RFM-PSO -RFM Score based PSO Task Scheduling in Cloud

R. Valarmathi, T. Sheela

Abstract: Resource Management problem is considered as the major issue in recent decades This paper presents a novel particle swarm optimisation algorithm, with RFM score. Our proposed algorithm is used to solve the task scheduling problem. In our proposed algorithm there are two phases. RFM analysis of customers is done to improve the user experience as well as to increase the profit of cloud provider. The tasks are ranked according to RFM score and given priority according to the best rank. Ranked tasks forms the initial population of Particle swarm optimisation (PSO). In the Second phase the tasks are classified as CPU-intensive and I/O intensive. The two-phase algorithm helps to improve the performance of scheduling. Our proposed algorithm uses Cloudsim and compared with the existing metaheuristic algorithms like ACO and GA. Experimental results show that the RFM-PSO algorithm outperforms the other algorithms.

I. INTRODUCTION

Cloud computing is an emerging computation model that provides users with software, platform and infrastructure as a service. Cloud Computing provides many attractive features like scalability and reliability. The cloud Computing can be defined as a model that enables ubiquitous, on-demand network access to a pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. In simple way cloud computing is "a type of Internet-based computing," where different services everything from application, data center, data, resources are delivered over the Internet on a pay-for-use basis. Consumers can make use of the cloud based technologies by accessing the resources on a pay per use basis and release the resources when it is no longer required .Hundreds of VMs are used and allocating these VMs to process manually is not easy. So, we need an efficient task scheduling algorithm in the cloud environment.

A Scheduling problem can be classified as either a optimization or decision problem[13]. The optimization problem identifies the best solution from the set of feasible solutions. The decision problem identifies whether the given possible solution achieves it objective or not. The job scheduling in a cloud computing platform can be considered in two different levels.1.User-level 2.Sytem Level. User level manages provision of services among the providers and the customers. System-level manages the resources within the data center.

Revised Manuscript Received on April 10, 2019.

R. Valarmathi, Research Scholar, Sathyabama Institute of Science and Technology and Assistant Professor, Sri Sairam Engineering College, Chennai, Tamil Nadu, India

T. Sheela, Professor, Department of Information Technology, Sri Sairam Engineering College, Chennai, Tamil Nadu, India.

Optimization algorithms are of many types and they are often nature-inspired. Many Metaheuristics algorithms are proposed such as particle swarm optimisation, Ant colony optimisation and some novel metaheuristic algorithms. Genetic algorithms (GA) uses genetic operators such as crossover and mutation. Solutions are represented as chromosomes or binary or real strings. Particle Swarm Optimisation (PSO) developed by Kennedy and Eberhart in 1995 was based on the swarming behavior of birds or fish. Genetic algorithms(GA) and PSO have the disadvantages with multimodal optimisation problems. The bat algorithm is a metaheuristic optimisation algorithm. It was proposed by Xin-She Yang in 2010,inspired by the echolocation behavior of microbats. Bats are the only mammals that have the capability of echolocation [2].

II. BACKGROUND WORK

Resource request changes over time based on the seasons and holidays. Sometimes there might be too many resources and very few request that leads to wastage of resources. There might be very few resources to satisfy too many request that might affect the performance.

Jobs may of different types. For example jobs may be uploading and downloading data, processing data, storing data or accessing a software .A task can be defined as request for a job of an application. Each task is assigned to the available servers and the completed task is returned to the user. Heterogeneous workloads may be bursty jobs that may require quick response times or long running jobs that may require intensive computing. 80% of the short duration jobs are batch jobs but 55-80% of the resources are allocated to long duration jobs that have fewer tasks compared to batch jobs.[9]. Tasks that cannot be immediately assigned on their arrival, wait in the queue to be serviced.. Task Scheduling delay is the waiting time of the task in the queue before it is being scheduled to a VM. This may be due to some low priority tasks need to wait in for a long time in the queue that can lead to starvation. Each task submitted need to be scheduled on virtual machines of different capacities. Submitted task may require different processing time and different computing resources. All the required virtual machines cannot be hosted in a single host according to the requirements of the tasks .To handle this, clustering of Virtual machines is done to service the incoming request.

In [7] jobs are classified based on short durations and long durations.tasks are clustered based on similar resource consumption. (cpu, memory, bandwidth and network capacity).

When a user submits a job ,he can demand a maximum resource required (CPU ,Memory)for each task and assign priorities, specify a scheduling class for the task with least latency sensitive(eg.batch processing tasks) to most latency sensitive tasks(eg.Web Servers) within the range 0-3. [8]

Computational Intensive tasks

Computationally intensive jobs require a lot of computations. eg.Bank deposits and withdrawal , scientific applications. Most of their execution time are spent in computation and they require only minimum amount of data. With increasing number of user access ,the task request also increases that reduces the processing speed of the system.

Data Intensive Tasks

Data Intensive tasks requires a large volumes of data to be processed and requires frequent disk reading and writing to the server. Most of their processing time are spent in I/O.eg. Reading and storing data from a database that involves a lot of I/O activity.[6]

Task waiting in the queue can be migrated from one VM to another. Computational intensive task can be migrated compared to data intensive tasks because in case of data intensive task ,the data also need to be moved and this involves additional communication overhead.[5]

Schedulers

Cloud Computing has emerged into an essential technology that allows the effective utilisation of resources distributed across the globe. Users may submit the jobs individually or submit thousand jobs collectively to access petabytes of data. Jobs submitted have many tasks that may demand data used by them. Allocating datafiles to the task is a serious problem. Thousands of jobs are submitted that need to access shared datafiles. [10]. Schedulers can be divided into job oriented systems and data-oriented system systems. In job oriented, datafiles transfer times are assumed to be negligible and are fixed in location. Here jobs are allocated to the computational nodes that reduces the overall makespan. Jobs are scheduled to the available computational nodes. In data oriented systems the jobs are fixed in location and the data are either replicated or moved to the location where the jobs can access the data. Here, data transfer time is considered to be more time consuming compared to executing the dependent jobs.

Computational Nodes(CNs)

Each CN consists of a number of homogeneous processors with same characteristics and a local storage capacity. Computational nodes are categorised based on their processing speed and the number of CPU's. the number of processors in a Computational node determines the ability of a CN to execute the moldable jobs with some degree of parallelism.

Storage Nodes(SN)

Data are stored in storage nodes. Storage nodes are either isolated or attached. Isolated storage nodes have more capacity and their only responsibility is to store the data. Attached storage nodes are they are available in the CN as local storage to store their local files and they can provide the data instantly to their CN's.

III. PROPOSED WORK

When the Users submit their request to the cloud, the task manager receives the request and classifies the request as computational intensive and I/O intensive based on their requirements. The scheduler is responsible for allocating the resources to the task. The scheduler receives the updated information from the Resource manager that monitors the resource usage of the host and virtual machines.

Let C_v be the capacity of the VM and C_t be the CPU requirement of the task. Let I_v be the I/O wait time of the VM and I_t be the I/O Wait time of the task. The task can be classified as CPU intensive or I/O Intensive based on the max(Ct/Cv,It/Iv). Two Queues are used to store the CPU intensive task and I/O intensive task before they are scheduled. The task are classified by the task manager and sent to the respective Queues.

A site with the required data may not have the required computing power and a site with the required computing power may not be having the required data, the job need to access. When both computationally intensive jobs and data intensive jobs are equally important factors for efficient utilisation of the system, so that make span and total data transfer time must be reduced. [11].

Jobs have the following characteristics Length of the job refers to the number of tasks in a job, number of CPU's required to run, list of datas required and execution time.

The response time can be minimised by transferring the input data to a data center where a large number of VMs are available. The input data located in DC_i is moved to DC_j that has large computational capabilities. All processing performed in the DC_i and the processed data will be transferred to the local site. Our model consists of

- a set of Independent tasks $N=\{T_1,T_2,.....T_n\}$
- a set of computational nodes $CN=(C_1,C_2,...C_n\}$
- a set of data files $D=(d_1,d_2...d_n\}$

Resource Scheduling problem in cloud can be defined as assigning m number of tasks/cloudlets T={ **= (T 1, T2,T3…..Tm)** } onto available n Number of resources R **R=(R1, R2 ,R3….Rn)** such that the fitness of N particular objectives **(F1,F2……Fn)** are maximised/minimised .A resource in cloud is a virtual machine defined by CPU, Memory and I/O.A task is defined by CPU Usage, task size, memory and deadline

PSO is a stochastic enhancement strategy [14] proposed by Kennedy and Eberhart in 1990's.The idea is derived from swarm model of bird behavior and fish schooling. Each particle has its own position and velocity. Each particle has an adapted value and seeks for an optimal solution in the solution space. PSO can be applied to a variety of scientific engineering applications. The conditions (1) and (2) are utilized for the speed updation and the position updation.

$$v_{id} = wv_{id} + c_1 rand()(p_{id} - x_{id}) \quad (1)$$

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

$$x_{id} = x_{id} + v_{id} \quad (2)$$

x_{id}: The present position of the particle.
v_{id}: The present speed of the particle.
p_{id}: The individual best position of the particle.

The individual best position of particle i is the best position by particle i up until now. There are two adaptations for keeping the neighbor's best vector, in particular lbest and gbest. In the nearby form, every particle monitors the best vector lbest achieved by its topological neighborhood of particles. Globally, the best vector gbest is controlled by all particles in the whole swarm.

In each generation, velocity and position of particle is updated as follows:

$$V_{k+1}^{i} = W_k V_k^{i} + c_1 r_1 (P_k^{i} - X_k^{i}) + c_2 r_2 (P_k^{g} - X_k^{i}) \quad (3)$$

$$X_{k+1}^{i} = X_k^{i} + V_{k+1}^{i} \quad (4)$$

Where, the factors r_1 and r_2 are arbitrary numbers in the vicinity of 0 and 1. c_1 and c_2 are speeding up co-efficient. W is the inertia weight

Initial population is based on the RFM score of the customers. In our proposed algorithm each particle update its best visited position(task schedule solution) and the global best position(solution) by moving forward. The fitness function evaluates the solution closest to optimum. The metrics used here is RFM Score. Customers are segregated in to groups based on how recently they have submitted the job, how frequently they are submitting the jobs and how much amount of hours they are utilising the resources based on which the cost of resources are calculated(Recency, frequency and monetary benefits).User on submitting the job enters the job queue ,their RFM Score is calculated. The user having the maximum RFM score is given the highest priority. Their user RFM score is compared with the RFM Score of the user already in the Queue. The jobs are sorted in descending order of RFM score. If it is greater than all others in the queue ,the user with the highest RFM score is backfilled in VM for execution. Customers are ranked based on the RFM score. Maximum ranking(MR) method proposed by Bentley and Wakefield [12] is used in which Individuals are ranked based on their best ranking position.

Fitness Function F(i)=(R+F+M)/3 (5)

$$fitnessvalue(FV) = \sum_{i=1}^{m} max(F(i)) \quad (6)$$

i-index of the task
m-No of Tasks

Customers are ranked according to the RFM Score and the incoming tasks are sorted in descending RFM score. Customers with the highest RFM score is given the highest priority. When task arrives with the highest RFM score compared to all other task in the Queue, task is given highest priority and backfilled. Then in the second phase the task is classified as CPU intensive or data intensive. The response time of Computationally intensive jobs will be reduced if it is assigned to less loaded VMs and in case data not available in local storage, data is pulled to the location where the job needs access to the data. On the other hand for data intensive jobs, jobs are pushed to the location where the data is available. Classification is assigning a label to it.

Binary Classification involves only two labels(0 or 1).We classified the tasks whether the data is available in local disk or remotely. Data available in local disk assigned with a label 0 and data available remotely with a label 1.The incoming jobs are classified as CPU-intensive or I/O intensive depending on the CPU, memory and network requirements .Tasks are classified as Ready Tasks (R) and Not Ready Tasks(NR).The tasks with available data in the local disk is called as Ready tasks. The tasks with data available in remote place are called as Not ready tasks. i.e the tasks is not ready for execution immediately as the data is not available. The tasks with the available data in the local disk are executed in the lower speed VMs. The tasks with data not available are sent to the location of VM where data is available. if that VM is freely available, the task is executed and results are transferred. If that VM is overloaded, data are pulled to the location where the task is executed.

First Phase-RFM Algorithm
1.User Submits the task
2.Initialise the Particles, Position and Velocity
3.Initialise Pbest and Gbest
4.Calculate the fitness function according to equation 6.
5.Update Velocity and Position
6.Update Pbest and Gbest.
7.End

Second Phase-Classification Algorithm

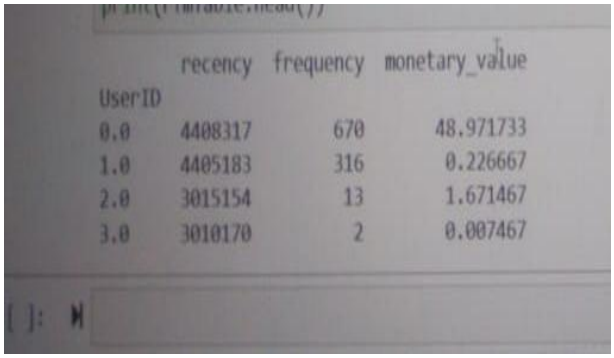
Classify the tasks as computationally intensive and Data Intensive

1.If the incoming task classified as Computationally Intensive
  (i)Assign the task to less loaded VMs
2.If the incoming task is data intensive
  (i)if data available in local disk of VM
    (i) Assign the task to that VM
  else
   (i)Check for the VM where the data is located
   (ii)if the data available VM is less loaded
    (i)Assign the task to that VM
   else if it is overloaded
    (ii)Replicate the data to the location of the task and execute it.

IV. EXPERIMENTAL RESULTS

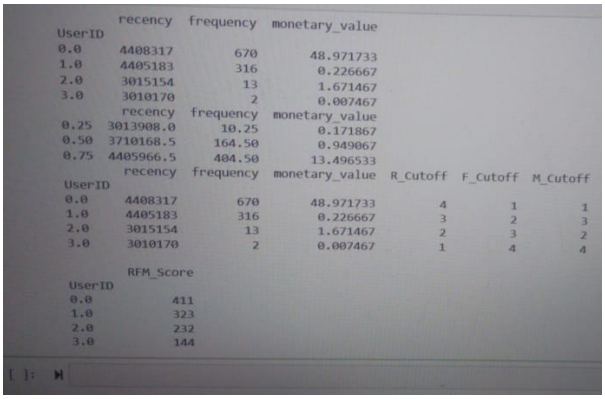
The RFM-PSO algorithm implemented using Cloudsim. Cloudsim is a simulation framework that consists of Datacenter(DC),Hosts(H),Virtual Machines(VMs),brokers and cloudlets[15].Datacenter consists of host of similar or different configurations. Host is represented using Host Id, processing power(MIPS) ,processing elements(PE),RAM ,Bandwidth and storage. Creation of VMs, Provisioning and Managing the VMs, Destroying the VMs are the responsibilities of Host. Virtual machines are represented by Virtual machine Id, Bandwidth, RAM, processing elements (PE)and processing Power.

Cloudlet represents the tasks in cloudsim. Broker acts a mediator for the users and Cloud Providers. The proposed algorithm was written in Python and simulated Cloudsim Toolkit 3.0.3.



UserID	recency	frequency	monetary_value
0.0	4408317	670	48.971733
1.0	4405183	316	0.226667
2.0	3015154	13	1.671467
3.0	3010170	2	0.007467

Fig. 1 RFM Calculation



UserID	recency	frequency	monetary_value	R_Cutoff	F_Cutoff	M_Cutoff
0.0	4408317	670	48.971733	4	1	1
1.0	4405183	316	0.226667	3	2	3
2.0	3015154	13	1.671467	2	3	2
3.0	3010170	2	0.007467	1	4	4

	recency	frequency	monetary_value
0.25	3013908.0	10.25	0.171867
0.50	3710168.5	164.50	0.949067
0.75	4405966.5	404.50	13.496533

UserID	RFM_Score
0.0	411
1.0	323
2.0	232
3.0	144

Fig. 2 RFM Score

The metrics used to measure the performance are based on reducing makespan, reducing the cost and maximising the utilisation of resources.

1. Makespan can be described as the longest time taken to complete the last task.

Makespan=Max(LF_t)

2. Execution Cost is the amount paid by the customers to cloud providers for the execution of task.

$$EC = \sum_{i=1}^{m} \sum_{j=1}^{n} VM_c * M_c * T_{ij} \quad (7)$$

VM_c-Amount spent on the VM per unit time

M_c-Migration cost of the task

T_i-amount of time Taken to execute the task

$VM_c=N/P_i$

N=Number of jobs in the Queue

P_i=Processing Power of the VM

3. Resource Utilization -Maximizing the utilization resources without keeping them idle to make profit.

$$Util = \sum (j.RR * j.RT)/(FT - ST) \quad (8)$$

RR-Requested Resources

RT-Runtime

ET-Finish Time

ST-Submit Time

N - number of resources.

j-jobs

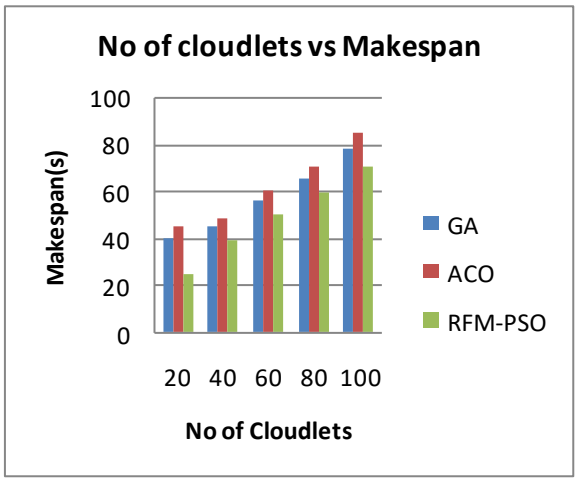


Fig. 3 No of Cloudlets Vs Makespan

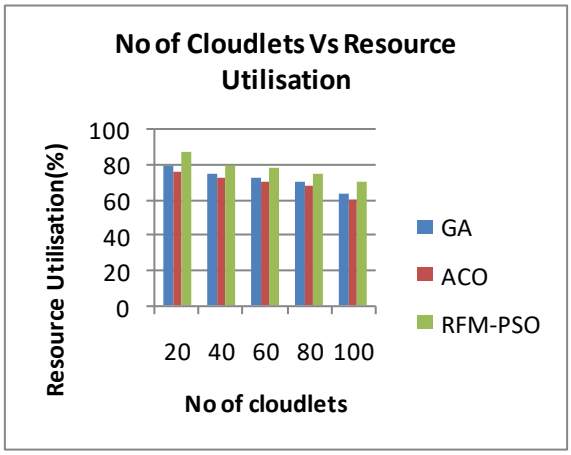


Fig. 4 No of Cloudlets Vs Resource Utilisation

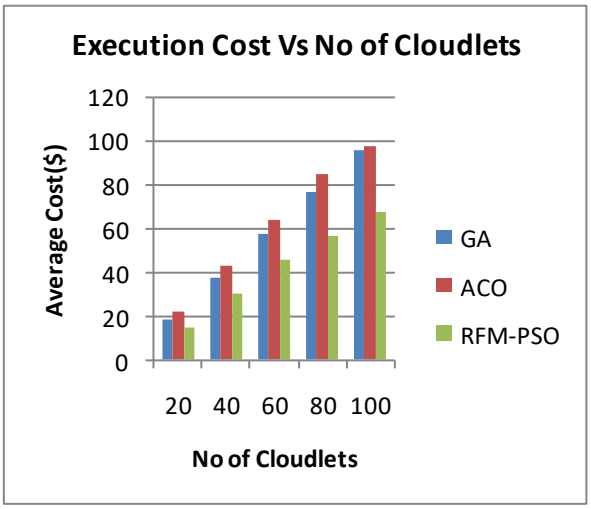


Fig. 5 Execution Cost Vs No of Cloudlets

The proposed RFM-PSO algorithm is compared with GA and ACO. The size of population in all cases are set to 100. The number of iterations on all cases set to 100.Fig 3.Shows the comparison of makespan among the RFM-PSO, ACO and GA.

Fig.4 shows the overall resource utilisation between the three algorithms RFM-PSO, GA, ACO. Fig 5.Shows the comparison of execution cost

V. CONCLUSION

The proposed method helps the cloud provider to increase the income by retaining the customers. The frequent customers also benefits by getting the task executed faster and in reduced waiting time. Cloud provider can provide some special offers to customers with highest RFM score. By giving preference to customer with the RFM Score the customer waiting time has been reduced and user experience has been improved. The makespan, resource utilisation has been better when compared to other algorithms. The outcomes of the algorithms shows that RFM-PSO performs better compared to other algorithms.

REFERENCES

1. Weiss A., "Computing in the Clouds," Net Worker on Cloud computing: PC functions move onto the web, vol. 11, no. 4, pp. 16-25, 2007
2. Richardson, P.: The secrete life of bats. http://www.nhm.ac.uk
3. Parikh SM. A survey on cloud computing resource allocation techniques. Nirma University International Conference on Engineering (NUiCONE); Ahmedabad. 2013. p. 1–5.
4. Anuradha VP, Sumathi D. A survey on resource allocation strategies in cloud computing. International Conference on Information Communication and Embedded Systems (ICICES); Chennai. 2014. p. 1–7.
5. Handbook of Cloud Computing, "Data-Intensive Technologies for Cloud Computing," by A.M. Middleton. Handbook of Cloud Computing. Springer, 2010
6. A Multiqueue Interlacing Peak Scheduling Method Based on Tasks' Classification in Cloud Computing Liyun Zuo, Shoubin Dong, Lei Shu, *Senior Member, IEEE,* Chunsheng Zhu, *Student Member, IEEE,*and Guangjie Han, *Member, IEEE*
7. A.K. Mishra, J.L. Hellerstein, W. Cirne, and C.R. Das, "Towards Characterizing Cloud Backend Workloads: Insights from Google Compute Clusters," ACM SIGMETRICS Performance Evaluation Rev., vol. 37, pp. 34-41, Mar. 2010.
8. Q. Zhang, M. F. Zhani, R. Boutaba, and J. H. L. Hellerstein,"HARMONY: Dynamic heterogeneity? Aware resource provisioning in the cloud," in *Proc. 33rd IEEE Int. Conf. Distrib. Comput. Syst.*, 2013, pp. 511–519.
9. M. Schwarzkopf and A. Konwinski, M. Abd-El-Malek, and J. Wilkes,"Omega: flexible, scalable schedulers for large compute clusters," in *Proc. 8th ACM Eur. Conf. Comput. Syst.*, 2013, pp. 351–364.
10. Holtman K. HEPGRID2001: a model of a virtual data grid application. In: Hertzberger LO, Hoekstra AG, Williams R, editors. HPCN Europe 2001.
11. McClatchey R, Anjum A, Stockinger H, Ali A, Willers I, Thomas M. Data intensive and network aware (DIANA) grid scheduling. Journal of Grid Computing 2007;5:43–64.
12. Bentley, P.J., Wakefield, J.P.: Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. In: Chawdhry, P.K., Roy, R., Pant, R.K. (eds.) Soft Computing in Engineering Design and Manufacturing. Part 5, June 1997, pp. 231–240. Springer, London (1997) (Presented at the 2nd On-line World Conference on Soft Computing in Design and Manufacturing (WSC2))
13. J. Blazewicz, Scheduling in Computer and Manufacturing Systems, Springer-Verlag New York, Inc., Secaucus, NJ,USA, 1996
14. J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of ICNN'95 - International Conference on Neural Networks, 1995, vol. 4, pp. 1942–1948
15. T. Goyal, A. Singh, and A. Agrawal, "Cloudsim: simulator for cloud computing infrastructure and modeling," Procedia Eng., vol. 38, no. Supplement C, pp. 3566–3572, 2012.