IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received September 14, 2017, accepted October 13, 2017, date of publication November 7, 2017, date of current version December 22, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2768665

Multi-Population Ant Colony Algorithm for Virtual Machine Deployment

XUEMEI SUN[1], KAI ZHANG[1], MAODE MA[2], AND HUA SU[1]
[1]School of Computer Science and Software, Tianjin Polytechnic University, Tianjin 300387, China
[2]School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798

Corresponding author: Xuemei Sun (seesea_sun@163.com)

This work was supported in part by the Young Scientists Fund of the National Natural Science Foundation of China under Grant 61402329 and in part by the Program for the Science and Technology Plans of Tianjin, China, under Grant 16JCTPJC46800.

ABSTRACT With the recent rapid development of cloud computing technology, how to reduce the costs of a cloud data center effectively has become an important issue. The study on virtual machine deployment mainly aims at deploying virtual machine resources required by users on a physical server rationally and effectively. This paper proposes a multi-population ant colony algorithm to solve problems of virtual machine deployment. With resource wastage and energy consumption as optimization objectives, this algorithm uses multiple ant colonies for the solution and determines strategies for information exchange among ant colonies according to the information entropy of each population to guarantee the balance of its convergence and diversity. The simulation results show that this algorithm has better performance than the single-population ant colony algorithm and can reduce resource wastage and energy consumption effectively for high-demand virtual machine deployment.

INDEX TERMS Multi-population, ant colony algorithm, cloud computing, virtual machine.

I. INTRODUCTION

As an important recent reform in the computer industry, cloud computing has developed rapidly and provided almost unlimited resources, such as virtual computation, storage and networks for users. Users only need to buy their required resources from cloud providers in a pay-on-demand way [1]. To meet the increasing user demand, cloud providers, such as Amazon and Google, are deploying numerous planet-scale data centers with high energy consumption worldwide, some of which are even composed of over millions of servers [2]. Research shows that expenditures of energy consumption account for a large part of the management of data centers and that the energy consumption of server and data equipment accounts for approximately 55% of the total energy consumption. Meanwhile, because high energy consumption means higher carbon emissions, large-scale data centers will have a great influence on the environment. In fact, an important factor causing such high energy consumption of data centers is the failure to make full use of existing computing resources. In traditional data centers, the average use ratio of servers only accounts for approximately 10%-15% of the total quantity of resources most of the time. This causes an excessive supply of resources, and most energies are consumed in this way [3]. In current cloud data centers, as cloud providers aim at guaranteeing the absolute reliability of virtual resources and services provided by them, excessive supply of resources has become a common phenomenon. How to reduce energy consumption under the cloud computing environment is always a research emphasis of scholars.

The realization of virtualization technology allows cloud data centers to deploy multiple virtual machines on a single server to reduce the problem of excessive supply of resources. Therefore, the deployment of virtual machines becomes an important research hotspot of resource deployment for cloud data centers. Regarding how to deploy virtual machines with minimum use of servers, most scholars' studies are realized through analog simulation. Currently, many non-intelligent algorithms have been proposed to solve this problem. Anton and Rajkumar et al. used the method of dynamic integration of a virtual machine to make idle servers enter sleep mode in real time for the purpose of reducing energy consumption [4]. Hadi and Massoud proposed an energy-saving-type virtual machine deployment algorithm [5]. At the beginning of the algorithm, multiple virtual machine copies are first created and then deployed through dynamic planning and local search, which effectively reduces the overall energy consumption of the server. Mark and Niyato et al. proposed an evolutionary algorithm based on demand forecasting [6] and optimized the deployment scheme of virtual machines according to demand forecasting.

Virtual machine deployment can similarly be deemed as a multi-dimensional bin packing problem [7]. It is known that the bin packing problem is an NP-hard problem [8]. Therefore, some intelligent optimization algorithms for solving the problem of the deployment of virtual machines will be a research direction. The known intelligent optimization algorithm has advantages, such as good robustness, strong universality, no need for special information regarding the problems, and parallel and efficient optimal performance [9], and can effectively adapt to the solution of the deployment of virtual machines, as reported in the literature [10]–[12]. In literature [10], Nakada and Hirofuchi et al. redefined the virtual machine deployment problem as a multi-objective optimization n problem and proposed a method based on the genetic algorithm to reduce the quantity of physical machines and thus reduce energy consumption. In literature [11], Agrawal and Bose et al. proposed a grouping genetic algorithm to solve the problem of server integration when virtual machines are incompatible. In literature [12], Xu and Fortes et al. proposed a two-stage control system to solve the virtual machine deployment problem: first, the genetic algorithm is used to calculate the program of deploying virtual machines on a specific server. Then, multiple objectives, such as waste of resources, energy consumption and heat dissipation cost, are subject to fuzzy logic optimization. Literature [13] is a typical example of current studies on the solution of the virtual machine deployment problem using the ant colony algorithm. It proposed an ant colony algorithm with multi-objective optimization for solving the virtual machine deployment problem. This algorithm solves the problem by using the ant colony algorithm, with the resource utilization rate and energy consumption as optimization objectives. It optimizes the solution set obtained in each iteration using the evolutionary multi-objective optimization algorithm and finally obtains a group of Pareto sets meeting the requirements of the problem to improve the resource utilization rate of the physical machine. Literatures [14]–[19] made improvements on this basis. However, the literatures and improvement above only considered a single population and are defective in terms of the diversity of solutions compared to the multi-population ant colony. The algorithm proposed in this paper is a further optimization for Literature [13], with the introduction of the multi-population ant colony concept, and determines strategies for information exchange among ant colonies according to the information entropy of each population to guarantee a balance between convergence and diversity of the algorithm.

Therefore, the contribution of this paper is the proposal of a multi-population ant colony algorithm (MCACS) for solving the problem of the deployment of virtual machines. The algorithm, which is based on ACS [20], increases the ant colony population to increase the diversity of solutions. It mainly uses the implementation idea of arranging multi-population ant colonies and conducting multi-path search according to the setting conditions. The optimal solution for each population is sought independently and their information communication is realized through information entropy. In the convergence process, using the early optimal solution set to replace the current solution set ensures rapid convergence to the optimal solution.

The structure of this paper is as follows: section 2 introduces relevant work for virtual machine deployment. Section 3 gives the mathematical model. Section 4 introduces the virtual machine deployment algorithm based on the improved multi-population ant colony proposed in this paper in detail. Section 5 analyzes the experimental result. Section 6 summarizes this paper.

II. METHODS

A. COMPUTING MODEL

In this paper, when multiple virtual machines are deployed on the same server, the CPU use ratio of this server is defined as the sum of the CPU use ratio of all virtual machines deployed on this server; likewise, for the memory use ratio. Meanwhile, this paper sets a threshold value for the server to prevent the performance penalty caused when the CPU or memory use ratio reaches or gets close to 100%.

B. RESOURCE WASTAGE MODEL

For each server, this paper calculates potential expenditures for the waste of resources using formula (1):

$$W_j = \left(\left|L_j^p - L_j^m\right| + \varepsilon\right) \Big/ \left(U_j^p + U_j^m\right) \tag{1}$$

where W_j refers to resource wastage of the j$^{\text{th}}$ server; L_j^p and L_j^m, respectively, refer to remaining CPU and memory resources of the server; ε is a very small, positive real number, the value of which is 0.0001; U_j^p refers to the CPU use ratio; U_j^m refers to the memory use ratio.

C. ENERGY CONSUMPTION MODEL

Because the energy consumption and CPU use ratio of the server have a certain linear relation, this paper defines the energy consumption of the server as formula (2):

$$P_j = \begin{cases} \left(P_j^{busy} - P_j^{idle}\right) \times U_j^p + P_j^{idle}, & U_j^{\mathrm{p}} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

where P_j^{busy} refers to the energy consumption of the server when it is busy; P_j^{idle} refers to the energy consumption of the server when it is idle. According to the information given in literature [13], this paper sets the energy consumption when the server is busy and idle to, respectively, 215 Watt and 162 Watt.

D. OPTIMIZATION FORMULA

To serve the purpose of minimizing resource wastage and energy consumption simultaneously, the method of multi-objective optimization is used to model the virtual machine deployment problem, as shown in formulas (3)(4)(5), at the top of the next page.

$$\text{Minimize } \sum_{j=1}^{m} W_j = \sum_{j=1}^{m} \left[y_j \times \frac{\left| \left(T_{pj} - \sum_{i=1}^{n} (x_{ij} \times R_{pi}) \right) - \left(T_{mj} - \sum_{i=1}^{n} (x_{ij} \times R_{mi}) \right) \right| + \varepsilon}{\sum_{i=1}^{n} (x_{ij} \times R_{pi}) + \sum_{i=1}^{n} (x_{ij} \times R_{mi})} \right] \tag{3}$$

$$\text{Minimize } \sum_{j=1}^{m} P_j = \sum_{j=1}^{m} \left[y_j \times \left(\left(P_j^{busy} - P_j^{idle} \right) \times \sum_{i=1}^{n} (x_{ij} \times R_{pi}) + P_j^{idle} \right) \right] \tag{4}$$

$$\text{s.t.} \begin{cases} \sum_{j=1}^{m} x_{ij} = 1, & \forall i \in I \\ \sum_{i=1}^{m} x_{ij} \times R_{pi} \leq y_j \times \mathrm{T_{pj}}, & \forall j \in J \\ \sum_{i=1}^{m} x_{ij} \times R_{mi} \leq y_j \times \mathrm{T_{mj}}, & \forall j \in J \\ \mathrm{y_j}, x_{ij} \in \{0, 1\} & \forall i \in I \quad \text{and } \forall j \in J \end{cases} \tag{5}$$

where R_{pi} and R_{mi}, respectively, refer to the quantity demanded of CPU and memory resources of each VM. T_{pj} and T_{mj}, respectively, refer to the threshold value of the CPU and memory use ratio of each server; x_{ij} and y_j are two binary variables, where x_{ij} indicates whether virtual machine i is deployed on server j and y_j indicates whether server j is used.

E. THE PROPOSED MULTI-POPULATION ANT COLONY ALGORITHM

The MCACS algorithm proposed in this paper mainly uses the implementation idea of arranging multi-population ant colonies and conducting multi-path search according to the setting conditions. The optimal solution for each population is sought for independently and their information communication is realized through information entropy. In thermodynamics, entropy is mainly used to explain disordered relations [21]. Here, information entropy is mainly used for illustrating the degree of diversity of solutions. Information entropy s $= -k \sum_{i=1}^{n} pi \ln pi$. p_{i} refers to the possibility of determining state i; $pi \geq 0$, $\sum_{i=1}^{n} pi = 1$. Information entropy determines the degree of diversity of solutions and is an important basis of pheromone updating among ant colonies.

Moreover, this paper defines the process of deploying a VM on a server as a step of an ant. Correspondingly, pheromone $\tau_{i,j}$ refers to the tendency of deploying virtual machine i on server j. The algorithm process of MCACS is as follows (Fig. 1): in the initialization phase of the experiment, all parameters are initialized, the initial setting of all pheromone matrices is τ_0, and the information entropy of all populations is set as 0. In the iteration phase, each ant colony is searched for according to the current pheromone concentration and heuristic information. A group of solution sets is obtained and the information entropy of the current population is calculated. When virtual machine i is deployed on server j, pheromone $\tau_{i,j}$ on this path is updated locally. After the completion of a round of iteration, current solution sets are subject to evolutionary multi-objective optimization and current optimum solution set setP is obtained. Solutions in the current setP are updated globally. Moreover, it is judged whether ant colonies satisfy conditions for communication. If so, interpopulational pheromone exchange is conducted. After the completion of all iterations, the final Pareto set setP is the final result.

F. INITIALIZATION PHASE

At the beginning of initialization, a group of initial solutions S_0 is produced using the FFD algorithm. Then, the initial pheromone τ_0 is calculated according to S_0. This paper defines pheromone $\tau_{i,j}$ as the tendency of deploying virtual machine i on server j. The calculation of τ_0 is shown in formulas (6)(7). Where $P'(S_0)$ and $W(S_0)$, respectively, refer to the standardized energy consumption and resource wastage of initial solution S_0. P_l^{Max} refers to the peak of energy consumption of server j. N refers to the number of virtual machines. M refers to the number of servers used by S_0. The pheromone matrix is initialized according to the τ_0 obtained.

$$\tau_0 = \frac{1}{n \times (P'(S_0) + W(S_0))} \tag{6}$$

$$P'(S_0) = \sum_{j=1}^{m} \left(\frac{P_j}{P_l^{Max}} \right) \tag{7}$$

$$i = \begin{cases} \arg\max_{u \in \Omega_k(j)} \left\{ \alpha \times \tau_{u,j} + (1-\alpha) \times \eta_{u,j} \right\}, \\ \qquad q \leq q_0 \\ s, \quad \text{otherwise} \end{cases} \tag{8}$$

$$\begin{cases} \sum_{u=1}^{m} x_{iu} = 0, \quad i \in \{1, \ldots, n\} \\ \sum_{u=1}^{n} \left(x_{uj} \times R_{pu}\right) + R_{pi} \le T_{pj} \\ \sum_{u=1}^{n} \left(x_{uj} \times R_{mu}\right) + R_{mi} \le T_{mj} \end{cases} \tag{9}$$

$$p_{ij}^{k} = \begin{cases} \dfrac{\alpha \times \tau_{i,j} + (1-\alpha) \times \eta_{i,j}}{\sum\limits_{u \in \Omega_k(j)} \left(\alpha \times \tau_{u,j} + (1-\alpha) \times \eta_{u,j}\right)}, & i \in \Omega_k(j) \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

G. SEARCH PHASE

In the search phase, ants begin seeking for the next target location. In this process, the pseudorandom proportional selection rule is used to select virtual machine i as the next virtual machine to be deployed on server j, as shown in formula (8).

Where q is a random number between 0 and 1, which is used to determine whether the current process is exploration or development. q_0 will take a fixed value according to the current experience. When q is less than or equal to q_0, ants start the development process; otherwise, ants start the exploration process. α refers to the relative importance of the pheromone matrix for search.

$\Omega_k(j)$ refers to the current optional VM set that satisfies formula (9).

S is a random parameter selected according to the probability distribution of the random proportional rule. Formula (10) refers to the probability that ant k selects virtual machine i for deployment on server j.

$\eta_{i,j}$, i.e., heuristic information, refers to the availability of deploying virtual machine i on server j. In this paper, the value of $\eta_{i,j}$ is calculated dynamically according to the current ant state before each step. Its calculation follows formulas (11)(12)(13).

$$\eta_{i,j,1} = \frac{1}{\varepsilon + \sum_{v=1}^{j} \left(P_v / P_v^{Max}\right)} \tag{11}$$

The formula above represents the effect of energy consumption of the current deployment state on heuristic information. The formula below represents the influence of resource wastage on heuristic information.

$$\eta_{i,j,2} = \frac{1}{\varepsilon + \sum_{v=1}^{j} W_v} \tag{12}$$

$$\eta_{i,j} = \eta_{i,j,1} + \eta_{i,j,2} \tag{13}$$

For this multi-objective problem, this paper uses formula (12) to represent the overall availability of deploying virtual machine i on server j.

H. UPDATING OF PHEROMONE

In the MCACS algorithm, the updating of the pheromone is divided into partial updating and global updating. Whenever a virtual machine i is deployed on server j in the search process, this paper updates the current pheromone according to the local pheromone updating rule, as shown in formula (14).

$$\tau_{i,j}(t) = (1-\rho_l) \times \tau_{i,j}(t-1) + \rho_l \times \tau_0 \tag{14}$$

where ρ_l refers to the local evaporation factor, the value of which is between 0 and 1. τ_0 is the initial pheromone.

When the search for all ant populations and the elimination and updating of the current setP are completed, this paper updates all solutions S in the current setP globally. Formula (15) is the global updating rule.

$$\tau_{i,j}(t) = (1-\rho_l) \times \tau_{i,j}(t-1) + \frac{\rho_g \times \lambda}{P'(S) + W(S)} \tag{15}$$

where ρ_g refers to the global evaporation factor, the value of which is between 0 and 1. λ is a self-adaptive coefficient used to control the influence of solution S on pheromone as time progresses, the value of which is calculated using formula (16).

$$\lambda = \frac{NA}{t - NI_S + 1} \tag{16}$$

where NA refers to the number of ants, NI_S refers to the round of iteration in which solution S is put into the current setP, and t refers to the current round of iteration. This global updating program aims at improving the experience accumulation of ant search.

I. CALCULATION OF POPULATION INFORMATION ENERGY

In this paper, the process of ants selecting virtual machine i for deployment on server j in the exploration follows probability distribution p_{ij}^k. According to formula (9), $p_{ij}^k \ge 0$ and $\sum_{i \in \Omega_k(j)} p_{ij}^k = 1$. Therefore, according to the scheme proposed in the literature, the information entropy for ant k deploying virtual machine i on server j can be defined. It can be calculated using formula (17).

$$s_j^k(sa) = -\sum_{i \in \Omega_k(j)} p_{ij}^k(sa) \ln p_{ij}^k(sa) \tag{17}$$

This formula refers to the uncertainty of ant k in population sa selecting virtual machine i for deployment on server j. The expression of the information entropy of ant k in population sa can be obtained, as shown in formula (18).

$$s^k(sa) = \sum_{j=1}^{m} s_j^k(sa) \tag{18}$$

Therefore, the expression of information entropy of population sa is as shown in formula (19).

$$s(sa) = \frac{1}{NA} \sum_{k=1}^{NA} s^k(sa) \tag{19}$$

Meanwhile, formula (19) shows the average uncertainty of population sa when the virtual machine scheme is established.

This definition predicts the convergence state of an ant colony and the evolution extent of an ant population according to features of information entropy and its changing process in combination with the process of ant colony algorithm, thereby yielding the rate of convergence of the self-adaptive adjustment algorithm. Meanwhile, it is a basis of inter populational pheromone communication.

J. ELIMINATION AND UPDATING OF PARETO SET

The Pareto set is a set composed of global non-dominated solutions. In this paper, current non-dominated solutions are stored with setP before the obtainment of the final Pareto set. For solutions produced in each round of iteration, two objective functions (3) and (4) are calculated and compared with other solutions in the current iteration and those in the current setP. If a solution is non-dominated, it is added to setP and solutions dominated by it in setP are eliminated. Finally, after the completion of all iterations, the current setP is the final required Pareto set in this paper.

K. INTERPOPULATIONAL PHEROMONE COMMUNICATION

In the algorithm, the communication among populations is conducted at a certain time interval, which is not changeless; instead, it is determined according to the information entropy of all colonies, i.e., it changes with the convergence of all populations. The time interval of communication among populations satisfies formula (20).

$$gap = k_1 \cdot e^{\frac{\sum_{i=1}^{SA} s_i}{SA}} \tag{20}$$

where k_1 is a constant, SA is the number of sub-populations and s_i is the information entropy of the i^{th} sub-population.

The selection of a communication colony is determined according to the information entropy of each colony. Let colony i choose colony j as the object of information communication; j can be determined using formula (21).

$$j = \arg\max_{1 \leq j \leq SA}(|s_i - s_j|) \tag{21}$$

where s_i and s_j are information entropies of colonies i and j at the current moment, respectively. Colonies with a high entropy will choose those with a low entropy for information communication. Thus, colonies with a low information entropy have relatively centralized pheromone distribution and can balance their own pheromone distribution through communication with colonies with a high information entropy. Similarly, colonies with a high information entropy have scattered pheromone distribution and can centralize their pheromone distribution through communication with colonies with a low information entropy.

When colony j is determined as the communication object of colony i, this paper updates the pheromone according to formula (22).

$$\tau_{uv}^i = \tau_{uv}^i + \lambda \Delta \tau_{uv}^i \tag{22}$$

where $\lambda = s_i - s_j$, where $\lambda_{\min} < \lambda < \lambda_{\max}$; $\lambda_{\min}$ and $\lambda_{\max}$ are constants denoting the minimum and maximum, respectively, of the updating coefficient. $\Delta\tau_{uv}^i$ is the pheromone of sub-population j on path (u, v).

L. DESCRIPTIONS OF PSEUDOCODE OF THE ALGORITHM

The pseudocode of the MCACS algorithm proposed in this paper is as follows:

Algorithm 1 MCACS algorithm description (Algorithm 1)

Input:
Set of VMs with their associated resource demand, Set of parameters
Output:
A Pareto set P
Begin
  //Initialization
Set values of parameters α, ρ_l, ρ_g, q_0, T_{pj}, T_{mj}, *NA, SA, M*
Initialize all pheromone values to τ_0
Initialize Pareto set P as empty
  //Iterative loop
Repeat
  //construct solution
Repeat
Introduce a new colony of ants
Ants search for solution of current colony and add it to current set temp
Calculate information entropy of current colony
Until all colonies have generated a solution
  //Evaluation
If a solution in set temp is dominated by another solution in set temp or the Pareto set P **Then**
Eliminate the solution
Else
Add the solution to the Pareto set P and all solutions dominated by the solution are eliminated from the set P
  //Global updating
For all solutions in set P **do**
Apply the global updating rule
End For
  //Pheromone exchange
If the conditions fit for pheromone exchange **Then**
Apply the pheromone exchange rule
Until the max number of iterations is reached
Return the Pareto set P

III. RESULTS AND DISCUSSION

A. SIMULATION ANALYSIS

To evaluate the performance of the MCACS for virtual machine deployment, this paper conducts a multi-group simulation experiment and compares the performance of the

MCACS algorithm to the single-population ant colony algorithm (VMPACS) and FFD algorithm proposed in literature [13], [22]. The experimental procedure is written using Eclipse and Java and operated on a computer equipped with a 2.60 GHz Intel Core i5 processor and 8 GB 1600 MHz of DDR3 internal memory. The operating system is OS X Yosemite 10.10.3.

B. SIMULATION SETTING

Table 1 shows the specific parameter settings of the multi-population ant colony algorithm and single-population ant colony algorithm compared in this experiment.

TABLE 1. Experiment parameters.

Parameters	Values	Notes
Na	10	Number of ants
Sa	5	Number of colonies
M	100	Number of iterations
A	0.45	Relative importance of pheromone trail
ρ_l	0.35	Local pheromone evaporating parameter
ρ_g	0.35	Global pheromone evaporating parameter
q_0	0.8	Fixed parameter determined by the relative importance of exploitation
T_{pj}	90%	Thresholds of CPU utilizations
T_{mj}	90%	Thresholds of memory utilizations

This paper produces a problem instance randomly according to literature [23]: 200 groups of VMs with their respective CPU and memory use ratio demands and a certain number of servers meeting the worst deployment scheme. For the convenience of experimentation, all servers have the same performance parameter in this paper. The specific algorithm of the instance is as follows:

Algorithm 2 Specific examples generated by the algorithm (Algorithm 2)

for $i = 1$ **to** n **do**
$Rpi =$ **rand**$(2\overline{Rp})$;
$Rmi =$ **rand**$(\overline{Rm})$;
$r =$ **rand**(1.0)
if $(r < P \wedge Rpi \geq \overline{Rp}) \vee (r \geq P \wedge Rpi < \overline{Rp})$ **then**
$Rmi = Rmi + (\overline{Rm})$;
End if
End for

where function **rand**(a) generates a double-type return value, the value of which is distributed at [0, a] uniformly. $\overline{Rp}$ refers to the benchmark demand of the CPU use ratio;

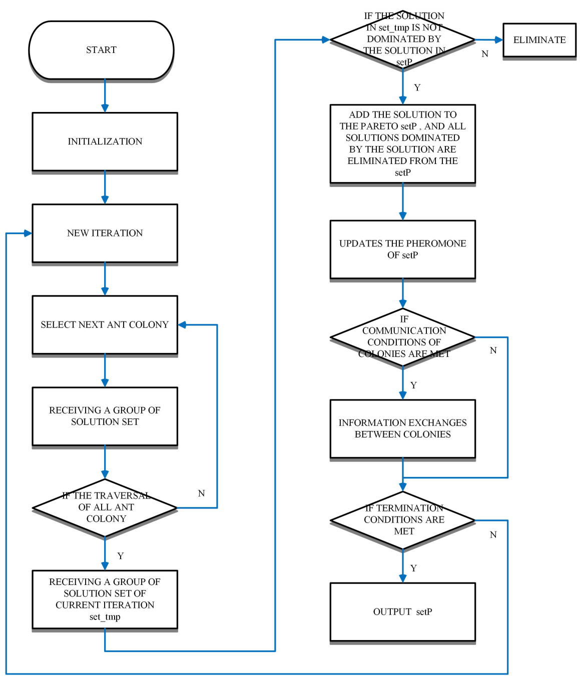

FIGURE 1. Algorithm flow chart.

$\overline{Rm}$ is the benchmark demand of the memory use ratio; P is a probability parameter, which is used to control the correlation between the CPU use ratio and memory use ratio in this paper.

This experiment has two groups of benchmark parameters and five probability parameters, which generate 200 groups of VMs. There are 2000 groups of instances in total. In this paper, $\overline{Rp}$ and $\overline{Rm}$ are set to 25% and 45%, respectively. When $\overline{Rp} = \overline{Rm} = 25\%$, the CPU use ratio and memory use ratio are distributed at [0, 50%] uniformly. When $\overline{Rp} = \overline{Rm} = 45\%$, the CPU use ratio and memory use ratio are distributed at [0, 90%] uniformly. Probability parameter P is set as 0.00, 0.25, 0.50, 0.75 and 1.0. When $\overline{Rp} = \overline{Rm} = 25\%$, the average coefficient of correlation between memory and CPU use ratios is, respectively, −0.749, −0.320, 0.123, 0.333 and 0.732. When $\overline{Rp} = \overline{Rm} = 45\%$, the average coefficient of correlation between memory and CPU use ratios is, respectively, −0.736, −0.378, 0.001, 0.375 and 0.762.

C. EXPERIMENTAL RESULTS AND ANALYSIS

To evaluate the performance of the algorithm proposed for the optimization of energy consumption and resource wastage, this experiment compares MCACS, VMPACS and FFD. The horizontal axis refers to the coefficient of correlation between the CPU and memory use ratios. Vertical axes correspond to energy consumption and resource wastage. Fig. 2-5 shows results representing the comparison of energy consumption and resource wastage in MCACS, VMPACS and FFD when $\overline{Rp} = \overline{Rm} = 25\%$ and $\overline{Rp} = \overline{Rm} = 45\%$.

According to Fig. 2, when $\overline{Rp} = \overline{Rm} = 25\%$, the MCACS algorithm has better performance than the VMPACS algo-

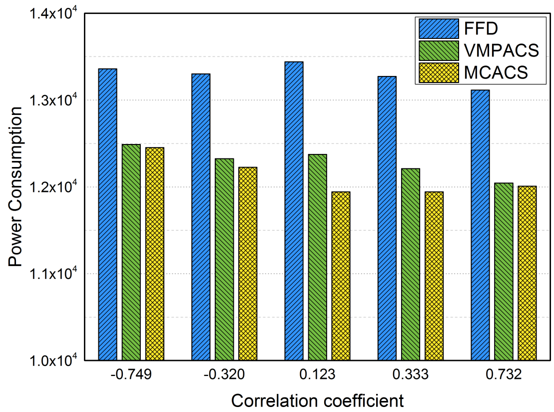

FIGURE 2. **Comparison of energy consumption ($\overline{Rp} = \overline{Rm} = 25\%$).**

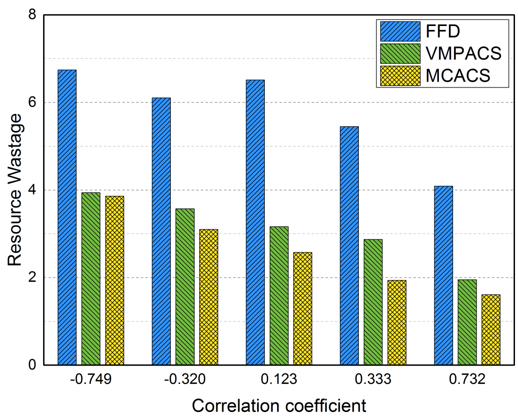

FIGURE 3. **Comparison of resource wastage ($\overline{Rp} = \overline{Rm} = 25\%$).**

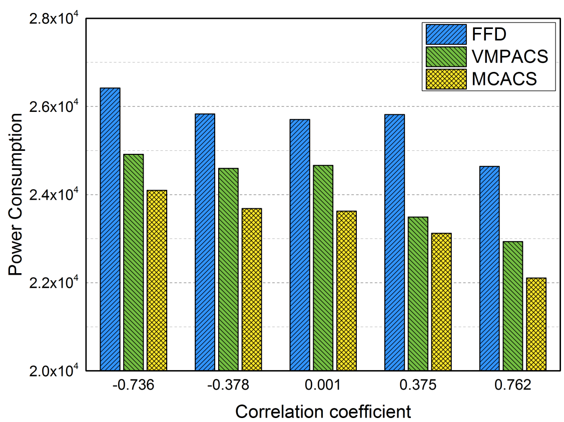

FIGURE 4. **Comparison of energy consumption ($\overline{Rp} = \overline{Rm} = 45\%$).**

rithm in most cases and its result has greater improvement compared to the result of the FFD algorithm. According to Fig. 3, when $\overline{Rp} = \overline{Rm} = 45\%$, the MCACS algorithm experiences a greater improvement of performance. Compared to the single-population VMPACS algorithm and FFD algorithm, it has lower energy consumption and resource wastage. Compared to the situation when $\overline{Rp} = \overline{Rm} = 25\%$, its optimization degree is higher. It is known that the CPU

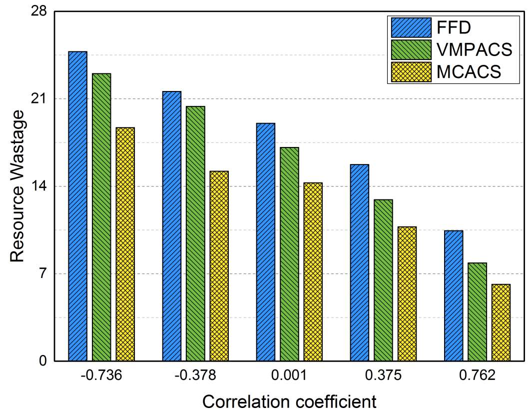

FIGURE 5. **Comparison of resource wastage ($\overline{Rp} = \overline{Rm} = 45\%$).**

TABLE 2. **Comparison of energy consumption and resource wastage ($\overline{Rp} = \overline{Rm} = 25\%$).**

correlation	algorithm	Energy consumption	Resource wastage
	FFD	13360.0075	6.740116465
-0.749	VMPACS	12487.763	3.937734441
	MCACS	12452.068	3.857396724
	FFD	13301.9565	6.105934192
-0.320	VMPACS	12323.872	3.573151541
	MCACS	12224.9855	3.099696628
	FFD	13439.286	6.510431033
0.123	VMPACS	12372.95	3.162386372
	MCACS	11943.2245	2.574478794
	FFD	13271.0935	5.448905535
0.333	VMPACS	12209.5095	2.870079328
	MCACS	11943.234	1.936115812
	FFD	13115.073	4.089105488
0.732	VMPACS	12043.4905	1.948132168
	MCACS	12007.0805	1.607041184

TABLE 3. **Comparison of energy consumption and resource wastage ($\overline{Rp} = \overline{Rm} = 45\%$).**

correlation	algorithm	Energy consumption	Resource wastage
	FFD	26416.8785	24.76813311
-0.736	VMPACS	24911.7765	23.00652206
	MCACS	24098.5355	18.68572
	FFD	25827.901	21.58120242
-0.378	VMPACS	24594.1705	20.3884245
	MCACS	23684.6825	15.20534489
	FFD	25703.1595	19.03818612
0.001	VMPACS	24661.1005	17.10133154
	MCACS	23626.9755	14.27346621
	FFD	25816.782	15.74009504
0.375	VMPACS	23490.2815	12.92682533
	MCACS	23121.4095	10.7516191
	FFD	24638.084	10.44302428
0.762	VMPACS	22934.465	7.869030958
	MCACS	22106.188	6.145850873

use ratio and memory use ratio are distributed at [0, 50%) and [0, 90%) uniformly, respectively, when $\overline{Rp}$ and $\overline{Rm}$ are 25% and 45%. Thus, the MCACS algorithm has advantages for handling the deployment problem of virtual machines with high resource demand (Tables 2 and 3).

According to the data analysis above, the optimization effect obtained by the multi-population ant colony algorithm is more obvious and superior compared to that of the single-population ant colony algorithm. This indicates that the coordination and cooperation among multi-population ant colonies through information entropy can solve the convergence problem of the algorithm better and meanwhile expand the search space and improve the diversity of solutions searched for.

IV. CONCLUSION

With the current constant development of cloud computing technology, how to deploy virtual machines on a physical server efficiently has become an important issue. This paper proposes a multi-population ant colony algorithm to solve the deployment problem of virtual machines. With resource wastage and energy consumption as optimization objects, this algorithm uses multiple ant colonies for the solution and determines strategies for information exchange among ant colonies according to the information entropy of each population to guarantee the balance of its convergence and diversity. The simulation results show that this algorithm has better performance than the single-population ant colony algorithm and can reduce resource wastage and energy consumption effectively for high-demand virtual machine deployment. Due to the current development of data centers and the constant expansion of their size, future research will further consider the execution efficiency of the algorithm, the reduction of execution time of the algorithm effectively, load balancing and network bandwidth so that it can further adapt to the current technological development.

REFERENCES

[1] K. Zhou, M. H. Afifi, and J. Ren, "ExpSOS: Secure and verifiable outsourcing of exponentiation operations for mobile cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 12, no. 11, pp. 2518–2531, Nov. 2017.

[2] M. Armbrust *et al.*, "A view of cloud computing," *Commun. ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[3] L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007.

[4] A. Beloglazov and R. Buyya, "Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers," *Concurrency Comput., Pract. Exper.*, vol. 24, no. 13, pp. 1397–1420, Sep. 2012.

[5] H. Goudarzi and M. Pedram, "Energy-efficient virtual machine replication and placement in a cloud computing system," in *Proc. IEEE 5th Int. Conf. Cloud Comput.*, Jun. 2012, pp. 750–757.

[6] C. C. T. Mark, D. Niyato, and T. Chen-Khong, "Evolutionary optimal virtual machine placement and demand forecaster for cloud computing," in *Proc. IEEE Int. Conf. Adv. Inf. Netw. Appl.*, Mar. 2011, pp. 348–355.

[7] T. Hirofuchi, H. Nakada, H. Ogawa, S. Itoh, and S. Sekiguchi, "Eliminating datacenter idle power with dynamic and intelligent VM relocation," in *Distributed Computing and Artificial Intelligence*. Berlin, Germany: Springer, 2010, pp. 645–648.

[8] J. Békés, G. Galambos, and H. Kellerer, "A 5/4 linear time bin packing algorithm," *J. Comput. Syst. Sci.*, vol. 30, no. 1, pp. 145–160, 2000.

[9] J. Rigelsford, "Intelligent optimization techniques: Genetic algorithms, tabu search, simulated annealing and neural networks," in *Industrial Robot*. Springer, 2000, pp. 5–27.

[10] H. Nakada, T. Hirofuchi, and H. Ogawa, "Toward virtual machine packing optimization based on genetic algorithm," in *Proc. Int. Work-Conf. Artif. Neural Netw.*, 2009, pp. 651–654.

[11] S. Agrawal, S. K. Bose, and S. Sundarrajan, "Grouping genetic algorithm for solving the serverconsolidation problem with conflicts," in *Proc. 1st ACM/SIGEVO Summit Genet. Evol. Comput.*, 2009, pp. 1–8.

[12] J. Xu and J. A. Fortes, "Multi-objective virtual machine placement in virtualized data center environments," in *Proc. IEEE/ACM Int. Conf. Green Comput. Commun. (GreenCom), Int. Conf. Cyber, Phys. Soc. Comput. (CPSCom)*, 2010, pp. 179–188.

[13] Y. Gao, H. Guan, Z. Qi, Y. Hou, and L. Liu, "A multi-objective ant colony system algorithm for virtual machine placement in cloud computing," *J. Comput. Syst. Sci.*, vol. 79, no. 8, pp. 1230–1242, 2013.

[14] M. H. Ferdaus *et al.*, "Virtual machine consolidation in cloud data centers using ACO metaheuristic," in *Proc. Eur. Conf. Parallel Process.*, 2014, pp. 306–317.

[15] H. Khani, A. Latifi, N. Yazdani, and S. Mohammadi, "Distributed consolidation of virtual machines for power efficiency in heterogeneous cloud data centers," *Comput. Elect. Eng.*, vol. 47, pp. 173–185, Oct. 2015.

[16] Q. Zheng, R. Li, X. Li, and J. Wu, "A multi-objective biogeography-based optimization for virtual machine placement," in *Proc. 15th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2015, pp. 687–696.

[17] Z. H. Zhan, X. F. Liu, Y. J. Gong, J. Zhang, H. S. H. Chung, and Y. Li, "Cloud computing resource scheduling and a survey of its evolutionary approaches," *ACM Comput. Surv.*, vol. 47, no. 4, p. 63, 2015.

[18] X. Sun, C. Chang, and H. Su, "Novel degree constrained minimum spanning tree algorithm based on an improved multicolony ant algorithm," *Math. Problems Eng.*, vol. 2015, pp. 1–13, Jan. 2015.

[19] Q. Zheng *et al.*, "Virtual machine consolidated placement based on multi-objective biogeography-based optimization," *Future Generat. Comput. Syst.*, vol. 54, pp. 95–122, Jan. 2016.

[20] M. Dorigo and L. M. Gambardella, "Ant colony system: A cooperative learning approach to the traveling salesman problem," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 53–66, Apr. 1997.

[21] X. Wei, "Improved ant colony algorithm based on information entropy," in *Proc. Int. Conf. Comput. Inf. Sci.*, Dec. 2010, pp. 519–520.

[22] D. S. Johnson, "Fast algorithms for bin packing," *J. Comput. Syst. Sci.*, vol. 8, no. 3, pp. 272–314, 1974.

[23] Y. Ajiro and A. Tanaka, "Improving packing algorithms for server consolidation," in *Proc. Int. CMG Conf.*, 2007, pp. 399–406.

XUEMEI SUN received the Ph.D. degree from the Tianjin University, Tianjin, China, in 2005. She is currently an Associate Professor and Master Tutor with Tianjin Polytechnic University. She has published over 20 papers in refereed conferences and journals. One of the representative papers, "Optimized Deployment of Wireless Sensor Networks Based on Ant Colony Algorithm Culture," was published in 2015 in the *Journal of Applied Mathematics and computing*. Her current research interest is cloud computing, artificial intelligence, wireless networks, and big data.

KAI ZHANG was born in Shanxi, China, in 1994. He received the bachelor's degree in computer science and technology from Tianjin Polytechnic University in 2016, where he is currently pursuing the master's degree in computer technology. During his master's degree, he studied under the supervision of A. P. Xuemei Sun and A. P. Hua Su from Tianjin Polytechnic University. His current research interests include mobile crowd sensing, experimental design, cloud computing, linear models, and data mining.

MAODE MA received the B.E. degree from Tsinghua University in 1982, the M.E. degree from Tianjin University in 1991, and the Ph.D. degree in computer science from The Hong Kong University of Science and Technology in 1999. He is currently an Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. He has authored or co-authored over 200 international academic publications. His extensive research interests include wireless networking and wireless network security. He is a Senior Member of the IEEE Communication Society and the IEEE Education Society and a member of a few technical committees in the IEEE Communication Society. He has been a member of the technical program committees for over 100 international conferences. He has been a General Chair, Technical Symposium Chair, Tutorial Chair, Publication Chair, Publicity Chair, and Session Chair for over 50 international conferences. He is an Editor-in-Chief of the *International Journal of Electronic Transport* and Topic Editor-in-Chief of the *International Journal of the Advancements in Computing Technology*. He currently serves as an Associate Editor of the IEEE COMMUNICATIONS LETTERS, Senior Editor of the IEEE COMMUNICATIONS SURVEYS and TUTORIALS, and an Associate Editor of the *International Journal of Network and Computer Applications*, the *International Journal of Security and Communication Networks*, and the *International Journal of Wireless Communications and Mobile Computing*.

HUA SU was born in China. She received the B.S. degree in computer and application from Tianjin Polytechnic University in 2000, the M.S. degree in computer and application from Tianjin Polytechnic University in 2005. Since 2000, she has been with the College of Computer Science and Software, Tianjin Polytechnic University, where she is currently an Associate Professor, and teaches some basic courses, such as computer networks. She is interested in trusted computing, mobile crowd sensing, and computer network.