Energy Conservation for Datacenters in Cloud Computing using Genetic Algorithms

Vijaya Kumar[1], Dr. G. A. Ramachandra[2]

[1]Computer Science and Technology, Sri Krishnadevaraya University, Anantapuramu, A.P, India

[2] Computer Science and Technology, Sri Krishnadevaraya University, Anantapuramu, A.P, India

Abstract: ***In recent developments most of the organizations are focused on reduce the investment to the work environment and important concern, how to design the infrastructure and how to utilize maximum resources.The optimization of energy consumption is important concern for design of day to day life and future computing and distinguish techniques to improve performance of workload demand in the dynamic environment such grids, clusters and clouds. This paper proposes an improved genetic algorithm based on time cost and energy consumption models and we use the Dynamic Voltage scaling (DVS) and Dynamic voltage frequency scaling(DVFS) methodologies for reduce energy consumption and determine the optimal placement of virtual machines in order to maximize the overall renewable energy usage and minimize the energy consumption.***

Keywords: Dynamic Voltage Scaling (DVS), Dynamic Voltage frequency scaling (DVFS), Datacenter, Energy consumption, Genetic Algorithm.

1. Introduction

One of the main challenges in recent years is to reduce the Energy usage in datacenter. For example operating middle size organization has organized medium size datacenter having capacity 80000 kW power and we can estimate the computing resource consume around 1- 5 % of the world's total power usage. In modern datacenters, are not only expensive to maintain infrastructure, unfriendly to the nature to the environment and carbon emissions effects to the nature[1]. Most of the IT organizations utilize the high amount of energy and huge amount of carbon footprints are incurred due to massive amount of powering the number of servers hosted to the datacenters. To allow computing facilities to operate on high power will lead to a temperature of computing systems for long time so to reducing power consumption for datacenters over world. We proposed a novel optimization technique Parallel genetic Algorithm- to get the optimal solution of Job shop scheduling problem by using distinguish constraints for utilization of resources and also reduce large number of execution times to find near optimal solutions in intensive data.

We proposed a scheduling of virtual machines to reduce power consumption of parallel tasks and find the problem to minimize task execution time and also reduce power consumption. The main objective of proposed work is to define an effective genetic based parallel scheduler can be implemented in dynamic grid environment and also allocate resources. For better solutions of larger problems we implemented Modified Hybrid genetic Algorithm and Partition genetic algorithm to execute and process in a parallel mode.

Virtualization has been reduced the energy cost of an infrastructure - Physical server consolidation.Let us consider 1MW data center with 1000 physical servers that consume 150 W each at cost of $ 0.15 per kilowatt/hr. To calculate the energy cost per year of 1000 physical server (150w/1000 * 0.15 * 24hr *365 days * 1000 servers = 197100). By using the Virtualization of these servers conservative consolidation ratio of 10: 1, each physical machine can manage the 10 virtual machines at a CPU utilization of 70 % of total energy cost. (150/1000*0.15*24hr*365 days * 100 = 1314000.

2. Related Work

In the last decade, Information Communication Technology sector (ICT) is increased and investment amount on infrastructure is very high and it effects to the environment. To minimize the emission of green houses in ICT sector and it causes global warming and most of IT organizations have deployed datacenters for provision of cloud computing hosting of the internet applications and also govt sector is to promote energy efficiency for datacenters and to minimize the impact on the environment. In datacenters have size of thousands of servers, physical machines, virtual machines and switches', using lower energy usage is creating complex issues to larger servers and disks to perform as fast within the required time period [2]. A major problem with VM migration, sending loads to remote datacenters causes delay costs and energy cost due to increase an amount of data and virtual machine capacity transferred over the internet. To achieve efficient processing and maximum utilization and also minimize energy consumption in datacenters, we proposed the Modified Hybrid genetic algorithm to solve the dynamic resource allocation along with integrated allocation of resources and energy efficient transport technology reductions in the power consumptions in datacenters [3].

Power Model: Energy efficiency can be improve PUE metric (Power usage effectiveness), which is measure the quality of the datacenter infrastructure and calculate the total building power to IT equipment power.

$$PUE = \frac{Facility\ \ Power}{IT\ \ Equipmentpower}`$$

In this Power Model we assume total power consumption of Server ' S ' has linear relationship with all physical machines power utilization , we calculate the physical machine utilization

***Power Utilization* = Total power supply / number of physical machines is in active mode**

We also calculate the CPU utilization of each physical machine and Virtual machines

CPU utilization = Sum of the CPU utilization/Usage of CPU utilization

Power CPU $_{\text{utilization}}$ = Physical machine idle state + (Maximum CPU utilization - Physical machine idle state) × no. of Virtual machines

Total Energy consumption in a host in the interval of time [t1,t2] is defined as

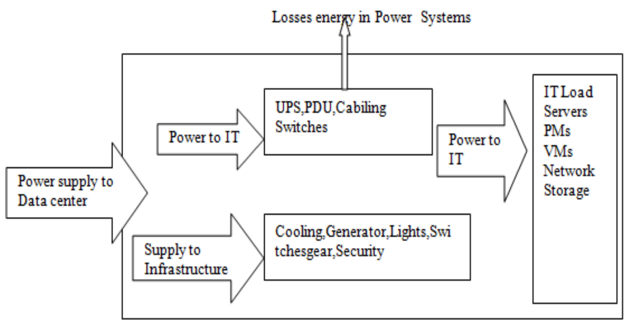
$$E = \int_{t1}^{t2} P(CPU\ \ utilization(t))\ dt$$

To increase the improving availability we will reduce the load across multiple systems, so load is reduced energy efficiency will be reduced and also we concern about to reduce losses in the power system and the power used for the infrastructure we used bulk amount of power consumption in the datacenters. If we reduce the IT load then automatically minimize the overall power supplying for the datacenter.Whenever we reduced the IT load, Infrastructure Load/IT load will always increase and it will effect to the PUE to increases the power usages.

$$PUE = \frac{IT\ \ load + Infrastructure\ \ Load}{IT\ \ Load}$$

$$= 1 + \frac{Infrastructure\ \ Load}{IT\ \ Load}$$

After implementation of virtualization in data centers, power usage effectiveness (PUE) , physical server consolidation effects on usage of the power usage.



Power Usage in Datacenter

3. Problem Description

We consider the problem of energy-efficient allocation of physical machines and virtual machines in cloud; we compute the scheduling problem as following

Given a set of m virtual machines to be placed on a set of n heterogeneous physical machines and each virtual machine VM_i requires p_e processing elements , Mbytes of physical memory, K bits of network bandwidth and the VM_i will be started at time (t) and completed the process at time $(t+t_n)$ without using the migration. We concern three types of computing resources such as processors, physical memory and network bandwidth.

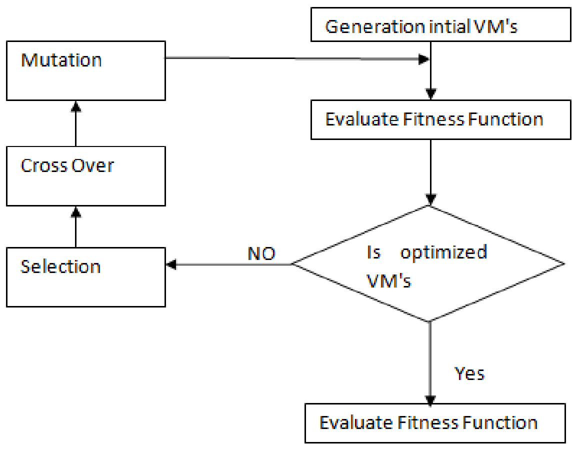
We assume that every host H_j can run any virtual machine and power consumption model of Hj has a linear relationship with CPU utilization.The objective of scheduling is exchange between minimizing total energy consumption in usage of maximum requirements of 'n' VMs, we contribute the below mentioned constraints

Constraint 1: We can run virtual machine and virtual machine migrated on identified host only.
Constraint 2: No Virtual machine request any resource is larger than capacity of resource in the host.
Constraint 3: Let Pi(t) be the set of indexes of VMs that are allocated to be host Hj, there is sum of total demand resource of allocated VMs is less than or equal to maximum capacity of the resource of the Hj. For every physical machine having each processing elements Hj(1,2,3,....m);

In this paper we proposed new model to solve the optimization of resource allocation problem for a group of users send their request to the servers within a specified interval of time. The main goal of this paper is to minimizing the average requests to the servers and we introduce integrated scheduling algorithms to schedule the virtual machines efficietively in datacenters.

In this genetic algorithm , we approach optimization and search technique based on the principles of genetic and natural selection, and it categories into four steps (a) Initialization (b) evaluation (c) exploitation (d) Exploration. For each iteration the genetic algorithm selects individual's virtual machines at random from the current datacenters to the servers and placed them optimal. To get optimal from the datacenters we can created using below rules

1) **Selection rules:** Select the individuals virtual machines from the servers which is having high configuration.
2) **Crossover rules:** To combine the virtual machines which is having the same capacity of size, applications and also users to optimal the next generation.
3) **Mutation rules:** To make them random changes to the genes or properties of the individual servers to create virtual machines.



In this paper we worked of resource scheduling algorithm using the properties of the Full Bin packing algorithm , Round Robin , Priority, Modified Round Robin + Priority scheduling algorithms, we compute the FBRRP scheduling algorithm which is consider as parameters in most efficient algorithm for allocating the virtual machines. Based on demanding for allocating the resource in enterprises and business marketing , number of users are access the different types of services is increases due to the increase the performance of resource allocation. We consider many aspects to increase the performance of allocation resource like Memory allocation for individuals virtual machines , High configuration CPU speed, Bandwidth[4-5]

3.1 Network Virtualization

Network Virtualization defined as logical networking devices and services such as logical ports, switches, routers, firewalls etc. We have designed based on sharing expensive of hardware and software resources like virtualization, provide access to the main servers from the different locations. To get the best optimization of network design we provide the best network design process to find the best layout components like reliability, transmission delay and cost[10].

In this approach we considering all terminal reliability and we assumption all edges in the network are identical and cost is depends on connected of two nodes. We make them as one reliability and cost alternative nodes for each pair of nodes, we will continue this approach to allow the edges to choose from different components with different reliabilities and costs. We used notations to describe the optimal design of the network for connecting and allowing edges from the different edges.

K: Number of options to connection for a edges
T: Option between the nodes
Xij: Edge option between nodes of i and j
r(Xij): Design option for reliability nodes
c(Xij): Unit cost of the edge option

Implementation: In genetic algorithm, for each network design 'x', formed into an integer vector, we consider as a chromosome, and each element of chromosome represents a possible connection of edge in the network. For each candidate architecture 'x' we have n × (n-1)/ 2 vector components and the value of each element connected to the specific edge with pair of nodes and connected with all possible chromosome of 0,1,2,3,....... k-1 and we get the possible network architectures is $K^{(n\times(n-1))/2)}$. To find the minimum cost of network architecture, it exceeds pre specified network reliability R_{min} and we consider a infeasible solution, breeding the feasible solution and infeasible solution we get the good feasible solution.

To solve the cost effective in servers we can partition into standard type and each server having their own energy instead of supplying power to all servers as same.

Table 1: Power allocation different types of Servers

Server Type	*Server Power*	*Network Allocation*	*Storage Allocation*
1U app server	250W	0.2	0.1
Virtual Server	90W	0.4	0.2
Web Blade	200W	0.3	0.1
ERP blade	200W	0.1	0.4
Mainframe	4000W	0.1	0.5
3U-10U Server	2000W	0.1	0.1

In datacenter, each server has a standard power level assigned allocation of power associated with networking and storage. We classification about the servers based in the power capacity.

- Assign all servers to a 'Server_Type '
- Assign to each user a number of servers form the Server_Type based on the 'MHGA'
- Compute the total power from all servers and normalize to compare the actual IT load power
- Implementation of PUE data to each server which is assigned to a Server_Type.

3.2 Dynamic Load Balancing and Distribution Algorithm (DLBDA):

We proposed DLBDA, which integrates the maximum resources discovery based on the user demand , server selection in the data center , virtual machine placement in the servers and requests from the users. In datacenters, the load balancing and distribution is working as per user demands, and also performance of servers is good , no need to implement DLBD algorithm. In some cases at a given particular time period, the server capacity maintaining overload and the server performance is decrease, reached the threshold point. To find the all traffic load and network proximity for each servers and separate the overload servers and under loaded server[7]. To find the under loaded servers list is selected and redirection to the overloaded servers list, so that we can minimized the traffic load to increases the performance of overloaded servers. For this we introduces load balancing point to share the usable server and mapping between the requests and allocate the set of suitable servers from under loaded servers[8]. By implementing the DLBD algorithm, we prevent web servers -- overloaded state and also multiple servers can serve in a peak demand, these web servers maintained the average load. If one of overloaded server and their average load is decreases due to the users in inactive and we can remove the web server at a time from the list and also replication for serving requests states changes into low priority[15]. When the users requests relinquish over time and allocate sufficient under loaded resources available to the datacenter and we can utilize them during the heavy load to the servers.

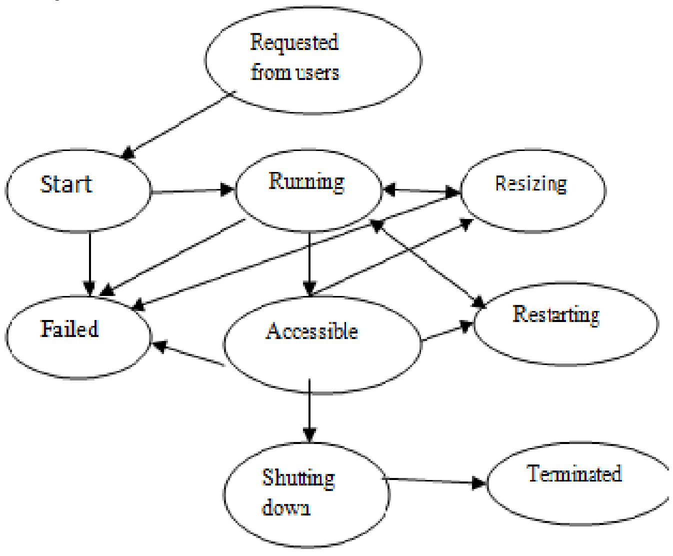


Figure 1: Load Balancing

Dynamic Load Balancing and Distribution Algorithm (DLBDA)

```
Step 1.  Begin
Step 2.  Switch on the all available servers in datacenter
Step 3.  for all r € R do
Step 4.  Search the list of all available servers n in different datacenters
Step 5.  for i =1 to n do
Step 6.  Separate the Overloaded server list (OSL) and Underloaded server list (USL)
Step 7.  end for
Step 8.  for j=1 to OSL do
Step 9.  Calculate incoming traffic load to the servers list based on the load balancing point (LBP)
Step 10. Measure network proximity from the OSL
Step 11. Calculate redirection to USL to minimization
Step 12. end for
Step 13. do
Step 14. Select OSL to optimal minimization
Step 15. Add LBP to OSL
Step 16. Redirected request to USL
Step 17. While alarm_flag = true
Step 18. if OSL > 1 then
Step 19. if OSL.avg Load <= alarm_LBP/2 then
Step 20. Remove least loaded server in OSL
Step 21. end if
Step 22. end if
Step 23. end for
Step 24. End
```

3.3 Virtual Machine Placement

To solve the problem of VM placement, we proposed Modified Hybrid genetic algorithm with multiple fitness and we divided the into two parts. First part is users requests to the VM provisioning and placement on hosts and second part is optimization of current allocation of VM's to get the optimization of first part, we implementation of Modification Hybrid genetic algorithm.

3.3.1 Initialization

To initialize a some part of the population and remaining part of population is initialized randomly, and offspring is applied by crossover between two parents selected randomly[11-12]. To improve the performance and bring the offspring to a minimum of we used two methods Remove optimal and local optimal and cost of the offspring is less than the cost of the any one of the parents then the parent with higher cost is removed from the population.[13] The offspring is added to the population , if the cost of the offspring is greater than the both of the parents then we make them as invalid. Shuffling the random number is generated within one and if the probability is less than the shuffling operator , randomly selected is removed and sequence is randomized and added to the population.[14]

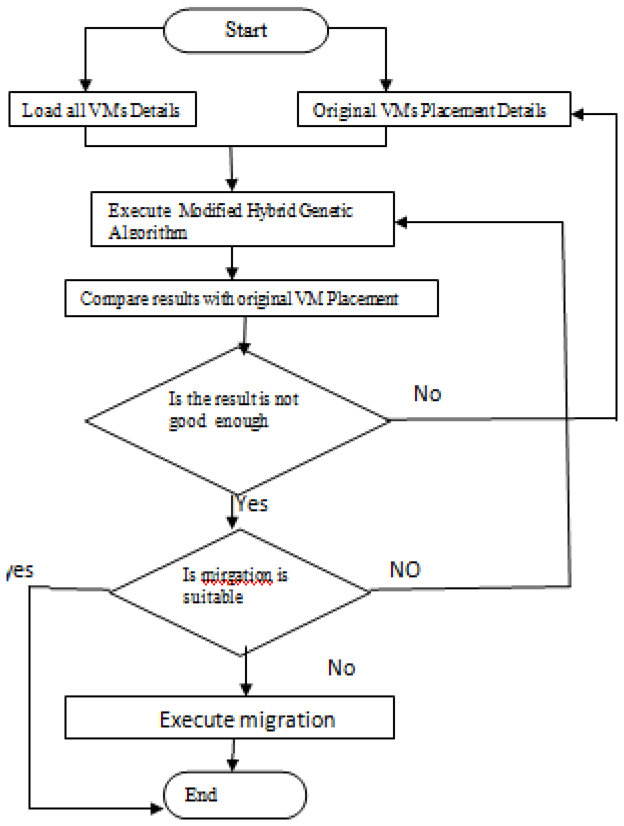

Start
Load all VMs Details
Original VMs Placement Details
Execute Modified Hybrid Genetic Algorithm
Compare results with original VM Placement
Is the result is not good enough
No
Yes
Is migration is suitable
yes
NO
No
Execute migration
End

Figure 2: VM Placement: MHGA

The flow diagram represents original placement information and load for all the virtual machines and the second stage MHGA executes the results and compared with the original placement information.[9-10] If the results is satisfied in the 3rd stage we move into the migration stage otherwise iteration will continue until the best optimal VM placement. In the migration stage, it chooses the best Virtual machines and also distances of the VM placements.

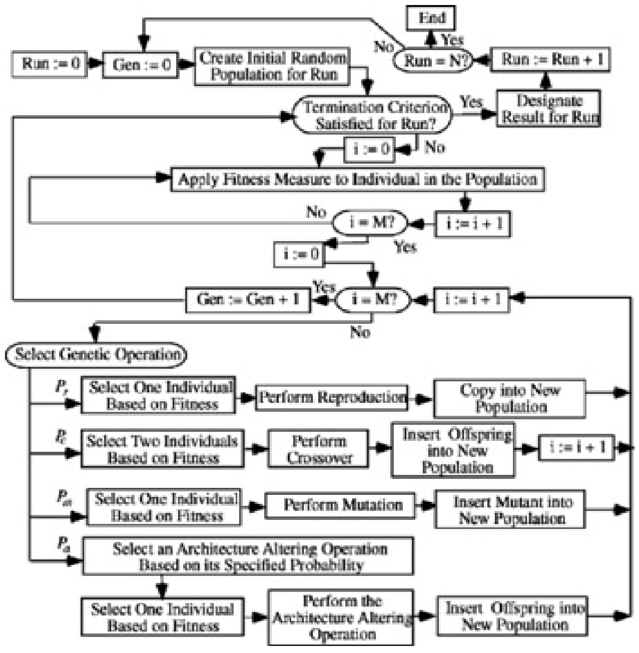

End
Yes
No
Run := 0
Gen := 0
Create Initial Random Population for Run
Run = N?
Run := Run + 1
Termination Criterion Satisfied for Run?
Yes
Designate Result for Run
i := 0
No
Apply Fitness Measure to Individual in the Population
No
i = M?
i := i + 1
i := 0
Yes
Gen := Gen + 1
Yes
i = M?
i := i + 1
No
Select Genetic Operation
Select One Individual Based on Fitness
Perform Reproduction
Copy into New Population
Select Two Individuals Based on Fitness
Perform Crossover
Insert Offspring into New Population
i := i + 1
Select One Individual Based on Fitness
Perform Mutation
Insert Mutant into New Population
Select an Architecture Altering Operation Based on its Specified Probability
Select One Individual Based on Fitness
Perform the Architecture Altering Operation
Insert Offspring into New Population

Figure 3: Flow Chat of MHGA

Algorithm A:

Step 1. Begin
Step 2. Initialize a first part of population using Initialization heuristics algorithm
Step 3. Initialize remaining part of population by using randomly
Step 4. Apply Remove optimal algorithm to all tours in the initial population
Step 5. Apply Local optimal algorithm to all tours in the initial population
Step 6. select two parents randomly
Step 7. Apply Crossover between parents and execute an offspring
Step 8. Apply Remove optimal algorithm to offspring
Step 9. Apply Local optimal algorithm to offspring
Step 10. if offspring < any one of the parents then replace the lowest offspring of the parent
Step 11. Shuffle any one of randomly selected from population
Step 12. Repeat Step 3 and Step 4
Step 13. Until number of iterations to get the optimal solution.
Step 14. End

For example, we consider a sequence of the cities ---A -----B-C-----D-----E----F------------G-----H------K., having distance for every city. In our scenario, Remove optimal , D is the city to be removed to perform the operations , E is the previous to the D and the next city is F, if the increases in the tour length after removing C to E to the next position , we decrease the tour length by moving into the remove city.

We can find out the decrease in the tour length: Decrease = Dist(C,D) + Dist(D,E)-Dist(C,E)

I. Remove Optimal: From above Algorithm A, we defined Remove optimal to increase the tour cost of city which is badly long distance.

a. Create a list containing the nearest N cities to a selected city.
b. Remove optimal, removes the selected city from the tour and form a N-1 cities.
c. Selected city which is nearest , is to reinserted in the tour and cost of the new tour length is calculated.
d. Continue the sequence which is produces the least cost is selected
e. Repeat 1-4 steps for each city in the tour.

II. Local optimal: From above Algorithm A, we select n consecutive cities (m1,m2,m3,m4,m5,....m n-1) from the tour and it arranges cities with minimum distance between the cities by searching all possible arrangements.

B. Crossover: To construct an offspring which is inheritance and adaptive the all properties from the parents structures and it acts an edge map. After getting the properties it stores information about all the connections to lead the distance between any two cities and each city have minimum two edges, maximum four edge associations from each parent

Algorithm B:
Step 1. Begin
Step 2. Initial city from one of the two parents from the selected city
Step 3. Remove optimal , all occurrence of the city which is selected two parents from the edge of map
Step 4. Selected city has entries in its edgelist go to step 5 otherwise go to step 6
Step 5. Find city in the edgelist of the selected city and get shortest edge from the tour, and broken randomly. go to step3
Step 6. If the parents chosen to all visited cities then stop. Otherwise randomly select an unvisited city and go to step 3
Step 7. Selected the city with least entries in its edge list as the next selected city, choose the nearest city.
Step 8. End.

4. Simulation Results

In genetic algorithm,. we implemented in C++ programming language for maximizing the fitness function by using $f(x) = x^2$, and the value ranges of x from 0 to 30. We implemented initial five populations of binary string and calculate x and fitness value $f(x) = x^2$. Use the tournament selection method to generate every new five populations and apply the cross-over operator for generating new populations. Apply mutation operator for population to get the best fitness value.

Enter the number of Population in each iteration is: 5
Enter the number of iteration is: 5
Iteration 1 is:

S.No	Population	Worst Fitness	Best Fitness
0	40269	0.40269	10.031417
1	1182511	1.82511	11.257072
2	1103802	1.03802	10.958928
3	1025375	0.25375	10.252110
4	1038920	0.3892	9.868195

Sum: 52.367722

Average: 10.473544
Maximum: 11.257072

Iteration 2 is:

S.No	*Population*	*Worst Fitness*	*Best Fitness*
5	1153524	1.53524	8.644072
6	31433	0.31433	9.864642
7	31433	1.3763	9.045399
8	8313	0.08313	10.042119
9	1074001	0.74001	9.298878

Sum: 46.895107
Average: 9.379022
Maximum: 10.042119

Iteration 3 is:

S.No	*Population*	*Worst Fitness*	*Best Fitness*
10	1186753	1.86753	11.619774
11	80292	0.80292	10.063322
12	158525	1.58525	9.255855
13	91516	0.91516	9.592331
14	103803	1.03803	10.959062

Sum: 51.490341
Average:10.298068
Maximum: 11.619774

Iteration 4 is:

S.No	*Population*	*Worst Fitness*	*Best Fitness*
15	1173828	1.73828	8.396161
16	64429	0.64429	10.632739
17	1099734	0.99734	9.900976
18	130327	1.30327	9.893287
19	1088392	0.88392	10.438623

Sum: 49.261787
Average: 9.852358
Maximum: 10.632739

Iteration 5 is:

S.No	*Population*	*Worst Fitness*	*Best Fitness*
20	1149948	1.49948	10.060295
21	10010	0.1001	9.999846
22	130283	1.30283	9.911272
23	88675	0.88675	10.36997
24	17727	0.17727	9.883524

Sum: 50.224907
Average: 10.044981
Maximum: 10.36997
After the 5 Iterations, the Maximum Value is: 11.619774

Compare between Iteration 1 and Iteration 2

Best Fitness (0-4)	*Best Fitness (5-9)*
10.031417	11.619774
11.257072	10.063322
10.958928	9.255855
10.25211	9.592331
9.868195	10.959062

Compare between Iteration 3 and Iteration 4

Best Fitness (10-14)	*Best Fitness (15-19)*
10.060295	8.644072
9.999846	9.864642
9.911272	9.045399
10.36997	10.042119
9.883524	9.298878

Compare between Iteration 4 and Iteration 5

Best Fitness (15-19)	*Best Fitness (20-24)*
8.644072	8.396161
9.864642	10.632739
9.045399	9.900976
10.042119	9.893287
9.298878	10.438623

Compare between Iteration 5 and Iteration 1

Best Fitness (20- 24)	*Best Fitness (0-4)*
8.396161	10.031417
10.632739	11.257072
9.900976	10.958928
9.893287	10.252110
10.438623	9.868195

Graphs Representation for Best Fitness Function

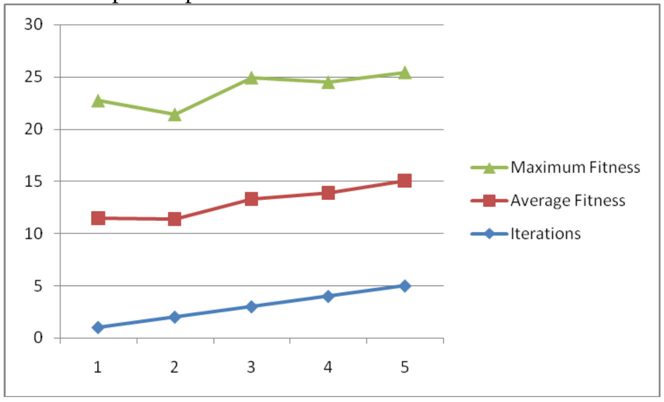


Figure 4: Best Fitness Value

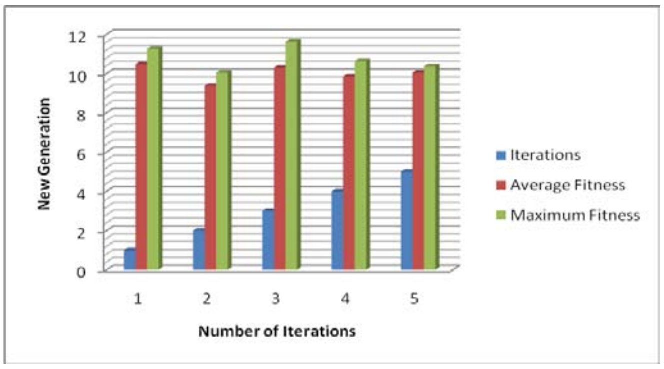


Figure 5: Compare between 5 iterations for Best Fitness Function

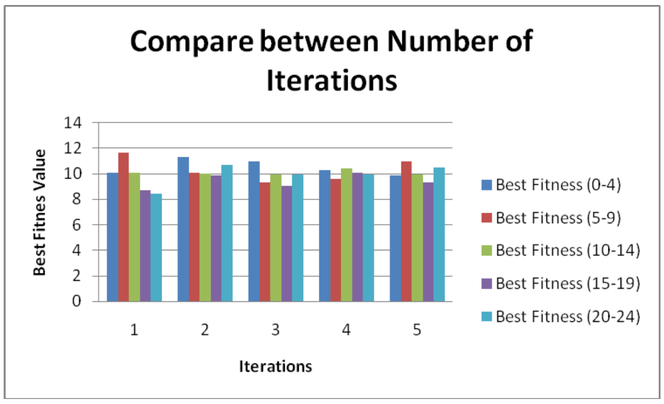


Figure 6: Comparing between Number of Iterations

5. Conclusion

In this paper, we optimize the energy consumption in datacenters in all levels. Our energy model is based on the Dynamic voltage scaling and improves the load balances by using Dynamic Load Balance Distribution Algorithm (DLBDA). We address the resource allocation and solved by adopting the features of Hybrid genetic algorithm and implemented the Modified Hybrid Genetic algorithm (MHGM) allocated the resources based on the Best Fitness values from the Populations in every iteration levels.

6. Acknowledgment

This research was supported by the Computer Science and Technology in Sri Krishnadevaraya University, Anantapuramu, Andhra Pradesh.

References

[1] Enokido T, Aikebaier A, Takizawa M. Process allocation algorithms for saving power consumption in peer-to-peer systems. IEEE Transactions on Industrial Electronics 2011; 58(6):2097–2105.
[2] Min-Allah N, Hussain H, Khan SU, Zomaya AY. Power efficient rate monotonic scheduling for multi-core systems. Journal of Parallel and Distributed Computing 2011. DOI: 10.1016/j.jpdc.2011.07.005.
[3] Buyya R. High-performance cluster computing: architecture and systems. Upper Saddle River, NJ, USA: Prentice Hall PTR; 1999.
[4] Kim FH, Beloglazov A, Buyya R. Power-aware provisioning of virtual machines for real-time cloud services.Concurrency and Computation: Practice and Experience 2011; 23(13):1491–1505.
[5] Beloglazov A, Buyya R, Lee YC, Zomaya AY. A taxonomy and survey of energy-efficient data centers and cloud computing systems. Advances in Computers 2011; 82:47–111
[6] Y.-T. Wang, and R. J. T. Morris, "Load Sharing in Distributed Systems," IEEE Transactions on Computers, Vol. C-34, No.3, March 1985, pp. 204--21
[7] Xhafa F, Carretero J, Abraham A. Genetic algorithm based schedulers for grid computing systems. International Journal of Innovative Computing, Information and Control 2007; 3(5):1053–1071
[8] Fonseca CM, Fleming PJ. An overview of evolutionary algorithms in multiobjective optimization. Evolutionary Computation 1995; 3(1):1–1.
[9] Khan SU. A self-adaptive weighted sum technique for the joint optimization of performance and power consumption in data centers. 22nd International Conference on Parallel and Distributed Computing and Communication Systems (PDCCS), USA, 2009; 13–18.
[10] T. Wood, P. Shenoy, A. Venkataramani, and M. Yousif, “Black-box and gray-box strategies for virtual machine migration,” in Proceedings of the 4th USENIX Symposium on Networked Systems Design and Implementation (NSDI’07), 2007, pp. 229–242.
[11] Mejia-Alvarez P, Levner E, Mossé D. Adaptive scheduling server for power-aware real-time tasks. ACM Transactions on Embedded Computing Systems 2004; 3(2):284–306.
[12] Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs. Springer: Berlin Heidelberg New York, 1992.
[13] Hotovy S. Workload evolution on the Cornell Theory Center IBM SP2. Job Scheduling Strategies for Parallel Proc. Workshop, IPPS’96, Honolulu, Hawaii, 1996; 27–40.
[14] Xhafa F, Carretero J, Abraham A. Genetic algorithm based schedulers for grid computing systems. International Journal of Innovative Computing, Information and Control 2007; 3(5):1053–1071.
[15] M.Aleksy, A.Korthaus, and M. Schader, “Design and implementation of flexible load balancing service for CORBRA- based applications,". in PDPTA;01: USA: IEEE Computer Society, June 2001

Author Profile

C. Vijaya Kumar is currently a Ph.D, student in the Department of Computer Science and Technology from Sri Krishnadevaraya University. He received the B.Sc degree in Computer science from Sri Krishnadevaraya University and M.C.A from the Sri Venkateswara University Tirupathi, in 2010, research interests include Cloud Computing, Hadoop and Data mining.

Dr. G. A. Ramachandra is an Associate professor at Sri Krishnadevaraya University. His research interests are Data Mining, Computer Networks and Cloud computing.