Received June 15, 2017, accepted July 15, 2017, date of publication July 31, 2017, date of current version August 29, 2017.

Digital Object Identifier 10.1109/ACCESS.2017.2733723

Energy-Efficient Many-Objective Virtual Machine Placement Optimization in a Cloud Computing Environment

XIN YE[1], YANLI YIN[1], AND LAN LAN[2]
[1]Institute of Information and Decision Technology, Dalian University of Technology, Dalian 116024, China
[2]International Studies of College, Shenyang University, Shenyang 110044, China

Corresponding author: Lan Lan (lanchristy@hotmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 71533001 and in part by the Fundamental Research Funds for the Central Universities of China under Grant DUT13JS11.

ABSTRACT Cloud data centres are faced with the serious problem of increasing energy consumption. Thus, the problem of virtual machine placement for energy saving is becoming a critical issue. Considering various requirements of cloud providers and users, a many-objective virtual machine placement model is built to minimize energy consumption and maximize load balance, resource utilization, and robustness. An energy-efficient KnEA (EEKnEA) algorithm is proposed to address this problem. EEKnEA is improved by proposing an energy-efficient-oriented population initialization strategy based on the knee point-driven evolutionary algorithm (KnEA), which is a high-performance algorithm for many-objective problems. The proposed model and performance of EEKnEA are evaluated in comparison to KnEA and other algorithms. Experimental results show that the proposed model is reasonable, and the EEKnEA algorithm outperforms its counterparts on this type of problem in terms of energy saving, load balance, and robustness.

INDEX TERMS Clouds, energy-efficiency, virtual machine placement, many-objective optimization.

I. INTRODUCTION

Cloud computing, as a new computing model and business model, provides users with computing power, storage space and application services through the network in the form of services. With the development of cloud computing, the scale of cloud data centres continues to be expanded, and a cloud data centre usually runs thousands of physical machines. Virtualization technology allows a physical machine to support multiple virtual machines, and each virtual machine runs various types of services and applications that have different resource requests and load changes [1]. Virtual machine placement refers to the allocation of virtual machines to the appropriate physical machines. Reasonable placement of virtual machines is a key factor in the smooth operation and energy saving of cloud data centres and has gradually become a research focus in recent years.

With the popularity of cloud computing, energy consumption of cloud data centres is ever increasing. According to Amazon's estimation, the energy-related costs of cloud data centres account for 42% of total operating costs [2]. In addition, the increase in energy consumption has led to a dramatic increase in CO_2 emissions and directly affected our environment [3]. Consequently, reduction of energy consumption is an urgent problem that must be solved, because it will not only reduce energy costs but also be conducive to environmental sustainability.

To meet the user's QoS (Quality of Service) requirements and the dynamic and uncertain resource requirements of virtual machines, cloud data centres tend to oversupply resources. To reduce energy consumption, reduction of resource oversupply is a useful approach in addition to using more energy-efficient infrastructure and improving resource utilization.

In addition to energy saving, cloud providers must also be concerned with load balance among physical machines and resource utilization. Moreover, due to the dynamic and random requirements of virtual machines, how to obtain a more robust virtual machine placement scheme has also received increasing attention from cloud providers. However, little consideration is given to the robustness of the virtual machine placement scheme in existing studies on the virtual machine placement problem.

Most existing studies do not comprehensively consider all objectives, instead only considering some of them.

Nevertheless, these objectives must be simultaneously considered in many scenarios. Therefore, a new many-objective virtual machine placement optimization model is built to minimize energy consumption and maximize load Balance, resource utilization, and robustness. This model considers actual requirements of cloud providers and offering users a good experience; it is therefore more practical.

Thus, an improved KnEA (Knee Point-Driven Evolutionary Algorithm), named EEKnEA (Energy-Efficient KnEA), is proposed to solve the above many-objective optimization problem. KnEA is a high-performance algorithm used to solve Many-objective Optimization problems (MaOPs) [4], so KnEA is chosen as the basis of the algorithm to solve the proposed model. To improve the performance of KnEA further, an Energy-Efficient-oriented Population Initialization Strategy is proposed to improve the population initialization in KnEA.

Finally, comparative experiments are conducted to evaluate the rationality of the model and the performance of the improved algorithm.

The remainder of this paper is organized as follows. In section II, existing work related to virtual machine placement is introduced. The virtual machine placement problem and its formulation are proposed in Section III. In Section IV, an improved algorithm based on KnEA, named EEKnEA, is proposed to solve the above problem. Experiment results are presented in Section V to evaluate the performance and efficiency of the improved algorithm. Finally, the conclusion and future work are given in section VI.

II. RELATED WORK

Virtual machine placement is a critical topic in the field of cloud computing. Related studies are summarized as follows.

A. OBJECTIVES AND CONSTRAINTS OF THE VIRTUAL MACHINE PLACEMENT PROBLEM

In the field of virtual machine placement in a cloud computing environment, different studies considered different objectives for different user requirements and scenarios. The considered objectives include load Balance among physical machines [5]–[7], maximizing utilization or minimizing wastage of various types of resources [8]–[10], and minimizing energy consumption [2], [3], [11], [12]. Most studies considered one or two of the above objectives, and fewer studies considered three or more objectives.

Of the studies, those on energy efficiency can be divided into two types: maximizing resource utilization to reduce energy consumption and direct reduction of energy consumption. The former type of study takes the minimum number of physical machines as the optimization objective and does not provide a specific measurement formula for energy consumption. The latter type of study provides a definition formula for energy consumption [3], [11]. However, most of these studies only considered the energy consumption of the physical machines, ignoring the energy consumption of communication among virtual machines in the network of a cloud data centre. Moreover, the energy consumption of communication among virtual machines is not trivial. Reference [2] categorized the communication between a pair of virtual machines into four types, defined the communication energy consumption as the function of the traffic and the communication type between virtual machines and achieved the ideal placement effect.

Concerning virtual machine load uncertainty, few studies consider the robustness of placement schemes, although a robust placement scheme can satisfy the changing requirements of users and the dynamic changes of various types of resource loads on physical machines. Thus, a robust placement scheme can ensure smooth operation of the cloud data centre and reduce virtual machine migration costs caused by placement scheme changes. Therefore, robustness of virtual machine placement should be considered.

Concerning the constraints of the virtual machine placement problem, existing studies generally considered the following two types of constraint. The first type of constraint is the capacity constraint of the physical machine. In other words, the sum of resource requirements of virtual machines placed on a certain physical machine cannot exceed the resource capacity of this physical machine. Second, a virtual machine can only be assigned to a physical machine.

B. OPTIMIZATION ALGORITHM OF THE VIRTUAL MACHINE PLACEMENT PROBLEM

Different algorithms are used to solve the virtual machine placement problem depending upon the number of objectives.

The heuristic algorithm and meta-heuristic algorithm are usually used to solve the single-objective virtual machine placement problem. Because the virtual machine placement problem can also be viewed as one type of special bin-packing problem, heuristic algorithms that are used to solve the classical bin-packing problem, such as FFD (First Fit Decreasing) and BFD (Best Fit Decreasing), are often used to solve such a problem [13]. In addition, some meta-heuristic algorithms are used to solve such problems, such as GA (Genetic Algorithm) [14] and ACO (Ant Colony Optimization) [15].

Meta-heuristic and Hybrid heuristic algorithms are often used to solve multi-objective virtual machine placement problems. For example, reference [3] applied the multi-objective ACO algorithm to solve the virtual machine placement problem with the two objectives of minimizing energy consumption and maximizing resource utilization. Reference [2] extended the GA with a local optimization procedure to enhance the convergence of the original GA.

The above studies are undoubtedly the basis of this paper, but there remain the following problems that must be improved. (1) Most existing studies consider fewer than 3 objectives; thus, they address multi-objective optimization problems (MOPs). However, in many practical applications, many factors must be considered, and the quantity of placing objectives is usually greater than 3. Such a type of problem belongs to many-objective optimization problems (MaOPs). (2) Energy consumption of communication should be considered, except for energy consumption of physical machines. And, a more energy-efficient placement scheme can be obtained if the energy saving is considered in the algorithm design. (3) Because of the dynamic randomness of virtual machine loads, the robustness of the virtual machine placement scheme should be considered. (4) In terms of algorithms, some mature MOEAs (Multi-objective Evolutionary Algorithms) for the virtual machine placement problem with 2-3 objectives (MOPs) cannot effectively solve the virtual machine placement problem with 4 objectives. It is necessary to improve the effective algorithm for MaOPs, for example, with KnEA, to solve the virtual machine placement problem with 4 objectives.

III. PROBLEM FORMULATION

A. PROBLEM DESCRIPTION

In a cloud data centre, a large number of physical machines are interconnected through a network. These physical machines are virtualized into a number of virtual machines running a variety of applications. The virtual machine placement problem can be described as shown in Fig 1. n virtual machines with different resource requests are assigned to appropriate physical machines of the m physical machines to achieve the goals, including minimizing energy consumption and maximizing resource utilization, load Balance, and robustness. Each physical machine provides multiple types of resources, such as CPU (in MIPS), memory (in GB), and network bandwidth (in GBs/day). The resource capacities of physical machines and the resource requests of virtual machines are generally different. The physical machine, virtual machine and their placement relationship, and the placement scheme can be described as follows.

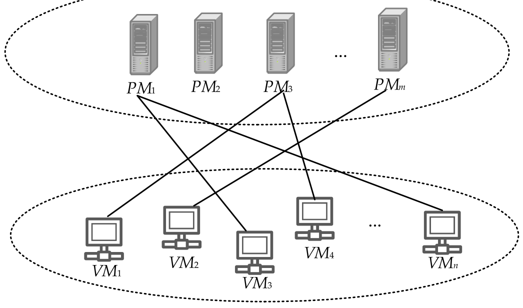


FIGURE 1. Schematic diagram of virtual machine placement problem.

1) PHYSICAL MACHINE

The set of physical machines is denoted as $PM = \{PM_1, PM_2, \cdots, PM_m\}$, and m is the number of available physical machines. Each physical machine is described formally as a triple $PM_j = (PM_j^{cpu}, PM_j^{mem}, PM_j^{band})$, $j = 1, 2, \ldots m$, where PM_j^{cpu}, PM_j^{mem} and PM_j^{band} represent the capacity of CPU, memory, and bandwidth of physical machine j, respectively.

2) VIRTUAL MACHINE

The set of virtual machines is denoted as $VM = \{VM_1, VM_2, \cdots, VM_n\}$, and n is the number of virtual machines to be placed. Each virtual machine is represented as $VM_i = (VM_i^{cpu}, VM_i^{mem}, VM_i^{band})$, $i = 1, 2, \ldots n$, where VM_i^{cpu}, VM_i^{mem} and VM_i^{band} represent the resource request amount of CPU, memory, and bandwidth of virtual machine i, respectively.

3) PLACEMENT RELATIONSHIP OF THE VIRTUAL MACHINE AND THE PHYSICAL MACHINE

The possible placement relationship between the virtual machine and the physical machine can be expressed as an $n \times m$ matrix A.

$$A = \begin{bmatrix} a_{11}, & a_{12}, & \cdots & a_{1m} \\ a_{21}, & a_{22}, & \cdots & a_{2m} \\ \cdots & \cdots & \cdots & \cdots \\ a_{n1}, & a_{n1}, & \cdots & a_{nm} \end{bmatrix}$$

where a_{ij} is a binary variable, and $a_{ij} = 1$ means that virtual machine VM_i is assigned to physical machine PM_j. For each VM_i, $\sum_j^m a_{ij} = 1$ and represents that each virtual machine can only be assigned to a physical machine.

4) PLACEMENT SCHEME

A virtual machine placement scheme can be represented as a vector:

$$\begin{gathered} VM_1, VM_2, \cdots, VM_n \\ S = (PM_x, PM_y, \cdots, PM_z). \end{gathered}$$

The length of S is the same as the number of virtual machines. Each element of S corresponds to a virtual machine, and the value of the element indicates the physical machine to which the virtual machine is assigned. For example, $x = 5$ indicates that the first virtual machine is assigned to physical machine 5.

B. ENERGY CONSUMPTION MODEL

The energy consumption considered in this paper includes the energy consumption of physical machines and the energy consumption of communication among virtual machines.

1) ENERGY CONSUMPTION OF PHYSICAL MACHINES

The energy consumption generated by the physical machine primarily includes the energy consumption of the CPU, memory and network interface. Moreover, compared with other system resources, the CPU accounts for most of the energy consumption. Therefore, the energy consumption of the physical machine need only consider the energy consumption of the CPU [2], [16]. Studies have shown that the energy consumption of a physical machine can be described by a linear function of its CPU utilization [17]. Additionally, studies have shown that the energy consumption of the idle physical machine is approximately 70% of the fully CPU-utilized physical machine [18], proving that shutting down an always-idle physical machine in a cloud data centre can reduce energy consumption. (We assume that physical machines are turned off when there is no virtual machine assigned to them.)

The energy consumption of the j-th physical machine can be defined as (1).

$$En(PM_j) = \begin{cases} P_{idle} + (P_{busy} - P_{idle}) \times U_j^{cpu}, & U_j^{cpu} > 0 \\ 0, & U_j^{cpu} = 0 \end{cases} \tag{1}$$

where P_{idle} and P_{busy} represent the average energy consumption when the physical machine is idle and fully utilized. U_j^{cpu} represents the CPU utilization of the j-th physical machine.

2) ENERGY CONSUMPTION OF COMMUNICATION AMONG VIRTUAL MACHINES

In addition to the energy consumption of physical machines, communication among virtual machines can also generate energy consumption. The magnitude of this type of energy consumption depends upon the network topology distance between virtual machines and the amount of communication data.

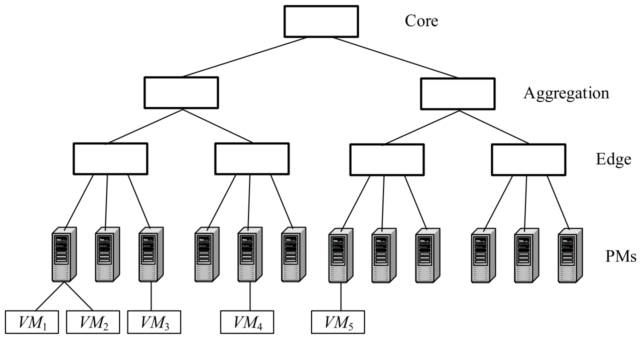


FIGURE 2. **Communication Network in Data Centres.**

The topology of the communication network in a cloud data centre is generally a tree. The communication between a pair of virtual machines can be classified into four categories according to the topological distance between the virtual machines [2]. As shown in Fig 2, the communication between VM_1 and VM_2 is an instance of the first type, and they are placed on the same physical machine. The first type of communication does not use any network communication device; thus, there is almost no energy consumption. The communication betweenVM_1 and VM_3 belongs to the second type. These two virtual machines are placed on different physical machines, but under the same edge. Their communication must use one network communication device. The communication between VM_1 and VM_4 is an example of the third type. They are placed on different physical machines under different edges, but under the same aggregation. Three network communication devices are used when they communicate. The fourth type of communication is the communication type of a pair of virtual machines that are placed on different physical machines under different aggregations. The communication between VM_1 and VM_5 is an example of this type, and five network communication devices are used. The more communication devices that are used for communication, the more energy that is consumed.

Let C_1, C_2, $C_3$and C_4 denote the sets of virtual machine pairs belonging to four different communication types. $C = C_1 \cup C_2 \cup C_3 \cup C_4$ represents the set of all virtual machine pairs. In accordance with the different types of communication, the energy consumption of transferring a unit of data between virtual machines in a virtual machine pair c can be defined as follows.

$$e(c) = \begin{cases} e_1, & \text{if } c \in C_1 \\ e_2, & \text{if } c \in C_2 \\ e_3, & \text{if } c \in C_3 \\ e_4, & \text{if } c \in C_4 \end{cases} \tag{2}$$

Let $l(c)$ be the amount of data transferred between virtual machines in c. The energy consumption of communication between these two virtual machines can be expressed as follows.

$$En(c) = l(c)^{*}e(c) \tag{3}$$

In summary, the total energy consumption of the cloud data centre is expressed as (4).

$$En = \sum_{j=1}^{m} En(PM_j) + \sum_{c \in C} En(c) \tag{4}$$

C. ROBUSTNESS MEASURE OF VIRTUAL MACHINE PLACEMENT SCHEME

In the cloud computing environment, various types of applications are deployed on the virtual machine. Demands of virtual machines for all types of resources are different because the applications are different and the time and frequency of users accessing related applications are different. Moreover, these demands show strong dynamism and randomness. The dynamic changes in demand result in dynamic changes of resource load for the virtual machine.

Placing virtual machines based on peak load can cause many idle resources, increasing costs and energy consumption. However, placing virtual machines based on the average load can result in a lower QoS and user experience. Consequently, only a strong robust virtual machine placement scheme can meet the needs of a variety of resource load dynamic changes and achieve a balance of cost, energy consumption, QoS and user experience.

For the dynamic load of virtual machines, the expectation and standard deviation of a virtual machine load can be calculated based on historical log data. For a given virtual machine, the expectation of one type of resource load represents the average demand level of that type of resource in the process of asking the cloud data centre for a resource. Moreover, the standard deviation of one type of resource load reflects how much the virtual machine load fluctuates. Therefore, the virtual machine placement scheme can achieve better robustness if the appropriate space is reserved by using the expectation of a virtual machine's resource demand as a reference and considering the level of a virtual machine's load fluctuations. Based on the idea mentioned above, the robustness indicators of a virtual machine placement scheme are constructed as follows.

Let $E(VM_i^{cpu})$, $E(VM_i^{mem})$, and $E(VM_i^{band})$ denote the expected value of CPU, memory, and bandwidth demand, respectively, of virtual machine i. Let $\sigma(VM_i^{cpu})$, $\sigma(VM_i^{mem})$, and $\sigma(VM_i^{band})$ be the standard deviations of the above three types of resource demand of virtual machine i, respectively. Then, using the CPU resource as an example, the reserved CPU resource of physical machine PM_j can be calculated with (5).

$$RE_j^{cpu} = PM_j^{cpu} - \sum_i E(VM_i^{cpu}) \cdot a_{ij}, \quad j = 1, 2, \cdots m \tag{5}$$

As mentioned above, the probability of physical machine overload is related to the size of the space reserved on the physical machine and the standard deviation of the resource load of virtual machines assigned to the physical machine. On the one hand, the larger the resource reserved space of the physical machine, the smaller the possibility of physical machine overload. However, reserving greater resources may result in lower resource utilization and higher energy consumption. On the other hand, when the resource reserved space of the physical machine is smaller and load fluctuations of virtual machines allocated on this physical machine are larger, the physical machine is more likely to overload. Thus, the greater the load fluctuations of virtual machines allocated on a physical machine, the more space on this physical machine that should be reserved.

Based on the above analysis, the resource reserved space of one physical machine should be multiplied by a certain weight to represent the robust performance of this physical machine. For example, the weight of reserved CPU resource space on physical machine PM_j is defined as (6).

$$w_j^{cpu} = \frac{\sum_i^n \sigma(VM_i^{cpu}) \cdot a_{ij}}{\sum_i^n \sigma(VM_i^{cpu})} \tag{6}$$

Further, the robustness measure of CPU resources reserved on PM_j can be expressed as follows.

$$\begin{aligned} R_j^{cpu} &= w_j^{cpu} \times RE_j^{cpu} \\ &= \frac{\sum_i^n \sigma(VM_i^{cpu}) \cdot a_{ij}}{\sum_i^n \sigma(VM_i^{cpu})} (PM_j^{cpu} - \sum_{i=1}^n E(VM_i^{cpu}) \cdot a_{ij}) \end{aligned} \tag{7}$$

Similarly, the robustness measures of memory and network bandwidth can also be expressed as (8) and (9).

$$R_j^{mem} = \frac{\sum_i^n \sigma(VM_i^{mem}) \cdot a_{ij}}{\sum_i^n \sigma(VM_i^{mem})} (PM_j^{mem} - \sum_{i=1}^n E(VM_i^{mem}) \cdot a_{ij}) \tag{8}$$

$$R_j^{band} = \frac{\sum_i^n \sigma(VM_i^{band}) \cdot a_{ij}}{\sum_i^n \sigma(VM_i^{band})} (PM_j^{band} - \sum_{i=1}^n E(VM_i^{band}) \cdot a_{ij}) \tag{9}$$

The total robustness measure of physical machine j can be formulated as (10).

$$R_j = R_j^{cpu} \cdot R_j^{mem} \cdot R_j^{band} \tag{10}$$

The average of the robustness measures of all physical machines represents the robustness measure of virtual machine placement Scheme S, as shown in (11).

$$R = \frac{1}{m'} \sum_{j \in PM'} R_j \tag{11}$$

where the set PM' represents the set of physical machines already used in scheme S and m' represents the number of physical machines in the set PM'.

D. RESOURCE UTILIZATION AND LOAD BALANCE

To satisfy the diverse requirements of the virtual machine placement problem, we also consider the resource utilization and load balance of physical machines with the exception that reducing energy consumption and robustness are considered.

1) RESOURCE UTILIZATION

Higher resource utilization can not only reduce resource wastage but also reduce the number of physical machines used. In other words, higher resource utilization can reduce energy consumption and the costs of a cloud data centre. Therefore, it is necessary to ensure higher resource utilization of the virtual machine placement scheme. According to the utilization ratio of each resource of each physical machine, the average utilization ratio of each resource of all physical machines can be obtained. Then, the resource utilization of scheme S can be obtained as shown in (12)–(14).

$$U_j^{cpu} = \frac{\sum_i^n \mathrm{E}(VM_i^{cpu}) \cdot a_{ij}}{PM_j^{cpu}} \tag{12a}$$

$$U_j^{mem} = \frac{\sum_i^n E(VM_i^{mem}) \cdot a_{ij}}{PM_j^{mem}} \tag{12b}$$

$$U_j^{band} = \frac{\sum_i^n E(VM_i^{band}) \cdot a_{ij}}{PM_j^{band}} \tag{12c}$$

$$\overline{U^{cpu}} = \frac{\sum\limits_{j\in PM'} U_j^{cpu}}{m'} \tag{13a}$$

$$\overline{U^{mem}} = \frac{\sum\limits_{j\in PM'} U_j^{mem}}{m'} \tag{13b}$$

$$\overline{U^{band}} = \frac{\sum\limits_{j\in PM'} U_j^{band}}{m'} \tag{13c}$$

$$U = (\overline{U_{cpu}} + \overline{U_{mem}} + \overline{U_{band}})/3 \tag{14}$$

where U_j^{cpu}, U_j^{mem}, and U_j^{band} represent the utilization ratio of the CPU, memory and network bandwidth of the physical machine PM_j. $\overline{U^{cpu}}$, $\overline{U^{mem}}$, and $\overline{U^{band}}$ represent the average utilization ratio of the CPU, memory and network bandwidth, respectively, of scheme S. U is the overall resource utilization ratio of scheme S.

2) LOAD BALANCE

The purpose of load balancing is to ensure that each physical machine resource is effectively allocated and equitably used, thereby satisfying the user's QoS requirements [19]. Load balancing of the virtual machine placement scheme is defined as follows [20].

$$L = \left(\sqrt{\sum_{j\in PM'} (U_j^{cpu} - \overline{U^{cpu}})^2} + \sqrt{\sum_{j\in PM'} (U_j^{mem} - \overline{U^{mem}})^2} + \sqrt{\sum_{j\in PM'} (U_j^{band} - \overline{U^{band}})^2}\right)/3 \tag{15}$$

E. MODEL OF VIRTUAL MACHINE PLACEMENT

Based on the previous analysis, the optimization model of the virtual machine placement problem can be formulated as follows.

$$\min: f_1 = En \tag{16}$$

$$\min: f_2 = c - R \tag{17}$$

$$\min: f_3 = 1 - U \tag{18}$$

$$\min: f_4 = L \tag{19}$$

$$\text{s.t.} \sum_{j}^{m} a_{ij} = 1, \quad i = 1, 2 \ldots n \tag{20}$$

$$\sum_{i}^{n} a_{ij} \cdot E(VM_i^{cpu}) \le PM_j^{cpu}, \quad j = 1, 2 \ldots m \tag{21}$$

$$\sum_{i}^{n} a_{ij} \cdot E(VM_i^{mem}) \le PM_j^{mem}, \quad j = 1, 2 \ldots m \tag{22}$$

$$\sum_{i}^{n} a_{ij} \cdot E(VM_j^{band}) \le PM_j^{band}, \quad j = 1, 2 \ldots m \tag{23}$$

Expressions (16)–(19) represent four optimization objectives, namely, energy consumption minimization, optimal robustness, resource utilization maximization, and load variance minimization of physical machines, where c in formula (17) represents a constant. To normalize the model, the robustness and resource utilization rate are subtracted by a fixed constant to transfer to minimize. Expression (20) indicates that a virtual machine can only be assigned to one physical machine. Expressions (21), (22), and (23) represent the three types of resource capacity constraints of the physical machines; that is, the sum of all types of resource requests by virtual machines placed on a physical machine cannot exceed the resource capacity of that physical machine.

IV. OPTIMIZATION ALGORITHM FOR VIRTUAL MACHINE PLACEMENT

The virtual machine placement problem proposed in Section III is a Many-objective Optimization problem. Multi-objective evolutionary algorithms that are used to solve 2 or 3 objectives are often not suitable to solve such a problem. Therefore, this paper proposes an improved algorithm based on KnEA, namely, EEKnEA, because KnEA is more effective for solving MaOPs. The improvements primarily include two aspects. On the one hand, the chromosome coding method, the crossover and mutation operators are designed based on the characteristics of the virtual machine placement problem. On the other hand, an Energy-Efficient-oriented Population Initialization Strategy is proposed to ensure the quality of the initial population and accelerate the convergence. The flow chart of the proposed algorithm EEKnEA is shown as Fig. 3.

The chromosome encoding method, related operators, and Energy-Efficient-oriented Population Initialization Strategy are presented as follows.

A. CHROMOSOME ENCODING

From section III, the goal of the virtual machine placement problem is to find the optimized placement scheme $S = (PM_x, PM_y, \cdots, PM_z)$. Therefore, a chromosome represents a feasible placement scheme, and the chromosome length is equal to the number of virtual machines to be placed. The value of each gene on the chromosome represents the physical machine ID to which the virtual machine is assigned. For example, as shown in Fig. 4, the value of the first gene is 15, meaning that virtual machine VM_1 is assigned to PM_{15}.

B. MAIN OPERATORS

1) CROSSOVER OPERATOR

Because the chromosome is encoded as an integer, a single-point crossover operator is chosen. That is, two chromosomes are randomly selected in the parents, and the crossover point of the chromosomes is randomly selected. Further, sequences beyond the point of either selected chromosome are swapped. After crossing, the newly generated chromosomes are examined to determine whether the crossed chromosomes still meet all constraints and to repair the unsatisfied chromosomes.

2) MUTATION OPERATOR

Each gene on a chromosome is randomly mutated with a certain probability. After the mutation, the new chromosomes

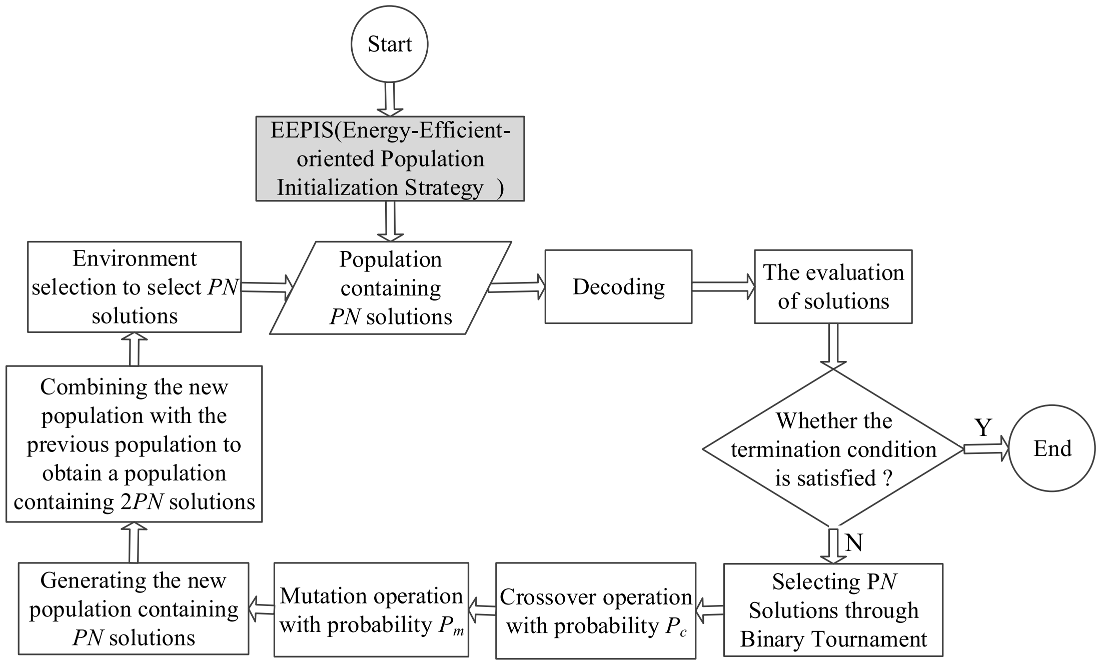

Start
EEPIS(Energy-Efficient-oriented Population Initialization Strategy)
Environment selection to select PN solutions
Population containing PN solutions
Decoding
The evaluation of solutions
Combining the new population with the previous population to obtain a population containing 2PN solutions
Whether the termination condition is satisfied ?
Y
End
N
Generating the new population containing PN solutions
Mutation operation with probability P_m
Crossover operation with probability P_c
Selecting PN Solutions through Binary Tournament

FIGURE 3. Algorithm flow chart of EEKnEA.

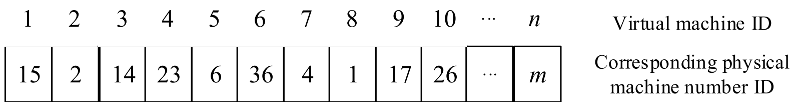

1 2 3 4 5 6 7 8 9 10 ··· n
Virtual machine ID
15 2 14 23 6 36 4 1 17 26 ··· m
Corresponding physical machine number ID

FIGURE 4. Chromosome example.

are also examined for feasibility, and the necessary repair operation is performed for unfeasible chromosomes.

3) SELECTION OPERATOR

The selection operator of KnEA is adopted in this paper, and its main operations are described briefly as follows. First, the solution set is sorted according to the Pareto dominating relationship among solutions, and some Pareto fronts are obtained. The distance between each solution and corresponding hyperplane is calculated. Then, a percentage of the solutions are identified as knee points according to the distance and adaptive neighbourhood strategy. Solutions are selected based on the following three evaluation criteria. (1) Solutions in the front with smaller No. are preferred. (2) Knee points are more preferred than non-knee points. (3) Solutions with a greater distance to the hyperplane are preferred. The priority relationship among the above three evaluation criteria is (1) $\succ$ (2) $\succ$ (3). After the above three criteria are applied, if the number of selected solutions remains less than the population size, the remaining number of solutions are randomly selected from the remaining solutions [4].

4) REPAIRING OPERATOR

The unfeasible chromosomes obtained after crossover or mutation operation need to be repaired to maintain feasibility. For an unfeasible chromosome, there are one or more physical machines violate the resource capacity constraint, that is total resource demand of virtual machines that placed on them exceed to their resource capacity. So, these physical machines that unsatisfied the capacity constraint are obtained firstly. And then, for each of these physical machines, one or more virtual machines placed on it are reallocated to other physical machines with sufficient free space to ensure the physical machine satisfies the resource capacity constraint.

C. ENERGY-EFFICIENT-ORIENTED POPULATION INITIALIZATION STRATEGY

To enhance the optimization effect and even accelerate convergence, an Energy-Efficient-oriented Population Initialization Strategy (EEPIS) is proposed to improve the population initialization in KnEA. There are two main reasons for this strategy.

On the one hand, the energy consumption of communication among virtual machines is an important part of the total energy consumption for cloud data centres. The energy consumption of communication is closely related to the amount of data transferred between virtual machines and the topology distance between the physical machines on which the virtual machines are placed. When two virtual machines are placed on the same physical machine, the energy consumption of

Algorithm 1 Energy-Efficient-Oriented Population Initialization Strategy

```
Require: PN (population size), VM(the set of virtual machines), PM(the set of physical machines), n (the number of VM),
  m (the number of PM), VM_request (the resource request of VM),PM_capacity (the resource capacity of PM), VM_commu
  (the amount of data transferred via communication among virtual machines)
1. obtain the set of sorted virtual machine groups $G_k$, $G_{k-1}$, ..., $G_2$ according to VM_commu
2. determine the number of selected virtual machine groups $N_j$ in each $G_j$
3. population ← ∅
4. for $p = 1, p <= PN$ do
5.     chromosome ← array($n$)
6.     PM_rest ← PM_capacity // PM_rest represents the remaining resource of PM
7.     VM_allocated ← ∅ // VM_allocated is the set of virtual machines that has be allocated
8.     for $j = k, j = j - 1$, j>=2 do
9.     chromosome,PM_rest,VM_allocated ← allocate(chromosome, $G_j$, $N_j$, PM_rest, VM_allocated) //search the top $N_j$
                 group of $G_j$ to allocate VM
10.    end for
11.    VM_rest ← VM-VM_allocated  //find the remaining virtual machines that have not been allocated
12.    for $VM_i$ in VM_rest do
           PM_available ← {PM|PM_rest(PM)>=VM_request ($VM_i$)}
14.        for $PM_j$ in PM_available do
15.            calculate $D_{ij}$ between $VM_i$ and $PM_j$
16.        end for
17.        chromosome($VM_i$) ← the $PM_d$ with the minimum $D_{ij}$ // allocate $VM_i$ to the $PM_d$
           that has the minimum $D_{ij}$
18.        PM_rest($PM_d$) ← PM_rest($PM_d$) - VM_request($VM_i$) //update PM_rest
19.    end for
       population($p$) ← chromosome
20. end for
21. return population
```

communication between these two virtual machines can be approximated to zero. Therefore, the energy consumption of communication can be greatly reduced if virtual machines with a large amount of data to be transferred are placed on the same physical machine.

On the other hand, similar to the classical bin-packing problem, all virtual machines should be placed on as few physical machines as possible to enhance resource utilization and reduce energy consumption.

The details of EEPIS are shown in Algorithm 1 and Algorithm 2.

1) GROUPING AND SORTING OF VIRTUAL MACHINES BY THE AMOUNT OF COMMUNICATION DATA

First, to identify virtual machines with a large amount of communication data, virtual machines are divided into a series of groups, and groups with the same number of virtual machines are sorted (Algorithm 1, line 1).

Let G_j be the set of groups with j virtual machines, and $2 \leq j \leq k$. Then, for each G_j, virtual machine groups are sorted in descending order by the total amount of communication data among virtual machines in each group. Where, for the group in G_j with $j \geq 3$, the total amount of communication data is the sum of the communication data volumes between two virtual machines.

In addition, the value of k can be determined based on the actual scenario of the placement problem, such as resource capacity of physical machines and resource requirements of virtual machines. If the value of k is too large, the number of groups in set G_k that satisfy the resource capacity constraints of physical machines might be very small, which not only contributes little to reducing energy consumption but also increases the calculation time and reduces optimization efficiency. For example, k may approximate the average number of virtual machines that a physical machine can accommodate.

2) SELECTION OF TOP N_j VIRTUAL MACHINE GROUPS TO GENERATE INITIAL PLACEMENT SCHEME FOR EACH G_j

Only some groups with a large communication data volume in each G_j are selected to generate an initial virtual machine placement scheme (Algorithm 1, line 2). In other words, only the top N_j groups of G_j are selected. There are two main reasons, as follows.

On the one hand, virtual machine groups in each G_j are sorted in descending order by communication amount. The total communication data volume of the virtual machine group with lower rank in G_j is less and might not be greater than a higher-ranking group of the set G_{j-1}. Therefore, the virtual machine group with low ranking might make little effort to reduce energy consumption.

Algorithm 2 Allocate*(chrom, G, N, PM_rest, VM_allocated)*

```
1. for $q = 1, q <= N$ do //search the top $N$ group of $G$ one by one
2.     random select a suitable physical machine $PM_r$ which satisfies $\sum VM\_request\ (G(q)) <= PM\_\ rest(PM_r)$
3.     $chrom\ (G(q)) \leftarrow PM_r$ //Allocate the virtual machines in $G(q)$ to$PM_r$
4.     $PM\_rest(PM_r) \leftarrow PM\_rest(PM_r) - \sum VM\_request\ (G(q))$ //update $PM\_rest$
5.     $VM\_allocated \leftarrow VM\_allocated \cup G\ (q)$
6.end for
7.return $chrom$, $PM\_rest$, $VM\_allocated$
```

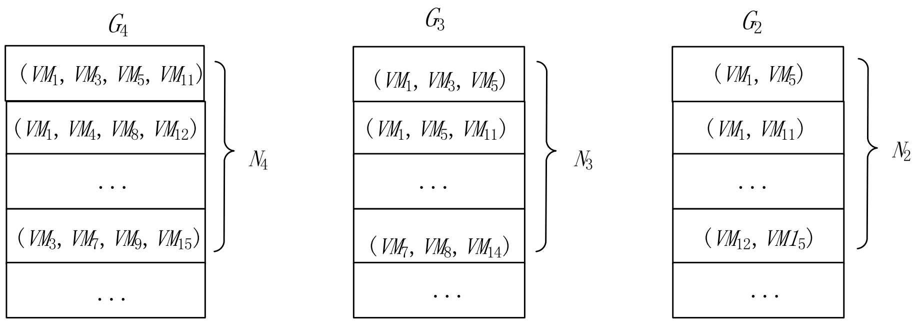


FIGURE 5. Schematic diagram of the grouping results.

On the other hand, the number of groups in all of G_j might be too large when the number of virtual machines to be placed is large, possibly leading to more optimization time and lower computational efficiency to search all groups.

The value of N_j, which represents the number of groups selected to generate the initial virtual machine placement scheme in G_j, should be determined by actual problem scenarios and real-time requirements to balance the effect and efficiency.

After the two above steps are executed, the results can be shown as Fig. 5.

3) INITIAL PLACEMENT OF THE VIRTUAL MACHINE GROUP

As mentioned above, virtual machine groups with a large amount of communication data should be placed on the same physical machine as much as possible to enhance the optimization effect and even accelerate convergence. Therefore, initial placement of the virtual machine groups is conducted to initialize each placement scheme, as shown in line 8-10 of Algorithm 1.

In general, the greater the number of virtual machines in a group, the greater the amount of communication data. Therefore, groups with larger numbers of virtual machines are preferentially chosen to be placed. Further, for G_j, each group of the top N_j groups is placed on a randomly selected physical machine in order, and the selected physical machine should satisfy resource requirements of virtual machines in the group.

During this process, the assigned virtual machines are recorded. When an assigned virtual machine is included in a virtual machine group waiting to be placed, this group is not assigned. For example, in Fig. 5, after the first group in G_4 is assigned to a physical machine, the second group is not assigned because the virtual machine VM_1 has been allocated.

4) INITIAL PLACEMENT OF REMAINING VIRTUAL MACHINE

After initial placement of the virtual machine group, there may still be a small number of virtual machines that are not assigned to the appropriate physical machines. Therefore, the initial placement of remaining virtual machines is conducted, as shown in line 11-19 of Algorithm 1.

At this time, free spaces of three types of resource in most physical machines are different and limited. Randomly assigning these remaining virtual machines to physical machines can lead to a bad optimization effect. Therefore, the greedy strategy is introduced to the initial placement of a remaining virtual machine.

Let D_{ij} denote the distance between the resource demand of each virtual machine VM_i to be allocated and the resource remaining capacity of each available physical machine PM_j that can accommodate this virtual machine. D_{ij} can be calculated as (24).

$$\begin{aligned} D_{ij} = sqrt((&PM_j^{rest\_cpu} - E(VM_i^{cpu}))^2 \\ &+ (PM_j^{rest\_mem} - E(VM_i^{mem}))^2 \\ &+ (PM_j^{rest\_band} - E(VM_i^{band}))^2) \end{aligned} \tag{24}$$

where $PM_j^{rest\_cpu}$, $PM_j^{rest\_mem}$, $PM_j^{rest\_band}$ represent the remaining amount of resources in PM_j. Then, physical machine PM_j corresponding to the minimum D_{ij} is selected to place VM_i to ensure the efficient use of the physical machines' resources as much as possible.

V. EXPERIMENTAL DESIGN AND RESULT ANALYSIS

A. DESIGN OF EXPERIMENTS

Experiments are designed to verify the rationality of the proposed model and the effectiveness of the improved algorithm in this paper. There are three goals for experiments, as follows.

(1) Comparison of EEKnEA with KnEA and NSGA-III on the quality of the optimal solution set.

(2) Comparison of EEKnEA and other virtual machine placement algorithms on different objectives.

(3) Evaluation of the robustness performance of the placement scheme obtained by the EEKnEA algorithm.

All experiments are performed on a computer with Intel Core i5 3.4 GHz PC and 4 GB memory. The software platform is MATLAB R2010a.

The following scenario is used in all experiments. Forty physical machines are provided for the placement of 50 virtual machines. Generally, the cloud data centre provides various physical machines to satisfy different requirements. Therefore, configurations of physical machines in the experiment are different, and their resource capacities are set randomly. At the same time, to reflect the diversity of resource requests by virtual machines, the expectation and standard deviation of resource requests for virtual machines are also set randomly.

The values of energy consumption when the physical machine is idle and fully utilized are 175w and 250w. The parameters about the communication network are set as follows: $e_1 = 0$, $e_2 = 1$, $e_3 = 3$, and $e_4 = 5$. The amount of data to be transferred between virtual machines is randomly generated, and the value is an integer between 0 and 10 (units). For the experimental scenario designed in this paper, a physical machine can on average accommodate 4 or 5 virtual machines; therefore, the value of k is set to 4.

The parameters of EEKnEA are described in Table 1 and are chosen based on pilot experiments.

TABLE 1. Parameters setup for the EEKnEA.

Parameters	Population size	Iterations	Crossover probability	Mutation probability
Value	200	2000	0.8	0.1

B. EXPERIMENTAL RESULTS ANALYSIS

1) COMPARISON OF EEKnEA WITH KnEA AND NSGA-III ON QUALITY OF THE OPTIMAL SOLUTION SET

To compare the quality of the optimal solution sets obtained by EEKnEA, KnEA and NSGA-III, HV (HyperVolume) is used as the evaluation indicator. HV can evaluate the convergence, uniformity and extensibility of the solution set and provides a comprehensive evaluation of the solution set [21]. HV calculates the super volume of the solution set to the negative ideal point. Therefore, the larger the HV, the farther the point in the solution set away from the negative ideal point and the better the quality of the solution.

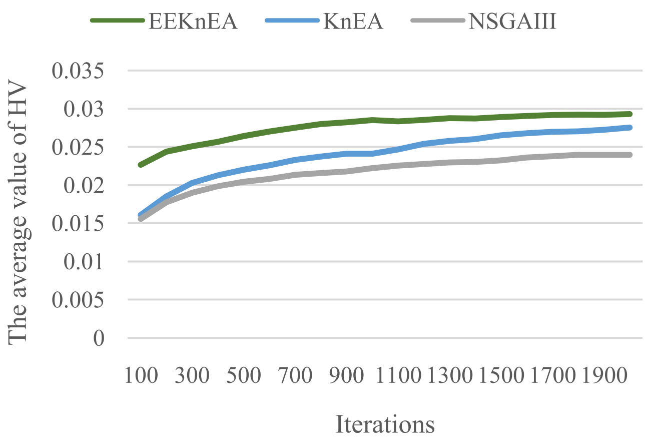


FIGURE 6. Changing trends of average value of HV with the increase of iterations.

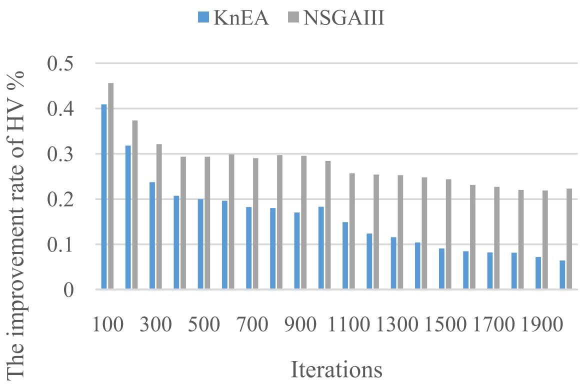


FIGURE 7. Improvement rate of average HV value of EEKnEA compared with KnEA and NSGA-III.

These three algorithms are performed 30 times for the scenario presented in part A of section V, respectively. The basic parameters of KnEA and NSGA-III are same as EEKnEA, such as iterations, mutation and crossover probability. The average values of HV in every 100 iterations using these three algorithms are shown in TABLE 2 and Fig. 6.

Further, the independent samples t test is performed to verify whether there is a statistically significant difference in the average values of these three algorithms. The test results show that there is a significant difference in the average values of HV between EEKnEA and two algorithms per 100 iterations. In addition, KnEA has a significant difference with NSGA-III in the average values of HV. For example, the test results of the 2000th iteration are shown in Table 3.

The improvement rate of average HV value of EEKnEA compared with KnEA and NAGA-III is shown in Fig. 7.

Based on the above analysis, the following observations can be made.First, EEKnEA performs better than KnEA and NSGA-III in terms of HV. Second, compared with NSGA-III and KnEA, the improvement rate of the average HV value of EEKnEA generally decreases with the

TABLE 2. Average HV values in every 100 iterations using EEKnEA, KnEA And NSGA-III.

Iterations	100	200	300	400	500	600	700
EEKnEA	0.024385	0.02508	0.025672	0.026406	0.027018	0.027524	0.027987
KnEA	0.018501	0.020269	0.021262	0.022004	0.022581	0.023276	0.02371
NSGA-III	0.015559	0.017751	0.01898	0.019843	0.020413	0.020804	0.021328
Iterations	**800**	**900**	**1000**	**1100**	**1200**	**1300**	**1400**
EEKnEA	0.028211	0.028519	0.024385	0.028345	0.028533	0.028763	0.028722
KnEA	0.024102	0.024104	0.018501	0.024663	0.025382	0.025773	0.02601
NSGA-III	0.021573	0.02178	0.022207	0.022549	0.02275	0.022954	0.023014
Iterations	**1500**	**1600**	**1700**	**1800**	**1900**	**2000**	
EEKnEA	0.028904	0.029053	0.029174	0.029224	0.029194	0.029301	
KnEA	0.026493	0.026778	0.026957	0.027017	0.027227	0.027521	
NSGA-III	0.023236	0.023595	0.0237767	0.023947	0.023951	0.023963	

TABLE 3. Test results of EEKnEA with KnEA and Nsga-iii in terms of average value of HV in the 2000th iteration.

		Levene's Test for Equality of Variances		*t*-test for Equality of Means		
		F	Sig.	t	df	Sig. (2-tailed)
Between EEKnEA and KnEA	**Equal variances assumed**	3.049	.086	4.435	58	.000
	Equal variances not assumed			4.435	56.643	.000
Between EEKnEA and NSGA-III	**Equal variances assumed**	.722	.401	11.083	38	.000
	Equal variances not assumed			11.083	37.992	.000
Between KnEA and NSGA-III	**Equal variances assumed**	.009	.925	9.165	58	.000
	Equal variances not assumed			9.165	54.839	.000

increase in iterations. As shown in Fig. 7, compared with NSGA-III, the improvement rate of average HV value of EEKnEA reaches up to about 45% at the 100th iteration, and it gradually decreases but always greater than 21%. In addition, compared with KnEA, the improvement rate of average HV value of EEKnEA is approximately 40% at the 100th iteration, and the improvement rate remains as high as 6% after 2000 iterations, as EEKnEA adopts the Energy-Efficient-oriented Population Initialization Strategy, which can produce a high-quality initial population to improve the efficiency of optimization and ensure the quality of the optimization results. In particular, EEKnEA can obtain better optimization results when the number of iterations is small. Thus, EEKnEA is very suitable for obtaining a satisfactory solution in a short time for the virtual machine placement problem with a high real-time requirement.

2) COMPARISON OF DIFFERENT ALGORITHMS IN TERMS OF EACH OBJECTIVE

To comprehensively evaluate the advantage of EEKnEA in terms of energy saving, robustness, load Balance and resource utilization, EEKnEA is compared with some popular virtual machine placement algorithms, including FFD (First Fit Decreasing), BFD (Best Fit Decreasing), GA (Genetic Algorithm) and KnEA. Considered that EEKnEA is improved on the basis of KnEA, and KnEA is better than NSGA-III in the performance of HV, NSGA-III is not chosen to be compared. FFD and BFD are heuristic algorithms extensively used to solve the bin-packing problem. FFD operates by first sorting the virtual machines to be placed in decreasing order by their resource requests and then placing each virtual machine on the first physical machine in the list with sufficient remaining space. Similarly, BFD first ranks the virtual machines in descending order by their resource requests and then places each virtual machine on the physical machine with the least amount of free space that can still hold the current virtual machine. Because the energy consumption is closely related to CPU utilization, the virtual machines are sorted by CPU request for FFD and BFD. In addition, EEKnEA is also compared with the single objective GA. Four GA-based optimization experiments are conducted by taking each of four objectives as the optimization objective, called GA_energy, GA_robust, GA_load, and GA_utilization. And each GA-based optimization experiment adopts the same coding method, single point crossover and mutation operators as EEKnEA employs. The population size of each GA-based optimization experiment is 200, and the maximum iteration is 200. In every iteration, individuals with optimal fitness are selected to the next generations.

In this paper, each algorithm runs 30 times. For each objective, we calculated some statistic indexes of experimental results in order to comprehensively compare the performance of different algorithms. The smaller the value of each objective the better it is based on the model built in Section III. The average value of 30 experimental results of each algorithm is calculated and shown as Fig. 8. Table 4 lists out the maximum, minimum and variance of experimental results besides the mean value. In addition, the quantiles of experimental results of different algorithms can be clearly shown in Fig. 9.

The comparison of EEKnEA with other algorithms in terms of average energy consumption is shown in Fig 8(a). EEKnEA is clearly more energy-efficient than other algorithms. In detail, it reduces energy consumption by approximately 28%, 27%, 24% and 1% compared with FFD,

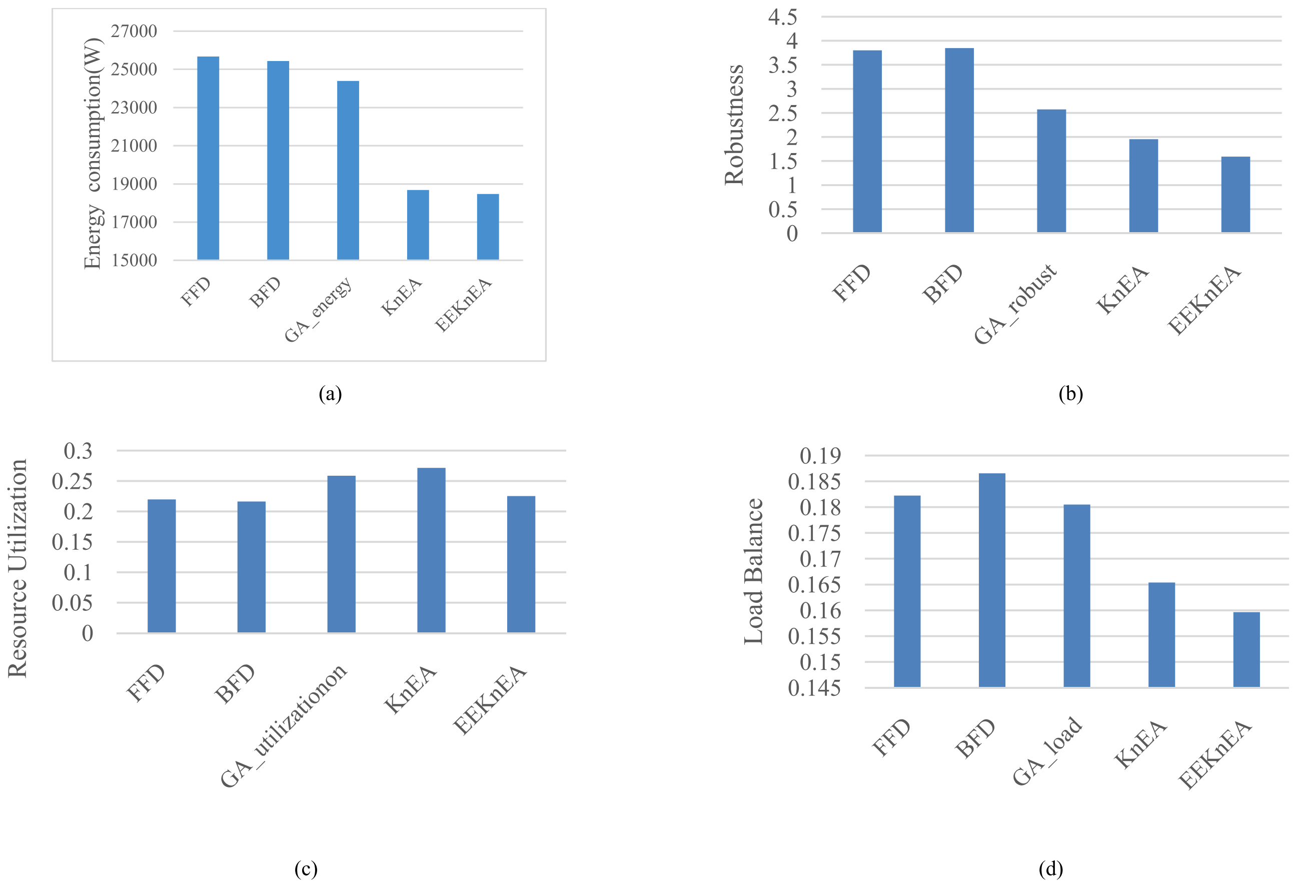


FIGURE 8. Comparison of different algorithms in terms of four objectives. (a) Energy consumption. (b) Robustness. (c) Resource utilization. (d) Load balance.

TABLE 4. Comparison of four objectives among different algorithms. (a) Energy consumption. (b) Robustness. (c) Resource utilization. (d) Load Balance.

(a)

Algorithm	FFD	BFD	GA_energy	KnEA	EEKnEA
mean	2.567028	2.543369	2.438862	1.868006	1.843737
maximum	2.611214	2.563323	2.456525	1.93392	1.90662
minimum	2.532545	2.519948	2.412643	1.82308	1.81911
variance	0.000215	0.000164	9.48E-05	0.00073	0.000283

(b)

Algorithm	FFD	BFD	GA_robust	KnEA	EEKnEA
mean	3.800193	3.847735	2.57443	1.95230	1.591029
maximum	4.012475	4.027348	3.207452	3.9847	2.84269
minimum	3.615778	3.583649	1.72997	0.063591	0.05345
variance	0.012376	0.014546	0.086833	1.134213	0.324181

(c)

Algorithm	FFD	BFD	GA_utilization	KnEA	EEKnEA
mean	0.21969	0.21626	0.258804	0.27147	0.22537
maximum	0.26739	0.26816	0.295450	0.32478	0.30716
minimum	0.17692	0.16744	0.22197	0.22178	0.11956
variance	0.01237	0.014546	0.086833	1.13421	0.324181

(d)

Algorithm	FFD	BFD	GA_load	KnEA	EEKnEA
mean	0.182254	0.186552	0.180512	0.165397	0.159634
maximum	0.22448	0.21248	0.20729	0.20022	0.18441
minimum	0.14996	0.15906	0.13763	0.14066	0.13098
variance	0.000318	0.000218	0.000234	0.000134	0.000193

BFD, GA_energy and KnEA, respectively, where the average energy consumption of the final solution set using EEKnEA is approximately 200 w/h less than that of KnEA. However, there are often thousands of physical machines running in a cloud data centre. When the number of physical machines and running time increase, this advantage becomes more obvious. For example, for a cloud data centre with 4000 physical machines running 24 hours every day for one year, the energy consumption using EEKnEA could save 175,200 KWh compared with using KnEA. In addition, it is obviously seen from TABLE 4 (a) that EEKnEA is better than other algorithms in terms of maximum and minimum. And EEKnEA's

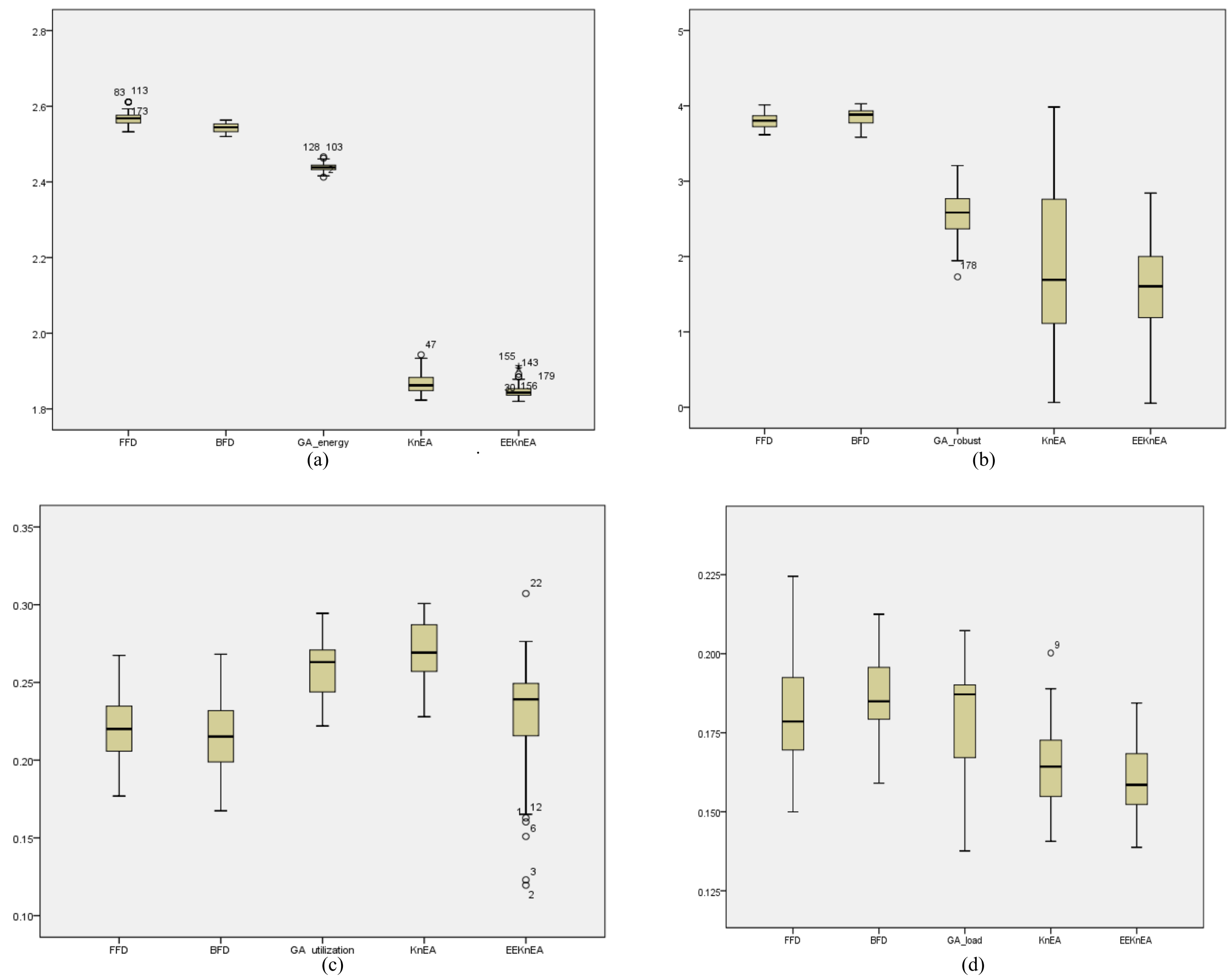


FIGURE 9. Boxplots of different algorithms of four objectives. (a) Energy consumption. (b) Robustness. (c) Resource utilization. (d) Load balance.

quantiles are also smaller than other algorithms as shown in Fig. 9 (a).

For the other three objectives, some conclusions can be obtained as follows. With regards to robustness, EEKnEA shows a very large advantage in mean, maximum, and minimum values, even if the variance of EEKnEA is larger than those of FFD, BFD and GA_robust. For resource utilization, although the mean value of EEKnEA is inferior to those of FFD and BFD, it is still superior to those of GA_utilization and KnEA, and achieves a best minimum; the goal of FFD and BFD is to use the least number of physical machines to place all virtual machines, which leads to high resource utilization. As EEKnEA needs to weigh four objectives at the same time, it does not perform as well as FFD and BFD in terms of resource utilization. For load balance, EEKnEA is superior to other algorithms in terms of mean, minimum, maximum and variance. And the median and quantile of EEKnEA are also better than other algorithms, which can be clearly seen from Fig. 9 (d).

In conclusion, the EEKnEA algorithm is significantly better than other algorithms in terms of energy consumption, robustness and load balance. Although EEKnEA is not the best performer in terms of resource utilization, it is possible to obtain solutions with better overall performance because it takes into account all the objectives.

C. THE EVALUATION OF THE ROBUSTNESS PERFORMANCE OF EEKnEA

For the same scenario described above, the comparative experiment is conducted to verify the robustness of the placement scheme. First, let SS_1 be the optimal solution set obtained by the proposed model and the EEKnEA algorithm. Second, the virtual machine placement model proposed in Section III is modified, and the robustness objective is removed from the model. Then, the same EEKnEA algorithm is used to resolve the modified model, and corresponding optimal solution set SS_2 is obtained. Third, the overload probabilities of SS_1 and SS_2 are calculated and compared.

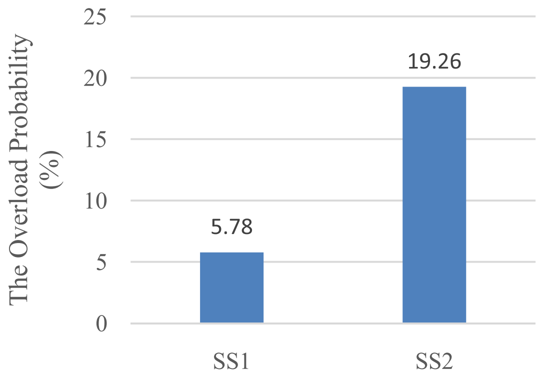


FIGURE 10. **The overload probability of SS_1 and SS_2.**

In order to compare the overload probability of SS_1 and SS_2, the simulation experiment is conducted. The load distribution of each virtual machine over a period of time is randomly generated according to its load expectation and standard deviation. Then, whether each physical machine is overloaded at each time can be acquired based on the load of virtual machines placed on the physical machine and the resource capacity of the physical machine. Next, the overload probability of a solution (also named placement scheme) at a given time is the ratio of the amount of physical machines that are overloaded at this time point to the total amount of physical machines used in this solution. Thus, the overload probability for each solution is expressed as the average of the overload probability of the solution at all times, and the overload probability of the solution set is expressed as the average of the overload probabilities of all solutions.

In this way, as shown in Fig. 10, the overload probability of SS_1 is 5.78%, and the overload probability of SS_2 is 19.26%. That is to say, when the robustness objective is not considered in the placement of the virtual machines, the probability of the physical machines being overloaded is more than three times than the placement considering robustness. Therefore, regardless of robustness, the number of virtual machines to be migrated due to overload can increase greatly, thereby increasing the migration costs and time. The reason is that the proposed model considers the reserved space of physical machines based on the dynamism and randomness of the virtual machines' load. In addition, 5.78% is the average overload probability of the solution set SS_1, and there are solutions with a smaller overload probability in the SS_1, which can be selected according to the actual demand.

VI. CONCLUSION

With the large amount of energy consumed by cloud data centres, how to assign virtual machines efficiently to available physical machines has become an essential research problem. In this paper, a many-objective virtual machine placement model considering energy consumption, resource utilization, load Balance and robustness is proposed at first. Then, the improved algorithm EEKnEA is proposed to solve this virtual machine placement problem. In EEKnEA, Energy-Efficient-oriented Population Initialization is proposed and used to obtain a higher quality initial population. Finally, the proposed model and algorithm are evaluated through experiments comparing KnEA and other algorithms. The results demonstrate that the proposed model and EEKnEA are competitive. On the one hand, EEKnEA performs better than other algorithms in reducing energy consumption and load balancing. On the other hand, the proposed model and EEKnEA can achieve a more robust placement scheme.

As we know, a good virtual machine placement scheme can reduce energy consumption and the possibility of physical machine overload. However, the abnormal load on physical machines, including both overload and underload, cannot be completely avoided. Therefore, in the future, when the load of the physical machine is abnormal, the dynamic migration optimization of the virtual machine will be studied to further reduce energy consumption and improve QoS. Besides, we would attempt to adopt some other crossover and mutation operators for obtaining the better performance of the algorithm. And the applicability of the proposed algorithm in cloud environment will be examined in the field of E-government and E-commerce in future.

REFERENCES

[1] J. M. Kizza, *Virtualization Security, Guide to Computer Network Security*. London, U.K.: Springer, 2015, pp. 473–490.

[2] M. Tang and S. Pan, "A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers," *Neural Process. Lett.*, vol. 41, no. 2, pp. 211–221, 2015, doi: 10.1007/s11063-014-9339-8.

[3] N. Akhter and M. Othman, "Energy aware resource allocation of cloud data center: Review and open issues," *Cluster Comput.*, vol. 9, no. 3, pp. 1163–1182, 2016.

[4] X. Zhang, Y. Tian, and Y. Jin, "A knee point-driven evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 6, pp. 761–776, Dec. 2014.

[5] J. Nie, "A research of virtual machine resource scheduling strategy based on cloud computing," *Commun. Comput. Inf. Sci.*, vol. 575, pp. 294–304, 2016.

[6] L. Zhu, R. Tang, Y. Tao, M. Ren, and L. Xue, "Multi-objective ant colony optimization algorithm based on load balance," in *Proc. Int. Conf. Cloud Comput. Secur.*, Nanjing, China, 2016, pp. 193–205.

[7] S. Filiposka, A. Mishev, and C. Juiz, "Balancing performances in online VM placement," *Adv. Intell. Syst. Comput.*, vol. 399, pp. 153–162, 2016.

[8] X. Fu and C. Zhou, "Virtual machine selection and placement for dynamic consolidation in cloud computing environment," *Frontiers Comput. Sci.*, vol. 9, no. 2, pp. 322–330, 2015.

[9] X. Yang, H. Zhang, H. Ma, W. Li, G. Fu, and Y. Tang, "Multi-resource allocation for virtual machine placement in video surveillance cloud," *Comput. Sci.*, vol. 9567, pp. 544–555, 2016.

[10] X. Li, Z. Qian, S. Lu, and J. Wu, "Energy efficient virtual machine placement algorithm with balanced and improved resource utilization in a data center," *Math. Comput. Model.*, vol. 58, nos. 5–6, pp. 1222–1235, 2013.

[11] Z. Zhou, Z.-G. Hu, and T. Song, "A novel virtual machine deployment algorithm with energy efficiency in cloud computing," *J. Central South Univ.*, vol. 22, no. 3, pp. 974–983, 2015.

[12] X. Zhang, Q. Yue, and Z. He, "Dynamic energy-efficient virtual machine placement optimization for virtualized clouds," in *Proc. Int. Conf. Elect. Inf. Technol. Rail Transp.*, 2014, pp. 439–448.

[13] M. Gabay and S. Zaourar, "Vector bin packing with heterogeneous bins: Application to the machine reassignment problem," *Ann. Oper. Res.*, vol. 242, no. 1, pp. 161–194, 2016.

[14] H. Zhong, K. Tao, and X. Zhang, "An approach to optimized resource scheduling algorithm for open-source cloud systems," in *Proc. Chinagrid Conf.*, Guangzhou, China, 2010, pp. 124–129.

[15] M. H. Ferdaus, M. Murshed, R. N. Calheiros, and R. Buyya, "Virtual machine consolidation in cloud data centers using ACO metaheuristic," in *Proc. Eur. Int. Conf. Parallel Process.*, Porto, Portugal, 2014, pp. 162a–167a.

[16] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future Generat. Comput. Syst.*, vol. 28, no. 5, pp. 755–768, 2012.

[17] X. Fan, W. Weber, and L. Barroso, "Power provisioning for a warehouse-sized computer," in *Proc. 34th Annu. Int. Symp. Comput. Archit.*, 2007, pp. 13–23.

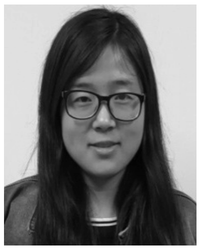
[18] L. Hu, H. Jin, and X. Liao, "Magnet: A novel scheduling policy for power reduction in cluster with virtual machines," in *Proc. IEEE Int. Conf. CLUSTER Comput.*, 2008, pp. 13–22.

[19] F. Ma, "Research on virtual machine placement and live migration in cloud data center," Ph.D. dissertation, Inst. Comput. Sci. Technol., Beijing Jiaotong Univ., Beijing, China, 2013.

[20] N. T. Hieu, M. Di Francesco, and A. Y. JÃdÃdški, "A virtual machine placement algorithm for balanced resource utilization in cloud data centers," in *Proc. IEEE 7th Int. Conf. Cloud Comput.*, Anchorage, AK, USA, Jun. 2014, pp. 474–481, doi: 10.1109/CLOUD.2014.70.

[21] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Trans. Evol. Comput.*, vol. 3, no. 4, pp. 257–271, Apr. 1999.

XIN YE received the Ph.D. degree from the Dalian University of Technology (DUT), Dalian, China, in 2006. He is currently an Associate Professor with DUT. His research interests include cloud computing, business process management, management information system, and software engineering.



YANLI YIN is currently pursuing the master's degree with the Dalian University of Technology. Her research interests is mainly cloud computing.

LAN LAN received the B.S. degree from Victoria University, Melbourne, Australia, in 2004, and the M.S. degree from University of Wollongong, New South Wales, Australia, in 2006. She is currently a Lecturer with Shenyang University, Shenyang, China. Her current research interest is cloud computing and system optimization.