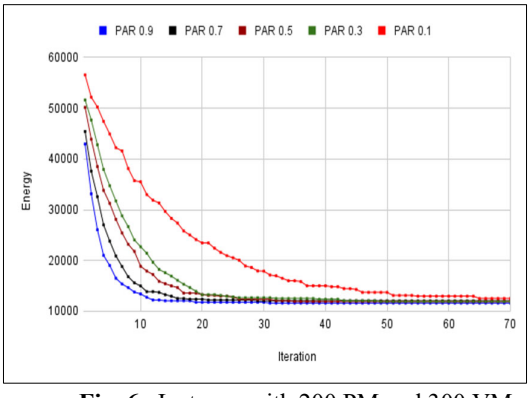

Fig. 6. Instance with 200 PM and 300 VM

Fig. 5 shows that HSA with 0.3 and 0.7 PAR gives the same optimal results for VMP instances with 100 PM and 200 VM. PAR value 0.7 has shown fast convergence and 0.3 is gradually decreasing with slower convergence than 0.7. Figs. (6-8) show the results for medium and large datasets. Results show that PAR has a strong impact in early iterations.

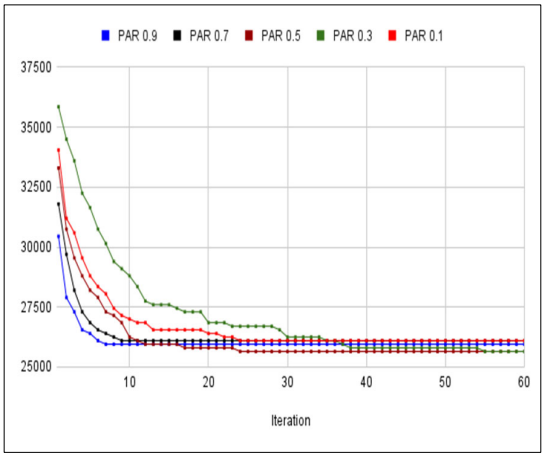

Fig. 7. Instance with 300 PM and 450 VM

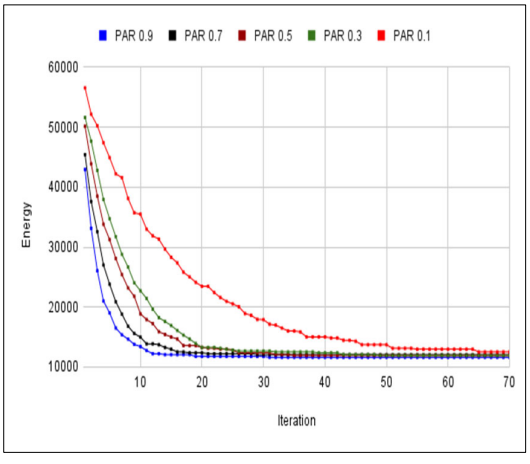

Fig. 8. Instance with 500 PM and 700 VM

Results show that performance of the harmony search algorithm changes with different PAR values. Harmony search algorithm with 0.3 PAR gives better results than other PAR values for large, medium, and small instances. Effect of PAR values on different instances is different.

4.3. HSA with Adaptive Parameter setting

Performance of four adaptive PAR strategies for HS algorithms are tested using 15 different instances. Two approaches increase PAR with increase in iterations. Table 3 presents the comparison of results obtained from the harmony search algorithm with static and adaptive PAR. The comparison was made with the best results obtained with static PAR and adaptive PAR strategy. Adaptive PAR gives better results for 60 percent of the datasets. Adaptive strategy gives better results for all small datasets than static PAR. From an experimentation view, it is difficult to test HSA with PAR values ranging from 0 to 1. Adaptive strategy is found better to achieve optimal or near to optimal solution to a variety of instances.

Table 3
Comparison of HSA with static and adaptive PAR

Dataset No.	Small/medium/large	Best results with static PAR from {0.1, 0.3, 0.5, 0.7, 0.9}	Adaptive PAR
1	Small	2682	**2532**
2	Small	4723	**4573**
3	Small	5851	**5701**
4	Small	7661	**7511**
5	Medium	10348	**10279**
6	Medium	**13601**	13751
7	Medium	**63302**	65102
8	Medium	11926	**11925**
9	Medium	17175	**16725**
10	Medium	**22681**	23431
11	Medium	25650	**25350**
12	Large	**28884**	29334
13	Large	**29184**	29484
14	Large	**34957**	35107
15	Large	35707	**28132**

Fig. 9 shows that all adaptive strategies work well for small instances. Fig. 10 shows that strategy 4 has no improvement for the first 60 iterations followed by rapid improvement. The convergence pattern of HSA with strategy 4 is strange for this instance. The same pattern is seen with other instances 200 PM & 300 VM and 300 PM & 450 VM. Fig. 11 shows that strategy 3 and 4 have a fast convergence towards the optimal value. For small, medium, and large data sets, the harmony search algorithm with decreasing PAR (strategy 3) is found to be better than other strategies.

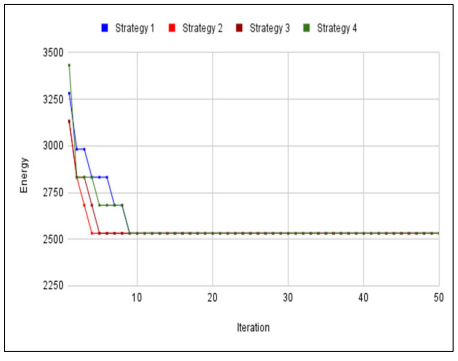

Fig. 9. Instance with 35 PM and 50 VM

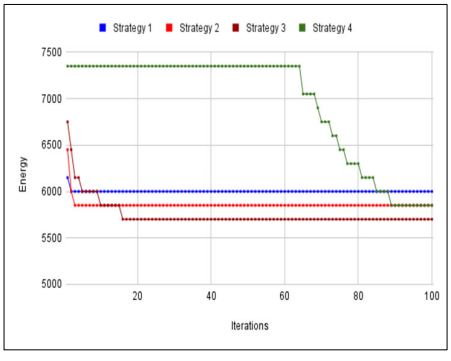

Fig. 10. Instance with 50 PM and 120 VM

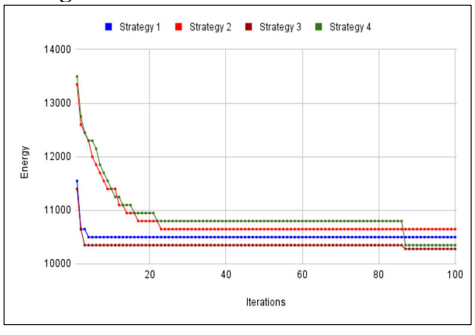

Fig. 11. Instance with 100 PM and 200 VM

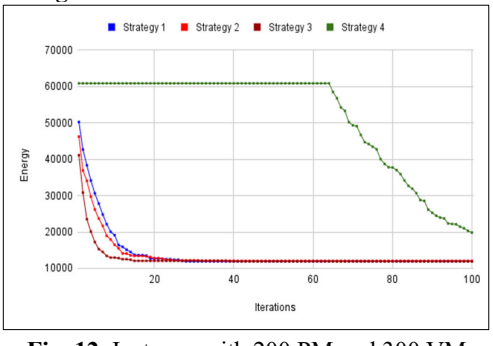

Fig. 12. Instance with 200 PM and 300 VM

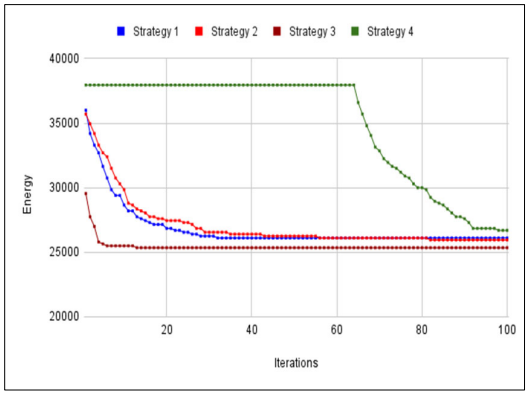

Fig. 13. Instance with 300 PM and 450 VM

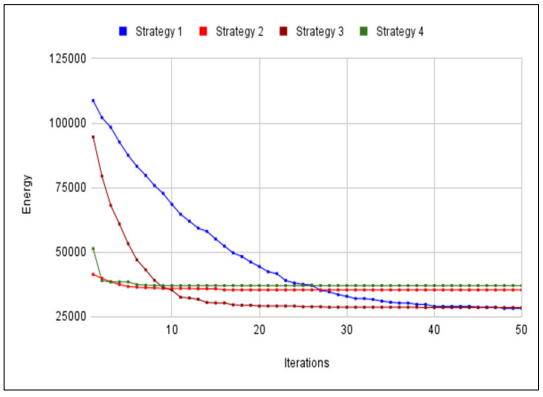

Fig. 14. Instance with 500 PM and 700 VM

4.4. Performance of hybrid Harmony search algorithm (simulated annealing, local search)

The performance of the HSA with simulated annealing compared to the earlier adaptive PAR strategies with small, medium and large instances. Hybrid HSA with simulated annealing gives better results than HSA with adaptive PAR for 60 percent of the datasets.

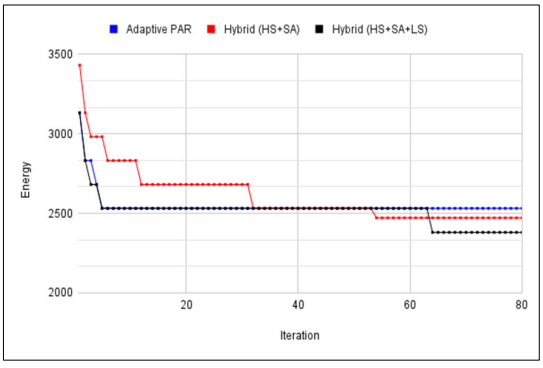

Fig. 15. Instance with 35 PM and 50 VM

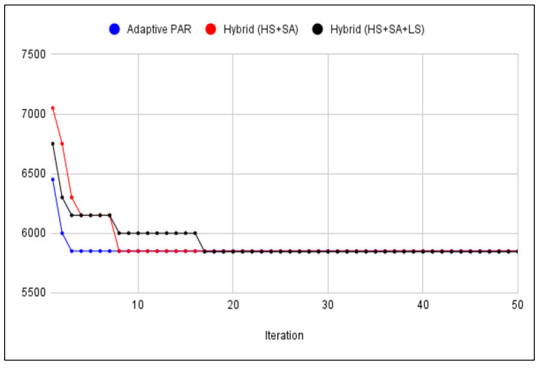

Fig. 16. Instance with 50 PM and 120 VM

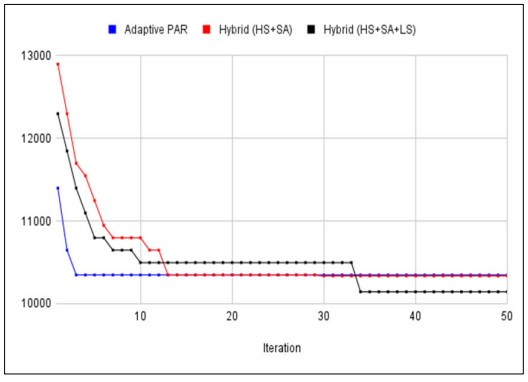

Fig. 17. Instance with 100 PM and 200 VM

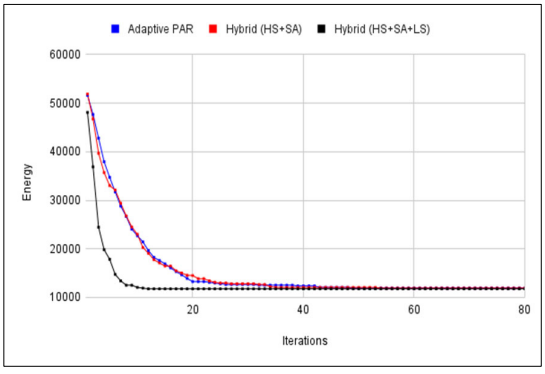

Fig. 18. Instance with 200 PM and 300 VM

Fig. 15 shows that hybrid HSA with local search has better results than adaptive HSA and hybrid HS with SA. Performance of all three algorithms is same for instance with 50 PM and 120 VM. For sample two medium datasets as shown in figure 19 and 20, hybrid HS with local search shows better results and fast convergence. Performance of the other two HS algorithms is similar to each other. Table 4 presents the comparison of results obtained by static HAS, HSA with adaptive PAR and hybrid HSA. Hybrid HS with SA and local search shows faster convergence in the early stage than other variations. Hybrid HS with SA and local search gives better results for 80 percent of the datasets than the other HS variations.

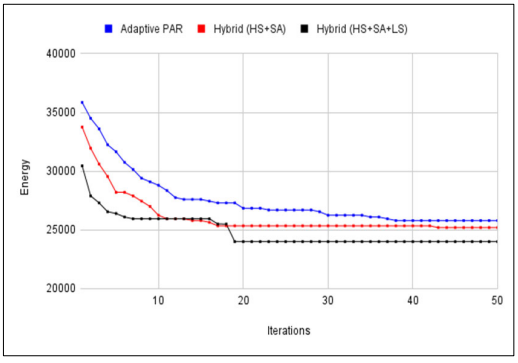

Fig. 19. Instance with 300 PM and 450 VM

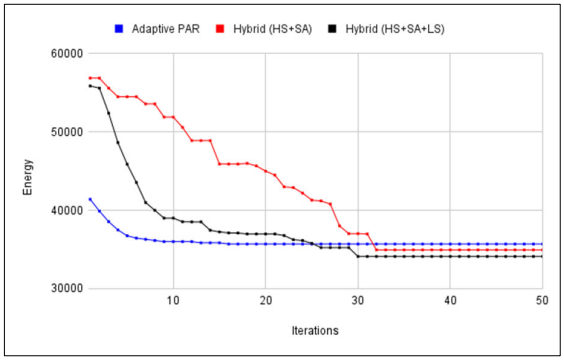

Fig. 20. Instance with 500 PM and 700 VM

Table 4
Comparison of all variations of HSA

Dataset No.	Small/medium /large	Best results with static PAR from {0.1, 0.3, 0.5, 0.7, 0.9}	HSA with Adaptive PAR	Hybrid HSA (HS+SA) with adaptive strategy 3	Hybrid HSA (HS+SA+LS) with adaptive strategy 3
1	Small	2682	2532	2472	**2381**
2	Small	4723	4573	4480	**4475**
3	Small	5851	**5701**	5845	5845
4	Small	7661	7511	7511	**7452**
5	Medium	10348	10279	10336	**10145**
6	Medium	13601	13751	13601	**13235**
7	Medium	63302	65102	62548	**62210**
8	Medium	11926	11925	11925	**11775**
9	Medium	17175	16725	17325	**16525**
10	Medium	22681	23431	22981	**21002**
11	Medium	25650	25350	25200	**24006**
12	Large	28884	29334	29034	**28569**
13	Large	29184	29484	28885	**28230**
14	Large	34957	35107	34127	**34001**
15	Large	35707	**28132**	34956	34122

5. Conclusions

This paper has presented a harmony search algorithm for solving virtual machine placement problems. The main objective is to minimize the energy consumption in cloud data centers. The problem formulation considered the CPU requirements from a virtual machine and physical machine with given constraints. The aim is to minimize energy consumption of cloud data centers with satisfying given constraints.

Random and normal distribution method is used to generate 15 dataset instances of virtual machine placement problem. The dataset is categorized as small, medium and large instances with respect to the number of virtual machines and physical machines in the instance.

The paper presents results of harmony search algorithm, adaptive HSA and two hybrid variations of HSA. Results of HSA with different PAR values are presented. Results of four PAR adaptive strategies are presented. The HS algorithm is hybridized with simulated annealing and local search algorithms. We have observed that with specific values of PAR the harmony search algorithm gives better results and helps to optimize energy. For all datasets PAR value 0.3 shows better results than other tested values. Adaptive PAR gives better results than static PAR for 60 percent of the datasets. Adaptive strategy 3 performed better than the other tested strategies. Hybrid HS with simulated annealing gives better results than adaptive PAR for 60 percent of datasets. Hybrid HS with simulated annealing and local search performs better results than other variations for 80 percent of the datasets.

References

Abdel-Basset, M., Abdle-Fatah, L., & Sangaiah, A. K. (2019). An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in a cloud computing environment. *Cluster Computing*, *22*(4), 8319-8334.

Abohamama, A. S., & Hamouda, E. (2020). A hybrid energy–aware virtual machine placement algorithm for cloud environments. *Expert Systems with Applications*, *150*, 113306.

Adamuthe, A. C., & Patil, J. T. (2018). Differential Evolution Algorithm for Optimizing Virtual Machine Placement Problem in Cloud Computing. *International Journal of Intelligent Systems & Applications*, *10*(7).

Adamuthe, A. C., Pandharpatte, R. M., & Thampi, G. T. (2013, November). Multiobjective virtual machine placement in a cloud environment. In 2013 International Conference on Cloud & Ubiquitous Computing & Emerging Technologies (pp. 8-13). IEEE.

Adamuthe, A., & Nitave, T. (2020). Harmony search algorithm with adaptive parameter setting for solving large bin packing problems. *Decision Science Letters*, *9*(4), 581-594.

Alharbi, F., Tian, Y. C., Tang, M., Zhang, W. Z., Peng, C., & Fei, M. (2019). An ant colony system for energy-efficient dynamic virtual machine placement in data centers. *Expert Systems with Applications*, *120*, 228-238.

Alicherry, M., & Lakshman, T. V. (2013, April). Optimizing data access latencies in cloud systems by intelligent virtual machine placement. In *2013 Proceedings IEEE INFOCOM* (pp. 647-655). IEEE.

Batista, D. M., Da Fonseca, N. L., & Miyazawa, F. K. (2007, March). A set of schedulers for grid networks. In *Proceedings of the 2007 ACM symposium on Applied computing* (pp. 209-213).

Beloglazov, A., & Buyya, R. (2010, May). Energy efficient allocation of virtual machines in cloud data centers. In *2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing* (pp. 577-578). IEEE.

Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. *Future generation computer systems*, *28*(5), 755-768

Biran, O., Corradi, A., Fanelli, M., Foschini, L., Nus, A., Raz, D., & Silvera, E. (2012, May). A stable network-aware vm placement for cloud systems. In *2012 12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (ccgrid 2012)* (pp. 498-506). IEEE.

Breitgand, D., & Epstein, A. (2011, May). SLA-aware placement of multi-virtual machine elastic services in compute clouds. In *12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops* (pp. 161-168). IEEE.

Chaisiri, S., Lee, B. S., & Niyato, D. (2009, December). Optimal virtual machine placement across multiple cloud providers. In *2009 IEEE Asia-Pacific Services Computing Conference (APSCC)* (pp. 103-110). IEEE.

Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., & Doyle, R. P. (2001). Managing energy and server resources in hosting centers. *ACM SIGOPS operating systems review*, *35*(5), 103-116.

Chen, G., He, W., Liu, J., Nath, S., Rigas, L., Xiao, L., & Zhao, F. (2008, April). Energy-Aware Server Provisioning and Load Dispatching for Connection-Intensive Internet Services. In *NSDI* (Vol. 8, pp. 337-350).

Dang, H. T., & Hermenier, F. (2013, November). Higher SLA satisfaction in datacenters with continuous VM placement constraints. In *Proceedings of the 9th workshop on hot topics in dependable systems* (pp. 1-6).

Dong, J., Wang, H., & Cheng, S. (2015). Energy-performance tradeoffs in IaaS cloud with virtual machine scheduling. *China communications*, *12*(2), 155-166

Fesanghary, M., Damangir, E., & Soleimani, I. (2009). Design optimization of shell and tube heat exchangers using global sensitivity analysis and harmony search algorithm. *Applied Thermal Engineering*, *29*(5-6), 1026-1031.

Fidanova, S. (2021). Multiple Knapsack Problem. In *Ant Colony Optimization and Applications* (pp. 9-18). Springer, Cham.

Gao, Y., Guan, H., Qi, Z., Hou, Y., & Liu, L. (2013). A multi-objective ant colony system algorithm for virtual machine placement in cloud computing. *Journal of computer and system sciences*, *79*(8), 1230-1242.

Gartner Estimates, I. C. T. (2007). Industry accounts for 2 percent of global CO2 emissions. *press release*.

Ghribi, C., Hadji, M., & Zeghlache, D. (2013, May). Energy-efficient vm scheduling for cloud data centers: Exact allocation and migration algorithms. In *2013 13th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing* (pp. 671-678). IEEE.

Hyser, C., McKee, B., Gardner, R., & Watson, B. J. (2007). Autonomic virtual machine placement in the data center. *Hewlett Packard Laboratories, Tech. Rep. HPL-2007-189*, *189*.

Jayasinghe, D., Pu, C., Eilam, T., Steinder, M., Whally, I., & Snible, E. (2011, July). Improving performance and availability of services hosted on iaas clouds with structural constraint-aware virtual machine placement. In *2011 IEEE International Conference on Services Computing* (pp. 72-79). IEEE.

Lee, K. S., Geem, Z. W., Lee, S. H., & Bae, K. W. (2005). The harmony search heuristic algorithm for discrete structural optimization. *Engineering Optimization*, *37*(7), 663-684.

Mahdavi, M., Fesanghary, M., & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied mathematics and computation*, *188*(2), 1567-1579.

Mann, Z. Á. (2015). Allocation of virtual machines in cloud data centers—a survey of problem models and optimization algorithms. *Acm Computing Surveys (CSUR)*, *48*(1), 1-34.

Mi, H., Wang, H., Yin, G., Zhou, Y., Shi, D., & Yuan, L. (2010, July). Online self-reconfiguration with a performance guarantee for energy-efficient large-scale cloud computing data centers. In *2010 IEEE International Conference on Services Computing* (pp. 514-521). IEEE.

Mohamadi, E., Karimi, M., & Heikalabad, S. R. (2011). A Novel Virtual Placement in Virtual Computing. *Australian Journal of Basic and Applied Scienes, Australia*, *5*(10), 1149-1555.

Networking, C. V. (2016). Cisco global cloud index: Forecast and methodology, 2015-2020. white paper. *Cisco Public, San Jose*, 2016.

Omran, M. G., & Mahdavi, M. (2008). Global-best harmony search. *Applied mathematics and computation*, *198*(2), 643-656.

Panigrahy, R., Talwar, K., Uyeda, L., & Wieder, U. (2011). Heuristics for vector bin packing. *research. microsoft. com*.

Pinheiro, E., Bianchini, R., Carrera, E. V., & Heath, T. (2001, September). Load balancing and unbalancing for power and performance in cluster-based systems. In *Workshop on compilers and operating systems for low power* (Vol. 180, pp. 182-195).

Salehi, M. A., Krishna, P. R., Deepak, K. S., & Buyya, R. (2012, June). Preemption-aware energy management in virtualized data centers. In *2012 IEEE Fifth International Conference on Cloud Computing* (pp. 844-851). IEEE.

Shuja, J., Gani, A., Shamshirband, S., Ahmad, R. W., & Bilal, K. (2016). Sustainable cloud data centers: a survey of enabling techniques and technologies. *Renewable and Sustainable Energy Reviews*, *62*, 195-214.

Singh, A., Korupolu, M., & Mohapatra, D. (2008, November). Server-storage virtualization: integration and load balancing in data centers. In *SC'08: Proceedings of the 2008 ACM/IEEE conference on Supercomputing* (pp. 1-12). IEEE.

Song, Y., Zhang, C., & Fang, Y. (2008, November). Multiple multidimensional knapsack problem and its applications in cognitive radio networks. In *MILCOM 2008-2008 IEEE Military Communications Conference* (pp. 1-7). IEEE.

Speitkamp, B., & Bichler, M. (2010). A mathematical programming approach for server consolidation problems in virtualized data centers. *IEEE Transactions on services computing*, *3*(4), 266-278.

Tang, M., & Pan, S. (2015). A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers. *Neural processing letters*, *41*(2), 211-221.

Usmani, Z., & Singh, S. (2016). A survey of virtual machine placement techniques in a cloud data center. *Procedia Computer Science*, *78*, 491-498.

Van den Bossche, R., Vanmechelen, K., & Broeckhove, J. (2010, July). Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads. In *2010 IEEE 3rd international conference on cloud computing* (pp. 228-235). IEEE.

Wang, M., Meng, X., & Zhang, L. (2011, April). Consolidating virtual machines with dynamic bandwidth demand in data centers. In *2011 Proceedings IEEE INFOCOM* (pp. 71-75). IEEE.

Wood, T., Shenoy, P., Venkataramani, A., & Yousif, M. (2009). Sandpiper: Black-box and gray-box resource management for virtual machines. *Computer Networks*, *53*(17), 2923-2938.

Xia, B., & Tan, Z. (2010). Tighter bounds of the First Fit algorithm for the bin-packing problem. *Discrete Applied Mathematics*, *158*(15), 1668-1675

Zhu, W., Zhuang, Y., & Zhang, L. (2017). A three-dimensional virtual resource scheduling method for energy saving in cloud computing. *Future Generation Computer Systems*, *69*, 66-74.