Applied Mathematics & Information Sciences
An International Journal

http://dx.doi.org/10.12785/amis/090236

VMs Placement Strategy based on Distributed Parallel Ant Colony Optimization Algorithm

Gaochao Xu[1,2], *Yushuang Dong*[1,*] *and Xiaodong Fu*[1]

[1] College of Computer Science and Technology, Jilin University, Changchun, China
[2] Symbol Computation and Knowledge Engineer of Ministry of Education, JiLin University, Changchun, China

Received: 7 Jun. 2014, Revised: 5 Sep. 2014, Accepted: 7 Sep. 2014
Published online: 1 Mar. 2015

Abstract: Cloud computing is at the forefront of information technology. Cloud computing led to abandon the use of expensive mainframe. It becomes a trend that data center uses the cluster which is relatively cheap and virtualization technologies to provide infrastructure services. To improve the utilization rate of the cloud center and decrease the operating cost, the cloud center provides services to users as required by sharding the resources with virtualization. Because consideration should be given to both QoS for users and cost saving for cloud computing providers, cloud providers try to maximize performance and minimize energy cost as well. In this paper, we propose a Distributed Parallel Ant Colony Optimization (DPACO) Algorithm of placement strategy for live virtual machines Live Migration on cloud platform. It executes the ant colony optimization algorithm parallelly and distributedly on several selected physical hosts in the first stage. Then it continues to execute the second stage ant colony optimization algorithm with solutions calculated by the first stage. The solution calculated by the second stage ant colony optimization algorithm is the optimal solution of our approach. The experimental results have shown that the proposed placement strategy of VM live migration is more effective and energy-efficient with ensuring QoS for users than other placement strategies on the cloud platform.

Keywords: Cloud computing, Virtual Machine Live Migration, Virtual Machine Placement strategy, Parallel Ant Colony Optimization, Dynamic Voltage Frequency Scaling

1 Introduction

With the development of cloud computing, a growing number of cloud providers provide the infrastructure as a service as their main operations. In order to improve the utilization rate of resources and decrease the operating costs, virtualization technology has been applied to the cloud computing [1,2,3]. Because of the development of Vmware, Xen virtualization technology, some researchers and industry enterprise has done the pioneering researches and applications in the field of cloud computing. With the continuous development of virtualization technology, more and more supports provide to the virtualization by the underlying hardware. It effectively reduces the performance overhead brought by the original virtualization and virtual machine communication latency. Cloud computing technology barriers are disappearing. But the distribution of virtual machines (VMs) will become sparse on cloud center with creating and closing the VMs. VMs live migration and the placement problem of VMs has attracted much more attention and became a research hotspot of cloud computing area quickly. It can be seen as packing problem and has been proved it is a NP-Completeness problem [4].

Traditional placement strateges include linear programming strategy [5,6,7] and constraints programming strategy [8,9]. With the increasing scale of the data center, there are more and more physical hosts and virtual machines on it. The traditional placement strateges are difficult to get the optimal solutions. Therefore heuristic algorithms are used to deal with the virtual machine placement problem. Energy consumption problem attract more and more attention from cloud service providers due to the increasing scale of cloud center. DVFS-enabled physical host can calculate the power consumption by the information such as the core voltage. So we can estimate the energy consumption of cloud center in this way and design the virtual machine

* Corresponding author e-mail: yushuangdong@gmail.com

placement strategy with the energy consumption as a standard.

Ant colony optimization algorithm is one of the solutions to the VMs placement problem. But the ant colony optimization algorithm may stop before it gets a good enough solution in case that there are a large number of servers in cloud platform and there are a certain number of VMs need to live migration. Amdahls law [10] showed that the performance of parallel program executed on single physical host is not better enough than serial program and the research [11,12] showed that we can get a better performance of parallel program by increasing the scale of the problem. Therefore, we propose a new distributed parallel ant colony optimization algorithm (DPACO) of placement strategy executed on several physical hosts to get a better and more accurate solution by increasing the iterative times for the large scale VMs live migration problem. We assign the performance per watt as a pheromone value. It executes on several selected physical hosts for the first stage ant colony optimization algorithm and gets several solutions. Then it collects the solutions which are calculated by the first stage and puts them into the second stage ant colony optimization algorithm. Finally, we get a relatively satisfactory solution calculate by the second stage ant colony optimization algorithm.

2 Related work

The proposed question which refers to find the target for VMs migration according to the standard of maximizing performance and minimizing energy costs as well has not been widely researched in the related field of VMs placement strategy for live migration process. Earlier researches place the virtual machine according to the standard of the utilization rate of resources [13,14,15,16,17]. Some recent researches consider the network factors as the standard of virtual machine placement strategy [18,19]. There also some researches consider the energy consumption as the standard of virtual machine placement strategy.

Von Laszewski G et al. have presented a scheduling algorithm to allocate virtual machines in a DVFS-enabled cluster [20]. The proposed algorithm focused on scheduling virtual machines in a compute cluster to reduce power consumption via the technique of DVFS (Dynamic Voltage Frequency Scaling). It dynamically scales the operating frequencies and voltages of the compute nodes in a cluster without degrading the virtual machine performance beyond unacceptable levels. Verma et al. [21] present a power-aware application placement controller in the context of an environment with heterogeneous virtualized server cluster. The placement component of the application management middleware takes into account the power and migration costs in addition to the performance benefit while placing the application containers on the physical servers. The presented architecture pMapper minimize the power consumption to a fixed performance requirement. In additional, reference [22] proposed a VM placement scheme meeting multiple resource constraints, such as the physical server size (CPU, memory, storage, bandwidth, etc.) and network link capacity to improve resource utilization and reduce both the number of active physical servers and network elements so as to finally reduce energy consumption. It is a novel greedy algorithm by combining minimum cut with the best-fit. Burak Kantarci et al. [23] propose a holistic approach for a large-scale Cloud system where the Cloud services are provisioned by several data centers interconnected over the backbone network. It is a Mixed Integer Linear Programming (MILP) formulation that aims at virtualizing the backbone topology and placing the VMs in data centers with the objective of minimum power consumption. Reference [24] analyze the mathematical relationship of these Service Level Agreements (SLA) and the number of servers that should be used and at what frequencies they should be running. It discuss a proactive provisioning model that includes hardware failures, devices available for services, and devices available for change management, all as a function of time and within constraints of SLAs. Mistral [25] is a holistic controller framework that optimizes power consumption, performance benefits, and the transient costs incurred by various adaptations and the controller itself to maximize overall utility. Mistral can handle multiple distributed applications and large-scale infrastructures through a multi-level adaptation hierarchy and scalable optimization algorithm. Reference [26] studies the inter-relationships between energy consumption, resource utilization, and performance of consolidated workloads. The study reveals the energy performance trade-offs for consolidation and shows that optimal operating points exist. It models the consolidation problem as a modified bin packing problem. EnaCloud [27] enables application live placement dynamically with consideration of energy efficiency in a cloud platform. EnaCloud uses a Virtual Machine to encapsulate the application, which supports applications scheduling and live migration to minimize the number of running machines, so as to save energy. Specially, the application placement is abstracted as a bin packing problem, and an energy-aware heuristic algorithm is proposed to get an appropriate solution. In addition, an over-provision approach is presented to deal with the varying resource demands of applications. It has been successfully implemented as useful components and fundamental services in the iVIC platform.

3 Distributed Parallel Ant Colony Optimization Algorithm of VMs Placement

There are w physical hosts in the cloud platform and n VMs with $(h_0, h_1, h_2, , h_{n-1})$Hz CPU and

$(m_0, m_1, m_2, , m_{n-1})$M RAM need to live migration. We assume that the physical hosts in cloud center are DVFS [28] enabled and the cloud center can satisfy the VMs live migration request, viz, w is big enough for n, viz, $w >> n$ and the physical hosts are in the same network environment. Solution space is recorded as follows: $P = (P_0, P_1, P_2, , P_{w-1})$. We need find n physical hosts to place the live migration VMs and the solution vector is as follows: $S = (S_0, S_1, S_2, , S_{n-1})$. Remaining available CPU resource of physical host P_i is as follows: $AF_i = (1 - U_i) \times F_i$. U_i is the CPU utilization rate of P_i. The parameter F_i is the CPU frequency of P_i. Remaining available memory resource of physical host P_i is as follows: $AM_i = TM_i CUM_i CRM$. TM_i is the total memory size of P_i. UM_i is the using memory size of P_i. RM is the reserve memory size of the system. P_i can be a member of solution only if $AF_i > h$ and $AM_i > m$.

As a user, cloud center should select the physical hosts with more remaining resources to load the migration VMs to improve the QoS for user. As a cloud operator, cloud center should improve the utilization rates of resources and decrease the energy costs to reduce the operating costs. Overall consideration, we assign the performance per watt as evaluation standard, viz, maximize performance and minimize energy costs as well. As showed in Fig.1, the idea of DPACO is divided into two stages. First stage: Ant colony optimization algorithm is executed in parallel on g selected physical hosts. Second stage: Algorithm gets the solutions calculated by the first stage from each selected physical host, and then it executes the ant colony optimization algorithm again like the first stage with collecting solutions.

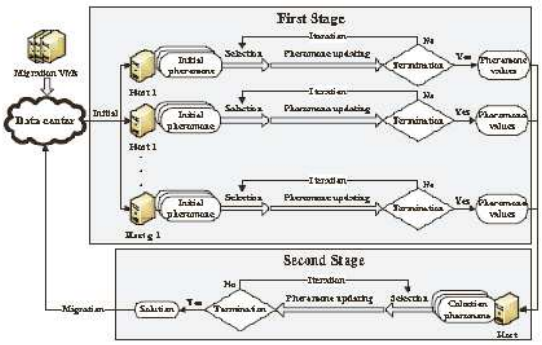

Fig. 1: Distributed parallel ant colony optimization algorithm of VMs placement

3.1 Pheromone Value Calculation

We assign the performance per watt as pheromone value. The performance VM_j can get from physical host P_i after VM_j migration to P_i is recorded as $AF_i \times T$. AF_i is the remaining available CPU resource and T is the work time. The energy consumption increment after VM_j migration to physical host Pi is recorded as ΔE_{ij}. The VCPU frequency of placement VM_j is h_jHz.The relationship among energy, voltage and frequency in CMOS circuits [28] is related by:

$$\begin{cases} E = C \times F \times V^2 \times T \\ F = K \times \dfrac{(V - Vt)^2}{V} \end{cases} \tag{1}$$

Where E is energy consumption, C is CPU circuit switching capacity, F is CPU frequency, V is CPU voltage, K is a factor which depends on technology and Vt is CPU threshold voltage. By formula (1), we can get the relationship between voltage and frequency as follows:

$$V = \sqrt{\frac{F \times Vt}{K} + \frac{F^2}{4 \times K^2}} + Vt + \frac{F}{2 \times K}$$

We can also get the energy consumption increment after VM_j migration to physical host P_i as follows:

$$\begin{aligned} \Delta E_{ij} = & C_i \times (F_i + h_j) \\ & \times \Big(\sqrt{\frac{(F_i + h_j) \times Vt_i}{K_i} + \frac{(F_i + h_j)^2}{4 \times K_i^2}} \\ & + Vt_i + \frac{F_i + h_j}{2 \times K_i}\Big)^2 \times T \\ & - C_i \times F_i \\ & \times \Big(\sqrt{\frac{F_i \times Vt_i}{K_i} + \frac{F_i^2}{4 \times K_i^2}} \\ & + Vt_i + \frac{F_i}{2 \times K_i}\Big)^2 \times T \end{aligned} \tag{2}$$

It updates the $F_i = F_i + h_j$ and $AF_i = AF_i - h_j$ dynamically and temporarily after ΔE_{ij} calculation. The updated F_i and AF_i just works in the process of the pheromone value calculation for current solution vector. Thus, the increment pheromone value I_j after VM_j migration to physical host S_j in our VM placement

strategy for solution vector $S=(S_0,S_1,S_2,,S_{n-1})$ can be expressed as follows:

$$\begin{aligned} I_j &= \frac{AF_j \times T}{\sum_{x=0}^{n-1} \Delta E_x} \\ &= \frac{AF_j}{C_x \times (F_x+h_x)} \\ &\quad \times (\sqrt{\frac{(F_x+h_x)\times Vt_x}{K_x} + \frac{(F_x+h_x)^2}{4\times K_x^2}} \\ &\quad + Vt_x + \frac{F_x+h_x}{2\times K_x})^2 \\ &\quad - C_x \times F_x \\ &\quad \times (\sqrt{\frac{F_x \times Vt_x}{K_x} + \frac{F_x^2}{4\times K_x^2}} \\ &\quad + Vt_x + \frac{F_x}{2\times K_x})^2 \end{aligned} \tag{3}$$

3.2 Pheromone Value Calculation

We assign the parallel parameter as g and set g as jump volume among the member of solution spaces of selected parallel physical. Instead of random way, we assign the solution space for each selected parallel physical host $P_x(0\le x<g)$ is as follow:

$$Q_x = \begin{cases} P_{i\times g+x}, & 0\le i<w/g, i\times g+x<w \\ P_{i\times g+x-w}, & 0\le i<w/g, i\times g+x\ge w \end{cases}$$

To ensure the algorithm executes correctly in this paper, we assume that $w/g>1$. For instance, we set $w=1000$, $g=4$. The solution spaces of selected parallel physicals are showed in Table 1:

Table 1: An example of solution spaces of selected parallel physicals

$Physicalhost(x)$	$Solutionspace(x)$
0	$(P_0,P_4,P_8,...,P_{992},P_{996})$
1	$(P_1,P_5,P_9,...,P_{993},P_{997})$
2	$(P_2,P_6,P_{10},...,P_{994},P_{998})$
3	$(P_3,P_7,P_{11},...,P_{995},P_{999})$

We assign the ants number is m. Maximum number of iterations is r. The pheromone trail decay coefficient is $\rho\in(0,1]$. The heuristic value of P_i for VM_j is as follow:

$$\begin{aligned} \eta_{ij} &= \frac{AF_i \times T}{\Delta E_{ij}} \\ &= \frac{AF_i}{C_i \times (F_i+h_j)} \\ &\quad \times (\sqrt{\frac{(F_i+h_j)\times Vt_i}{K_i} + \frac{(F_i+h_j)^2}{4\times K_i^2}} \\ &\quad + Vt_i + \frac{F_i+h_j}{2\times K_i})^2 \\ &\quad - C_i \times F_i \\ &\quad \times (\sqrt{\frac{F_i \times Vt_i}{K_i} + \frac{F_i^2}{4\times K_i^2}} \\ &\quad + Vt_i + \frac{F_i}{2\times K_i})^2 \end{aligned} \tag{4}$$

We define the pheromone value at iteration time $l(0\le l<r)$ of P_i for VM_j as $\tau_{ij}(l)$.We assign the initial pheromone value $\tau_{ij}(0)$ of P_i for VM_j is $\tau_{ij}(0)=\rho\times\eta_{ij}$.

3.3 Selection process in the First Stage

Selection operations select the physical host for VM to migration according to the probability value. The $a(0\le a<m)$ ant selection probability $\phi_{ij}^a(l)$ at iteration time $l(0\le l<r)$ of physical host P_i for VM_j is as follows:

$$\phi_{ij}^a(l) = \begin{cases} \dfrac{[\tau_{ij}(l)]^\alpha \times [\eta_{ij}]^\beta}{\sum_{b=0}^{w-1}[\tau_{bj}(l)]^\alpha \times [\eta_{bj}]^\beta}, & AF_j>h_j, AM_i>m_j \\ 0, & AF_j\le h_j, AM_i\le m_j \end{cases}$$

The parameters α and β are used for controlling the relative weight of the pheromone trail. The selection probability area σ_{ij} for ant a at iteration time l of physical host P_i for VM_j between 0 and 1 is as follows:

$$\sigma_{ij}^a(l) = \begin{cases} [0,\sigma_{ij}(l)), & i=0 \\ [\sum_{b=0}^{y-1}\sigma_{bj}^a(l), \sum_{b=0}^{y}\sigma_{bj}^a(l)], & 0<y<w-1 \\ (\sum_{b=0}^{y-1}\sigma_{bj}^a(l), 1], & y=w-1 \end{cases}$$

The probability of each physical host is obtained by calculated $\sigma_{ij}^a(l)$ according to probability theory and mathematical statistics. At the iteration time $l(0\le l<r)$, each migrate VM has a probability wheel as shown in Fig.2. A pointer randomly rotates on the probability wheel. The physical host which the area where the pointer finally stops represents is the target physical host of live

migration. In the specific implementation, a random number limited between 0 and 1 can be used to find the location of live VM migration by finding the range within which the random number is. By utilizing this approach, each migrant VM gets the location selection of live VM migration.

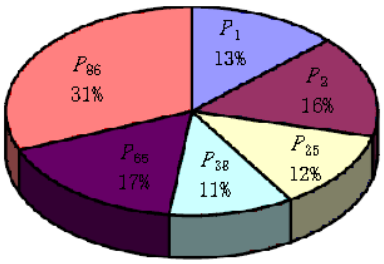

Fig. 2: Example of the probability wheel

3.4 Pheromone Value Updating in the First Stage

If all m ants complete finding the physical hosts for n VMs to migration at the iteration time $l(0 \le l < r)$, the process update the pheromone value for physical hosts. Because migration VMs are heterogeneous, we record pheromone value for each migration VM separately. The pheromone value at iteration time $(l+1)(l+1<r)$ for S_j of solution vector $S = (S_0, S_1, S_2, , S_{n-1})$ can be expressed as follows:

$$\begin{aligned}\tau_j(l+1) &= (1-\rho)\times\tau_j(l) + I_j \\ &= (1-\rho)\times\tau_j(l) + \\ &\frac{AF_j}{C_x\times(F_x+h_x)} \\ &\times\Big(\sqrt{\frac{(F_x+h_x)\times Vt_x}{K_x}+\frac{(F_x+h_x)^2}{4\times K_x^2}} \\ &+Vt_x+\frac{F_x+h_x}{2\times K_x}\Big)^2 \\ &\sum_{x=0}^{n-1} -C_x\times F_x \\ &\times\Big(\sqrt{\frac{F_x\times Vt_x}{K_x}+\frac{F_x^2}{4\times K_x^2}} \\ &+Vt_x+\frac{F_x}{2\times K_x}\Big)^2\end{aligned} \tag{5}$$

After calculating the pheromone value at iteration time $(l+1)$, we update and record the pheromone value for P_i corresponding to S_j.

3.5 Iteration and Termination in the First Stage

The algorithm judges whether it meets the iterative terminal conditions in the first stage. The algorithm stops iterating the first stage if it meets the iterative terminal conditions, otherwise it continues iterating the first stage. The solution vector that has maximum pheromone value is the optimal solution vector of the first stage. The pheromone value of the solution vector is as follow:

$$\begin{aligned}I = &\frac{\sum_{z=0}^{n-1} AF_z}{C_x\times(F_x+h_x)} \\ &\times\Big(\sqrt{\frac{(F_x+h_x)\times Vt_x}{K_x}+\frac{(F_x+h_x)^2}{4\times K_x^2}} \\ &+Vt_x+\frac{F_x+h_x}{2\times K_x}\Big)^2 \\ &\sum_{x=0}^{n-1} -C_x\times F_x \\ &\times\Big(\sqrt{\frac{F_x\times Vt_x}{K_x}+\frac{F_x^2}{4\times K_x^2}} \\ &+Vt_x+\frac{F_x}{2\times K_x}\Big)^2\end{aligned} \tag{6}$$

The iterative terminal conditions of the first stage are as follows:

1. Iterative times meet the preset maximum iterative times of the first stage. We set the maximum iterative times of the first stage with r. The value of r is related to w/g and n. 2. It reaches a certain proportion to get the same solution vector with maximum pheromone value by m ants. We set the proportion of the first stage termination with μ.

3.6 Ant Colony Optimization in Second Stage

After completing the iteration in the first stage, algorithm collects the pheromone values of physical hosts for each VM calculated by the first stage from each selected physical host for the second stage. The solution space of the second stage is all solution space. We assign the ants number is $g\times m$. Maximum number of iterations is $g\times r$. The pheromone trail decay coefficient is $\rho \in (0,1]$. Selection process of the second stage selects the solution vectors the same as the first stage.

After selection process and pheromone value updating in the second stage, the algorithm judges whether it meets the iterative terminal conditions in the second stage. The algorithm stops iterating the second stage if it meets the iterative terminal conditions, otherwise it continues iterating the second stage. The solution vector that has maximum pheromone value is the optimal solution vector of the algorithm. The iterative terminal conditions of the second stage are as follows:

1. Iterative times meet the preset maximum iterative times of the first stage. We set the maximum iterative times of the first stage with $g \times r$. The value of $g \times r$ is related to w and n.

2. It reaches a certain proportion to get the same solution vector with maximum pheromone value by $g \times m$ ants. We set the proportion of the first stage termination with $1-(1-\mu)/g$.

4 Evaluation

In order to simulate a dynamic cloud platform, we utilize a cloud simulator named CloudSim toolkit [29] with the version 3.0.3. The CloudSim framework can dynamically create different kinds of entities and remove data center entities at run-time. The CloudSim framework can also calculate the status information of entities during the simulation period such as resource utilization, power consumption, etc. We choose 6 kinds of host models as showed in Table 2 for the CloudSim experiments.

Table 2: Host models for CloudSim platform in the experiments

Host	CPU	RAM (G)
IBM System X3650 M4	2 x [Intel Xeon E5-2660 2200 MHz, 10 cores]	64
IBM System X3300 M4	2 x [Intel Xeon E5-2470 2300 MHz, 8 cores]	24
Dell PowerEdge R710	2 x [Intel Xeon X5675 3066 MHz, 6 cores]	24
Dell PowerEdge R610	2 x [Intel Xeon X5670 2933 MHz, 6 cores]	12
Acer Altos AR580 F2	4 x [Intel Xeon X4607 2200 MHz, 6 cores]	64
Acer Altos R380 F2	2 x [Intel Xeon X2650 2000 MHz, 8 cores]	24

According to Table 3, we need to create power model classes for each kind of host models to calculate the power consumption of the hosts in CloudSim platform [30].

In the experiments, some parameters of the hosts are needed by the DPACO algorithm. The parameters C, K and Vt of the hosts should have been obtained from the hardware providers, but the CloudSim platform does not provide these parameters. So we need to calculate the approximate values of the parameters. Firstly, we get two sets core voltage and core frequency of each kind of host models and calculate the power consumptions. Then we utilize the matlab [31] to solve the multiple equations established by formula (1) according to the information of Table 4. The values of parameters are showed in Table 4.

The class PowerHost of CloudSim platform does not contain the member variables of C, K and Vt. We create a new class extends the class PowerHost by adding the member variables of C, K and Vt so that the entities in the experiments can record the information of parameters for DPACO. In the experiments, a data center consisting of w hosts is created. These hosts are composed of 6 kinds of host models described above averagely. Then the data center creates d VMs according to Table 5 averagely with full utilization model as the original loads of data center.

Table 3: Benchmark results summary of host models [30]

Host	Power consumption for different target loads (W)										
	0%	10%	20%	30%	40%	50%	60%	70%	80%	90%	1
IBM X3650	52.7	80.5	90.3	100	110	120	131	143	161	178	203
IBM X3300	50.8	74.3	84.1	94.5	106	122	141	164	188	220	260
Dell R710	62.2	104	117	127	137	147	157	170	187	205	227
Dell R610	61.9	102	115	126	137	149	160	176	195	218	242
Acer AR580	109	155	170	184	197	211	226	252	280	324	368
Acer R380	52.9	77.1	85.4	94	102	110	124	141	162	186	215

Table 4: Parameters summary of host models

Host	V(V)	F(MHz)	E(W)	ΔE(W)	$C(\mu$F)	$K(10^9)$	Vt(V)
IBM X3650	0.806 1.172	800 2100	106.364 191.636	85.272	0.501	87.565	0.422
IBM X3300	0.986 1.433	821.5 2135.9	101.075 231.461	130.386	0.631	57.787	0.512
Dell R710	0.857 1.246	1066.4 2932.6	131.781 217.428	85.647	0.526	72.411	0.468
Dell R610	0.906 1.317	980 2744	129.754 226.535	96.781	0.566	65.298	0.502
Acer AR580	0.994 1.445	800 2100	192.273 348	191.727	0.617	85.299	0.521
Acer R380	0.953 1.386	700 1900	98 200.5	102.5	0.59	55.441	0.514

Table 5: VM models for loads of data center

Item	VM models							
	1	2	3	4	5	6	7	8
VCPU (MHz)	1000 $\times$1	1200 $\times$2	1300 $\times$2	1400 $\times$4	1500 $\times$4	1600 $\times$6	1800 $\times$6	2000 $\times$8
RAM (M)	512	1024	1024	2048	2048	4096	4096	8192

In the experiments, the data center migrate n VMs according to Table 6 with full utilization model.

Table 6: VM models for migration

Item	VM models									
	1	2	3	4	5	6	7	8	9	10
VCPU (MHz)	1000 ×1	1100 ×2	1300 ×2	1400 ×4	1500 ×4	1600 ×6	1700 ×6	1800 ×8	1900 ×8	2000 ×10
RAM (M)	256	512	512	1024	1024	2048	2048	4096	4096	8192

In this scenario, we compare efficiency of DPACO with ST (Static Threshold) [30] which set the utilization threshold to 0.9, IQR (Inter Quartile Range) [30] which set the safety parameter to 1.5, LR (Local Regression) [30] which set the safety parameter to 1.2, LRR (Local Regression Robust) [30] which set the safety parameter to 1.2 and MAD (Median Absolute Deviation) [30] which set the safety parameter to 2.5 in different conditions.

4.1 Comparison in Performance Per Watt

In this experiment, we set the hosts number of the data center $w = 1600$ and the VMs number of migration $n = 100$. We adjust the VMs number d as the original loads from 0 to 5000 and allocate these VMs to the hosts randomly. It represents different load levels of the data center. All idle hosts are switched to sleep state. The experiment is designed for verifying the efficiency of DPACO for VMs migration in performance per watt of a cloud center with different original loads. As illustrated in Fig.3, the cloud center implementing DPACO placement strategy for VMs migration with different original loads gets higher performance per watt than other placement strategies. Further, when $d \leq 2000$, viz, many hosts are also in an idle state, performance per watt of DPACO placement strategy is increasing with adding the original loads. This is because the hosts at the states idle to load will consume more power than the hosts with a slightly loads. When $2000 < d \leq 3000$, viz, most of hosts at a low loading state, performance per watt of DPACO placement strategy is relatively stable. When $d > 3000$, viz, the data center at the certain loading state, performance per watt of DPACO placement strategy begun to decline gradually. This is because the hosts at a certain loads will consume more and more power with adding the system overhead.

Then, we set the original loads $d = 3000$ as fixed value and adjust the VMs number of migration n from 10 to 500. All idle hosts are switched to sleep state. We compare efficiency of DPACO for VMs migration in performance per watt of a cloud center with different number of migration VMs. As illustrated in Fig.4, the cloud center implementing DPACO placement strategy for VMs migration of different number of VMs migration

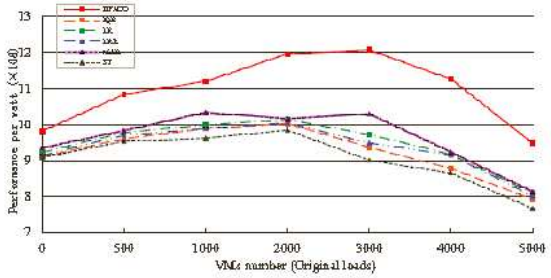

Fig. 3: Comparison in performance per watt with different original loads

gets higher performance per watt than other placement strategies. Further, with increasing of the number of migration VMs, there has a little impact on the performance per watt of DPACO placement strategy for VMs migration and performance per watt of other placement strategies decline obviously. We draw the conclusion that DPACO gets more stable performance per watt than other placement strategies. This is because the DPACO placement strategy is the heuristic approach for performance per watt.

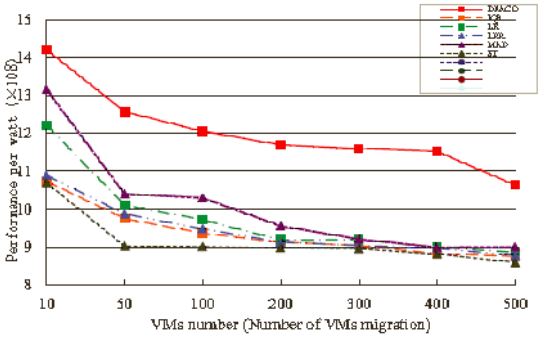

Fig. 4: Comparison in performance per watt with different number of VMs migration

4.2 Comparison of the Number of Failures in VM Migration Events

In this experiment, we set number of VMs migration n from 10 to 500. We set the original loads $d = 4000$ as fixed value and allocate these VMs to the hosts randomly. All idle hosts are switched to sleep state. Migration events have caused some failures in VM migration due to the non-availability of the selected hosts for some VM migration requests. As illustrated in Fig.5, DPACO

performs better in finding the fit hosts in the dynamic resource pool and has a lower VM migration failure rate in VM migration events than other placement strategies. This is because other policies cant adjust the solution with the environment changed and can't deal with the migration failures efficiently. DPACO policy can efficiently detect the host failures and adjust the solution simultaneously.

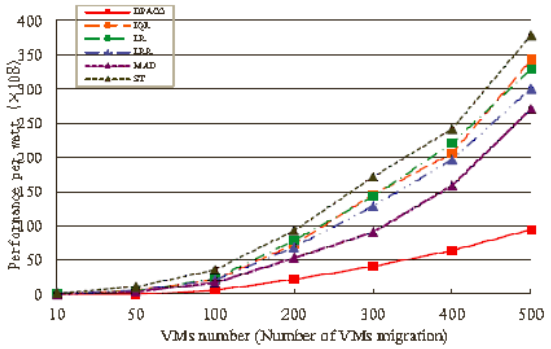

Fig. 5: Comparison of the number of failures in VM migration events

5 Evaluation

In this paper, we presented the design, implementation and evaluation of a placement strategy of virtual machines live migration based on distributed parallel ant colony optimization algorithm on cloud platform. The algorithm is divided into two stages to get a better and more accurate solution. We assign the performance per watt as evaluation standard. It executes the first stage ant colony optimization algorithm on the several selected physical hosts and puts the pheromone values calculated by the first stage into the second stage ant colony optimization algorithm. Then it gets a relatively good solution calculated by the second stage. The experimental results show that our approach is an efficient and high performance per watt placement strategy for VMs migration.

To further improve the performance of placement strategy for VMs migration, there are also many problems need to be solved in the future. The number of parallel executions in the first stage g should be related to the size of solution space w and the number of VMs migration n. We plan to assign the value of g according to w and n. In the judgment of iterative terminal conditions, the maximum iteration times should be related to the size of solution space w and the number of VMs migration n. We plan to assign the maximum iteration times according to w and n. There is also an open question on the termination of the two stages. It is the proportion value to get the same solution vector with maximum pheromone value by ants. Our approach is appropriate for the case that all physical hosts in solution space are in a fast LAN and the same network environment. We plan to extend our approach to WAN and different network environment. Our approach uses the parallel ant colony optimization algorithm. We plan to use other heuristic algorithms such as genetic algorithm, bee colony algorithm and particle swarm optimization to implement our approach and compare their various performance.

Acknowledgement

The authors would like to thank the editors and anonymous reviewers for their valuable comments.

References

[1] Nakajima J, Lin Q, Yang S, et al. Optimizing Virtual Machines Using Hybrid Virtualization. Proc. Of the 2011 ACM Symposium on Applied Computing (SAC11). 573-578 (2011).
[2] Nathan Regola, Jean-Christophe Ducom. Recommendations for Virtualization Technologies in High Performance Computing. 2010 IEEE second International Conference on Cloud Computing Technology and Science. 409-416 (2010).
[3] Barham P, Dragovic B, Fraser K, et al. Xen and the art of virtualization [C]. New York, USA: Proceedings of the 19th ACM Symposium on Operating Systems Principles. (2003).
[4] Garey Michael R, Johnson David S. Computers and intractability-a guide to the theory of np-completeness [M]. San Francisco: W H Freeman Co. (1979).
[5] Chaisiri S, Lee B S, Niyato D. Optimal virtual machine placement across multiple cloud providers[C]. Services Computing Conference, 2009. APSCC 2009. IEEE Asia-Pacific. IEEE. 103-110 (2009).
[6] Bichler M, Setzer T, Speitkamp B. Capacity planning for virtualized servers[C]. Workshop on Information Technologies and Systems (WITS), Milwaukee, Wisconsin, USA. (2006).
[7] Speitkamp B, Bichler M. A mathematical programming approach for server consolidation problems in virtualized data centers[J]. Services Computing, IEEE Transactions on. **34**, 266-278 (2006).
[8] Van H N, Tran F D, Menaud J M. Performance and power management for cloud infrastructures[C]. Cloud Computing (CLOUD), 2010 IEEE 3rd International Conference on. IEEE. 329-336 (2010).
[9] Hermenier F, Lorca X, Menaud J M, et al. Entropy: a consolidation manager for clusters[C]. Proceedings of the 2009 ACM SIGPLAN/SIGOPS international conference on Virtual execution environments. ACM. 41-50 (2009).
[10] Amdahl G M. Validity of the single processor approach to achieving large scale computing capabilities[C]. Proceedings of the April 18-20, 1967, spring joint computer conference. ACM. 483-485 (1967).

[11] Gustafson J L. Reevaluating Amdahl's law[J]. Communications of the ACM. **31**, (1988).
[12] Sun X H, Ni L M. Scalable problems and memory-bounded speedup[R]. Institute For Computer Applications in Science and Engineering Hampton VA. (1992).
[13] Machida F, Kim D S, Park J S, et al. Toward optimal virtual machine placement and rejuvenation scheduling in a virtualized data center[C]. Software Reliability Engineering Workshops, 2008. ISSRE Wksp 2008. IEEE International Conference on. IEEE. 1-3 (2008).
[14] Kochut A. On impact of dynamic virtual machine reallocation on data center efficiency[C]. Modeling, Analysis and Simulation of Computers and Telecommunication Systems, 2008. MASCOTS 2008. IEEE International Symposium on. IEEE. 1-8 (2008).
[15] Viswanathan B, Verma A, Dutta S. Cloudmap: Workload-aware placement in private heterogeneous clouds[C]. Network Operations and Management Symposium (NOMS), 2012 IEEE. IEEE. 9-16 (2012).
[16] Bobroff N, Kochut A, Beaty K. Dynamic placement of virtual machines for managing sla violations[C]. Integrated Network Management, 2007. IM'07. 10th IFIP/IEEE International Symposium on. IEEE. 119-128 (2007).
[17] Wang M, Meng X, Zhang L. Consolidating virtual machines with dynamic bandwidth demand in data centers[C]. INFOCOM, 2011 Proceedings IEEE. IEEE. 71-75 (2011).
[18] Hou W, Guo L, Wei X, et al. Multi-granularity and robust grooming in power-and port-cost-efficient IP over WDM networks[J]. Computer Networks. **56**, 2383-2399 (2012).
[19] Cuomo F, Cianfrani A, Polverini M, et al. Network pruning for energy saving in the Internet[J]. Computer Networks. **56**, 2355-2367 (2012).
[20] von Laszewski G, Wang L, Younge A J, et al. Power-aware scheduling of virtual machines in dvfs-enabled clusters[C]. Cluster Computing and Workshops, 2009. CLUSTER'09. IEEE International Conference on. IEEE. 1-10 (2009).
[21] Verma A, Ahuja P, Neogi A. pMapper: power and migration cost aware application placement in virtualized systems[M]. Middleware 2008. Springer Berlin Heidelberg. 243-264 (2008).
[22] Dong J, Jin X, Wang H, et al. Energy-Saving Virtual Machine Placement in Cloud Data Centers[C]. Cluster, Cloud and Grid Computing (CCGrid), 2013 13th IEEE/ACM International Symposium on. IEEE. 618-624 (2013).
[23] Kantarci B, Foschini L, Corradi A, et al. Inter-and-intra data center VM-placement for energy-efficient large-Scale cloud systems[C]. Globecom Workshops (GC Wkshps), 2012 IEEE. IEEE. 708-713 (2012).
[24] Abdelsalam H S, Maly K, Mukkamala R, et al. Analysis of energy efficiency in clouds[C]. Future Computing, Service Computation, Cognitive, Adaptive, Content, Patterns, 2009. COMPUTATIONWORLD'09. Computation World. IEEE. 416-421 (2009).
[25] Jung G, Hiltunen M A, Joshi K R, et al. Mistral: Dynamically managing power, performance, and adaptation cost in cloud infrastructures[C]. Distributed Computing Systems (ICDCS), 2010 IEEE 30th International Conference on. IEEE. 62-73 (2010).
[26] Srikantaiah S, Kansal A, Zhao F. Energy aware consolidation for cloud computing[C]. Proceedings of the 2008 conference on Power aware computing and systems. USENIX Association. **10** (2008).
[27] Li B, Li J, Huai J, et al. Enacloud: An energy-saving application live placement approach for cloud computing environments[C]. Cloud Computing, 2009. CLOUD'09. IEEE International Conference on. IEEE. 17-24 (2009).
[28] Hsu C H, Kremer U, Hsiao M. Compiler-directed dynamic frequency and voltage scheduling[M]. Power-Aware Computer Systems. Springer Berlin Heidelberg. 65-81 (2001).
[29] Calheiros R N, Ranjan R, Beloglazov A, et al. CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms[J]. Software: Practice and Experience. **41**, 23-50 (2011).
[30] Beloglazov A, Buyya R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in Cloud data centers[J]. Concurrency and Computation: Practice and Experience. **24**, 1397-1420 (2012).
[31] MathWorks T. Matlab[J]. The MathWorks, Natick, MA. (2004).

Gaochao Xu Professor of College of Computer Science and Technology, Jilin University, China. His research interests are in the areas of Distributed System, Grid Computing, Cloud Computing, Internet Things, etc. He has published 55 research articles.

Yushuang Dong PhD of College of Computer Science and Technology, Jilin University, China. His research interests are in the areas of Distributed System, Grid Computing, Cloud Computing, Internet Things, etc. He has published 8 research articles.

Xiaodong Fu senior engineer of College of Computer Science and Technology, Jilin University, China. His research interests are in the areas of Distributed System, Grid Computing, Cloud Computing, Internet Things, etc. He has published 14 research articles.