Mobile Networks and Applications
https://doi.org/10.1007/s11036-019-01376-7

Network-aware Virtual Machine Migration Based on Gene Aggregation Genetic Algorithm

Yi Jiang[1] · Jinjin Wang[1,2] · Jieke Shi[1] · Junwu Zhu[1] · Ling Teng[1]

Abstract

As a key technology of cloud computing, virtualization technology enables multiple virtual machines (VMs) to run on a host to meet the operational needs and environmental requirements of different applications, improving the efficiency of the host. However, the resource of the hosts is limited. When the VMs runs too many tasks, the host will be overloaded and exception occurs. Regarding the issue above, this paper considers the communication cost of virtual machine (VM) migration and proposes a VM Migration Algorithm based on Gene Aggregation Genetic Algorithm (VMM-GAGA). VMM-GAGA mainly solves the problem of allocation between VMs which to be migrated and underutilized hosts. In VMM-GAGA, a novel genetic coding method based on gene aggregation algorithm is proposed. The algorithm performs gene aggregation operations on VMs that have more communication and meet the conditions, which effectively reduces the number of genes in the chromosome., Experiments show that compared with the traditional genetic algorithm, VMM-GAGA reduces search time and communication costs.

Keywords Virtualization · Virtual machine migration · Gene aggregation · Genetic algorithm

1 Introduction

With the further development of distributed computing and grid computing, cloud computing came into being [1, 2]. It can provide users with flexible and scalable, low-cost and personalized computing infrastructure requirements and services under the premise of guaranteeing the quality of service (QoS) [3–5]. Virtualization technology plays a very important role in cloud computing [6, 7]. Virtualization technology can virtualize a computer into multiple VMs with different operating systems and different application running environments. VMs are isolated from each other and do not affect each other. Therefore, the isolation and security of the application are guaranteed [8, 9].

In a data center, if a host runs too many VMs and needs to handle too many tasks, the host may be overloaded, which is likely to have a service level agreement (SLA) violation and exception occur [10]. In the data center, hosts with fewer VMs residing have low load and cannot fully utilize the hardware resources of the host, resulting in the waste of resources. Virtual machine migration can migrate VMs on overloaded hosts to low-load hosts, which prevents the host from being overloaded. In addition, VM migration can achieve load balancing and reduce energy costs and communication costs in the data center [11]. VM migration mainly involves energy consumption, migration costs, and communication costs. Communication between VMs cannot be ignored. However, most of the research on VM migration ignores the communication between VMs. Therefore, this paper mainly considers the communication cost of VM migration. VM migration can be seen as an optimization issue. Genetic algorithms and bee colony

This work was supported in part by the National Natural Science Foundation of China under grant nos. 61872313; the Key Research Projects in Education Informatization in Jiangsu Province under grant 20180012; by the Postgraduate Research and Practice Innovation Program of Jiangsu Province under grant KYCX18 2366; and by Yangzhou Science and Technology under grant YZ2018209, YZ2019133; and by Yangzhou University Jiangdu High-end Equipment Engineering Technology Research Institute Open Project under grant YDJD201707.

✉ Yi Jiang
jiangyi@yzu.edu.cn

Jinjin Wang
1245571129@qq.com

1 College of Information Engineering, Yangzhou University, Yangzhou, Jiangsu, China

2 School of Information Engineering, Yancheng Teachers College, Yancheng, China

Published online: 14 January 2020

algorithms solve many optimization problems [12, 13]. The existing genetic algorithm-based VM migration solves the network-aware VM migration problem [14], but there are still three problems. The first problem is that when the size of the VM is relatively large, the algorithm result is not very satisfactory. The second problem is that the constraints of VM migration are not considered in the process of executing the genetic algorithm, and the resulting solution may overload the target host after the VM is migrated; the third problem is that the offspring directly replace all parents in the process of genetic algorithm population replacement. That is to say, if the fitness of the offspring is less than the fitness of the parents, the parents will be directly eliminated, the resulting solution in this way may not be the optimal solution or the approximate optimal solution.

In this paper, three aspects are improved in VMM-GAGA: (1) Innovatively proposed the coding method of gene aggregation, which reduced the number of genes of chromosomes and improved the performance of genetic algorithms; (2) The constraint condition is added during the execution of the algorithm, thereby ensuring that the host does not overload after the migration; (3) "The survival of the fittest": When the population is replaced, individuals with high fitness are allowed to enter the next generation, while individuals with low fitness are directly eliminated.

VM migration in the data center is divided into two steps. The first step is to select the VMs which to be migrated on overloaded hosts. The second step is to find the mapping between the VMs which to be migrated and the underutilized hosts. For the first step of VM migration, this paper uses VMs-GSA (VMs Greedy Selection Algorithm) proposed in the literature [15] to select the VMs which to be migrated on the overloaded host. For the second step of VM migration, this paper proposes VMM-GAGA to achieve the mapping of the VMs which to be migrated and underutilized hosts.

The main contributions of this paper are summarized as follows: (1) Establish a communication model for the data center, taking full account of the communication costs in the data center; (2) Propose a coding method based on gene aggregation algorithm, which effectively reduces the number of chromosomes, and thus reduces the convergence time of the genetic algorithm; (3) Apply VMM-GAGA to the second process of VM migration to find mappings between VMs which to be migrated and the low-load host.

The rest of the paper is organized as follows. Section 2 is related work. Section 3 is the problem formalization and system model. Section 4 proposes VMM-GAGA to find the mappings between selected VMs and low-load hosts. Section 5 is the experimental results. Section 6 summarized the paper.

2 Related work

VM migration can reduce the cost of data center and improve the efficiency of the host. In recent years, VM migration has become a hot research topic. Literature [14] examines the issue of VM migration in the context of overused cloud data centers. It applies genetic algorithms and bee colony algorithms to VM migration. The goal is to find the optimal solution or approximate optimal solution through repeated iterations. Its encoding method is that uses a vector to represent the VM and host mapping, the length of the vector indicates the number of VMs and the position of the vector represents the host. In fact, this coding method is the same as in literature [17]. They are all general coding methods in VM migration based on genetic algorithm. First, it generates an initial population of Popsize, then crosses and mutates the initial population to produce new progeny. Next, select the individuals with large fitness in the offspring as the optimal individual. Finally, iterate the process until the stop condition is met, and the iteration of the population stops. Experiments show that when the number of VMs is small, the genetic algorithm will get better results. Because this results in lower communication costs. When the number of VMs is large, the results of the bee colony algorithm are better.

Beloglazov et al. believe that data center run applications consume a lot of power. The reason is that the data center not only bears huge operating costs but also increases carbon emissions and affects the environment [16]. The power consumption of computers in the data center is mainly CPU consumption, so they focus on the management and utilization of CPU power consumption. For the placement of new VMs, they proposed a modified best fit decrement algorithm (Modified Best Fit Decreasing, MBFD). In this algorithm, they sort all VMs according to their CPU utilization, and assign to hosts with the least increase in power consumption sequentially. There are two steps for optimizing VM allocation issues. The first step is the choice of VMs. First, sort VMs in descending order according to their CPU utilization and select some VMs until the host is not overloaded. The second step is to use MBFD algorithm to assign the selected VMs to the host. This literature mainly considers the energy consumption of the data center, proposes MBFD to perform virtual machine migration, and reduces the energy consumption of the data center. However, they did not consider the communication costs and migration costs associated with moving VMs with high CPU utilization.

Literature [17] comprehensively considers energy consumption, communication cost and migration cost, and proposes a VM migration algorithm called BGM-BLA (Binary

Graph Matching-based Bucket-code Learning Algorithm). This is a method that similar with genetic algorithms. First, it generates the bucket code, then crosses and mutates until the solution is good enough. The authors ignored the original placement of the VMs, which increased migration costs.

Literature [18] proposed an integrated algorithm that considers the load of the host. In the process of selecting VMs, the number of VMs and various other important factors are considered. Energy consumption of data center is minimized by shutting down underutilized hosts. The literature [19] applies the firefly algorithm to the VM migration problem and migrates the heavily loaded VMs to the low-load host. Literature [20] proposes three strategies for dynamically managing VMs. Literature [21, 22] studied a new VM migration method based on energy saving. Besides, literature [23–25] considered the VM migration problem from the aspect of energy consumption. They only consider the energy consumption of the data center, but ignore the communication cost. Most of the literature only considers the power cost of the data center, ignoring the communication costs between virtual machines. The method of select the VMs which to be migrated of these literatures mainly considers the CPU utilization factor of the virtual machine, and select VMs with a large CPU utilization to migrate.

In this paper, we considered the communication cost between VMs and established a communication model for the data center. VM migration is divided into two parts in this paper. Firstly, VMs-GSA is used to select the virtual machine to be migrated, which consider the CPU utilization and load size of the VM, and then VMM-GAGA is proposed to obtain the mapping between the VMs which to be migrated and the underutilized hosts.

3 Problem formalization and system model

3.1 Problem formalization

Suppose there are K hosts or servers in the data center. The undirected graph $DC = (H, D)$ represents the communication relationship between hosts in the data center. $H = \{h_1, h_2, \ldots, h_k\}$ denotes a set of hosts. $\forall \left(h_i, h_j\right) \in D$ denotes the edge between host h_i and h_j. The weight $W(h_i, h_j)$ between h_i and h_j indicates the communication distance between h_i and h_j. We use the symbol $\tau\ (h_i)$ to represent the load threshold of the host h_i. If the load of host exceeds its threshold $\tau\ (h_i)$, then this host h_i is overloaded. We use $sl(h_i)$ to represent the security level value of h_i. If the load of h_i is less than $sl(h_i)$, h_i is underloaded or underutilized. We use the symbol H^- to represent the set of low-load hosts. The overloaded host and the underutilized host are denoted by the symbols h_j^+ and h_j^-, respectively. o_j indicates the size of the resource that h_j^- can provide, that is, the amount of free resource. o_j is obtained by the threshold of h_j^- minus the load of h_j^-.

The set of VMs in the data center is represented by $V = \{V_1, V_2, \ldots, V_O\}$. The communication relationship of VMs in the data center is represented by a graph, $AG = (V, E)$, where $E = \left\{(V_i, V_j)|V_i, V_j \in V\right\}$ (if there exists communication between V_i and V_j) represents a set of edges. The weight $GW(V_i, V_j)$ of the edge (V_i, V_j) represents the amount of communication between V_i and V_j. $O(V_i)$ represents the occupied resources of V_i, such as the CPU utilization or the occupied memory of V_i. The VMs-GSA is used to select the set of VMs which to be migrated on the overloaded hosts. The set of VMs which to be migrated is represented by $VM = \{vm_1, vm_2, \ldots, vm_N\}$, where $VM \in V$.

Definition 1 Any VM in the data center that communicates with the VM to be migrated vm_i is called the neighbor of vm_i. The set of neighbors of vm_i is represented by $GN\ (vm_i) = V_l|(vm_i, V_l) \in E, \forall V_l \in V, vm_i \in VM$.

The source host of vm_i is represented by $r(vm_i)$, which is the host where the VM was migrated before. $M = \left|H^-\right|$ indicates the number of underutilized hosts in the data center. N indicates the number of VMs which to be migrated. The matching result between the VMs which to be migrated and the underutilized hosts is represented by an $N \times M$ matrix $X = (x_{ij}|i = 1, 2, \ldots, N;\ j = 1, 2, \ldots, M)$, where x_{ij} is shown in Eq. 1.

$$x_{ij} = \begin{cases} 1 & vm_i \text{ matches } h_j^- \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$x_{ij} = 1$ means that the requirement of vm_i can be satisfied by h_j^-. The number of VMs that allocated to h_j^- is $\sum_{i=1}^{N} x_{ij}$.

The purpose of this paper is to reduce the communication costs incurred during VM migration. With the reduction of communication cost, energy consumption will be reduced correspondingly. The mapping function $\sigma : VM \rightarrow H^-$ represents the mapping between VMs which to be migrated and the underutilized hosts, that is, $\sigma\ (i) = j, x_{ij} = 1$. The destination host of vm_i is h_j^-.

3.2 System model

Figure 1 is an abstract relationship diagram of VMs in the data center. The rectangle represents the host in the data center. There are two types of circles in a rectangle. The filled circle represents the VM to be migrated selected by VMS-GSA, and the open circle indicates that the VM is not selected for migration. Therefore, both h_1 and h_3

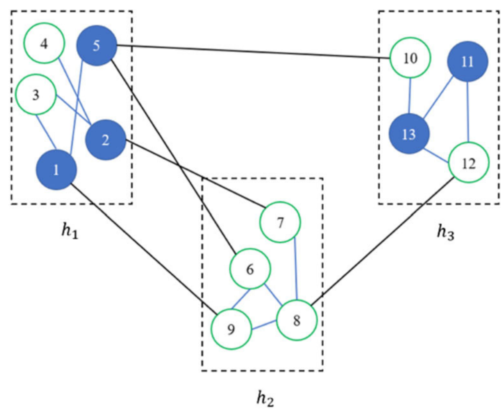


Fig. 1 Relationship of VMs in the data center

are overloaded hosts. h_2 is an underutilized host. The edges between VMs indicate that there is communication between them and that they are neighbors. For example, the neighbors of VM 5 is VM 1, 6, 10. Therefore, the communication cost of the VM 5 is related to the communication and distance of its neighbors 1, 6, 10.

Definition 2 The communication cost of vm_i matching the underutilized host h_j^- is defined as below:

$$Gcost_i = \sum_{V_l \in GN(vm_i)} GW(vm_i, V_l) W(h_{\sigma(i)}^-, h_{\sigma(l)}^-) \tag{2}$$

If $\sigma(i) = \sigma(l)$, it means that the VM vm_i and its neighbor vm_l are migrated to the same underutilized host, $W(h_{\sigma(i)}^-, h_{\sigma(l)}^-) = 0$. If the neighbor of the VM to be migrated is not the VM to be migrated, the destination host of this neighbor is the source host.

The network-aware VM migration problem based on gene aggregation genetic algorithm is defined as an optimization problem as shown in formula (3):

$$(S^*, X^*) = \arg\min_{S,X} \sum_{vm_i \in VM} Gcost_i \tag{3}$$

$$\text{s.t.} \quad \sum_{j=1}^{M} x_{ij} \le 1, \forall 1 \le i \le N \tag{4}$$

$$\sum_{i=1}^{N} x_{ij} O(vm_i) \le o_j, \forall 1 \le j \le M \tag{5}$$

The aim is to find a mapping between the virtual machine to be migrated and the host to be loaded that makes communication cost less. In the allocation process of the VMs which to be migrated, two constraints of formula (4) and formula (5) are satisfied: (1) vm_i can only match at most one underutilized host. (2) The sum of the occupied resources of the VMs matched by h_j^- cannot be greater than the idle resources of h_j^-.

The network-aware VM migration problem is a variant of the multiple pack problem [14], which is a NP-complete problem. Therefore, the goal of network-aware VM migration is to find an approximate optimal solution.

4 VMM-GAGA design

In this section, we consider applying genetic algorithms to VM migration issues. The purpose is to generate a mapping relationship between the VMs which to be migrated and the underutilized hosts that minimizes communication cost. This paper proposes VMM-GAGA to obtain the mapping relationship.

Figure 2 is the VMM-GAGA execution flow chart. Firstly, we designed the coding method and the fitness function for the Virtual Machine Migration Problem Based on Genetic Algorithm (VMMP-GA) problem. Then, use the roulette selection method to select some individuals as parents to breed the next generation. The two-point crossover and bit flipping variants used in this paper. The population evolves and converges to output the best

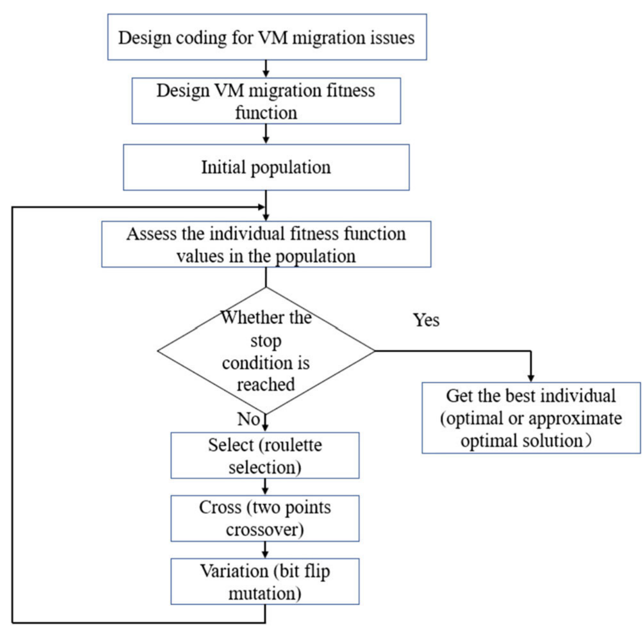


Fig. 2 VM migration problem solving process based on VMM-GAGA

individual, the most suitable, optimal or near-optimal solution. It should be noted that two constraints are enforced when the algorithm is executed. The gene aggregation coding method directly satisfies the constraint condition (1). Therefore, VMM-GAGA only needs to satisfy the constraint condition (2).

4.1 Gene aggregation genetic algorithm design

4.1.1 Encoding based on gene aggregation algorithm

The coding based on the gene aggregation algorithm establishes the mapping relationship between the VMs which to be migrated and the underutilized hosts essentially. The general coding method of VM migration is that the gene of the chromosome represents the destination host of the vm_i, and the location of gene in the chromosome string represents vm_i. The number of VMs which to be migrated is the length of the chromosome.

If the number of VMs which to be migrated is large, the length of chromosome will be very large, which will increase the search space of the genetic algorithm. time This will lead to an increase in the search time of the genetic algorithm and easy to fall into local optimum. When the number of VMs is relatively large, the communication cost of the solution obtained by the genetic algorithm is not very satisfactory in the VM migration based on genetic algorithm in literature [14]. In addition, the extended time of the genetic algorithm search will increase the time of VM migration, resulting in an abnormality in the data center and an increase in the probability of violating the service level agreement, which is not conducive to the operation of the data center application and the processing of the data. Therefore, this section has made improvements to the above problems, and proposed the concept of gene aggregation and applied it to genetic algorithm.

Definition 3 Gene aggregation. If the two VMs which to be migrated have smaller occupied resources and more communication, and the sum of their occupied resources is smaller than the value of the idle resources of any one of underutilized hosts, the two VMs can form a group, as a gene of the chromosome. they will migrate to the same underutilized host at the same time.

The core idea of gene aggregation is that two VMs which to be migrated with more communications can be migrated to the same underutilized host. it can not only reduce the communication cost of the data center, but also reduce the number of chromosomal genes and decrease the search space of the genetic algorithm. Figure 3 is an example of gene aggregation algorithm result. The number of genes is reduced, and the search space is reduced.

The conditions that two VMs which to be migrated vm_a and vm_b perform gene aggregation operation are as follows:

- $O(vm_a) + O(vm_b) < o_{j^*}, \forall vm_a \in VM, \forall vm_b \in (VM \setminus vm_a), \forall h_j^- \in H^-$.
- $vm_b \in GN(vm_a)$.

o_{j^*} indicates the minimum amount of idle resources of the underutilized hosts. The communication relationship of the VMs which to be migrated selected by the algorithm VM-GSA from the overloaded hosts is represented by a map $MG = (VM, ME)$, where VM represents a set of vertices, that is, the VMs which to be migrated. $ME = \left\{(vm_i, vm_j) | vm_i, vm_j \in VM\right\}$ (if there is communication between vm_i and vm_j indicates the set of edges. The weight $GW(vm_i, vm_j)$ of the edge (vm_i, vm_j) represents the traffic between vm_i and vm_j). The set of vm_i' neighbor is represent by $NE(vm_i) = \{(vm_i, vm_l) | \forall vm_i \in VM, (vm_i, vm_l) \in ME\}$.

Algorithm 1 is the execution flow of the gene aggregation algorithm. The input of the algorithm is $MG = (VM, ME)$, and the output is the result of gene aggregation. The algorithm is described in detail below.

Algorithm 1 Gene aggregation algorithm.

Input:
$MG = (VM, ME)$: Diagram of all VMs which to be migrated;
$H^- = \{h_1^-, h_2^-, \ldots, h_M^-\}$: Set of underutilized hosts.
Output:
$result$: Gene aggregation result set.

Find h_s^- with the smallest amount of idle resources in H^-
$result \leftarrow \emptyset$
while $ME \neq \emptyset$ **do**
  Find (vm_i, vm_j) with the largest weight $GW(vm_i, vm_j)$ in the ME
  if $O\left(vm_i\right) + O\left(vm_j\right) < o_s$ **then**
    $result \cup \left\{\left\{vm_i, vm_j\right\}\right\}$
    $ME \setminus \left(NE\left(vm_i\right) \cup NE\left(vm_j\right)\right)$
  else
    $ME \setminus \left\{(vm_i, vm_j)\right\}$
  end if
end while
return $result$

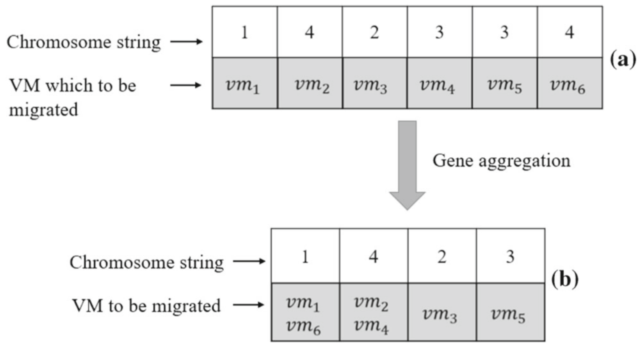


Fig. 3 Example of gene aggregation algorithm result

First, find the host h_s^- with the smallest free resource in the set H^- (line 1). The result of the gene aggregation is represented by $result$ (line 2). If $ME \neq \emptyset$, find the edge (vm_i, vm_j) with the largest weight in the ME. If the sum of the occupied resources of the vertices vm_i and vm_j of the edge (vm_i, vm_j) is less than o_s, the two vertices can be aggregated into one gene of the chromosome and added to result as a set (lines 3-6). Then, remove all edges of the connection of the vertex vm_i and vm_j from the set ME (line 7). If the occupied resources of vm_i and vm_j are not less than o_s, only the edge (vm_i, vm_j) is deleted from the set ME. The algorithm loops until the ME becomes an empty set and output $result$. The gene aggregation algorithm ends.

Figure 4 is an example of the gene aggregation algorithm. The communication relationship diagram of the VMs which to be migrated is shown in Fig. 4a. The vertices represent the VMs which to be migrated, and the weights of the

Table 1 Resource requirements of the VMs which to be migrated

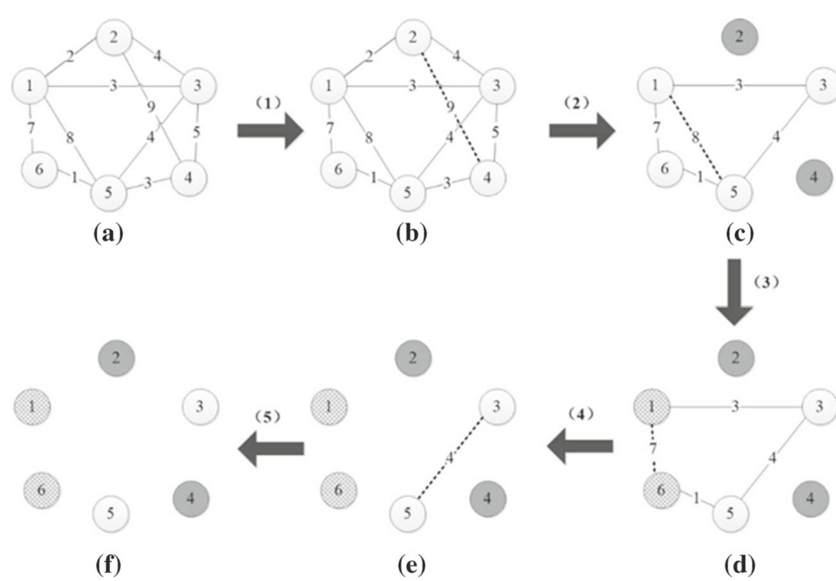
vm_i	$O(vm_i)$
1	4
2	4
3	6
4	3
5	5
6	2

edges represent the traffic of these VMs. Table 1 shows the resource requirements of the VMs which to be migrated.

Assume that the minimum free resources amount of underutilized hosts in the data center is 8. The weight of the edge (2, 4) is 9, which is the largest of the weights of all edges, as shown in Fig. 4b. The sum of the resource occupations of the VM 2 and the VM 4 is 7<8, which satisfies the condition. These two vertices 2 and 4 can be aggregated into one gene, and $result = \{\{2, 4\}, \{1, 6\}\}$. Then delete all edges that are connected to vertex 2 and vertex 4, as shown in Fig. 4c.

Next, the algorithm continues to find the edge with the largest weight in Fig. 4c, and that is (1, 5). The sum of the resource occupations of the VM 1 and the VM 5 is 9>8. Vertex 1 and 5 do not satisfy the condition. Therefore, the algorithm only deletes the edge (1, 5) and the result is shown in Fig. 4d. The algorithm continues to find the edge with the largest weight and the result is (1,6). The sum of the resource occupations of the VM 1 and the VM 6 is 6¡8.

Fig. 4 An example of the gene aggregation algorithm



Therefore, the two vertices 1 and 6 can be aggregated into one gene, and $result = \{\{2, 4\}, \{1, 6\}\}$. Then remove all edges that are connected to vertex 1 and vertex 6, as shown in Fig. 4e.

There is only one edge (3, 5) in Fig. 4e, and the sum of the resource occupations of the VM 3 and the VM 5 is 11>8. So, vertices 3 and 5 do not satisfy the condition. Algorithm only removes edges (3, 5) the result is shown in Fig. 4f. If $ME = \emptyset$, the algorithm stops. The result of the gene aggregation is $result = \{\{2, 4\}, \{1, 6\}\}$. The number of genes in the chromosome has changed from 6 to 4. As shown in Fig. 3.

4.1.2 Fitness function

The fitness objective function refers to the communication cost function of the mapping between the VMs which to be migrated and the underutilized hosts in VMMP-GA. In this paper, The smaller the communication cost function, the smaller the value of individual fitness objective function and the higher the individual fitness.

Therefore, the fitness function of the individual X_j of VMM-GAGA is defined as follows.

$$F(X_j) = \sum_{vm_i \in VM} Gcost_i \tag{6}$$

$$\text{s.t.} \quad \sum_{j=1}^{M} x_{ij} \le 1, \forall 1 \le i \le N \tag{7}$$

$$\sum_{i=1}^{N} x_{ij} O\left(vm_i\right) \le o_j, \forall 1 \le j \le M \tag{8}$$

The individual X_j, that is, the jth individual in the population, represents the matching result X between the VMs which to be migrated and the underutilized hosts. $Gcost_i$ represents the communication cost function that vm_i matches its destination host.

4.1.3 Population initialization

Population is a set of individuals. In VMM-GAGA, Population initialization is the process of generating individual processes by the gene aggregation algorithm coding. Algorithm 2 represents the process of VMM-GAGA population initialization.

The algorithm has two input, which are $VM = \{vm_1, vm_2, \ldots, vm_N\}$ and $H^- = \left\{h_1^-, h_2^-, \ldots, h_M^-\right\}$, respectively. The output is the initial population set GP.

Algorithm 2 VMM-GAGA population initialization algorithm.

Input:
$VM = \{vm_1, vm_2, \ldots, vm_N\}$: All VMs to be migrated selected by the algorithm VM-GSA from the overloaded host;
$H^- = \left\{h_1^-, h_2^-, \ldots, h_M^-\right\}$: Set of underutilized hosts.
Output:
GP: Initial population.

1: $GP \leftarrow \emptyset, i = 1$
2: **while** $|GP| < P_{size}$ **do**
3: Encoding VMs and underutilized host randomly, generate X_i
4: **if** X_i satisfy constraint condition **then**
5: $GP \cup \{X_i\}$
6: $i++$
7: **end if**
8: **end while**
9: **return** GP

The symbol P_{size} represents the number of individuals in the population. First, the VMs which to be migrated and the underutilized hosts are randomly matched and encoded to generate individual X_i (lines 1-3). If X_i is not satisfies the constraint of formula (4), continue looping to produce another individual. The algorithm loops until the number of initialized populations reaches P_{size} and all individual in initialized populations meet the two constraints (lines 4-8). Finally, the initialized population GP is obtained (line 9).

4.1.4 Population initialization

Traditional genetic algorithm operations include selection, crossover, and variation. The genetic operators used in VMM-GAGA are described below.

Select Operator - Roulette selection method. In VMM-GAGA, this paper uses the roulette selection method to select parents in the population. In the roulette selection method, all individuals have opportunity to be selected, and individuals with high value of individual fitness objective function are more likely to be selected. Therefore, the roulette selection method can ensure the diversity of the population.

Crossover operator - two-point crossover. The advantage of two-point crossover is that it can search the problem space more thoroughly without reducing the performance of the genetic algorithm.

Mutation operator - bit flip mutation. The number of genes is represented by the symbol Num. The bit flipping mutation randomly generates a position point a, $1 \leq a \leq Num$. then, the gene number at this position is randomly changed to another number.

The crossover and mutation operation in VMM-GAGA is performed with a certain probability P_{cross} and $P_{mutation}$, respectively. If the individuals generated by genetic operator does not satisfy the constraint, the parent will re-operate until the new individual satisfy the constraint or the operation reaches m times.

4.1.5 Population replacement

In VMM-GAGA, the population replacement method is "survival of the fittest". that is, after the parents generate offspring, there will be four candidate which including two parents and two offspring. In the candidate set, two individuals with smaller fitness objective function values are selected to join the new population. Two individuals with large fitness objective function values are eliminated.

4.2 Allocation algorithm

In this section, we consider using VMM-GAGA to obtain mappings between VMs which to be migrated and underutilized hosts. The VMM-GAGA allocation algorithm refers to the process of obtaining the optimal individual through the selection, crossover, mutation and update iteration of the initial population, that is, the search process of the optimal solution or approximate optimal solution of the genetic algorithm in the solution space.

Algorithm 3 is the process of allocation algorithm in VMM-GAGA. The input of the algorithm is the initial population set GP which obtained by Algorithm 2. The output is the optimal individual X_{best}.

Firstly, the algorithm selects the individual X_{best} with the smallest fitness function value in the initial population (line 1). The symbol T indicates the total number of algorithm iterations. (lines 2-3).

The first step in the iterative process of genetic algorithms is to produce the next generation. The set of the next generation is represented by offspring. Let the next generation set offspring be an empty set (line 4). The algorithm selects two parents in the population GP using the roulette algorithm. Assume that the currently selected parents are X_a and X_b (line 7), respectively. Then, perform the two-point crossover on X_a and X_b to produce offspring X_c and X_d (line 8). If X_c and X_d do not satisfy the constraint, then the parents X_a and X_b re-cross until the next generation meets the constraint or the operation reaches m times. (lines 9-12).

Algorithm 3 VMs allocation algorithm based on gene aggregation genetic algorithm.

Input:
GP: The set of initial population;
Output:
X: Mappings obtained by VMM-GAGA allocation algorithm.

```
1:  Find the individual X_best with the lowest fitness function value in GP
2:  j = 0, X ← X_best
3:  while j < T do
4:      offspring ← ∅
5:      while |offspring| < P_size do
6:          k_1 = 1, k_2 = 1, k_3 = 1
7:          Two parents are selected by the roulette algorithm in GP, X_a and X_b
8:          (X_c, X_d) ← Crossover(X_a, X_b)
9:          while Sol_c or Sol_d are not satisfy constraint condition && k_1 < m do
10:             (X_c, X_d) ← Crossover(X_a, X_b)
11:             K_1 + +
12:         end while
13:         X_e ← Mutation(X_c)
14:         while Sol_e is not satisfy constraint condition && k_2 < m do
15:             X_e ← Mutation(X_c)
16:             k_2 + +
17:         end while
18:         X_f ← Mutation(X_d)
19:         while Sol_f is not satisfy constraint condition && k_2 < m do
20:             X_f ← Mutation(X_d)
21:             k_3 + +
22:         end while
23:         offspring ← Select two individuals with smaller fitness function value in (X_a, X_b, X_e, X_f)
24:     end while
25:     GP ← offspring, Population replacement
26:     Find X_offsring−best with the lowest fitness function value in GP
27:     if F(X_offsring−best) < F(X) then
28:         X ← X_offsring−best
29:     end if
30: end while
```

Next, bit flip mutation operation is performed on X_c and X_d to generate new individuals X_e and X_f (line 13 and line 18). If X_e and X_f do not satisfy the constraint, the X_e and X_f are performed the bit-flip mutation method until the generated individual satisfies the constraint or the mutation operation reaches m times (lines 14-17, 18-22). Then, two individuals with smaller fitness function values are selected in the parent and next generation individual sets (X_a, X_b, X_e, X_f) and placed in the next generation generation set offspring (line 23). The algorithm loops until the number of next generations is equal to P_{size}.

After the emergence of the next generation, the next step is the replacement of the population, which is the "survival of the fittest" of the population. The algorithm directly replaces the original population set GP with the next generation set offspring (line 25). Next, in the updated population, find the individual $X_(offsring - best)$ with the smallest fitness function value, and then compare $X_(offsring - best)$ and X. If $X_(offsring - best)$ is less than the fitness function value of X, then the iterative optimal individual $X_(offsring - best)$ replaces X (lines 26-29).

At this point, this iteration cycle is completed. The algorithm is iterated until the number of iterations reaches T. Finally, the optimal individual X is output, and the entire VMM-GAGA allocation algorithm ends.

Compared with the literature [14], this paper proposes an innovative gene aggregation coding method that reduces the search space and search time of the genetic algorithm; the constraint condition is added in the execution of the genetic algorithm, thus ensuring that the final result can satisfy the VM migration conditions; improved the "survival of the fittest" when the population is replaced, so that the fitness of individuals entering the next generation is improved.

The time complexity of Algorithm 3 finding the best individual in the initial population GP on line 1 is $O(P_{size})$, where P_{size} denotes the number of population. The time complexity of the lines 5-24 generation process is $O(mP_{size})$. Since the total number of iterations of the population is T, the time complexity of the entire iterative process of the genetic algorithm is $O(mTP_{size})$. Therefore, the time complexity of the VMM-GAGA allocation algorithm is $O(mTP_{size})$.

5 Experiment and result analysis

In this section, we performed simulation experiments for VMM-GAGA, and five performance indicators are mainly considered: (1) Comparing the communication costs generated by the general coding method and the gene aggregation coding mode; (2) Comparing the search time of general coding method and gene aggregation coding method; (3) Comparing the effects of crossover probabilities of VMM-GAGA on communication cost; (4) Compare the effects of mutation probability of effects on communication cost.(5) Comparison of communication cost between general genetic algorithm and VMM-GAGA under different $|GP|$.

5.1 Experimental design

The simulation code for this paper is written in Java, using eclipse development tools, and run on a local computer. The configuration of the local computer is as follows: CPU: Intel Core i7. The Simulation experiment parameters are shown in Table 2.

In order to reduce the randomness of the experimental data, we will conduct 200 experiments and average the experimental results.

5.2 Analysis of experimental results

Figure 5 is the comparison of communication cost between general genetic algorithms and VMM-GAGA. The horizontal axis represents the number of genes (the number of VMs which to be migrated), and the vertical axis represents the communication cost. The number of genes increased from 4 to 10. increasing by one each time. The crossover probability is 0.3 and the mutation probability is 0.1. The number of VMs and hosts is respectively 29 and 8. VMs are randomly assigned to the hosts, resulting in overloaded hosts and low-loaded hosts. As shown in Fig. 5, communication cost generated by VMM-GAGA is less

Table 2 Simulation experiment parameter set

Parameter	Description	Value interval
$O(vm_i)$	VM occupies resources	[7, 14]
$W'(\mathrm{V}_i, \mathrm{V}_j)$	Traffic between VMs	[1, 15]
$W(h_i, h_j)$	Distance between hosts	[1, 5]
$\tau(h_j)$	Host threshold	[40, 45]
$sl(h_j)$	Host security level	[30, 35]
m	Operation operator maximum execution times	5
$\|GP\|$	Initial population	[10, 60]
P_{cross}	Cross probability	[0.1, 0.7]
$P_{mutation}$	Mutation probability	[0.1, 0.5]
T	Number of iterations	1000

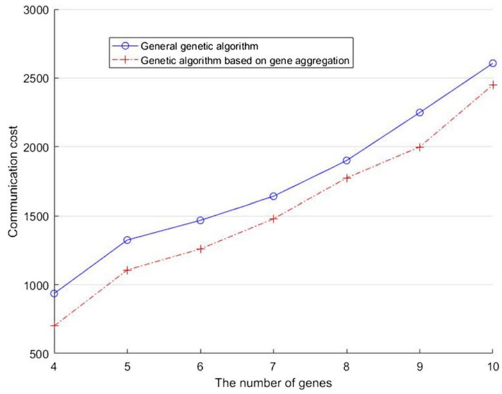


Fig. 5 Comparison of communication cost between general genetic algorithm and VMM-GAGA

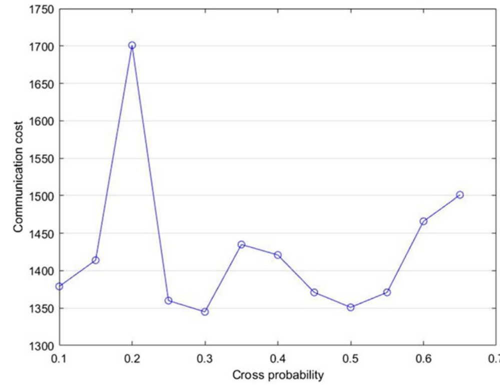


Fig. 7 Effect of crossover probability on VMM-GAGA results

than the communication cost generated by general genetic algorithm.

Figure 6 shows the comparison of runtimes between the general genetic algorithm and VMM-GAGA. The horizontal axis is the number of genes (the number of VMs which to be migrated), and the vertical axis is the running time. The number of genes increased from 4 to 10, increasing by one each time. The crossover probability is 0.3 and the mutation probability is 0.1. The number of VMs and hosts is 29 and 8, respectively. VMs are randomly assigned to the host, resulting in an overloaded host and a low-loaded host. It can be seen from Fig. 6 that the run time of VMM-GAGA is reduced by 10% to 25% compared with the general genetic algorithm.

Figure 7 shows that the effect of different crossover probabilities on communication cost. The horizontal axis is the crossover probability and the vertical axis is the communication cost. The mutation probability is increased from 0.1 to 0.65, increasing by 0.05 each time. The mutation probability is 0.1. The number of VMs and hosts is 29 and 8, respectively. VMs are randomly assigned to the hosts to generate overloaded hosts and low-loaded hosts. Figure 7 shows that the communication cost is the highest when the crossover probability is 0.2, and the communication cost is the lowest when the crossover probability is 0.3. Therefore, the crossover probability of VMM-GAGA is 0.3.

Figure 8 is the effect of different mutation probabilities on communication cost. The horizontal axis is the probability of mutation, and the vertical axis is the communication cost. The mutation probability varies from 0.1 to 0.5, increasing by 0.1 each time. The crossover probability is 0.3. The number of VMs and hosts is 29 and 8, respectively. It can be seen from the Fig. 8 that the communication cost is the highest when the mutation probability is 0.3, and the communication cost is the lowest when the mutation probability is 0.4. Therefore, the mutation probability of VMM-GAGA is 0.4.

Figure 9 is the comparison of communication cost between general genetic algorithm and VMM-GAGA under different initial population size $|GP|$. The horizontal axis represents $|GP|$, and the vertical axis represents the communication cost. The mutation probability is 0.4, The mutation probability is 0.3. The number of chromosomal genes is 8. Figure 9 shows that the communication cost generated by VMM-GAGA is lower than the general

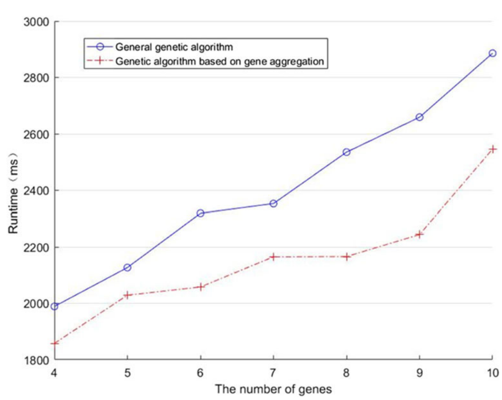


Fig. 6 Comparison of run time between general genetic algorithm and VMM-GAGA

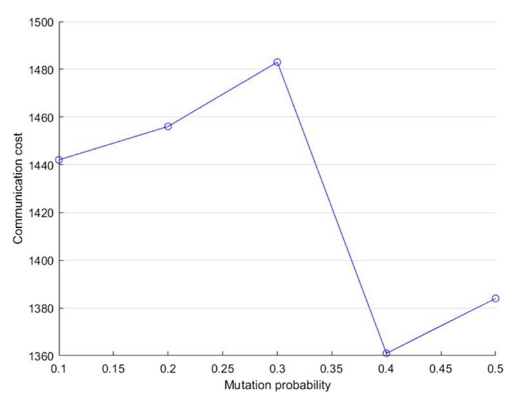


Fig. 8 Effect of mutation probability on VMM-GAGA results

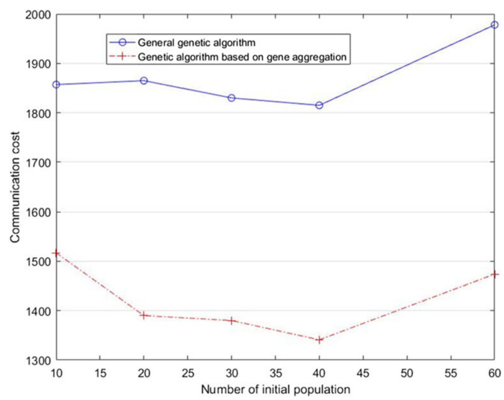


Fig. 9 Comparison of communication cost between general genetic algorithm and VMM-GAGA under different $|GP|$

genetic algorithm. The communication cost decreases with an increase in $|GP|$. However, when $|GP|$ is 60, the communication cost increases. As the initial population size increases, so does the runtime. Therefore, when $|GP|$ is 40, the results of VMM-GAGA are relatively good.

6 Summary

VM migration avoids host overload and reduces data center costs. In this paper we studied Network-aware VM migration due to host overload in the data center and applied the genetic algorithm to the problem of VM migration. We proposed VMM-GAGA to obtain the mapping between VMs which to be migrated and the underutilized hosts. In VMM-GAGA, a novel gene aggregation coding method is studied, which is that two VMs which meet the conditions with larger communication are aggregated to represent the position of a gene. This coding method can reduce the number of genes in the chromosome and the search space of the entire genetic algorithm. This paper constrained the process of implementing genetic algorithms. Only individuals who meet the conditions can enter the next generation, ensuring that the final solution obtained by the VMM-GAGA does not the risk of overloading again after migration. The experiment shows that the running time of VMM-GAGA is reduced about 10%∼25%, and the communication cost of the data center is relatively reduced as well.

References

1. Yang C, Huang Q, Li Z, Kai L, Fei H (2017) Big data and cloud computing: innovation opportunities and challenges. Int J Digital Earth 10(1):13–53
2. Rings T, Caryer G, Gallop J, Grabowski J, Kovacikova T, Schulz S, Stokes-Rees I (2009) Grid and cloud computing: opportunities for integration with the next generation network. Journal of Grid Computing 7(3):375
3. Wang L, Von Laszewski G, Younge A, He X, Kunze M, Tao J, Fu C (2010) Cloud computing: a perspective study. N Gener Comput 28(2):137–146
4. Zhang Q, Cheng L, Boutaba R (2010) Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications 1(1):7–18
5. Al-Dhuraibi Y, Paraiso F, Djarallah N, Merle P (2017) Elasticity in cloud computing: state of the art and research challenges. IEEE Trans Serv Comput 11(2):430–447
6. Mijumbi R, Serrat J, Gorricho JL, Bouten N, De Turck F, Boutaba R (2015) Network function virtualization: State-of-the-art and research challenges. IEEE Commun Surv Tutorials 18(1):236–262
7. Kumar R, Charu S (2015) An importance of using virtualization technology in cloud computing. Global Journal of Computers & Technology 1(2)
8. Malhotra L, Agarwal D, Jaiswal A (2014) Virtualization in cloud computing. International Journal of Computer Science & Mobile Computing 3(8)
9. Younge AJ, Henschel R, Brown JT, Laszewski GV, Qiu J, Fox GC (2011) Analysis of virtualization technologies for high performance computing environments. In: IEEE International Conference on Cloud Computing
10. Razali RAM, Rahman RA, Zaini N, Samad M (2014) Virtual machine migration implementation in load balancing for cloud computing. In: International Conference on Intelligent & Advanced Systems
11. Mishra M, Das A, Kulkarni P, Sahoo A (2012) Dynamic resource management using virtual machine migrations. IEEE Commun Mag 50(9):34–40
12. Gao H, Fu Z, Pun CM et al (2018) A multi-level thresholding image segmentation based on an improved artificial bee colony algorithm[J]. Comput Electr Eng 70:931–938
13. Gao H, Shi Y, Pun CM et al (2018) An improved artificial bee colony algorithm with its application[J]. IEEE Trans Ind Inf 15(4):1853–1865
14. Zhang W, Han S, Hui H, Chen H (2016) Network-aware virtual machine migration in an overcommitted cloud. Future Generation Computer Systems p S0167739X1630053X
15. Zhu J, Wang J, Zhang Y, Jiang Y (2018) Virtual machine migration method based on load cognition. Soft Computing (5), 1–10
16. Beloglazov A, Abawajy J, Buyya R (2012) Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Futur Gener Comput Syst 28(5):755–768
17. Reguri VR, Kogatam S, Moh M (2016) Energy efficient traffic-aware virtual machine migration in green cloud data centers. In: 2016 IEEE 2Nd International Conference on Big Data Security on Cloud (bigdatasecurity), IEEE International Conference on High Performance and Smart Computing (HPSC), and IEEE International Conference on Intelligent Data and Security (IDS), IEEE, pp 268–273
18. Shayeji MHA, Samrajesh M (2012) An energy-aware virtual machine migration algorithm. In: International Conference on Advances in Computing & Communications
19. Kansal NJ, Chana I (2016) Energy-aware virtual machine migration for cloud computing - a firefly optimization approach. Journal of Grid Computing 14(2):327–345
20. Wang X, Du Z, Chen Y, Yang M (2015) A green-aware virtual machine migration strategy for sustainable datacenter powered by renewable energy. Simul Model Pract Theory 58:3–14
21. Shi W, Liu Z (2018) Energy-saving scheduling algorithm for cloud computing center based on virtual machine migration. Computer and Digital Engineering 46(1):39–41

22. Zhao D, Shen S, Wu ZY (2018) An online migration solution for virtual machines for energy saving. Computer Technology and Development 28(2):78–82
23. Dong J, Jin X, Wang H, Li Y, Zhang P, Cheng S (2013) Energy-saving virtual machine placement in cloud data centers. In: 2013 13Th IEEE/ACM International Symposium on Cluster, Cloud, and Grid Computing, IEEE, pp 618–624
24. Wu X, Zeng Y, Lin G (2017) An energy efficient vm migration algorithm in data centers. In: 2017 16Th International Symposium on Distributed Computing and Applications to Business, Engineering and Science (DCABES), IEEE, pp 27–30
25. Bharathi PD, Prakash P, Kiran MVK (2017) Energy efficient strategy for task allocation and vm placement in cloud environment. In: 2017 Innovations in Power and Advanced Computing Technologies (i-PACT), IEEE, pp 1–6