The 3rd International CUA Graduate Students Symposium

سومین سمپوزیوم بین‌المللی دانشجویان تحصیلات تکمیلی دانشگاه‌های عضو اتحادیهٔ قفقاز

University of Mohaghegh Ardabili

دانشگاه محقق اردبیلی

June 5-6, 2016

16–17 خردادماه

Task Scheduling in Cloud Computing Using Particle Swarm Optimization

Shahram Jamali
Fatemeh Alizadeh
Soheila Sadeqi

Department of Engineering, University of Mohaghegh Ardabili , Iran
Department of Engineering, University of Mohaghegh Ardabili , Iran
Department of Engineering, University of Mohaghegh Ardabili , Iran

Abstract

Cloud computing is the growth of distributed computing, parallel computing, utility computing and grid computing, or defined as the commercial implementation of these computer science theories. One of the fundamental issues in cloud environment is the task scheduling which plays the key role of efficiency of the whole cloud computing facilities. Scheduling maps the user's tasks to resources to be executed efficiently in order to benefit both the service providers and customers. Since the cloud task scheduling is an NP-hard optimization problem, many meta-heuristic algorithms have been proposed to solve it. In this paper a policy based on particle swarm optimization compared with genetic algorithm and FCFS, has been introduced. PSO is a population-based search algorithm based on the simulation of the social behaviour of birds within the flock. The main goal in this research is minimizing the makespan and waiting time of a given tasks set. Proposed policy and two other algorithms have been simulated using Cloudsim toolkit package. The results showed that PSO performed better than genetic and FCFS algorithms.

Keywords: Cloud computing, task scheduling, particle swarm optimization, makespan.

1.Introduction

Cloud computing is developed based on various recent progressions in virtualization, heterogeneous distributed computing, grid computing, web computing, utility computing and autonomic computing. Also by definition, a cloud computing environment has a large pool of easily usable and accessible

virtualized resources (such as hardware, development platforms and/or services). These resources can be dynamically re-configured to adjust to a variable load (scale), allowing also for an optimum resource utilization.
Millions of user send your request to cloud resource. Scheduling these jobs is a challenge to cloud system. Different scheduling method are discussed in [2], [3], [4], [5], [6], [7] and [8]. Although cloud computing systems these days provide a better way to perform the submitted jobs in terms of responsiveness, scalability, and flexibility, most job scheduling problems on cloud environment are still either NP-hard or NP-complete. The rule-based scheduling algorithm (e.g., exhaustive and deterministic scheduling algorithm) are simple and easy to implement, therefore they are widely used on nowadays cloud computing systems, but in large- scale scheduling problems the result of these scheduling method are often far from optimal and are not appropriate. We want to use a meta-heuristic algorithm for solving this problem, because the heuristic algorithm are working better in large-scale scheduling.

Some of these meta-heuristic algorithms are nature-inspired, e.g., Simulated Annealing (SA) [9], Genetic Algorithm (GA) [10], Ant Colony Optimization (ACO) [11], etc. There are also non-nature-inspired metaheuristics, such as Tabu Search [12] and Threshold Accepting (TA) [13]. One of the newest heuristic algorithms is PSO (particle swarm optimization). This algorithm that has developed by Kennedy and Eberhart [14] is one of the evolutionary algorithms which simulates social behaviour of flock of birds or groups of fishes toward their desired destination. In addition to benefits of heuristic algorithms, such as flexibility and acceptable calculations, PSO has consistent performance and easy implementation.

The rest of the paper is organized as follow: In section 2, we present our approach for scheduling problem. In section 3 experimental results are represented, and in section 4 conclusions are presented.

2.Method

To generalize the discussion, the assumption is that there is a set of cloud customer tasks and each task that is submitted by user is independent from other tasks. Each task is allowed to be processed on any given available resource. A task is processed on one resource at a time and given resource are available continuously. In this paper we suppose that number of tasks is more than number of resources. The cloud computing discussion in this study highlighted the fact that cloud task scheduling scenario is actual either NP-hard or NP-complete problems. The problem description can be presented as follows.

Inputs: Set of tasks is defined as T = $\{T_1, T_2, \dots, T_i, \dots, T_n\}$, where $i \in [1, n]$ and n is the number of independent tasks. Set of resource is defined as $M = \{M_1, M_2, \dots, M_j, \dots, M_m\}$, in which $j \in [1, m]$ and m is the number of available resource.

Output: An efficient Gantt chart of scheduling, including the assignment of tasks on available resource and makespan.

Constraints: The processing time of each task is resource-dependent. Each task must be completed without interruption once started. Resource cannot perform more than one task at a time. The tasks is executed on machines for sequential form. When task i is assigned to machine$j$, X_{ij} become 1 otherwise it become 0. In this part two fundamental conditions are considered:

$$\sum_j^m X_{ij} = 1 \; \forall \, i \in T \quad (1) \qquad\qquad X_{ij} \in \{0, 1\} \; \forall \, j \in M, i \in T \quad (2)$$

First limitation ensures that each task is assigned to only one processing resource.

Objectives: The aim is to mapping tasks to appropriate virtual machines in order to minimizing makespan and waiting time. The makespan is the completion time of tasks and waiting time is sum of wait time of all tasks in each machine for execution.

PSO is an algorithm proposed by Kennedy and Eberhart in 1995, which is a population-based search algorithm inspired by bird flocking and fish schooling. Social behaviour of these organisms motivated them to look into the effect of collaboration of species on to achieving their goals as a group. A large number of birds or fishes flock synchronously, change direction suddenly, and scatter and regroup together according to the particle and social experience. The discrete, binary version of the algorithm was presented also by Kennedy and Eberhart to operate on discrete binary variables [15]. The binary PSO was implemented by [16] to solve task scheduling problem and used by [17] and [18] to job scheduling in grid computing. This algorithm works by having a population (swarm) of candidate solutions (particles) that travels in the problem space searching for an optimum solution.

Each solution is evaluated by the fitness function to be optimized. Position of each particle represents a potential solution in the problem space. Each particle has D-dimensional velocities which are calculated as probabilities that change during the time particles move in space. At each iteration the velocity and position of each particle is stochastically updated by combining the particle's current solution, the particle's personal best solution or pbest, and the global best solution or gbest over all particles. In the following, the formulas that exists for updating position and velocity:

$$X_i^{k+1} = X_i^k + V_i^{k+1} \tag{3}$$

$$V_i^{k+1} = wV_i^k + c_1 r_1 (pbest_i^k - X_i^k) + c_2 r_2 (gbest - X_i^k) \tag{4}$$

At first we should have an accurate mapping between particles of PSO and problem solutions. If we suppose that there are n tasks and we aim to distribute them on m processing machines then particles should be defined in form of $m\times n$ matrixes. All elements of position matrix are 0 or 1 and in each Column there is only one 1 and other elements are zero.

Velocity matrix dimensions, Pbest and gbest are $m\times n$ matrixes with 0 and 1 elements exactly similar to position. Pbest matrix represents the best position of particle from beginning of the algorithm and gbest owns the best position between all particles. In each step of the algorithm particles are evaluated by fitness function of the algorithm and update pbest, gbest and position and velocity matrixes in terms of fitness value. Fitness function is defined based on our objective. We want to minimize makespan and waiting time. This process is repeated until stop criterion of algorithm is satisfied which in our algorithm it is equal to number of iterations. Flowchart of PSO algorithm is as Fig. 1. The objective function of our approach are formulated as follows:

$$\text{Minimize} \quad F(s) = \alpha\, C_{max}(s) + \beta\, W(s) \qquad (5)$$

Where $C_{max}(s)$ is the completion time of the last job (also called makespan); W(s) is sum of the waiting time of each task which waits for execute.

$$C_{max} = Max_{j\in M} C_j \quad , \quad C_j = \sum_{i\in T} p_{ij} \qquad (6)$$

Suppose that execution time of task i on computational node j is known, and it is equal to p_{ij}. The load of each machine is C_j.

$$W(s) = \sum_{i=1}^{m} \sum_{j\in T} w_{ij} \qquad (7)$$

Table 1

The parameter setup of VMs in c

Parameter	Value
CPU	2500, 2000, 1000, 500 MIPS
Ram	870, 1740, 1740, 613 MB
BW	100 M/S
The number of PEs	1, 1 ,1, 1

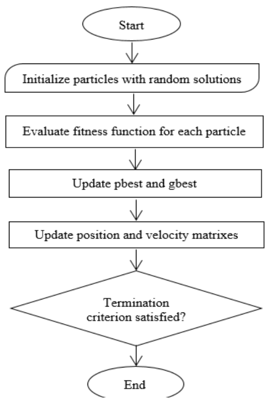
Start
Initialize particles with random solutions
Evaluate fitness function for each particle
Update pbest and gbest
Update position and velocity matrixes
Termination criterion satisfied?
End

Fig. 1 Flowchart of PSO algorithm

Table 2

The parameter setup of Tasks in datacentre

Parameter	Value

3.Results and Discussion

The proposed scheduling policy has been implemented in Cloudsim toolkit. Our experiments generated a data center with 50 host and 50 virtual machines. The parameter setup of hosts, VMs in the data center and tasks is shown in Table1 and Table2, respectively. In order to examine the performance of the proposed algorithm we have implemented Genetic algorithm and FCFS too and then compared the three mentioned policy in terms of Markesan and waiting time minimization.

Our experiments depicts that even if both Genetic algorithm and PSO algorithm show acceptable results, it can be said in large scale problem PSO algorithm shows better results than Genetic algorithm and FCFS policy. This algorithm can be used in cloud computing environment for efficient scheduling of tasks on existing resources, so that completion time of tasks and waiting time become minimized. Final results for three algorithms are shown in Fig. 2, Fig. 3 and Fig. 4.

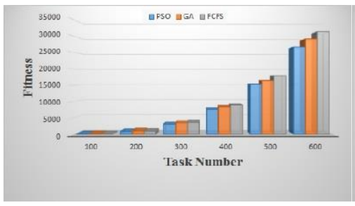

Fig 2. Comparing performance of different algorithms based on fitness

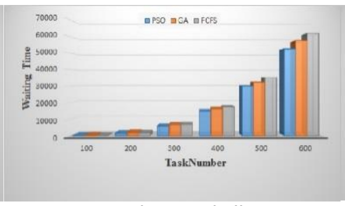

Fig. 3. Comparing performance of different algorithms based on Makespan

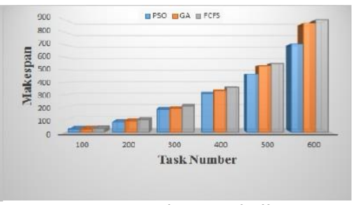

Fig 4. Comparing performance of different algorithms based on waiting time

References

1.Mell. P., & Grance. T. (2011). The NIST Definition of Cloud Computing-Recommendations of the National Institute of Standards and Technology. NIST. NIST Special Publication, 800-145, pp. 1–3.
2.Paul, M., & Sanyal, G. (2012). Survey and analysis of optimal scheduling strategies in cloud environment. IEEE.
3.Jeyarani, R., & Ram, R., & Versants, Nagaveni, N. (2010). Design and Implementation of an Efficient Two Level Scheduler for Cloud Computing Environment. IEEE.

4.Qi-yi, H., & Ting-lei, H. (2010). An optimistic job scheduling strategy based on QoS for Cloud Computing. IEEE.

5.Lee, K., & Fu, M., & Kuo, Y. (2011). A hierarchical scheduling strategy for the composition services architecture based on cloud computing. IEEE.

6.Leey, G., & Chunz, B., & Randy H. Heterogeneity-Aware Resource Allocation and Scheduling in the Cloud. University of California.

7.Wang, Sh., & Yan, K., & Wang, Sh. (2011). Ching-Wei Chen, "A Three-Phases Scheduling in a Hierarchical Cloud Computing Network. IEEE.

8.Peixoto, M.L.M., & Santana, M.J., & Estrella, J.C., & Tavares, T.C., & Kuehne, B.T., and Santana, R.H.C. (2010). A Metascheduler architecture to provide QoS on the cloud computing. IEEE.

9.Kirkpatrick, S., & Gelatt Jr, C.D., and Vecchi, M.P. (1983. Optimization by Simulated Annealing. Science, vol. 220, no. 4598, pp. 671-680.

10.Holland, J.H. (1975). Adaptation in Natural and Artificial Systems. Univ. of Michigan Press.

11.Bonabeau, E., & Dorigo, M., and Theraulaz, G. (2000). Inspiration for Optimization from Social Insect Behavior. Nature, vol. 406, pp. 39-42.

12.Glover, F., & Laguna, M. (1997). Tabu Search. Kluwer Academic Publishers.

13.Dueck, G., & Scheuer, T. (1990). Threshold Accepting: A General Purpose Optimization Algorithm Appearing Superior to Simulated Annealing. J. Computational Physics, vol. 90, pp. 161-175.

14.Kennedy, J., & Eberhart, R.C. (1995). Particle swarm optimization. Proc, IEEE Conf. Neural Netw., vol. IV, IEEE, Piscataway, NJ, pp. 1942-1948.

15.Kennedy, J., & Eberhart, R. Particle swarm optimization. Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, pp. 1942–1948.

16.Pierobom, J. L., & Delgado, M. R., and Kaestner C. A. (2011). Particle swarm optimization for task assignment problem. 10th Brazilian Congress on Computational Intelligence (CBIC2011), vol. 10, pp. 1–8.

17.Zhang, L., & Chen, Y., and Sun, R. (2008). A task scheduling algorithm based on PSO for grid computing. International Journal of Computational Intelligence Research. vol. 4, no. 1, pp. 37–43.

18. Izakian, H., & Ladani, B. T., & Zamanifar, K., and Abraham, A. (2009). A novel particle swarm optimization approach for grid job scheduling. Information Systems, Technology and Management - Communications in Computer and Information Science, vol. 31, pp. 100–109.