International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-5 Issue-6, August 2016

A Particle Swarm Optimization Approach With Migration for Resource Allocation in Cloud

Aleena Xavier T, Rejimoan R.

Abstract— Cloud computing is an emerging technology. The main motivation behind the proposed work is to design a Cloud Broker for efficiently managing cloud resources and to complete the jobs within a deadline. The proposed approach intends to achieve the objectives of reducing execution time, cost and workload based on the defined fitness function. The work is simulated in CloudSim and the results prove the effectiveness of the proposed work. A better allocation was achieved when all of the three factors were considered. The analysis of work was done by comparing one of the previous works where only time and cost were the objectives. By plotting a graph against Response time and deadline and another graph depicting the relation between the idle time and deadline this result has been proved.

Keywords—Resource allocation, Job scheduling, Cloud Computing, IaaS, Particle Swarm Optimization

I. INTRODUCTION

Cloud computing is a technology that provides the users with requested services dynamically and efficiently. These services have three major divisions, Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). The most beneficial users of SaaS are those who cannot afford computing requirements of sophisticated softwares. PaaS mainly focus on developers and provide them with the application frameworks for software development. IaaS on the other hand is based on virtualization of resources, it provides with the most basic computational resources to users. Efficient delivery of services and its flexibility has made cloud a good choice in terms of computing technologies. This environment is made up of three components Cloud service provider (CSP), Cloud user and Cloud broker. The cloud users request for some software, hardware or QoS parameters from CSP, where CSP manages the heterogeneous cloud nodes to provide these services on time. Cloud brokers do the most tedious task in this resource assignment, where the satisfaction of the user is depended on how efficient this resource allocation is. To meet the objectives of cloud users and cloud service providers we have proposed a Particle Swarm Optimization (PSO) based approach.

Manuscript published on 30 August 2016.
***** Correspondence Author (s)
Aleena Xavier T, Department of Computer Science, Sree Chitra Thirunal College of Engineering, Thiruvananthapuram (Kerala) - 695018. India.
Rejimoan R, Department of Computer Science, Sree Chitra Thirunal College of Engineering, Thiruvananthapuram (Kerala) - 695018. India.

Here an additional job scheduling algorithm is integrated with PSO so that the selection of resources is tailored for a particular job. Our proposed algorithm has completed the jobs within the deadline in reduced time, cost and keeping a balanced workload. Resource allocation in cloud is complex due to its dynamicity; it is usually defined as a NP complete problem. They are solved by Heuristic or Meta Heuristic algorithms in which some are biologically inspired algorithms [1]. PSO comes under the category of swarm intelligence. Eberhart and Kennedy [2] in 1995 derived PSO based on the social behaviours of bird flocking and fish schooling. Many previous works have been done using PSO for resource allocation in cloud. Most of them focused on time, cost or energy. Our proposed algorithm focuses on embarrassingly parallel jobs. They are given an individual tagging by deadline and then the algorithm select a suitable resource so that it can be done within minimal time, cost and with a balanced workload. This approach was simulated using Cloud Sim and was proved efficient than existing systems.The paper is organized as section I gives an introduction to the work, section II includes the related works, section III elaborates the methodology, section IV describes the simulation and results that validate the proposed work and in section V the work is concluded.

II. RELATED WORKS

Different resource allocation strategies were used in cloud. Aman kumar et al. [3] proposed an optimal allocation using improved genetic algorithm (IGA). This approach uses shorter genes and dividend policy as fitness function. Chromosomes are half in size than those used in grid computing as the computing nodes are not specified with jobs. Here capital gain is the fitness term and dividend represents the overall state of chromosomes that are duplicated. Instant request are then allocated according to the fitness function and the basic genetic operations. This work is compared with first fit and round robin algorithms. M. c. D. Pandit et al. [4] explains resource allocation as a multi parameter bin packing problem. A request is said to be served if the request requirements can be allocated to resource bin. Even when a single parameter is completely filled then that bin or resource is no more useful. By simulated annealing initially random request are allocated, thereafter the cost is calculated and the costlier is swapped with a random low cost job. The major drawback is that unwanted allocations are entertained and the temperature value is an assumption from the beginning.

Suraj Pandey et al. [5] describe a method based on particle swarm optimization and it is used for data intensive workflow application.

Here the average computational cost is calculated by running the jobs in a known resource. Initial mapping is done without considering the relations between the jobs. The method optimizes individual task and gives a better distribution of workflow. When compared to best resource selection technique it proved three times better. Chandrashekar S.Pawar et al. [6] in this work the resource allocation is based on priority. When higher priority jobs come the lower priority is preempted and in equal priority a new virtual machine is allotted. The requirements of the tasks are specified by SLA. This approach makes use of cloud list scheduling and cloud minimum scheduling algorithms in preempting tasks. Biao Song et al. [7] proposed a resource allocation strategy in cloud where the jobs are basically divided into light load and heavy load. The objective of allocation is different for both categories. In light load the focus is on providing QoS and in heavy load resource utilization is given importance.

Eun-Kyu Byuna et al. [8] proposed partition based time balanced algorithm for resource allocation with minimized cost. The core work focus on minimizing the cloud instances at each time slot. M. Mezmaz et al [9] used a hybrid genetic algorithm where the main objectives of the work were to minimize the makespan and energy consumption. O.O. Sonmez et al. [10] on the other hand proposed a new concept of economy driven job allocation. Work validates that it is a good choice for sequential workflow applications.

S. Chaisiri et al. [11] proposed optimal cloud resource provisioning algorithm based on integer programming. Cost is the major objective in this work. Here there are two types of cost that is summed up. They are, cost for on demand plan and cost for advance reservation plans. Resources are allocated to the one with minimum cost. Tasgetiren et al. [12] work proves that PSO makes, 57 out of 90 well known sequential strategies better.

III. METHODOLOGY

In the existing system jobs and Resources are allocated and scheduled so that the Cloud Users can complete their jobs with minimal time (within the deadline) and cost. PSO is proved to be better than Ant Colony Optimization [13], Genetic Algorithm [14] and Rank Based Allocation in previous work [15]. In this system the execution time is added to workload whenever a particular job is assigned to that resource. This workload is subtracted from the deadline and the remaining time is calculated. Only if the execution time of the new job is less than the remaining time only then it is allocated to that resource. In this scenario when there is high performance node there is a chance that it is able to do about 80% of the total job in a schedule. When this happens the other nodes stay idle and the workload on this high performance computer is far more compared to others. So in brief the concept of balanced workload for resources is not considered here.

In a schedule consider there are ten different jobs namely J_1, J_2...J_{10} and the high performance node (R_j) that can do eight of these jobs. That is when J_1 is allocated; the workload of R_j is added with the execution time of J_1. Now while considering J_2 to J_8 the remaining time is less than execution time of each of these jobs, this is because R_j have a higher MIPS than any other nodes in cloud. In this situation the algorithm in turn allocates all of these eight jobs to R_j. The objective of our work is to include a workload parameter so that resource allocation is done with minimum cost, time (within deadline) and in balanced workload. Cloud users request for a set of jobs $J_1,J_2 \ldots J_i$ in a schedule S_i, these jobs are independent in nature and each of them request for some resources like RAM, Bandwidth, Processor speed, Memory etc. Initially heterogeneous nodes are registered to Cloud broker. When a request or job arrives then cloud broker invokes the first module that is the request management module. Here the jobs sorted based on deadline and then a matched list of resources is generated for each job. This is done by comparing the requirements of the jobs and the resources available in the registered nodes. Each of the nodes in the matched list is capable of doing the job within the deadline. This list is then sent to the Resource allocation module. The major work of this approach is done in this module. Here the execution time, cost and workload is calculated initially and then the EPSO is invoked for resource allocation. The fitness function and the velocity function include three objectives that are to minimize the execution cost, time and workload.

$$ExET_{Ji}=ET_{Ji}+TT_{Ji}$$

$$ET_{Ji}=JLen_{Ji}/PC_{Rj}$$

$$TT_{Ji}=FS/2*BW_{Rj}$$

The second objective is the minimization of the total execution cost ExEC of jobs $J_1,J_2 \ldots J_i$ in a schedule S_i and it is calculated by equations

$$ExEC_{Ji} = EC_{Ji}+TC_{Ji}$$

$$EC_{Ji} = ET_{Ji}Cost$$

$$TC_{Ji} = TT_{Ji}Cost$$

The Final objective is to have a balanced workload of jobs $J_1,J_2...J_i$ in a schedule S_i and it is calculated based on equation

$$WL_{Rj} =WL_{Rj} +ExEC_{Ji}$$

The fitness function F is defined to solve the above-said scheduling multi-objective optimization problem in Cloud and it is given by equation

$$F = w_1A_{JiRj}ExET_{Ji}+w_2A_{JiRj}ExEC_{Ji}+w_3A_{JiRj}WL_{Rj}$$

$ExET_{Ji}$	Expected execution time
ET_{Ji}	Execution time of J_i on Resource R_j
TT_{Ji}	Transfer Time of J_i on Resource R_j
$JLen_{Ji}$	Job Length of J_i
PC_{Rj}	Processing Capability of Resource R_j
FS	File Size of J_i
BW_{Rj}	Band Width of R_j
$ExEC_{Ji}$	Expected execution Cost
EC_{Ji}	Execution Cost of J_i on Resource R_j
TC_{Ji}	Transfer Cost of J_i on Resource R_j
WL_{Rj}	Workload of Resource R_j
w_1,w_2,w_3	Weights assigned to control the fitness function
A_{JiRj}	Variable is assigned 1 if the job is allocated to resource Rj and zero otherwise

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

Initially heterogeneous nodes are registered to the cloud service provider. When a user provide with a job then it is the duty of cloud broker to find the best resource to do that job. Cloud broker initiates the process by invoking the matchmaking algorithm this is shown in Algorithm 1. The jobs are sorted based on deadline and then for each job J_i each resource R_j is compared so that a Resource list that matches the requirements of each job is generated and returned. The resources in the matched list are capable of doing the job within the deadline. This list is given to the Resource allocation algorithm that is shown in Algorithm 2.

Algorithm 1: Matchmaking Algorithm

Ji.RList: Resource List of job J_i

Input: Set of jobs in a schedule S_i
Output: List of matched Resources, capable of doing each job

1. Sort the jobs based on deadline
2. For Each job J_i in schedule S_i
2.1. J_i.RList=∅
3. For each resource R_j
4. Check if
4.1. J_i.RAM=R_j.RAM AND
J_i.Storage=R_j.Storage AND
J_i.Bandwidth=R_j.Bandwidth
4.2. then J_i.RL=J_i.RL+R_j
5. End
6. End
7. Return RL

In Resource Allocation algorithm initially the list of jobs assigned to a resource is a null set. Then Excepted execution time, Excepted execution cost and Workload calculations are done for each resource in the matched list assuming that job J_i is assigned to it. These values are passed on to the Enhanced Particle swarm Optimization algorithm (EPSO) which is shown in Algorithm 3. The Actual workload is calculated by adding up the execution time of the job with the workload that is already assigned to the resource. This calculation is done only after the resource is returned from EPSO.

Algorithm 2: Resource Allocation Algorithm

ωR_j : List of jobs assigned to R_j
AW.R_j : Actual Workload of R_j

Input: Matched Resource List
Output: Resource that do the job with minimum time, cost and in balanced workload

1. For Each job J_i in schedule S_i
2. ωR_j= ∅
3. For each resource R_j
3.1 ET_{Ji}=$JLen_{Ji}$/PC_{Rj}
3.2 TT_{Ji}=FS/2*BW_{Rj}
3.3 $ExET_{Ji}$=ETJ_i+TT_{Ji}
3.4 EC_{Ji}=ET_{Ji}Cost
3.5 TC_{Ji}=TT_{Ji}Cost
3.6 $ExEC_{Ji}$=EC_{Ji}+TC_{Ji}
3.7 WL_{Rj}=WL_{Rj}+$ExEC_{Ji}$
4. End
5. R_j=EPSO () //Resource that can do the job with minimum time, cost and in balanced workload is returned to R_j
7. AW.R_j=AW.R_j+ETJ_i
8. ωR_j=ωR_j ∪ R_j

In EPSO the Personal best value of the resource R_j and the Global best value of the swarm is initialized to infinity. The Personal best position of a particle in the matched list of resources for a particular job is initialized to zero in all three dimensions, similarly Global best position is also initialized to zero at each dimension. Velocity is initialized to a random value between -1 and 0.

Each resource is assigned with its Expected execution time, Expected execution cost and workload parameters and thereby defining is position in the swarm. Then for each resource the fitness value is calculated based on the three objectives, ie Execution Time, Execution Cost and a balanced workload. The Personal best value is updated with the fitness or the Personal best value whichever is less. Based on the update of Personal best value the Personal best position is also updated. Before moving on to Global best value update the algorithm checks if the execution time of job J_i in resource R_j is less than the remaining time the resource can run within the deadline. Only then the Global value is updated similar to Personal best value. The motion of particles are determined by a term called velocity and it is calculated based on the amount the particle have to travel from its current position to local and global best positions.

Algorithm 3: Enhanced Particle Swarm Optimization Algorithm

Pbest Position: Personal Best Position of Resource R_j
Gbest Position: Global Best Position of the swarm
$F(R_j)$: Fitness of R_j
ET_{Ji}: Execution Time of job J_i
RT_{Rj}: Remaining Time of Resource R_j
$\overrightarrow{v(t+1)}$: Velocity at the time t+1
$\overrightarrow{XRJ(t+1)}$: Position of particle R_j
Pbest value of R_j: Personal best Value of resource R_j
Gbest value: Global Best Value of the swarm

Input: Matched Resource List
Output: Resource that do the job with minimum time, cost and in balanced workload

1. For Each Resource R_j in matched list J_i.RList
1.1 Pbest value of R_j=1
1.2 Gbest value of the Swarm=1
1.3 Pbest Position of R_j having three dimensions=(0,0,0)
1.4 Gbest Position for the swarm=(0,0,0)
1.5 Velocity for each particle R_j in the swarm=-rand-1,0
1.6 Each resource R_j in the swarm in the matched list=($ExET_{Ji}$, $ExEC_{Ji}$, WL_{Rj})

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

2. End
3. Repeat until maximum iteration
   3.1 For each particle R_j in the matched list J_i.RList
   3.2 $F(R_j)=w_1A_{JiRj}ExET_{Ji}+w_2A_{JiRj}ExEC_{Ji}+w_3A_{JiRj}WL_{Rj}$
   3.3 Rj.Pbestval=minF(Rj),Rj.Pbestvalue
   3.4 Rj.Pbestpos=($ExET_{Ji}$, $ExEC_{Ji}$, WL_{Rj})
   3.5 if $ET_{Ji} < RT_{Rj}$
      3.5.1 Gbestval=minF(Rj),Gbestva
      3.5.2 Gbestpos=($ExET_{Ji}$, $ExEC_{Ji}$, WL_{Rj})
      3.5.3 End
   3.6 End
4. For each particle Rj in the matched list Ji.RList

$$4.1\ \overrightarrow{v(t+1)}=\overrightarrow{v(t)}+\omega\left(rand+C_1\left(\overrightarrow{R_j.Pbestpos}-\overrightarrow{X_{R_j}}\right)\right)+\omega(rand+C_2(\overrightarrow{Gbestpos}-\overrightarrow{X_{R_j}})$$

$$4.2\ \overrightarrow{X_{R_j}(t+1)}=\overrightarrow{X_{R_j}}+\overrightarrow{v_{R_j}(t+1)}$$

   4.3 End
5. End

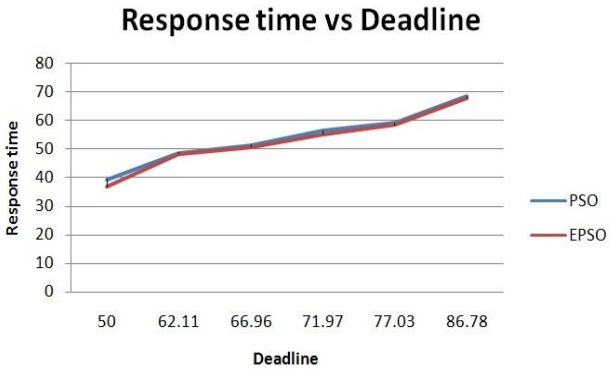
EPSO gives a resource that does the job with minimum Execution time (within the deadline), Execution cost and with balanced workload. This is returned to Resource Allocation algorithm and then the actual workload of the resource is updated with the execution time of assigned job.

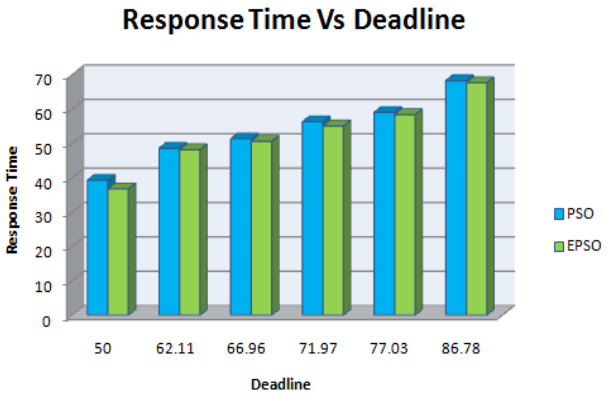
IV. REULTS AND DISCUSSION

The work is simulated in Cloud Sim and the results prove the effectiveness of the proposed work by minimizing the completion time, cost and workload. A better allocation was achieved when all of the three factors were considered. The analysis of work was done by comparing one of the previous works where only time and cost were the objectives.
This is measured by two metrics namely response time of cloud and the idle time of resources.

A. Response Time of Cloud

Response Time is the time taken by the node to process the job requested by the user and sent out its first response. It has two parts, first is the time taken to process the job and the second is the time taken for data transfer from the server to client.

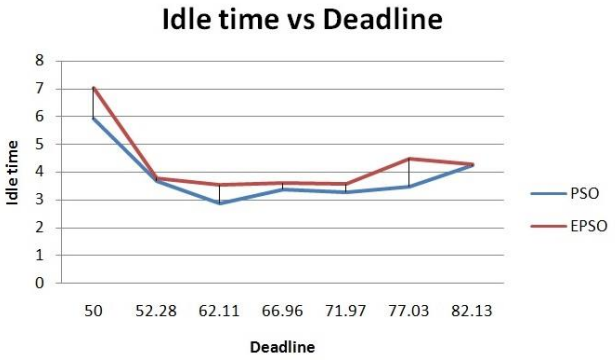


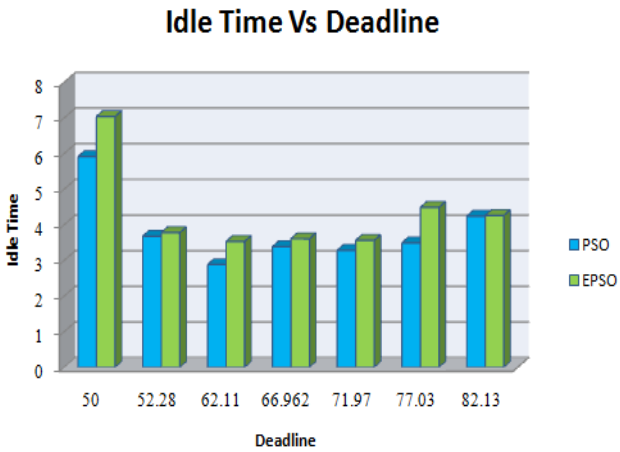


The graph depicts a slight reduction in response time of cloud while using Enhanced Particle Swarm Optimization. This explains that the time to give an initial response to user has reduced and this can in turn increase the satisfaction on the user. Amount of reduction in response time indicate that EPSO does a better allocation than PSO. By reduced response time more and more jobs can be completed within the deadline.

B. Idle Time of Nodes

Idle time is the time a node waits until it is allocated with another job. While using EPSO there is increase in idle time than while PSO. During idle time the nodes goes on a low power mode. That is with fewer numbers of nodes a faster execution of jobs are obtained by using EPSO. Adding workload as a new objective to existing system has positively transformed the allocation in cloud.





Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

V. CONCLUSION

The integration of deadline-based job scheduling with EPSO-based resource allocation mechanism prioritizes the user application requests based on deadline and selects the cloud resource that complete the jobs with minimal time, cost and with balanced workload. By considering deadline as one of the important factor a better user satisfaction level was achieved. The proposed research work is evaluated by simulating it with CloudSim. The performance metrics calculated from the simulation experiments are Response time of cloud and Idle time of node. It has been observed that EPSO gives a better allocation of resources by reducing the response time and increase in Idle time implies that with fewer numbers of nodes a faster execution of jobs was obtained by using EPSO. Proposed work was proved to be better than existing PSO algorithm for resource allocation. The future work can explore further integration of semantic resource discovery and Service Level Agreement (SLA) to improve the job mapping process and Quality of Service (QoS).

REFERENCES

1. S. Binitha, S.Siva Sathya, A survey of bio inspired optimization algorithms, Int.J. Soft Comput. Eng. (IJSCE) (ISSN: 2231-2307) 2 (2) (2012).
2. J. Kennedy, R. Eberhart, Particle swarm optimization (PSO), in: Proc. IEEE International Conference on Neural Networks, Perth, Australia, 1995, pp. 1942–1948.
3. Aman kumar, Emmanueel S.Pilli and R.C.Jshi," An efficient framework for resource allocation in cloud computing" ,in IEEE 4th ICCCNT - 2013, Tiruchengode, India
4. M. c. D. Pandit and N. Chaki, "Resource allocation in cloud computing using simulated annealing," IEEE applications and innovations in mobile computing, 2014.
5. Suraj Pandey, Linlin Wu, Siddeswara Mayura Guru, Rajkumar Buyya, "A particle swarm optimization-based heuristic for scheduling workflow applications in cloud computing environments, in: AINA'10 Proceedings of the 2010 24th IEEE International Conference on on Advanced Information Networking and Applications
6. Chandrashekar S.Pawar and Rajnikant B.Wagh, Priority Based Dynamic Resource Allocation in Coud Computing, International Symposium on Cloud ans Services Computing, 2012, pp.1-6.
7. Biao Song, Mohammad Mehedi Hassan, Eui-nam Huh, A novel heuristicbased task selection and allocation framework in dynamic collaborative cloud service platform, in: CloudCom 2010, pp. 360–367
8. Eun-Kyu Byuna, Yang-Suk Keeb, Jin-Soo Kimc, Seungryoul Maeng, Cost optimized provisioning of elastic resources for application workflows, Future Gener. Comput. Syst. 27 (2011) 1011–1026.
9. M. Mezmaz, Choon Lee Young, N. Melab, E.-G. Talbi, A.Y. Zomaya, A bi-objective hybrid genetic algorithm to minimize energy consumption and makespan for precedence-constrained applications using dynamic voltage scaling, in: 2010 IEEE Congress on Evolutionary Computation, CEC, 18–23 July 2010
10. O.O. Sonmez, A. Gursoy, A novel economic-based scheduling heuristic for computational grids, Int. J. High Perform. Comput. Appl. 21 (1) (2007) 21–29.
11. S. Chaisiri, Bu-Sung Lee, D. Niyato, Optimization of resource provisionin cost in cloud computing, IEEE Trans. Serv. Comput. 5 (2) (2012) 164–177.
12. M.F. Tasgetiren, Y.-C. Liang, M. Sevkli, G. Gencyilmaz, A particle swarm optimization algorithm for makespan and total flowtime minimization in the permutation flowshop sequencing problem, European J. Oper. Res. 177 (3) (2007) 1930–1947
13. M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Trans. Syst. Man Cybern. B 26 (1) (1996) 29–41.
14. Genetic Algorithm, J.H. Holland, Genetic algorithms and the optimal allocation of trials, SIAM J. Comput. 2 (2) (1973) 88–105.
15. Thamarai Selvi Somasundaram, Kannan Govindarajan, "CLOUDRB: A framework for scheduling and managing High-Performance Computing (HPC) applications in science cloud", Future Generation Computer Systems 34 (2014) 47–65.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)