Application of Ant Colony Optimization in Cloud Computing Load Balancing

Zheng-Tao Wu

Application of Ant Colony Optimization in Cloud Computing Load Balancing

Wu Zheng-tao

School of information engineering, Shanghai Maritime University, Shanghai, China, 201306

Abstract. In cloud computing, because all the tasks are submitted and requested by the users, it is the key to improve the quality of service whether all tasks are scheduled to every host to achieve the shortest completion time for all tasks. In this paper, the basic ant colony algorithm is modified appropriately to enable task scheduling and load balancing. The simulation results show that the ant colony algorithm can effectively obtain the global optimal solution and realize the shortest task completion time.

Keywords: cloud computing; load balance; Ant Colony Optimization

0 Introduction

Cloud computing is a new business and computing model which is proposed in recent years. Thousands of hosts are linked together over the network to form an ultra large pool of resources. Google corporation is the first to use cloud computing service， its scale has reached millions of machines, and other companies such as Microsoft, Amazon, Aliyun and so on, also reached the scale of several hundred thousand units[1]. Depending on the type of service, cloud computing is divided into three layers, namely, IaaS (infrastructure as a service), PaaS (platform as a service), and SaaS (software as a service). No matter which layer, there is the need to balance the task of scheduling to the host, that is, load balancing. However, due to the large number of computers in the cloud computing platform, the complexity of the structure and the difference of the allocation of resources, it is difficult to achieve load balancing for such large-scale, heterogeneous and non-centralized task scheduling. The unbalanced load will lead to the decrease of system efficiency and throughput, and the instability of the daily operation, which will seriously affect the quality of service of cloud computing[2].

Ant Colony Optimization (ACO) is a kind of bionic intelligent algorithm to find the optimal solution, simulating the foraging behavior of ants. The wisdom of a single ant is not high, but it achieves an optimal population effect by sharing information among groups. Ant colony algorithm is a kind of positive feedback algorithm, which has good robustness and adaptability, and is easy to combine with other algorithms[3]. Therefore, ant colony algorithm can be used in load balancing.

The ant colony system was firstly proposed by Italy scholars Dorigo, Maniezzo and others in 1990s. When they studied ants foraging, they found that the behavior of a single ant was relatively simple, but the whole ant colony could show some intelligent

behavior. For example, an ant colony can find the shortest path to a food source in different environments. This is because ants in the ant colony can transfer information through some kind of information mechanism. Further studies have shown that ants release a substance called "pheromones" on their path.

The ants in the colony can apperceive pheromones, they will follow the path with high concentration pheromone, and every ant will leave the pheromones on the road, which forms a kind of positive feekback mechanism. Thus, after a period of time, the whole ant colony will arrive at the food source along the shortest path.

The basic idea of applying ant colony algorithm to solve the optimization problem is to use the path of the ant to represent the feasible solution of the problem to be optimized. All the paths of the whole ant colony form the solution space of the optimization problem. The shorter path ants release more pheromones, and as time goes on, the pheromone concentration on the shorter path increases gradually, and the number of ants in the path is increasing. Finally, the whole ant will concentrate on the best path under the positive feedback. At this time, the corresponding solution is the optimal solution of the problem to be optimized.

Researchers based on the study of ACO use traveling salesman problem as a test of performance, not others. In this paper, the basic ACO is modified accordingly, using cloudsim platform as simulation experiment, and design, modify the corresponding parameters, in order to design a load balancing algorithm.

1 Mathematic Model of Task scheduled

ACO is a new bionic intelligent algorithm. The first ant guides and influences the selection of the following ants by means of pheromone, so as to achieve the goal of cooperation among ants. The application of ant colony algorithm in the task scheduling of cloud computing is studied in this paper.

In a data center of cloud computing, there are n virtual hosts, they are set as $Vm_1, Vm_2, Vm_3, \cdots, Vm_n$,Each virtual host has a different MIPS (millions of instructions per second). At the same time, there are m tasks that need to be scheduled to work on the n virtual hosts. The total amount of time spent in these tasks is:

$$\mathrm{T}_{total} = \max(T_1, T_2, \cdots, T_n) \tag{1}$$

Where T_i indicates that the i virtual host completes the time allocated to all of its tasks, calculated by formula (2) and formula (3).

$$\mathrm{t}_j = \frac{MI_j}{MIPS} \tag{2}$$

$$\mathrm{T}_i = \sum_{j=1}^{k} t_j \tag{3}$$

In the above two formulas, suppose that the performance of platform I virtual host is MIPS, and there are k tasks allocated on it, the number of instructions corresponding to each task is MI_j(Million Instructions), so the time for each task is t_j.

2 Ant Colony Optimization

2.1 Theory of Ant Colony Optimization

After finding the food, the ants will randomly explore the way from home to food, and leave on the road with pheromones that are inversely proportional to the length of the journey. Subsequent ants will be based on the concentration of pheromones, select the nearest road, so as to achieve the group to find the best path of the results. If the shortest path is interrupted, the ants will randomly explore and converge quickly to the new shortest path.

The basic ACO in the TSP (traveling salesman problem) on the state as follows: at time 0, all the ants and n city will initialize m ants randomly in a city as a starting point, each edge between the city has an initial pheromone concentration $\tau_{ij}(0)$, Each ant places its city as the first element of the tabu list. Each ant then selects the next city to move, based on the probability function of the two parameter(α, β) (formula (6)), moving from the city I to the city J, and fill out the corresponding tabu list. After a cycle of N times, all ants go through all the cities and fill out the tabu list. After that, calculate the path length of each ant L_k, and then update corresponding pheromone $\Delta\tau_{ij}^k$ according the formula (4)。Save the shortest path that ants walk through, and get the shortest path $minL_k$ after one traversal，And empty all the tabu tables and start the next traversal loop. Repeat the process until the number of iterations reaches the required number of iterations(N_C),get the shortest path $minN_C$ in the global situation。

$$\tau_{ij}(\mathrm{t}+1) = \rho\tau_{ij}(\mathrm{t}) + \sum \Delta\tau_{ij}^k \tag{4}$$

where $\tau_{ij}(\mathrm{t}+1)$ is the updated pheromone concentration, ρ is the rate of pheromone emission, $\sum \Delta\tau_{ij}^k$ is the sum of pheromones left by all ants.

$$\Delta\tau_{ij}^k = \begin{cases} \dfrac{Q}{L_k} \ through\ the\ road \\ 0\ not\ through\ the\ road \end{cases} \tag{5}$$

where Q is a constant，L_k is the path length of ants.

The probability function of the first k ant moving from city I to city J is:

$$p_{ij}^k = \begin{cases} \dfrac{(\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}{\sum_k (\tau_{ij})^{\alpha}(\eta_{ij})^{\beta}}, j \in J(i) \\ 0 \quad , other \end{cases} \tag{6}$$

where $J(i)$ means In the tabu list of K, the J city has not passed yet.

In summary, fig.1 shows the flow chart using ACO to solve the TSP problem.

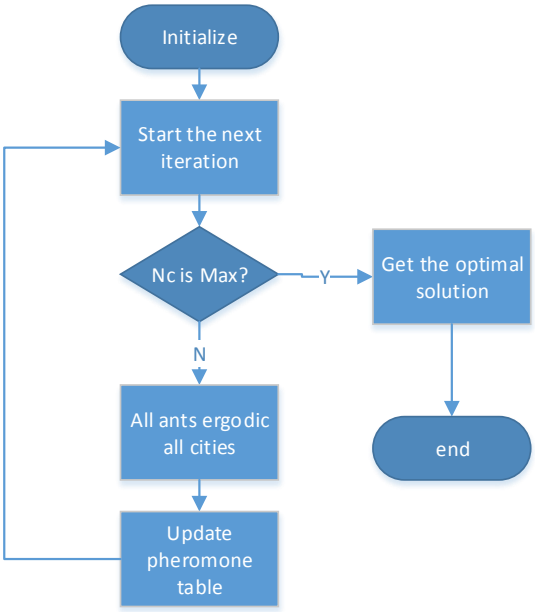

Fig.1.ACO solve TSP problem

2.2 Realization of Ant Colony Optimization

In the solution of the shortest time, m ants were placed in the initial position, according to the rules of a certain probability choice，every ant can arrange Vm for next task, until all tasks completed, and then pheromone increment table will be updated based on the total time. After all ants have finished their work, the best solution is selected from all candidate allocation and the historical optimal solution, and it is recorded, then the homologous pheromone table is updated. Then the pheromone is corrected, simulating the ants releasing pheromone on the optimal path and the natural volatilization of pheromone, leading the ant to find the global optimal solution. Fig.2 shows the flow chart of the algorithm implementation.

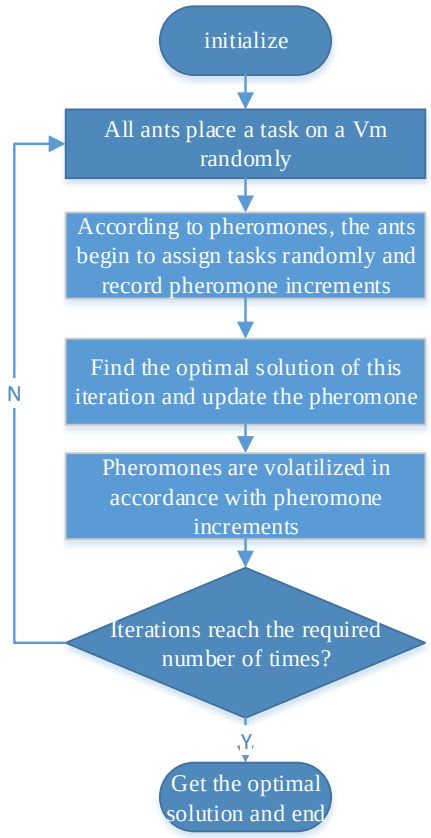

Fig.2 the flow chart of the algorithm implementation

(1) calculation rules of pheromone increment table

After each walk, each ant leaves a certain amount of pheromones according to the length of the path. The more ants there are in the same path, the more pheromones they leave behind at the same time. The pheromone increment table records the size of the pheromone emitted by the ants after each cycle. In this paper, the pheromone increment with Δ_{ij}^{k}, Δ_{ij}^{k} calculation rules such as formula (7) for:

$$\Delta_{ij}^{k}=\begin{cases}\dfrac{Q}{L}, & j \in Vm_i \\ 0, & j \notin Vm_i\end{cases} \tag{7}$$

In the formula, the Q is expressed as a constant, and L is the total time after which the current ant is assigned the task. Each ant will leave pheromones on its path, and in the end, the global pheromone increment is the sum of the pheromone left by all the ants.

(2) calculation rule of task transfer probability

When the ant chooses the virtual host for the next task, it is determined by the pheromone left before it and a heuristic information. The probability formula for

assigning the task J to the virtual host I is shown in formula (6), where τ_{ij} is pheromone concentration, η_{ij} is the heuristic information. The formula of η_{ij} is given by formula (8) and (9):

$$\eta_{ij} = \frac{1}{D_{ij}} \tag{8}$$

$$D_{ij} = \frac{Task_i}{MIPS_i} + \frac{L_j}{BW_i} \tag{9}$$

Where $Task_i$ represents the total number of tasks allocated on table I Vm, $MIPS_i$ is the execution speed of the CPU for this Vm, L_j is the length of the task to be assigned, and BW_i is the interface bandwidth of this Vm, which is a constant in this paper. α, β are control parameters that control the relative importance of pheromone and heuristic information.

(3) Amendment rules for pheromone tables

When all ants completing one iteration and getting an optimal solution, pheromone table is needed to have a comprehensive revision according to the pheromone emitted by ants and the phenomenon that simulates the natural evaporation of residual pheromones, the correction rule is such as formula (10):

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \sum \Delta_{ij}^{k} \tag{10}$$

Where $\tau_{ij}(t+1)$ is the pheromone table after correction，$\tau_{ij}(t)$ is pheromone table before correction，ρ is the control parameter, indicates the degree of evaporation of pheromones，$\sum \Delta_{ij}^{k}$ is the sum of pheromone increments left by all ants. The purpose of pheromone correction is to assign as many pheromones as possible to the optimal allocation mode, and better guide the selection of ants. The rule will not only store the pheromones left by the ants, but will also evaporate them properly.

2.3 Algorithm parameter design

According to the above principle and various rules, this paper has designed and stipulated a series of parameters. In the formula (7), the value of the constant Q will affect the left ant pheromone on global pheromone influence, it will weaken the influence of a better path on later ants if Q is too small，otherwise it will faster convergence which can easily fall into local optimal solution. According to the task length and the MIPS value of Vm, repeated experiments were conducted, and finally 100 was selected. In the formula (6) and (10), α，β，ρ are control parameter, where α，β respectively indicate the importance of pheromone and heuristic information, and ρ is the volatilization rate of pheromones. Studies have shone that when α is 1, the resulting optimum is better than the other. When value of β is from 1 to 5, the optimal solution quality will gradually become better, and when β is more then 5, the quality of the solution can be reduced, therefore, 5 is assigned to β. P value changes between 0.3~0.9, the solution changed little, when its value is 0.5, the quality of solution is the best, ρ is suggested to value of 0.5[5]. In formula (9), MIPS is randomly chosen in 100~300, and BW is the interface bandwidth. In this paper, the default value of simulation software is taken. The task length of all tasks is also random, with a range

of 10000~50000.

3 Simulation and results

This paper uses cloudsim3.0 simulation platform to do simulation experiments. By inheriting the DatacenterBroker class, we extend the DatacenterBrokerACO class to ACO, and compile the Ant class and ACO class. We can regulate single ant activity and ant colony activity separately.

3.1 Introduction of cloudsim

Cloudsim is a cloud computing simulation platform developed by Melbourne University in Australia on the basis of Gridsim. CloudSim has several versions, and has now grown to version 3 of CloudSim. CloudSim 3 uses a layered architecture. Simulation layer provide a support for the configuration and simulation of virtual data center environment simulation for cloud computing, including virtual machines, memory, capacity, and bandwidth interface. This layer is used to study strategy of virtual machine to the host distribution, and the implementation of virtual machine scheduling function expansion core. The top layer is the user code layer, which provides basic entities such as hosts, applications, virtual machines, user data and application types, and agent scheduling policies. CloudSim 3 provides virtualization engines that build and manage multiple virtualized tasks on data center nodes and flexibly switch between time sharing and spatial sharing policies when virtualized services are allocated. Cloudsim 3 is open source, users can expand the open source code by expanding the interface, realizing their own scheduling strategy, and test and verify the corresponding function based on specific environment and configuration.

3.2 Result of Simulation

Before the simulation starts, the paper specifies some parameters in the ant colony algorithm, as shown in table 1:

Table.1. parameters of ACO

Q	α	β	ρ
100	1.0	5.0	0.5

In order to ensure the heterogeneity of all virtual hosts, Java is used to randomly specify 20 different MIPS hosts. The range of MIPS is 100~300, and the value of MIPS is shown in table 2:

Table.2.the MIPS of 20 Vm

VmId	1	2	3	4	5
Mips	228	299	108	204	152

VmId	6	7	8	9	10
Mips	275	202	278	109	195
VmId	11	12	13	14	15
Mips	208	294	196	143	140
VmId	16	17	18	19	20
Mips	209	292	194	212	255

At the same time, 100 tasks with different task lengths are generated randomly, and the length is 10000~50000.

In order to determine the optimal solution, the number of ants is taken as half of the number of virtual hosts, 1000 iterations, and the final total running time is 813.1402877697841. Task assignment is shown in table 3:

Table.3. task allocation

VMID	TASK ID							
0	27	73	97	50	16	15		
1	72	0	56	58	68	94	42	14
2	9	84	43					
3	3	74	46	54	13	17		
4	11	49	86	24				
5	90	88	98	93	47	8		
6	7	71	62	30				
7	63	59	22	37	55	95	87	
8	38	70	77					
9	51	89	75	10				
10	52	36	91	18				
11	85	44	53	92	45	12	19	
12	33	61	99					
13	29	39	83	35				
14	32	78	4					
15	28	66	1	26	20			
16	6	21	64	67	40	82		

17	31	34	57	80	81		
18	23	65	96	76	2	60	48
19	25	69	79	5	41		

4 Conclusion

In this paper, a method of load balancing for Cloud Computing Based on ant colony algorithm is proposed, and simulation experiments are carried out. The method guarantees that all virtual hosts can receive task assignments as evenly as possible, and that the total time of task completion is as short as possible. It can be seen that ACO has the following characteristics in load balancing:

As the ants continue to spread pheromones to the environment, new information will soon be updated to the environment, with dynamic characteristics.

Each ant makes its own choice according to the environment, and also sends out its pheromones. Therefore, ACO has distributed features.

Ants share environmental information and influence environment at the same time, so ACO has the characteristics of cooperative operation.

These characteristics are consistent with the super large scale, heterogeneous and real-time characteristics of cloud computing, so ACO is important for load balancing. Although ACO can solve the problem of load balancing, but also has the following problems need further study:

Load balancing is a multi-dimensional and multi constrained QoS problem. In this paper, the shortest time problem in load balancing is discussed, as well as the minimum transfer and the economic principle.

ACO needs to initialize some parameters, and the choice of parameters has an important influence on the performance and the final results. Therefore, in the cloud computing environment, the setting of various parameters needs further study.

The search time of ACO is too long, and the convergence speed is slow. How to combine ACO with neural network and particle swarm algorithm to improve the performance and efficiency of the algorithm in order to meet the requirements of response time and response speed of cloud computing.

References

1. Liu Peng. Cloud computing (Third Edition) [M]. Beijing: Publishing House of electronics industry, 2015.
2. Li Fenghua. Research on load balancing scheduling algorithm of cloud computing resource based on ant colony algorithm [D]. Yunnan: Yunnan University, 2013.
3. Zhang Ning. Research on task scheduling of cloud computing based on adaptive ant colony algorithm [D]. Dalian: Dalian University of Technology, 2015.
4. Kong Lingjun, Zhang Xinghua, Chen Jianguo. Basic ant colony algorithm and its improvement [J]. Journal of Beihua University, 2004, 5 (6):572-574.

5. Ye Zhiwei, Zheng Zhaobao. The parameters alpha, beta in the ant colony algorithm, with an example of TSP problem [J]. Journal of Wuhan University, 2004 of P set, 29 (7):597-601.
6. Wang Xiajun.Cloudsim, cloud computing simulation tools, research and application of [J]. micro computer applications, 2013, 29 (8):59-61.
7. Tian Qinghua.An Abstract of Ant Colony Optimization [J].Shijiazhuang: Journal of Shijiazhuang Vocational Technology Institute,2004,16(6):37-38.
8. Zhang Hang,Luo Xiong.Status and Prospect of Ant Colony Optimization Algorithm [J].Hunan:Information and Control,2004,33(3):318-324.