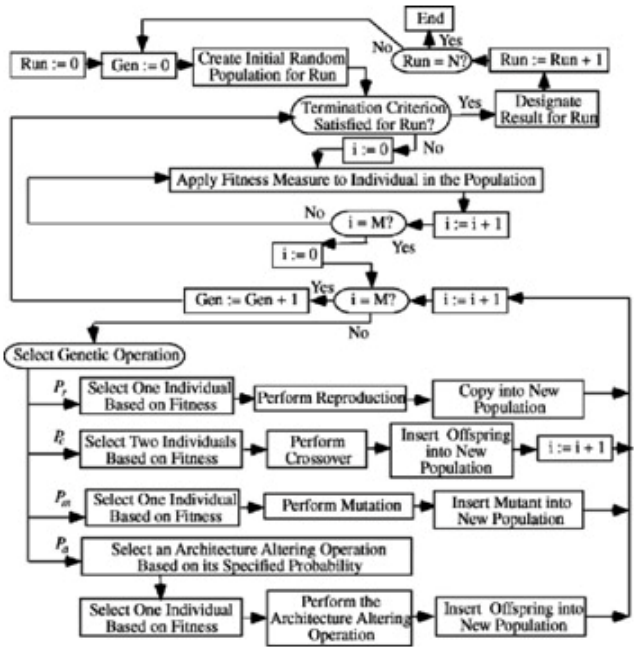


**Figure 3:** Flow Chat of MHGA

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

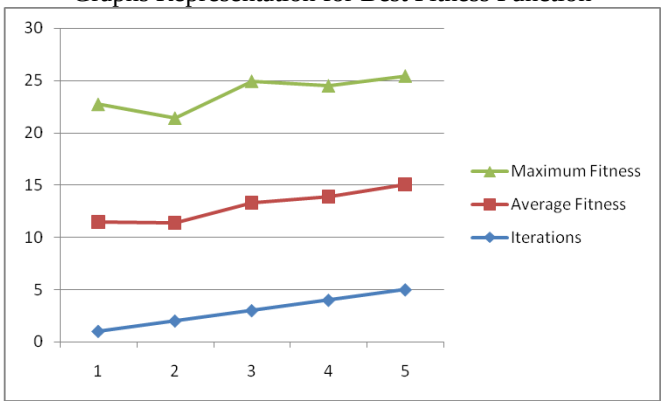


**Figure 4:** Best Fitness Value

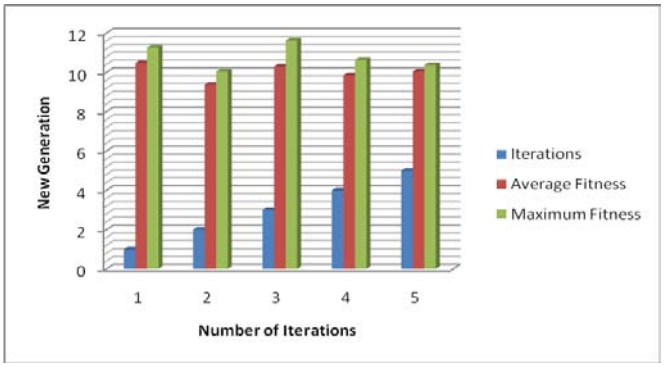


**Figure 5:** Compare between 5 iterations for Best Fitness Function

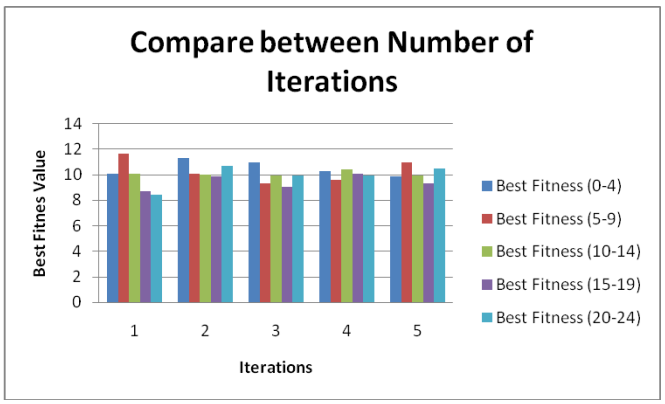


**Figure 6:** Comparing between Number of Iterations

## 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.