International Journal of Cloud Computing and Services Science (IJ-CLOSER)
Vol.3, No.3, June 2014, pp. 190~199
ISSN: 2089-3337

Energy Efficient and High Performance Scheduling Algorithm for Grid Computing

Nidhi Jain*, Inderveer Chana**
* M.E (S.E), Departement of Computer Science and Engineering, Thapar University
** Associate Professor, Departement of Computer Science and Engineering, Thapar University

Article Info

Article history:

Received Apr 17th, 2014
Revised May 29th, 2014
Accepted June 11th, 2014

Keyword:

Energy Efficient
High Performance
QoS

ABSTRACT

Grid computing is an epitome that furnishes seamless access to widely distributed resources for solving the big computing applications. Large amount of power consumption in grid computing infrastructure is the main concern because it conducing high operational costs and carbon emission to the environment. To reduce the power consumption without degrading the performance is the main challenge of grid computing. In this paper we present a scheduling algorithm which focuses on simultaneous optimization of both energy consumption and performance. We take an energy aware layered architecture in which Task analyzer (layer 1) analyses & sort the tasks on the basis of their complexities and dependencies then Provisioning Manager (layer 2) provisions the resources and tasks on the basis of energy and performance. Then Scheduler (layer 3) schedules the tasks on the resources in accordance of EEHP algorithm. We have validated our approach by conducting an experiment using the Gridsim toolkit. The results demonstrate that a large amount of energy can be saved without degrading the performance.

Corresponding Author:

Nidhi Jain,
Departement of Computer Science and Engineering,
Thapar University,
Patiala, Punjab.
Email: nidhijain9151@gmail.com

1. INTRODUCTION

Grid computing computes the large-scale problems in science, engineering, and commerce by uniting the power of disseminated resources dynamically based on their availability, potentiality, makespan, cost, and users QoS requirements [3]. Grid computing provides reliable, uniform and seamless access to distributed resources. So with grid computing, distributed resources can be efficiently utilized, big problems can be easily solved. But computational grid infrastructure requires large amount of energy consumption. Recent surveys show that power consumption within IT infrastructure is increasing rapidly. In 2006, about 61 billion of kilowatt-hour energy was consumed in data centers [5] [6]. This amount is equivalent to the double of the energy consumed in 2000 year. Increase in energy consumption by high performance computing system and distributed system becomes an important concern for technical, financial and environmental reasons [2]. The primary goal of these large-scale distributed systems is to provide the high performance. But now cost spent in energy consumption is increasing so rapidly that there is a great need to control it. Distributed system management is in dilemma and facing the question "Whether to control the energy consumption at the cost of performance is having worth or not". So to come out from this energy-performance trade off, an energy & performance efficient algorithm is required. To reduce the energy consumption of large scale infrastructures a large amount of papers have been focused and numerous algorithms have been proposed to improve the energy efficiency without degrading the performance. Here an energy-conscious algorithm with high performance is proposed that consider processors with high performance/watt, energy-efficient storage media and complexities of tasks and dependency i.e. a communication latency of a task.

In this concern, we have analyzed several ways for reducing power consumption in scientific grid environments. This work looks into the heterogeneous grid resources towards preceding the execution of "complex and dependent tasks" on more energy and performance efficient base. The rest of the paper is organized as follow. Section 2 presents the Energy aware grid layered architecture and Energy Efficient High Performance (EEHP) scheduling algorithm is explained in detail. Section 3 contains tests and simulation results performed on Gridsim toolkit. Finally, Section 4 shows conclusions and future directions.

2. ENERGY EFFICIENT AND HIGH PERFORMANCE SCHEDULING APPROACH

Three layered energy and performance aware grid architecture is proposed as depicted in Figure 1. It focuses on reducing the energy consumption simultaneously improving the performance in Grid environment. It is described in three layers which are Grid Portal, Provisioning manager and Scheduler.

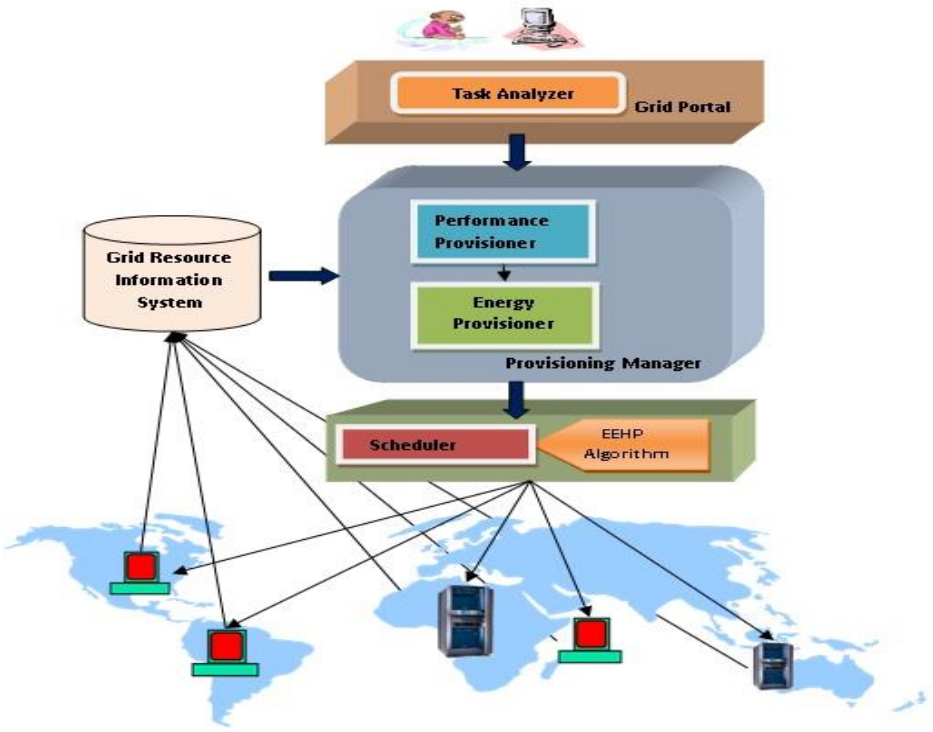

Figure 1. Energy-Aware Layered Grid Architecture

2.1. Grid Portal

Grid Portal provides an interface to the users through which user can submit the task and the information related to the task that is required for its execution as shown in Figure 2. Information gathered from interface is stored and used by task analyzer to analyze the task as how much complex the task and how much dependent it is. To measure the complexity of task, Lines of Code (LOC) and Function Point metrics have been used. Function Point considers 14 complexity factors that counts processing complexity of the project or a task. To measure the dependency in a task Fan-in, Fan-out and Information Flow Metric have been used.

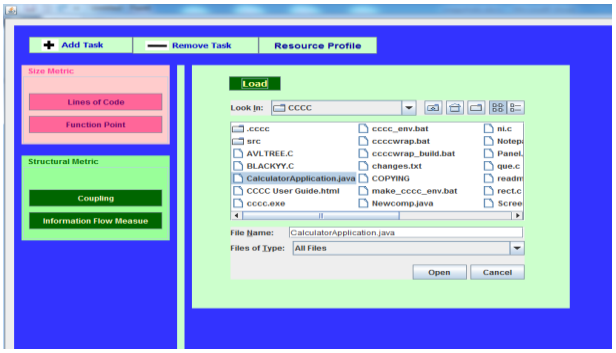

Figure 2. Grid Task Submission Interface

2.1.1. Task Analyzer

Information about the tasks like size, Complexity and dependency of a task is collected through Grid Portal then analyze these tasks on the basis of collected information as which task is heavily loaded, complex or dependent. A set of five tasks have been taken for the experimental purpose i.e. shown in Table 1 with their collected information. To calculate the Function Point metric of the tasks, a separate interface is provided to the users as shown in Figure 3. To calculate the LOC, Fan-in, Fan-out and Information Flow Metric of the tasks, C and C++ code counter (CCCC) free software tool [4] for the measurement of source code related metrics is used. These entire submitted tasks shown in Table 1 are analyzed on the basis of size and structural metric and sufferage value [1] based on time and energy using the following equation:

$$weight = size * \mathrm{var}_{time} * \mathrm{var}_{energy} + dependency \rightarrow 1$$

- *Size*: Size of a task can be calculated using either LOC [26] metric or Function Point metric [27]. To calculate the weight of a task, size metric is used because tasks execution time and energy consumption differs as the size and complexity of task varies. Function Point metric considers 14 technical complexity factors that are used to provide an indication of problem complexity [15]. Higher the complexity or size of task then task requires more time and energy to execute. LOC of the tasks which are specified in Table 9 is calculated using CCCC tool and Function Point is calculated by providing separate interface for its calculation.

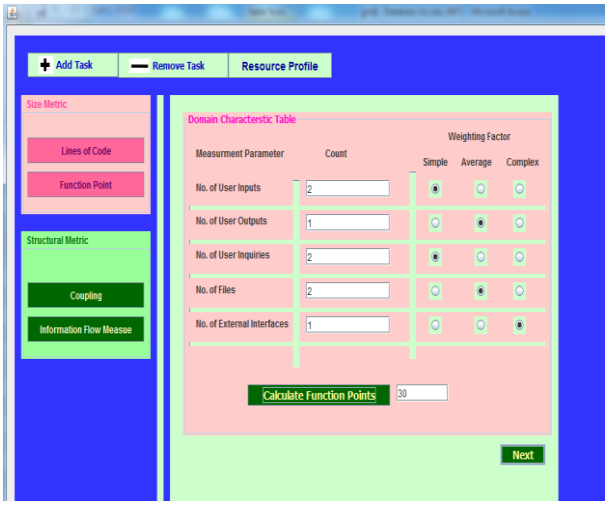

Figure 3. Function Point Interface

- var_{time}: It refers to the Variance of time i.e. the difference between the execution time of a task on execution upon the most energy efficient resource and least energy efficient resource. This factor is responsible for maintaining the performance of a task. Task with high value of var_{time} shows that its performance will be affected more if it does not allots to the more energy efficient resource. So the task with high value of var_{time} is supposed to be of high weight. Value of var_{time} for the tasks is calculated using the past data of the resources i.e. stored in database.

- var_{energy}: It refers to the Variance of energy consumption i.e. the difference between the energy consumption by most energy efficient resource and least energy efficient resource while executing a particular task. This factor is responsible for maintaining the energy consumption of a task. A task with high value of var_{energy} shows that it will consume more energy if it does not allots to the more energy efficient resource. So the task with high value of var_{energy} is supposed to be of high weight. Value of var_{energy} for the tasks is calculated using the past data of the resources i.e. stored in database.
  var_{time} and var_{energy} [28] Factors are based on the idea of sufferage value [1].

- *Dependency*: It refers that how much modules of a task are dependent on each other. If dependency of a task is higher it means communication latency is high. During communication between modules, CPU sits idle while current module is busy in doing input/output operations as other modules are dependent on that module. It means processor is not doing anything but still consuming energy. So the task with high dependency has been assigned high weight than low dependency task because for task with no dependency CPU is busy all time in executing some task while in highly dependent task CPU sits idle and consumes energy even when it does not execute any task. So it is worth to assign more energy efficient resource to high dependency task. To measure dependency following metrics have been used:

— *Coupling between Objects*: It refers to the count of other modules of a program which are coupled to the current module either as a client or a provider. High coupling indicates the dependency between the modules is high [14].

— *Information flow measure*: It is measured as the square of the product of the fan-in and fan-out of a single module. So it is a combined measure of Fan-in & Fan-out [14].

Thus, the tasks are prioritized. Tasks with high weight have been assigned high priority. After calculating weight, tasks are sorted with task ID using bubble sorting.

Table 1.Task Information

Tasks	**Size (Lines of Code)**	**Size(Function Point)**	**Dependency (Information Flow metric)**	Var_{time} **(secs)**	Var_{energy} **(watts)**	**Millions of Instructions (MI)**
Gridlet 0	286	28	0	5	3917	222340
Gridlet 1	474	15	0	5	8143	4741236
Gridlet 2	1788	82	1	7	29817	13178798
Gridlet 3	116	30	0	5	936	20000011
Gridlet 4	85	8	0	5	936	12356785

2.2. Provisioning Manager

It handles the provisioning of task and resources both for performance to satisfy the quality requirements of users and for reducing the energy consumption to reduce the cost. It analyzes the task information collected from the grid interface in terms of energy consumption and their performance requirements (time and cost). It also takes all the information about resources from grid resource information system as shown in Figure 1 and arranges them on the basis of energy and performance efficiency. It focuses on Energy and Performance factors.

2.2.1. Performance Provisioner

It provisions the tasks and resources both for Performance. In case of tasks, by using the past information of execution time of the tasks on different resources, variance of time is calculated i.e. a difference between the execution time of a task on the most performance efficient resource and least efficient resource. The task having maximum value of variance of time affects the performance highly as calculated by equation 1. So a task with high value of variance of time assigned high priority to schedule and execute first. In case of resources, Joulesort benchmark [17] is used that defines energy and performance efficient resources. These efficient resources use SSD (Solid State Disks) drives that are better in performance [19] [20]. Resources are shown in Table 2. The resources that are proved as an energy and performance efficient by Joulesort benchmark have been considered.

2.2.2. Energy Provisioner

It provisions the tasks and resources both to reduce the energy consumption. In case of tasks, by using the past information of energy consumption of the tasks on different resources, variance of energy is calculated i.e. a difference between the energy consumption by a task on the most energy efficient resource and next efficient resource. The task having maximum value of variance of energy affects the energy consumption more as shown in equation 1. So a task with high value of variance of energy assigned high priority to schedule and execute first. In case of resources, Joulesort benchmark is used that defines energy-efficient resources. Joulesort as a benchmark and it is selected to measure the energy efficiency because sort stresses the all core components of a system: CPU, memory and I/O [17]. Five most energy efficient resources according to Joulesort benchmark have been taken to carry out the simulation in Gridsim as shown in Table 2 below. Mips Rating is a metric to measure the performance of processor [25]. MIPS value of all these processors mentioned in Table 2 is calculated using clock rate and No. Of records sorted per joule information [16].

Table 2.Resources List [18][16]

Resources	Configuration	Clock Rate	Mips Rating	Power consumption in watts for 10 GB data
Intel Core i7-2700K	16GB RAM, 16 x 300 GB Intel 710 Series SSDs, 1 160 GB Intel 510 Series SSD	3.5 GHz	897500	165
Intel Core i5-2400S	16GB RAM, 7 x 120 GB Intel 510 Series SSDs	2.5 GHz	873750	93
Intel Xeon L3426	12GB RAM, Fusion-io ioDrive (80GB), 4 x Intel X25-E (3 x 32GB, 1 x 64GB)	1.86GHz	561250	105
Intel Atom 330	4GB RAM, 4 x Super Talent UltraDrive GX MLC 256GB	1.6GHz	310000	142
Quad Core AMD Opteron 2373	16GB RAM 80GB FusionIO	2.01GHz	145000	252

2.2.3. Scheduler: It collects all the information from the provisioning manager and schedules the tasks to the resources according to the EEHP (Energy-efficient and High performance) algorithm.

EEHP Algorithm: It assigns the high priority task i.e. complex in size, highly dependent to the most energy and performance efficient resource. As shown in equation 1, task with high weight has high priority. Then map the high priority task to the most energy and performance efficient resource that are found out by joulesort benchmark. It is explain in detail in Figure 4.

Energy-Efficient High Performance Algorithm

```
1 //Declarations and definitions
2 T = {t1, ..., tn}  // set of the tasks
3 R = {r1, ..., rn} // set of energy and performance efficient grid resources
4 loc (t) // size of tasks in terms of lines of code
5 fp (t) // function point value of the tasks
6 var_time(t)// variance of time
7 var_energy(t) // variance of energy
8 dependency(t) // dependency of modules in a task
9 LT [] // list of tasks prioritized by most complex, dependent and  decreasing energy
waste
10 LR [] // list of resources which are highly energy and performance efficient resources
11 pos(L, criterion) // insert position of new element on the basis of weight criteria at
list L
12 weight(t) // the weight of a task
13 power(r) // power consumption rate of resource
14 // Task Analyzer Phase
15 For each task t in T do
16 weight(t) = analyzer(t)
17 end for
18 //Provisioning Phase
19 For each t in T do
20 p = pos(LT, weight(t))
21 add(t, ET, v) // add t task to LT list at p position
22 end for
23 // Scheduling Phase
24 while ∃t in LT do
25 t = LT.get(0) // the most complex, dependent and power expensive task
26 re = LR.get(0)// the most energy & performance efficient resource
27 schedule(t,re) // assigns t task to the re
28 end if
29 remove(LT, 0) // removes first item list
30 end while
```

Figure 4.EEHP Algorithm

3. RESULTS AND ANALYSIS

Gridsim [21] is used to simulate the grid environment consisting of five most energy efficient resources as given in Table 2. For each resource their configuration, Clock rate, MIPS rating and power consumption rate is given. Joulesort benchmark defined them as most energy and performance efficient resources.

Tasks are prioritized using equation 1 and according to its configuration then map to the energy and performance efficient resources according to Joulesort benchmark as given in Table 2 and give the following result: Table 3 shows the simulation result for EEHP algorithm. It shows the energy consumption in joules, Execution time and cost spent on the tasks.

Table 3.Simulation Results for EEHP Algorithm

Tasks	Execution time(Secs)	Cost(Rs)	Execution Energy(watt-secs(Joules))	Resource
Gridlet 2	15.683	47.051	2587.843	Resource 0
Gridlet 4	15.1422	45.426	1408.228	Resource 1
Gridlet 1	8.4476	25.342	887.0018	Resource 2
Gridlet 3	64.5161	193.548	9161.295	Resource 3
Gridlet 0	2.5333	7.6001	638.4115	Resource 4
Total	106.3219	318.9671	14682.7793	

Table 3 shows the simulation result of EEHP algorithm on Energy-efficient resources found out by Joulesort benchmark. Table 4 shows the simulation result of EEHP algorithm on other resources. On comparing these results we can conclude vast amount of energy can save as well as performance can improve by using highly efficient resources.

Comparative Analysis of EEHP algorithm with other algorithms

For comparing the different algorithms, a set of resources have been taken as shown in Table 4. Different algorithms have been executed in Gridsim [21] using these resources and compare on the basis of energy,

time and cost. Table 5 shows the simulation result of EEHP algorithm executed using Table 4 resources. Table 6 shows the simulation result of HGreen algorithm. Table 7 shows the simulation result of Greedy-min heuristic i.e. a DVS approach. Table 8 shows the simulation result of Combined MINmin heuristic i.e. an approach for multicore heterogeneous grid computing environment. All these algorithms attempts to reduce the energy consumption and improve performance. Figure 4 depicts the comparative analysis of EEHP algorithm with others in graphical form and also shows how EEHP algorithm is performed better than other algorithms in terms of both energy and consumption.

Table 4.Resources List for Comparison [5]

Resources	CPE	Mips Rating	Power consumption in watts (SPECpower benchmark)
HP ProLiant DL360 (Intel Xeon L5520)	1486	2528	81
Bull NovaScale R440 (Intel Xeon X5500)	2931	2802	70
HP ProLiant DL380 (Intel Xeon 5570)	392	1515	182
SGI Altix XE250 (Intel Xeon X5200)	781	1122	165
Dell PowerEdge 2970 (AMD Opteron 2356)	535	1090	129
Supermicro 6025B -3RV (Intel Xeon E5345)	455	1074	209

Table 5.Simulation Results of EEHP Algorithm (With Table 4 resources)

Tasks	Execution time(Secs)	Cost(Rs)	Execution Energy(wattsecs(Joules))	Resource
Gridlet 2	455.1343	1365.403	27308.063	Resource 4
Gridlet 4	4849.601	14548.804	426764.945	Resource 2
Gridlet 1	3586.411	10759.234	555893.7821	Resource 5
Gridlet 3	16807.731	50423.195	2504352.057	Resource 3
Gridlet 0	190.3867	571.1601	41694.689	Resource 1
Total	25889.264	77667.7961	3556013.536	

Table 6. Simulation Results of HGreen Algorithm

Tasks	Execution time(Secs)	Cost(Rs)	Execution Energy(watt-secs(Joules))	Resource
Gridlet 2	454.134	1362.403	27248.063	Resource 4
Gridlet 1	1861.767	5585.302	163835.554	Resource 2
Gridlet 0	169.184	507.553	26223.608	Resource 5
Gridlet 3	16806.731	50420.195	2504352.057	Resource 3
Gridlet 4	10525.370	31576.111	2305056.145	Resource 1
Total	29817.186	89451.564	5026715.282	

Table 7.Simulation Results of Greedy-Min

Tasks	Execution time(Secs)	Cost(Rs)	Execution Energy(watt-secs(Joules))	Resource
Gridlet 0	77.616	232.8483	4656.967	Resource 4
Gridlet 2	517.228	1551.685	45516.100	Resource 2
Gridlet 1	3586.411	10759.234	555893.782	Resource 5
Gridlet 4	10383.852	31151.558	1547194.088	Resource 3
Gridlet 3	17035.784	51107.353	3730836.804	Resource 1
Total	31600.891	94802.6783	5884097.741	

Table 8.Simulation Results of Combined MINmin Heuristic

Tasks	Execution time(Secs)	Cost(Rs)	Execution Energy(watt-secs(Joules))	Resource

Gridlet 0	77.616	232.8483	4656.967	Resource 4
Gridlet 2	517.228	1551.685	45516.100	Resource 2
Gridlet 1	3088.753	9266.259	531265.532	Resource 1
Gridlet 4	10384.852	31154.558	1547343.088	Resource 5
Gridlet 3	15128.601	45385.804	2344933.211	Resource 3
Total	29197.05	87591.1543	4473714.898	

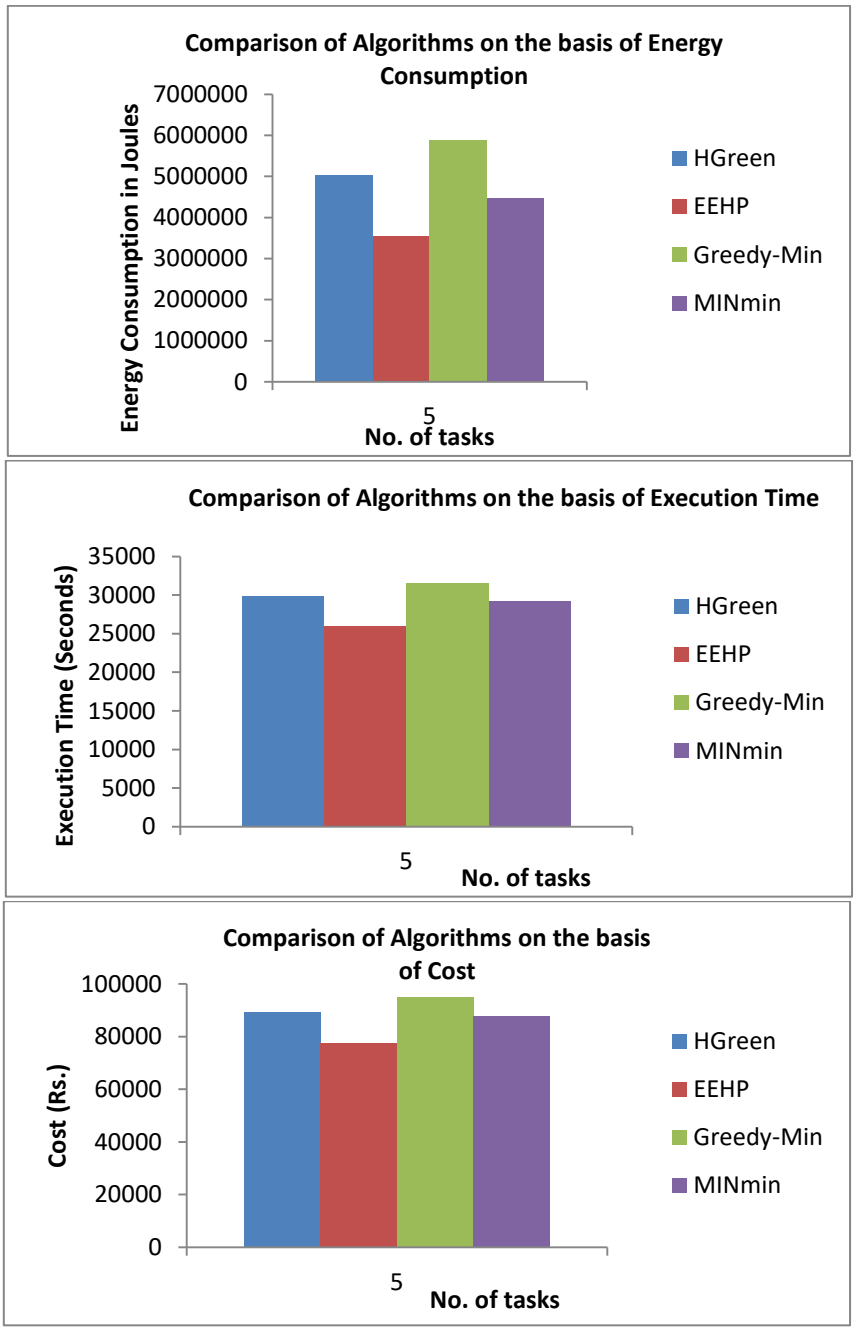

Figure 5. Comparison of EEHP algorithm with other algorithms in Graphical Form

4. CONCLUSION AND FUTURE WORK

The objective of this work is to reduce the power consumption without degrading the performance in global grids. For achieving this challenge, resources that are proved as energy and performance efficient by

joulesort benchmark have been used and tasks are prioritized by considering complexity of tasks, dependency within the task modules and variance of energy consumption and execution time of task on different resources. It is different from other energy aware approaches because of following aspects:

- It uses JouleSort benchmark to select the resources. JouleSort is a system-level benchmark for energy efficiency that is useful across many types of systems.
- It considers performance along with energy consumption.
- It considers the dependency within the task to prioritize it.
- It conceives the variance of energy and time i.e. the difference between energy consumption and execution time of tasks by most energy efficient and least energy efficient resources.

Simulation results have been shown that Energy-efficient and high performance algorithm reduce the power consumption without degrading the performance.

For future works, it is intended to improve the proposed technique by including more efficient factor for task scheduling. It is also foreseen to perform real tests instead of simulations.

ACKNOWLEDGEMENTS

This research was supported by AICTE under Research Promotion Scheme Ref. No.8023/RID/RPS-151 (pvt)/2011-2012. Authors' addresses: Dr. Inderveer Chana, Computer Science and Engineering Department, Thapar University, Punjab, Email: inderveer@thapar.edu ; Nidhi Jain, Computer Science and Engineering Department, Thapar University, Punjab, Zip Code-147001, Email: nidhijain9151@thapar.edu ;

REFERENCES

[1] Yu, Jia, Rajkumar Buyya, and Kotagiri Ramamohanarao,"Workflow scheduling algorithms for grid computing," Metaheuristics for scheduling in distributed computing environments. Springer Berlin Heidelberg, 2008, pp. 173-214.

[2] J. Kaplan, W. Forrest, N. Kindler, Revolutionizing data center energy efficiency, Tech. Rep. July, McKinsey Company Technical Report, 2008.

[3] Fangpeng Dong, and Selim G. Akl, "Scheduling algorithms for grid computing: State of the art and open problems," in *School of Computing, Queen's University, Kingston, Ontario* ,2006.

[4] CCCC tool online Available at: http://cccc.sourceforge.net/

[5] Coutinho, Fábio, Luís Alfredo V. de Carvalho, and Renato Santana,"A workflow scheduling algorithm for optimizing Energy-Efficient grid resources usage,"*Dependable, Autonomic and Secure Computing (DASC), 2011 IEEE Ninth International Conference on*. IEEE, 2011.

[6] R. Brown et al., "Report to congress on server and data center energy efficiency: Public law 109-431," Lawrence Berkeley National Laboratory, 2007.

[7] Lindberg, Peder, et al. "Comparison and analysis of greedy energy-efficient scheduling algorithms for computational grids." *Energy-Efficient Distributed Computing Systems* (2011), pp. 189-214.

[8] Garg, Saurabh Kumar, and Rajkumar Buyya. "Exploiting heterogeneity in Grid computing for energy-efficient resource allocation." *Proceedings of the 17th International Conference on Advanced Computing and Communications*. 2009.

[9] Wang, Lizhe, et al. "Towards energy aware scheduling for precedence constrained parallel tasks in a cluster with DVFS." *Cluster, Cloud and Grid Computing (CCGrid), 2010 10th IEEE/ACM International Conference on*. IEEE, 2010.

[10] Li, Chunlin, and Layuan Li. "Utility-based scheduling for grid computing under constraints of energy budget and deadline." *Computer Standards & Interfaces*31.6 (2009), pp. 1131-1142.

[11] Orgerie, A-C., Laurent Lefèvre, and J-P. Gelas. "Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems."*Parallel and Distributed Systems, 2008. ICPADS'08. 14th IEEE International Conference on*. IEEE, 2008.(read it)

[12] Da Costa, Georges, et al. "The green-net framework: Energy efficiency in large scale distributed systems." *Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on*. IEEE, 2009

[13] Fernández-Montes, Alejandro, et al. "Smart scheduling for saving energy in grid computing." *Expert Systems with Applications* 39.10 (2012): 9443-9450.

[14] Software Metrics-Best Practices for successful it management pp -73-74, 15-21

[15] Software Engineering: A Practitioner's Approach

[16] Pillai, Padmanabhan, et al. "FAWNSort: Energy-efficient Sorting of 10GB, 100GB, and 1TB."

[17] Rivoire, Suzanne, et al. "JouleSort: a balanced energy-efficiency benchmark."*Proceedings of the 2007 ACM SIGMOD international conference on Management of data*. ACM, 2007

[18] Sort benchmarks online Aviable at: http://sortbenchmark.org/

[19] Beckmann, Andreas, et al. "Energy-efficient sorting using solid state disks."*Sustainable Computing: Informatics and Systems* 1.2 (2011): 151-163.
[20] Agrawal, Nitin, et al. "Design Tradeoffs for SSD Performance." *USENIX Annual Technical Conference*. 2008.
[21] Buyya, Rajkumar, and Manzur Murshed. "Gridsim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing." *Concurrency and computation: practice and experience* 14.13-15 (2002): 1175-1220.
[22] Chen, Yiyu et al,: Managing server energy and operational costs in hosting centers.33(1). In: *ACM SIGMETRICS Performance Evaluation Review*, (2005).
[23] Verma, Akshat,Ahuja, Puneet et al: Power-aware dynamic placement of HPC applications. In: *Proceedings of the 22nd annual international conference on Supercomputing*. ACM,(2008).
[24] Flissi, Areski, Philippe, Merle: A generic deployment framework for grid computing and distributed applications. In: *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*.,Springer Berlin Heidelberg, pp. 1402-1411.(2006)
[25] Measuring Computer Performance: A Practitioner's Guide ,By David J. Lilja
[26] "LOC" [Online]. Available: http://www.projectcodemeter.com/cost_estimation/help/GL_sloc.htm
[27] Abbas HeydarNoori , "Function Point" [Online]. Available: https://cs.uwaterloo.ca/~apidduck/CS846/Seminars/abbas.pdf
[28] Jingcao Hu, and Radu Marculescu. "Energy-and performance-aware mapping for regular NoC architectures," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol 24, pp. 551-562, 2005.

BIOGRAPHY OF AUTHORS

Nidhi Jain obtained his B.Tech with Distinction from TIT&S (MDU), Bhiwani in 2012. Presently she is pursuing M.E. (Software Engineering) from Thapar University, Patiala. Her research area is Grid Computing.

Dr. Inderveer Chana is Ph.D in Computer Science with specialization in Grid Computing and M.E. in Software Engineering from Thapar University and B.E. in Computer Science and Engineering. She joined Thapar University in 1997 as Lecturer and has over fourteen years of experience.