Chapter 5

A meta-heuristic-based virtual machine consolidation algorithm

This chapter[1] *introduces a new algorithm for VM consolidation in cloud data centers. It uses live migrations of VMs during the execution of cloudlets so that underloaded physical servers can be switched off by transferring those VMs to other physical machines. The VM consolidation problem in cloud data centers is NP-hard. Hence a meta-heuristic approach called Water Wave Optimization (WWO) is used to implement VM consolidation. The proposed approach produces a near-optimal solution using an objective function that minimizes energy utilization.*

5.1 Introduction

Cloud computing depends on shared computing resources instead of having on-premise servers to deal with user applications. In other words, cloud computing is taking services and moving them outside an organization's firewall. Virtualization technology is one of the key fundamental technologies to enable cloud computing. It improves cloud resource utilization and sharing. It was extensively used mainframe systems to improve

[1]Medara, Rambabau, et al. "Energy Efficient Virtual Machine Consolidation Using Water Wave Optimization." *2020 IEEE Congress on Evolutionary Computation (CEC).* IEEE, 2020.

manageability, reliability, and resource utilization [117]. The capability of virtualization is broadly used in workload consolidation, workload isolation, and workload migration.

Virtualization technology does a great deal of cost-saving, energy-saving, hardware performance enhancement for cloud providers. It allows the cloud server to be shared among multiple applications of various users at the same time [118]. Virtualization is achieved by assigning a logical name to physical resources and providing a pointer to that resource on a need basis. Virtualization not only provides a way to execute multiple shared applications but also helps in sharing hardware resources of processor, memory, and network bandwidth. Virtualization in cloud computing is done by a hypervisor or Virtual Machine Monitor (VMM) which creates an abstraction layer between the software and the hardware in use. In this system, a server can be virtualized into multiple instances at a time, but every instance is logically isolated from each other for security reasons.

The various virtual machines (VM) running on different physical machines of a datacenter need to get processor time, memory, and bandwidth for the completion of tasks assigned to it. The VMM or hypervisor is responsible for the allocation of VMs on the PMs and this is done by the various VM allocation algorithms. This chapter concern with the energy-efficient allocation techniques and those that perform dynamic migrations of VMs. The process in which dynamic migration of the VMs happens during runtime to reduce a load of overloaded physical machines (PM) and to put to sleep the underloaded PMs is called VM Consolidation. Switching off the underloaded hosts will help us reduce the energy consumption of the data centers and this can be done by migrating the VMs from those machines to other machines [119]. This chapter presents an energy-aware VM consolidation technique that uses a new meta-heuristic called water wave optimization (WWO). The original WWO algorithm was proposed for the continuous space multidimensional optimization problem. Later, it was adapted to solve the famous discrete optimization problem of the Travelling Salesman Problem (TSP) [120] by modifying its original parameters. This work motivated us to try the WWO by modifying the parameters and apply to our multidimensional discrete optimization problem of energy-efficient VM Consolidation.

The rest of this chapter is organized as follows. We introduce the related work in Section 5.2. The Cloud Data Center (CDC) model is discussed in Section 5.3. The overview of the water wave theory is introduced in Section 5.4. The proposed WWO-based VM Consolidation

algorithm is presented in Section 5.5. Section 5.6 presents the experimental setup and discussion on results. Finally, Section 5.7 concludes our work and discussed the future directions.

5.2 Related work

There have been many works done to minimize the energy consumption of data centers by maximizing resource utilization. By applying the Ant Colony Optimization (ACO) algorithm, an approach [121] resolves the Multi-dimensional Bin Packing (MDBP) problem by effectively consolidating the VMs in the PMs through workload placement in a cloud environment. Despite this, the energy-aware VM placement scheduling approach [122] measures the fitness value between VMs using an ACO algorithm to identify the past optimal placement in the corresponding PM rather than measuring the fitness between the VMs and PMs. It tends to provide the solution with a minimum number of PMs, which helps to reduce the overall energy consumption. ACO metaheuristic-based scheduling method [123] consolidates the VMs by incorporating the vector algebra, which minimizes the energy consumption and reduces resource wastage in the cloud environment. A genetic algorithm also has been used by the existing researchers to develop the energy-efficient cloud datacenter through VM consolidation [124].

To reduce the energy consumption through minimizing the time, memory, and cost consumption, the task-based load balancing approach [125] employs the Particle Swarm Optimization (PSO) algorithm to migrate only several tasks to the identical VM resource rather than migrating the entire overloaded VM. A modified PSO algorithm [126] ensures the energy-efficient VM placement in the cloud data center by optimizing the operators and parameters of the PSO algorithm. In order to minimize the energy utilization in the workflow scheduling, the recent research works have focused on modeling the energy-aware scheduling algorithms through resource hibernation, dynamic power management, or Dynamic Voltage and Frequency Scaling (DVFS) techniques. The multi-objective Discrete PSO (MODPSO) approach [60] employs the hybrid PSO algorithm and DVFS technique to reduce the energy utilization by the cloud infrastructure, which averts the compromise between the performance and energy consumption by handling multiple QoS requirements. Greedy-based heuristics proposed in [127] set upper

and lower utilization threshold for Central Processing Unit (CPU) utilization. If exceed the upper threshold or utilization drops below the lower threshold then the VM migration plan enforced.

Zheng Yu-Jun [128] proposed a new optimization technique called Water Wave Optimization (WWO) which is a metaheuristic inspired by shallow water wave models. WWO approach is competitive with few state-of-the-art evolutionary algorithms such as ACO, BA, BBO, IWO, etc., and effective for real-time applications [129]. The proposed algorithm in this chapter adapts the idea of WWO algorithm to effectively place virtual machines on a selected physical machine in an energy-efficient way while satisfying Quality of Service (QoS) requirements.

5.3 System Model

A heterogeneous Cloud Data Center (CDC) contains m physical machines (PMs). Each PM is characterized by computing resources such as CPU, memory, storage capacity, and network I/O. The performance of the CPU is defined in terms of Millions of Instructions Per Second (MIPS). The PMs are virtualized to serve many users at any given time. Users present their request for provisioning of v virtual machines of PM. The length of the user request is specified in Millions of Instructions (MI). We used the greedy-based approach called Best Fit (BF) for the initial allocation of VMs to PM. The BF algorithm is a well-known heuristic for the bin-packing problem [130].

The VM resource utilization changes with time due to dynamic workloads. Hence, the initial provision of VMs to PM needs to be enhanced with an efficient VM consolidation approach. Our proposed WWO-VMC algorithm is applied periodically to optimize the VM placement depending on the workload. The BF algorithm is used to optimize the resource utilization locally, whereas our proposed WWO-VMC approach is used to optimize resource utilization globally. Based on the percentage of CPU utilization we categorize the PM into one of the three categories PM_{under}, PM_{over} and PM_{normal}. We consider PM as underutilized PM_{under} if the CPU utilization is under 40%, PM as over-utilized PM_{over} if CPU utilization is over 90%, and all other PMs are normal utilized PM_{normal} .

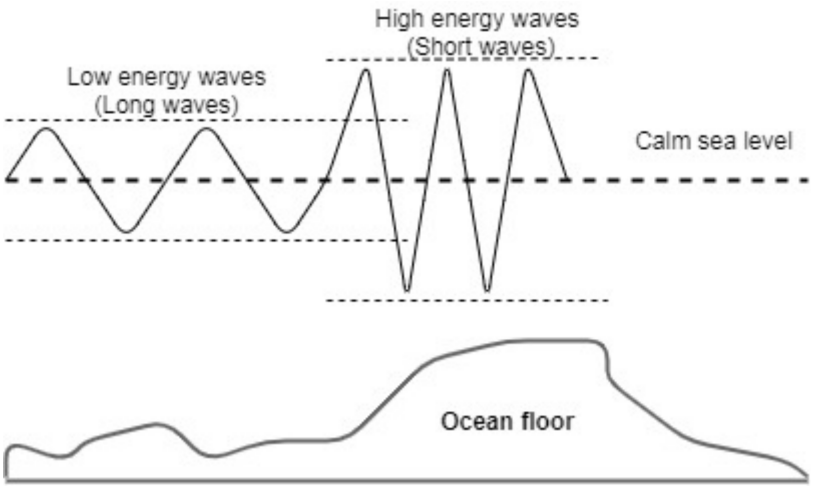

FIGURE 5.1: Shallow and deep water wave models

5.4 Theory of Water Wave

Isaac Newton was the first person to make an effort on the theory of water waves. Further, the linear theory of water waves was studied by French mathematicians Laplace, Lagrange, Poisson, and Cauchy and they made authentic theoretical advances [131]. The WWO approach proposed in [128] is an efficient technique for global optimization. The shallow water wave theory is the basis for WWO, which uses a numerical approach to analyze the evolution of wave amplitudes (heights), periods (wavelengths), and propagation directions under different conditions such as nonlinear wave interactions, wind force, and frictional dissipation [129]. The solution space in the WWO algorithm is comparable to the seabed region. The fitness of any point in the solution space is estimated inversely by the depth of the seabed. The fitness functions of waves measure high if the distance is less to the still water [128]. The various changes in the shape of water waves such as amplitude or height and wavelength are depicted in Figure 5.1.

The high energy waves having large amplitude i.e., good wave produce a high-quality solution, and low energy waves having long wavelength produce poor solutions. The majority of evolutionary algorithms keep up a population of solutions; similarly, WWO maintains a solution. Each solution of which is similar to a "wave" with two parameters height and wavelength. The height of wave $h \in Z^+$ (integer domain) and wavelength $\lambda \in R^+$ (real domain). These parameters are initialized as constants $h = h_{max}$ and $\lambda = 0.5$ [60]. The

best solution exploration procedure in the WWO algorithm is modeled as wave propagation, wave refraction, and wave breaking.

5.5 Energy Efficient VM Consolidation with WWO

As discussed in Section 5.3, each PM provides one or more VMs, and both PMs and VMs are categorized by their resource utilization. In the context of VM migration a PM can be either a potential source PM pm_{source} where VMs already residing or a target PM pm_{dest} for VM migration. All PM_{under} and PM_{over} are members of pm_{source} set, whereas all PMs except PM_{over} are members of pm_{dest} set. The proposed WWO-VMC algorithm creates set of tuples T, where each tuple t in T consists of three elements $t = (pm_{source}, vm, pm_{dest})$ where pm_{source} is the source host, vm is the selected VM for migration and pm_{dest} is destination host machine. Our proposed algorithm aims to minimize the number of active PMs needed to host all VMs without compromising their performance. This can be done by enforcing the VM consolidation algorithm.

In this work, we redesign the original WWO algorithm for VM Consolidation to minimize energy consumption. As every solution in WWO is a migration plan or set of migrations, hence, we considered every solution as a wave $\mathbf{x}$. A wave has a set of three-parameter tuple t, height h and wavelength λ (initially height is constant h_{max} and wavelength is 0.5). The fitness of $\mathbf{x}$ depends on the number of migrations (M) and the number of sleeping hosts or PM (Ps): As the number of migrations increase, fitness decreases, and as the number of sleeping hosts increases, fitness also increases.

We initialize the population as a set of migration plans as in the original WWO algorithm and apply the three operators wave propagation, refraction, and breaking. Which evolve the population up to the termination state is satisfied continually. However VM consolidation is a combinatorial problem, therefore WWO cannot be directly applied for energy-efficient VM Consolidation, so we have to redefine its operator's propagation, refraction, and breaking as discussed in the following sections.

5.5.1 Propagation for Energy Efficient VMC

In the propagation operator, we can see wavelength λ of solution as the probability for mutation, and we will decide whether to mutate or not using: for a good solution and small probability of mutation the value of λ will be small; on the other hands, a bad solution will have a large λ value, and hence will have a considerable probability of being mutated.

Each wave **x** generates another wave **x'** if its wavelength is greater than a randomly generated value r between 0 and 1, by this method –

If $r < \lambda$ then for each tuple t in wave x, we will randomly either add a new tuple or replace the current tuple (with a tuple from the set of all tuples excluding the tuples from current wave x) or we will remove the current tuple from current wave x; If the fitness of the current wave x is less than the fitness of the newly generated wave x' then replace x by x'(with $h = h_{max}$) else set $h = (h-1)$ for the current wave x. After each generation, x updates its wavelength using equation (5.1).

$$\lambda = \lambda \cdot \alpha^{-(f(\mathbf{x})-f_{min}+\varepsilon)/(f_{max}-f_{min}+\varepsilon)} \tag{5.1}$$

where, $f(x)$ is fitness function, $f(x) = P_s^{\gamma} + 1/(\varepsilon + M)$, P_s is the number of sleeping hosts(PM), M is the number of migration, γ is a parameter defining the relative importance of P_s, α is wavelength reduction coefficient, f_{min} and f_{max} are minimum and maximum fitness of the current solution (Migration plan). To avoid division by zero, we choose a small value for ε.

5.5.2 Refraction for Energy Efficient VMC

The purpose of Refraction in WWO is if we know that a solution is bad, then we try to improve the solution by giving it some good features of the known best solution. Technically If height h of some wave/solution **x** becomes zero, then **x** is replaced by new solution **x'** which will be centered between **x** and global best solution x^*.

In our case, our solution is the migration plan having some tuples showing each migration. So whenever the height of the solution vanishes, to make the current solution (migration

plan) better, we add or replace some tuple (migrations) with tuples from the global best solution so far i.e. x^*.

We set the height of the new solution to h_{max}, and update the wavelength :

$$\lambda' = \lambda \cdot \frac{f(\mathbf{x})}{f(\mathbf{x'})} \tag{5.2}$$

where $f(\mathbf{x})$ and $f(\mathbf{x}')$ is the fitness of old solution and new solution respectively.

5.5.3 Breaking for Energy Efficient VMC

If the propagated solution is better than the known global best solution so far, i.e., $f(\mathbf{x}) > f(x^*)$, then we check in the neighborhood of the solution **x** for a better solution than **x**, and if we find a solution better than **x** say **x'**, then the global best solution is updated as $x^* = \mathbf{x'}$ otherwise update $x^* = \mathbf{x}$.

We will find the neighborhood solution and update the global best solution using the following way: For k in (1,k_{max}) do: Choose some tuples from solution **x**, and for each of those chosen tuple either add or replace the tuple with a tuple from the set of all tuples excluding the tuples from current wave **x**, or remove the tuple from solution **x**. if $f(\mathbf{x}) > f(x^*)$ then update $x^* = \mathbf{x}$. where k_{max} is a predefined number whose value is 12.

5.5.4 Algorithm framework

In this section, we presented our proposed algorithm and discussed time complexity. The main focus of our algorithm is to maximize the inactive machines, in other words, minimize the active machines to conserve energy. The pseudo-code for the proposed WWO-VMC algorithm is given in Algorithm 5. It initializes the population randomly (line 1) and creates tuples with three elements each $t = (pm_{source}, vm, pm_{dest})$ where pm_{source} is the source host, vm is the selected VM for migration and pm_{dest} is destination host machine. The number of tuples keeps changing according to the algorithm whenever to add or remove the tuple step comes. The three operators used in this algorithm are propagation, refraction, and breaking. Propagation (lines 5-14) enables to accept tuple or create a new tuple based

on the fitness of the population. For better wave propagation we consider parameters r_1 and r_2. Based on the series of preliminary experiments we tuned r_1 and r_2 to 0.28 and 0.64 respectively. The breaking operator (lines 17-19) enables an exhaustive search for selecting a source for migration. Finally, the refraction (lines 21-24) avoids early convergence by improving population diversity.

Algorithm 5 WWO-VMC algorithm

```
1  Randomly initialize a population P of migration plans;
2  while iterations < max_number_of_iterations  do
3    for each wave x in population P do
4      initialize x' = x
5      for each tuple t in x do
6        r = rand(0,1)
7       if rand() < x.λ then
8         if r < r1 then
9           add new tuple in x'
10        else if r < r2 then
11          replace t by new tuple in x'
12        else
13          remove t from x'
14      end for
15   end for
16   if f(x') > f(x) then
17     if f(x') > f(x*) then
18       perform breaking operation (Section 5.5.3)
19     update x with x'
20   else
21     x.h = x.h − 1
22     if x.h == 0 then
23       perform refraction operation (Section 5.5.2)
24       update wavelength based on equation (5.1)
25 end while
26 return x*.
```

Our proposed WWO-VMC algorithm framework is clear and easy to understand. The algorithm performs well with fewer populations. We consider the recommended values of control parameters for the algorithm such as the maximum wave height h_{max}, the number of breaking directions k, wavelength reduction coefficient α as in [128]. The WWO-VMC algorithm minimizes the number of active PMs to reduce the energy consumption of data centers while preserving the QoS requirements.

Let n be the number of waves in the population, m be the number of physical machines (PM), v be the number of Virtual Machines (VM). Our algorithms have three types of operation in a single iteration over the population. Propagation: The time complexity of propagation is linear in the number of tuples in n waves of the population. The worst case of the number of the tuples in a wave is $O(m^2 * v)$ which implies the worst-case time complexity of this operation is $O(n * m^2 * v)$. This is the costliest operation in a single iteration. Breaking: Let k be the coefficient of breaking. The time complexity of breaking is $O(k * m^2 * v)$. Refraction: It has the same worst-case complexity as that of propagation operation. Let I be the number of iterations in the algorithm. So, the overall time complexity of the WWO-VM Consolidation algorithm is $O(I * n * m^2 * V)$.

5.6 Experimental setup and results

The technology stack used for the simulation of results is JAVA language and the framework used was CloudSim Plus. Moreover, our target is IaaS clouds which provide unlimited resources to cloud users on a payment basis. Conducting repeatable experiments on such infrastructure is expensive, so we used the simulation model. CloudSim Plus is a toolkit with a full-featured and flexible simulation framework. It enables users to model cloud scheduling applications for simulation, and experimentation. Users are allowed to focus on specific system modeling issues to be explored, without regarding the functional level characteristics related to data center infrastructure and Services. CloudSim Plus framework has all the basic classes required for the simulation of various processes of the cloud. It allows one to extend the basic classes to implement the modified algorithms for the specific simulation. We have extended the basic VM allocation with migration to implement our WWO based VM allocation with migration. For the simulation of our algorithm, we have used the open data, provided by Numerical Aerodynamic Simulation (NAS) Systems Division at NASA Ames Research Center. The workload was logged for three months from October 1993 in a 128-node iPSC/860 hypercube [132].

We used the cleaned and converted log in Standard Workload format. SwfWorkloadFileReader class in Cloudsim Plus was used to read and build cloudlets using this file. Each cloudlet was assigned a suitable VM, and correspondingly hosts were created according to the requirements of VMs at the initial simulation. For our hosts, we

TABLE 5.1: Parameters in the WWO-VMC approach.

γ	ε	r	α	λ	k
5	0.00001	(0, 1)	(1.001, 1.01)	0.5	(1, 12)

TABLE 5.2: Amount of energy consumption of HP G3 server at different load levels.

Load Levels(%)	**Energy Consumption (Watts)**
0	105
10	112
20	118
30	125
40	131
50	137
60	147
70	153
80	157
90	164
100	169

selected one HP ProLiant ML110 G3 server (with the configuration of (1 x [Pentium D930 3000 MHz, 2 cores], 4GB)). To evaluate the performance of our implemented algorithm, we considered two metrics energy consumption and the number of migrations. The WWO parameters and values which we were obtained in a series of preliminary experiments and that were used in the proposed WWO-VMC approach present in Table 5.1.

5.6.1 Simulation Results

The energy consumed in the whole energy utilized by the physical machines while running the VMs during the simulation. The objective function was designed in such a way that energy consumption is minimized by our algorithm. The energy consumed by a single physical machine depends on the utilization of resources of a machine like CPU, memory, and bandwidth. It has been noted that power consumption by the utilization of CPU exceeds the other factors by a margin. So most of the approaches to calculate energy are based on modeling of energy based on the utilization of CPU. The SPECpower benchmark is the real-world workload that we used to evaluate our algorithm on CloudSim Plus. The energy consumption of the HP G3 server at different workload levels illustrated in Table 5.2.

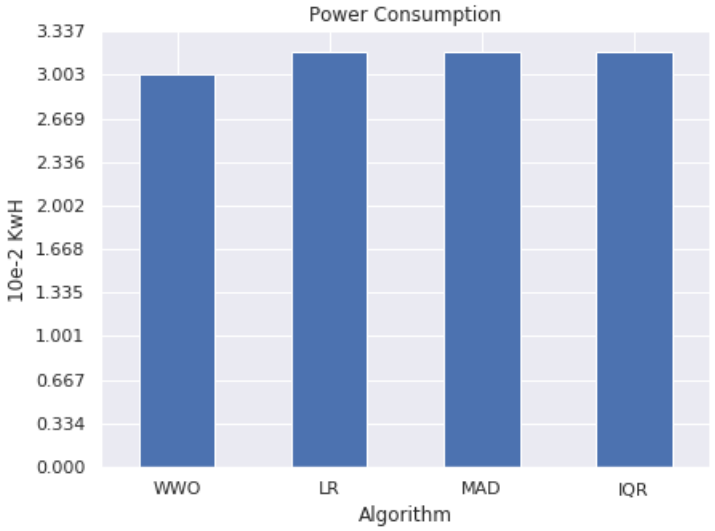

FIGURE 5.2: 15 Cloudlets

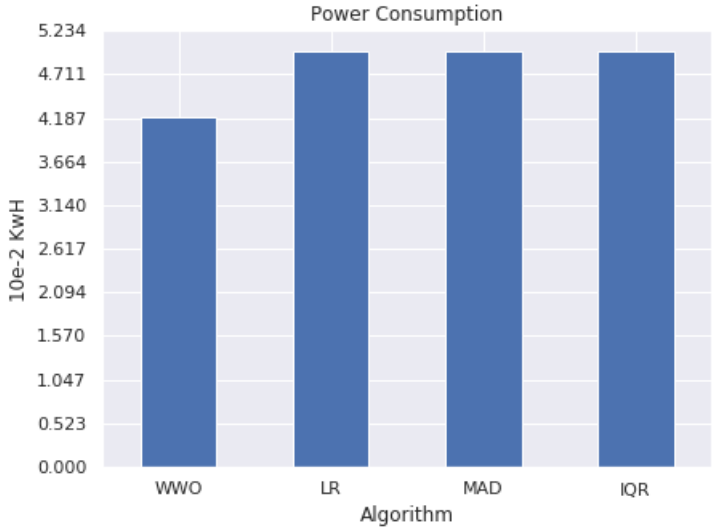

FIGURE 5.3: 20 Cloudlets

We have compared our WWO-VMC algorithm with three well-known heuristic methods for dynamic VM reallocation in [127]. These algorithms keep and use CPU utilization between upper and lower thresholds. When a PM is underutilized then its VMs are consolidated for load balancing and when the PM is overutilized (exceeds a threshold) then its VMs are reallocated for load-balancing. To estimate the PM utilization these heuristics adapt the utilization threshold dynamically based on LR (Local Regression), MAD (Median Absolute Deviation), and IQR (Interquartile Range).

We used different workloads to evaluate the performance of our proposed approach. The

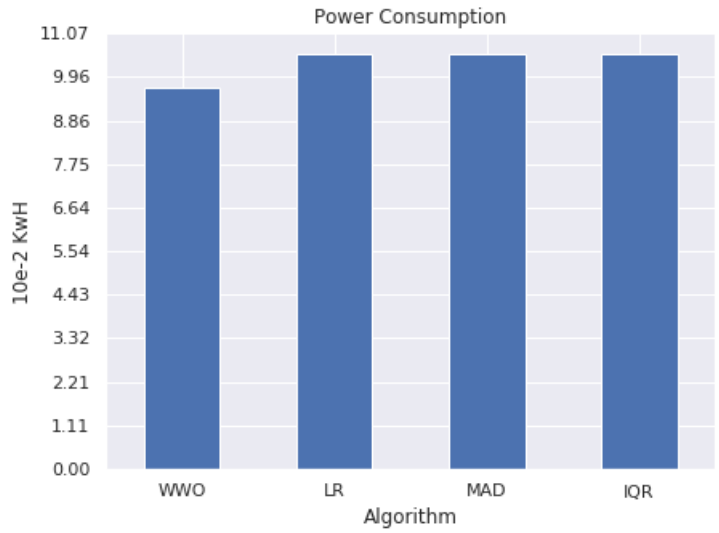

FIGURE 5.4: 30 Cloudlets

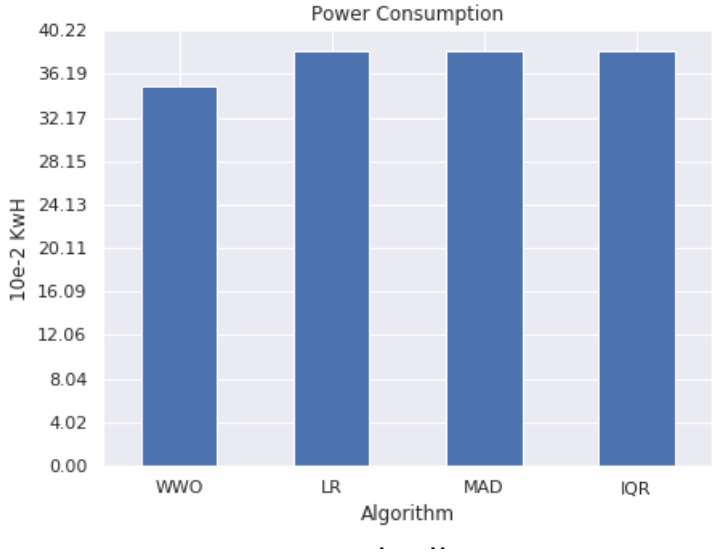

FIGURE 5.5: 60 Cloudlets

simulation results are depicted in Figures 5.2 to 5.7 and it is clear that results show that our proposed WWO-VMC approach surpassed the other algorithms in energy saving. And a significant energy saving observed for large workloads.

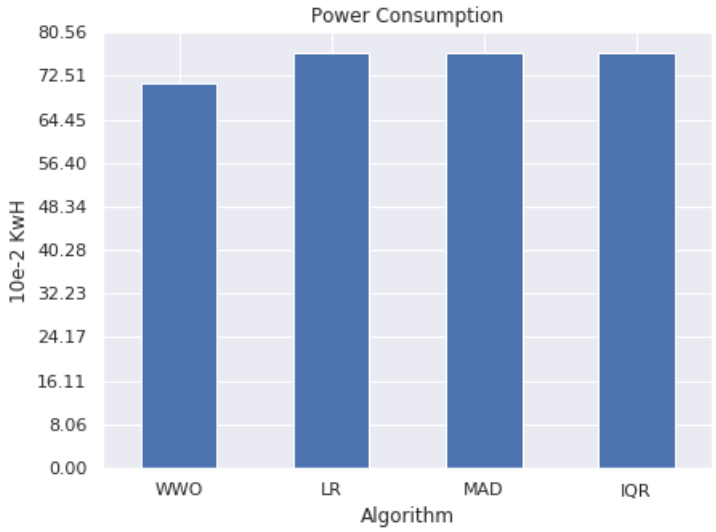

FIGURE 5.6: 90 Cloudlets

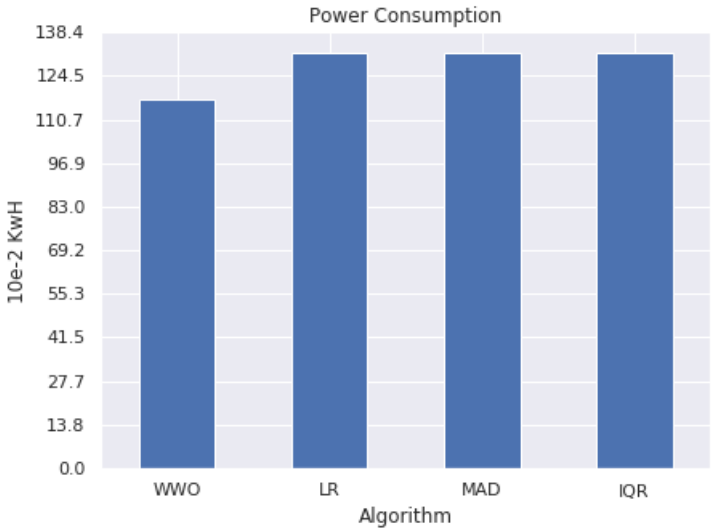

FIGURE 5.7: 120 Cloudlets

5.7 Conclusion

Cloud providers use different energy management strategies to maximize ROI. Energy-efficient VM consolidation is one such strategy to minimize monitoring costs of clouds. In this chapter, we present a dynamic efficient technique to place virtual machines on cloud servers by minimizing energy consumption. Our proposed WWO-based VM consolidation algorithm consumed 10 percent less energy compared to the standard

dynamic migration algorithms like MAD, IQR, and LR. This algorithm has the scope of optimizations in the parts of modification of parameters and definition of operations of refraction, propagation, and breaking. Further, the runtime of the algorithm can be optimized so that the convergence of the algorithm is faster.