INTRODUCTION

Cloud computing as a service currently very popular, has gained concerns of many areas. Some business, having high demand of currency, request that the system should finish the tasks submitted by users as soon as possible. In order that the system can handle service requests rapidly, it becomes the significance and difficulty in cloud computing study that how to schedule tasks efficiently and reasonably.

In the cloud computing environment, tasks scheduling problem is a Non-deterministic Polynomial[1]. Genetic algorithms[2], Particle Swarm Optimization[3], Ant Colony Optimization[4] have been used to solve the task scheduling problems in cloud computing. Hua Xiayu[5] put forward a method solving the resource scheduling problem based on Ant Colony Optimization. The Ant Colony Optimization has better optimization capability, but this kind of algorithm’s program is complex, initial pheromone is scarce, and the convergence velocity is slow[6].

Based on the study of Ant Colony Optimization and Particle Swarm Optimization, this paper proposes a resource scheduling method combining Particle Swarm Optimization and Ant Colony Optimization. The PSO is abbreviation for Particle Swarm Optimization and ACO is abbreviation for Ant Colony Optimization. The algorithm absorbs the PSO’s fast initial convergence velocity, and can easily program, and its performance is better than Genetic algorithm, but in late stage, it is lack of the local search capability, and the convergence velocity is slow.

ABSTRACT: Cloud computing system needs to process a large amount of data and has arduous tasks. An important question in cloud computing research is how to scheduling the resource rationally in this vast system. This paper presents an algorithm of resource scheduling combining Particle Swarm Optimization and Ant Colony Optimization. Firstly, get a good initial solution quickly through PSO algorithm, then ACO use the initial solution to determine the generation of initial pheromone distribution which is used to calculate the optimal solution through iteration. By experiment, the scheduling algorithm which combines PSO and ACO is better than both of the algorithms based on PSO or ACO.
computing task scheduling is to minimum the total task completion time by allocating these various subtasks into resources reasonably.

2.1 The representation of feasible solutions

2.1.1 Represented by n-dimensional Vector
All subtasks set as: \( T = \{ t_1, t_2, \ldots, t_n \} \).
All virtual resource nodes set as: \( V = \{ v_1, v_2, \ldots, v_m \} \).
\( t_i \) represents the i-th subtask\((i=1, 2, \ldots, n)\), \( v_j \) represents the j-th virtual resource node\((j=1, 2, \ldots, m)\).

An n-dimensional vector \( X \) represents the allocation relationship between the tasks and virtual resources nodes.
\( X = \{ x_1, x_2, x_3, \ldots, x_i, \ldots, x_n \} \).
If assigned the i-th task to the j-th virtual resource node, the value of \( x_i \) is j. Obviously \( 1 \leq x_i \leq m \).

2.1.2 Represented by an n×m matrix
An n×m matrix can also represent the allocation relationship between the tasks and virtual resources nodes, which named Distribution Relationship Matrix in this paper.
\[
Y = \begin{pmatrix}
y_{11} & y_{12} & \cdots & y_{1m} \\
y_{21} & y_{22} & \cdots & y_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
y_{n1} & y_{n2} & \cdots & y_{nm}
\end{pmatrix}
\] (1)

\[
y_{ij} = \begin{cases} 
1 & \text{if } t_i \text{ are allocated to } v_j \\
0 & \text{other conditions}
\end{cases}
\] (2)

Because one task can only be assigned to one virtual node, so the matrix \( Y \) comply with restriction of Equation. (3).
\[
\sum_{j=1}^{m} y_{ij} = 1
\] (3)

Both vector \( X \) and matrix \( Y \) may represent a feasible solution. \( X \) and \( Y \) are just different forms, and they can be transformed into each other. The vector \( X \) could be transformed into \( Y \) by the following method.

Given vector \( X = \{ x_1, x_2, x_3, \ldots, x_i, \ldots, x_n \}, i \in \{1, 2, \ldots, n\}, j \in \{1, 2, \ldots, m\}, 1 \leq x_i \leq m \).
Every values of elements of matrix \( Y \) based on the values of vector \( X \).
\[
Y = \begin{pmatrix}
y_{11} & y_{12} & \cdots & y_{1m} \\
y_{21} & y_{22} & \cdots & y_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
y_{n1} & y_{n2} & \cdots & y_{nm}
\end{pmatrix}
\] (4)

\[
y_{ij} = \begin{cases} 
1 & \text{if } x_i = j \text{ in vector } X \\
0 & \text{other conditions}
\end{cases}
\] (5)

Apparently, the restriction of Equation. (6) is fulfilled.
\[
\sum_{j=1}^{m} y_{ij} = 1
\] (6)

2.2 The definition of evaluation function
The i-th task’s length is the instructions number of \( t_i \). The j-th virtual node’s speed is how many instructions the virtual resource node \( v_j \) can run in unit time. The time length that node \( v_j \) runs task \( t_i \) is \( s_{ij} \), and apparently \( s_{ij} \) is equal to length if \( t_i \) divided by speed of \( v_j \). The node \( v_j \) may run more than one task, the total time length that \( v_j \) run all the tasks on itself is \( s_j \). \( S_{\text{max}}=\max\{s_j\}, j \in \{1, 2, \ldots, m\} \). So the feasible solution’s evaluation function is \( S_{\text{max}} \). The goal of resource scheduling method is to calculate the solution \( X \) so \( X \)’s evaluation function \( S_{\text{max}} \)’s value is minimum.

3 RESOURCE SCHEDULING METHOD

PSO and ACO algorithm both have shortcomings. PSO has fast initial convergence velocity, but in late stage, it is lack of the local search capability, and the convergence velocity is slow. The ACO’s initial pheromone is scarce, and the convergence velocity is slow, but it has better optimization capability. This paper proposes a cloud computing resource scheduling method based on PSO and ACO algorithm. Firstly, use PSO’s fast convergence to get initial solution. According to the initial solution generate ACO’s initial pheromone distribution. At last, use ACO to get the optimal solution of cloud computing resource scheduling.

3.1 Algorithmic process
The flowchart of cloud computing resource scheduling method based on PSO and ACO shown in Figure. 1.

3.2 Resource scheduling algorithm process

3.2.1 Generate particles randomly
A feasible solution \( X \) is ann-dimensional vector. \( X \) is one particle’s position, so the current position of k-th particle is expressed as: \( X^k = \{ x_{1k}, x_{2k}, x_{3k}, \ldots, x_{nk} \} \). Similarly, the best position particle k has ever visited is called Best Particle Position and is expressed as: \( X_{k \text{pbest}} = \{ x_{1k}, x_{2k}, x_{3k}, \ldots, x_{nk} \} \), and the Best Group Position is expressed as: \( X_{g \text{best}} = \{ x_{1g}, x_{2g}, x_{3g}, \ldots, x_{ng} \} \). \( 1 \leq x_{ik}, x_{jk} \leq l, i \in \{1, 2, \ldots, n\} \).

Generate some initial feasible solutions. Every feasible solution is a particle. With these initial solutions, good solution will be calculated quickly through PSO.

3.2.2 Movement of particles’ positions
This step is changing the initial feasible solutions. Each iteration in PSO, the movement of a particle’s solutions is determined by 3 parts. First part is the current state of this particle; the second part is self-cognition - the movement of a particle’s...
position is influenced by the Best Particle Position of this particle; and the third part is group-cognition - the movement of a particle’s position is influenced by the Best Group Position[10]. Feasible solutions are changed by using genetic cross in genetic algorithm. In the d-th iteration, cross the particle k’s solution \(X_d^k\) with its Best Particle Solution \(X^k_{gbest}\), then cross the \(X_d^k\) with the Best Group Solution \(X^g_{gbest}\). This particle’s new feasible solution is generated. Calculate all particles’ evaluation function value \(S_{max}\), and then update \(X^k_{gbest}\) and \(X^g_{gbest}\) according to \(S_{max}\). That’s all operations in once iteration.

3.2.3 Produce a good initial solution
Repeat step (b) until the number of iterations is reached. The \(X^k_{gbest}\) is a good initial solution.

\[
p_{ij}^k(t) = \begin{cases} 
\frac{[info_{ij}(t)]^\alpha [rand_{ij}(t)]^\beta}{\sum_{y_{ij} \in avoid-k} [info_{ij}(t)]^\alpha [rand_{ij}(t)]^\beta} & y_{ij} \notin avoid - k \\
0 & \text{other conditions}
\end{cases}
\]  
\[
\text{rand}_{ij}(t) = \frac{1}{s_{ij}^k(t)}
\]

where \(info_{ij}(t)\) represents how much pheromone the node \(y_{ij}\) remains at time t. \(\text{rand}_{ij}(t)\) is inspiring message. avoid-k is “unreachable list” of ant k. When ant k selects node \(y_{ij}\), all the nodes of i-th row in matrix \(Y\) - \(\{y_{ij}\}_{j=1}^m\) are added to avoid-k to make restriction of Equation. (3) holds. \(\alpha\) and \(\beta\) are constants, reflecting the importance of pheromone and inspiration information.

Ant selects a node \(y_{ij}\), that means to allocate \(t_i\) into \(v_j\).

3.2.7 Spreading pheromone
Ant moves to a node \(y_{ij}\) and spreads pheromone into this node, while the pheromone of all nodes are evaporating.

Pheromone of all nodes are updated by the Equation. (9) and Equation. (10).

\[
\text{info}_{ij}(t + 1) = (1 - \rho)\text{info}_{ij}(t) + \sum_{k=1}^h \text{data}_{ij}^k(t)
\]

\[
\text{data}_{ij}^k(t) = \begin{cases} 
\frac{q}{s_{ij}^k(t)} \text{antchoose}_{ij} & \text{antchoose}_{ij} \\
0 & \text{other conditions}
\end{cases}
\]

where \(\rho\in[0,1)\) is the pheromone evaporation coefficient, so \((1-\rho)\) is pheromone residue coefficient. \(\text{data}_{ij}^k(t)\) in the Equation. (9) is the amount of pheromone ant k spreads on the node \(y_{ij}\) when it chooses node \(y_{ij}\). \(Q\) is a constant, \(h\) is the number of ants. \(s_{ij}\) is the time length that node \(v_j\) runs task \(t_i\), which is discussed in detail in Section 2.2.

When ants have selected all the nodes and each ant has generated a feasible solution, make a global pheromone update in all nodes of \(Y\).

\[
\text{info}_{ij}(t + n) = (1 - \rho)\text{info}_{ij}(t) + \sum_{k=1}^h \text{data}_{ij}^k(t)
\]

\[
\text{data}_{ij}^k(t) = \begin{cases} 
\frac{q}{s_{ij}^k(t)} \text{antchoose}_{ij} & \text{antchoose}_{ij} \\
0 & \text{other conditions}
\end{cases}
\]

3.2.8 Generate optimal solution
Repeats steps (f)(g), optimal solution is generated when iteration stops. The optimal solution is a group of nodes - \(\{y_{1m_1}, y_{2m_2}, \ldots, y_{jm_j}, y_{nm_n}\}\) from \(\{y_{ij}\}_{n\times m}\) where \(m_j\in\{1, 2, \ldots, m\}\). Optimal solution’s evaluation function value \(S_{max}\) is minimum.

3.2.4 Using ACO calculating better solution
Referring Distribution Relationship Matrix \(Y\), define \(\{y_{ij}\}_{n\times m}\) is a node-set, which consists of an undirected complete graph \(G(V,E)\).

3.2.5 Pheromone initialization
After the step (c), The \(X^k_{gbest}\) is best solution at present. Transform the \(X^k_{gbest}\) into a Resource Allocation Matrix \(Y_{ori}\) as the initial solution of ACO. According to the initial solution \(Y_{ori}\), set the pheromone of nodes of which values increase a certain number.

3.2.6 Place ants randomly, ants choose path
Place ants randomly. It means that generate some initial feasible solutions, just like step (a), every feasible solution is an ant.

At time t, ant k’s probability of selecting \(y_{ij}\) as next node is calculated by Equation. (7) and Equation. (8).
4 EXPERIMENTAL ANALYSIS

Use Matlab to do comparative tests under the same condition. The number of tasks will be set from 20 to 100, virtual computing resource nodes will be set 8.

Take ACO, PSO algorithm and fusion algorithm in same tests. In this section, the fusion algorithm is called ACO-PSO algorithm. Perform 10 times, and average the results. The 3 algorithms have same execution parameters. The specific parameters are as follows:

- **PSO**, the number of particles is set to 10, the number of iterations is set to 30.
- **ACO**, the number of ants is set to 10, the number of iterations is set to 30. $\alpha = \beta = 0.5, \rho = 0.9$.
- **ACO-PSO** algorithm, the particles (ant) number is set to 10, PSO part iterations number is set to 20, the ACO part iterations number is set to 10. ACO section $\alpha = \beta = 0.5, \rho = 0.9$.

Through simulation experiments, the comparison of different algorithms’ spending time on implementing the tasks is presented in Figure. 2 and Figure. 3.

![Figure 1. The Flowchart of Scheduling Method.](image)

![Figure 2. ACO and ACO-PSO Task Spending Time Comparison.](image)
From Figure 2 and Figure 3, we can conclude that ACO-PSO algorithm takes less time than that single algorithm takes with the increasing task number. In other words, ACO-PSO algorithm has better resource scheduling effect.

5 CONCLUSION

Aiming cloud computing resource scheduling problem, propose a cloud computing resource scheduling method based on Particle Swarm Optimization and Ant Colony Optimization. Firstly, get a good initial solution quickly through PSO algorithm. Then, through ACO, use the initial solution to determine the generation of initial pheromone distribution which is used to calculate the optimal solution through iterative. The simulation results proved that ACO-PSO algorithm has better optimization capability. Only the time factor in task execution is considered in this paper, the actual situation is more complex, more factors have to be considered. Therefore, it need further research to make cloud computing task scheduling method more efficient.

6 ACKNOWLEDGMENT

This paper is funded by the project of The State Grid Corporation of China in 2014 “Research and development on the information safety threat analysis technology during the network access process”.

REFERENCES

[1] Liu, Y., Zhao, Z., Li, X., Kong, L., Yu, S. & Yu, Y. 2012. Resource Scheduling Strategy Based Optimized Generic Algorithm in Cloud Computing Environment. Journal of Beijing Normal University (Natural Sciences), no. 4, vol. 48, Aug. 2012: 378.
[2] Zhong, Q., Xie, T. & Chen, H. 2000. Task Matching and Scheduling by Using Genetic Algorithms. Journal of Computer Research & Development., no. 10, vol. 37, Oct. 2000: 1198-1201.
[3] Chen, J. & Pan, Q. 2008. Discrete Particle Swarm Optimization Algorithm for Solving Independent Task Scheduling. Computer Engineering., no. 6, vol. 34, Mar. 2008: 214-218.
[4] Chen, Z. 2010. Resource Allocation for Cloud Computing Base on Ant Colony Optimization Algorithm. Journal of Qingdao University of Science and Technology (Natural Science Edition), no. 6, vol. 33, Dec. 2012: 620-622.
[5] Hua, X., Zheng, J. & Hu, W. 2010. Ant Colony Optimization Algorithm for Computing Resource Allocation Based on Cloud Computing Environment. Journal of East China Normal University (Natural Science), no. 1, Jan. 2010: 128-133.
[6] Wang, D. & Li, Z. 2013. A Task Scheduling Algorithm Based on PSO and ACO for Cloud Computing. Computer Applications and Software., no. 1, vol. 30, Jan. 2013: 291-292.
[7] Xiong, C., Feng, L., Chen, L. & Su, J. 2012. Study on Task Scheduling Algorithm Based on Genetic Algorithm in Cloud Computing. Journal of Huazhong University of Science and Technology (Natural Science Edition), vol. 40, sup. 1, Dec. 2012: 2-3.
[8] Li, J. & Peng, J. 2011. Task Scheduling Algorithm Based on Improved Genetic Algorithm in Cloud Computing Environment. Journal of Computer Applications, no. 1, vol. 31, Jan. 2011: 184-186.
[9] Tian, H., Xie, F. & Ni, J. 2011. Resource Allocation Algorithm Based on Particle Swarm Algorithm in Cloud Computing Environment. Computer Technology and Development., no.12, vol. 21, Dec. 2011: 23-24.
[10] Wen, X. 2013. Study on Resources Scheduling Based on ACO Algorithm and PSO Algorithm in Cloud Computing. M.S. thesis, Dept. Software, Jiangxi Normal University, Nanchang, Jiangxi: 22.