American Journal of Applied Sciences 11 (6): 872-877, 2014
ISSN: 1546-9239

doi:10.3844/ajassp.2014.872.877 Published Online 11 (6) 2014 (http://www.thescipub.com/ajas.toc)

SIMULATED ANNEALING APPROACH TO COST-BASED MULTI- QUALITY OF SERVICE JOB SCHEDULING IN CLOUD COMPUTING ENVIROMENT

[1]Monir Abdullah and [2]Mohamed Othman

[1]Department of Information Technology, Thamar University, Thamar, Yemen
[2]Department of Communication Technology and Network, Institute of Mathematical Science, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor D.E., Malaysia

Received 2014-01-17; Revised 2014-02-05; Accepted 2014-03-18

ABSTRACT

Cloud computing environments facilitate applications by providing visualized resources that can be provisioned dynamically. The advent of cloud computing as a new model of service provisioning in distributed systems, encourages researchers to investigate its benefits and drawbacks in executing scientific applications such as workflows. One of the fundamental issues in this environment is related to task scheduling. Cloud task scheduling is an NP-hard optimization problem and many meta-heuristic algorithms have been proposed to solve it. A good task scheduler should adapt its scheduling strategy to the changing environment and the types of tasks with minimum scheduler execution time. A Genetic Algorithm (GA) for job scheduling has been proposed and produced good results. The main disadvantage of GA algorithm is time consuming problem. In this study, a novel Simulated Annealing (SA) algorithm is proposed for scheduling task in cloud environment. SA based approach produced comparative result in a minimal execution time.

Keywords: Simulated Annealing Algorithm, Cloud Computing, Quality of Service

1. INTRODUCTION

Cloud computing (Sridhar, 2009) is a recent trend technology, where user can rent software, hardware, infrastructure and computational recourse as per user basis. Users can submit their job into cloud for computational processing or leave their data in cloud for storage. Different users has different Quality of Service (QoS) requirement. Cloud scheduler must be able to schedule the jobs such a way that cloud provider can gain maximum benefit for his service and QoS requirement user's job is also satisfied.

As job scheduling problem is a kind of combinatorial problem, it cannot be formulated as a linear programming or it impossible to find the globally optimal solution by using simple algorithms or rules. It is well known as NP-complete problem. It depends on size of problem. To solve this problem some kind of branch and bound and other approximation method has been developed, however the obtained results is really unpredictable and require a lots of time, which is almost non applicable in cloud environment.

Furthermore, the objective functions of cloud computing may be too complex, depending on the business orientation of cloud environment, which is impossible to solve in linear time by using conventional scheduling algorithm. GA and IGA approaches have been developed, which provide a cost-based multi QoS scheduling in cloud (Dutta and Joshi, 2011). However, the disadvantage of GA approaches is time consuming problem. In this study we try to adapt SA approach to the problem with proposing good initial solution.

Moreover, in order to realize the full potential of the cloud platform, an architectural framework for efficiently coupling public and private clouds is necessary.

Corresponding Authors: Monir Abdullah, Department of Information Technology, Thamar University, Yemen-
Mohamed Othman, Department of Communication Technology and Network,
Institute of Mathematical Science, Universiti Putra Malaysia, 43400 UPM Serdang, Selangor D.E., Malaysia

As resource failures due to the increasing functionality and complexity of hybrid cloud computing are inevitable, a failure-aware resource provisioning algorithm that is capable of attending to the end-users QoS requirements is paramount (Javadi *et al*., 2012).

The rest of the study is organized as follow: Section 2 gives with related works; in section 3, a cloud scheduling environment and a set of mathematical equations has been developed to formulate the problem; in section 4 we gives SA based approach to address this problem; section 5 gives the experimental results; the last part concludes with future work.

2. MATERIALS AND METHODS

In cloud computing, end users do not own any part of the infrastructure. The end-users simply use the services available through the cloud computing paradigm and pay for the used services. The cloud computing paradigm can offer any conceivable form of services, such as computational resources for high performance computing applications, web services, social networking and telecommunications services (Mezmaza *et al*., 2011).

Job scheduling in cloud has an excellent history in research area. Alot's of algorithms have been proposed for this scheduling problem. It includes optimistic differentiated job scheduling (Li, 2009), multiple QoS scheduling (Xu *et al*., 2009), fuzzy bee colony (Zhao *et al*., 2009) and lots more. GA has been successfully applied in heterogeneous system (Mukherjee and Sahoo, 2009), grid computing (Daoud and Kharma, 2006). Most of these researches assume that each job has fixed amount of execution time, but it is not the caseof real world cloud computing. The time of execution depends on the performance of his service and QoS requirement user's job are also satisfied. GA has been developed, which provide a cost-based multi QoS scheduling in cloud (Dutta and Joshi, 2011) and produced good results. In GA, a collection of chromosomes is generated as an initial population based on random Initialization.

IGA approach is also proposed (Abdullah and Othman, 2012) with good initial solution and produced good results. The initial population is generated based on application hint (GA-Hint). The basic idea behind the method GA-Hint is using application specific information (e.g., the ratio of job expected instruction count to the cost of the instruction in the process to generate a seed chromosome of good quality (high fitness value).

The disadvantage of GA approaches is time consuming. Thus in our paper we try to adapt a new optimization algorithm to the problem. When SA was first proposed, it was mostly known for its effectiveness in finding near optimal solutions for large-scale combinatorial optimization problems (Manikas and Cain, 1996) such as the Traveling Salesperson Problem (TSP), buffer allocation in production lines (Spinellis and Papadopoulos, 2000; Mezmaza *et al*., 2011) and chip placement problems in circuits (Ingber and Rosen, 1992) (finding the layout of a computer chip that minimizes the total area). But recent approaches of SA have demonstrated that this class of optimization approaches could be considered competitive with other approaches for solving optimization problems (Abdullah *et al*., 2006).

3. SCHEDULING ENVIRONMENTS AND COST MODELS

3.1. Scheduling Model Environment

The proposed model of scheduling environment mainly consists of five components: A set of users (cloud customers), Preprocessing unit and task classifier, data center/executer, datacenter manager and Job scheduler. More details can be found in (Abdullah and Othman, 2012; 2013).

3.2. Cost Model

Let consider the following cost factor: α_i be the cost per instruction for processing unit i and βj indicates the delay cost ofjob j. Suppose, N machines with M jobs and assign these M jobs in to N machines (M = N), in such an order that following condition can be satisfied:

Form user side, finish Time (Tf) must be less than the worst case completion Time (Twcc), scheduling must be done such way to preserve QoS and to avoid possible starvation:

$$Tf \leq Twcc$$

This condition must be satisfied anyhow, otherwise the job is considered as a failure job and the corresponding scheduling is illegal.

From cloud provider side, to minimize the cost spend on the job: Suppose ith machine is assigned to jth job. Then the cost for execution job j is ICj* α_i (while IC is the instruction count). Let Ψj, estimated delay cost for job j, can be defined as:

$$\Psi j = \begin{cases} 0 & if\ Td \geq Tf \\ \beta * (Tf - Td) & if\ Td < Tf \end{cases}$$

where, Td is the deadline for job j and Tf is the estimated finish time, when job j is assigned to process unit i. Thus overall cost to execute all N jobs can be given by:

$$\varsigma = \sum_{i=1}^{N}(ICJ * ai) + \Psi j$$

Thus, cloud provider's aim is to schedule jobs (i.e., find a permutation: N->M) such a way which minimize the value of ζ:

$$Min(\varsigma) = \left[\left| Min \sum_{i=1}^{n}((ICj * ai) + \Psi j) \right. \right]$$

As there are N number of jobs and M number machines and assuming that all machines are capable to perform any job, then there are total N*M numbers of way to assignment. Thus this problem is a kind of NP-Complete problem (Dutta and Joshi, 2011). A probabilistic search algorithm can solve this assignment problem in finite time.

4. SA-BASED APPROACH

SA originated in the annealing processes, found in the rmodynamics and metallurgy. SA algorithm has been used to approximate the solution of very large combinatorialo ptimization problems (Kirkpatrick, 1984) for example, NP-hard problems. It isbased upon the analogy between the annealing of solids and solving optimization problems.

SA exploits an analogy between the way in which a metal cools and freezes into a minimum energy crystalline structure (thean nealing process) and the search for a minimum in a more general system. SA'smajor advantage over other methods is its ability to avoid becoming trapped at local minima. SA algorithm was used for resource scheduling for many problems and produced good results (Ismail and Loh, 2009; Rakkiannan and Palanisamy, 2012). For certain problems, SA algorithm may be more effective than exhaustive enumeration-provided that the goal is merely to find an acceptably good solution in a fixed amount of time, rather than the best possible solution. Compared with the other optimization techniques such as GA, by and large, SA is convergence faster than GA runs (Ingber and Rosen, 1992; Wang and Zeng, 2010).

4.1. Initial Solution

Some experiments have been carried out with random initialsolutions. Computational time to converge towards a good solution is prohibitively long. Therefore, GA-Hint approach (Abdullah and Othman, 2012) which we call it here opposite () approach is used to calculate the initial solution of SA, which yields a good initialsolution. The best result is taken (the minimum cost) be for running the SA algorithm. Again, it is better to start with the best solution that has been heuristically built.

4.2. Move Sets in SA Algorithm

Move sets are the most important operators in the SA algorithm in this research. There are three types of move sets in this research which are Inversion, Translation and Switching. To explain the move sets, suppose that we have a sorted vector of integer numbers from 1 to 10as shown in **Fig. 1**.

In our research we have implemented only the Inversion move set. The Transition move set is time consuming while the Switching is simple (Ingber and Rosen, 1992). In the Inversion move set, we have generated two random points and then reversed the order. For example, if we generate the random points as 6 and 9, then we invert the numbers from 9 to 6 and store them in this particular order as shown in **Fig. 2**.

In the proposed approach, we sorted the machines in a vector. Then, for each iteration, before we schedule the job on the machine, we invert the vector to get a new solution. It means that, the machine taken in differently for each iteration of.

4.3. Application of SA to the Problem

The initial solution is calculated using opposite () Approach because it produced good solution previously in IGA approach (Abdullah and Othman, 2012). We start at the initial temperature T. We execute the SA using the Inversion move set technique discussed in Section.

Then, we calculate the new cost and assign to it a variable called new gain. If the new gain is less than or equal to the current gain then the solution is accepted. If the new gain is more than the current gain then we calculate exp (-E/T) and generate a random number. If the random number generated is less than $e^{(-E/T)}$ then the solution is accepted, otherwise the solution is rejected. If the solution is accepted then the new gain is assigned to the current gain and the new sorted vector is used for generating the next solution. If the solution is not accepted then the current gain does not change and the sorted vector will remain the same. I iterations are performed for each temperature value and after M iterations, the value of the temperature is changed to be T = T * x. The stopping condition T_s is chosen as 5 and is fixed. It is used to stop the algorithm when there is no change to the solution after T_s iterations. In this way the algorithm terminates either on reaching the final temperature T_{final} or after T_s iterations.

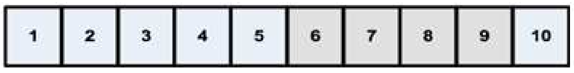

Fig. 1. Sorted vector

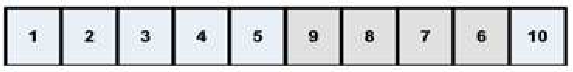

Fig. 2. After applying inversion move set

We repeat the process if T is greater than the final temperature and if t_{stop} is greater than 0 ($t_{stop} > 0$). Finally, the algorithm stops after having met any of the two conditions. At this point, the cost is returned, corresponding to the final solution obtained after applying the simulated annealing algorithm as shown below:

```
Calculate the cost using Oposite () initial solution;
Initial temperature T:
    While ($T>T_{final}$ and $T_{stop}>0$) do
   for I = 1 to I do
Inversion-move-set ()
        Calculate the new solution;
         If the new solution < = current solution then
the solution is accepted;
         Else
            R = Random Number ($0<R<1$); Y = exp(- E/T);
            If (R<Y) then accept the solution;
            Else reject the solution
            End if
            End if
         If we accept the solution then
            Current solution is set to the new solution;
         Else the old solution is kept;
        End if
        If we accept the new solution then $T_{stop} = T_s$;
      Else $T_{stop} = T_{stop}$-1;
      End if
      End for
Change the temperature by a factor x, T = T * x
      End while
```

5. RESULTS

To evaluate the performance of the algorithms, they have been simulated to find best schedule for different number of jobs and different number of machines. A number of jobs having different attribute are generated randomly and also a number of processing unit having random attribute are generated randomly. The simulation results proved that the SA approach will give good result with minimal execution time. Thus, we will compare the performance between Mapping, GA and SA approaches with these types of parameters. For instance, M different jobs having different characteristics are given generated randomly. Similarly N different process units attribute with random characteristics are generated.

To investigate the effectiveness of the proposed SA as compared to other based approaches, we direct schedule job i to process unitj, then we calculate the cost. We examined the overall performance of each algorithm by running them under 100 randomly generated cloud configurations. We set the GA related parameters ($r_x = r_{gx} = r_m = r_{gm} = 0.6$) with values that we found after some preliminary experiments. We tried different number of jobs and processing units: 10, 20, 40, 60 and 100.

6. DISCUSSION

For SA approach, After testing of many cases, the number of rounds was set at 100 (to calculate the average), with starting temperature at 1000, final temperature at 0.05, cooling rate at 0.9, the number of iterations per temperature at only 20 and T_{stop} rounds at20. If no further improvement is noticed, we then set the stop condition at only 20. The SA parameters values are also used in previous researches such as (Abdullah *et al*., 2006).

The most important advantage of SA-based over GA is in terms of the scheduler execution time. SA-based model outperforms the GA by a larger gap when running time is considered. The cost and the execution time of SA-based here prove the suitability of SA discussed in section 3. SA is often used when the search space is discrete for instance as is shown in **Fig. 3**.

When the opposite () model is used in SA based as heuristic technique, it speedup the algorithm. The integration between SA based algorithm and opposite () model successfully produces better results in a minimal time. It is also seen that SA based algorithm has better performance for any number of machines and jobs less than 100. When the number of machines and jobs is more than or equal 100, the GA is better than other algorithms. But, when we compare the execution time of both approaches, it is clear that SA approach is faster than GA approach as shown in **Fig. 4**.

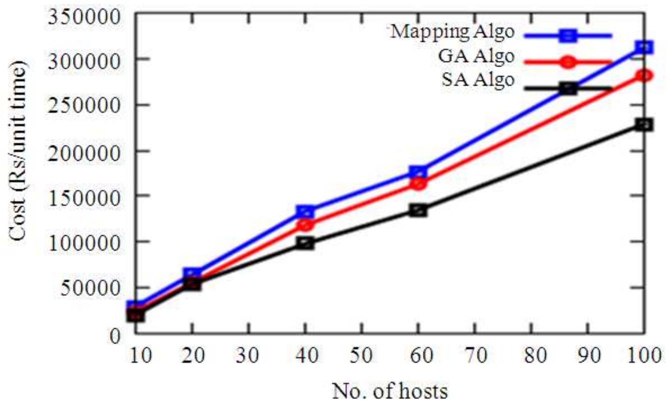

Fig. 3. COST Vs no. of hosts/jobs for all models

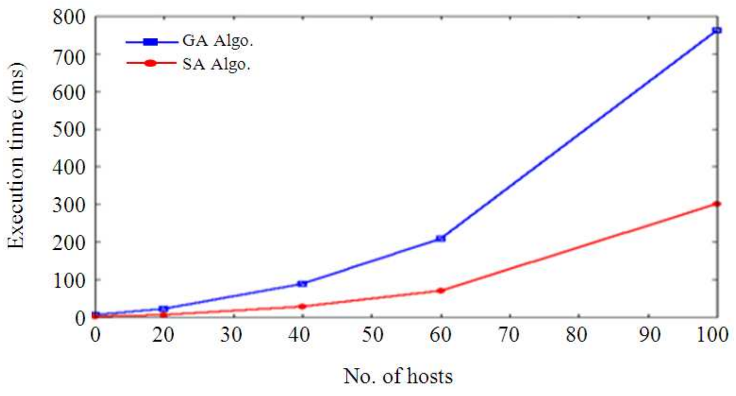

Fig. 4. Execution time of GA and SA models

7. CONCLUSION

In this study, SA algorithm has been developed. The proposed SA-based approach can be a competitive choice for scheduling in cloud environment. Analysis and a number of results show that this approach for job scheduling not only guarantees the QoS requirement of customer job but also ensure to make best profit of cloud providers. It has also concern about real execution time of jobs in different systems as well as deadline and penalty cost in the algorithm. For future works, we will try to evaluate the algorithms with some other QoS parameters and conduct more experiments in a more realistic environments.

8. ACKNOWLEDGEMENT

This study was partially supported by the Malaysian Ministry of Education, Fundamental Research Grant Scheme (FRGS Number: FRGS/02/01/12/1143/FR).

9. REFERENCES

Abdullah, M. and M. Othman, 2012. An improved genetic algorithm for job scheduling in cloud computing environment. Global J. Technol., 2: 291-296.

Abdullah, M. and M. Othman, 2013. Cost-based multi-QoS job scheduling using divisible load theory in cloud computing. Proc. Comput. Sci., 18: 928-935. DOI: 10.1016/j.procs.2013.05.258

Abdullah, M., M. Othman and R. Johari, 2006. An efficient approach for message routing in optical omega network. Int. J. Comput. Int. Manage., 14: 50-60.

Daoud, M.I. and N. Kharma, 2006. An efficient genetic algorithm for task scheduling in heterogeneous distributed computing systems. Proceedings of the IEEE Congress Evolutionary Computation, (CEC' 06), IEEE Xplore Press, Vancouver, BC, pp: 3258-3265. DOI: 10.1109/CEC.2006.1688723

Dutta, D. and R.C. Joshi, 2011. A genetic-algorithm approach to cost-based multi-QoS job scheduling in cloud computing environment. Proceedings of the International Conference Workshop Emerging Trends, Feb. 25-26, India, pp: 422-427. DOI: 10.1145/1980022.1980111

Ingber, L. and B. Rosen, 1992. Genetic algorithms and very fast simulated annealing: A comparison. Mathe. Comput. Model., 16: 87-100. DOI: 10.1016/0895-7177(92)90108-W

Ismail, Z. and S.L. Loh, 2009. Ant Colony Optimization for Solving Solid Waste Collection Scheduling Problems. J. Math. Stat., 5: 199-205. DOI: 10.3844/jmssp.2009.199.205

Javadi, B., J. Abawajyb and R. Buyya, 2012. Failure-aware resource provisioning for hybrid cloud infrastructure. J. Parallel Distribut. Comput., 72: 1318-1331. DOI: 10.1016/j.jpdc.2012.06.012

Kirkpatrick, S., 1984. Optimization by simulated annealing: Quantitative studies. J. Stat. Phy., 34: 975-986. DOI: 10.1007/BF01009452

Li, L., 2009. An optimistic differentiated service job scheduling system for cloud computing service users and provider. Proceedings of the 3rd International Conference Multimedia Ubiquitous Engineering, Jun. 4-6, IEEE Xplore Press, Qingdao, pp: 295-299. DOI: 10.1109/MUE.2009.58

Manikas, T.W. and J.T. Cain, 1996. Genetic algorithms Vs. Simulated annealing: A comparison of approaches for solving the circuit partitioning problem. Technical Report 96-101, Department of Electrical Engineering, University of Pittsburgh.

Mezmaza, M., N. Melabb, Y. Kessaci, Y.C. Lee and E.G. Talbi *et al*., 2011. A parallel bi-objective hybrid metaheuristic for energy-aware scheduling for cloud computing systems. J. Parallel Distribut. Comput., 71: 1497-1508. DOI: 10.1016/j.jpdc.2011.04.007

Mukherjee, K. and G. Sahoo, 2009. Mathematical model of cloud computing framework using fuzzy bee colony optimization technique. Proceedings of the 9th International Conference Advances Computing Control Telecommunication Technologies ACT, Dec. 28-29, IEEE Xplore Press, Trivandrum, Kerala, pp: 664-668. DOI: 10.1109/ACT.2009.168

Rakkiannan, T. and B. Palanisamy, 2012. Hybridization of genetic algorithm with parallel implementation of simulated annealing for job shop scheduling. Am. J. Applied Sci., 9: 1694-1705. DOI: 10.3844/ajassp.2012.1694.1705

Spinellis, D.D. and H.T. Papadopoulos. 2000. A simulated annealing approach for buffer allocation in reliable production lines. J. Annals Operat. Res., 93: 373-384. DOI: 10.1023/A:1018984125703

Sridhar, T., 2009. Cloud computing: A primer part 1: Models and technologies. Int. Protocol J., 12: 2-19.

Wang, M. and W. Zeng, 2010. A comparison of four popular heuristics for task scheduling problem in computational grid. Proceedings of the 6th International Conference Wireless Communications Networking Mobile Computing, Sep. 23-25, IEEE Xplore Press, Chengdu, pp: 1-4, DOI: 10.1109/WICOM.2010.5600872

Xu, M., L. Cui, H. Wang and Y. Bi, 2009. A Multiple QoS constrained scheduling strategy of multiple workflows for cloud computing. Proceedings of the IEEE International Symposium Parallel and Distributed Processing with Applications, Aug. 10-12, IEEE Xplore Press, Chengdu, pp: 629-634.DOI: 10.1109/ISPA.2009.95

Zhao, C., S. Zhang, Q. Liu, J. Xie and J. Hu, 2009. Independent tasks scheduling based on genetic algorithm in cloud computing. Proceedings of the 5th International Conference Wireless Communications Networking Mobile Computing, Sep. 24-26, IEEE Xplore Press, Beijing, pp: 1-4, DOI: 10.1109/WICOM.2009.5301850.