International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8 Issue-4, November 2019

An Efficient Grouped Task Scheduling and Resource Allocation in Cloud Computing Environments

Ilya Musa Adamu, Ismail Zahraddeen Yakubu, A.Y Gital, S. Boukari

Abstract: Task scheduling in cloud is the allocation of resources to a task at a particular time. In cloud, scheduling strategy is defined or adapted by a scheduler according to the changing environment. Allocation of resource with poor capacity to a task may lead to increase in execution time of the task. Problem of resource under utilization may also occur when a resource with high capacity is allocated to a task that requires a resource with lesser capacity. In this paper we proposed an Efficient Grouped Task Scheduling (EGTS) and resource allocation to minimize average waiting time, average execution time and increase resource utilization. EGTS classify Tasks into two groups of similar task type, and sort the tasks in the order of their respective deadlines. Task in each group is allocated Virtual Machine with capacity equal to the average capacity required by tasks in that group. An experiment was conducted using CloudSim to exhibit EGTS and the result shows minimal average execution time, average waiting time and a higher resource utilization compared to Min-Min and Max-Min.

Keywords: Cloud Computing, Task Scheduling, Virtual Machines, Execution Time, Waiting Time, resource utilization, EGTS, CloudSim.

I. INTRODUCTION

Cloud computing environment contains limited computing, storage and network resources. Large number of clients competes for the limited shared resources for data storage, execution of applications and other required operations. The cloud service providers provisioned the resources required by the clients that seem to be unlimited in the client view. Task scheduling is the allocation of limited, shared resources to tasks at a particular time (Wadhonkar & Theng 2016). The major challenge in cloud is the allocation of limited resources to large number of tasks such that the required quality of service is provided.

Manuscript published on November 30, 2019.

* Correspondence Author

Ismail Zahraddeen Yakubu*, Ismail Zahraddeen Yakubu is a PG student of Computer Science and Engineering in SRM Institute of Science and Technology, Tamil Nadu,

Ilya Musa Adamu,. is a PhD student of Computer Science from Abubakar Tafawa Balewa University, ATBU, Bauchi Nigeria

Dr A.Y Gital, is a Doctor of Computer Science from Abubakar Tafawa Balewa University, Bauchi Nigeria

Dr. S. Boukari, Professor Souley Boukari, is a Professor of Computer Science from Abubakar Tafawa Balewa University, Bauchi Nigeria

In order to maximize the utilization of cloud resource (Cloud service provider profit), minimize waiting time and execution time of a task, different resource allocation mechanisms were proposed. To support multiple clients in cloud, resources are partitioned into independent virtual machines that are capable of executing client task. The virtual machines are allocated to various tasks at a particular time. During task scheduling, a task may be allocated a Virtual Machine with less capacity as required by the task which may lead to higher execution time of the task. Also a task may be allocated to a virtual machine with greater capacity compared to the task required capacity, this lead to under utilization of the virtual machine (Zhang & Zhou 2018). To minimize resources under utilization, execution time and waiting time of a task we proposed an Efficient Grouped Task Scheduling and resource allocation that classify task based on their types, schedule the task based on their respective deadline and map each task to a Virtual Machine with capacity equal or nearly equal to the capacity required by the task.

II. RELATED WORKS

Pankaj et al (2016), proposed a scheduling mechanism that takes care of task deadline and based on the length and deadline of the task, it allocate a credit to the task. The mechanism then schedule the task based on its demand. Chitgar. et. al (2019), grouped virtual machines into Low MIPs, Medium MIPS and High MIPS. Two threshold values T1 and T2 are used to identify the virtual machine to assign to an incoming task. On arrival of a new task, the length of the task is compared with threshold T1. If the threshold is greater than the task length, then the task is assigned a virtual Machine from low MIPS Virtual Machine class, if T1 is less than the length of the task and T2 is greater than the length of the task, then it is assigned to a virtual machine from Medium MIPS class and High MIPS otherwise. Mahendra. et. al (2018), proposed a method that uses task length and task run time to rank all the task in the system. An incoming task is assigned a rank using analytical hierarchical processes (AHP) which and then placed in the queue while still maintaining the ranking. Lastly BATS + BAR are used to allocate resources to the task.Alworafi et al (2017), proposed an allocation technique to satisfy the client demand. The approach in this paper was based on the various resource demands with different cost.Hend et al (2017), categorized task into five groups based on user, task type, task size and task latency. Each category contain task of similar type. Category of task with high value of attributes is scheduled first. The tasks within the categories are scheduled based on their execution time.

Growing number of institutions are increasingly adopting Cloud into their organization to serve their users better and reduce cost. Rindos et al (2014), concluded in their paper that numerous academic institutions are keying into VCL. They clearly stated the current trends in enhancing VCL, and the cloud educational tool that would eventually enhance content delivery in the cloud.

Ajay Thomas et. al (2017), proposed a scheduling method that takes into account the deadline of each task that arrives into the system. The tasks are sorted based on the deadline so that each task has a deadline less than the subsequent task. The required resource for each task is allocated to the task for execution before the deadline of the task expires. This method focuses on satisfying the client demands but fails to satisfy the cloud service provider demand. Zhang and Zhou (2018), classify user task into groups using Bayesian classifier. Authors use the historical data of the previously executed task to create virtual machines capable of handling such task. The tasks in each group are matched to virtual machines generated from the historical data. Chauhan. et. al (2016) proposed a mechanism for task scheduling that takes into account the task deadline and length to assign a credit to task. The mechanism also considers the various demands of the task to generate a schedule for execution of task. In Li et. al. (2017), authors tries to maximize cloud provider benefit by classifying task into three groups. The groups includes: fixed interval job which needs a resource within time interval, time window interval and time slice job. The jobs are grouped together based on similarity of resources required and the resources are allocated sequentially using a linear relaxation technology and a group constraint interval coloring technology. Pande. et. al (2016) proposed a scheduling technique for heterogeneous cloud. The customer oriented technique chooses from the multiple clouds, a cloud with least execution time and allocate a task to the chosen cloud. Idle cloud systems are allocated task to create Makespan balance across the heterogeneous cloud system. In Sing. et. al (2014), authors proposed a multiple criteria preference synthesis. The method ranks the user task using the synthesized analytical hierarchical process (AHP) and ensures consistency in the ranking through Euclidean distance and minimum violation. Mittal and Katal (2016), proposed an algorithm that consider the various existing scheduling techniques and adopt one of them to be used in scheduling task based on the current situation and the calculations performed. Distribution of task across the available resources is done to minimize Makespan. Adamu. et. al (2019), proposed an improved round robin scheduling algorithm by assuming the quantum time to be equal to the burst time of the initial request. the quantum time dynamically varies after the execution of reach user request. the method also allocate the CPU to the currently executing request for the remaining CPU bust time if the remaining bust time is less than the quantum.

III. PROPOSED METHOD

Efficient Grouped Task Scheduling classify task into two groups (Urgent and Normal) based on task type. The first group contains a set of urgent task and the second group contains a set of normal task. EGTS sorts the tasks in each group based on their respective deadlines, if two tasks have equal deadline, then the first task to arrive is considered. EGTS schedule the first group maintaining the order of task in the group and then schedule the second group. The VM capacity of each group is initialized to the average required capacity of all task T_i in the group for i= 1, 2, ..., ., .m. Two types of Virtual Machines (UrgentVM and NornalVM) with capacities greater or equal to the average capacities of the two Task Groups are created for processing of Task to minimize response time of the system. Each task of a group is mapped to a Virtual Machine that is capable of processing task of that group.

A. Efficient Grouped Task Scheduling Algorithm

Input: Task T = {T_1, T_2, ...T_n}.

Output: A Queue of Task arranged in order of task type and task deadline.

```
For i = 1 : n do
    TaskType = Type of task Ti          /* get the type of task */
    VMCapacityRequired = Capacity of VM Required by Task Ti /* get the required VMCapacity of the task */
    TaskDeadLine = Deadline of  Task Ti  /* get the deadline of the task */
    If TaskType = = Urgent          /* check if task is urgent */
        UrgentCapacity = UrgentCapacity + VMCapacityRequired    /* add the required capacity of the
                                                                Current task to the required   capacity of
                                                                the previous task */
        UrgentCounter = UrgentCounter + 1  /* increment the number of urgent task*/
        For Countj = 2 to  n      /* sort task in Urgent list based on their respective deadline */
            Counti = Countj – 1;
            While Counti > 0 and Deadline of  UrgentList [Counti] > TaskDeadLine do
                UrgentList[Counti + 1] = UrgentList[Counti]
                Counti = Counti – 1
                UrgentList[Counti + 1] = Ti
        End For
    Else  /* task is normal */
        NormalCapacity   = NormalCapacity  +  VMCapacityRequired  /*  add
the required capacity of the
```

```
                                        Current task to the required   capacity of
                                        the previous task */
            NormalCounter = NormalCounter + 1 /* increment the number of normal task*/
          For Count_j = 2 to  n /* sort task in Normal list based on their respective deadline */
               Count_i = Count_j – 1;
               While Count_i > 0 and Deadline of  NormalList [Count_i] > TaskDeadLine do
                NormalList[Count_i + 1] = NormalList[Count_i]
                Count_i = Count_i –1
                 NormalList[Count_i + 1] = T_i
            End For
        End If
    End For
    UrgentCapacity = UrgentCapacity / UrgentCounter /* Calculate the average required capacity of the Urgent task */
    NormalCapacity = NormalCapacity/ NormalCounter /* Calculate the average required capacity of the Normalt task */
    Merge Urgent List and Normal List into single list
```

B. Task to VM Mapping Algorithm

Input: List of Task sorted in order their respective deadline, list of VM's
Output: Mapping of Task to VM

```
For i=1 : n do
        TaskType = Type of task T_i
         If TaskType = = Urgent
                Search for UrgentVM from VM's List
                Allocate any found UrgentVM to Task T_i
        Else
                Search for NormalVM from VM's List
                Allocate any found NormalVM to Task T_i
        End If
   End For.
```

IV. SIMULATION PROCESS

CloudSim 3.0 was used to simulate the cloud resources used in exhibiting the proposed method. The Process inherits and adopts classes such as cloudlet, DataCenterBroaker, VM and Host to exhibit the scheduling algorithm proposed in this paper.

V. ANALYSIS OF RESULTS

A total of 1000 task were partitioned into 10 groups with each group containing 100 tasks of the two types (Urgent and Normal). The individual groups were submitted to the system for execution and the average waiting time, execution time and resources utilization of the 10 groups were computed.

Methods	Average Execution Time (ms)	Average Waiting Time (ms)	VM Utilization
EGTS	6.4	5.9	23
Min-Min	32.3	6.2	5
Max-Min	48.4	16	3

Table 1: Comparison of Average Execution Time, Average Waiting Time and VM Utilization for Urgent Task

Methods	Average Execution Time (ms)	Average Waiting Time (ms)	VM Utilization
EGTS	10.3	9.2	19.1
Min-Min	*43.7*	*11.01*	*10.04*
Max-Min	59.5	22.1	10.01

Table 2: Comparison of Average Execution Time, Average Waiting Time and VM Utilization for Normal Task.

Table 1 show the comparison of Average execution time, average waiting time and VM utilization for Urgent task of EGTS, Min-Min and Max-Min Methods

Table 2 show the comparison of Average execution time, average waiting time and VM utilization for normal task of EGTS, Min-Min and Max-Min Methods

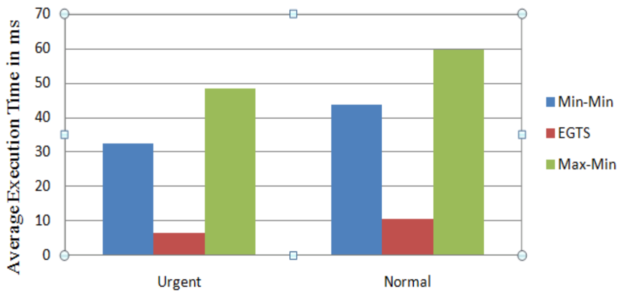

Figure 1: Comparison of Average Execution Time

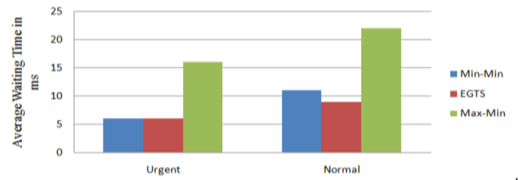

Figure 2: Comparison of Average Waiting Time

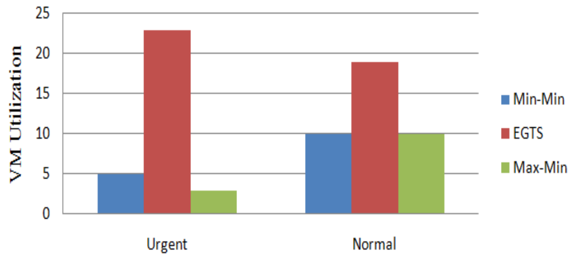

Figure 3: Comparison of VM Utilization

Figure 1 and Figure 2 above shows that EGTS records a minimum average execution time and average waiting time than the widely known Min-Min and Max-Min Method. Also figure 3 shows that EGTS records higher resource utilization than Min-Min and Max-Min Method.

VI. CONCLUSION

In this Paper an Efficient Grouped Task Scheduling and resource allocation was proposed to minimize average waiting time, average execution time and increase resource utilization. The EGTS method was exhibited using CloudSim and the result of the simulation yield a minimum average execution time, average waiting time and higher resource utilization compared to the widely known Min-Min and Max-Min.

REFERENCES

1. Chauhan, P. K., Jaglan, P., & Dabas, P. (2016). An intensify deadline aware credit based cloud task scheduling. *2016 International Conference on Computing, Communication and Automation (ICCCA)*. doi: 10.1109/ccaa.2016.7813911
2. Chitgar, N., Jazayeriy, H., & Rabiei, M. (2019). Improving Cloud Computing Performance Using Task Scheduling Method Based on VMs Grouping. *2019 27th Iranian Conference on Electrical Engineering (ICEE)*. doi: 10.1109/iraniancee.2019.8786391
3. Gawali, M. B., & Shinde, S. K. (2018). Task scheduling and resource allocation in cloud computing using a heuristic approach. *Journal of Cloud Computing*, *7*(1). doi: 10.1186/s13677-018-0105-8
4. Alworafi, M. A., Al-Hashmi, A., Dhari, A., Suresha, & Darem, A. B. (2017). Task-Scheduling in Cloud Computing Environment: Cost Priority Approach. *Proceedings of International Conference on Cognition and Recognition Lecture Notes in Networks and Systems*, 99–108. doi: 10.1007/978-981-10-5146-3_10
5. Hend Gamal El Din Hassan Ali, Saroit, I. A., & Kotb, A. M. (2017). Grouped tasks scheduling algorithm based on QoS in cloud computing network. *Egyptian Informatics Journal*, *18*(1), 11–19. doi: 10.1016/j.eij.2016.07.002
6. Wadhonkar, A., & Theng, D. (2016). A survey on different scheduling algorithms in cloud computing. *2016 2nd International Conference on Advances in Electrical, Electronics, Information, Communication and Bio-Informatics (AEEICB)*. doi: 10.1109/aeeicb.2016.7538374
7. Zhang, P., & Zhou, M. (2018). Dynamic Cloud Task Scheduling Based on a Two-Stage Strategy. *IEEE Transactions on Automation Science and Engineering*, *15*(2), 772–783. doi: 10.1109/tase.2017.2693688
8. Rindos, A., Vouk, M., & Jararweh, Y. (2014). The virtual computing lab (vcl): an open source cloud computing solution designed specifically for education and research. International Journal of Service Science, Management, Engineering, and Technology (IJSSMET), 5(2), 51-63.
9. Ajay, T. S., & Santhiya, C. (2017). Dynamic resource scheduling using Delay time algorithm in Cloud environment. 2017 *2nd International Conference on Computing and Communications Technologies (ICCCT)*. doi: 10.1109/iccct2.2017.7972238.
10. Li, M., Sun, Y.-E., Huang, H., Yuan, J., Du, Y., Bao, Y., & Luo, Y. (2017). Profit maximization resource allocation in cloud computing with performance guarantee. *2017 IEEE 36th International Performance Computing and Communications Conference (IPCCC)*. doi: 10.1109/pccc.2017.8280482
11. . Pande, S. K., Panda, S. K., & Das, S. (2016). A Customer-Oriented Task Scheduling for Heterogeneous Multi-Cloud Environment. *International Journal of Cloud Applications and Computing*, *6*(4), 1–17. doi: 10.4018/ijcac.2016100101
12. Singh, A., Dutta, K., & Singh, A. (2014). Resource Allocation in Cloud Computing Environment using AHP Technique. *International Journal of Cloud Applications and Computing*, *4*(1), 33–44. doi: 10.4018/ijcac.2014010103
13. Mittal, S., & Katal, A. (2016). An Optimized Task Scheduling Algorithm in Cloud Computing. *2016 IEEE 6th International Conference on Advanced Computing (IACC)*. doi: 10.1109/iacc.2016.45
14. Adamu, I. M., Gital, A. Y., & Yakubu, I. Z. (2019). Performance Evaluation of Hybrid Round Robin Algorithm and Modified Round Robin Algorithm in Cloud Computing. *International Journal of Recent Technology and Engineering 2*, *8*(2), 5047–5051. doi: 10.35940/ijrte.a9139.078219

AUTHORS PROFILE

Professor Souley Boukari, is a Professor of Computer Science from Abubakar Tafawa Balewa University, Bauchi Nigeria where He is currently involved in teaching, research and supervision of students both undergraduates and Postgraduates in Computer Science. He has a B-Tech, M.Sc. and PhD in Computer Science from ATBU. His current research areas are Data Mining, Machine Learning, Artificial Intelligence, Software Engineering, cyber security and Digital Forensic.

Ilya Musa Adamu, is a PhD student of Computer Science from Abubakar Tafawa Balewa University, ATBU, Bauchi Nigeria and a Semester abroad Student at SRM Institute of Science and Technology, Tamil Nadu, India. He is currently involved in teaching, research and supervision of students both undergraduates and Postgraduates in Computer Science at the Federal Polytechnic Bauchi Nigeria. He has a B-Tech in Computer, M.Sc. in Computer Science. His current research areas are cloud computing and Data communication and Network.

Dr. Abdulsalam Y Gital, is a Doctor of Computer Science from Abubakar Tafawa Balewa University, Bauchi Nigeria where He is currently involved in teaching, research and supervision of students both undergraduates and Postgraduates in Computer Science. He has a B-Tech, M.Sc. and PhD in Computer Science. His current research areas are Modeling and Simulation, Cloud Computing, Collaborative Virtual Environment and Computer Communication and Network.

Ismail Zahraddeen Yakubu is a PG student of Computer Science and Engineering in SRM Institute of Science and Technology, Tamil Nadu, India. He is a Staff in Department of Computer Science at Federal Polytechnic Bauchi Nigeria. He has a B.Sc. in Computer Science. His current research areas are Cloud Computing, Data communication and Networks, Image Processing, Database, Data Mining, Software Engineering and Internet of Things.