Article

Not peer-reviewed version

Cloud Computing Transformed IT Organizations With Its On-Demand Model, Notably Infrastructure as a Service (IaaS). Cloud Providers Manage Extensive Physical Devices, Consuming Substantial Energy and Bandwidth for Data Traffic During Virtual Machine D

Ketema Deresa Biru *

Posted Date: 5 February 2024

doi: 10.20944/preprints202402.0170.v1

Keywords: cloud computing; data center; energy efficient virtual machine placement; physical machine

Article

Bi-Objective Optimization of Bandwidth Resources and Energy Consumption for Efficient Virtual Machine Placement in Cloud Computing

Ketema Deresa [1] and Frezewd Lemma [2,*]

[1] Department of Computer Science and Engineering, Oda Bultum University, Oromia, Ethiopia
Ketemaderesa123@gmail.com

[2] Department of Computer Science and Engineering, Adama Science and Technology University, Oromia, Ethiopia

* Correspondence: frzwdl@gmail.com

Abstract: Cloud computing transformed IT organizations with its on-demand model, notably Infrastructure as a Service (IaaS). Cloud providers manage extensive physical devices, consuming substantial energy and bandwidth for data traffic during virtual machine deployment. Balancing energy and bandwidth are critical. This thesis presents BOHGOA integrated with ACO to optimize virtual machine placement in clouds, simultaneously considering bandwidth and energy. It yields Pareto-optimal solutions for this trade-off. Using CloudSim, BOHGOA outperformed GA, ACO, and FFD algorithms, reducing bandwidth consumption by 54.14%, 32.11%, 57.47% (240 VMs) and 38.12%, 22.76%, 47.05% (500 VMs) respectively. Additionally, it decreased physical machine energy usage by 37.70%, 34.01%, 40.14% (240 VMs) and 27.50%, 22.28%, 30.52% (500 VMs) respectively. These results underscore BOHGOA's effectiveness in optimizing cloud VM placement.

Keywords: Cloud Computing,Data Center, energy efficient Virtual Machine Placement, Physical Machine

Introduction

Cloud computing has transformed computing services from purchasable parts to online services accessible through the cloud. This shift has spurred applications like big data analysis, IoT, and machine learning to utilize cloud computing. It offers flexible resources like processing power, applications, networks, and storage (Hashem et al., 2015). Pay-as-you-go models have popularized resource sharing, fostering an innovative business model among commercial applications. Leading providers like Microsoft, Amazon, and Google offer IaaS, PaaS, and SaaS, enabling users to access services anytime, anywhere [3]

However, increased cloud infrastructure usage has led to a surge in energy consumption within data centers, impacting the environment due to high energy demands. The energy consumption of data centers, particularly host energy, depends on usage patterns [10]. Furthermore, VM placement impacts network traffic, affecting performance due to potential congestion between distant VMs [16]. Virtualization, a foundational aspect of cloud computing, aims to optimize resource use but presents challenges in achieving eco-friendly cloud operations [11].

Despite various approaches to improve VM placement, bi-objective minimization of bandwidth and energy in cloud data centers remains underexplored. As the problem complexity grows, finding optimal solutions becomes computationally demanding (Fatima et al., 2018). Addressing these issues, an evolutionary algorithm employing bi-objective optimization aids in finding near-optimal solutions by balancing conflicting objectives iteratively [14]. Hence, this study proposes a bi-objective hybrid genetic optimization algorithm to efficiently place virtual machines in the cloud, reconciling the conflict between bandwidth resource and energy consumption objectives during optimization.

 doi:10.20944/preprints202402.0170.v1

Problem Statements

Data centers, pivotal in modern digital infrastructure, face a major challenge due to their high energy consumption, leading to increased costs and carbon emissions. Efficient placement of virtual machines (VMs) in cloud computing impacts infrastructure performance, considering factors like bandwidth resources and energy consumption.

Large applications, such as those based on Map Reduce, require multiple communicating VMs, straining bandwidth resources like links and switches in the cloud's Fat-Tree topology [19]. Inefficient VM deployment results in resource consumption within data centers, prompting various proposed solutions. While single-objective optimization falls short in effectively deploying VMs, multi-objective optimization techniques have gained traction [5][6].

The weighted sum approach, a traditional method for multi-objective optimization, yields a single optimal solution based on assigned weights but doesn't comprehensively balance trade-offs between objectives [7].

Failure to consider bandwidth resources might lead to network congestion, while ignoring energy usage could result in inefficient energy consumption during VM placement. Our focus is on proposing an efficient VM placement strategy that optimizes both bandwidth resources and energy consumption simultaneously, aiming for a balanced trade-off between these dual objectives in the cloud environment.

Contributions

Our main contribution lays particular emphasis on bi-objective optimization of bandwidth resource and energy consumption for efficient VMP. To summarize, the contributions of this paper are as follows:

- This study introduced bi-objective optimization techniques for optimizing bandwidth and energy simultaneously for efficient VMP in cloud computing.
- To tackle the bi-objective optimization problem, we proposed a bi-objective hybrid genetic optimization algorithm.
- The proposed bi-objective hybrid genetic optimization algorithm combines genetic algorithms with ACO to enhance the search for optimal solutions.
- We utilized pareto set for non-dominated solution selection

Therefore, we have addressed the challenge of optimizing both objectives aims to achieve a tradeoff between them in cloud environments through a bi-objective optimization approach.

The remainder of this paper is organized as follows. Section 2 summarizes related works. Section 3 The Proposed Methodology, Section Experimental Results and discussion. Section 5 Conclusion and Future Work.

1. Related work

To explore the gap, an extensive literature review based on the previous domain of studies were explored. Following the collection of all relevant materials, a comparison, and contrast of studies on the subject is undertaken. So, in this related work, algorithms, methods, and different procedures have been investigated based on recent studies and journals.

[4] formulate the virtual machine placement problem as a minimization objective to reduce the power consumption in a data center. They have proposed a modified discrete firefly algorithm (MDF-PC) to reduce the power consumption of the data center by efficiently allocating the virtual machine to the appropriate physical machine such that the waste is very minimal. The fundamental aim of the proposed work is to schedule the virtual machine as densely as possible on a minimal number of servers using the proposed modified discrete firefly algorithm for power consumption. An objective function is modeled to estimate the power consumption of the data center. The proposed discrete firefly algorithm is compared with the native genetic algorithm (GA) and particle swarm optimization (PSO) algorithms in terms of minimizing energy consumption, showing that the proposed algorithm works better. However, bandwidth resources were not considered.

[8] proposed a virtual machine consolidation algorithm based on the dynamic load mean and multi-objective optimization in cloud computing, in which the host load status is comprehensively measured based on multidimensional resources and the dynamic characteristics of the system load are considered. The optimized ant colony algorithm is then used to obtain the optimal mapping between the virtual machines and the hosts to achieve multiple objectives such as resource utilization, energy consumption, migration, and communication overhead optimization. They compared the dynamic load mean virtual machine consolidation (DLMMVMC) with the Ant Colony system virtual machine consolidation (ACS-VMC) in terms of convergence speed.

Finally, the outcomes of the experiment demonstrate that the DLMM-VMC is successful in reducing energy consumption, resource utilization, and migration overhead when compared with the ACS-VMC. This paper ignores the energy consumption generated by other devices in the data center and its impact on the system's performance. As well as bandwidth resources usage due to traffic demand between virtual machines, is not properly studied.

[17] developed an initial virtual machine that was placed fault-tolerantly using a multifaceted optimization strategy. To manage the virtual machine fault-tolerant placement process for virtual machine static placement in star topological data centers, a virtual machine fault-tolerant management system is established in this work at the unified resource layer. The multi-objective optimization model of initial virtual machine fault-tolerant placement, which is based on several criteria, is suggested to be solved using a heuristic ant colony algorithm. The service-providing VMs are placed by the ant colony algorithms, and the redundant VMs are placed by the conventional heuristic algorithms. According to experimental results from simulation, the proposed approach produces a better VM fault-tolerant placement solution than the conventional first fit (FF) or best fit descending (BFD) approach. But multiple VM failures and bandwidth resources were not taken into account.

According to [12] a multi-objective linear integer programming approach has been presented that aims to simultaneously optimize the number of virtual machines hosted, resource usage, and active servers to lower power consumption. Weighted-sum approach which is traditional multi-objective optimization is used, and implemented in the Julia package MultiJuMP, to obtain non-dominated solutions to the multi-objective problem. They claimed that by setting these new objectives, it would be possible to increase client satisfaction while cutting down on operational costs for data centers. Lastly, they note that the heterogeneous DCs perform better in terms of the number of hosted VMs, the amount of resource wastage, the number of used PMs, and the amount of power consumption. However, they failed to provide a good compromise between the conflicting objects and failed to consider the data centers' bandwidth resource usage.

(Xing et al., 2022). The proposed ACO algorithm aims to minimize energy and networking components during VM placement using a weighted sum approach, which is suitable for a single optimization approach. The approach aggregates two objectives into one objective.

A mathematical model to calculate the power consumption of networking devices based on inter-VM data transfer was proposed. However, this algorithm suffers from issues such as unstable convergence speed, the significant impact of parameter settings on results, and unstable results due to randomness. This study did not properly consider the tradeoff between bandwidth resources and energy consumption, where one objective is not dominating another but rather getting a non-dominated optimal solution. They used an approach, which is a valuable method for single-objective optimization by considering the relative importance of objectives, and they overlooked important trade-offs between the objectives and failed to capture the true nature of the problem.

[13] Suggested an energy-efficient virtual machine allocation framework to address the energy consumption, resource wastage, and QoS degradation of cloud data centers. An online multi-resource feed-forward neural network (OM-FNN) was used to predict the different resource demands. Based on the prediction result, virtual machine allocation was done on energy-efficient PMs. A novel OM-FNN predictor is developed to forecast the utilization of multiple resources simultaneously with lesser time and space complexity as compared to multiple conventional neural networks. Future applications are grouped into clusters based on their predicted resource

 doi:10.20944/preprints202402.0170.v1

requirements, and the number and type of virtual machines suitable for a cluster are scaled automatically. The selected auto-scaled virtual machines are placed on energy-efficient servers by applying the proposed multi-objective virtual machine placement algorithm. The result shows that the Online Multi-Resource Feed-forward Neural Network (OM-FNN) predictor shows better accuracy and lesser time and space complexity over a traditional single-input and single-output feed-forward neural network (SISO-FNN) predictor. However, bandwidth resources of VMs according to their inter-dependency while considering resource management are not considered.

[1] study, the energy consumption of the data centers is minimized by minimizing the number of physical servers by using the hybrid virtual machine placement algorithm. An improved permutation-based GA (IGA-POP) and resource-aware best-fit strategy virtual machine placement algorithm is proposed. The number of active servers is reduced through efficient, balanced usage of the multidimensional resources of active servers. The experimental outcomes demonstrated that the suggested VMP algorithm is superior to the best fit (BF), genetic algorithm (GA), sine-cosine algorithm (SCA), and best fit (BF). However, bandwidth resources in the data center were not considered.

[2].have formulated the problem of virtual machine positioning as just an NP-hard problem to obtain effective use of total memory resources and the full use of treatment resources. This problem formulation was addressed by the ACO algorithm. The proposed algorithm is built to efficiently deal with ample solution space and obtain a collection of non-dominated solutions. The main focus of this study is to put forward a virtual machine migration technique in the cloud environment that improves CPU, memory utilization and traffic minimization in the cloud data center using the ACO algorithm. However, in this study, the server's energy consumption was not taken into account properly.

2. The Proposed Methodology

I have used BOHGOA in my proposed approach to achieve the optimal mapping of virtual machines to physical hosts. Since we are optimizing two solutions (bandwidth resources and energy consumption), I used a bi-objective function. This approach is used to optimize bandwidth resources and energy consumption simultaneously. To strengthen the performance of the BOHGOA, it has been integrated with ACO. his integration aims to enhance the algorithm's capabilities and improve its effectiveness in solving optimization problems. Population-based approaches are typically used by these bi-objective optimization techniques to find the best solutions. Additionally, to tradeoff between the two objectives described above, I used the Pareto-dominance concept during the selection process. For the bi-objective optimization function, Pareto optimal solutions, or Pareto Set (PS), are formed, in which there is no solution dominated by any other solution.

Problem formulation

The virtual machine placement problem is formulated as a minimization objective to reduce bandwidth resource and power consumption in the data center. The objective is to minimize both bandwidth resource and power consumption in the data center. consumption and minimization of energy consumption for efficient VMP problems as the following Equation (1) and (2).

Minimization formulation for Bandwidth resource consumption

$$\text{Minimize} \sum R_{B_{total}} = \sum_{vi \in V} \sum_{vj \in V} N_{p(vi),(vj} \cdot T_{vi,vj} \quad \text{eq (1)}$$

Minimization formulation for Energy consumption.

$$\text{Minimize} \sum EC(PM) = \sum P_{(pj)} = y_{pj} \cdot \left(p_j^{idle} + (1-k) \cdot p_j^{full} \cdot u_{pj}^{cpu}\right) \quad \text{eq (2)}$$

Subject to:

$$\bigcup_{pj \in p} V p_j = V \quad \text{eq (3)}$$

 doi:10.20944/preprints202402.0170.v1

$$V_{pi} \bigcap V\, p_j = \emptyset, \quad \forall p_i, \forall p_j \in P, p_i \neq p_j \qquad \text{eq (4)}$$

$$\sum_{vi \in V_{p_j}} R_{vi}^{cpu} \leq C_{pj}^{cpu} \qquad \text{eq (5)}$$

$$\sum_{vi \in V_{p_j}} R_{vi}^{mem} \leq C_{pj}^{mem} \qquad \text{eq (6)}$$

$$B_{ei}^{Csm} < B_e^{max} \qquad \text{eq (7)}$$

Where Constraint in eq (3) ensures that all VMs in V must be successfully placed. Constraint in eq (4) ensures that each VM is placed on a single PM. Constraints in eq (5) - (6) define that the CPU and memory resources occupied by all VMs on PM pj cannot exceed the maximum CPU and memory capacities of pj, respectively. Constraint in eq (7) ensures that the bandwidth consumption of link ei cannot exceed a threshold value, B_e^{max}.

Bi-objective Hybrid Genetic Optimization Algorithm

This section mainly focuses on the Bi-objective optimization algorithm, Bi-objective Hybrid Genetic optimization Algorithm that used in my approach. A genetic algorithm is a powerful global optimization method that can effectively seek optimum solutions in a large research space.

The limitation of genetic algorithms is, that they often require numerous iterations to converge on the optimal solution. This is because genetic algorithms rely on the principles of evolution, which means that they required to repeatedly generate and evaluate new candidate solutions to find the best possible solution. This repeated process of generating new solutions can result in many redundant iterations, which can lead to inefficiencies in the search process. Additionally, genetic algorithms can sometimes get trapped in local optima, which can limit their ability to search the global optimum.

Another limitation of GA is that, it has no feedback mechanism to guide the search process. The algorithm generates a set of candidate solutions, evaluates their fitness, and selects the best solutions for further evolution. However, there is no mechanism to provide feedback on the quality of the solutions or guide the search toward more promising areas of the search space.

By combining the strengths of both genetic and ACO, it is possible to improve the efficiency and effectiveness of virtual machine placement in cloud computing. The genetic algorithm provides a powerful global search capability to explore the search space and find high-quality solutions, while ACO helps to keep from fall in local optima and quickly converge to the optimal solution.

Ant Colony Optimization (ACO) Algorithm

ACO algorithm is a metaheuristic inspired by the behavior of real ants in their search for the shortest path to food sources. Ants tend to choose the paths marked by the strongest pheromone concentration. The ACO algorithm is an essential system based on agents that simulates the natural behavior of ants, including the mechanisms of cooperation and adaptation. Pheromone can be deposited on the components and/or the connections used in a solution depending on the problem(Duan & AI, 2016).

Integration of GA with ACO

GA is a powerful optimization algorithm known for its ability to generate diverse solutions for VM placement. It is used to generate a diverse set of solutions, while the ACO algorithm is used to refine these solutions and ensure that the algorithm does not get stuck in local optima. The GA algorithm generates a population of individuals that represent candidate solutions. Fitness function is used to evaluate the quality of these solutions considering factors such as bandwidth resource utilization and energy usage.

 doi:10.20944/preprints202402.0170.v1

The best-fit individual with the highest fitness value is then selected to be used as the initial pheromone for the ACO algorithm.

$$I_i^S(0) = \varepsilon . G_n \qquad eq(8)$$

Here, $I_i^S(0)$ represents the initial pheromone, ε is a constant and for this work, we have chosen, $\varepsilon = 0.1$, and it determines the balance between exploration and exploitation of the solution space G_n is the GA's optimal solution that is selected based on its fitness value.

ACO works by simulating the behavior of ants that deposit pheromones on the ground as they search for food. The pheromones represent the solutions' quality found by the ants. In the ACO algorithm for VMP, ants construct their paths by selecting physical machines for each VM based on the available heuristic information and the pheromone trails left by other ants. These paths represent potential configurations for VM placement.

$$p_{(i,j)} = \frac{(\tau(i,j)^{\alpha} . (\eta_{(i,j)})^{\beta}}{\sum_{j \in pj}[\tau(i,j)]^{\alpha} . [\eta_{(i)}]^{\beta}} \qquad eq(9)$$

Where, $p_{(i,j)}$ is the probability of assigning the unassigned VM j to server i. $\tau(i,j)$ is the heuristic information that represents the desirability of server pj for assigning VMi related to minimum bandwidth resource and energy consumption. $\eta(i,j)$ is the pheromone on the trail between server pi and VMj. α and β are parameters that control the relative importance of the pheromone and heuristic information, respectively. According to preliminary experiments, the value of both α and β we used for the experiment is, α =1 and β=2.

Heuristic information

The heuristic information indicates the desirability of assigning VM v to PM p.

$$\tau_{(i,j)} = \frac{1}{Bei^{csm} + p(pj)} \qquad eq(10)$$

As the ants build their paths and update the pheromone trail, they incorporate information about the bandwidth resource utilization and energy usage of the selected placements.

$$\eta(i,j) = (1-\varepsilon).\eta(i,j) + \varepsilon.\Delta t \qquad eq(11)$$

Where ε is the pheromone volatilization factor, which indicates the degree of pheromone evaporation per unit of time. The value is given in the range of (0,1]. $\Delta t(i,j)$ is the amount of pheromone deposited on the VMi-Pj pair by the ant that finds the solution currently. It is calculated using the following formula of equation (4.16):

The amount of pheromone deposited on the VMi-Pj pair by the ant is represented as follows:

$$\Delta t = \frac{1}{[C_{pj}^{cpu} - R_{vi}^{cpu}] + [C_{pj}^{mem} - R_{vi}^{mem}] + [B_e^{max} - B_{ei}^{Csm}]} \qquad eq(12)$$

Where, $[C_{pj}^{cpu} - R_{vi}^{cpu}] + [C_{pj}^{mem} - R_{vi}^{mem}] + [B_e^{max} - B_{ei}^{Csm}]$ are the corresponding normalized remaining CPU and RAM and Bandwidth on server j (the ratio of remaining resources to the resource capacity).

Once the ants have constructed their paths and updated the pheromone trail, the crossover operator is applied. This operator combines the paths of two ants to create a new path, allowing exploration of different placement possibilities. The selection of paths for crossover and mutation is influenced by the pheromone trail values associated with each path.

Paths with higher pheromone trail values, indicating better utilization of bandwidth resources and energy efficiency, have a higher probability of being chosen for crossover and mutation. The crossover operation can be implemented by exchanging the physical machines assigned to VM that are in the same ant's path. This exchange allows for the exploration of alternative physical machine assignments while considering the constraints of the problem.

 doi:10.20944/preprints202402.0170.v1

The specific implementation algorithm steps for the Bi-objective hybrid genetic optimization algorithm are as follows:

Input: Pop (population size), max-generation, VM (the set of virtual machines), PM(the set of physical machines).

Output: Pareto set of non-dominated solutions.

Begin

1. Obtain the set of sorted virtual machines in descending order of their traffic demand, PM
2. Initialize the Population with the Pop individual.
3. Evaluate the fitness of each individual in the population according to Eq (3) – Eq (7).
4. Create a Pareto set from the population regarding bi-objective to generate non-dominated solutions using Eq (1) & Eq (2).
5. While the termination condition is not true do the following.
6. Generate new population pop.
7. Apply a selection operator to choose individuals for the next generation. Individuals with higher fitness values have a greater chance of being selected.
8. Transform selected individuals into ACO using eq (8).
9. Initialize pheromone for each VM-PM pair
10. Calculate the desirability of $\tau(i,j)$ according to Eq. (10)
11. Select the physical machine for the virtual machine using Eq (11)
12. Update the pheromone of each path using Eq(11)
13. Select the best ant
14. Apply crossover operator.
15. Apply a mutation operator to randomly modify some individuals in the population.
16. Evaluate the fitness of each individual in the population according to Eq. (3)–Eq (7).
17. Apply a feasibility repair algorithm
18. apply a local optimization algorithm
19. Return non-dominated solution

End

Algorithm of Crossover operator

 doi:10.20944/preprints202402.0170.v1

Input: Parent Solution PM1 = [a1 a2...an], PM2 = [b1 b2...bn] with n as the number of hosts. Objective functions: obj_bandwidth, obj_energy.

Output: New Solution NS = [c1 c2...cn] as the new solution where n is the number of hosts.

1. fit1_BW &En = Fitness(PM1) based on objective function obj_bandwidth,& obj_energy
2. fit2_ BW &En= Fitness(PM2) based on objective function obj_bandwidth, obj_energy
3. for j = 1 to n do
4. randomly generate a real value between 0 and 1, r, so that, $0 \leq 1 \leq r$.
5. Compute the total fitness value for bandwidth and energy.

   Total_fit = fit1_BW & En + fit2_BW & En.
6. if r < fit1_BW &En / fit1_BW &En + fit2_ BW &En)
7. Set NS[j] = PM1[j]
8. End if.
9. Else
10. - Set NS[j] = PM2[j]
11. End else.
12. End for.

Return the new solution: NS

Where, PM1 & PM2 represents set of host1 and host1 in population respectively.

Algorithm of Mutation operator.

Input: already {VMs mapped to PMs}. m: number of PMs, n: number of VMs as chromosome Chr

Output: {VMs mapped to PMs} as new chromosome NewChr

Begin

Put NewChr = Already {VMs mapped to PMs}.;

// Select 2 host machines i and j such that

Draw two random numbers i and j, where i is a pi and pj is a PM such that $Pi.CPUUtil <$ C_{pi}^{cpu} and $Pj.CPUUtil <$ C_{pj}^{cpu} and $pi^{bcsm} < pj^{bcsm}$;

//From host I and j select one of the best as b (s for select) and other as remaining (r for reject) such that

Pb .avg$CPUUtilization$ < threshold && Pb .$CPUUtilization > Pr$.$CPUtilization$

 doi:10.20944/preprints202402.0170.v1

Assign VMs from Pr to Pb in NewChr.

Return {VMs mapped to PMs} as NewChr.

End

Algorithm of Crossover operator

Input: Parent Solution PM1 = [a1 a2...an], PM2 = [b1 b2...bn] with n as the number of hosts. Objective functions: obj_bandwidth, obj_energy.

Output: New Solution NS = [c1 c2...cn] as the new solution where n is the number of hosts.

1. fit1_BW &En = Fitness(PM1) based on objective function obj_bandwidth,& obj_energy
2. fit2_ BW &En= Fitness(PM2) based on objective function obj_bandwidth, obj_energy
3. for j = 1 to n do
4. randomly generate a real value between 0 and 1, r, so that, $0 \leq 1 \leq r$.
5. Compute the total fitness value for bandwidth and energy.

   Total_fit = fit1_BW & En + fit2_BW & En.
6. if r < fit1_BW &En / fit1_BW &En + fit2_ BW &En)
7. Set NS[j] = PM1[j]
8. End if.
9. Else
10. - Set NS[j] = PM2[j]
11. End else.
12. End for.

Return the new solution: NS

Where, PM1 & PM2 represents set of host1 and host1 in population respectively.

Algorithm of Mutation operator.

Input: already {VMs mapped to PMs}. m: number of PMs, n: number of VMs as chromosome Chr

Output: {VMs mapped to PMs} as new chromosome NewChr

Begin

Put NewChr = Already {VMs mapped to PMs}.;

// Select 2 host machines i and j such that

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 5 February 2024 doi:10.20944/preprints202402.0170.v1

Draw two random numbers i and j, where i is a pi and pj is a PM such that $Pi.CPUUtil <$ C_{pi}^{cpu} and $Pj.CPUUtil <$ C_{pj}^{cpu} and $pi^{bcsm} < pj^{bcsm}$;

//From host I and j select one of the best as b (s for select) and other as remaining (r for reject) such that

Pb .avg$CPUUtilization$ < threshold && Pb .$CPUUtilization > Pr$.$CPUtilization$

Assign VMs from Pr to Pb in NewChr.

Return {VMs mapped to PMs} as NewChr.

End

Pseudo-Code of infeasible solution repairing algorithm

```
1. for j = 1 to N do // where N is the number of PMs
2.     if V[j].violation = 1 then
3.         select = V[j].pointer
4.         V[j].pointer = V[j].pointer.next
5.         fixed = false
6.         j' = j + 1
7.         avail^CPU = Cpj'^cpu - R_vi^cpu of j'
8.         avail^mem = Cpj'mem - Rvimem of j'
9.         avil^Bw = B^max_ei - B^csm_ei of e connecting j and j'
10.        while j' ≠ j do
11.            if avail^CPU ≥ select.v[j] and avail^mem ≥ select.v[j] and avil^Bw ≥ select.v[j].trafficdem
                   then
12.                select.next = V[j'].pointer
13.                V[j'].pointer = select // update the selected VM of j' PM
14.                Cpj'^cpu += select.v[j]
15.                Cpj'^mem += select.v[j]
16.                B^max_ei += select.v[j].traffic-dem
17.                fixed = true
18.            end if
19.            j' = j' + 1
```

 doi:10.20944/preprints202402.0170.v1

20. avail^{CPU} = Cpj'^{cpu} - $R_{vi}{}^{cpu}$ of j'

21. avail^{mem} = Cpj'^{mem} - $R_{vi}{}^{mem}$ of j'

22. avil^{Bw} = B^{max}_{ei} - B^{csm}_{ei} of e connecting j and j'

23. end while

24. end if

25. end for

The local optimization Algorithm

The local optimization technique employs a heuristic algorithm to decrease the number of PMs in the VM placement, thereby lowering the overall energy used by the PMs.

It uses a heuristic algorithm to minimize the number of active servers; after that, it looks for a physical machine with a capacity greater than or equal to all of the other co-hosted virtual machines on the current physical machines. If it finds one, it offloads all of the co-hosted virtual machines from the server (PMi) to the server (PMj); after that, it changes PMi's status to hibernate mode. By doing this, it is possible to decrease the amount of bandwidth resources consumed as well as power consumption.

Pseudo-Code of the local optimization algorithm.

Input: Pop as a population with PopSize individuals, n: number of VMs, m: number of PMs

Output: The optimal population with the minimum number of used PMs

1. for i = 1 to PopSize do
2. for j = 1 to the number of PM do
3. if PM[i][j].Status = Active then
4. j' = j + 1;
5. while (j' <= m) do
6. if PM[i][j'].Status = Active then
7. if PM[i][j'].Cpj'^{cpu} - PM[i][j'].Upj'^{cpu} >= PM[i][j].Upj^{cpu} and
8. PM[i][j'].Cpj'^{mem} - PM[i][j'].Upj'^{mem} >= PM[i][j].Upj^{mem} and
9. B^{csm}_{ei} + PM[i][j'].Upj'^{bw} - PM[i][j].Upj^{bw} <= B^{max}_{ei} then // migrate vms
10. PM[i][j'].Cpj'^{cpu} = PM[i][j'].Upj'^{cpu} + PM[i][j].Upj^{cpu};
11. PM[i][j'].Cpj'^{mem} = PM[i][j'].Upj'^{mem} + PM[i][j].Upj^{mem};
12. PM[i][j'].B^{max}_{ei} = PM[i][j'].Upj'bw + PM[i][j].B^{csm}_{ei};
/* Remove(drop) VMj from PMj and hibernate it */
13. PM[i][j].Status = "Off hibernate";

 doi:10.20944/preprints202402.0170.v1


```
                    end if
                    j' = j' + 1;
                end while
            end if
        end for
    end for
```

Algorithm of Metaheuristic search for Pareto set Ranking

Input: pareto size, Set of Hosts and VMs with Initial Solution.

Output: Set of Pareto non-dominated solutions

Begin.

1. For each new solution:
2. If a new solution is not dominated by any solution in the Pareto set
3. Add a new solution to the Pareto set
4. End if
5. For each solution in the Pareto set.
6. If the solution is dominated by a new solution:
7. Delete the solution from the Pareto set,
8. End if
9. End for
10. if pareto set size exceeds pareto size
11. apply ranking of solution in increasing order based on bandwidth & energy
12. Select the top Pareto set solutions with the lowest values of bandwidth resource and energy consumption.
13. End for
14. Output: Set of Pareto non-dominated solutions

End

Pseudo-code of a greedy heuristic algorithm that orders VMs in descending order based on their traffic demands

Here is a possible pseudo-code of a heuristic algorithm that orders VMs in descending order based on their traffic demand and stores them in a list called vmsList.

 doi:10.20944/preprints202402.0170.v1

Input: a set of virtual machines with their traffic demands

Output: a list of virtual machines ordered by their traffic demand

```
1.  vmList = {};
2.  Ttemp = Ttraffic
3.  i = 1
4.  Ttemp = sort(Ttemp, by traffic_demand, order descending)
5.  while i <= size(Ttraffic);
6.        (vmi, vmj) = Ttraffic[k]
7.         if vmi not in vmList:
8.         add vmi to vmList
9.         if vmj not in vmList:
10.        add vmj to vmLis;
11.        i = i + 1
12.    end of while
13. output vm as vmList for virtual machine placement
14. end
```

3. Experimental Results and discussion.

In this chapter, I discussed the algorithm parameter setup and the outcome evaluation metrics, which are bandwidth resource and energy consumption reduction. With a bi-objective hybrid genetic optimization algorithm, I was compared to other algorithms, including the Genetic Algorithm (GA), Ant Colony Optimization (ACO), and First Fit Decreasing (FFD) algorithm.

To evaluate our proposed BOHGOA, we compared it with GA, ACO, and, FFD in terms of total bandwidth resource consumption and total energy consumption. For the simulation of the cloud environment, we used 10 different test cases that were run with varying numbers of VMs and different workload conditions. We have chosen the policy of minimum migration time (MMT) for VM selection with the value 1.2 as a safety parameter.

Table 1. Configuration Setup of CloudSim Tools and test problems.

Scheme 3.		CloudSim
Version		3.0.3
Datacenter Characteristics	PM-type-1	MIPS: 40000, Storage:1TB, RAM: 32GB, BW: 1 Gbit. Power busy pj is set to 550 W OS: Linux. System Architecture: x86. VMM: Xen
	PM-type-2	MIPS: 24000, Storage:1TB, RAM: 16GB, BW: 1 Gbit. Power busy pj is set to 113 W. OS: Linux.System Architecture: x86. VMM: KVM
Test problems	Number of VM40 – 500	
	Number of PM 20-80	

 doi:10.20944/preprints202402.0170.v1

Bandwidth Resource Consumption Evaluation of BOHGOA

In this section, we evaluated the proposed algorithm's bandwidth resource consumption compared to existing algorithms. We have compared our proposed algorithm's efficiency and performance by measuring its bandwidth resource usage and energy consumption during

VMP in cloud computing with each of the existing algorithm.

As shown in Figure 1. above, BOHGOA decreased Bandwidth resource consumption by 54.14%, 32.11%, and 57.47% when compared to GA, ACO, and FFD for 240 VMs placement respectively. Also, as depicted in Figure 2, it reduced by 38.12%, 22.76%, and 47.05% when compared to the GA, ACO, and FFD for 500 VMs placement respectively. These reductions exhibit BOHGOA's efficiency to minimize bandwidth resource usage more successfully than the other methods. Therefore, BOHGOA reduces link congestion and improves overall network performance by efficiently deploying virtual machines.

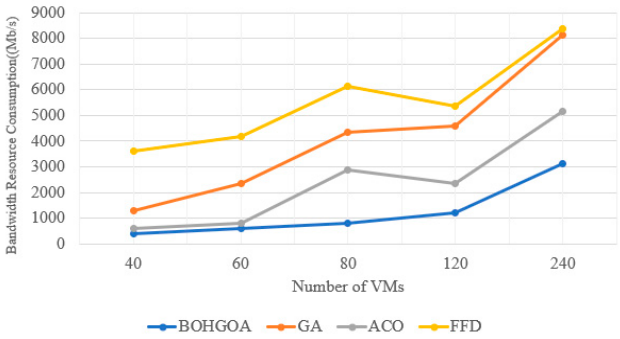

Figure 1. Bandwidth resource consumption of PM of 240 VMs Placement.

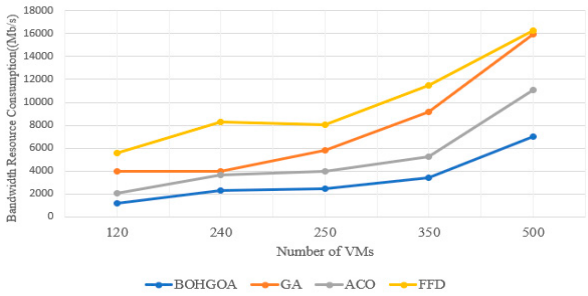

Figure 2. Bandwidth resource consumption of PM of 500 VMs Placement.

Energy Consumption Evaluation of BOHGOA

In this section, we focused on evaluating the efficiency of BOHGOA in reducing physical machine energy consumption. The primary goal of the investigation was to evaluate how effectively BOHGOA manages and optimizes physical machine energy usage in comparison with the other algorithms. Energy consumption is an important consideration in data centers because it has a direct influence on environmental sustainability. BOHGOA outperformed the three other algorithms.

Here, when compared to GA, ACO, and FFD algorithms with BOHGOA. BOHGOA achieved an overall reduction in energy usage of physical machines for the placement of 240 VMs by 37.71%, 34.01%, and 40.14%, respectively as depicted in Figure 3 above.

 doi:10.20944/preprints202402.0170.v1

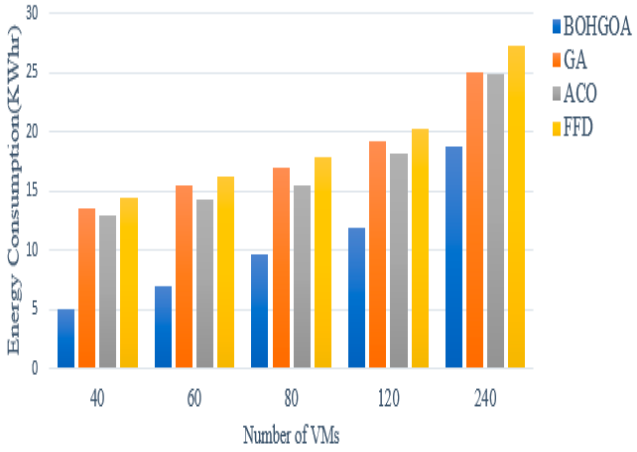

Figure 3. Energy consumption of PM of 240 VMs Placement.

This means that BOHGOA consistently outperforms these algorithms, resulting in notable energy savings. Similarly, as seen in Figure 4 above, for the placement of 500 VMs, BOHGOA achieved a reduction in energy usage of physical machines by 27.50 %, 22.28%, and 30.52% when compared to GA, ACO, and FFD algorithms respectively. Therefore, these findings emphasize the significant advantages of BOHGOA in optimizing energy usage and improving overall efficiency in virtual machine placement. This energy-saving feature is critical for lowering operational costs and decreasing the environmental impact of data centers.

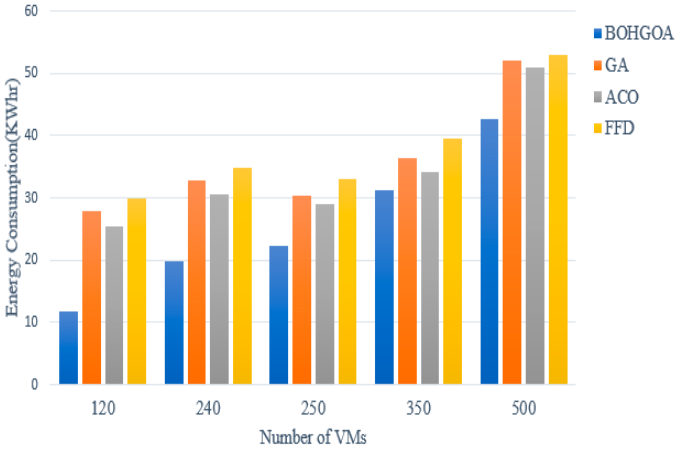

Figure 4. Energy consumption of PM of 500 VMs Placement.

Complexity analysis of BOHGOA

To simplify the analysis of BOHGOA further we used the term time complexity. Time complexity is an abstract way to represent the running time of algorithm in terms of rate of growth and Big-O notations only. It is an approximate estimation of how much time an algorithm will take for large value of input size(Arora & Barak, 2006).

Here we discussed the time complexity of our proposed algorithm. The proposed BOHGOA approach is the execution of two meta-heuristics algorithms: GA and ACO. Therefore, the total time complexity can be determined by computing the complexity of the two methods given. So, the complexity of each meta-heuristic is determined by adding up the complexity of all the procedures.

 doi:10.20944/preprints202402.0170.v1

In this regard, having n VMs, m PMs, we considered the parameters defined in section Table 4. As shown in Tables 5 and 6 above, we have set the number of solutions namely, the population size, and number of ants in both meta-heuristics during the initializing phase as fixed numbers.

Table 5. Time complexity.

Procedure	Complexity
Population-size	O (Population-size* m * n^2)
Idle servers	O (Population-size* m * n)
Roulette wheel	O (population-size * Crossover rate)
Crossover rate	O (population-size * crossover rate)
Mutation rate	O (population-size *mutation rate)

Complexity for GA: O (maximum iteration * population-size * Crossover rate * m2 * n^2).

Table 6. Time complexity.

Procedure	Complexity
Solution	O (number of ants* m *n^2)
Update	O (number of ants *m*n)
Idle servers	O (number of ants *m*n)

The Overall complexity for ACO: O (maximum iteration * number of ants * m^2 * n^2).

The Overall Complexity of BOHGOA.

O (maximum iteration * population-size * Crossover rate * m2 * n^2) + O (maximum iteration * number of ants * m^2 * n^2) =O [(Population-size* Crossover rate + number of ants) *(maximum iteration * m^2 * n^2)] = O (maximum iteration * m^2 * n^2). Therefore, the overall complexity of the BOHGOA algorithm can be simplified to O (maximum iteration * m^2 * n^2) This means that the complexity is mainly dependent on the maximum number of iterations and the sizes of the server set (m) and the VM set (n). These factors have a more significant impact on the overall complexity compared to the population size and number of ants.

4. Conclusion and Future Work

Cloud computing has revolutionized IT practices, leading to increased demand for cloud data centers, which consume substantial energy and bandwidth. Efficient virtual machine placement (VMP) in cloud environments is challenging due to its NP-hard classification, necessitating the use of evolutionary algorithms.

This thesis has focused on bi-objective optimization of bandwidth resources and energy consumption for VMP in cloud computing. Addressing existing shortcomings, our proposed algorithm, BOHGOA, integrates genetic algorithms with ant colony optimization for optimal VM placement, minimizing both bandwidth resource utilization and energy consumption.

BOHGOA offers a range of solutions along the Pareto front, representing trade-offs between bandwidth and energy. Experimental results demonstrate its superiority over GA, ACO, and FFD in terms of both bandwidth and physical machine energy consumption.

However, several research challenges remain unaddressed:

Security Consideration: Future work should include security aspects during VM placement to protect sensitive data and prevent unauthorized access.

Dynamic SLA Management: Algorithms should adapt to fluctuating SLAs while minimizing infrastructure costs by efficiently utilizing physical servers.

Fault Tolerance: Considering fault tolerance during VM placement is crucial to minimize the impact of faults and ensure the reliability of cloud data centers.

Addressing these challenges will contribute to more robust and adaptable solutions for optimizing VM placement in cloud environments.

Finding: This work was supported by the Adama Science and Technology university grant number (ASTU/SM-R/ 828/23).

Data Availability Statement: All data generated or analyzed during this study were included in this article.

Acknowledgments: Ketema Deresa is a Master's student in the Department of Computer Science and Engineering at Adama Science and Technology University in Oromia, Ethiopia. The research described in this paper was conducted under the guidance and supervision of Dr.-Ing Frezewd Lemma. Dr. Lemma served as the advisor for this research project, providing valuable insights and expertise in the field of cloud computing.

Conflict of Interest: The Authors declare that they have no conflicts of interest to disclose regarding this article publication.

Reference

1. Abohamama, A. S., & Hamouda, E. (2020). A hybrid energy–Aware virtual machine placement algorithm for cloud environments. Expert Systems with Applications, 150, 113306. https://doi.org/10.1016/j.eswa.2020.113306
2. Arivuselvi, A., Amudha, T., & Babu, P. D. (2021). An Effective Ant Colony Optimization Methodology For Virtual Machine Placement In Cloud Data Centre. Turkish Journal of Computer and Mathematics Education, 12(10), 3460–3467.
3. Arnold, T., He, J., Jiang, W., Calder, M., Cunha, I., Giotsas, V., & Katz-Bassett, E. (2020). Cloud Provider Connectivity in the Flat Internet. Proceedings of the ACM SIGCOMM Internet Measurement Conference, IMC, 230–246. https://doi.org/10.1145/3419394.3423613
4. Balaji, K., Kiran, · P Sai, & Sunil Kumar, · M. (2022). Power aware virtual machine placement in IaaS cloud using discrete firefly algorithm. Applied Nanoscience, 1, 3. https://doi.org/10.1007/s13204-021-02337-x
5. Ganesan, S., & Ganesan, S. (2021). A multi-objective secure optimal vm placement in energy-efficient server of cloud computing. Intelligent Automation and Soft Computing, 30(2), 387–401. https://doi.org/10.32604/iasc.2021.019024
6. Hariharan, B., Siva, R., Kaliraj, S., & Prakash, P. N. S. (2021). ABSO: an energy-efficient multi-objective VM consolidation using adaptive beetle swarm optimization on cloud environment. Journal of Ambient Intelligence and Humanized Computing, 0123456789. https://doi.org/10.1007/s12652-021-03429-w
7. Hashem, I. A. T., Yaqoob, I., Anuar, N. B., Mokhtar, S., Gani, A., & Ullah Khan, S. (2015). The rise of "big data" on cloud computing: Review and open research issues. Information Systems, 47, 98–115. https://doi.org/10.1016/J.IS.2014.07.006
8. Li, P., & Cao, J. (2022). A Virtual Machine Consolidation Algorithm Based on Dynamic Load Mean and Multi-Objective Optimization in Cloud Computing. Sensors 2022, Vol. 22, Page 9154, 22(23), 9154. https://doi.org/10.3390/S22239154
9. Mellette, W. M., Snoeren, A. C., & Porter, G. (2016). P-FatTree: A multi-channel datacenter network topology. HotNets 2016 - Proceedings of the 15th ACM Workshop on Hot Topics in Networks, 78–84. https://doi.org/10.1145/3005745.3005746
10. Niranjan, U. V., & Pillai, K. K. G. (2019). Energy Management of Cloud Data Center Using Neural Networks. Proceedings - 7th IEEE International Conference on Cloud Computing in Emerging Markets, CCEM 2018, 85–89. https://doi.org/10.1109/CCEM.2018.00022
11. Rawas, S., & Zekri, A. (2018). Location-Aware Energy-Efficient Workload Allocation in Geo Distributed Cloud Environment. Journal of Computer Science, 14(3), 334–350. https://doi.org/10.3844/JCSSP.2018.334.350
12. Regaieg, R., Koubàa, M., Ales, Z., Aguili, T., & Aguili, · Taoufik. (2021). Multi-objective optimization for VM placement in homogeneous and heterogeneous cloud service provider data centers. Computing, 103(6), 1255–1279.
13. Saxena, D., & Singh, A. K. (2021). A proactive autoscaling and energy-efficient VM allocation framework using online multi-resource neural network for cloud data center. Neurocomputing, 426(xxxx), 248–264. https://doi.org/10.1016/j.neucom.2020.08.076
14. Sloss, A. N., & Gustafson, S. (2019). 2019 Evolutionary Algorithms Review. 81 Su, S., Su, Y., Shao, F., & Guo, H. (2016). A Power-Aware Virtual Machine Mapper Using Firefly Optimization. Proceedings - 2015 3rd International Conference on Advanced Cloud and Big Data, CBD 2015, 96–103. https://doi.org/10.1109/CBD.2015.25
15. Wang, H. (2022). Research on the Application of Genetic Algorithm in Physical Education. Journal of Mathematics, 2022. https://doi.org/10.1155/2022/8477945

16. Yu, L., Shen, H., Cai, Z., Liu, L., & Pu, C. (2018). Towards Bandwidth Guarantee for Virtual Clusters under Demand Uncertainty in Multi-Tenant Clouds. IEEE Transactions on Parallel and Distributed Systems, 29(2), 450–465. https://doi.org/10.1109/TPDS.2017.2754366
17. Zhang, W., Chen, X., & Jiang, J. (2021). A multi-objective optimization method of initial virtual machine fault-tolerant placement for star topological data centers of cloud systems. Tsinghua Science and Technology, 26(1), 95–111. https://doi.org/10.26599/TST.2019.9010044