A Workload Balanced Approach for Resource Scheduling in Cloud Computing

Conference Paper · August 2015

DOI: 10.1109/IC3.2015.7346649

1 author:

Ritu Kapur
Centre for Development of Advanced Computing
32 PUBLICATIONS **122** CITATIONS

A Workload Balanced Approach for Resource Scheduling in Cloud Computing

Ritu Kapur
Department of Computer Science & Engineering
National Institute of Technical Teachers Training and Research
Chandigarh, India
ritukapur1591@gmail.com

***Abstract*—The era of cloud based multimedia applications has lead to a huge increase in the no. of requests on cloud. The increased no. of requests on cloud leads to an increased workload, making workload balancing an important QoS Parameter. Workload Balancing also leads to a judicious use of resources like electricity etc. and thus promotes the concept of Green IT. The paper presents a new Load Balanced Resource Scheduling Algorithm (LBRS) which performs the resource scheduling in a balanced banner. The simulations and results demonstrated in the paper prove that the LBRS algorithm outperforms an existing approach.**

Keywords—Cloud Computing; Load Balancing; QoS; Reliability; Throughput; Response Time; Energy Efficiency; Green IT

I. Introduction

Computing may be defined as a process of utilizing computer technology to complete a task. Cloud computing is a computing paradigm shift where computing is moved away from personal computers or an individual application server to a "cloud" of computers. It is a service that charges based on the amount of computing resources that we use, also known as the pay-per use service or pay-as-you-go service. The name comes from the use of a cloud-shaped symbol as an abstraction for the complex infrastructure it contains in system diagrams.

Cloud Computing has emerged as a popular area of research and development in the past few years. Although no unique definition of Cloud Computing exists, many standard organizations continue to give various definitions for it [1]. According to National Institute of Standards and Technology (NIST) definition, "Cloud computing is defined as a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction".

Resource Scheduling is a process of allocating resources to various requests issued by the user at a particular time. Such process has to respect constraints given by the requests and the cloud. These constraints need to be specified by the cloud customer and form the part of the QoS agreement made with the cloud service provider. These constraints may depend upon a number of QoS parameters. Some of the QoS parameters are:-

- **Resource Cost:** The total cost to be paid by the cloud customer in order to access the resources for a certain period of time is called resource cost.
- **Response Time:** The time spent by a request in the waiting queue till it gets the first time to use the CPU.
- **CPU Utilization:** The total percentage of time for which CPU was utilized or used i.e. was not idle.
- **Throughput:** Total no. of tasks executed (or requests served) per unit time.
- **Waiting Time:** The total time spent by the request waiting in the ready queue after the first response from CPU.
- **Turnaround Time:** The total time taken by a request to get completely served, including its response time, waiting time and service time.
- **Fairness:** The principle that states that every request should get equal share of CPU time.

The main goal is to maximize the CPU Utilization, maximize the Throughput, minimize the Response Time, minimize the Waiting time, minimize the Turnaround Time, minimize the Resource Cost and obey the Fairness principle.

II. Ease of Use

Zhu et al. [1] explains the basics of multimedia applications on cloud computing by highlighting its various characteristics, the importance of QoS, illustrating the architecture of various types of distributed computing and the importance of Load Balancing and User-level Parallelization and Task Level Parallelization between various servers.

Wu et al. [2] propose a dynamic cloud resource allocation algorithm which uses dynamically created network information and capacities as input. The algorithm effectively provides Video on Demand with low costs to use the cloud resources.

Surveys in (**[4], [5]-[6]**) provide a brief overview of basic terminology and concepts used in cloud and the concept of Resource Allocation and Load Balancing in Cloud. Surveys in **[4]-[6]** give a detailed description of notation, representation,

978-1-4673-7948-9/15/$31.00 ©2015 IEEE

classification schemes and models on Resource Scheduling Problem.

Xin Sun et al. [7] also found the resource scheduling problem reducible to NP- Hard problem as the authors in [15] and thus the authors in [7] model the problem as a Multi- Dimensional Combinatorial Optimization problem with Constraint Satisfaction (**MCO-CS**). The authors in [7] propose a heuristic algorithm called Multi-Attribute Decision based Integrated Resource Scheduling (**MADIRS**) providing a sub-optimal solution. Similarly, **M. Bendraouche et al. [8]** proposed a scenario of scheduling jobs known as **scheduling with agreements** (**SWA**) and proved that it is reducible to NP Hard problem set. With the help of various theorems and corollaries the authors in [8] successfully proved that SWA and the Resource Scheduling problem both are polynomial equivalents on the based on their asymptotic complexity.

R. Santosh et al. [9] presents a non-preemptive scenario to schedule real time systems which provides the solution of aborting the tasks when the deadline is missed. **R. Santosh et al. [10]** on the other hand migrates the task which misses its deadline to another VM so that the computation effort done so far doesn't get wasted.

A. Aggarwal et al. [3] propose a Generalized Priority algorithm to optimize the performance of tasks and compare the performance with the FCFS execution scenario and Round Robin Execution scenario. Authors in [3] propose the task scheduling scenario with load balancing thus maximizing the resource utilization and the throughput of the system.

Authors in [47]-[54] focus on the aspect of QoS. **R. Jeyarani et al. [11], [12]** propose a two-level scheduler; First, a meta-scheduler based on Quality of Service desired by the user and thus known to be user-centric and second, a VM scheduler based on the Backfill Strategy based on backfilling algorithm [13], thus outperforming the basic CloudSim Toolkit [14].

W. Dong et al. [19] proposed a QoS-based model for monitoring the resource availability to schedule various requests in the cloud. The authors in **[20]** propose a different version of branch and bound method for the resource-constrained project scheduling.

G. Guo-ning et al. [21] propose a genetic simulated annealing algorithm for task scheduling based on various QoS requirements to improve the performance and maximize resource utilization. In the process of annealing, some inferior solutions are also accepted at random along with the found optimal solution, which increases the probability of finding a global optimal solution, thus guaranteeing the convergence of the algorithm.

M. Dorigo et al. [22], [23] present the basic **Ant Colony Optimization (ACO)** algorithm. **K. Li et al. [24]** embeds the load balancing parameter into it and thus presents a Load Balanced ACO i.e. **LBACO** algorithm. According to the authors in [22], [23] ants have a general property of locating their path to food. In this phenomenon of locating their food, ants keep on depositing a chemical substance known as pheromone. This pheromone in turn helps new ants on their path to find the shortest route to the food. The path having largest amount of pheromone represents the shortest route. So this process thus leads the new ants to their location in minimum time through the shortest path. The authors in [24] show that the proposed algorithm outperforms the basic ACO and FCFS method.

Two important problems proposed by **Xiaoming Nan et al. [15]** are: to minimize the *response time* and minimize the *resource cost.* **Xiaoming Nan et al. [15]** categorize resources on the basis of two pricing schemes: the Reservation Scheme and the On-Demand Scheme; Charges in the Reservation Scheme being lower than the latter. The authors in [15] thus try to optimize the cost by selecting the best type and no. of resources depending on the basis of above price schemes. Also the author tries to minimize the response time by proposing an Optimal Analytical solution for it. The Resource Cost Minimization problem is a NP-Hard problem and thus a greedy algorithm is proposed for it which gives a close to optimal solution.

Xiaoming Nan et al. [16] categorize various services in different QoS classes. The incoming requests are then processed on the basis of these QoS classes. The scheduling of these requests is done in the FCFS and priority method with the conclusion of the priority method outperforming the FCFS.

Xiaoming Nan et al. [17, 18] divide the cloud service into three consecutive phases: *schedule, computation* and *transmission.* Improper resource assignment in the phases will result in resource wastage and decreased QoE/QoS. Authors in [17] proposed the concept of optimizing the resource allocation based on the concept of single class service case and the multiple-class service case in the queuing model. Authors in [18] further refine the concept studied in [17] by embedding priority service scheme in the basic queuing model studied in [17]. In each case authors in [17] and [18] formulate and solve the resource allocation optimization problems to minimize the mean response time and minimize the resource cost, respectively.

This paper presents a new **Load Balanced Resource Scheduling (LBRS) algorithm,** which when compared with the algorithm in [15], outperforms it. The Simulations demonstrated prove the above fact. The CERS algorithm considers load balancing as an important QoS parameter, performs a check for its necessity and if required (only when requests are quite large or load gets increased with passage of time), does the load balancing and optimizes the performance as well as the overall resource cost.

J. James et al. [25] present a similar work in which certain weights are assigned to various servers on the basis of their serving capability or configuration and the requests are then scheduled onto these servers in accordance to the assigned weights, with the server with larger weight assigned getting more requests scheduled to be served.

Ritu Kapur [26] present a review on various Nature Inspired Algorithms like Ant System (AS), ACO, PSO, Firefly

Algorithm (FA), Artificial Bee Colony (ABC) algorithm and a comparison between them on the basis of various parameters like their inspiring behavior, their inspiring phenomena and their application areas etc.

Ritu Kapur [27] states that meta-heuristic techniques like Genetic Algorithm (GA) and Simulated Annealing based GA algorithm etc. can also be employed for efficient load balanced resource scheduling and presents a review of the same.

III. PROPOSED WORK

Resources are categorized on the basis of two pricing schemes: the Reservation Scheme and the On-Demand Scheme as [15]. These resources are further classified into three classes with the Virtual Machine (VM) instances having different resource costs and different service times respectively [15]. The research work presented in this paper considers the importance of load balancing in the scheduling process and presents a new load balanced resource scheduling (LBRS) algorithm.

A. Objectives of the proposed approach

- To design a Load Balanced Scheduler
- To categorize the resources into two basic types: Reservation Based and On-Demand Based as in [15]
- To implement the existing approach proposed by [15] in PHP environment
- To compare the performance of the proposed approach with the existing approach [15] on the basis of Resource Cost, both calculated in PHP environment

B. Steps in the Methodology

The proposed approach performs the following steps:

1. Compute the cost per service rate of each reserved Class i VM instance, q_i, and arrange it in ascending order
2. Schedule the reserved VMs on the basis of order computed in step 1
3. If all reserved VMs are consumed, calculate the no. of requests left to be served
4. Calculate the required no. of on-demand resources from the respective available classes n the inverse proportion of the ratio of their respective service rates and cost per VM
5. Optimize the ratio obtained in previous step
6. Schedule the requests on the basis of above ratio
7. Repeat the steps 3 to 6 till all requests are served

C. Methodology Flowchart of LBRS Algorithm

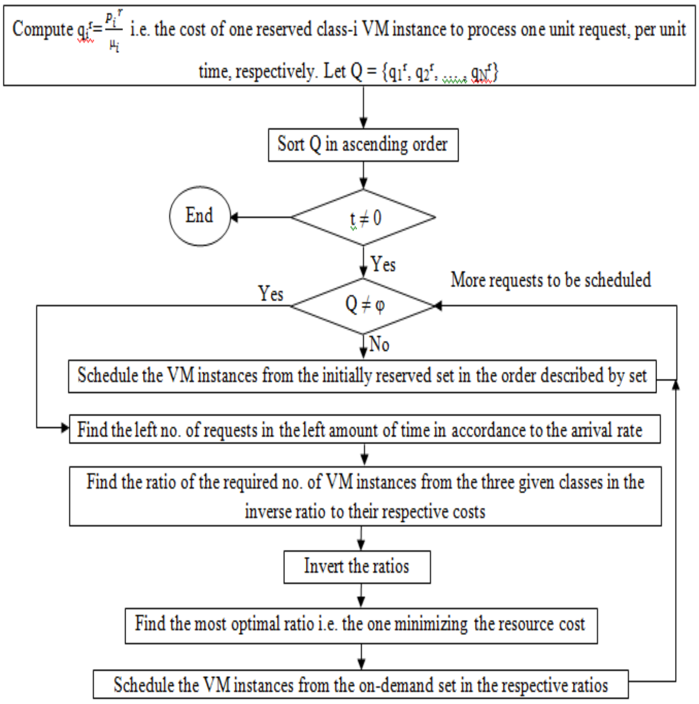

Compute $q_i^r = \frac{p_i^r}{\mu_i}$ i.e. the cost of one reserved class-i VM instance to process one unit request, per unit time, respectively. Let $Q = \{q_1^r, q_2^r, \ldots, q_N^r\}$
Sort Q in ascending order
End
$t \neq 0$
Yes
Yes
More requests to be scheduled
$Q \neq \varphi$
No
Schedule the VM instances from the initially reserved set in the order described by set
Find the left no. of requests in the left amount of time in accordance to the arrival rate
Find the ratio of the required no. of VM instances from the three given classes in the inverse ratio to their respective costs
Invert the ratios
Find the most optimal ratio i.e. the one minimizing the resource cost
Schedule the VM instances from the on-demand set in the respective ratios

Fig. 1 Methodology Flowchart of LBRS Algorithm

D. Tools and Platform

PHP (Hypertext Pre-Processor) platform has been used for implementation. WAMP (Windows Apache MySQL PHP) Server is used as a local host to run the PHP script. Besides PHP, HTML (Hypertext Markup Language) and CSS (Cascading Style Sheets) is used to design the basic web pages for the demonstration part. The reason why we decided to use PHP is because:

- PHP is an open source and is freely available
- PHP is a server side scripting language which is used to make websites
- It is platform independent
- We can run PHP on any operating system like Windows, Linux etc
- Its execution is faster than other competing technologies like Java and .NET (Network Enabled Technology)
- Web hosting for PHP is free

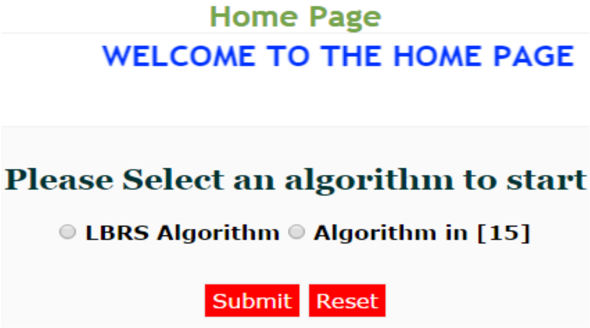
IV. SIMULATION AND RESULTS

The simulations are done in PHP environment. HTML and CSS is used for the designing purpose and PHP is used as the coding language. The price rates and configurations (e.g. service time) of various VMs considered in simulations are same as defined by Amazon EC2 and taken by [15]. The price rates and other parameters are:

Reservation cost of various VM instances, $\mathbf{P^r}$= {0.05$/h, 0.20$/h, 0.40$/h},
On-Demand cost of various VM instances, $\mathbf{P^d}$= {0.085$/h, 0.34$/h, 0.68$/h},
Initially Reserved VM instances, $\mathbf{K^{ini}}$= {60, 30, 20},
Service Time of various VM instances, **μ**, (mew) = {25 requests/s, 97 requests/s, 185 requests/s},
For the purpose of simulation, two interfaces are designed; the LBRS algorithm interface and the existing approach in [15].

A. Home Page

The home page lets you choose an algorithm to perform the execution.



Home Page

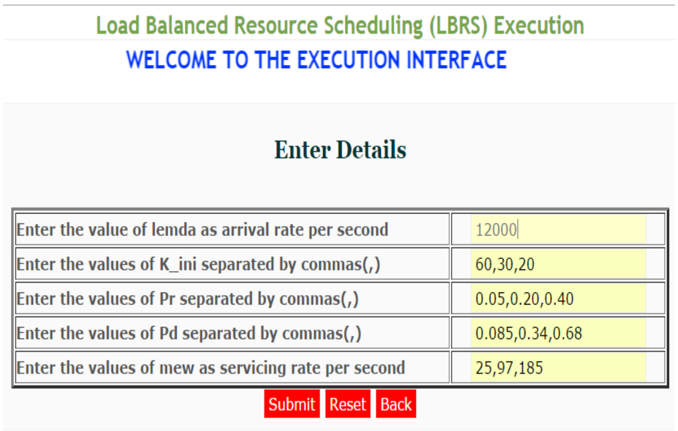
WELCOME TO THE HOME PAGE

Please Select an algorithm to start

Fig. 2 Home Page of the interface

B. LBRS Algorithm Execution Interface

Load Balanced Resource Scheduling Algorithm is shown below. The entire algorithm is coded PHP and designed with HTML and CSS. Various input parameters as shown below are entered into the interface and the corresponding resource cost is computed on the basis of the LBRS algorithm and the results are compared by computing the result for the same parameters from the second interface i.e. the interface for the Algorithm in [15].



Load Balanced Resource Scheduling (LBRS) Execution

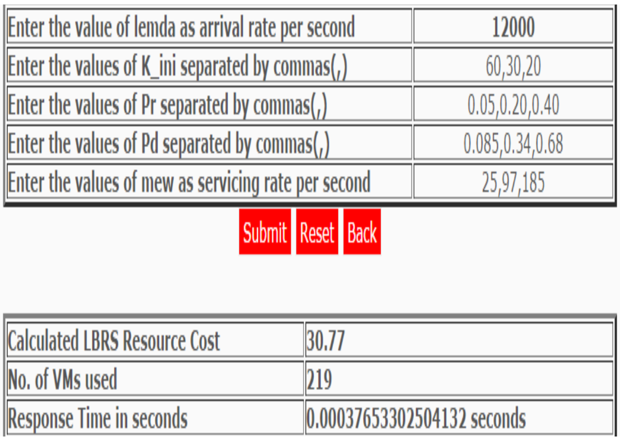
WELCOME TO THE EXECUTION INTERFACE

Enter Details

Enter the value of lemda as arrival rate per second	12000
Enter the values of K_ini separated by commas(,)	60,30,20
Enter the values of Pr separated by commas(,)	0.05,0.20,0.40
Enter the values of Pd separated by commas(,)	0.085,0.34,0.68
Enter the values of mew as servicing rate per second	25,97,185

Fig. 3 LBRS Execution Interface

After entering the details and pressing submit, the overall resource cost by LBRS algorithm is displayed as follows:



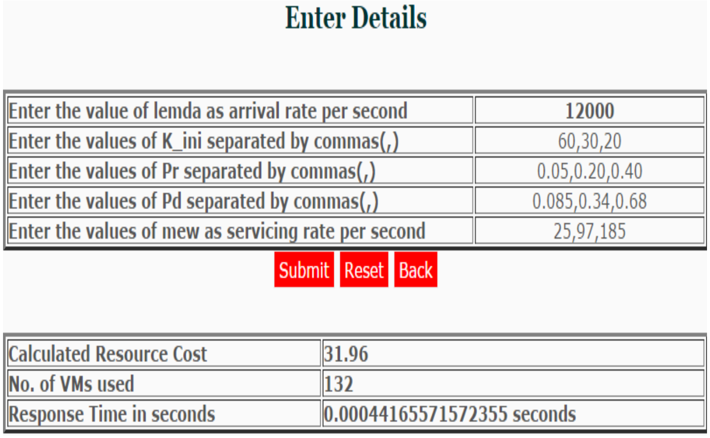
Enter the value of lemda as arrival rate per second	12000
Enter the values of K_ini separated by commas(,)	60,30,20
Enter the values of Pr separated by commas(,)	0.05,0.20,0.40
Enter the values of Pd separated by commas(,)	0.085,0.34,0.68
Enter the values of mew as servicing rate per second	25,97,185

Calculated LBRS Resource Cost	30.77
No. of VMs used	219
Response Time in seconds	0.00037653302504132 seconds

Fig. 4 LBRS Results Screenshot

C. Algorithm in [15]

On clicking on back, home page is displayed again from where the algorithm as proposed by [15] can be selected for comparison. On clicking it, the interface is displayed. Entering the same values and clicking on submit, the overall resource cost is displayed as follows:



Enter Details

Enter the value of lemda as arrival rate per second	12000
Enter the values of K_ini separated by commas(,)	60,30,20
Enter the values of Pr separated by commas(,)	0.05,0.20,0.40
Enter the values of Pd separated by commas(,)	0.085,0.34,0.68
Enter the values of mew as servicing rate per second	25,97,185

Calculated Resource Cost	31.96
No. of VMs used	132
Response Time in seconds	0.00044165571572355 seconds

Fig. 5 Algorithm [15] Execution Interface

In the above manner, results are calculated corresponding to a number of different values of arrival rate, lemda (λ) and are then analyzed.

V. ANALYSIS AND CONCLUSION

This section compares and analyzes the results obtained for resource cost corresponding to different values of time. The results so obtained are listed in the table below.

Table I shown on the next page compares the resource cost obtained by varying the arrival rate of the requests.

TABLE III. RESOURCE COST (IN $) OBTAINED BY THE RESPECTIVE ALGORITHM IN THE RESPECTIVE TIME (IN SECONDS)

Arrival Rate (Requests/second)	LBRS Algorithm's Resource Cost ($)	Algorithm in [15]'s Resource Cost ($)
1000	2	2
2000	4.2	4.2
3000	6.2	6.2
4000	8.2	8.2
5000	10.6	10.6
6000	12.6	12.6
7000	14.6	14.6
8000	17	17
9000	20.57	20.4
10000	23.63	23.46
11000	27.37	27.88
12000	30.77	31.96
13000	34.51	33.66
14000	37.57	38.76
15000	41.31	42.84
16000	44.37	43.86
17000	48.11	50.32
18000	51.85	53.72
19000	54.91	54.06
20000	58.65	61.2

The graph plotted corresponding to the listed values in TABLE I. is shown in figure 6.

A. *Defining the Graph*

In **figure 6**, x-axis represents the Arrival Rate of requests, λ, measured in requests/second and y-axis represents the total Resource Cost, measured in $.

B. *General Observation*

The graph shown in **figure 6** is a combination of outputs obtained in the twenty cases, listed in Table I above, and shows the plots of the two algorithms, LBRS and the algorithm proposed by [15]. As seen from the Table I, the resource cost obtained in case of LBRS is lower as compared to that of the algorithm in [15] for almost all the cases listed above. Also as shown by figure 6, after λ=9000 requests/second the LBRS algorithm clearly provides a lower bound for the algorithm in [15].

As represented from the graph in figure 6, both the graphs have same resource cost initially for some period of time, after which, the LBRS algorithm provides lower resource cost than the algorithm proposed by [15].

Also, as shown by figure 6, LBRS algorithm has a constant positive slope throughout the observation whereas, the algorithm in [15] initially has a constant slope, same value as that for LBRS algorithm, which then changes to a graph with a variable slope changing periodically, giving rise to a periodic curve.

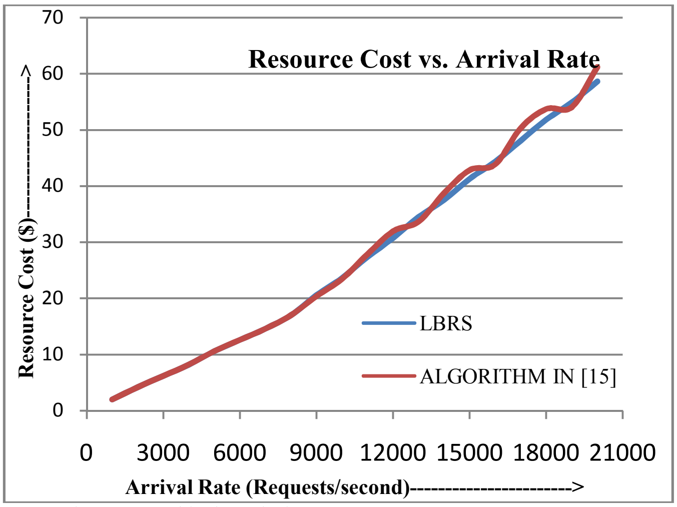


Fig. 6 Graphical Analysis

C. *Critical Observation*

For $0<=\lambda<=8000$, it is clearly visible that both the graphs have a negligible slope. For $9000<=\lambda<=20000$, it can be concluded from the values listed in table I that the LBRS algorithm produces comparatively lower resource costs as compared to the algorithm in [15]. From the graph in figure 6, it can be concluded that the graph for LBRS algorithm is linear throughout the observation period whereas, the algorithm in [15] changes from linear to a periodic curve.

During this stated time period i.e. $0<=\lambda<=8000$, the graphs appear to be linear and continue to ascend with a small equal slope value, with both the graphs appearing to be merged into each other. From $9000<=\lambda<=20000$, it can be seen that the LBRS algorithm has a slightly lower slope than the algorithm in [15] and is linear in nature. For $9000<=\lambda<=20000$, algorithm in [15] changes to a graph which is periodic in nature, with the LBRS algorithm providing a lower bound to it, throughout this time period.

D. *Justification and Analysis*

As both the graphs have continuous positive slope, i.e. are ascending continuously away from x-axis, it proves that the Resource Cost increases with the corresponding increase in time. Thus we can conclude that:

$$\text{Resource Cost} \propto \text{Arrival Rate } (\lambda) \tag{1}$$

This is because, with the increase in the no. of arrival rate, the no. of requests increase, thus increasing the overall Resource Cost.

The uniform increase in the resource cost is also because the resources are always allocated with respect to a constant fixed ratio in case of the LBRS algorithm.

E. Conclusion

Thus we clearly conclude that on the basis of the results obtained in various simulations and demonstrated through various figures (fig. 2-6) and the above graphical analysis that the LBRS algorithm clearly outperforms the algorithm in [15] leading to lesser overall resource cost.

V. FUTURE WORK AND USE

Further analysis of LBRS algorithm against other QoS parameters like Response Time, Throughput and CPU Utilization etc. is possible. Also its comparison with various meta-heuristic algorithms like Genetic Algorithm (GA) and Ant Colony Optimization Algorithm (ACO) [26] etc. may provide some useful insights.

The algorithm can also be embedded with any standard existing approach of scheduling the resources. Also as the LBRS algorithm leads to an efficient utilization of resources, it serves the aim of Green IT and makes a contribution towards a better future.

REFERENCES

[1] W. Zhu, C. Luo, J. Wang, and S. Li, "Multimedia cloud computing," Signal Processing Magazine, IEEE, vol. 28, no. 3, pp. 59–69, 2011.

[2] Y. Wu, C. Wu, B. Li, X. Qiu, and F. Lau, "Cloudmedia: When cloud on demand meets video on demand," in Proc. IEEE Conference on Distributed Computing Systems (ICDCS), pp. 268–277, 2011.

[3] A. Aggarwal and S. Jain, "Efficient Optimal Algorithm and Task Scheduling in Cloud Computing Environment", International Journal of Computer Trends and Technology (IJCTT), vol. 9, pp. 344-349, March 2014.

[4] Avnish Kumar Yadav and Maitreyee Dutta, "A Novel Approach to Provide Broking Service in Cloud Computing", Thesis, NITTTR, Panjab University, Chandigarh, India, 2014.

[5] G. Li, H. Sun, H. Gao, H. Yu, and Y. Cai, "A survey on wireless grids and clouds," in *8th IEEE International Conference on Grid and Cooperative Computing,* pp. 261–267, 2009.

[6] W. Herroelen, E. Demeulemeester and B. De Reyck, "Resource-constrained project scheduling—A survey of recent developments", *Computers and Operations Research*, vol. 25 (4), pp. 279–302, 1998.

[7] X. Sun, S. Su, P. Xu and L. Jiang, "Optimizing multi-dimentional resource utilization in virtual data center" in *Proc. IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT),* pp. 1-6, 2011.

[8] M. Bendraouche, M. Boudhar and A. Oulamara, "Scheduling: Agreement Graph vs. Resource Constraints", in *European Journal of Operational Research,* pp. 585-586, 2015.

[9] R. Santhosh and T. Ravichandran, "Non-Preemptive on-line scheduling of real-time services with task migration for cloud computing", *European Journal of Scientific Research*, vol. 89 (1), pp.163-169, 2012.

[10] R. Santhosh and T. Ravichandran, "Pre-emptive Scheduling of On-line Real Time Services With Task Migration for Cloud Computing" in *Proc. IEEE Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME),* pp. 1-6, 2013.

[11] S. Sadhasivam, R. Jayarani, N. Nagaveni and R. V. Ram, "Design and Implementation of an efficient Two level Scheduler for Cloud Computing Environment" in *Proc. of International Conference on Advances in Recent Technologies in Communication and Computing*, pp. 1-3, 2009.

[12] R. Jayarani, N. Nagaveni and R. V. Ram, "Design and Implementation of an efficient Two level Scheduler for Cloud Computing Environment" in *Proc. Of 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pp. 1-3, 2010.

[13] D Talby and G. Feitelson, "Supporting Priorities and improving Utilization of the IBM SP Scheduler Using Slack-Based BackFilling", *IEEE Transactions on Parallel and Distributed systems*, vol. 18 (10), 1999.

[14] R. N. Calheiros, R. Ranjan, C. A. F. D. Rose, and R. Buyya, "CloudSim: A Novel Framework for modeling and Simulation of Cloud Computing Infrastructures and Services", 2009.

[15] X. Nan, Y. He, and L. Guan, "Optimization of Workload Scheduling for Multimedia Cloud Computing", in the *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–4, 2013.

[16] X. Nan, Y. He, and L. Guan, "Towards Optimal Resource Allocation For Diffentiated Multimedia Services in Cloud Computing Environment", in the *Proc. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)*, pp. 1–5, 2014.

[17] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud based on queuing model", in the *Proc. IEEE International Workshop on Multimedia Signal Processing (MMSP)*, pp. 1–6, 2011.

[18] X. Nan, Y. He, and L. Guan, "Optimal resource allocation for multimedia cloud in Priority Service Scheme", in the *Proc. IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–4, 2012.

[19] W. E. Dong, W.U. Nan and L. Xu, "QoS-oriented Monitoring Model of Cloud Computing Resources Availability", in *Proc. of IEEE International Conference on Computational and Information Sciences,* pp. 1537-1540, 2013.

[20] Z. Dong, W. Nan and L. Xu, "The Bilateral Resource Integration Service System", in the *Proc. of IEEE International Conference on Computational and Information Sciences,* 2011.

[21] G. Guo-Ning and H. Ting-Lei, "Genetic Simulated Annealing Algorithm for Task Scheduling based on Cloud Computing Environment", in *Proc. of International Conference on Intelligent Computing and Integrated Systems,* pp. 60-63, 2010.

[22] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey" in *Theoretical Computer Science*, vol. 344 (2–3), pp.243–278, 2005.

[23] M. Dorigo and L.M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem", in IEEE Transactions on Evolutionary Computation, pp. 53–66, 1997.

[24] K. Li, G. Xu, G. Zhao, Y. Dong and D. Wang, "Cloud Task scheduling on Load Balancing Ant Colony Optimization", in the *Proc. of IEEE Sixth Annual ChinaGrid Conference*, pp. 3-9, 2011.

[25] Jasmin James and Bhupendra Verma, "Efficient VM Load Balancing Algorithm For a Cloud Computing Environment", International Journal on Computer Science and Engineering, Vol. 4 No. 09, September, 2012, pp. 1658-1663, 2012.

[26] Ritu Kapur, "Review on Nature Inspired Algorithms in Cloud Computing", in *Proc. of IEEE International Conference on Computing, Communication and Automation (ICCCA-2015),* School of Computer Science and Engineering, Galgotias University, Uttar Pradesh, India, May 15-16, 2015.

[27] Ritu Kapur, "Review of various Meta- Heuristic Techniques in Cloud", in the Proc. *of International Conference on Recent Trends in Engineering, Science and Technology*, International Journal of Advance Research in Science and Engineering, AR Publications, JNU, Delhi, 2015.