ATLANTIS PRESS

Advances in Engineering Research, volume 114

International Conference on Advances in Materials, Machinery, Electrical Engineering (AMMEE 2017)

Improved Ant Colony Scheduling Algorithm for High-score Data Storage Model Based on Cloud Platform

Peng Zhang

Nanometer Information Department of Suzhou Industrial Park Institute of Services Outsourcing, Suzhou, 215123, China

zhangp@siso.edu.cn

Keywords: Cloud Platform, Ant Colony, QoS.

Abstract. In order to find the optimal scheduling, to ensure the global optimization, and to adapt to the dynamic environment, the adaptive capacity remote sensing data cloud platform scheduling strategy is proposed based on the cloud platform high score data storage model. The main contents are as follows. The characteristics of the task of high score data in cloud environment are analyzed, and the applicability of ant colony algorithm in task scheduling of cloud computing platform is pointed out. The initial set of ant colony algorithm is optimized by combining the two parts. The quality of service (QoS) is used as the ant colony algorithm pheromone, and the ant colony algorithm scheduling strategy with QoS and double pheromone updating model are designed. It can be observed that in the process of dealing with large-scale data tasks, the traditional ACO algorithm needs to be further considered node load balancing problem. And the scheduling algorithm to node QoS is as a pheromone, the resource node reliability, response time and load rate are taken into account as it is in task scheduling, which makes the node with better performance easy to be selected. It can be concluded that this strategy can improve the traditional ant colony algorithm and get the local optimal solution. At the same time, the load balance of the resource node is improved to a certain extent.

1. Introduction

With the unprecedented development of civil space information technology, high-resolution remote sensing data resources become more abundant (Kefayat, et al., 2015). The scale of data constantly increases, while cloud computing technology becomes more and more mature. It has become an inevitable trend of development that in the cloud computing environment to establish a unified, perfect high score remote sensing data sharing service platform (Agarwal and Jain, 2014). And it has become the focus of the current research that the system platform how to efficiently manage high scores of data and provide good service in the cloud environment (Rreed, et al., 2014).

In the cloud platform, in order to ensure large-scale high score data efficient organization and management, selecting the appropriate data storage program is needed (Xue, et al., 2014). Based on the high score data storage model of the applicable cloud environment, in order to ensure the load balancing and minimum consumption in the parallel task scheduling process, to select the appropriate scheduling strategy is needed (Tawfeek, et al., 2013). As an intelligent self-heuristic algorithm, ant colony algorithm is suitable for solving the optimal solution of multi-objective, so it is suitable for scheduling problem in cloud environment. However, due to the greedy characteristics of the ant colony algorithm, it is often easy to get the local optimal solution (Yin, et al., 2014). In order to find the optimal scheduling, to ensure the global optimization, and to adapt to the dynamic environment, the adaptive remote sensing data cloud platform scheduling strategy is proposed based on the high score data storage model.

2. Experiment

2.1 High Score Data Task Clustering

Cloud data center resources were usually provided on demand and according to the amount of the corresponding cost to pay. The resources included data processors, data storage, and bandwidth and so on. High-score data scheduling tasks usually had the following characteristics. Firstly, tasks were not dependent on each other, and it can be parallel operation. Secondly, remote sensing data Single task requests were usually large, and existing resources were often difficult to meet the requirements of multi-concurrent remote sensing data requests. Thirdly, the request task usually took the geographical attribute. High-resolution remote sensing data was different from other, it had time and space characteristics. Thus, the high score data request task also had a geographic attribute. Due to the special geographical characteristics of the data, continuous access to the same geographical location or similar geographical location of the data access speed were relatively fast. If the high score data task that had the similar location was executed, geographically different tasks were concurrently executed, then the overall efficiency of the task queue could be improved. Therefore, firstly, it was necessary to abstract and cluster high score data tasks. The classical hierarchical clustering algorithm was applied, which was suitable for use when the data size was relatively large. From the clustering form, it can be divided into two clustering methods, including aggregation and splitting. Aggregate hierarchical clustering was to regard each element as a cluster, then according to the similarity rule, and finally wait until it became a cluster or satisfy a certain condition after the stop. Based on the characteristics of remote sensing data, the average distance was took as a similarity rule.

The high score data task set was abstracted as $T = \{T_i | 1 \le i \le n\}$. Each task corresponds to a set of physical locations was $P = \{P_j | 1 \le j \le n\}$. Cluster set was $C = \{C_k | 1 \le k \le n\}$. Algorithm flow was as the following.

Step 1: $D_{min}(C_i, C_j) = |M_i - M_j|$. M_i was the average of the geographical locations P_j in the cluster C_i.

Step 2: The average distance between two clusters was calculated.

Step 3: The two clusters with the smallest distance were aggregated

Step 4: Step1 was repeated to calculate the distance between the new cluster and the other clusters.

Step 5: Step2 and step3 were repeated until the end condition was encountered and the remaining n clusters were left. The clustering was stopped when there was n remaining cluster. The choice of n based on the actual physical node resources and processing capacity to set. The task was usually much larger than the amount of resources, when the resources were not sufficient to meet the request, the clustering algorithm was started, the task will be merged into n class for subsequent calls. Therefore, the task can be divided into concurrent tasks and serial tasks. The same cluster of tasks in the implementation of the implementation of the first task of the data can be called in the buffer for a task to continue to quickly call. The tasks of different clusters were concurrently executed, it was needed to follow a certain scheduling strategy. An algorithm for dealing with high-score data concurrent task scheduling in cloud platform was proposed.

2.2 Based on the Ant Colony Algorithm Scheduling Strategy Design

The traditional ant colony algorithm was a kind of intelligent optimization algorithm, and the positive feedback mechanism was used to imitate the ant foraging way to find the optimal path. Ants distributed pheromones during the crawling process, which made the pheromone value of the optimal path higher and higher, and it was chosen by more ants to get the optimal solution. The algorithm had robustness, positive feedback and parallelism. However, it was easy to fall into the local optimal when the optimal solution problem of large-scale data was solved. Therefore, the scale of parallel candidate tasks was greatly reduced by initial classification of scheduling tasks, and then the ant colony algorithm was used to perform task scheduling. The efficiency of the algorithm was improved.

The updating rule of pheromone model was the important focus of ant colony algorithm. This principle was applied to high score data task scheduling. The task was requested as artificial ants, resources were as a node for ants routing. The cluster was clustered by clustering, and each cluster was abstracted as an ant. According to the pheromone value, the expected probability and the heuristic function, the resource node was selected. After a number of iterations, a reasonable scheduling sequence can be found as a task optimal solution. The waiting sequence task was marked as $task_i = \{task_1,\ task_2 \dots\dots task_n\}$, the cloud platform resource set is marked as $ant_j = \{ant_1,\ ant_2 \dots\dots ant_m\}$. And the ith task in the jth resource on the expected processing time ETC_{ij} was a two-dimensional matrix, the delay time of the task i was $Wait(task_i)$.

1). The heuristic function of the high score data scheduling task was as follows.

$$\eta_1 = {}^{1}/_{Wait(task_i)} \tag{1}$$

$$\eta_2 = {}^{1}/_{ETC_{ij}} \tag{2}$$

$$ETC_{ij} = \frac{Mip_i}{Cpu_j} + \frac{Data_i}{BW_j} \tag{3}$$

The first equation showed that shorter the delay time of task i was, the higher the value of the heuristic factor η_1 was. The second equation showed that the faster the processing speed of the task i on the resource j was, the higher the value of the heuristic factor η_2 was. The third equation showed that the processing time of the task i on the resource j, which the calculated amount of i was Mip_i , the calculated capacity of j was Cpu_j , the amount of data that i requested was $Data_i$, the bandwidth of j was BW_j .

2). The probability that the resource was selected was as below.

$$P_j^{ant_i}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{1s}(t)^{\beta_1}] \cdot [\eta_{2s}(t)^{\beta_2}]}{\sum_{s \in allowed_i}\{[\tau_{is}(t)]^{\alpha} \cdot [\eta_{1s}(t)^{\beta_1}] \cdot [\eta_{2s}(t)^{\beta_2}]\}}, & \text{if j} \in \text{allowed}_i \\ 0, \quad otherwise \end{cases} \tag{4}$$

The fourth equation represented the probability that the ith task ant_i chose the resource j at the moment t. α was the causal factor of the pheromone, which can reflect the effect of the residual pheromone on the path and the resource matching during the scheduling task. β Was the expected heuristic factor that represented the impact of foresight in finding the optimal path $\tau_{ij}(t)$ Showed i worked at j, which was the pheromone value. However, $\eta_{ij}(t)$ showed i worked at j, which was the inspiration factor.

3).Pheromone model design and updating strategy: The QoS of the resource node was used as the routing standard, which was pheromone. The attributes related to the QoS of the resource node usually included response time, reliability, and load rate. The formula of the pheromone was expressed by the fifth equation, and the QoS k represented the time t. During the process of a ant to find the routine, the task i was corresponding to the selected resource k of the QoS, R_1 was as constant, the sixth equation showed the attribute value calculation of the node QoS.

$$\Delta\tau_i^a = \frac{R_1}{QoS_k^t} \tag{5}$$

$$QoS_k^t = \text{Re}\; sp_k + \text{Re1}_k + Weight_k \tag{6}$$

The first was partial update of pheromone. The seventh equation showed the update of pheromone about the ant in the(i,j) , which was the pheromone that $\tau_{ij}(t+n)$ at the time of $(t+n)$ and (i,j) , then $\tau_{ij}(t)$ was the pheromone value at the time of t and (i,j) , ρ was the volatile factor of pheromone. $\Delta\tau_{ij}(t+n)$ Was the added value of pheromone at the time of $(t+n)$ and (i,j) . a was the ant number, m was the total number of ants.

$$\tau_{ij}(t+n) = \rho \cdot \tau_{ij}(t) + \Delta\tau_{ij}(t+n)$$
$$\Delta\tau_{ij}(t+n) = \sum_{\alpha=1}^{m} \Delta\tau_{ij}^{a}(t+n) \tag{7}$$

The second was global update of pheromone.

$$\Delta\tau_i^a = \frac{R_2}{QoS_{bestk}^t} \tag{8}$$

When all the tasks were allocated, the optimal scheduling scheme in the iteration can be obtained. The global pheromone update strategy was shown as the eighth equation, which R_2 was the constant. QoS_{bestk}^t Showed the QoS value of the most suitable node k that was selected at the moment$t$.

In summary, the improved ant colony algorithm was put into the high score data cloud platform scheduling environment, the implementation of the algorithm was shown in figure 1.

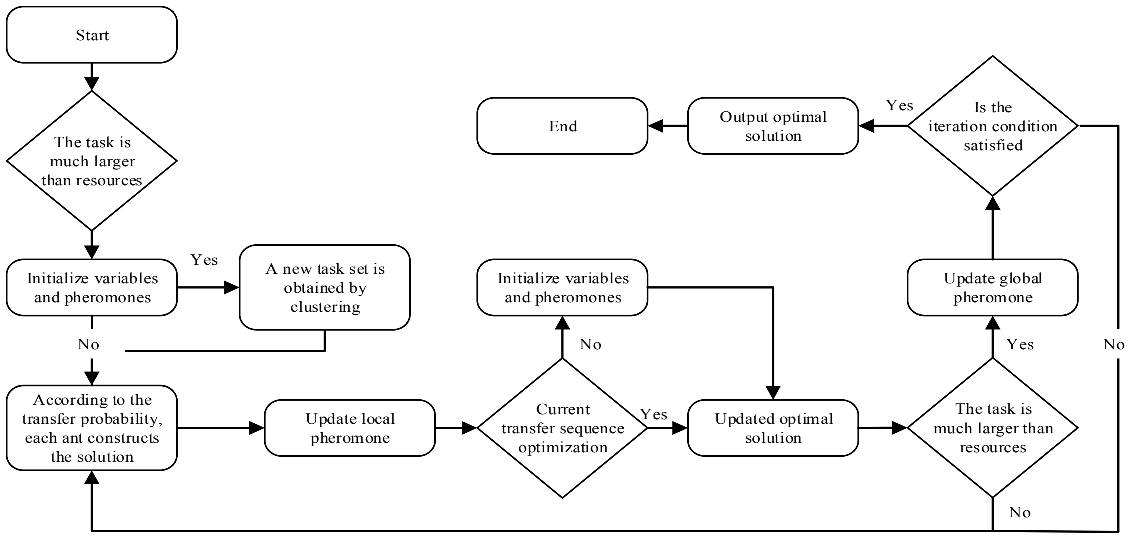
Start
The task is much larger than resources
Initialize variables and pheromones
Yes
A new task set is obtained by clustering
No
According to the transfer probability, each ant constructs the solution
Update local pheromone
Current transfer sequence optimization
Yes
No
Initialize variables and pheromones
Updated optimal solution
The task is much larger than resources
Yes
No
Update global pheromone
Is the iteration condition satisfied
Yes
No
Output optimal solution
End

Figure 1. Improved ant colony algorithm execution strategy

3. Results and Discussions

The experimental environment was consisted of a core controller and six storage servers. The master platform used a domestic server, the main configuration: 2*Intel Xeon CPU were 6 core 12 threads 2.1GHz. The memory was 64GB; the hard disk was 8TB 7200 r/min. And it was gigabit network interface. 6 storage server used the general PC, the main configuration included 1*Intel Xeon, CPU4 core 3.3GHz; memory 8GB; hard drive 1TB7200 r/min and gigabit network interface. The related operating system and software included windows server 2012, windows 7, VS2012, postgresql and so on.

In order to evaluate the performance of the modified ant colony scheduling algorithm (Q-ACO), the algorithm was simulated in the simulated cloud environment. According to the ant colony algorithm parameter setting rules and many experiments, the parameters were involved in the value, as shown in table 1.

Table 1. Parameter values of each algorithm

Algorithm name	Parameter name	Value
	α	6
	β1	4
ACO algorithm	β2	0.8
	ρ	0.3
Q-ACO algorithm	R1	1
	R2	1

The traditional ant colony algorithm ACO and the scheduling algorithm Q-ACO were realized respectively about the task completion time test in the simulation environment. Among them, the high score data requirements task range was 100 to 500, the number of iterations was 50 times, the total completion time of the ACO and Q-ACO algorithm task scheduling was counted. Through the experimental data summary, a line chart was formed, which was as shown in figure 2.

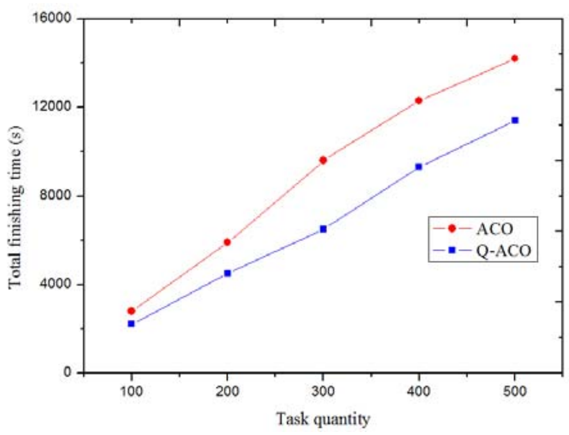

Figure 2. Comparison of the total processing time of different tasks

Figure 2 showed the total time to complete the trend under the same number of iterations, different tasks, the two algorithms for task scheduling. It can be observed that when the task was small, the completion time of the two algorithms was relatively similar. As the number of tasks increased, the task scheduling time increased, but when the task reached a certain scale, the ACO algorithm had a large increase in scheduling time. The Q-ACO algorithm was relatively flat and the Q-ACO algorithm was about 20%, which was lower than the ACO algorithm's overall task execution time. This was due to the same number of iterations. The ACO algorithm was more likely to fall into the local optimal problem for the task scheduling process under the condition of large amount of tasks. The Q-ACO algorithm firstly summarized the task of high score data, the task size was reduced. At the same time, the local update and global update of the pheromone update model wee used, and the obtained task scheduling sequence was better, thus the task processing efficiency and shortening the task execution time were improved.

Load balancing test: By calculating the real-time load weight of each resource node, and then the ninth equation was used to find the weight of each resource node as a whole load LW (load weights).

$$LW = {\varepsilon' max - \varepsilon' min}\Big/_{\varepsilon' avg} \quad (9)$$

As it was shown from the ninth equation, when the system load balance was good, the load between the resource nodes was very close, the value was closer to 0. When the load was not balanced, the value will deviate from 0. The two algorithms were recorded in the case of 100 to 500 different tasks of the server, the summary was shown as in figure 3.

From figure 3, there was obvious fluctuations about LW by the use of ACO algorithm server fluctuations, the maximum load and the minimum load difference were nearly twice as high as the average load. And there was little fluctuations in Q-ACO algorithm. And the overall was closer to 0. This was because the traditional ACO algorithm needed to be further considered when the large-scale data tasks were dealt with. And the scheduling algorithm to node QoS was as a pheromone. The resource node reliability, response time and Load rate were took into account when it was task scheduling, which made the node with better performance was greatly improved, in order that the dynamic adjustment ability of the resource node load was realized. In addition, through the local update and the global updating of the pheromone update mode, the instant update of the node pheromone was ensured, and it also saved the system overhead in some extent.

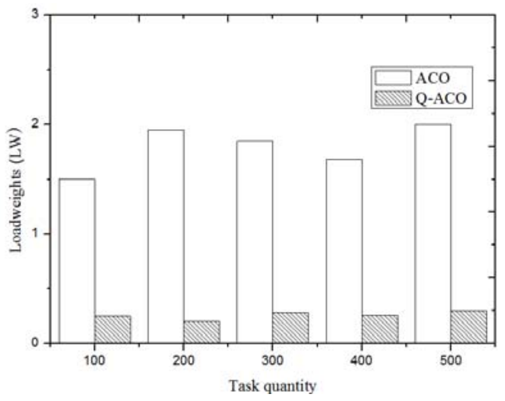

Figure 3.Comparison of server load values for different tasks

4. Conclusion

The improved ant colony scheduling algorithm is applied to the cloud platform of the national high resolution geodetic system data center system. The scheduling algorithm realizes the scientific scheduling of the system platform to the high concurrent task demand of the users and improves the utilization rate of the system resources. And the system external data sharing service is guaranteed. The good operation of the system verifies the reliability, efficiency and applicability of the research results

References

[1]. Reed, M., Yiannakou, A., & Evering, R. (2014). An ant colony algorithm for the multi-compartment vehicle routing problem. Applied Soft Computing, 15, 169-176.
[2]. Kefayat, M., Ara, A. L., & Niaki, S. N. (2015). A hybrid of ant colony optimization and artificial bee colony algorithm for probabilistic optimal placement and sizing of distributed energy resources. Energy Conversion and Management, 92, 149-161.
[3]. Yin, P. Y., Chang, R. I., Chao, C. C., & Chu, Y. T. (2014). Niched ant colony optimization with colony guides for QoS multicast routing. Journal of Network and Computer Applications, 40, 61-72.
[4]. Xue, S., Li, M., Xu, X., Chen, J., & Xue, S. (2014). An ACO-LB algorithm for task scheduling in the cloud environment. Journal of Software, 9(2), 466-473.
[5]. Tawfeek, M. A., El-Sisi, A., Keshk, A. E., & Torkey, F. A. (2013, November). Cloud task scheduling based on ant colony optimization. In Computer Engineering & Systems (ICCES), 2013 8th International Conference on (pp. 64-69). IEEE.
[6]. Agarwal, D., & Jain, S. (2014). Efficient optimal algorithm of task scheduling in cloud computing environment. arXiv preprint arXiv:1404.2076.