



Cloud Scheduling Using Mumbai Dabbawala

S.K.Senthil Kumar¹, P.Balasubramanie²

¹Assistant Professor, Department of Computer Science and Engineering, Jansons Institute of Technology, Coimbatore, Tamil Nadu, India
skenthilkumar@jit.ac.in

²Professor, Department of Computer Science and Engineering, Kongu Engineering College, Erode, Tamil Nadu, India
pbalu_20032001@yahoo.co.in

Abstract— Cloud Computing is a revolutionary way of providing shared resources over the Internet. Though there are numerous researchers working towards the development of Cloud computing, still Cloud Computing is in its infancy. So to reap its full benefits, much research is required across a broad array of topics. The objective of our research is to achieve and improve the reliability cloud services and availability of cloud resources. We try to maximize the utilization of resources to keep working resources available for tasks that are yet to come and also concentrate on the reliability of cloud services. We propose a new scheduling algorithm called Dabbawala cloud scheduling Algorithm based on Mumbai Dabbawala delivery system. In our proposed system the tasks are grouped according to its cost required to complete in a Cluster and its VM resources. We find the lowest cost cluster and its VM for each task requested and group it together for getting services as in the Hadoop Map Reduce model. We have two phases called mapping the tasks and reduce the mapped tasks. The algorithm consists of four Dabbawala for each tasks to get serviced. From this algorithm, we compare some available scheduling algorithms. We achieve considerable gain in time and resources utilization.

Keywords—Cloud Computing, Cloud Reliability, Cloud Availability, Cloud Scheduling, Dabbawala, Hadoop MapReduce

I. INTRODUCTION

Cloud Computing is an upcoming and economic way of providing shared resources over the Internet [1] to the users and user's tasks. A cloud Computing service differs [2] from conventional way of hosting of applications and services in three principal aspects. First, it provides on demand services, typically by the minute or the hour; second, Resource scaling is elastic, since the user can have as much or as little of a service as they want at any given time; and third, the service is fully automated and managed by the Cloud service provider – user needs have a subscription and computer, Internet access. Cloud customers need to pay only for the services requested and they also a customized service level agreement (SLA), which is a acceptance by negotiated and agreed between a customer who requested for the cloud service and a cloud service provider: the cloud service provider is required to provide the service requests from a customer within accepted and negotiated cloud service quality of service (QoS) requirements for a given price and agreements.

The importance in utilization of Cloud Computing awareness arises in the provided opportunity that its ability to provide for the development of application services without the requirement of a prior to deployment hardware, software and maintenance expenditure. Furthermore, in the scientific field of study, Cloud Computing proved us with the ability to lease computational, platform and software resources from its virtually infinite pool for use of very High Performance Computing (HPC) facilities. In this way, even small institutions and businesses, or individuals can have access to a powerful large number of cloud computing VM computational resources at a negligible cost of maintaining a supercomputer and High Performance Computing center. Due

to dynamic nature of cloud environments, different user's unexpected requests and latency dependent of load, providing expected quality of service while avoiding under and over-provisioning is not a simple task [3].

Cloud computing is still in its early stages, [4] so to gain its full benefits, much research is required across an extensive assortment of issues. Today almost all the fields such as all Financials Institutions, Businesses, Education and even the Healthcare system are depends on cloud computing, Reliability of cloud services and availability of cloud resources are the most expected properties required by the clients. To achieve these properties, there are several parameters needs to be considered, such as Hardware performance, Internet capacity, Network traffic and so on. For the above parameters numerous researches are going on.

One of the important research issues which need to be concentrated for Cloud Computing's proficient performance to achieve the Reliability of cloud services and availability of cloud resources is scheduling [4]. The goal of scheduling is to map tasks to appropriate resources that optimize one or more objectives which includes maximize the utilization of resources. Since the Cloud has finite number of Virtual resources but exaggerated as infinite resources. Scheduling [4] in cloud computing belongs to a category of problems known as NP-hard problem due to large solution space and thus it takes a long time to find an optimal solution. There are no algorithms which may produce optimal solution within polynomial time to solve these problems. Scheduling allows optimal allocation of resources among given tasks in a finite time to achieve desired quality of service. Our objective is to improve the utilization of each virtual resource to improve the expected properties. With our knowledge and understanding the various researchers' articles, we propose a new scheduling algorithm called Dabbawala cloud scheduling Algorithm based on Mumbai Dabbawala delivery system. In our proposed system the tasks are grouped according to its cost required to complete in a Cluster and its VM resources. We find the lowest cost cluster and its VM for each task requested and group it together for getting services as in the Hadoop Map Reduce model. We have two phases called mapping the tasks and reduce the mapped tasks. The algorithm consists of four Dabbawala for each tasks to get serviced. From this algorithm, we compare some available scheduling algorithms. We achieve considerable gain in time and resources utilization

II. DABBAWALA – AN INTELLIGENT DELIVERY SYSTEM



Figure 1. One of the Transport for Dabbawala

A Dabbawala [5] [6] is a lunch box delivery man most commonly in Mumbai-India, who is being part of a lunch box delivery system, collects fresh food in lunch (Dabba) boxes from the houses of Mumbai people, who is working in within the any part of Mumbai, in the delayed morning, delivers the lunches to the their workplace, predominantly using conventional low cost transport such as bicycles and the railway local trains (Figure 1) and returns the empty boxes back to the worker's residence in the same day t afternoon. They are also made use of by meal suppliers in Mumbai where they ferry ready, cooked meals from central kitchens to the customers and back. In Mumbai, most office workers prefer to eat home-cooked food in their workplace rather than eat outside at a food stand or at a local restaurant, usually for reasons of taste and hygiene. A number of work-from-home women also supply such home-cooked meals, delivering through the Dabbawala network. A collecting Dabbawala,

usually on bicycle, collects dabbas either from a worker's home or from the dabba makers. As many of the carriers are of limited literacy, the dabbas (boxes) have some sort of distinguishing mark on them, such as a colour or group of symbols. The Dabbawala then takes them to a sorting place, where he and other collecting Dabbawalas sort the lunch boxes into groups. The grouped boxes are put in the coaches of trains, with markings to identify the destination of the box. The markings include the railway station to unload the boxes and the destination building delivery address. At each station, boxes are handed over to a local Dabbawala, who delivers them. The empty boxes are collected after lunch or the next day and sent back to the respective houses.

A. Dabba Markings

The “address” of the customer is painted on the top by the Dabbawala. The coding system [19] “speaks” to its bunch of illiterate workers who provide cheap labour and a committed workforce. The code, which is painted on the Dabba top, is restricted first by the size of the top itself – six inches in diameter. The codes use colour, dashes, crosses, dots and simple symbols to indicate the various parameters such as originating suburb, route to take, destination station, whose responsibility, the street, building, floor, etc. The work is known for its ingenuity, special codes and markings. The Dabbawalas do not need to know the precise home address since they know the address in the collection area by heart. If a new customer appears in his area, the Dabbawala will do the complete journey to check the address of delivery and check with other colleagues to see who has a free place in his crate to add one more dabba. Once the chain has been established and all the necessary stops for exchange decided, the address on the dabba is marked. Figure 2. Shows the sample marking in the lunch Box.

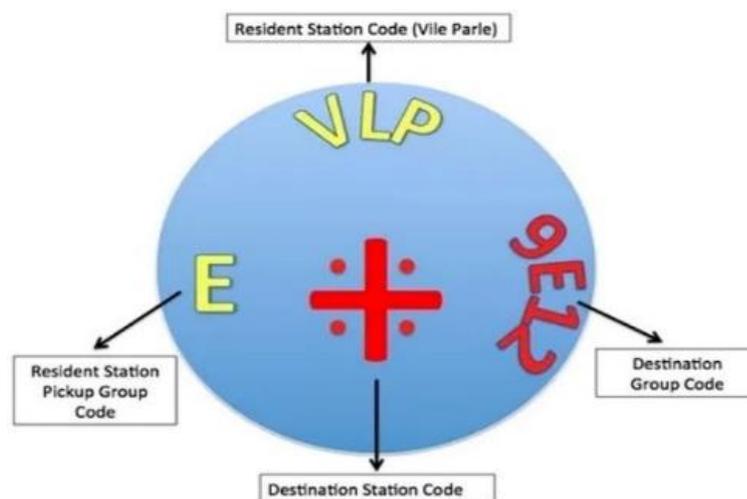


Figure 2. Unique Marking in Lunch Box Top.

B. Procedure for Delivery System.

- The first Dabbawala collects the lunchbox from the household and marks it with a unique code
- Each Dabbawala meets at a designated place where the boxes get sorted and grouped into a carriage
- The second Dabbawala marks the carriage uniquely to represent the destination and puts that in a local train. The markings include the local rail station to unload the boxes and the building address where the box has to be finally delivered.
- The third one travels along with the dabbas in the local train to handover the carriages at each station.
- The fourth Dabbawala picks up the dabbas from the train, decodes the final destination and delivers it.
- The process is just reversed in the evening to return the empty lunchboxes.

C. Apache Hadoop

Apache Hadoop, similar process to Dabbawala [19], is a framework to process large amounts of data in a highly parallelized and distributed environment. It solves the problem of processing petabytes of data by slicing the dataset into individual chunks that can be processed individually by inexpensive machines in a cluster. Apache Hadoop has two components – 1) A file system called HDFS that is designed to deal with the distributed data in a highly reliable way and, 2) the MapReduce engine that processes each slice of the data by applying the algorithm. For example, the Indian Meteorological department would have recorded the temperatures of each city on a daily basis for the last 100 years. Undoubtedly, this dataset would run into a few Terabytes! Imagine the computing power that is required to query

this dataset to find the city with the highest temperature in the last 100 years. This is exactly where Hadoop can play a role! Once the Terabyte sized dataset is submitted to HDFS, it would slice the dataset into equal chunks and distributes each chunk to a machine running within the cluster. Then, the developer would need to write the code in two parts – 1) The code that finds the maximum temperature per each slice of dataset running on each machine (Mapper) and, 2) the code that can collect and aggregate the output of the previous step to find the city with maximum temperature (Reducer). Figure 3. Shows the way of MapReduce works.

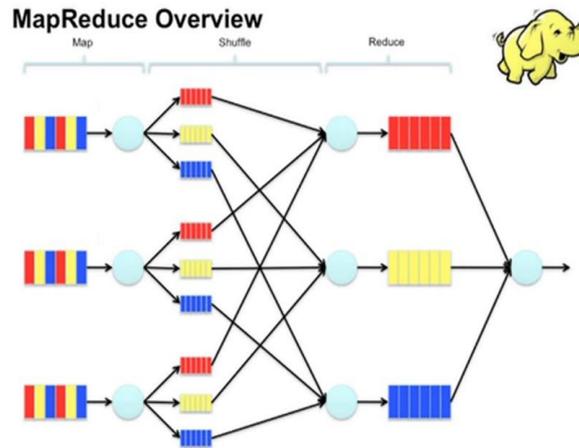


Figure 3. The MapReduce Procedure

MapReduce is precisely the algorithm that helps developers perform these two steps efficiently. If the developer writes the MapReduce code to find the city with the maximum temperature on a tiny dataset with just a few records, that same code will seamlessly work against Petabytes of data! Effectively, Apache Hadoop makes it easy to process large datasets by letting the developers focus on the core logic than worrying about the complexity and size of the data. In between the Map and Reduce phases, there are sub processes to shuffle and sort the data to make it easy for the reducers to aggregate the results. Below is an illustration of MapReduce process. Just like HDFS slices and distributes the chunk of data to individual nodes, each household submits the lunchbox to a Dabbawala. All the lunchboxes are collected at the common place for tagging them and to put them into carriages with unique codes. This is the job of the Mapper. Based on the code, carriages that need to go to the common destination are sorted and on-boarded to the respective trains. This is called Shuffle and Sort phase in MapReduce. At each railway station, the Dabbawala picks up the carriage and delivers each box in that to respective customers. This is the Reduce phase.

III.RELATED STUDY

There are several research were done on cloud scheduling for various objectives. We represent some of them here. The author [7] proposed a resource scheduling algorithm called as MinTotal DBP for gaining the minimum cost for cloud utilization in Gaming cloud. Their Main Objective is to maintain the Minimum total cost for using the Cloud for gaming. The author [8] proposed an algorithm for online cloud storage by their clients' request to store in the cloud servers' hard dive. As every hard drives has the same size, this algorithm assumed that each hard drive as a bin and the data to be stored in this drive as item so they followed the classical Bin Packing algorithm to achieve the minimum storage. The author [9] proposed a multi dimension based bin packing model with Ant Colony based optimization for workload consolidation problem for energy gain in the cloud. Author [10] proposed a hybrid job scheduling approach based on the aid of Genetic Algorithm (GA) with Fuzzy Algorithm with consideration of the load balancing of system. They proposed the above system with the objective in mind to assign the jobs to the resources by considering the VM MIPS and length of the job. The author [11] proposed a Multi-Objective Genetic cloud resource scheduling Algorithm (MO-GA) to minimize energy consumption and to maximize the profit of cloud service provides under several the constraints of deadlines times. Some researchers accomplished research on [12] [13] batch scheduling algorithm using FCFS first come and first service based scheduling algorithm was proposed to utilization with and without backfilling model. The authors [20] proposed a Gang scheduling is an alternative to batch scheduling. It is a scheduling algorithm for parallel systems that schedules the related tasks, processes or threads to run simultaneously on different VM resources. Gang scheduling is based on ouster out matrix. Gang scheduling is useful for applications where performance of application degrades when any part of the application is not running. The author [14] proposed a Backfilling and Migration Gang Scheduling for better utilization of resource and to improve the cloud's performance. Authors in [15], studied several job scheduling algorithm and compare them and concluded as the good cloud job scheduling algorithm is supposed to schedule the resources in optimized use of the resource. The Author [16] proposed a deadline constrained heuristic based Genetic Algorithm (HGA) to schedule the resources to tasks with bottom-level (B-Level) and Top-Level (T-Level) priority.

Author [17] proposed a Bin Packing based cloud resource scheduling called VISBP variable item Size Bin Packing algorithm with one, two and multiple Dimensional Bin Packing and evaluated.

In cloud, the requested tasks may be classified as lite task when the task requires very less memory to accommodate its data and instructions and very less CPU requirement to complete the services. A task may be classified as data intensive task when it requires more memory to store data and very less CPU utilization. On the other hand, the task may be called processor intensive, when the task requires less memory to store and high CPU requirement to complete the service. Finally, a task may be complex, when it requires more memory to hold the data and instruction and more CPU requirement to complete the service. Same as a task may be an atom task, if the task cannot be divided. Some task can be called as molecule, if we can split it into two or more sub tasks and execute.

IV. PROPOSED MODEL

We propose a new method of resource scheduling in cloud that follows the Mumbai Dabbawala and the Hadoop MapReduce model to achieve and improve the Reliability in cloud services and Availability of cloud resources. When we compare the Mumbai Dabbawala procedure and Cloud Computing, we find them mostly close enough together. With slight changes in the Cloud process we can achieve our objective of maximizing the resource utilization for achieving and improving the Reliability in cloud services and Availability of cloud resources. Our proposed model collects the tasks and separates the tasks into groups based on the cost of cluster to schedule and allocate.

In our proposed system, the user's tasks are received and shuffled and sorted in the same kind. In the proposed process of cloud scheduling, First Dabbawala collects all tasks requested by any user using the Flexible Quantum Time slice (FQTS) [18] and also collects the basic information such as User-id, Arrival-Time, Deadline-Time and Task-Priority. And places in the tasks pool in ascending order on Deadline Time, SLA to the user, and arrival time and task priority. Collect the historical [18] log details for each task on each cluster and its Resource (VM). These details include Task-id, User-Privilege, User-SLA, Task-CPU-Requirements, Task-Mem-Requirements, Expected-Execution-Time, Success-Ratio, VM-Type and Cluster network band-width, and present network traffics. Once the first Dabbawala completes its work it submits all the details of the tasks into a common place where all other first Dabbawalas submitted. Now the Second Dabbawala marks the task group uniquely to represent the destination. The markings include the cluster and VM. The tasks are sorted in-terms of destination. Third Dabbawala Schedules and allocates where the tasks are finally placed for getting service based on destination cluster and resource (VM).

A. Advantages

Advantages of the proposed system are first the resources can be utilized more by same kind of tasks. This algorithm is efficient because it knows the information and previous historical Logs, so we can segregate as per the actual requirements. With these two situations are solved. First, when we schedule blindly (without knowing the tasks attribute) higher VM Resource may be allocated where the resources may be underutilized. Second, a complex task may be allocated on weak resource which may lead to failure on services and execution time may be stretched. These both definitely affect the reliability of cloud services and availability of cloud resources.

B. Limitations

The limitations of our proposed Dabbawala Scheduling Algorithms is the resources have to wait until all the tasks are scheduled and allocated which makes the delay and the resources are not utilized. This leads to underutilization in cloud services.

C. Proposed Algorithm.

Dabbawala Cloud Scheduling Algorithm

- Step 1 : First Dabbawala
- i) Receive all requested tasks from user on FQTS style of time with User-id, Arrival-Time, Deadline-Time and Task-Priority. Place in the Task Pool Queue in ascending order of deadline time.
 - ii) Collect Accumulated historical Log details for each Task from Database such as Task-id, User-Privilege, User-SLA, Task-CPU-Req, Task-Mem-Req, Expected-Execution-Time, Success-Ratio, VM-Type, Cluster Network Band-Width and Network Traffics.
 - iii) Calculate Cluster Cost for each task on each cluster with its VM.
 - iv) Create and attach a unique code for each task with low cost cluster and its Resource (VM)
 - v) Place all uniquely marked tasks in common place by all n Home Dabbawala
- Step 2 : Each Dabbawala meets at a designated place where the tasks get sorted and grouped into a group based on the destination code of unique mark of each task

- Step 3 : Second Dabbawala
Marks the Task group uniquely to represent the destination. The markings include the cluster and VM
- Step 4 : Third Dabbawala
Schedule using First Fit (FF), Any Fit (AF), Best Fit (BF) and Worst Fit (WF) with various FQTS such as FQTS =0, 200 MS, 400 MS, 600 MS, 800 MS and 1000 MS, and allocate where the tasks are finally placed for getting service based on destination cluster and resource (VM).
- Step 5 : Fourth Dabbawala - (Follow-up)
i) Record all logs and update the history of information for each Task
ii) Collect all results to the source cluster and handover the result to corresponding user.

Dabbawala Cloud Scheduling Algorithm Follow-up (works separate piece of code in all cluster simultaneously)

- Step 1 : Check any VM completes its service of this cluster then do steps 2 to 8 else do step 1.
- Step 2 : Calculate Waiting time of i^{th} task T_i as $WT_i = DT_i - AT_i$
- Step 3 : If any task T_i released from VM_j then do step 3
- Step 4 : i) Calculate Service Time $ST_i = RelT_i - DT_i$
ii) Find Total Time of $TOTT_i = WT_i + ST_i$
iii) Find total turnaround time = $\sum TOTT_i$
- Step 5 : Write Log Report for each event
- Step 6 : Find Success Score SS_i
- Step 7 : Update all VMs and Tasks Attributes.
- Step 8 : Do Step 1

V. EXPERIMENTAL SETUP

We implement our proposed Dabbawala based Cloud Scheduling algorithm using the Java 1.7.0.75 and also we implement the VISBP (compare 1) algorithm, MinTotal Dynamic Bin Packing (compare 2), and Bottom-level and top-level GA (BTGA) (compare 3) in the following hardware configurations. We have 3 modules of IBM Blade server with Xeon processor with 8 GB RAM and 1.5 TB storage act as cloud cluster with 26 systems of Intel I3 core – 64 bit processors with 4 GB RAM and 500 GB Storage. We assigned 10 system as cluster 1, 8 system belongs cluster 2 and cluster 3. We installed VMware to make Virtual Machine. Each Physical machine has 3 VMs of 2 Nos. of 1GB RAM with 1 CPU core and 1 No. of VM with systems 2 GB RAM and 2 CPU Cores. In our implemented system, as a whole, we have 20 VM with 1 GB RAM and 1 CPU Core and 10VMs with 2GB RAM 2 CPU cores in cluster 1. 16 VM with 1 GB RAM and 1 CPU Core and 8 VMs with 2GB RAM, 2 CPU cores in cluster 1 and 16 VM with 1 GB RAM and 1 CPU Core and 10VMs with 2GB RAM, 2 CPU cores in cluster 3. As a whole we have 78 VMs.

VI. SOLUTION SPECIFICATION

We formulate a test bed to evaluate the proposed algorithm. First we try to find the optimum FQTS for our set of 5 test tasks generated randomly. We evaluate our proposed algorithm with all possible scheduling methods such as First Fit (FF), Any Fit (AF), Best Fit (BF) and Worst Fit (WF) with various FQTS such as FQTS =0, 200 MS, 400 MS, 600 MS, 800 MS and 1000 MS.

First we test our algorithm with FQTS with 0. This means, algorithm works as a pure dynamic Bin Packing algorithm. For testing, we follow the FF (First Fit), the task arrived in a Poisson distribution, are assigned in the VM which opened earlier if the constraints satisfied. If no VM is fit for assign then a new VM is opened to process. We test this algorithm with our 5 set of task list generated. The total time to complete to process all the tasks by our algorithm is compared and a graph drawn. The algorithm is tested for FQTS = 0 and AF (Any Fit), the received task directly assigned to any randomly chosen VM. If the chosen VM could not satisfy the constraints then the algorithm tries to make use of any other available a new VM is opened to process. The total time to complete to process all the tasks by our algorithm is compared and a graph drawn. For Best fit (BF) for FQTS = 0, the received task find the available VM whose CPU percentages and memory percentage is exactly or very nearly matches, in order to enhance the utilization of the resource. For Worst Fit (WF), the received task is assigned to the VM whose CPU and Memory is least used for other tasks. If no VM is available then a new VM is opened to process the task. We test our algorithm with FQTS = 200 MS is fixed to activate the hybrid type; here the receiving system receives all the arrived tasks for the period of 200 MS. After 200 MS the queued tasks are sorted on priority and maximum to minimum size of the tasks and assign to the VM based on methods such as FF, AF, BF and WF. The total time to complete is noted. Same as the FQTS = 400, 600, 800 and 1000 MS for all the assigning methods.

From our evaluation of our proposed algorithm Dabbawala with FQTS = 0, 200, 400, 600, 800, and 1000, we plot the graphs which are all shown in figure 4 (a), (b), (c), (d) and (e). By carefully analysing those graph we understand, our algorithm yields better optimum lowest lime to complete the service for the tasks given. From this we conclude that our proposed algorithm can produce expected results at FQTS = 600. From our evaluation of our proposed algorithm Dabbawala with FQTS = 0, 200, 400, 600, 800, and 1000, we plot the graphs which are all shown in figure 1 (a), (b), (c), (d) and (e). By carefully analysing those graph we understand, our algorithm yields better optimum lowest lime to complete the service for the tasks given. From this we conclude that our proposed algorithm can produce expected results at FQTS = 600.

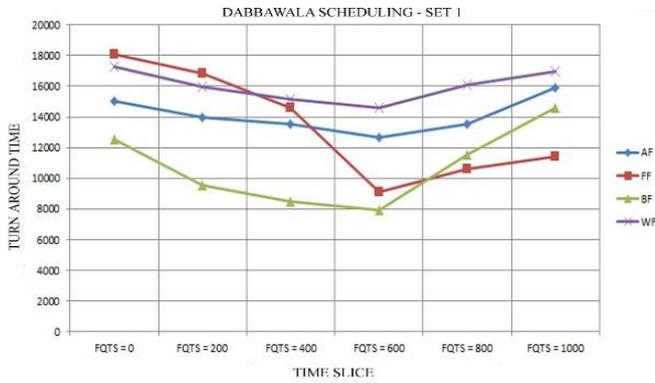


Figure 4.1. Dabbawala Algorithm – Set 1.

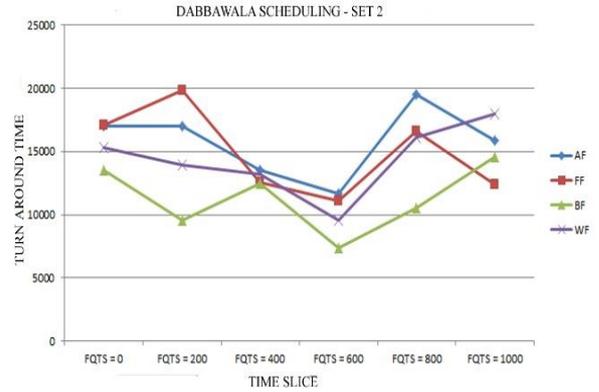


Figure 4.2. Dabbawala Algorithm – Set 2.

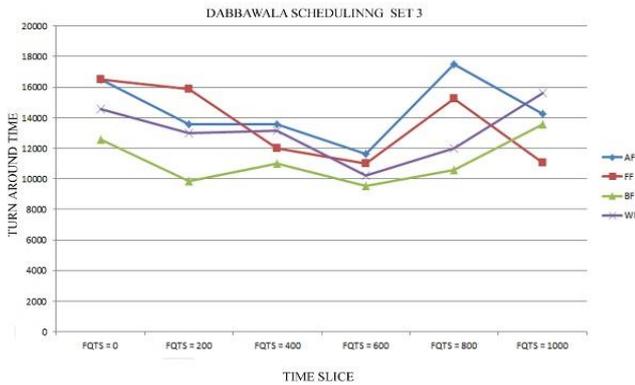


Figure 4.3. Dabbawala Algorithm – Set 3.

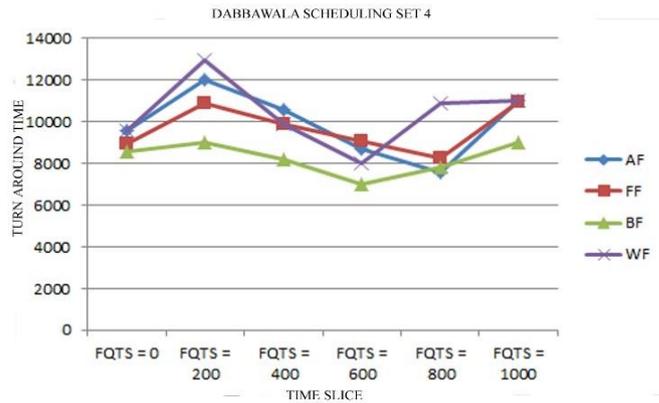


Figure 4.4. Dabbawala Algorithm – Set 4.

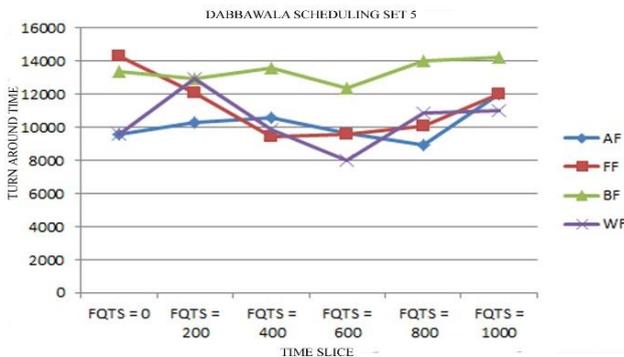


Figure 4.5. Dabbawala Algorithm – Set 5.

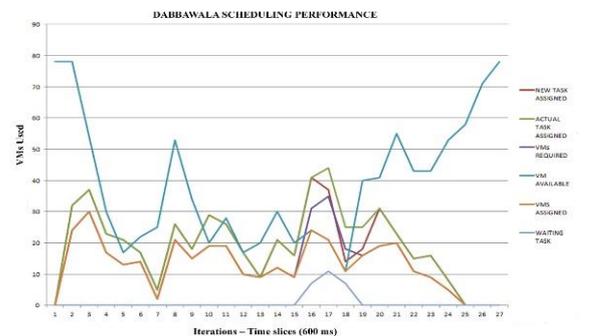


Figure 5. Dabbawala Scheduling algorithm Performance

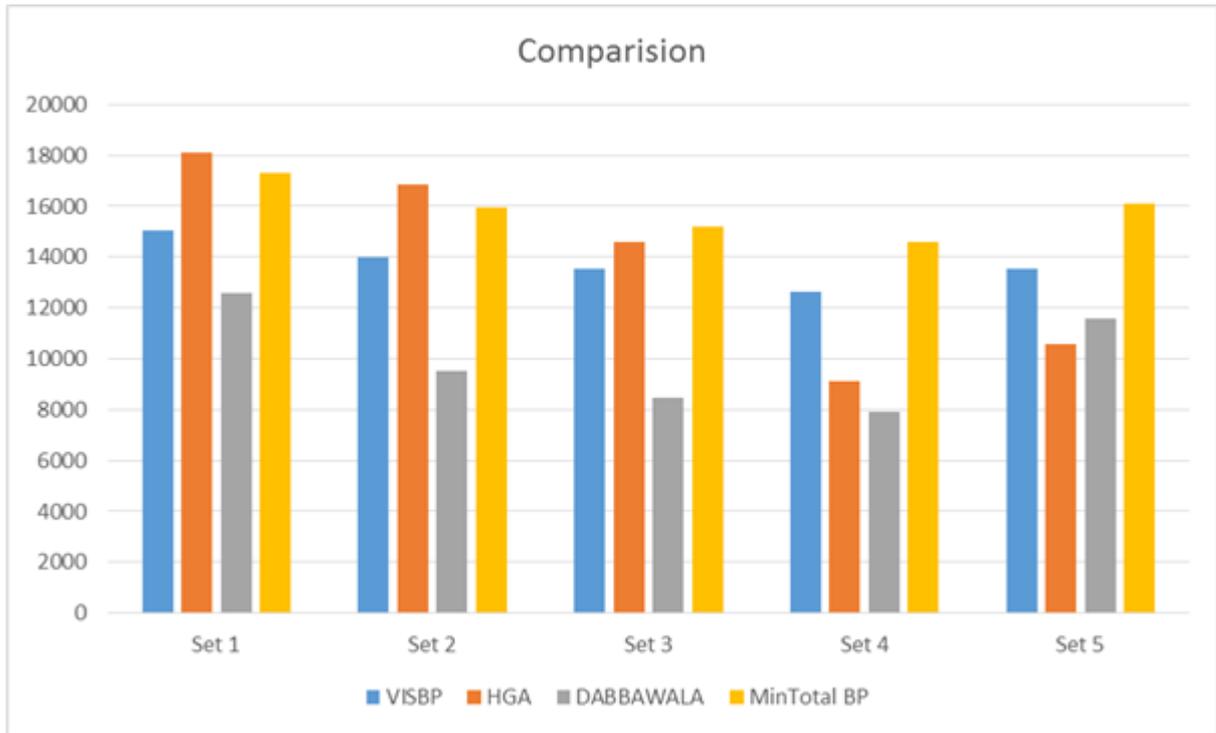


Figure 5. Comparison of proposed Dabbawala scheduling and other scheduling algorithms

VII. COMPARISON

Our proposed Dabbawala algorithm is compared to system already proposed by other researchers. First we compare with MinTotal DBP [7] Minimum total Dynamic Bin Packing proposed by Yusen Li and et al. Second we compare our proposed algorithm with HGA [16] algorithm with B-level and T-level priority scheduling algorithm proposed by Verma, Amandeep, and Sakshi Kaushal. Third, we compare with VISBP [17] a multi-dimensional Bin Packing algorithm proposed by Song, Weijia and et al. From the graph (Figure 5) shows the performance of our proposed system. The Graph (figure 3) shows the performance of our system with other three systems. From the graph we understand that our proposed Dabbawala Cloud Scheduling algorithm schedule the tasks with better reliability and availability compare with other systems.

VIII. CONCLUSION

Our Proposed algorithm Dabbawala Cloud Scheduling algorithm finds the load balance and low cost cluster and resources in the cloud system. The resources that are having high network capacity and less traffic is selected and utilized. So the tasks can be completed in its service as expected and effectively improves utilization as maximum as possible. On proper analysis and evaluation, we understand that our proposed system performs better than other scheduling algorithm proposed by other researchers. To improve more we are planning improve the scheduling algorithm, we have a plan to add a monitoring phase to understand and follow the cloud traffic status slightly before the situation. So we can minimize the waiting time and total turnaround time of the requested tasks.

REFERENCES

- [1]. Moschakis, I. A., & Karatza, H. D. (2012). "Evaluation of gang scheduling performance and cost in a cloud computing system". *The Journal of Supercomputing*, 59(2), 975-992.
- [2]. Khazaei, H., Mišić, J., & Mišić, V. B. (2012). "Performance analysis of cloud computing centers. In *Quality, reliability, security and robustness in heterogeneous networks*", (pp. 251-264). Springer Berlin Heidelberg.
- [3]. K. Xiong and H. Perros (2009), "Service performance and analysis in cloud computing", Volume 0, pages 693–700, Los Alamitos, CA, USA.

- [4]. Kalra, M., & Singh, S. (2015). A review of metaheuristic scheduling techniques in cloud computing. *Egyptian Informatics Journal*.doi:10.1016/j.eij.2015.07.001
- [5]. Ravichandran, N. (2005). "World Class Logistics Operations: The Case of Bombay Dabbawallahs", Indian Institute of Management, Ahamadabad, Research and Publications, India, W.P. No. 2005-09-01.
- [6]. Percot, M. (2005). Dabbawalas, tiffin carriers of Mumbai: answering a need for specific catering. halshs-00004513, <https://halshs.archives-ouvertes.fr/halshs-00004513>
- [7]. Li Yusen et al (2014), "On dynamic bin packing for resource allocation in the cloud." In Proceedings of the 26th ACM Symposium on Parallelism in Algorithms and Architectures, pp. 2-11. ACM.
- [8]. Bein Doina et al (2012), "Cloud storage and online bin packing." In Intelligent Distributed Computing V, pp. 63-68. Springer Berlin Heidelberg, 2012.
- [9]. Feller Eugen et al (2011), "Energy-aware ant colony based workload placement in clouds." In Proceedings of the 2011 IEEE/ACM 12th International Conference on Grid Computing, pp. 26-33. IEEE Computer Society.
- [10]. Javanmardi, Saeed, et al. (2014), "Hybrid job scheduling algorithm for cloud computing environment." Proceedings of the Fifth International Conference on Innovations in Bio-Inspired Computing and Applications IBICA 2014. Springer International Publishing.
- [11]. Zitzler Eckart et al (2000), "Comparison of multiobjective evolutionary algorithms: Empirical results" *Evolutionary computation* 8, no. 2: 173-195.
- [12]. Sandholm Thomas et al (2015), "QoS-Based Pricing and Scheduling of Batch Jobs in OpenStack Clouds." arXiv preprint arXiv:1504.07283.
- [13]. Mathew Tojo et al (2014) "Study and analysis of various task scheduling algorithms in the cloud computing environment." In *Advances in Computing, Communications and Informatics (ICACCI, 2014 International Conference on)*, pp. 658-664. IEEE.
- [14]. Liu, Yu et al (2015), "DeMS: A hybrid scheme of task scheduling and load balancing in computing clusters." *Journal of Network and Computer Applications* (2015).
- [15]. Zhang Yanyong et al (2003), "An integrated approach to parallel scheduling using gang-scheduling, backfilling, and migration." *Parallel and Distributed Systems, IEEE Transactions on* 14, no. 3 (2003): 236-247.
- [16]. Verma Amandeep, and Sakshi Kaushal. (2014), "Deadline Constraint Heuristic-Based Genetic Algorithm for Workflow Scheduling in Cloud." *International Journal of Grid and Utility Computing* 5, no. 2: 96-106.
- [17]. Song Weijia et al (2014), "Adaptive resource provisioning for the cloud using online bin packing." *Computers, IEEE Transactions on* 63, no. 11 (2014): 2647-2660
- [18]. S. K. Senthil Kumar and P. Balasubramanie (2012), "Dynamic Scheduling for Cloud Reliability Using Transportation Problem," *Journal of Computer Science*, vol. 8, no. 10, pp. 1615–1626, 2012.
- [19]. <https://www.quora.com/What-is-an-intuitive-explanation-of-MapReduce>
- [20]. <https://en.wikipedia.org/wiki/Dabbawala>.
- [21]. Liu, Yu et al (2015), "DeMS: A hybrid scheme of task scheduling and load balancing in computing clusters." *Journal of Network and Computer Applications* (2015).

BIBLIOGRAPHY



S.K.Senthil Kumar is working as Assistant Professor, CSE department in Jansons Institute of Technology, Coimbatore. He received his Master degree in Engineering in Computer science and Engineering in Anna University, Chennai in 2007 and Master in Computer Applications in 1998 from Bharathidasan University. He is doing the research in improving reliability of cloud service and availability of cloud resources from 2008 in the Anna University of Chennai, India under the supervision of Dr. P.Balasubramanie.



Dr.P.Balasubramanie has obtained his PhD degree in theoretical computer science in the year 1996 from Anna University, Chennai. He was awarded junior research fellow by CSIR in the year 1990. Currently he is a professor in the Department of Computer Science and Engineering, Kongu Engineering College, Perundurai, Tamilnadu. He has published more than 100 research articles in International/National Journals. He has also authored six books. He has guided 22 Ph.D scholars and guiding 10 research scholars. His areas of interest include theoretical computer science, data mining, image processing and optimization Techniques.