International Journal of Applied Engineering Research ISSN 0973-4562 Volume 12, Number 17 (2017) pp. 6663-6669

Task Scheduling on Virtual Machines using BAT Strategy for Efficient Utilization of Resources in Cloud Environment

T Sunitha Rani

Research Scholar, Bharathiar University, Coimbatore 641 046, India.
Head, Department of Computer Science, M.O.P. Vaishnav College for Women, Chennai 600 034, India.
Orcid ID: 0000-0002-2813-5711

Dr. Shyamala Kannan

Research Supervisor, Bharathiar University, Coimbatore 641 046, India.
Associate Professor, P G & Research Department of Computer Science, Dr.Ambedkar Govt. Arts College, Chennai 600 039, India.

Abstract

Cloud computing is characterised by shared infrastructure, dynamic provisioning, network access and managed metering. It has surfaced as a foremost archetype, utilizing virtualized infrastructure to control several complex servers on various execution environments. Virtual machines (VMs) are considered as the facilitators and processing units in cloud environment. VMs can be instantiated in variable number depending on the needs of the users. Efficient utilization of these VM instances is achieved through effective task scheduling mechanism which also enables even distribution of work load among nodes resulting in improve response time. This paper focuses on optimal utilization of VMs, modelled around directed acyclic graph with topological ordering to schedule task using nature inspired BAT algorithm. This algorithm minimizes the waiting time of a task and idle time of VMs as its convergence rate is high. Simulated results for bat algorithm using MATLAB reveal that the performance of BAT is better than First-In-First-out (FIFO), Particle Swarm Optimization (PSO) and Harmonic Search (HS).

Keywords: Task Scheduling, BAT Algorithm, Cloud Computing, Directed Acyclic Graph, Topological ordering.

INTRODUCTION

Cloud environment is considered as an unparalleled magnificence in the field of information technology that provides a capability to share computing resources efficiently among clients. The ability of cloud to scale horizontally, vertically or diagonally for dynamic allocation of resources enables it, to effectively meet the uncertain quantitative requirements depending on the qualitative needs of users [18][20]. The fundamental building block of cloud computing is virtualization which provides a flexibility to assign or reassign computing resources to applications on-demand in a user-customized milieu [4][7][10]. Virtual Machine Monitor (VMM) or Hypervisor helps in achieving virtualization that improves resource availability, reduces complexity and avoids infrastructure associated perils. Lack of agility to utilize the full potential of VMs has forced to employ BAT algorithm to explicitly propagate, active state among VMs that are likely to be idle most of the time in the course of execution of a given task. Hence, effective task scheduling is considered as one of the main challenges in cloud computing as it enhances resource utilization and minimizes underutilization of nodes. It also helps in achieving even workload at all nodes ensuring improved response time and minimized waiting time of tasks [26]. It significantly improves overall performance and maintains system stability. Reduction in waiting time of tasks helps in improving responsiveness of the VMs/nodes. Improved response time and decreased waiting time results in efficient scheduling and high resource utilization. This paper presents a scheduling technique that best utilizes the resources in cloud. Experimental results using MATLAB prove that this method efficiently utilizes the resources by reducing the idle time of VMs and waiting time of tasks.

REVIEW OF LITERATURE

Cloud computing is a service oriented architecture [24] which needs effective task scheduling to achieve high resource utilization with balanced load of VMs and user satisfaction. Load balancing facilitates increased performance, high security and system stability. A survey on load balancing techniques [8] has accentuated the significance of having VMs with balanced load, as load imbalance reflects badly on system's stability and performance. It has also demystified the necessary metrics involved in balancing the load among VMs in cloud environment. Several nature inspired algorithms [3][5][6][14[17][25] like honey bee, ant colony, particle swarm, cuckoo and fire fly have been reviewed to schedule tasks and balance the load. Priority based approach and make-span (overall task completion time) minimization have improved the average execution time and reduced the waiting time of tasks for virtual machines [12][15][22]. Main factors which affect

scheduling process are communication cost and execution time [16]. Many algorithms have been developed to minimize these factors. Compromised-time-cost scheduling minimizes the mean execution time and cost [11]. Multi-objective genetic algorithm based task scheduling model helps in improving the overall performance of cloud computing and profit of service provider with deadline as specified constraint [9].

The stability of the system depends on the state of processor and strength of an algorithm to reduce response time. Obtaining throughput using stochastic process has been identified as an alternative for the best-fit-scheduling [23]. Maximum resource utilization (CPU, memory and band width) is also achieved with load balancing. In addition to the above mentioned metrics, energy consumption and carbon emission have been exploited to obtain an eco-friendly cloud computing environment [21]. Though these algorithms have been advantageous in certain aspects, BAT-inspired algorithm is better in terms of efficiency, accuracy, success rate and convergence rate [2] when applied to task scheduling.

MATHEMATICAL MODEL

A directed graph $G(V, E)$ with no directed cycles is termed as DAG. Any task T represented as a rooted tree forms a special type of DAG. The number of DAGs on n nodes [1][13] can be obtained from the recurrence equation

$$\mathcal{G}_n = \sum_{\varphi=1}^{n} (-1)^{\varphi-1} \binom{n}{\varphi} 2^{\varphi(n-\varphi)} \mathcal{G}_{n-\varphi}\,, \quad n > 0$$

while $\mathcal{G}_0 = 1$.

One such DAG is represented in Figure1. Each sub-task is a vertex in DAG and the edges exhibit the precedence constraints. A precedence constraint of an edge (v_i, v_j) means v_i precedes v_j. Any DAG is characterized by topological ordering. A topological order of any directed graph $G(V, E)$ is an ordering of its nodes v_k where k=*1, 2, ..., n* such that for every edge (v_i, v_j), $i<j$, means that the task v_i must occur before v_j. Also if any graph G demonstrates a topological order, then G is a DAG [19]. Topological ordering helps in resolving dependencies and maintaining inter dependency among the tasks while processing the tasks. Consider a request as task T which is divided into sub-tasks, T_1 to T_{11} represented as Directed Acyclic Graph (DAG) as in Figure1, for sequential and parallel execution on three different VMs with varying micro instructions per second (MIPs), so as to minimize the total completion time of the task T and maximize the performance of VMs. These sub-tasks are executed on different nodes of a distributed network. The possibility of representing graph G in a topological order as in Figure 2 reveals that G is a DAG.

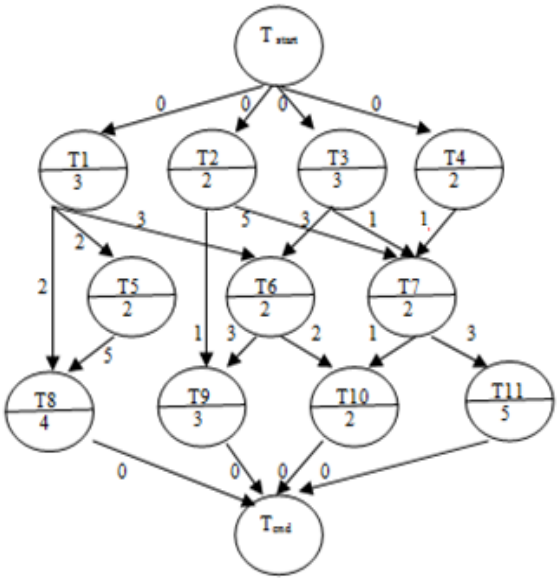

Figure 1: DAG with scheduled sub-tasks and average execution time

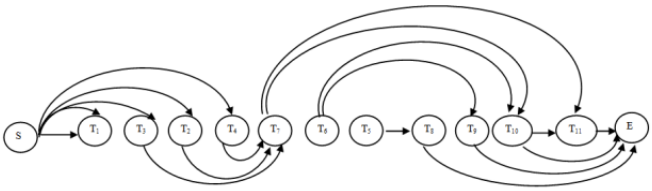

Figure 2: Topological order of sub-tasks T_1 to T_{11}

In Figure 1 the DAG *G (V, E)*, $V=v_1$ to v_n represents the sub-tasks and $E=e_1$ to e_n represents the weights. Sub-tasks T_1, T_2, T_3 *and* T_4 can be executed in parallel. T_5 requires execution of T_1 whereas T_6 is executed after the completion of $T_1$and T_3. T_7 requires execution of T_2, T_3 and T_4. T_8 and T_9 are executed after the execution of T_1, T_5 and T_2, T_6 respectively. Execution of T_{10} takes place after T_6 and T_7 whereas T_{11} happens after T_7.

Sub-Task Analysis

A set of 11 sub-tasks, each with variable lengths and three different virtual machines VM_1, VM_2 and VM_3 with a speed of 304 MIPs, 274 MIPs and 565 MIPs respectively have been considered arbitrarily for execution. Execution time of each sub-task on three virtual machines is evaluated as the ratio of length of the sub-task T_j to MIPs of VM_i and recorded in Table 1.

$$Execution\ time = \frac{\text{Length of } T_j}{\text{MIPs of } VM_i} \text{ where } i = 1, 2, 3 \text{ and } j = 1, 2, \ldots\ldots, 11$$

Graph in Figure 3 represents the coded values for the execution time and their average with a class interval of

length 30 from 0 to 210 for the values in Table 1.

Table 1: Length and Execution time of each sub-task on VM_i

Sub-Task	Length	Execution Time		
		VM_1 340 MIPs	VM_2 274 MIPs	VM_3 565 MIPs
T_1	23132	76.09	132.94	42.84
T_2	12469	41.02	71.66	23.09
T_3	25924	85.28	148.99	48.01
T_4	15613	51.36	89.73	28.91
T_5	13258	43.61	76.20	24.55
T_6	12443	40.93	71.51	23.04
T_7	14744	48.50	84.74	27.30
T_8	30652	100.83	176.16	56.76
T_9	25553	84.06	146.86	47.32
T_{10}	13835	45.51	79.51	25.62
T_{11}	36500	120.07	209.77	67.59

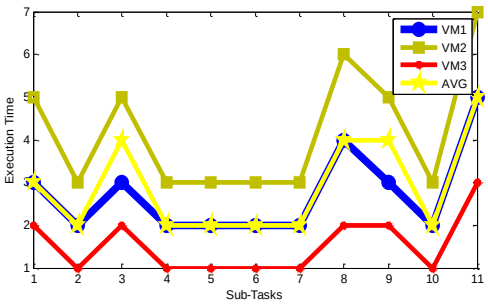

Figure 3: Coded Values for Execution Time and Average of each sub-task on all VMs

Scheduling with FIFO

Sub-Tasks schedule is formed using the FIFO algorithm. The order of execution of sub-tasks on three virtual machines is shown in Table 2 and the total elapsed time for this schedule resulted in 19ms as portrayed in Figure 4.

Table 2: FIFO schedule on VMs

VM	Schedule
VM_1	T_1, T_4, T_7, T_{10}
VM_2	T_2, T_5, T_8, T_{11}
VM_3	T_3, T_6, T_9

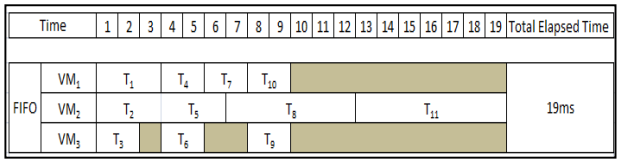

Figure 4: Total elapsed time with FIFO schedule

BAT Algorithm

BAT behaviour inspired optimization algorithm is a technique with a random probability distribution which may be analysed statistically. It is modelled on the way the BATs approach their prey which is analogous to identifying less loaded machines. Uncertain requirements in terms of qualitative concepts and their quantitative representation reflect randomness, fuzziness, and the relationship between both. The waiting time of a task is minimized as the convergence rate is increased by using the BAT algorithm. It is revealed from the simulation that good function optimization is achieved with the BAT algorithm. Characteristics like population information, communicating mechanism and random flight of BATs are used for exploration and exploitation to obtain improved rate of convergence and precision. Successful preying of BAT is achieved with echo location. BAT obtains its prey's current position, distance and direction from echolocation. The velocity (φ) with which the BATs fly at a particular time, their frequency (ψ) and position (ξ) are obtained using equations (1), (2) and (3). Random vector (τ) whose value ranges from 0 to 1is used to calculate the frequency.

$$\psi_i = \psi_{min} + (\psi_{min} - \psi_{max}) * \tau \qquad (1)$$

$$\varphi_i^{j+1} = \varphi_i^j + \left(\xi_i^j - \xi_b\right) * \psi_i \qquad (2)$$

$$\xi_i^{j+1} = \xi_i^j + \varphi_i^{j+1} \qquad (3)$$

This paper is modelled using BAT algorithm with adjustable parameters to schedule the sub-tasks as in Figure 5.

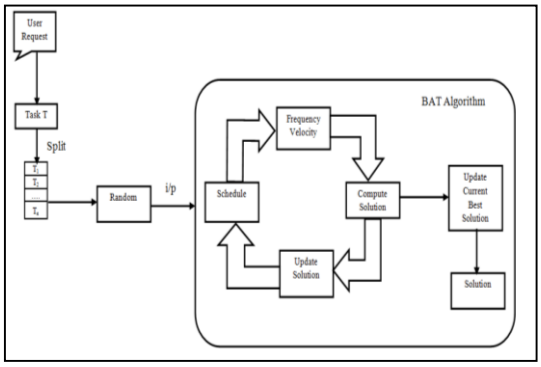

Figure 5: Proposed System Architecture

This paper aims at emphasizing the use of BAT-inspired task scheduling strategy for efficient utilization of resources that results in even work load. A balanced work load assures that every VM does more or less the same amount of work at any instant.

Algorithm: 1 – BAT algorithm to obtain schedule

***Parameters**: n- Size of the population, l- Loudness, p- Pulse rate, d- dimension.*

***Function** τ=Fun(u)*

*τ =30+sum(u.^2-(10*cos(2*(22/7)*u)))*

// Rastrigin's function is chosen arbitrarily.

*1: **Initialise** n, l, p, d, ψ_{min}= -10, ψ_{max}=10;*

*2: **Initialise** Lb(1,j)=-3, Ub(1,j)=6;*

*3: **Evaluate** Fitness(i)=Fun(ξ (i, j));*

*4: **Display** best(k, j)=ξ (k, j);*

*5: **Iterate***

*6: **Evaluate** ψ(i)= ψ_{min}+(ψ_{min}- ψ_{max})*rand;*

*7: **Evaluate** φ(i, j)= φ(i, j)+(ξ(i, j)-best(k, j))* ψ(i);*

*8: **Evaluate** ζ(i, j)= ξ (i, j)+ φ(i, j);*

*9: **If** rand > p*

*10: **Evaluate** ζ(i, j)=best(k, j)+0.001*rand(1, j);*

*11: **end;***

*12: **If** (Fnew<=Fitness(i)) & (rand<l)*

*13: **Update** ξ (i, j);*

*14: **end;***

*15: **Update** best if the current solution is an improved one.*

*16: **end;***

Algorithm: 2 – Pseudo code of proposed algorithm

*1: **for each** task T consider User's Requests*

*2: **Split** T into Sub_tasks (T_i)*

*3: **Call** Bat_algo(Sub_task)*

*4: **Iterate** (Repeat Bat_ algo until Best_sol)*

*5: **end;***

*6: Apply C**olumn Maxima** to Best_sol*

7: Schedule thus obtained is the input to VMs.

All sub-tasks T_1 *to* T_{11} are scheduled on three VMs using **BAT algorithm** and **column maxima** technique in such a way that the total execution time is minimized and the load is balanced on all VMs. When the parameters *l* and *p* are set to 1 and 0 respectively, BAT reduces to PSO and when both these parameters are set to 0.7, BAT reduces to Harmony Search [25]. Order of Execution of tasks on virtual machines using BAT, PSO and HS is displayed in Table 3 and the total elapsed time for each of these schedules is portrayed in Figure 6.

Table 3: Sub-task Execution Schedules on VMs

VM	BAT Schedule	PSO Schedule	HS Schedule
VM_1	T_1, T_6, T_9, T_{11}	T_3, T_6, T_8, T_{10}	T_3, T_4, T_7, T_{11}
VM_2	T_3, T_4, T_7	T_2, T_5, T_9, T_{11}	T_1, T_6, T_8, T_{10}
VM_3	T_2, T_5, T_8, T_{10}	T_1, T_4, T_7	T_2, T_5, T_9

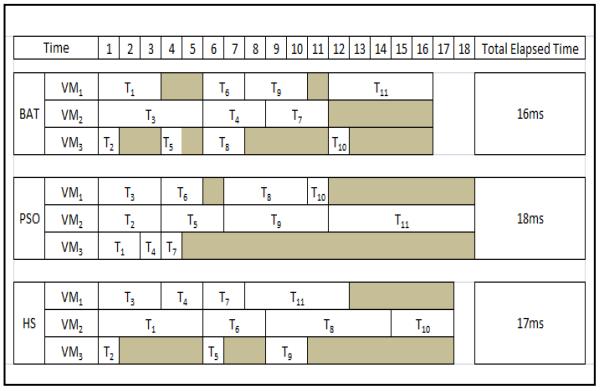

Figure 6: Total elapsed time with various schedules

RESULTS AND INTERPRETATION

Input to the VMs are the *n* sub-tasks obtained from the DAG. The schedule for each VM_i, i=1 to *m*, is obtained by applying BAT algorithm and column maxima. With 11 sub-tasks and three VMs, schedule obtained for VM_1, VM_2 and VM_3 using FIFO, PSO, HS and BAT resulted in a total elapsed time of 19 ms, 18ms, 17ms and 16 ms respectively. Figure 7 reveals that the execution time on VM2 is very high when compared to VM1 and VM3 using FIFO, PSO and HS whereas the total execution time using BAT is moderate on all the machines showing that the load is balanced. Figures 8.1, 8.2, 8.3 and 8.4 reveal the mean and standard deviation for the schedules obtained with each of these algorithms. Figure 9 depicts that the load is not evenly distributed by the use of FIFO, PSO and HS as the idle time of VM1 and VM3 is very high when compared to BAT which distributes the load in a way much

better than FIFO, PSO and HS. Hence it is clear that the idle time of VMs is reduced drastically with BAT.

Table 6: Total Execution Time and Idle Time on VMs with various Strategies

Strategy	Execution Time			Idle Time		
	VM_1	VM_2	VM_3	VM_1	VM_2	VM_3
FIFO	221.46	533.79	118.37	312.33	0	415.42
PSO	272.55	504.49	99.05	93.74	0	314.31
HS	305.21	460.12	94.96	154.91	0	365.16
BAT	321.14	323.45	130.03	2.31	0	193.42

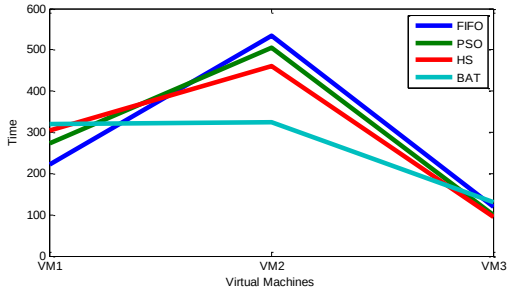

Figure 7: Graph representing total execution time of sub-tasks

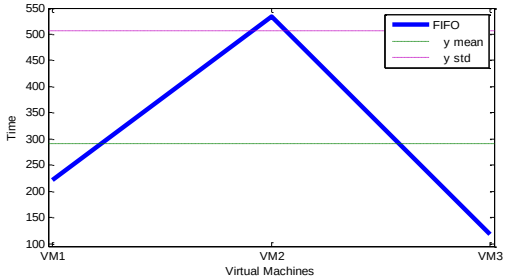

Figure 8.1: FIFO y-mean and y-standard deviation

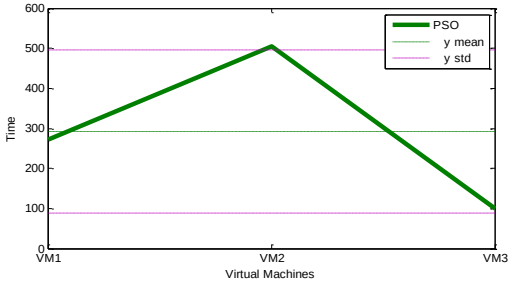

Figure 8.2: PSO y-mean and y-standard deviation

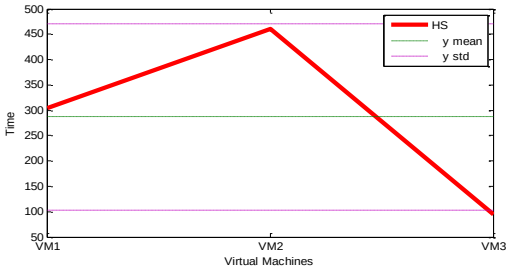

Figure 8.3: HS y-mean and y-standard deviation

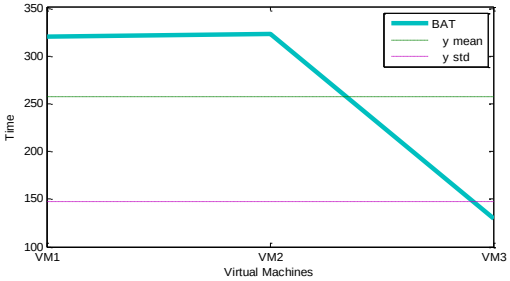

Figure 8.4: BAT y-mean and y-standard deviation

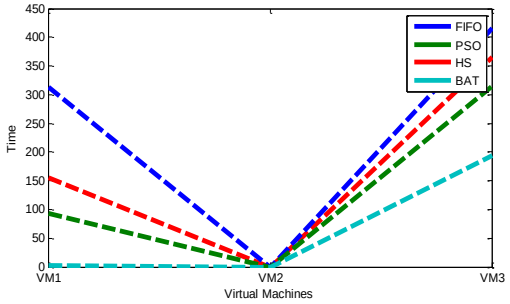

Figure 9: Graph representing idle time of VMS using FIFO, PSO, HS and BAT

CONCLUSION

In this paper a new task scheduling strategy based on the behaviour of BAT has been proposed for cloud environment. This strategy not only increases the resource utilization but also shares the load evenly among the VMs by reducing the idle time. Simulated results using MATLAB reveal that BAT inspired algorithm reduces the response time of tasks as its convergence rate is very high. Proposed framework when compared with existing standard algorithms provided improved results in terms of time taken to obtain an optimum value for arriving at a schedule that well balances the load.

REFERENCES

[1] Alexandra Olteanu, Andreea marin, ‘Generation and Evaluation of Scheduling DAGs: How to provide

similar evaluation conditions', Computer Science Master Research, Vol. 1, No. 1, (2011)

[2] Ali M Alakeel: A Guide to Dynamic Load Balancing in Distributed Computer Systems. International Journal of Computer Science and Network Security, Vol. 10, No.6 (2010)

[3] Anju Baby: Load balancing in Cloud Computing Environment using PSO Algorithm. International Journal for Research in Applied Science and Engineering Technology, Vol. 2 Issue IV, ISSN: 2321-9653, (2014)

[4] Belén Cruz Zapata, José Luis Fernández-Alemán and Ambrosio Toval 'Security in Cloud Computing: a Mapping Study, Computer Science and Information Systems, 12(1):161-184, (2015)

[5] DervisKaraboga and BahriyeBasturk. 'A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm', Journal of Global Optimization, Vol. 39, pp. 459-471, (2007)

[6] Dinesh Babu L D, P Venkata Krishna: Honey Bee Behaviour inspired Load Balancing of Tasks in Cloud Computing Environment. Applied Soft Computing 13, 2292-2303, (2013)

[7] Hung, Q.Y., Huang, T.L. 'An Optimistic Job Scheduling Strategy based on QoS for Cloud Computing', IEEE International Conference in Intelligent Computing and Integrated Systems (ICISS), Guilin, pp.673-675, (2010)

[8] Iztok Fister, Dusan Xin-She Yang: A Hybrid BAT Algorithm. Original Scientific Paper. Elektrotehniski Vestnik 80(1-2): 1-7, (2013)

[9] Jing Liu, Xing-Guo Luo, Xing-Ming Zhang, Fan Zhang. 'Job Scheduling Model for Cloud Computing Based on Multi-Objective Genetic Algorithm', International Journal of Computer Science, (2013)

[10] M. Kandan, R. Manimegalai, 'Strategies for Resource Allocation in Cloud Computing – A review', International Journal of Applied Engineering Research (IJAER), pp. 1-10, Volume 10, Number 76(2015) Special Issue.

[11] Ke, L., Hai, J., Jinjun, C., Xiao, L., Dong, Y., and Yun, Y. 'A Compromised-Time-Cost Scheduling Algorithm in SwinDeW-C for Instance-Intensive Cost-Constrained Workflows on Cloud Computing Platform', International Journal of High Performance Computing Applications, pp.1-16, (2010)

[12] Kousik Dasgupta, Brototi Mandal, Paramartha Dutta, Jyotsna Kumar Mondal, Santanu Dam: A Genetic Algorithm (GA) based Laod Balancing Strategy for Cloud Computing. Procedia Technology, 340-342, Science Direct 10, (2013)

[13] Maciej Dziemiancauk: Counting Bipartite, K-Colored and Directed Acyclic Multi Graphs through F- Nominal Coefficient

[14] Narmartha Swarnkar, Atesh Kumar Singh, Dr. R. Shankar: A Survey of Load Balancing Techniques in Cloud Computing. International Journal of Engineering Research & Technology, ISSN: 2278-0181, Vol. 2 Issue 8, (2013)

[15] Nidhi Jain Kansal, Inderveer Chana: Cloud Balancing Techniques: A Step Towards Green Computing. International Journal of Computer Science. Vol. 9, Issue 1, (2012)

[16] O.M. Elzeki, M.Z. Rashad, M.A. Elsoud. 'Overview of Scheduling Tasks in Distributed Computing Systems', International Journal of Soft Computing and Engineering , Vol-2, Issue-3, (2012)

[17] Paulin Florence A, Shanthi V: A Load Balancing Model Using Fire Fly Algorithm in Cloud Computing. Journal of Computer Science 10(7): 1156-1165, ISSN: 1549-3636, (2014)

[18] Rajkumar Buyyaa, Chee Shin Yeo, Srikumar Venugopal, James Broberg, Ivona Brandic. 'Cloud Computing and Emerging IT platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility,' Journal Future Generation Computer Systems, Vol. 25 no. 6, (2009) ELSEVIER

[19] Robert Sedgewick and Kevin Wayne: Algorithms, 4th Edition. algs4.cs.princeton.edu

[20] Rodrigo N. Calheiros, Rajiv Ranjan, RajakumarBuyya. 'Virtual Machine Provisioning Based on Analytical Performance and QoS in Cloud Computing Environments', International Conference on Parallel Processing, IEEE Computer Society, 0190-3918/11, DOI 10.1109/ICPP.2011.17, (2011)

[21] Saibal K Pal, C S Rai, Amrit Pal Singh: Comparative Study of Firefly Algorithm and Particle Swarm Optimization for Noisy Non-Linear Optimization Problems. I.J. Intelligent Systems and Applicatios, 10, 50-57, (2012)

[22] Shamsollah Ghanbari, Mohamed Othman. 'A Priority based Job Scheduling Algorithm in Cloud Computing', International Conference on Advances Science and Contemporary Engineering, (ICASCE 2012), pp:778 – 785, (2012)

[23] Siva Theja Maguluri R Srikant, Lei Ying: Stochastic Models of Load Balancing and Scheduling in Cloud Computing Clusters. Proceedings IEEE Infocom, (2012)

[24] Taerim Lee, Hun Kim, Kyung-Hyune Rhee, and Sang Uk Shin, 'Design and Implementation of E-Discovery as a Service based on Cloud Computing', ComSIS Vol. 10, No. 2, Special Issue, (2013)

[25] Xin-She Yang: A New Metaheuristic BAT-Inspired Algorithm. arXiv: 1004.417ov1, (2010)

[26] Youchan Zhu, Huili Liang. 'Research for the virtual machine-oriented cloud resource scheduling algorithm', 6th International Conference on Information Management, Innovation Management and Industrial Engineering, pp:133-136, (2013)