Research on Energy-aware Virtual Machine Scheduling in Cloud Environment

Gang JIN\textsuperscript{1,*}, Lei LIU\textsuperscript{1}, Peng ZHANG\textsuperscript{1}, Man YU\textsuperscript{2}

\textsuperscript{1}College of Computer Science and Technology, Jilin University, Changchun 130000, China
\textsuperscript{2}Jilin Provincial People’s Congress, Changchun 130000, China

Abstract

Nowadays, there are lots of problems in Cloud data center such as high energy consumption and low efficiency. We model a cloud environment with physical hosts and virtual machines, and design an Energy-Aware Virtual Machine Scheduling strategy (EAVMS) to reduce energy consumption and improve utilization rate by integrating system resource such as computing, storage and bandwidth resources. Experimental results show that the algorithm effectively reduce energy consumption, while improving system multi-dimensional resource utilization and minimize cost of virtual machine migration.

Keywords: Cloud Computing; Particle Swarm Optimization; Green Computing; Virtual Machine

1 Introduction

With the rapid development of computer science, Big Data era has come. The entire society including politics and economy depends on information systems. Information industry is inseparable from the consumption of electric power. Electricity consumption of large-scale computing clusters including cloud computing data center has already become a big issue. Cloud computing providers must guarantee the quality of service, and Service Level Agreement (SLA) is widely used to describe service quality. How to maximize the utilization rate of the whole system and reduce unnecessary energy consumption is an issue which cloud computing providers must concern.

Green computing which aims to study new computing systems and models with low-energy and low-cost is currently widely concerned. The number of users is growing exponentially in cloud environment, and the rapid growth of users is more likely to cause a serious waste of resources. Reducing energy consumption of cloud computing data center is an important area of green computing applications.

We proposed an Energy-Aware Virtual Machines Scheduling strategy which can reduce energy consumption, improve system utilization rate and minimize the cost of virtual machine migration.
2 Related Work


3 Mathematical Models

3.1 Physical host model

\[ H = \{h_0, h_1, \ldots, h_{m-1}\} \] represents the set of physical hosts which may have different types in cloud environment, and \( h_j \) represents the No. \( j \) host in physical hosts set \( H \), and

\[ h_j = \langle h_{cpu}^j, h_{mem}^j, h_{net}^j, h_{stor}^j \rangle, \quad (j \in [0, m-1]) \] (1)

In which, \( h_{cpu}^j \) is the CPU resource of host \( h_j \); \( h_{mem}^j \) is the memory resource; \( h_{net}^j \) is the network resource and \( h_{stor}^j \) is storage capacity.

Tasks assignment and VM migration are affected by physical hosts network bandwidth. We define \( h_{band} = \langle h_{net}^1, h_{net}^2, \ldots, h_{net}^m \rangle \) as the entire cloud computing systems network bandwidth status.

We define \( H_{dis} \) as the distance matrix which represents the actual distance between physical hosts and

\[
H_{dis} = \begin{pmatrix}
\text{dis}_{11} & \ldots & \text{dis}_{1m} \\
\vdots & \ddots & \vdots \\
\text{dis}_{m1} & \ldots & \text{dis}_{mm}
\end{pmatrix}
\] (2)
The distance vector from physical host $h_j$ to the others is $dis_j = \langle dis_{j1}, dis_{j2}, \ldots, dis_{jm} \rangle$.

### 3.2 Virtual machine model

$V = \{vm_0, vm_1, \ldots, vm_{q-1}\}$ represents the set of VMs in cloud environment, and $q = |V|$ is the number of virtual machines in set $V$. Different users and applications desire different hardware configuration, so cloud must provide heterogeneous virtual machines, and virtual machine’s configuration parameters are based on user needs varies. Specifically, we define $cv_i$ as configuration parameters of $vm_i$.

$$cv_i = \langle cv^i_{cpu}, cv^i_{mem}, cv^i_{net}, cv^i_{stor} \rangle, \ (i \in [0, q - 1]) \quad (3)$$

### 3.3 Virtual machine migration cost

Migration of virtual machines is a dynamic complex process. When a virtual machine migrates from one physical host to another, data and storage structures in memory correspondingly transfer to the target host. The memory migration process is divided into three stages: Push, Stop-and-Copy, Pull. During the process of migration, memory will continue to be sustained rewrite, so migration will not complete until all the dirty pages are transferred. We need to define a virtual machine migration cost evaluation mechanism.

$cost^k_{ij}$ represents the migration cost of VM $vm_k$ migrates from $h_i$ to $h_j$, and

$$cost^k_{ij} = \begin{cases} \frac{\varphi sec_{mem}^k}{\min(h_i^{net}, h_j^{net})} + \frac{dis_{ij}}{\theta}, & i \neq j \\ 0, & i = j \end{cases} \quad (4)$$

In which, $sec_{mem}^k$ is the memory size of virtual machine $vm_k$; $h_i^{net}$ and $h_j^{net}$ are the bandwidth of physical machine $h_i$ to $h_j$. $\varphi$ is the rate of dirty pages cost coefficient, represented by a random number in $[1, 2]$. When $\varphi$ is high, it shows that dirty pages rate is high and the migrating cost is high; $\theta$ is a random number in $[0.5, 1]$ which is shows how the routing forward is impact on the transmission time; $dis_{ij}$ is the distance among hosts.

$X_{mn}$ represents the virtual machine deployment matrix.

$$X_{mn} = \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1n} \\ x_{21} & x_{22} & \ldots & x_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ x_{m1} & x_{m2} & \ldots & x_{mn} \end{pmatrix} \quad (5)$$

In which, $X_{mn}$ is $0 - 1$ variable, $x_{ij} = 1$ shows that task $vm_i$ is running on physical machine $h_j$ and $x_{ij} = 0$ shows that task $vm_i$ is running on physical machine $h_j$.

We define $loadvm_k$ as virtual machines load state model.

$$loadvm_k = \langle id_k, cv_k, task\ finish_k, belong_k \rangle \quad (6)$$
In which, \( id_k \) is the unique identification of \( vm_k \) in the virtual machine set \( V \). \( cv_k \) is the parameter configuration. \( taskfinish_k \) is the latest finishing time of task queue, \( belong_k \) is the physical host which \( vm_k \) belongs to.

### 3.4 Physical host energy model

This article is based on the mechanism of resource utilization and energy consumption has a linear relationship. We define \( U_j \) as physical host \( h_j \) s multidimensional weighted resource utilization

\[
U_j = a \sum x_{ji} \cdot c_{v_{cpu}}^i \cdot \frac{r_{cpu}}{r_{cpu}^j} + b \sum x_{ji} \cdot c_{v_{mem}}^i \cdot \frac{r_{mem}}{r_{mem}^j} + c \sum x_{ji} \cdot c_{v_{net}}^i \cdot \frac{r_{net}}{r_{net}^j} + d \sum x_{ji} \cdot c_{v_{stor}}^i \cdot \frac{r_{stor}}{r_{stor}^j}
\]

(7)

In which, \( x_{ji} \) indicates whether virtual machine \( v_i \) runs on physical host \( h_j \). \( U_j \) is weighted summation of each resource dimension.

We define \( E_j \) as energy consumption per unit of time of physical host \( h_j \).

\[
E_j = (P_{j}^{max} - P_{j}^{min}) \cdot U_j + P_{j}^{min}
\]

(8)

In which, \( P_{j}^{max} \) is the full load power consumption of physical host \( h_j \), and \( P_{j}^{min} \) is the empty load power consumption of physical host \( h_j \). \( E_j \) is linearly related with \( U_j \). Here we assume that we get energy consumption per unit of time of the physical host at full-load and no-load.

We define \( W(r_j, t) \) as Period energy consumption of \( h_j \).

\[
W(r_j, t) = \int E_j dt \approx E_j \cdot span
\]

(9)

We assume that a task exclusive a VM, and a tasks time span is related to the tasks number of instructions processed and VMs CPU processing capacity.

\[
span = \frac{t_{command}}{r_{cpu}^k}
\]

(10)

### 4 Virtual Machine Scheduling Algorithm

Tasks in cloud environment are massive and arrived at any time. In order to improve the processing efficiency, we adopt a task stack buffer centralized processing mode. Specifically, when a task arrived, we don’t assign it to a VM immediately, but add it to task stack. When the task number reaches \( m \), we centralized allocate all \( m \) tasks. We set a time threshold to avoid the number of tasks is too small in a period of time. When the time since last allocation reaches \( t_s \), no matter how many tasks in the stack, we carry out a centralized distribution.

We use a two-stage load balancing scheduling algorithm to assign tasks set \( T_x \). Firstly, we assign all tasks to available VMs which are controlled by virtual machine monitor. If all tasks are assigned successfully, the algorithm is complete. If there are one or many tasks cannot be assigned, it means all available VMs cannot carry these tasks, then new VM must be created to carry them. The new VM will find a suitable placement physical host based on two-stage load balancing scheduling algorithm.
For the purposes of green computing, we set each physical machine into 3 states: open, sleep, down. Corresponding physical hosts in H are divided into 3 subsets: \( H_{open}, H_{sleep}, H_{down} \).

\[
H = \{H_{open}, H_{sleep}, H_{down}\} \tag{11}
\]

Physical host controller adjusts physical hosts status according to current operating status. When the tasks are too many, we need to open more hosts; when idle hosts increase, we need to make some physical host into sleep mode or even turn them down.

With the completion and arrival of tasks, virtual machines are dynamically created and destroyed. We design an Energy-aware virtual machine scheduling strategy (EAVMS). The algorithm is encoded by the migration program, carried out by particle swarm optimization in the solution space, and ultimately found a lower energy consumption and migrating less costly migration program.

EAVMSs goal is to find a virtual machine migration program, which will have low energy consumption and low migration cost. Each virtual machine migration program can correspond to a particle in Particle Swarm Optimization (PSO), and the target solution is corresponding to the global PSO algorithm optimal solution. Fitness degree of a particle is determined by target function. When a particles fitness degree is high, two conditions must be met: 1. Low energy cost; 2. Low VMs migration cost.

### 4.1 Coding rule

Integer encoding scheme is adopted in EAVMS, and each particle represents a virtual machine placement scheme. \( x_i \) represents the No. i particle, and \( x_i = (x_{i1}, x_{i2}, \ldots, x_{iq}) \). \( x_{id} \) in particle \( x_i \) means the host ID which carry VM \( vm_d \).

### 4.2 Decoding

Each particle’s coding vector \( x_i \) need to be turned to virtual machine deployment matrix \( X_{mn} \). According to deployment matrix, the energy consumption \( E_j \) and the migrating cost \( cost_{ij}^k \) can be calculated.

### 4.3 Fitness

We define \( f(x_i) \) as the Fitness Function of particle \( x_i \).

\[
f(x_i) = \lambda_1 \sum_{j=1}^{m} E_j + \lambda_2 \sum_{i=1}^{q} cost_{ij}^k \tag{12}
\]

In which, \( \sum_{j=1}^{m} E_j \) represents the energy consumption amount of this deployment schema. \( cost_{ij}^k \) represents migration cost of virtual machine \( vm_k \). \( \lambda_1 \) and \( \lambda_2 \) are two positive constants, as a weighting factor to balance energy saving and migration cost.
4.4 Constraints

Because the capacity of each physical host on each dimension is limited, the sum of resource amount all virtual machines running on physical host $h_j$ on each dimension cannot exceed the upper limit of $h_j$. The original PSO algorithm does not fully meet the requirements of our green migration schedule. Not all of the solutions in solution space are feasible solutions. We need a constraint program to measure a solution is or not a viable solution.

Criteria to judge the feasibility of a solution is that there is no one physical host, the total resource amount of virtual machines running on it consumes more than physical host resources. When the solution $x_k$ is a feasible solution, we denoted $\text{allowed}(x_k) = 1$, with the proviso that

$$\forall i, h_j^i \geq \sum_{j=1}^{n} \text{cpu}_p \cdot x_{ij}, \quad p \in \{\text{cpu, mem, net, stor}\}$$  \hspace{1cm} (13)

If the above condition is satisfied, we denoted as $\text{allowed}(x_k) = 0$, which means $x_k$ is an infeasible solution.

For an infeasible solution $x_i$, we define constraint violation degree $\text{Vio}(x_i)$

$$\text{Vio}(x_i) = \sum_{i=1}^{m} \sum_{p \in \{\text{cpu, mem, net, stor}\}} \alpha_p \cdot (\max \left( \sum_{j=1}^{q} (\text{cpu}_p \cdot x_{ij} - r_p^i), 0 \right))$$  \hspace{1cm} (14)

$\text{Vio}(x_i)$ reflects an infeasible solution $x_i$’s degree of exceeding the upper limit of physical host capacity. $\alpha_p$ represents a weight coefficient of resources. When $x_i$ is a feasible solution, $\text{Vio}(x_i) = 0$.

4.5 Particle comparison

When we compare the advantages and disadvantages of the two solutions may encounter three cases: 1. Both solutions are feasible solutions; 2. Both solutions are infeasible solutions; 3. One is feasible solution, the other is infeasible solution. We need to develop a standard to judge the quality of the two solutions.

When particle $x_i$ and $x_j$ are both feasible solution, if $f(x_i) < f(x_j)$, it shows that $x_i$ is better than $x_j$.

When particle $x_i$ and $x_j$ are both non-feasible solution, if $\text{Vio}(x_i) < \text{Vio}(x_j)$, it shows that $x_i$ is better than $x_j$.

When particle $x_i$ is non-feasible solution and $x_j$ is feasible solution, if there are: a. $f(x_i) < f(x_j)$; b. $\text{Ed}(x_i) < \text{Ed}(x_j)$; it shows that $x_i$ is better than $x_j$. $\text{Ed}(x_i)$ is the Euclidean distance from $x_i$ to globally optimal solution. Or that $x_j$ is better than $x_i$.

4.6 Particle location and speed updating

Suppose that in $q$-dimension target searching space, there are $m$ particles in a population, particle $x_i$’s location can be represented as a $q$-dimension vector $x_i = (x_{i1}, x_{i2}, \ldots, x_{id})$, $i = 1, 2, \ldots, m$ and the flying speed is $v_i = (v_{i1}, v_{i2}, \ldots, v_{id})$, the best location that particle
The particle $x_i$ searched is $p_i = (p_{i1}, p_{i2}, \ldots, p_{id})$ and the whole particle swarm search the best location is $g = (g_1, g_2, \ldots, g_d)$.

We define speed updating formula

$$v_i = v_i + c_1 r_1 (p_i - x_i) + c_2 r_2 (g_d - x_i) \tag{15}$$

We define location updating formula

$$x_i = x_i + v_i \tag{16}$$

c_1 and c_2 are learning factors and their value range are nonnegative constant, $r_1$ and $r_2$ are random numbers in $[0, 1]$. The particle’s speed cannot over than predesigned maximum speed $v_{max}$. When the particle’s every move is over, the self-best location and global best location should be updated.

4.7 Algorithm termination condition

Algorithm uses iterative algebraic as termination condition.

5 Simulation Experiment

In order to verify the effectiveness of the algorithm, we use Cloudsim platform to run algorithm. By comparing the state of system before and after algorithm executed, we can see that the system resource utilization is improved, and the total energy consumption is also reduced.

Our experiments were conducted on six kinds Cloudsim platform simulation configuration.

Configuration 1: 6 physical hosts; 30 VMs; 100 tasks;
Configuration 2: 10 physical hosts; 50 VMs; 150 tasks;
Configuration 3: 20 physical hosts; 100 VMs; 300 tasks;
Configuration 4: 30 physical hosts; 150 VMs; 450 tasks;
Configuration 5: 50 physical hosts; 250 VMs; 750 tasks;
Configuration 6: 100 physical hosts; 500 VMs; 1500 tasks.

Fig. 1: Contrast of physical machine open number before and after virtual machine migration
For each configuration, we conducted ten experiments, and recorded number of open state physical hosts, average physical resource utilization and total amount of energy cost before and after migration.

Based on different number of tasks, VMs, and hosts, we can see that number of opened state physical hosts decrease after migration. Especially when the number of tasks, the number of VMs and the number of hosts increase, more physical hosts went into the closed state. Through experiments we conclude that, after the use of EAVMS algorithm, about 10%-20% physical hosts can be turned off.

![Fig. 2: Average utilization rate before and after virtual machine migration](image)

We compared average resource utilization of the physical hosts in cloud before and after migration. Since the algorithm will integrate system resources, close redundant physical hosts, after migrating resource utilization significantly improved in all configurations. The experimental data show that resource utilization increased by 8%-19%.

![Fig. 3: The overall energy consumption before and after virtual machine migration](image)

After the virtual machine migration is complete, the entire energy system is reduced to an average of 4%-11%.

## 6 Conclusion

This article proposed a virtual machine migrating scheduling strategy and the algorithm take the migrating cost and energy consumption into consideration. This solution can lower the migrating cost, reduce energy consumption and improve resource utilization of the whole system.
Acknowledgement

This research work was supported by Program for New Century Excellent Talents in University (NCET100436), and Key Program for Science and Technology Development of Jilin Province (20130206052 GX).

References