**CHAPTER 4**

**IMPLEMENTATION AND RESULT DISCUSSION**

**4.1 IMPLEMENTATION PLATFORM**

The simulation platform is CloudSim v3.0.3 [6] integrated with NetBeans v8.0.2 IDE and Apache Commons Mathematics Library. CloudSim is a Java library for the simulation of cloud scenarios. It provides essential classes for describing datacenters, hosts, computational resources, virtual machines, applications, users, and policies for the management of various parts of the system such as scheduling and provisioning. CloudSim provides a generalized and extensible simulation framework that enables seamless modeling and simulation of app performance. By using CloudSim, developers can focus on specific system design issues that they want to investigate, without getting concerned about details related to cloud-based infrastructures and services.

NetBeans IDE is a Java based open source Integrated Development Environment that allows applications to be developed in modules and can be used by third party developers. It is a cross platform IDE that is primarily used for Java development but can also be used for development in other languages. NetBeans supports many languages like Java, C/C++, XML and HTML, PHP, JavaScript and JSP.

Apache Commons Math is a Java library that provides reusable and lightweight mathematics and statistics functions. It is modular and can be easily integrated with existing applications to provide mathematics functions. It is a limited dependency based library that requires not much external dependencies beside basic java dependencies.

**4.2 SIMULATION SETUP**

The following assumptions are made with respect to the simulation for the algorithm:

* Any VM is capable of servicing a request.
* The servers / hosts are assumed to homogeneous in nature.
* Grid based energy supply is used for operation of a datacenter when renewable energy falls below a threshold.
* Cost of generation of renewable energy is assumed to be negligible.
* The capacity of renewables and grid energy are properly provisioned according to workload.

Table 4.1 lists the simulation parameters used in the algorithm. These values are used to tune the algorithm to varying levels of accuracy.

Table 4.1 Simulation parameters

|  |  |
| --- | --- |
| **PARAMETERS** | **VALUES** |
| No of datacenters | 20 |
| No of hosts per datacenter | 5-10 |
| No of VMs | 100 |
| No of cloudlets | 500 |
| Number of PEs / VM | 4 |
| Number of MIPS / VM | 2500 |
| Length of time slot | 200 ms |
| Solar conversion efficiency | 20% |
| Wind conversion efficiency | 30% |
| Total solar irradiation area | 15000 m2 |
| Total rotor area | 25000 m2 |
| Density of Air | 1.225 kg/m3 |
| Eco factor of renewable energy | 1.0 |
| Eco factor of brown energy | 9.0 |
| Tunable parameter V | 0.1 |
| T | 0.85 |
| Nc | 20 |
| Ns | 5 |
| Ned | 6 |
| Ped | 0.25 |

**4.3 PERFORMANCE EVALUATION**

1. **Comparison of Cheap First Scheduling and QoS First Scheduling**

Figure 4.1 Finish Times of Cheap First vs. QoS First Scheduling

The above graph shows the comparison between Cheap First and QoS First Scheduling. From the graph, it is clear that QoS First Scheduling performs better than Cheap First since cheap first scheduling typically routes requests to the datacenter with minimum electricity price causing a lot of requests to be routed to a single datacenter in the same time slot.

1. **Evaluation of Lyapunov Optimization based Load Scheduling and Power Management under different V values**

Figure 4.2 Finish Times of Lyapunov Optimization with different V values

The above graph shows the evaluation of Lyapunov optimization based load scheduling and power management at different V values. V is a tunable parameter in Lyapunov optimization which takes values between [0,1]. Here, V values are considered at 0.1, 0.33 , 0.5. These values cause minimum finish times amongst other values and are considered for comparison. The graph shows that at V =0.1, the finish times of the cloudlets are minimum.

1. **Comparison of QoS First Scheduling and Lyapunov Optimization based Load Scheduling and Power Management**

Figure 4.3 Finish Times of Lyapunov Optimization vs. QoS First Scheduling

The graph above shows the comparison of Lyapunov optimization based Load Scheduling and Power Management vs. QoS First Scheduling. From the graph, it is clear that Lyapunov optimization efficiently improves the finish times of the cloudlets.

1. **Evaluation of Modified BFOA based Load Balancing and Power Management Algorithm at different thresholds**

Figure 4.4 Finish Times of M-BFOA at different Thresholds

The above shows the comparison between M-BFOA based Load balancing and power management algorithm at different threshold levels. The thresholds are used to determine if the load balancing should occur by comparing against the standard deviation of system load and can take values between [0,1]. Here the threshold levels are fixed at 0.85, 0.8, 0.75, 0.7. It is evident from the graph that at T=0.85, the cloudlets execute with minimum finish times.

1. **Comparison of Finish Times of Lyapunov Optimization and M-BFOA based load balancing algorithm**

Figure 4.5 Finish Times of Lyapunov Optimization vs. M-BFOA

The graph shows the comparison of finish times between Lyapunov optimization based Load scheduling and power management algorithm vs. M-BFOA based Load balancing and power management algorithm. The tunable parameter V in Lyapunov optimization is set at 0.1 and the threshold level in M-BFOA based algorithm is set at 0.85 since these two values generate minimum finish times. From the graph, it is clear that M-BFOA based algorithm is performs much more efficiently than Lyapunov Optimization in reducing finish times of cloudlets.

1. **Comparison of Time Average Eco Aware Power Cost of various strategies**

Figure 4.6 Eco Aware Power Cost of various strategies

The graph shows the comparison of time average eco aware power cost of different strategies. The strategies compared are M-BFOA based algorithm, Lyapunov based algorithm and the scheduling strategies with the power plans. The M-BFOA and Lyapunov based algorithms use Power Plan A for power management since it yields minimum eco aware power cost. From the graph, it is clear that the M-BFOA based load balancing and power management algorithm efficiently optimizes the time average eco aware power cost of the system, successfully integrating renewable energy into the system (green cloud datacenters).

1. **Comparison of Migration Counts of Lyapunov Optimization and M-BFOA based Load balancing algorithm**

Figure 4.7 Number of Migrations of Lyapunov vs. M-BFOA

The graph shows the comparison between Lyapunov optimization based algorithm and M-BFOA based algorithm in terms of number of migrations. Initially, the M-BFOA algorithm has higher number of migrations in the first two time slots since the system tends to be unbalanced and the algorithm proceeds to balance the system by migrating some tasks to other underloaded VMs. Once load balancing is completed, the number of migrations drastically reduces more than the Lyapunov based algorithm.

Table 4.2 shows the comparison of Lyapunov optimization and M-BFOA load balancing and power management algorithms against three factors: makespan, eco aware power cost, number of migrations.

Table 4.2 Comparison of Lyapunov vs. M-BFOA based algorithm

|  |  |  |
| --- | --- | --- |
| **PARAMETERS** | **Lyapunov** | **M-BFOA** |
| Makespan | 1439.2 ms | 1236.2 ms |
| Eco aware power cost | 43.94 | 29.01 |
| Number of migrations | 24 | 44 |

The proposed M-BFOA based load balancing and power management algorithm efficiently improves Makespan, Eco aware power cost and Number of migrations of the system better than the existing Lyapunov based algorithm. The proposed method shows 35% improvement in eco aware power cost and 45% improvement in reducing the number of task migrations in the system.