TELKOMNIKA, Vol.10, No.7, November 2012, pp. 1560~1566
e-ISSN: 2087-278X

Scheduling Workflow in Cloud Computing Based on Hybrid Particle Swarm Algorithm

Sheng-Jun Xue, Wu Wu*
Nanjing University of Information Science & Technology
School of Computer and Software, Nanjing, China
*corresponding author, e-mail: wuwuyunhu@126.com

Abstract

The cost minimization with due dates in cloud computing workflow is an intractable problem. Taking the characteristics in cloud computing of pay-per-use and resource virtualization into account, in this paper, we present a QoS-based hybrid particle swarm optimization (GHPSO) to schedule applications to cloud resources. In GHPSO, crossover and mutation of genetic algorithm is embedded into the particle swarm optimization algorithm (PSO), so that it can play a role in the discrete problem, in addition, variability index, changing with the number of iterations, is proposed to ensure that population can have higher global search ability during the early stage of evolution, without the premature phenomenon. A hill climbing algorithm is also introduced into the PSO in order to improve the local search ability and to maintain the diversity of the population. The simulation results show that the GHPSO achieves better performance than standard particle swarm algorithm used in minimize costs within a given execution time.

Keywords*: cloud computing, PSO, workflow, hill climbing algorithm*

1. Introduction

Cloud computing does not have a common definition, Vaquero et al. [1] have compared more than 20 different concepts of cloud computing in order to give a definition. Cloud computing is the further development of distributed computing, parallel processing and grid computing, and is internet-based computing [2]. Cloud computing is a new type of shared infrastructure which can connect huge pools of systems, provides users with a variety of storage and computing resources via the Internet [3]. The business features of cloud computing require it to meet users' application needs.These applications can usually be broken down into a number of tasks associated with each other.The complex constraints between tasks can be described by directed acyclic graph (DAG) [4]. Therefore an effective scheduling optimization is needed in order to achieve the goal that workflow tasks can complete an entire application.

In general, execution time and computation cost are two typical restrictions in the model of "pay-per-use". In the reality environment, users are charged while using resources, so workflow scheduling under cloud computing not only aims to minimize execution time, but also considers the cost, it means workflow scheduling under cloud computing is a time-cost optimization problem, i.e. minimize computation cost with the deadline constraints.

Task scheduling in cloud computing is an NP-hard problem, PSO as one of the heuristic algorithms has been applied in solving scheduling problem and other NP-hard problems [5-6], it is relatively easy to implement compared with the ant colony optimization algorithm and genetic algorithm [7]. PSO has been used in the workflow scheduling problem under the cloud computing environment [8]. However, PSO has some disadvantages, such as poor local search ability and not suitable for problems in discrete areas. This article proposes a workflow scheduling strategy in cloud computing based on hybrid particle swarm algorithm (GHPSO) in order to solve these shortages.

2. Time-cost Problem Formulation

2.1. Mathematical model

Constraints between workflow tasks can be modeled as a directed acyclic graph. Next, denote a workflow application as a DAG represented by $G = \{T, E, W, C\}$, where

Received July 15, 2012; Revised September 10, 2012; Accepted September 25, 2012

T={1,2,…,n}(|T|=n)is the set of tasks, and E represents the data dependencies between these tasks, (t_i, t_j) means that t_j can be implemented only by computed entire task t_i ; P={p_1,p_2,…,p_m}is a set of compute sites, w_{np} indicates that the execution time of task n on compute site p; $\mathrm{pre}(t_i)$ is the direct precursor node of task t_i; $\mathrm{suc}(t_i)$ is the direct subsequent node of task t_i ; $pre^*(t_i)$ represents all precursor nodes of task t_i ; $suc^*(t_i)$ represents all subsequent nodes of task t_i; t_i^E stands for the earliest execution time of taskt_i; t_i^Sstands for the real execution time of task t_i.

$$t_j^E = \begin{Bmatrix} 0, \ \mathrm{pre}(t_i) = \emptyset \\ \max_{t \in pre^*(t_i)} \{t_i^E + \sum_{p=1}^{m} w_{ip} \times x_{ip}\}, pre(t_i)! = \emptyset \end{Bmatrix} \quad (1)$$

Eq.(1) stands for the earliest execution time of taskt_j, where$x_{ip} \in \{0,1\}$, it means the value equals 1 when task $t_i$chooses compute site p from its resource pool, otherwise 0. Figure 1.is a simple instance of workflow.

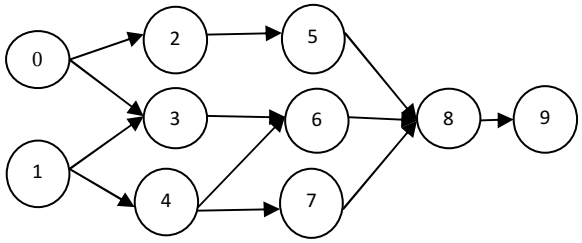

Figure 1. A simple instance of workflow

2.2. Fitness function

In order to facilitate the simulation study, this paper is to make the following assumptions:

(1) the performance of resources in cloud computing pools(compute sites) is different;
(2) tasks have no-interrupted priority, the same task cannot run on two different compute sites;
(3) each compute site can only handle one task;
(4) ignore the memory, I/O, communication cost and other resource requirements.

The fitness function of cloud computing workflow tasks scheduling problem is defined as following:

Workflow deadline σ_n is usually given in advance (expressed in constant), it represents the final completion time of all tasks the workflow requires. In this article, we propose a scheduling strategy based on QoS restricts, each task in workflow is assigned to a particular compute site in order to minimize cost within the given deadline. Its concrete formal description is given in Eq. (2) and Eq. (3).

$$C_{\mathrm{total}} = min \sum_{i \in T} \sum_{1 \le p \le m} x_{ip} \, c_{ip} \quad (2)$$

$$\begin{cases} t_i^E \le t_j^E + \sum_{p=1}^{M} w_{jp} \times x_{pj}, t_i \in pre(t_j), \forall (i,j) \in E \\ \sum_{j=1}^{M} x_{ij} = 1, \forall i \in T \\ f_n \le \sigma_n \\ x_{ij} \in \{0,1\} \end{cases} \quad (3)$$

where f_n is the workflow completion time, $f_n \le \sigma_n$ says to meet the deadline constraints, $\sum_{j=1}^{M} x_{ij} = 1$ means that the task can only choose one compute site to perform, $x_{ij} \in \{0,1\}$ is a Boolean variable, as mentioned above, x equals 1 when task $t_i$chooses compute site j from its resource pool, otherwise 0. $C_{total}$is the objective function, that minimizes workflow cost.

3. Hybrid particle swarm algorithm

Particle Swarm Optimization is a new intelligent global optimization algorithm, it was proposed in 1995 by Eberhart and Dr. Kennedy, Dr. [9-10]. Particles update velocity and position in each iteration according to the following formula:

$$v_{id}^{k+1} = w \times v_{id}^{k} + c_1 r_1 \left(p_{id} - x_{id}^{k}\right) + c_2 r_2 (p_{gd} - x_{id}^{k}) \quad (4)$$

$$x_{id}^{k+1} = x_{id+}^{k} v_{id}^{k+1} \quad (5)$$

where k represents the k-generation algebra evolution, in Eq.(4) w is the inertia weight, c_1 and c_2 is the learning factor, also known as the acceleration coefficient, r_1 and r_2 is the random number between 0 and 1, these two parameters is used to maintain the diversity of the population. v_{ij} is a linear combination of three parts.

3.1. Particle encoding method

For the generated DAG topological Sequence, each node corresponds to a task; each task corresponds to a dimension of particle. The dimension of particle encoding d=n means that the dimension of the particles solution space corresponding to the number of workflow tasks. Solution of the particle in each dimension represents a service corresponding to the compute site number. In search process, the values of each dimension of particles in the resource pool keep changing, and re-construct a new particle after evolution according to the new location of each dimension. In order to ensure the solution space can be fully searched, the quantity of the particle follows the principle that it is no less than twice of the number of tasks.

For the instance in Figure 1. , Table 1 constructs one of the particles, this particle has 10 dimensions, representing 10 task options, and the value of the particle in each dimension represents the number of compute site which the task assigns to.

Table 1. Task-compute site allocation

task	1	2	3	4	5	6	7	8	9	10
Compute site	2	0	3	1	3	2	5	4	6	5

Therefore a particle is coded as shown in Table 2:

Table 2.Particle encoding

particle	2	0	3	1	3	2	5	4	6	5

3.2. Improvement strategies

The nature of particle swarm optimization algorithm is to use the individual extremum information and global extremum to update the velocity and position of the particles. For task scheduling problem, the speed in particle swarm algorithm is hard to express, so crossover operator in genetic algorithm can be used, the resulting solution is a new location.

As we all known, PSO is suitable for continuous areas, now, it can be suitable for discrete problem by introducing the crossover strategy and mutation strategy of the genetic algorithm [11] into it. $c_1 r_1 \left(p_{id} - x_{id}^{k}\right) + c_2 r_2 (p_{gd} - x_{id}^{k})$ is viewed as crossover operator, make the current solution do crossover operation on individual extremum, and make the result do crossover operation on global extremum, $w \times v_{id}^{k}$ can be viewed as mutation operator of genetic algorithm, the variation of the above results is a new location. Meanwhile, in order to improve local search ability of particle swarm optimization algorithm and reduce the particle swarm algorithm premature convergence, hill climbing algorithm is introduced into the particle swarm algorithm, in GHPSO, do climbing operation on global extremum in each generation.

(1) Crossover strategy

Select the location to be exchanged in accordance with a given crossover probability in the second sequence, the value of the selected position in the second sequence will exchange with the value of corresponding position in the second sequence, and randomly generate to be exchanged in accordance with the crossover probability location.

(2) Mutation strategy

In order to guarantee the diversity of the population, we introduce the mutation strategy in genetic algorithm into PSO and give a variability index. Since the PSO search process is a complex nonlinear process, we need to maintain population diversity at the early stage of

evolution and have a high global search capability, while at the later stage of evolution we need to focus on improving the local search ability to have a high search precision, coefficient of variation should therefore be to maintain a larger value in the long evolution of the early and later stage of evolution a long time to maintain a smaller value. The coefficient of variation given as follows:

$$\lambda = \begin{cases} -a \times k^2 + {}^1\!/_2 & 0 < k < 0.5K \\ a(k-K)^2 & 0.5K < k < K \end{cases} \quad (6)$$

where K is the given number of iterations, a describes the degree of curve bending, take $a=1/K^2$ here, take an integer value of k in practice. Function image is shown as Figure 2:

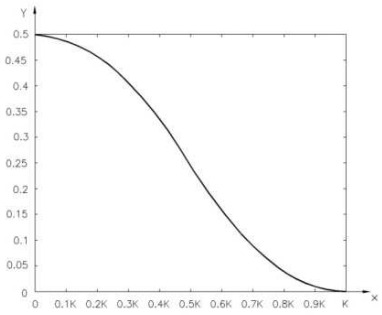

Figure 2. Function image

The mutation strategy is as following:

- Randomly select λ*K compute sites from the sequence, and then generate the same number of random numbers ranging in the interval (0, m], replace the selected compute site.
- The detailed operating methods of crossover and mutation are as following:
- In each iteration, make the distribution sequence of particle i which is represented by$C_0(i)$ do crossover operation on gbest, and the result is $C_1^1(i)$; make $C_1^1(i)$do crossover operation with pbest, and the result is $C_1^2(i)$, and then do mutation operation on $C_1^2(i)$, the result is $C_1(\mathrm{i})$, calculate the fitness value according to the current location.

(3) Hill climbing

Hill-climbing algorithm is an iterative improvement algorithm and a partial selection of the best methods; it is a heuristic method which uses feedback information to help generate solutions of the decision-making.

The main idea of hill-climbing algorithm is to start from the current node, and compare with the surrounding neighbors. If the current node is the largest, then return to the current node as the maximum (i.e. the peaks of the highest point); otherwise, highest neighbor replace the current node, in order to achieve the goal of climbing to the peaks of the height, keep doing this loop until achieve the goal.

Do hill-climbing operation when all things have done in each iteration, randomly select two dimensions from global extreme position vector in each generation, and exchange their values. Compare the fitness value of the particle position after exchanged with the fitness value of current particle individual position, and update the individual vector pbest of particles.

3.3. Steps of GHPSO

The steps of GHPSO is mainly divided into three parts: first, initialize the particles, second, use the crossover and mutation operation to improve PSO, so that it can be applied to this discrete problem, third, make the climbing operation after the end of each iteration to improve the local search ability. The specific steps are shown in Figure 3.

Step 1 : Initialize the particles; make each particle fit the restrictions in Eq. (3), and set the maximum number of iterations.

Step 2 : For each particle, based on crossover strategy and mutation strategy to update the fitness value of particle position and the new location.

Step 3 : Set the deadline, go to Step 2 when the execution time exceeds the deadline, or else turn to Step 4.
Step 4 : Randomly select two dimensions from global extreme position vector in each generation, and exchange their values.
Step 5 : Compare the fitness value of the particle position after exchanged with the fitness value of current particle individual position, and update the individual vector pbest of particles.
Step 6: Compare the fitness value of the particle individual position with the fitness value of global extremum, choose a minimum value of the individual to the current global extremumgbest.
Step 7 : Do a judgment that if the climbing times is meet, if it is, forward to step 8, otherwise to step 4.
Step 8 : Do a judgment that if the maximum number of iterations is meet, if it is, forward to step 9, otherwise to step 2.
Step 9 : Output the scheduling scheme and its fitness value.

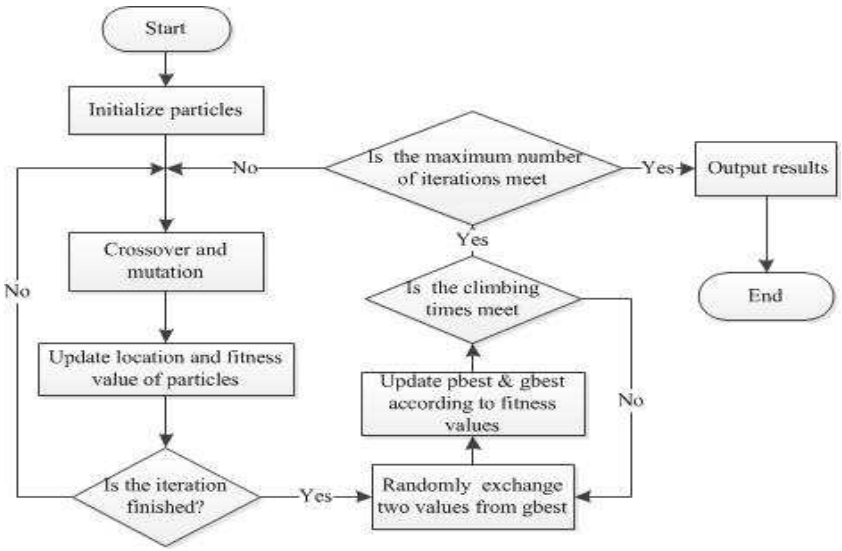

Figure 3. The flow chart of GHPSO

4. Experiments and results

In order to evaluate the performance of GHPSO, we test GHPSO in a simulated environment for instance using Java programming language. For standard PSO, c1=c2=2, w=0.8. In GHPSO, some strategy are used, such as randomized, multi-point crossover and uneven variation, where the crossover probability is set to 0.3, mutation probability is determined by the variability index. Computing power of compute sites is randomly generated, ranging in the interval [5, 9], the prices of the compute sites are randomly generated, ranging in the interval [50, 99], which is aim to simulate the performance of different compute sites. Mission length is randomly generated; ranging in the interval [100,199], the task processing time is represented by the quotient of task length and computing power. The number of iterations I = 100. Because compute sites performance and task length is generated randomly, repeat 100 times and take the average result in each case in order to maintain the reliability of the results.

4.1. Performance comparison

Simulation results of the workflow tasks in Figure 1 are shown in Table 3, as the number of iterations increasing, according to the PSO, compare the particle swarm optimized by crossover operation with GHPSO in this dissertation. The number of iterations k = 100, and take average value by running 100 times, population size N = 30. Can be evidently seen from the table that GHPSO's performance of global optimization has been greatly improved, the minimum value is significantly superior to the standard particle swarm algorithm and the simple

cross-Particle swarm optimization (CPSO), iteration termination conditions are that when 80% of the individual optimal value in the particle swarm is the same as the global optimum, and we think that the algorithm is nearly optimal solution, the algorithm can stop iteration. It can be seen from Table 3 that nearly 20 percent of the iterative process does not need to run to the last step and can reach the optimal solution while the number of iterations is set to 100 in hybrid particle swarm algorithm. Compared to average number of iterations which is 100 in the standard particle swarm, GHPSO significantly has faster convergence ability, and significantly enhances the search efficiency.

Table 3.Results of Figure1.

algorithm	Maximum	Minimum	average value	Variance	Times
PSO	19105	18345	18641.09	25595.48	100
CPSO	18988	18402	18640.08	19138.17	94.56
GHPSO	18402	18345	18345.57	32.16	80.05

4.2. The influence of the task magnitude on algorithm performance

DAG that represents the workflow tasks randomly generated by DAG generator written in accordance with the different scale in the interval [20,100], in increments of 10 and rounded, the population size N = 200. It can be seen from Fig. 3 that the quality of the optimal solution for GHPSO is far better than that of the standard particle swarm algorithm and particle swarm optimization with crossover factor under different scale of the problem. During the experiment, GHPSO does not spend more time than the other two algorithms, and can find a higher quality solution, with the growth of the scale of the problem, GHPSO in some extent is also better than standard PSO; GHPSO is suitable for solving large-scale problems. From the experiment it is easy to see that GHPSO has a significant effect in cost optimization, GHPSO can be used to save cost for large-scale workflow.

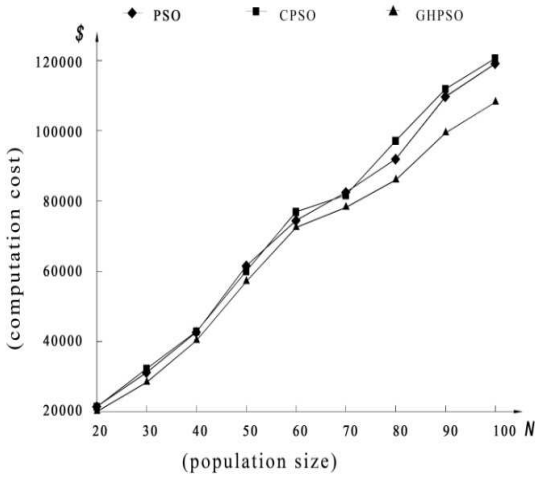

Figure 4. Optimal solution of algorithms in the different problem scale

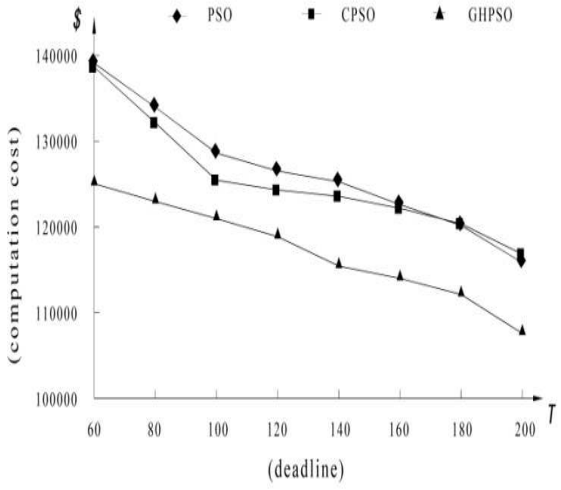

Figure 5.The cost in different deadlines

4.3. The influence of the task magnitude on algorithm performance

The deadline σ_n is a given constant; it represents the latest finish time that users expect. Different deadlines will produce different effects on performance of the algorithm, in general, that is, the smaller the deadline, the greater the cost. Fig. 4 describes the cost floating with the deadline according to the workflow example in the scale of 100 tasks, where deadlines range in the interval [60,200], in increments of 20.

It can be seen from Figure5; solution quality of GHPSO is getting higher and higher with the deadlines increasing, and it trends to global optimal solution. This indicates that GHPSO has better practicability in the cost minimization with due dates in cloud computing workflow.

5. Conclusion

Workflow scheduling based on QoS constraints in cloud computing is an intractable problem. This paper presents GHPSO to achieve the scheduling goals, thanks to PSO is suitable for continuous areas, the crossover strategy and mutation strategy of the genetic algorithm is embedded into PSO, so that it can play a role in the discrete problem, and introduces hill climbing algorithm into the particle swarm algorithm in order to improve the local search ability. After compared with the standard particle swarm algorithm on a large number of examples, GHPSO in this paper greatly improves the solution quality, so it can be used as an effective way to solve the cost minimization problem with due dates in cloud computing.

In addition, during the actual work flow scheduling, the problem is not simply to calculate the minimum cost within a constrained time period, nor to ask for minimum executing time under the condition of constrained cost, it is a multi-target problem. So multi-target workflow scheduling in cloud computing is the study goal in the next stage.

References

[1] Vaquero L, Rodero- Merino L, Caceres J, Lindner M. A break in the clouds: towards a cloud dentition. *ACM SIG-COMM computer communications review*.2009.

[2] Armbrust, M., Fox, A., et al.Above the Clouds: A Berkeley View of Cloud Computing. *Technical Report* No. UCB/EECS-2009-28, University of California at Berkley, USA .2009; February 10.

[3] M.A. Vouk. Cloud computing - issues, research and implementations.*Journal of Computing and Information Technology.*2008; 16 (4):235–246.

[4] Yu, J., Buyya, R.A taxonomy of scientific workflow systems for Grid computing.*SIGMOD Record, Special Section on Scientific Workflows*.2005; 34(3):44-49.

[5] B. Yu, X. Yuan, and J. Wang. Short-term hydro-thermal scheduling using particle swarm optimization method. *Energy Conversion and Management*.2007; 48(7):1902–1908.

[6] Veeramachaneni, K., Osadciw, L.A.*Optimal scheduling in sensor networks using swarm intelligence*. In: Proceedings of 38th Annual Conference on Information Systems and Sciences, 2004:17-19.

[7]Sen S, Roy P, Chakrabarti A, Sengupta S. Generator Contribution Based Congestion Management Using Multiobjective Genetic Algorithm. *TELKOMNIKA Indonesian Journal of Electrical Engineering*.2011; 9(1): 1-8.

[8] Pandey, S.; Linlin Wu; Guru, S.M.; Buyya, R. *Advanced Information Networking and Applications* (AINA), 2010 24th IEEE International Conference on Digital Object Identifier: 10.1109/AINA. 2010; 400-407.

[9] Eberhart, R. C., and Kennedy, J. *A new optimizer using particle swarm theory*. Proc. Sixth Intl. Symp.on Micro Machine and Human Science (Nagoya, Japan), IEEE Service Center, Piscataway, NJ. 1995; 39-43.

[10] Kennedy J, Eberhart RC.*Particle swarm optimization*.Proceedings of the IEEE international conference on neural networks. IEEE Press, Piscataway.1995;1942–1948.

[11]Bhaskar MM, Benerji M, Sydulu M. A Hybrid Genetic Algorithm Approach for Optimal Power Flow.*TELKOMNIKA Indonesian Journal of Electrical Engineering*. 2011; 9(2): 211-216.