Research on Resource Scheduling Based on Optimized Particle Swarm Optimization Algorithm in Cloud Computing Environment

Lin Qian\textsuperscript{a}, Lin Wang\textsuperscript{b}, Mingjie Xu\textsuperscript{c}, Jun Yu\textsuperscript{d}, Guangxin Zhu\textsuperscript{e}, Yang Ling\textsuperscript{f}
State Grid Electric Power Research Institute (SGEPRI) Nanjing, China
\textsuperscript{a}qianlin@sgepri.sgcc.com.cn, \textsuperscript{b}wanglin18@sgepri.sgcc.com.cn, \textsuperscript{c}xumingjie@sgepri.sgcc.com.cn, \textsuperscript{d}yujun@sgepri.sgcc.com.cn, \textsuperscript{e}zhuguangxin@sgepri.sgcc.com.cn, \textsuperscript{f}842587420@qq.com

Abstract. Aiming at uneven resource distribution in cloud computing and bad distribution effect this paper proposes a resource scheduling strategy of cloud based on dynamic migration of virtual machine technology. During the migration process, it firstly selected the virtual machine by considering the effect of migration and migration speed then determined the hotspots based on index smooth forecasting with window thinking. It used the particle swarm optimization annealing thinking and long-term optimization goals in the process of migration to search optimal position. By Clouds simulation framework the experiment simulated the appearances of the SLA violation rate, the rate of surplus resources energy and migration times. The simulation results show that this method can achieve good results in multiple optimization goals of improving service quality, improving resource utilization and reducing energy consumption, and effectively improving the performance of cloud computing platform.

1. Introduction
Cloud computing is a new computing and service model developed from distributed computing. The use of virtualization technology to provide flexible and extended storage and computing resources has become a hot topic in the community \cite{1-6}. The most important part of cloud computing is virtual machine dynamic migration and target host search. A lot of research has been done on it at home and abroad. Ref. \cite{7} gives a triggering strategy based on fractal method, predicting resource usage rate and determining overload nodes, and combining target selection strategies for multi-criteria decision making. Ref. \cite{8} gives a virtual machine allocation scheme based on genetic algorithm, and uses the fitness proportional selection method to select genetic operations to prevent rapid convergence to local optimal values and achieve physical node load balancing. In Ref. \cite{9}, in the migration trigger phase, the CPU double threshold triggering is used to save energy while ensuring SLA. In Ref. \cite{10}, the maximum and minimum ant algorithm is used for multi-objective virtual machine placement, taking into account SLA, resource utilization and energy consumption. Ref. \cite{11} uses predictive-based packet genetic algorithm for multi-objective optimization of virtual machine placement, predicting node load changes, and ensuring multiple service target levels of CPU, memory, and I/O through genetic algorithms, resulting in physical node usage and minimizing physical node usage and virtual machine migrations. In Ref. \cite{12}, a performance prediction algorithm based on Singular Value Decomposition Theorem (SVD) is
proposed, and the migration triggering strategy is improved according to the usage rate. This method can cope with sudden load and can reduce unnecessary migration and achieve load balancing. Ref. [13] predicts physical machine and virtual machine load for hotspot judgment and virtual machine migration through exponential smoothing prediction model, which effectively reduces the number of virtual machine migrations.

At present, there are many algorithms for target search in virtual machine migration, such as genetic algorithm, ant colony algorithm and particle swarm algorithm, as well as performance improvement algorithms. Among many algorithms, the particle swarm[14] optimization algorithm does not require the optimization problem to be micro, steerable, continuous, etc., and has the characteristics of fast convergence, simple algorithm, easy implementation, less control parameters, fast calculation speed, etc. Combining the resource load of cloud platform, the dynamic migration process of virtual machine and the advantages and disadvantages of traditional particle swarm, this paper proposes the use of window-based exponential smoothing prediction and double threshold to determine hotspots in the process of virtual machine dynamic migration. Considering the migration effect and virtual machine, the migration speed selects the virtual machine that needs to be migrated, and the particle swarm optimization algorithm with the idea of introducing annealing is used to realize the multi-objective optimization of the cloud computing platform. When the cloud computing platform is built, virtual machine initial placement is also a key factor affecting the dynamic migration of virtual machines on the cloud computing platform. Therefore, the improved particle swarm optimization algorithm realizes the initial allocation of virtual machines in the cluster and the resource scheduling during the running process. Finally, according to the importance of inertia weight in particle swarm search, the inertia weight is determined by experimental comparison for the characteristics of cloud computing platform.

2. Cloud Computing Platform Multi-target Virtual Machine Initial Placement

2.1. Particle Swarm Search Algorithm Based on Simulated Annealing Thinking

1) Particle swarm algorithm modeling

The n virtual machines are numbered and formed into a queue. The mapping relationship between the virtual machines and the m physical nodes is obtained through the search algorithm, and the virtual machine is placed on the corresponding physical node for optimization purposes where the position and velocity of each particle of the group is expressed as shown in formula (1) and (2).

\[ x_i^k = \{x_{i1}^k, x_{i2}^k, \ldots, x_{ij}^k, \ldots, x_{in}^k \} \quad 1 \leq x \leq m - 1 \]  \hspace{1cm} (1)

\[ v_i^k = \{v_{i1}^k, v_{i2}^k, \ldots, v_{ij}^k, \ldots, v_{in}^k \} - v_{\max} \leq v \leq v_{\max} \]  \hspace{1cm} (2)

In formula (1): \( x_{ij}^k \) represents the physical node number of the \( j \)th virtual machine represented by the \( i \)th particle in the \( k \)th iteration. In formula (2): \( v_{ij}^k \) represents the \( j \)th velocity of the \( i \)th particle in the \( k \)th iteration; \( v_{\max} \) is a constant that limits the particle's flight range and avoids excessive offsets that affect the convergence speed of the algorithm. Usually \( v_{\max} \) is defined by experience as 10% to 20% of the problem space.

The speed and position of each particle is updated every iteration through formula (3) and (4).

\[ v_{id}(t + 1) = \omega v_{id}(t) + c_1 r_1 \{P_{id}(t) - x_{id}(t)\} + c_2 r_2 \{P_{gd}(t) - x_{id}(t)\} \]  \hspace{1cm} (3)

\[ x_{id}(t + 1) = x_{id}(t) + v_{id}(t + 1) \]  \hspace{1cm} (4)

where \( i \) is the particle number in the group; \( d \) is the dimension of each particle, the value range is the number of virtual machines; \( r_1, r_2 \) are 0~1 uniformly distributed random numbers; \( c_1, c_2 \) are learning
factors, usually $c_1$ and $c_2$ take equal values from 0 to 4; $\omega$ is the inertia weight; $P_{id}$ is the best position vector for iteration to the individual $i$ at the moment; $P_{gd}$ is the best position vector in the current population.

2) Particle swarm algorithm modeling

The inertia weight in the particle swarm [15] is the key factor affecting the search results and convergence speed of the algorithm. When the inertia weight is large, the particle swarm optimization algorithm has better global search ability, and it can determine more specific solutions with smaller fitness function values in all solutions. When the inertia weight is small, the particle swarm optimization algorithm has better local search and can enable detailed local search of algorithms in a small range. In the whole iterative process of particle swarm optimization, firstly, a large inertia weight is set, and the range of the optimal value is quickly determined in the global range. Then, in the later stage of the algorithm, a small inertia weight is set to make the algorithm search further optimal values in a small range so that the search algorithm can have better search results and faster convergence.

The inertia weight is gradually reduced from a larger value to a smaller value, which can effectively improve the particle swarm search performance. The inertia weight setting is as shown in formula (5).

$$\omega = (\omega_{\text{max}} - \omega_{\text{min}}) \times \tan[m \times (1 - (i/\text{num})^k)] + \omega_{\text{min}}$$  \hspace{1cm} (5)$$

Where the value of $m$ ensures that the value of $\omega$ varies between $[\omega_{\text{min}}, \omega_{\text{max}}]$; the value of $k$ affects the speed of inertia weight reduction; $\text{num}$ is the total number of iterations; $i$ is the number of current iterations. As the number of iterations increases, the inertia weight decreases, and the reduction speed slows down, which is beneficial to change from global search to local search, while ensuring that the algorithm does not converge too quickly.

2.2. Select Target Physical Node Based on Roulette Idea

The roulette idea [16] first counts and categorizes the number of all possible solutions, and assigns 1 proportionally to the number of each solution. Abstractly, all kinds of solutions are assigned to a disc by proportional proportion. The number of solutions is large, and the area occupied by the disk is large, and vice versa. In the selection solution phase, a pointer of 0 to 1 is used to simulate the pointer in the middle of the wheel, the pointer is rotated, and the last pointed solution is the final selected output solution.

The idea of roulette is to randomly select a set of solutions in multiple sets of solutions, but it is not completely random, but consider the number of possible solutions. Adding roulette ideas to a search algorithm with multiple sets of possible solutions can select the current optimal solution within a certain probability, and also have a certain probability to select the current suboptimal but the optimal solution relative to the long-term running process.

The particle swarm optimization algorithm that introduces roulette ideas is to obtain the optimal solution for virtual machine placement by preserving the possible $N$ sets of solutions found in the particle swarm iteration process. According to the $N$ group, an $N \times n$ matrix is constructed, and the physical node labels of the virtual machines that need to be migrated are obtained. According to the different proportions of the different solutions in the solution set, the probability and statistics method is used to count the number of labels in each possible solution, and the expected and standard deviations of the multiple possible solutions corresponding to the migration virtual machine are calculated. If the absolute value of a solution and the expected difference is greater than the standard deviation, then this solution is an impossible solution and is not considered when choosing to place a virtual machine. Then the physical nodes are selected with random probability in the optimal solution and the suboptimal solution, so that the particle swarm search results are not only optimized for the current problem, but also the long-term goal optimization.
3. Virtual Machine Dynamic Migration Process

3.1. Load Forecasting Determines Physical Hotspots

Temporary high node utilization is a common phenomenon in cloud computing platforms. If this is a hotspot, factors will lead to unnecessary migration and waste of system resources. Introducing the window idea and using the time series based predictive exponential smoothing model to determine hotspots can effectively avoid hotspots misjudgment and reduce the number of migrations. When the CPU utilization warning value in the window exceeds the threshold range more than the set number of times and the current CPU utilization exceeds the threshold range, the smooth exponential model is used to predict whether the next moment exceeds the threshold. If the predicted value exceeds the threshold range, the migration is triggered, otherwise the number of window warnings is increased by 1 and the migration is not triggered.

Based on historical values, the CPU usage at the next moment is predicted. The load at time $t+1$ is as shown in (6).

$$x_{t+1} = \alpha x_t + \alpha^2 x_{t-1} + \cdots + \alpha^{n+1} x_{t-n} + \alpha_t$$

Where $\alpha$ is the smoothing index prediction parameter, which is a positive number less than 1, indicating the degree of influence on the CPU usage value in the next time window; $\alpha_t$ is a normal distribution random variable to ensure that the predicted value has a certain randomness.

3.2. Select a Migration Virtual Machine in the Hotspots

1) The virtual machine selection policy combined with the CPU usage and the virtual machine memory size is used to ensure the migration quality when the virtual machine is migrated. The evaluation function is as shown in formula (7).

$$Q = \frac{U_{CPU}}{U_{ram}}$$

Where $U_{CPU}$ is the usage of the virtual machine CPU, $U_{ram}$ is the memory of the virtual machine. When the CPU usage is high and the memory is small, the $Q$ value is large. Migrating this virtual machine can eliminate hotspots faster, the CPU scene data is less, and the data in the memory is less, which can shorten the migration time.

2) The virtual machine migration process requires a large amount of real-time data to be replicated. The virtual machine evaluation formula was given as shown in formula (8).

$$V = \frac{U_{FCPU} - U_{CPU}}{U_{FCPU}} \times \frac{U_{FR} - U_{ram}}{U_{FR}} \times \frac{U_{FS} - U_{storage}}{U_{FS}}$$

3) Considering the migration time and migration effect, it can effectively reduce node utilization and reduce the number of migrations in a short migration time, thus ensuring efficient and stable operation of cloud computing, as shown in formula (9).

$$E = a \times Q + b \times V$$

where $a$, $b$ are weights and they are set to equal values, that is, selecting the virtual machine to be migrated is neither the shortest migration time nor the best migration effect, so that both aspects can be considered at the same time, and can effectively eliminate cluster hotspots in a short period of time.
3.3. Search Virtual Machine Placement Target
If the number of virtual machines to be migrated is small, the physical machine is searched and compared sequentially, the optimal fitness value is found, and the virtual machine is deployed to the physical node. If the number of virtual machines that need to be migrated is large, the physical machine is searched by the particle swarm algorithm used in the initial deployment. The pseudo-code of the cloud computing platform virtual machine dynamic migration process is as follows:

1. Initializing a physical node queue according to the detected value;
2. while(Physical node queue > 0) do
   3. if Using formula (9), the predicted value exceeds the threshold && Migration queue is not empty
      4. if Predicted value exceeds minimum value
         5. Move out all virtual machines of the node to join the migration queue;
      6. else Predicted value exceeds maximum value
         7. Select a virtual machine according to formula (12) and join the migration queue.
      8. end if
   9. if Migration queue length > 1/5 of the number of online hosts
      10. Calling the particle swarm algorithm function to search for the migration map;
      11. else Traverse physical machines and all virtual machines to be migrated
      12. Use Greedy thinking to find the target node;
      13. end if
   14. Migrate the virtual machine to the corresponding node by migration mapping;
   15. if There is an overload node after placement
      16. Start the physical node to join the physical node queue and search again.
   17. end if
18. while end;
19. End

In the virtual machine migration process, the cluster monitoring module monitors the running status of physical machines and virtual machines, including resource utilization, user creation, system-triggered requests for migrating virtual machines, and physical node energy consumption. When the node utilization is greater than the highest threshold, the appropriate virtual machines is selected to move out and place it on other nodes to reduce the load. When the node utilization is less than the minimum threshold, all the virtual machines on the node are moved out, and the node is shut down to achieve the energy saving goal. The search algorithm is selected by the migration queue and the ratio of physical machines that can be placed, and the cloud computing platform resource scheduling is completed according to the obtained solution migration virtual machine.

4. Experimental Results
Experiment with the CloudSim simulation cloud computing [17] platform, from the SLA violation rate, resource consumption and energy consumption to the sequential placement of non-migration methods (FIFS), greedy algorithms, particle swarm optimization (PSO) and multi-objective particle swarms with simulated annealing ideas Long-term load algorithm (MTPSO) for comparative analysis. In the simulation experiment, 400 different physical nodes of different types were constructed, and 263 virtual machines of four different configurations were created, and 5 real-time cloud computing tasks were run on each virtual machine. A one-week simulation experiment was carried out on the above algorithms to verify the long-term performance of the algorithm in the cloud computing platform.

The number of migrations [18] of the greedy algorithm, the improved particle swarm algorithm, and the standard particle swarm algorithm for the seven-day cloud computing platform is shown in Table 1.
**Table 1. Virtual machine migration**

|       | 1d  | 2d  | 3d  | 4d  | 5d  | 6d  | 7d  |
|-------|-----|-----|-----|-----|-----|-----|-----|
| MTPSO | 275 | 301 | 307 | 310 | 313 | 315 | 320 |
| PSO   | 265 | 286 | 300 | 308 | 315 | 324 | 331 |
| Greedy algorithms | 12943 | 22897 | 30975 | 38405 | 46256 | 54331 | 62447 |

The PSO in Table 1 has the optimal fitness function at the beginning, so the number of migrations is small, but in the later stage of simulation, the MTPSO based on long-term load shows better optimization effect, the migration times are less than PSO, and the greedy algorithm has the most migrations.

In summary, the experimental analysis shows that in the cloud environment, for the three conflicting goals, the weight combination method is used to achieve the SLA violation rate, energy consumption and resource utilization balance optimization, and the annealing algorithm is used to optimize the inertia weight. At the same time, the Roulette idea is introduced, which makes the multi-objective particle swarm optimization algorithm based on annealing idea have better performance than traditional particle swarm optimization algorithm in long-term optimization.

5. Conclusion

This paper comprehensively considers the SLA, resource utilization, energy consumption and migration times in cloud computing platform resource scheduling, and gives a multi-objective particle swarm long-term optimization algorithm based on annealing theory, and optimizes the initial placement and operation of virtual machines in the initial stage of cloud computing platform establishment. During the dynamic migration process of the virtual machine, the SLA of the system is guaranteed and the balance of resource utilization, energy consumption and migration times is achieved. Simulation experiments show that the proposed method can achieve better results in multiple optimization objectives of ensuring service quality, improving resource utilization and reducing energy consumption, and effectively improving the performance of the cloud computing platform.

**Acknowledgments**

This work was financially supported by the State Grid Corporation of Science and Technology (WBS number: 521104170019).

**References**

[1] Chang D C, Xu G C. Live Virtual Machine Migration Algorithms [J]. Application Research of Computers, 2013, 30 (4): 971 - 976.
[2] Zuo L Y, Cao Z B. Review of Scheduling Research in Cloud Computing [J]. Application Research of Computers, 2012, 29 (11): 4023 - 4027.
[3] Lin W W, Qi S Y. Survey of Resource Scheduling in Cloud Computing [J]. Computer Science, 2012, 39(10): 1 - 6.
[4] Sadhasivam S, Nagaveni N, Jayarani R, et al. Design and Implementation of an Efficient Two-level Scheduler for Cloud Computing Environment [C] //Proc of International Conference on Advances in Recent. Technologies in Communication and Computing, 2009: 884 - 886.
[5] Garg S K, Yeo C S, Anandasivam A, et al. Environment-conscious Scheduling of HPC Applications on Distributed Cloud-oriented Data Centers [J]. Journal of Parallel and Distributed Computing, 2011, 71.
[6] Zhu J Y, Xiao D, Wang F. Multi-dimensional QoS Constrained Scheduling Mechanism Based on Load Balancing for Cloud Computing [J]. Computer Engineering and Applications, 2013, 49 (9): 85 - 89.
[7] Shi Y B. A Load Balancing Algorithm Based on Dynamic Migration of Virtual Machines in Cloud Computing Environment [D]. Shanghai: Fudan University, 2011.
[8] Wang S. Research on Cloud Computing Resource Load Balancing Scheduling Algorithm Based
on Minimum Migration Cost [D]. Chengdu: University of Electronic Science and Technology, 2013.

[9] Xia G M, Zeng J A. Stochastic Particle Swarm Optimization Algorithm Based on the Genetic Algorithm of Roulette Wheel Selection [J]. Computer Science and Engineering, 2007, 29 (6): 51 - 54.

[10] Ma F. Research on Virtual Machine Placement and Live Migration in Cloud Data Center [D]. Beijing: Beijing Jiaotong University, 2013.

[11] Li Q, Hao Q F and Xiao L M, et al. Adaptive Management and Multiple Objective Optimization for Virtual Machine Placement in Cloud Computing [J]. Chinese Journal of Computer, 2011, 34 (12): 2253 - 2264.

[12] Li Z H. Research on Cloud Computing Virtual Machine Cluster Deployment and Dynamic Migration Technology [D]. Hangzhou: Zhejiang University, 2013.

[13] Wei L, Huang T and Chen J Y, et al. Workload Prediction-based Algorithm for Consolidation of Virtual Machines [J]. Journal of Electronics & Information Technology, 2013, 35 (6): 1271 - 1276.

[14] Lin L J. Research and Application of Simulated Annealing Particle Swarm Optimization Algorithm [D]. Jinan: Shandong Normal University, 2010.

[15] Eberhart R C, Shi Y. Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization[C] / /Proc of Congress on Evolutionary Computation. 2000: 84-88.

[16] Wang F, Qiu Y H. A Hybrid Particle Swarm Algorithm with Roulette Selection Operator [J]. Journal of Southwest China Normal University: (Natural Science Edition), 2006, 31 (3): 93 - 96.

[17] Wang D W, Liu X M. A Resource Scheduling for Cloud Computing Platform of Power System Simulation Based on Dynamic Migration of Virtual Machine [J]. Automation of Electric Power Systems, 2015, 39 (12): 97 - 105.

[18] Fang Shuo, Kanagave R, Lee B S, et al. Power-efficient Virtual Machine Placement and Migration in Data Centers[C] / /Proc of Green Computing and Communications, IEEE and Internet of Things ( iThings/CPSCom), IEEE International Conference on and IEEE Cyber, Physical and Social Computing. 2013: 14.