Virtual Machine Scheduling to Minimize Resource Wastage in Cloud Computing Environment

Zhu Yahui, Chen Dan

College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China

email: zyh_nuaa@163.com

Keywords: Cloud Computing; Particle Swarm Optimization; Virtual machine scheduling

Abstract. Virtual machine scheduling is the process of selecting the most suitable server in cloud data centers to deploy newly created VMs. The optimal placement is important for improving resource utilization and reducing resource wastage in a cloud computing environment. In this article, we propose an Self-Adaptive Particle Swarm Optimization for the virtual machine scheduling problem. Our algorithm focuses on efficient VM allocation to physical servers in order to minimize the total resource wastage and the number of servers used. Simulation experiments were designed to evaluate the proposed algorithm with performance and scalability. Its solution performance are compared with PSO and GPSO scheduling strategies. The results show that the proposed algorithm is more efficient and effective than the methods we compared to.

Introduction

Cloud computing is the latest development of parallel computing, grid computing and distributed computing [1]. The adoption and deployment of cloud computing platforms have many attractive benefits, such as reliability, quality of service and robustness[2].There are some key technologies that make cloud computing possible. One of the most important is virtualization which has made the task no longer assigned to a physical machine, but made the user's requests to be mapped to a virtual machine, while the virtual machine can be mapped to different physical machines.

Virtual machine scheduling is a process of mapping virtual machines to physical machines. As virtualization is a core technology of cloud computing, virtual machine (VM) scheduling has become a hot topic recently. *U.S. Economist Special Report* shows that the current physical resource utilization of cloud data centers is generally only maintained at 5% to 20% [3]. Therefore, a large number of servers are in idle state. With the use of server virtualization technology, the same server can run multiple virtual machines, and it also can adjust the capacity of virtual machine dynamically [4]. So virtual machine scheduling is an important approach for improving resource utilization in cloud infrastructures. However, the majority of the studies on virtual machine scheduling focus on resource utilization [5], load balancing [6] or QOS(Quality Of Service)[7], few researches consider the balance of resource utilization. If the owners of cloud data centers cannot effectively deal with different types of VM requests, some resources may become overloaded while the others remain underutilized. Eventually, such unbalanced use of resources may result in the unnecessary activation of physical servers [8]. To this regard, choosing the most appropriate target physical machine to place VMs can minimize resource wastage.

In conclusion, although researches have achieved some results in the field of virtual resources scheduling, there are still some shortcomings. For example, the existing works are not fully consider the diversity of cloud resources, the utilization of cloud resources is not balance which lead to the wastage of cloud resources, the utilization of cloud resources is low, and so on. To solve above problems, we proposed a resource utilization balance model and proposed an SA_PSO to solve this model.

Resource Utilization Balance Model

As the infrastructure is fully virtualized in cloud computing environment, all the applications run on VMs and these VMs are placement on physical servers. So the problem of VM scheduling can be considered as the multidimensional vector packing problems. The size of VM can be seen as a d-dimensional vector and each dimension represent one type of the cloud resources(example CPU, memory, bandwidth). For example, Figure shows three VMs deployed on one server where each VM has two dimensions(CPU, memory). We can see that this server will not be able to deploy a new VM because the CPU utilization is too high although there also have a lot of memory resources. This will lead to a waste of these memory resources. So the utilization of each resource should be balanced because unbalanced use of resources may prevent any further VM deployment thus wasting cloud resources. The utilization of different types of cloud resources on each server may vary greatly with different VM scheduling strategy. Our main objective is to balance the utilization of different types of cloud resources.

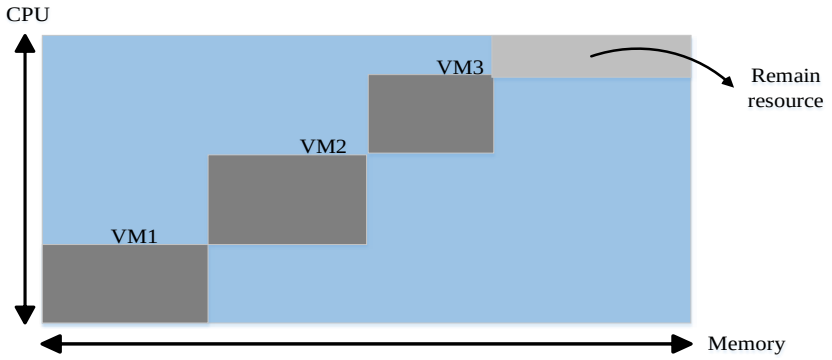

Figure1 three VMs run on a server

Resource Utilization Balance Model

The remaining resources available on each server depend upon the VM placement strategy used while deploying the VMs on physical servers. Therefore, it is always desired that the VMs should be deployed in such a way so as to minimize the total resource wastage. To fully utilize multi-dimensional resources, following equation is used to calculate the potential cost of the wasted resources [9]:

$$W_j = \frac{|R_j^c - R_j^m| + \varepsilon}{U_j^p + U_j^m} \tag{1}$$

Here, W_j represent the resource wastage of jth server, U_j^p and U_j^m represent normalized CPU and memory utilization i.e. ratio of the used resource to the total resource. R_j^c and R_j^m represent the normalized remaining CPU and memory resource i.e. ratio of remaining resource to the total resource. ε is a small positive number and its value is set to 0.0001. The key idea behind the above equation is to balance the remaining resources along different dimensions and to make the effective use of the resources in all dimensions.

Model Formulation

Suppose that there are n VMs that are to be placed on m servers. Let us assume that none of the VMs requires more resources that can be provided by any single server. Let R_j^c and R_j^m be the CPU and memory demand of ith VM, where T_j^c and T_j^m are the CPU and memory thresholds of jth server. A binary variable x_{ij} indicates that ith VM is running on the jth server. The variable y_j indicates whether the server is turned on or off. Since, the objective is to minimize the total resource wastage; the placement problem can be formulated as:

$$\min \sum_{j=1}^{m} W_j = \sum_{j=1}^{m} y_j \frac{|(T_j^c - \sum_{i=1}^{n}(x_{ij}R_j^c)) - (T_j^m - \sum_{i=1}^{n}(x_{ij}R_j^m))| + \varepsilon}{\sum_{i=1}^{n}(x_{ij}R_j^c) + \sum_{i=1}^{n}(x_{ij}R_j^m)} \tag{2}$$

Subject to:

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \forall i \in I \tag{3}$$

$$\sum_{i=1}^{n} x_{ij} R_j^c \le T_j^c \cdot y_j \qquad \forall j \in J \tag{4}$$

$$\sum_{i=1}^{n} x_{ij} R_j^m \le T_j^m \cdot y_j \qquad \forall j \in J \tag{5}$$

$$y_j, x_{ij} \in \{0,1\} \qquad \forall j \in J and \forall i \in I \tag{6}$$

VM Scheduling Strategy Based On SA_PSO

Particle Swarm Optimization

PSO is an intelligent optimization algorithm that is based on swarm intelligence and was first introduced by Eberhart and Kennedy in 1995[10]. PSO is used for solving optimization problems that inspired from the characteristics of population behavior, each particle represents a potential solution and represented by velocity and current position. Each particle corresponds to a fitness value that determined by the fitness function, and the fitness value can judge the merits of the particles. The velocity determines its moving direction and can dynamically adjust with the best experience of their own and other particles. Particles can update their speed and position by individual extreme and global extreme in each iteration. Original PSO updates its velocity and position by the following:

$$V_{id}^{t+1} = \omega V_{id}^{t} + c_1 r_1 (P_{id}^{t} - X_{id}^{t}) + c_2 r_2 (P_{gd}^{t} - X_{id}^{t}) \tag{7}$$

$$X_{id}^{t+1} = X_{id}^{t} + V_{id}^{t+1} \tag{8}$$

Where t is the current number of iterations; V_{id}^{t} represents the speed of ith particle; X_{id}^{t} indicates the position of the ith particle; P_{id}^{t} and P_{gd}^{t} denotes the local best and global best positions of a particle; ω is the inertia weight; c_1 and c_2 are two positive numbers called learning factors; r_1 and r_2 are two random numbers in interval[0,1].

Self-Adaptive Particle Swarm Optimization

Researches and practices show that PSO has some characteristics as fast convergence and good robustness. But it also has some defects. Such as premature convergence, search accuracy and the late iterative efficiency are not high[10]. In order to improve the global convergence of the PSO, we have improved the original PSO as the following:

Inertia weight ω has a great influence on the performance of PSO, it has a tendency to expand the search space and the ability to explore new areas. Analyzing the inertia weight ω of the formula(5) we can find: V_{id}^{t+1} will has a small opportunity to close the global optimal position (P_{gd}^{t})and history optimal of current individual(P_{id}^{t}) when the value of ω is large, therefore V_{id}^{t+1} will deviate from the optimal position and make it possible to explore the better position. It will increase new exploration and beneficial to find the global optimal point as soon as possible that is very beneficial in the early iteration, but disadvantageous in the later iteration because it is not beneficial for the global convergence. When the value of ω is small, contrary to the above situation. The ω was linear decreasing with the increase of iterations in general practice[11]. But the general linear decreasing strategy would have some problems: a) ω will decrease quickly and can't keep a larger value for a long time at the beginning of the PSO. b) It can find the global optimal point in the early iteration, but jump out of the best point because of the value of ω is large thereby reducing the search ability. Therefore, inertia weight ω should not only change with the number of iterations but also change according to the distance between current position and optimal position. In this paper, we proposed a new self-adaptive algorithm in order to adjust the value of ω in order to take into account of both local and global search ability of particles.

Assuming D_i represents the distance of the i[th] particle to the global optimal particle P_{gd}^t, pre-designed two parameters related to distance: the maximum distance $D_{\max}$ and the minimum distance $D_{\min}$. When D_i is greater than the maximum distance $D_{\max}$, the inertia weight ω set as maximum weight $w_{\max}$. When D_i is smaller than the maximum distance $D_{\min}$, the inertia weight ω set as minimum weight $w_{\min}$. When D_i is between in $D_{\min}$ and $D_{\max}$, the inertia weight ω should be nonlinear dynamic adjustment. An adjustment method for self-adaptive inertia weight ω is as shown in Formula (9).

$$w = \begin{cases} w_{\min} & D_i \le D_{\min} \\ w_{\min} + \dfrac{D_i - D_{\min}}{D_{\max} - D_{\min}} e^{-(t/T_{\max})^2} (w_{\max} - w_{\min}) & D_{\min} \le D_i \le D_{\max} \\ w_{\max} & D_i \ge D_{\max} \end{cases} \tag{9}$$

In formula (8), $T_{\max}$ denotes the maximal number of iterations, T represents the current number of iterations. From the values of ω we can see that ω will not only reduce with the number of iterations but also change with the change of the particle's fitness value. So that the ω of optimal particle will has a small value.

A basic PSO is only suitable for solving continuous problems. But VM scheduling problem is a discrete optimization problem. Therefore, we must redefine the parameters and operators of the PSO to solve VM scheduling problem.

1) Encoding Scheme: A two-dimensional encoding scheme is as show in Figure 2 is used for solving the VM scheduling problem. The first dimension of a particle is an n-bit binary vector where n is the total number of physical servers. Every bit in the first dimension is associated with a server. If the bit is '1' it means that the server is in active state and at least one VM is running on it. If the bit is '0' then it means that corresponding server is not used. The second dimension contains the set of subsets comprising the VMs to be placed on those servers[10]

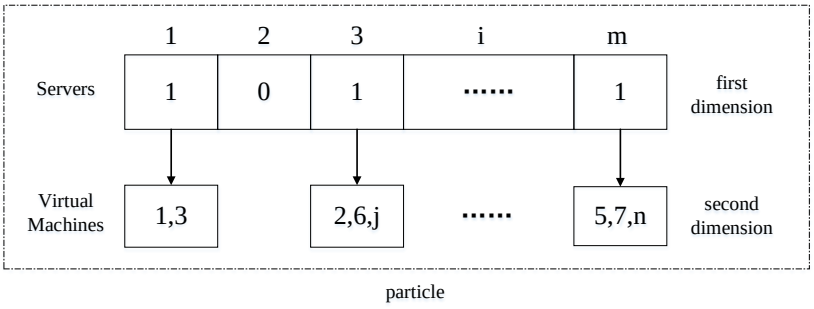

Figure 2 Two dimensional Encoding Scheme

2) Parameters and Operators of SA_PSO Used: The parameter and operators of the SA_PSO used in this work are presented below[10]:

a) particle position(X_i^t):The particle position X_i^t is redefined as an n-bit vector where $X_i^t = (X_{i1}^t, X_{i2}^t, ..., X_{in}^t)$,n represent total number of physical servers in cloud data center. If j^{th} bit is 1, it means that j^{th} server is in use state and if the bit is 0 then the corresponding server is not used.

b) particle velocity(V_i^t):The particle velocity V_i^t is an n-bit binary vector where $V_i^t = (V_{i1}^t, V_{i2}^t, ..., V_{in}^t)$ .The velocity of a particle represents an adjustment decision of the VM scheduling solution. If the bit is 1, then the corresponding server and VM running on that server must be reevaluated, and the value is 0 otherwise.

c) subtraction operator: The subtraction operator was redefined to calculate the difference between the two VM scheduling solutions. The symbol Θ was used to represent the subtraction operator. If a bit of a solution X_i^t is equal to corresponding bit of another solution X_j^t, then the bit

value in the result will be 1, otherwise 0. For example $(1,0,0)\Theta(1,1,0)=(1,0,1)$.

d) addition operator: The addition operator was used to represent the particle velocity update. The velocity update of a particle will be caused by current velocity inertia, local best position and global best position. The symbol $\oplus$ was used to represent the addition operator. Assuming there are n physical servers in a cloud data center, then $(P_1V_1^t \oplus P_2V_2^t \oplus ...P_nV_n^t)$ denotes that a particle updates its velocity by using V_1^t multiply a probability P_1,..., V_n^t multiply a probability P_n. For example, $0.3(1,0,1,0) \oplus 0.7(1,1,0,0)=(1,\#,\#,0)$. If the value of # is less than 0.5, then the value of # will be set as 0, otherwise 1.

e)multiplication operator: The multiplication operator was used to update the particle position. The symbol $\otimes$ was used to represent the multiplication operator. $X_i^t \otimes X_i^{t+1}$ denotes the particle position update operator based on the velocity X_i^{t+1}.

The bit value of the position vector is changed when the corresponding bit of the velocity vector is '1'.If the corresponding bit of the velocity vector is '0' then it shall not be adjusted. For example, $(1,0,1,0) \otimes (1,1,0,0)$,where $(1,0,1,0)$ is the position vector and $(1,1,0,0)$ is velocity vector. The first and second bit values of velocity vector are equal to 1, then the status of the first and second server in the corresponding VM scheduling solution should be updated.

Finally, the velocity and position update operators are as follows:

$$V_i^{t+1} = wV_{id}^t \oplus c_1r_1(P_{id}^t \Theta X_{id}^t) \oplus c_3r_3(P_{gd}^t \Theta X_{id}^t) \quad (14)$$

$$X_i^{t+1} = X_i^t \otimes V_i^{t+1} \quad (15)$$

3)Position Update Strategy In SA_PSO: If the bit value in velocity vector is 0, then corresponding bit value in position vector is adjusted. Corresponding VMs in current scheduling solution will be replaced by the VM set of j^{th} server of current optimal scheduling solution.

VM Scheduling Algorithm Based On SA_PSO

The SA_PSO based VM scheduling algorithm deploy VMs on different servers so that resource utilization can be more balanced. The algorithm start with random initialization of a population, After initialization, all particles traverse in the solution space and move towards to optimal position.

The detailed process of the algorithm is described as follows:

Step1 Generating the initial population.

Step2 Evaluating the fitness function and finding the P_{gd}^t of populations.

Step3 Update the velocity of each particle according to Equation (14).

Step4 Update the position of each particle according to Equation (15). After updating the position, the scheduling solution may not be feasible, i.e. a single VM may be deployed on two different servers. So these repeated VMs must be removes from corresponding server for achieving a feasible solution. But the removing operation may delete the non-repetitive VMs. Therefore, these accidently deleted VMs must be reallocated.

Step5 The reallocating operation adopts the random allocation strategy. While allocating the VMs, one of the active servers will be considered randomly. If none of active servers can allocate this VM, then a new server is turned on and the current VM is deployed on this server.

Step6 Update P_{id}^t by selecting the best position among every particle.

Step7 Return the P_{gd}^t position which represents the global optimal VM placement solution.

Step8 Repeat Step2～Step7 for specify the number of times.

Experiment and analysis

In order to verify the feasibility of the proposed model and algorithm, this paper designe 2 experiment. Experiment 1 to verify the model and algorithm proposed in this paper can balance per physical machine utilization rate of resources, reduce waste of resources; Experiment 2 shows that the model can increase the utilization rate of resources.

The number of physical machines and virtual machine number is always set to the same to support each physical machine running a virtual machine. Physical machine processing capacity of CPU is 2000 MIPS, memory capacity is 4096 MB. Randomly The attribute of the virtual machine is generate randomly, the attribute of CPU is in interval [500,1000] and the attribute of memory is in interval [1000,2000]. The number of particles is 1000. The iterations is 100.

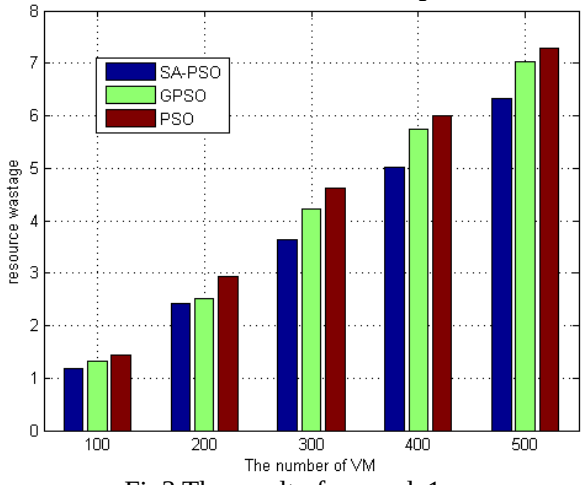

Fig3 The result of example1

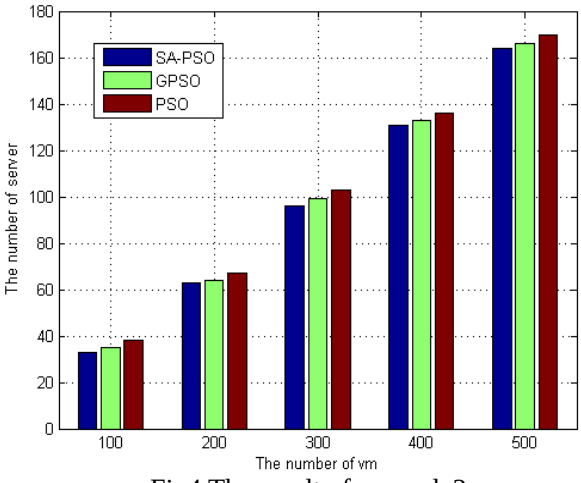

Fig4 The result of example2

We can see SA_PSO can reduce resource wastage from Fig3 and increase resource utilization from Fig4 compared to PSO and GPSO[11].

Conclusion

We propose a resource utilization balance model, in this model, VM scheduling problem is modeled as a bin-packing problem. We further propose an SA_PSO to solve this model. Simulation results show our model can effectively improve the utilization of cloud resources and reduce the resource wastage. But in this paper we do not take into account the energy consumption and in the subsequent research work to consider cloud data center energy consumption.

References

[1] Xu C, Zhao F, Wang Z, et al. Design of cloud computing architecture for power system analysis[C]// TENCON 2013 - 2013 IEEE Region 10 Conference (31194). IEEE, 2013:1 - 4.

[2] M. Randles, D. Lamb, E. Odat, A. Taleb-Bendiab, Distributed redundancy and robustness in complex systems, J. Comput. System Sci. 77 (2) (2011)293–304.

[3] L. Siegele, “Let It Rise: A Special Report on Corporate IT”, The Economist, (2008).

[4] Jung S J, Bae Y M, Soh W. Web Performance Analysis of Open Source Server Virtualization Techniques[J]. International Journal of Multimedia & Ubiquitous Engineering, 2011.

[5] Barros A, Brito A, Brasileiro F, et al. Optimizing resource utilization in private clouds with an opportunistic approach[C]// Cloud Computing and Communications (LatinCloud), 2nd IEEE Latin American Conference on. IEEE, 2013:21-22.

[6] Soni G, Kalra M. A novel approach for load balancing in cloud data center[C]// Advance Computing Conference (IACC), 2014 IEEE International. IEEE, 2014:807-812.

[7] Li H, Ge S, Zhang L. A QoS-based scheduling algorithm for instance-intensive workflows in cloud environment[C]// Control and Decision Conference (2014 CCDC), The 26th Chinese. IEEE, 2014:4094-4099.

[8] Hieu N T, Francesco M D, Yla-Jaaski A. A Virtual Machine Placement Algorithm for Balanced Resource Utilization in Cloud Data Centers[C]// 2014 IEEE 7th International Conference on Cloud Computing (CLOUD). IEEE Computer Society, 2014:474-481.

[9] Kumar D, Raza Z. A PSO Based VM Resource Scheduling Model for Cloud Computing[C]// Computational Intelligence & Communication Technology (CICT), 2015 IEEE International Conference on. IEEE, 2015:213-219.

[10] Wang S, Liu Z, Zheng Z, et al. Particle Swarm Optimization for Energy-Aware Virtual Machine Placement Optimization in Virtualized Data Centers[C]// Parallel and Distributed Systems (ICPADS), 2013 International Conference on. IEEE, 2013:102-109.

[11] Sahal R, Khattab S M, Omara F A. GPSO: An improved search algorithm for resource allocation in cloud databases[C]// Computer Systems and Applications (AICCSA), 2013 ACS International Conference on. IEEE, 2013:1-8.