2016 International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery

An Improved Ant Colony Optimization Algorithm for Improving Cloud Resource Utilization

NIE Qingbin[1,2]
[1]College of Mobile Communication
Chongqing University of Posts and Telecom
Chongqing, China
E-mail: 270104318@qq.com

LI Pinghua[1,2]
[2]College of Foreign Languages
Jinjinang College, Sichuan University
Meishan, China
E-mail: 746064110@qq.com

***Abstract*: In order to improve the cloud computing utilization, an Improved Ant Colony Optimization (IACO) is proposed. The proposed IACO algorithm improves pheromone factors and inspired factors innovatively based on the existent algorithms. Emulation tests are conducted in the CloudSim and the results indicate that IACO is superior to the conventional ACO and the latest IABC in task executing efficiency.**

Keywords-Cloud computing; Ant colony algorithm; utilization; task allocation;

I. INTRODUCTION

Cloud computing is a computing model by which the customers get access to the dynamically retractile virtue resources via the Internet. Cloud system distributes pertinent resources needed to execute tasks based on customers' demands. One of the key targets of this system is to distribute resources in a more rational way to customers, and thus optimize the utilization of the whole system. Currently, there are various resource distributing algorithms by Cloud Computing, and among those some advanced dispatching algorithm are proposed. For example, Zhuo et al. [5] have proposed a scheduling model for cloud computing resource and Improved Artificial Bee Colony Algorithm (IABC) is proposed in this work. This cloud computing model integrates an optimal value of an individual at present and random rectors into the bee colony searching process to accelerate the searching rate and enhance the searching ability. Simulation results in that paper show that IABC does improve resources utilization in cloud computing and reduce the time period for accomplishing tasks. In addition, Zhang et al. [6] proposed a task scheduling algorithm based on Load Balancing Ant Colony Optimization in Cloud Computing. This algorithm keeps load balance by adjusting pheromone factors and improving updating rules for pheromone factors. Of all the experiments and study on the algorithms of cloud resource distribution done by researchers home and abroad, task aims vary from one particular algorithm to another. Some algorithms set the ultimate goal as to reduce task executing time to the minimum, trying to speed up executing rate while reducing task executing time. Others aim at obtaining load equality to the highest level while taking into account the built-in attribute and load of each resource joint. Many of those algorithms are improved for higher resource distributing efficiency and load equality based on bionic ones such as ant colony algorithm, wasp colony algorithm, leapfrog algorithm, particle swarm algorithm and simulated annealing algorithm. These aforementioned algorithms improve the task distributing in cloud computing at different levels. However, none of these takes time cost and cloud system efficiency into account. In the process of task execution in cloud computing system, the working efficiency is largely dependent on the two factors: time and cost for task execution, and the two factors are the goals for almost every improved algorithm. Take it into consideration, time and cost for task execution are chosen and focused on as the prominent factors in this algorithm .
Based on the existing ant colony algorithms, this paper proposes the Improved Ant Colony Optimization (IACO) which aims at improving pheromone factors and inspired factors by reducing time period and cost. This algorithm optimizes resources utilization effectively with the reduced time period and the lowest cost.

II. ANT COLONY ALGORITHM

We first introduce the conventional Ant Colony algorithm. Suppose the number of ants in a colony is n, and the number of cities is m. The ant colony problem can be formally defined as follow:

Distribute n ants randomly to m cities, and the probability for ant k in city i at time point to choose the next city j is:

$$p_{ij}^{k}(t)=\begin{cases}\dfrac{[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}}{\sum\limits_{k\in allowed_k}[\tau_{ij}(t)]^{\alpha}[\eta_{ij}(t)]^{\beta}} & \text{j}\in allowed\\ 0 & \text{j}\notin allowed\end{cases} \tag{1}$$

where $\tau_{ij}(t)$ indicates concentration of pheromone factors in path (i,j) at time point t ; $\eta_{ij}(t)$ indicates inspired factors in path (i,j) at time point t and defines $\eta_{ij}(t)=\dfrac{1}{d_{ij}}$, d_{ij} indicates the length of path (i,j) . α is the inspired factors for the pheromone factors, β is the expected inspired factors, which indicates the importance of heuristic function.

In Eq. (1) the function of $tabu_k(k=1,2\cdots n)$ makes the records of cities through which the ant passes, and the cities which have been visited will be put into the forbidden list. Parameter $allowed_k$ indicates the next city note which will be visited by ant k.

II. Task Allocation by the advanced ant colony algorithm

In this section, we discuss our IACO for cloud computing to improve the efficiency while the cost is considered.

A. Discription o f Task Allocation by Cloud Computing

The IACO process of resource allocation by cloud computing can be described as follow.

At first, in cloud computing, n separate tasks will be distributed to m virtue machines $(m<n)$. Parameter $T=\{t_1,t_2,\cdots t_n\}$ indicates task congregation, parameter $VM=\{vm_{1,}vm_{2,}\cdots vm_m\}$ stands for virtue machine congregation. The distribution relationship between tasks and virtue machines can be described by matrix k:

$$k=\begin{pmatrix} k_{11} & k_{12}\cdots & k_{1n} \\ k_{21} & k_{22}\cdots & k_{2n} \\ \vdots & \vdots & \vdots \\ k_{m1} & k_{m2}\cdots & k_{mn} \end{pmatrix} \qquad (2)$$

where the matrix entry k_{ij} indicates the distributing relationship of task t_i in virtue machine vm_j. The optimization goal is to allocate the resource allocation to achieve the optimal utilization for the whole could system.

B. Time Cost Constraint Function

Let tc_{ij} be the executing time period of task t_i in virtue machine vm_j. In addition, the executing time period tc_{ij} is defined by the combination of time period ta_{ij} in which the task is translated to the virtue machine and the time period tb_{ij} in which the task is processed in this virtue machine. The combination can be denoted by:

$$tc_{ij}=ta_{ij}+tb_{ij} \qquad (3)$$

The function of tb_{ij} can be described as follow:

$$tb_{ij}=\frac{tasksize_i}{cp_j} \qquad (4)$$

where $tasksize_i$ indicates the length of task t_i.

Let cp_j be the calculating capacity of virtue machine vm_j, which can be described by the function as follow:

$$\begin{aligned} cp_j &= vm\_num_j \times vm\_mips_j \\ &+vm\_bw_j + wm\_mem_j \end{aligned} \qquad (5)$$

where vm_num_j represents the amount of processors in virtue machine vm_j, and vm_mips_j is the calculating rate of virtue machine vm_j, and vm_bw_j indicates the available network bandwidth of the virtue machine vm_j, and vm_mem_j is the RSM of the machine.

Let W be the resource allocation plan by the task allocation algorithm. Because all the tasks distributed in the virtue machines are executed at the same time, the time period for finishing all the tasks in the system is the maximization of the tc_{ij}, which is described by $tc_{\max}$.

$$tc_{\max}=\max(tc_{ij}) \qquad (6)$$

where $utc(vm_j)$ is the cost of resource consumption by the virtue vm_j machine in each time unit.

The total cost for finishing all the tasks from an allocation plan W can be described as following, in which all the $utc(vm_j)$ of each virtue machine are combined by time period tc_j:

$$\text{cps(W)}=\sum_{j=1}^{m} tc_j \times utc(vm_j) \qquad (7)$$

where $tc_{\min}$ is the processing time period of one task proposed by customers in the virtue machines with the best performance, and it can be define as :

$$tc_{\min}=\frac{\sum_{i=1}^{n} tasksize_i}{m\times\max(\text{cp}_j)} \qquad (8)$$

where $tc_{\max}$ represents the processing time period of one task proposed by customers in the virtue machines with the worst performance, and it can be define as :

$$tc_{\max}=\frac{\sum_{i=1}^{n} tasksize_i}{m\times\min(\text{cp}_j)} \qquad (9)$$

where m indicates the number of virtue machines, $\max(\text{cp}_j)$ and $\min(\text{cp}_j)$ represents the calculating capacity of virtue machines with the best and worst performance respectively.

Thus the time restraint function is as follow:

$$tc(\text{W})=\frac{tc(\text{W})-tc_{\min}}{tc_{\max}-tc_{\min}} \qquad (10)$$

As indicated in Eq. (10), the lower the value of $tc(\text{W})$ is, the shorter the task processing time is.

Let $vmcps_{\min}$ be the cost of processing one task proposed by customers in the virtue machine with the lowest $utc(vm_j)$. Furthermore, let $vmcps_{\max}$ be the cost of processing one task proposed by customers in the virtue machine with the highest $utc(vm_j)$. The parameters $vmcps_{\min}$ and $vmcps_{\max}$ can be evaluated by:

$$vmcps_{\min} = tc_{\min} \times \min(\mathrm{utm}(vm_j)) \quad (11)$$

$$vmcps_{\max} = tc_{\max} \times \max(\mathrm{utm}(vm_j)) \quad (12)$$

Further, the time restraint function can be defined as follow:

$$cps(\mathrm{W}) = \frac{cps(\mathrm{W}) - vmcps_{\min}}{vmcps_{\max} - vmcps_{\min}} \quad (13)$$

Eq. (13) indicates that the lower the value of $cps(\mathrm{W})$ is, the less the task processing cost is.

The restraint function of resource selection $c(\mathrm{W})$ can be defined as:

$$c(\mathrm{W}) = t \times tc(\mathrm{W}) + c \times cps(\mathrm{W}) \quad (14)$$

where $t \in [0,1]$ is time factor, $c \in [0,1]$ is cost factor. It has $t + c = 1$, when $t = 0.5, c = 0.5$, the cost is relatively low and the processing time is the shortest.

C. The Advanced Transferred Porbablity Equation

We then define pheromone factors in this section. The advanced equation of information concentration rate at time point $t+1$ in virtue machine vm_j is:

$$\tau_{ij}(t+1) = (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \quad (15)$$

$\Delta\tau_{ij}(t)$ indicates the increment of information concentration rate, and it can be modified by function in Eq. (14). $c(\mathrm{W})$ and the advanced factor can be defined as follow:

- When an ant reaches the corresponding allocation plan by searching the route, all the virtue machines in this route will update the pheromone factors in the local scale. At this time point $\Delta\tau_{ij}$ can be defined as follow:

  $$\Delta\tau_{ij}(t) = \frac{Z_1}{c(\mathrm{W}_{inc})} \quad (16)$$

  where Z_1 indicates constant value; W indicates the allocation plan for ant i in iteration nc.

- When all the ants have searched the routes and found the best routes in this iteration, all the virtue machines in the route will have updated the pheromone factors on the whole. $\Delta\tau_j$ can be defined as follow:

  $$\Delta\tau_{ij}(t) = \frac{Z_2}{\min(c(\mathrm{W}_{inc}))} \quad (17)$$

  where $\min(c(\mathrm{W}_{inc}))$ indicates the best allocation plan in iteration nc for the task, Z_2 is constant value.

With the above definition, we present the IACO steps in the following Section.

III. CALCULATION PROCESSES

Our proposed IACO has the following procedure for effective resource allocations:

- The first step is to initialize all the pheromone values of virtue machines in cloud computing, and define the time cost of resource allocation and the related parameters in the loaded restraint function and the finishing conditions.
- The second step is to put ants in the virtue machines randomly, and each ant will choose the appropriate resource for the next task based on the advanced transferred probability equation.
- The next step is that after an ant has finished its searching task, which means that it has reached the best allocation plan, virtue machines in the route will update the pheromone factors according to Eq. (16).
- This is the step that after all the ants finish searching for virtue machines, all the machines will be updated the allocated on the whole according to equation (17).
- In the end, IACO evaluate the iteration nc, if $nc < nc_{\max}$, then $nc = nc + 1$, jump to step 3. Otherwise, the calculation is finished, and the optimal allocation plan is output.

In the next Section, we present simulation result to verify the proposed IACO procedure and present the results.

IV. RESULTS ANALYSIS

In order to test the feasibility and effectiveness of the proposed IACO algorithm, simulation tests are conducted in the platform CloudSim for cloud computing.

In this experiment, the number of data centers in simulator on CloudSim is 5, and the tasks number from 50 to 200. The task length is 5000-15000MI (Million Instruction). There are all together 25 virtue machines with capacity 500~2000MIPS. RAM is 512-2048MB, and bandwidth is 5000-10000b/s. The value of $rcu(vm_j)$ which means consumption cost in each time unit, is 10($).

We use the following parameters in the algorithm in the simulation:

$$\alpha = 2, \beta = 3, \rho = 0.05, t = 0.5, c = 0.5, a = \frac{1}{3},$$

$$b = \frac{1}{3}, c = \frac{1}{3}, \Delta\tau_{ij} = 0, nc = 30$$

Experiments are compared by the traditional algorithm ACO, the latest algorithm IACO, and IABC in the same conditions. The processing time and costs are compared and the results are illustrated by graphs as follow where the cost unit is ($) and time unit is second.

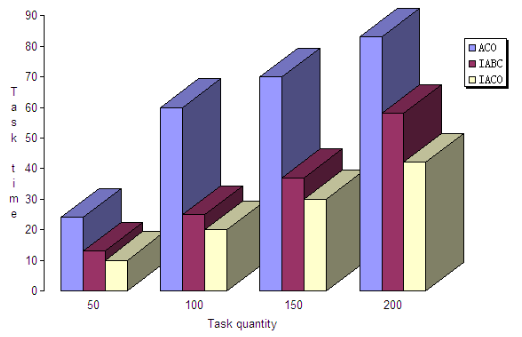

Figure 1.Compare the Task time

As is shown in graph 1.1, processing time by algorithm LBACAO and TCLB-EACO does not make a great difference when the number of tasks is 50. But as the amount of tasks increases with the same amount, the distributing time by algorithm TCLB-EACO is apparently shorter than that of algorithm ACO and LBACO.

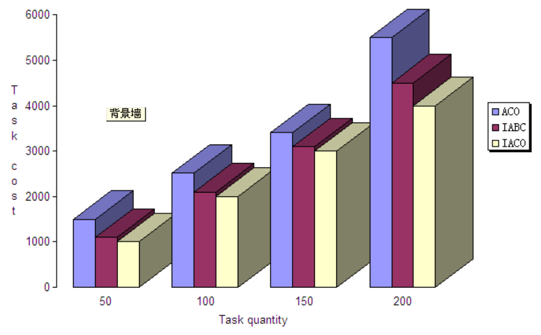

Figure 2: Compare the Task Cost

As is shown in graph 1.2, processing cost by algorithm LBACAO and TCLB-EACO does not make a great difference when the number of tasks is 50,100,150. But as the amount of tasks increases, the distributing cost by algorithm TCLB-EACO is apparently lower than that of algorithm ACO and LBACO.

From the simulation results of the comparison above, we can see that this algorithm TCLB-EACO for task and resource distributing leads to not only shorter time and lower costs, but also a higher utilization rate for the whole system.

V. CONCLUSION

The advanced ant algorithm IACO based on time, cost and load balance is proposed in this paper in order to optimize task allocation in cloud computing. Test results show that IACO is superior to IABC and ACO in reducing processing time and cost and optimizing resource utilization rate in the cloud system.

REFERENCES

Article in a journal:

[1] LI Jian-feng and PENG Jian, “Task scheduling algorithm based on improved genetic algorithm in cloud computing environment”, Journal of Computer Application, vol. 31, Jan. 2011, pp.184-187.

[2] NING Bin, GU Qiong, WU Zhao, YUAN Lei and HU Chun-yang, “Bats algorithm research in cloud computing resource scheduling based on membrane computing”, Application Research of Computers, vol. 3, May 2015, pp. 830-833.

[3] LUO Jian-ping and LI Xia Chen Min-rong, “Improved Shuffled Frog Leaping Algorithm for Solving CVRP”, Journal of Electronics & Information Technology, vol. 33, Feb. 2011, pp. 429-434.

[4] ZHA Ying-hua and YANG Jing-li, “Task scheduling in cloud computing based on improved ant colony optimization”, Computer Engineering and Design, vol. Dec. 34, 2013, pp. 1716-1816.

[5] ZHUO Tao and ZHAN Ying, “Scheduling Model Cloud Computing Resource Based on Improved Artificial Bee Colony Algorithm”, Microelectronics & Computer, vol. 31, Jul. 2014, pp. 147-150.

[6] ZHANG Huan-qing, ZHANG Xue-ping, WANG Hai-tao and LIU Yan-han. “Task Scheduling Algorithm Based on Load Balancing Ant Colony Optimization in Cloud Computing”, Microelectronics & Computer, vol.5, May 2015, pp. 31-40.

[7] E. Dodonov and R. deMell, “A novel approach for distributed application scheduling based on prediction of communication events”, Future Generation Computer Systems, vol. 26, May 2010, pp. 740-752.

[8] M. Brototi, D. Kousik and D. Paramartha, “Load Balancing in Cloud Computing using Stochastic Hill Climbing-A Soft computing Approach”, Procedia Technology, vol. 4, 2012, pp. 783-789.

[9] M. Alakeel, “A guide to dynamic load balancing in distributed computer systems”, International Journal of Computer Science and Network Security, vol. 10, Jun, 2010, pp. 153:160.

[10] M. Cristian, P. Elina and G. G. Carlos, “An ACO-inspired algorithm for minimizing weighted flow time in cloud-based parameter sweep experiments”, Advances in Engineering Software, vol. 56, Jun. 2013, pp. 38-50.

[11] P.Mell and T. Grance, “The NIST definition of cloud computing (draft) ”[S]. NIST Special Publication, 2001.

Article in a conference proceedings:

[12] E Feller, L. Rilling and C. Morin, “Energy-aware ant colony based workload placement in clouds”, Proc. IEEE Symp. International Conference on Grid Computing/ACM 12th, IEEE Press, IEEE Computer society, 2011, pp: 26-33.

[13] W. Joel, R. Deepak and H. Kirsten, “A slot allocation scheduling optimizer for Map Reduce workloads”, Proc. IMC Symp. International Middleware Conference, Germany, 2010, pp. 1-20.

[14] ZHENG Z N and WANG R, “An approach for cloud resource scheduling based on parallel genetic algorithm”, Proc. Of 3 rd International Conference on Computer Research and Development, 2011, pp. 444-447.