IEEE ITOEC(ISSN:2693-289X)

Cloud computing resource node allocation algorithm based on load balancing strategy

Ye Bo
State Grid Xinjiang Electric Power Research Institute, Urumqi, China
bby90031@163.com

2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC) | 978-1-6654-3185-9/22/$31.00 ©2022 IEEE | DOI: 10.1109/ITOEC53115.2022.9734399

Abstract—**At present, cloud computing has developed into a high-performance computing service model integrating the technical advantages of distributed computing, utility computing, load balancing, parallel computing, network storage, hot backup redundancy and virtualization. However, due to the lack of effective algorithm at the beginning of DAG task scheduling to ensure the load balance of resource node allocation during task scheduling, this paper proposes a cloud computing resource node allocation algorithm (NA-LB) based on load balancing strategy. The algorithm records the changes of virtual machine operating parameters in the cloud computing resource pool as vector values of spatial vectors, Therefore, the mathematical calculation method of load balance is given and used as the load balance index to measure the allocation of resource nodes. At the same time, the task replication strategy is used to further optimize the task resource node allocation process. The experimental results show that NA-LB algorithm can effectively shorten the DAG data processing time and show good load balance.**

Keywords—Cloud computing; DAG task scheduling; resource node allocation; Load balance; task replication strategy

I. INTRODUCTION

Scheduling in cloud computing is generally divided into two parts: Resource Scheduling: resource scheduling refers to the reasonable and effective management and use of physical resources; Task scheduling: reasonably allocate tasks to appropriate computing resources for execution A normal user service process is: the user submits a task to the cloud, the task scheduler assigns the task to appropriate computing resources for execution, and then feeds back the results to the user after the task is completed Among them, task allocation is particularly important. A reasonable and efficient scheduling strategy can greatly improve the performance of cloud computing system The following focuses on task scheduling. The service delivery model in cloud computing is divided into four parts: users submit tasks; The task manager divides it into multiple subtasks; The task scheduler establishes a mapping relationship between subtasks and physical resources through scheduling technology; After completing the task, summarize and feed back to the user From this, we can see that from the mission to the completion, scheduling technology is the priority among priorities in the whole task execution process, which affects the efficiency of the whole system, the quality of user service, the load balancing of the system, and the energy consumption of the system. Therefore, a scheduling technology suitable for cloud computing environment is very important.

In order to prolong the service life of devices, it is necessary to balance the power consumption by scheduling computing to devices with higher power and computing power, that is, improving task scheduling can effectively reduce the power consumption cost of cloud computing Therefore, task scheduling and processing allocation have become the key problems to be solved in cloud computing. Dag task scheduling has its own characteristics in terms of scheduling strategy and problem solving. Among them, priority table scheduling algorithm is widely used because of its excellent performance.

II. DEFINE CLUSTER LOAD BALANCING INDICATORS

A. Mathematical model of computing nodes

The working state of the virtual machine node of the system can be expressed by the parameter vector. A virtual machine has k States, expressed as $\alpha = (\partial_1, \partial_2, \cdots, \partial_k)$.

When computing data is allocated to some computing units, their parameter values will change accordingly. At this time, the change of the current working state of the virtual machine can be seen according to the parameter change. We assume that the number of virtual machines in the cloud computing resource pool is M , which can be displayed according to the state of each parameter of the virtual machine during task processing, During task processing, set the proportion of all parameters in the cloud computing system to $a_1, a_2, \cdots, a_k$ respectively, so that $a_1 + a_2 + \cdots + a_k = 1$, then the running state of all virtual machines is

$$U = \{x_{i1}, x_{i2},, x_{ik} \mid 1 \le i \le M, 0 \le x_{ik} \le a_k\} \tag{1}$$

Here, the jth parameter for the virtual machine i is expressed as x_{ij} . In the working state of the virtual machine at this time, the expression of each state parameter is

$$X_1 = \frac{1}{M}\sum_{i=1}^{M} x_{i1}, X_2 = \frac{1}{M}\sum_{i=1}^{M} x_{i2}, \cdots, X_k = \frac{1}{M}\sum_{i=1}^{M} x_{ik} \tag{2}$$

$G(X_1, X_2, \cdots, X_k)$ is used to represent the center of gravity coordinates of the space vector composed of virtual machines in the computing node, and the load balance of the whole system is judged by him. We can abstract all servers into each vector in the parameter vector space, The load balance of the system is judged by the overall distribution of each point in the vector space. If more than 90% of the virtual machine computing points are distributed in the surrounding space centered on the center of gravity, we say that the system has good load balance; If the nodes are scattered, it is judged that the load balance of the system is not good.

B. Define system load balance index

When we consider the load balance of the cloud computing system, we always use the normalization method, that is, we average the distance between all virtual computing nodes in the vector space and the center of gravity of the system, and then normalize the results. We define this normalized value as the load balance of the system, expressed as

$$LB = \frac{\sum_{i=1}^{M}\sqrt{\sum_{j=1}^{k}\left(x_{ij}-X_j\right)^2}}{\frac{M\sqrt{a_1{}^2+a_2{}^2+\cdots+a_k{}^2}}{2}} \tag{3}$$

Here, take the standard deviation for the distance from all virtual computing nodes to the center of gravity, and use $\sqrt{a_1{}^2+a_2{}^2+\cdots+a_k{}^2}/2$ to represent the maximum value of the standard deviation. Obviously, the load balance of cloud computing system is the worst at this time.

When measuring the load balance of the cloud computing system, we use LB to represent the real value of the system load balance degree when the computing data is distributed to a computing node at a certain time. Therefore, we can see that LB can show the load balance index of the cloud computing system in real time. Of course, the value range of LB is between 0 and 1, And we can know that the smaller the value of LB, the better the load balance of the system at this moment. We take it as a key reference for selecting virtual machine in data calculation, so that the system can show good load balance in task processing.

III. DAG Task Modeling

When presenting the data with correlation, we use DAG diagram to describe it in detail. The points in the diagram are used to represent a single calculation data, and the edge in the diagram is used to determine which calculation task can be performed first.

Here, $G=\{V,E,W,C\}$ is used to define a data group with four related elements. The meaning of each element is:

1) Here, $G=\{v_1,v_2,\cdots,v_i,\cdots,v_N\}$ is used to describe all task sets in DAG diagram. v_i is each calculation task. The number of all calculation characters is described by sum $N=|V|$, and $1\le i\le N$;

2) The sequential set between all calculated data (v_i,v_j) is represented by $E=\{e_{i,j}\mid v_i,v_j\in V\}\subseteq V\times V$, and the sequential edges in DAG are described by $e_{i,j}$;

3) Here we use W as matrix $N\times M$. That is, the set of resource consumption when N calculation data are processed on M calculation resources. We use $w(v_i,p_m)$ as the resource consumption of calculation data v_i on calculation resource p_m, then the average calculation resource consumption of all calculation data is

$$\overline{w(v_i)}=\frac{1}{M}\sum_{m=1}^{M}w(v_i,p_m) \tag{4}$$

Here, we define the total resource consumption during the transfer between calculation data as set C. when two calculation data with direct correlation are processed on the same calculation resource, the resource consumption during the transfer between calculation data is 0.

Sometimes, in order to describe data task processing more intuitively, it is necessary to define entry computing resources, exit computing resources, precursor resource set of computing resources, successor resource set of computing resources, data start processing time and earliest processing completion time in DAG task diagram.

Definition 1: the entry calculation data has no precursor calculation data, and the exit calculation data has no subsequent calculation data. When there are more than one entry calculation data in the DAG diagram, we can add a false entry with calculation resource consumption of 0, and point to the false entry with the edge with weight of 0. The method is the same when dealing with the exit point.

In the DAG diagram, we define the set of direct precursor calculation data of calculation data v_i as the parent resource set of v_i, representing $pred(v_i)$; The direct subsequent calculation data set of calculation data v_i is the sub resource set of v_i, which represents $inhe(v_i)$.

Generally, the start processing time of computing data v_i on computing resource p_m will be subject to two key factors:

1) The start processing time of the calculation data v_i on the calculation resource p_m. The start processing temporary time on the computing resource p_m is the time interval from the time when the computing data v_i is divided into the computing resource p_m to the time when the computing data v_i is just to be processed, which is recorded as $usa(p_m)$. If the computing data starts to be processed just after the allocation, then

$$usa(p_m)=ct(v_t,p_m) \tag{5}$$

Here, $ct(v_t,p_m)$ is used to represent the processing completion time of the calculation data v_t on p_m.

The time when the direct precursor data processing result of the calculation data v_i is sent to the calculation resource p_m. If the calculation data v_i starts to be processed, then all the processing results of the direct precursor calculation data must be received by the calculation resource p_m, then we can calculate the processing time of the data v_i on the calculation resource p_m, that is

$$st(v_i, p_m) = \max\left[usa(p_m), \max\left(ct(v_j, p_x) + C(e_{i,j})\right)\right] \tag{6}$$

Here, $C(e_{i,j})$ is used to represent the time when the processing result is sent to the computing resource. We take the cumulative value of the processing time and the processing time of the computing data v_i on the computing resource p_m as the processing completion time $ct(v_i, p_m)$ of the computing data v_i on the computing resource p_m, that is

$$ct(v_i, p_m) = st(v_i, p_m) + w(v_i, p_m) \tag{7}$$

The DAG diagram of Fig. 1 is a correlation calculation data group, in which there are 8 data tasks, v_1 is the entry calculation data and v_8 is the exit calculation data. The directed edge in the figure indicates the dependency between data points, and the number in the figure indicates the resource communication consumption between data; The data in the dotted line is the calculation data of the same level, that is, they are the direct follow-up data of the precursor data.

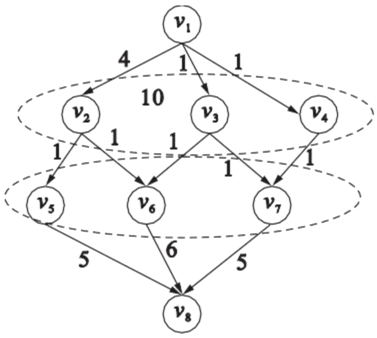

Fig. 1. DAG diagram of correlation data

IV. Analysis Of NA-LB Algorithm

A. Task priority calculation

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc, and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

The flow chart of NA-LB algorithm setting priority for tasks can achieve fairness and efficiency in task scheduling. This paper modifies the priority calculation formula of table scheduling algorithm HEFT, and adds two key factors: the output value of tasks and the output communication value of tasks. Here, we use the number of direct successor tasks of the task to represent the output of the calculated data v_i, that is $H(v_i)$. After analysis, we come to a rule that the larger the output value of the data, the greater the impact on the subsequent data in processing. When the data is processed first, The concurrent processing of subsequent data will be better. The export transfer value of calculation data v_i is defined as the sum of all export transfer times of the calculation data, then

$$B(v_i) = \sum_{v_j \in inhe(v_i)} C(e_{i,j}) \tag{8}$$

Through a lot of analysis, we found that the greater the resource consumption $B(v_i)$ of data export transmission, the greater the impact on the subsequent data processing. When we give priority to it, we can greatly reduce the waiting time of subsequent data. Now we comprehensively consider the above two key factors and optimize the priority calculation method, that is

$$PR(v_i) = \left(\alpha H(v_i) + \beta B(v_i)\right) SL(v_i) \tag{9}$$

Here, we set α And β as dynamic adjustment factor; And a priority $SL(v_i)$ is set in advance for data processing, that is

$$SL(v_i) = \begin{cases} \overline{w(v_i)} + \max\limits_{v_j \in inhe(v_i)} \left[C(e_{i,j}) + SL(v_i)\right] & (1 \le i \le N) \\ 0 & (i = N) \end{cases} \tag{10}$$

B. Select virtual machine computing resources

Here, we process the data according to the weight. The goal is to obtain the shortest time for all data to be processed. Taking the load balancing index and the processing time of the critical path behind the currently processing data v_i as the reference weight, we choose the computing resources with small data weight processing weight, Allocate the data to it for processing. The calculated resource weight value of data v_i of $sw_{i,m}$ is

$$sw_{i,m} = \min_{1 \le m \le M} \left(\left(st(v_i, p_m) + INHE\_KEY(v_i)\right) LB\right) \tag{11}$$

LB is defined here as the load balancing index;

$INHE\_KEY(v_i) = HE\_KEY(v_i) + C(e_{i,j}) + w(v_i, p_m)$, $HE\_KEY(v_i)$ is the key data set of v_i

C. Data distribution and data replication

Now let's start data processing. First, we scan the DAG data relation diagram in depth to calculate the critical path. We assign the highest priority to the data on the critical path in the data of the same level, and sort the non critical data in descending order according to the $PR(v_i)$ value. When multiple data $PR(v_i)$ are equal, Then the data task with a large amount of subsequent data will be selected for priority processing

Now, the data with high priority will be selected from the data task column and assigned to the computing resource with the largest value of $sw_{i,m}$. If there are more than two identical data, we select the computing resource with the smallest load balancing index value from the computing resources with the value of $sw_{i,m}$, and then assign the data task to him for processing.

Now, we want to continue to reduce the time after all data processing is completed, so as to greatly improve the task processing efficiency of the cloud computing cluster. We continue to improve the data allocation processing process on the basis of the mature applied task replication technology. Now, according to the priority size, the direct precursor data of data v_i is re queued and collected as

$$pred(vi) = \{v_{i,1}, v_{i,2}, \cdots, v_{i,n}\} \quad (12)$$

If data $v_{i,1}$ meets the following conditions

$$ct(v_{i,1}, p_x) + C(e_{i,i,1}) = \max_{v_{i,y} \in pred(v_i)} (ct(v_{i,y}, p_x) + C(e_{i,i,y})),$$

We record $v_{i,1}$ as the key precursor data of data v_i . After the above analysis, if you want to catch up with the earliest processing time of data, you can copy the key precursor data $v_{i,1}$ of current data v_i to the target computing resource p_m , Define another definition restriction: the cumulative value of the processing time when the data $v_{i,1}$ is scheduled to the computing resource p_x and the time $C(e_{i,1,i})$ when the processing result is sent to the computing resource p_m is greater than the processing time when the data $v_{i,1}$ is copied to the computing resource p_m , then

$$ct(v_{i,1}, p_m) < ct(v_{i,1}, p_x) + C(e_{i,1,i}) \quad (13)$$

Now let's assume that the conditional expression (12) conflicts, then we should continue to select the high priority precursor data $v_{i,2}$ in the direct precursor data queue of data v_i for consideration, It can be allocated to the zero task timeline $fts(v_i, p_m)$ of the computing resource where the data v_i is located for processing. Now we define a definition restriction: the length of the zero task timeline $fts(v_i, p_m)$ is greater than the resource consumption $w(v_{i,2}, p_m)$ when the data $v_{i,2}$ is processed on the computing resource p_m , then

$$fts(v_i, p_m) > w(v_{i,2}, p_m) \quad (14)$$

Now let's assume that data v_i has multiple precursor data tasks, then we will consider data $v_{i,3}$ in sequence. At this time, the zero task timeline is

$$f = fts(v_i, p_m) - w(v_{i,2}, p_m) \quad (15)$$

The restrictions here are

$$fts(v_i, p_m) - w(v_{i,2}, p_m) \geq w(v_{i,3}, p_m) \quad (16)$$

Now, we will continue to consider the following precursor data until the termination constraint is not met when a data task is replicated. At this time, the processing time of any precursor data to be allocated will be greater than the value of the zero task timeline of computing resources. The reason for the termination of the replication restriction condition is

$$fts(v_i, p_m) - \sum_{j=2}^{r} w(v_{i,j}, p_m) < w(v_{i,r+1}, p_m), (2 \leq r \leq n) \quad (17)$$

V. Conclusion

This paper proposes a cloud computing resource node allocation algorithm (NA-LB) based on load balancing strategy. Through a large number of experiments and applications, the algorithm shows excellent load balancing and greatly improves the data processing efficiency of cloud computing cluster. Especially for large-scale cloud computing task processing, it will have a broader application. Although the power consumption of cloud computing can be reduced by optimizing the scheduling scheme, the computing process must be uploaded to the cloud every service delivery in cloud computing services. Even if the scheduling scheme is improved, the problem of network delay can not be effectively solved.

In order to solve the problem of network delay in cloud computing, edge computing has emerged Edge computing is different from cloud computing. Intuitively, edge computing is service delivery at the "edge" of the network, and the data is calculated and stored near users Compared with cloud computing, it is closer to users. The most intuitive result is to reduce network delay, network bandwidth requirements, transmission delay during data computing or storage, and effectively reduce the loss speed of physical devices In addition, compared with cloud computing, edge computing can migrate computing and communication overhead from nodes with limited battery or power supply to edge nodes with large power resources.

References

[1] Efficient distribution of requests in federated cloud computing environments utilizing statistical multiplexing[J] . Moslem Habibi,MohammadAmin Fazli,Ali Movaghar. Future Generation Computer Systems . 2018.

[2] Task Scheduling in Cloud Computing using Lion Optimization Algorithm[J] . Nora Almezeini,Alaaeldin Hafez. International Journal of Advanced Computer Scienc . 2017.

[3] A WOA-Based Optimization Approach for Task Scheduling in Cloud Computing Systems[J] . Chen Xuan,Cheng Long,Liu Cong,Liu

Qingzhi,Liu Jinwei,Mao Ying,Murphy John. IEEE Systems Journal . 2020.

[4] Load balancing task scheduling based on genetic algorithm in cloud computing. WANG T,LIU Z,CHEN Y,et al. IEEE 12th International Conference on Dependable,Autonomic and Secure Computing （DASC） . 2014.

[5] Multi-objective list scheduling of workflow applications in distributed computing infrastructures[J] . Hamid Mohammadi Fard,Radu Prodan,Thomas Fahringer. Journal of Parallel and Distributed Computing . 2014 (3).

[6] Large-scale compute-intensive analysis via a combined in-situ and co-scheduling workflow approach. Sewell C,Heitmann K,Finkel H,et al. International Conference for Hingh Performance Computing . 2017.

[7] A view of cloud computing[J] . Michael Armbrust,Armando Fox,Rean Griffith,Anthony D. Joseph,Randy Katz,Andy Konwinski,Gunho Lee,David Patterson,Ariel Rabkin,Ion Stoica,Matei Zaharia. Communications of the ACM . 2010 (4).

[8] Energy efficient utilization of resources in cloud computing systems[J] . Young Lee,Albert Zomaya. The Journal of Supercomputing . 2012 (2).