STUDY OF TASK SCHEDULING STRATEGY BASED ON TRUSTWORTHINESS

Jun QIN[1,2], Yanyan SONG[1] and Ping ZONG[2,3]

[1]Communication University of China, Nanjing, China
[2]Nanjing University of Posts and Telecommunications, China
[3]Nanjing University of Science and Technology Zijin College, China

ABSTRACT

MapReduce is a distributed computing model for cloud computing to process massive data. It simplifies the writing of distributed parallel programs. For the fault-tolerant technology in the MapReduce programming model, tasks may be allocated to nodes with low reliability. It causes the task to be reexecuted, wasting time and resources. This paper proposes a reliability task scheduling strategy with a failure recovery mechanism, evaluates the trustworthiness of resource nodes in the cloud environment and builds a trustworthiness model. By using the simulation platform CloudSim, the stability of the task scheduling algorithm and scheduling model are verified in this paper.

KEYWORDS

Cloud Environment, Failure Recovery Mechanism, Task Scheduling Algorithm

1. INTRODUCTION

Cloud computing is the commercial development of distributed computing, parallel computing, and grid computing [1]. Cloud computing realizes the sharing of resources such as computing facilities, storage devices, and applications through the Internet, and provides various services such as computing and storage for different users. The application areas of cloud computing cover storage and backup, content push, e-commerce, high-performance computing, media services, search engines and other fields. The large-scale data collection, analysis, and in-depth utilization of cloud computing in these fields will gain more knowledge and benefits. The processing of big data has become an inevitable trend and important direction of development. The traditional programming model is difficult to meet the requirements of large-scale data processing. The MapReduce programming model proposed by Google can solve data-intensive problems. The MapReduce programming model plays a vital role in cloud computing [2].

MapReduce is a programming paradigm. It can use hundreds or thousands of servers in a cluster environment to achieve strong scalability [3]. Using functional programming ideas, a calculation is divided into two calculation processes: map and reduce. MapReduce divides a large computing task into multiple small computing tasks, and then assigns each small computing task to each computing node in the cluster, and keeps track of the progress of each computing node to determine whether to re-execute the task, and finally collects the calculation results on each node and output. MapReduce is a computing architecture design, which is based on the master-slave structure design of JobTracker and TaskTracker. JobTracker is responsible for specific task division and task monitoring and determines whether a task needs to be rolled back. TaskTracker is responsible for the execution of specific tasks, acquires data for each task assigned to itself, maintains communication with JobTracker to report its own status, and outputs calculation results and other

calculation processes. If the TaskTracker node does not respond within the specified time, the node will be marked as invalid. All tasks completed on the failed node will be set to an unexecuted state and assigned to be executed again on other TaskTracker nodes. This fault-tolerant mechanism is expensive, poor efficiency, and wastes a lot of resources. In view of the shortcomings of the MapReduce fault-tolerant mechanism, the system regards the failure of the TaskTracker node as a normal state and no special processing is required in the MapReduce parallel programming model [4]. So the task of the failed node is only scheduled to be executed on other nodes. Since all Map tasks completed on the failed node are stored on the local disk, all completed map tasks must be re-executed. To avoid the rescheduling of tasks. This paper proposes a reliability task scheduling strategy with a node failure recovery mechanism.

This paper makes the following assumptions about the cloud environment platform:

(1) The cloud environment platform is heterogeneous.
(2) All nodes in the heterogeneous platform only have two working states: normal operation state and failure state. When a node fails, it will be in a failed state.
(3) The failure state of any node has nothing to do with other nodes and will not affect other nodes in a normal state.
(4) If a node is in an idle state when it fails, the node will be replaced by another node in the standby state, which does not affect the schedulability of the task. If there are tasks being executed on the failed node, the failure recovery mechanism and replacement mechanism will be used to ensure that the node is restored to a normal operating state.

2. TRUSTWORTHINESS MODEL

The degree of trust is a parameter for evaluating the reliability of a system or product. The reliability means that a certain system or product is trustworthy [5,6]. As far as a certain system or product is concerned, the task can be considered reliable when it can be completed normally according to user requirements. But if a task cannot continue to work to complete tasks in accordance with user requirements, then we think it is unreliable. For any product, the higher its reliability, the higher the trustworthiness of users. From the perspective of the definition of reliability, reliability refers to a measure of the trustworthiness of the system by users. That is the system completes the task specified by the user within the specified conditions and time, and does not cause the possibility of system failure during the task operation. Broadly speaking, reliability is the result of the user's subjective judgment on the system or product with the user's trustworthiness. In order to quantify the reliability, the reliability of a product or system in the industry is measured by its longest continuous working time without failure. The longer the working hours, the higher the probability of failure.

2.1. Trustworthiness Metrics

The trustworthiness measurement index can be divided into trustworthiness, failure rate, mean time before failure and average repair time. The trustworthiness means the probability that the system completes the task index within the specified time according to the needs of the user. Failure rate refers to the probability of system failure in the next unit time after time t, when the product or system has not failed after working to a certain time t and the product or system has not failed. Failure rate is a common quantitative feature for reliability analysis of a system or product. The higher the probability of failure of a product or system, the lower its trustworthiness. Mean time before failure assumes that under the same test conditions, the failure times of N irreparable systems or products are t1, t2, ...tN, and the average value can be obtained. For some systems or products that fail and cannot be repaired, the mean time before failure is the average life. The

average repair time refers to the average repair time of the system or product that can be repaired. The repair time of the system or product is not a definite time.

2.2. Establish a Trustworthiness Model

Definition 1: Node model. Assuming that the nodes in the cloud environment are heterogeneous, there are differences in the performance of each node Nj in N. The node model to be studied is described as an undirected graph G(T, E). The node set is N={N1,N2……,Nm}. M is the number of nodes. E represents the edge set of a graph, which mainly represents the relationship between nodes. For different nodes, their computing power is in the different levels. The response time of the same task on different nodes is also different. To record the response time of the task on different nodes, it is represented by an n*m matrix RT, and RTij represents the running time of task i on node j. A matrix TI of order m*m is used to represent the communication volume between nodes, where the communication volume between node a and node b is represented as TIab. In the undirected graph G(T, E), the virtual machines are independent of each other and have no dependencies.

Definition 2: Task response time Tij refers to the time it takes for all tasks to complete and return results, including task waiting time WTij, task transmission time CTij, and task execution time RTij.

$$T_{ij} = WT_{ij} + CT_{ij} + RT_{ij} \quad (1)$$

The time when all tasks are completed, we use ST to indicate.

$$ST = \max \quad T_{ij} \quad (2)$$

Definition 3: The trustworthiness of nodes is mainly considered from two aspects of node failure rate and failure repair rate. The failure of the node is mainly the communication link between the nodes and whether the node itself fails [7]. The failure recovery distribution is recoverable failure and unrecoverable failure, and communication link failure is unrecoverable failure. We define Ik to indicate whether the node is recoverable, if it is an unrecoverable failure, then Ik is 0, otherwise Ik is 1.

It is assumed that the probability of node failure and the probability of communication link failure between nodes obey the Poisson distribution with parameters σ and ξ, respectively. The probability of node failure p times in the time interval [0,t] $\lambda_p(t) = e^{-\sigma t}/p!$. In the same way, it can be known that the probability of communication link failure p times within the time interval [0, t] is $\theta_p(t) = e^{-\xi t}/p!$ It is assumed that the probability of a node failure and recoverable obeys the Poisson distribution with a parameter of ε, where the communication link cannot be recovered when the communication link fails. It is supposed that the probability of the node is repaired p times in the time interval [0,t] is $\mu_p(t) = e^{-\varepsilon t}/p!$.

Define Pj (t) as the probability that there is no failure (ie p=0) on node j in the time interval [0,t], and its probability value is:

$$P_j(t) = e^{-\sigma_j t} = e^{-\sigma_j RT_{ij}} \quad (3)$$

Define Cj (t) as the probability of no failure (ie p=0) on the communication link between node a and node b in the time interval [0, t], and its probability value is:

$$P_j(t) = e^{-\xi t} = e^{-\xi_j TI_{ab}/Net_{ab}} \quad (4)$$

In Equation 4, TIab/Netab represents the communication time between nodes on the communication link.

Define the probability that Rj(t) does not occur repair on node j in the time interval [0,t] (that is, p=0), and its probability value is:

$$R_j(t) = e^{-\varepsilon} = e^{-(1-I_j)\varepsilon_j RT_{ij}} \quad (5)$$

According to the above introduction, the probability of node j to complete task i is Kij.

$$K_{ij} = P_{ij} \times C_{ij} \times R_{ij} = e^{-\sigma_j RT_{ij}} \times e^{-\xi_j TI_{ab}/Net_{ab}} \times e^{-\varepsilon_j RT_{ij}} = e^{-\sigma_j RT_{ij} - \xi_j TI_{ab}/Net_{ab} - \varepsilon_j RT_{ij}} \quad (6)$$

To improve the parallelism of applications, a job is often divided into multiple tasks and executed in parallel on multiple nodes at the same time. A task is executed on only one node. When all tasks return task execution results, it indicates that the job has been successfully completed. Assuming that N(j) is the set of nodes performing the task, the reliability of the task can be expressed as:

$$Trust(N) = \prod_{i=1}^{n} \prod_{j=1}^{m} K_{ij} = \prod_{i=1}^{n} \prod_{j=1}^{m} e^{-(\sigma_j RT_{ij} + \xi_j TI_{ab}/Net_{ab} + \varepsilon_j RT_{ij})} = \prod_{i=1}^{n} \prod_{j=1}^{m} e^{trust_{ij}} \quad (7)$$

The task scheduling strategy that aims at maximizing the reliability of task scheduling and minimizing the total response time of tasks is a common task scheduling model in the cloud environment. To minimize the total response time of the task and optimize the reliability of the scheduling strategy, the objective function is set as equation 8 according to equations 2 and 7.

$$MinF = -x\, ln(Trust(N)) + ST \quad (8)$$

Since the value of the total response time of the task is larger than the value range of the reliability, this paper use x as a scale factor to coordinate the proportion of reliability and time cost to prevent the time cost from controlling the objective function value.

3. RELIABILITY TASK SCHEDULING ALGORITHM

Aiming at the task scheduling problem in cloud environment, the trust evaluation model considering failure recovery mechanism is introduced into ant colony simulated annealing algorithm. Meanwhile an ant colony simulated annealing algorithm considering failure recovery mechanism is proposed in this paper.

Using SA-based Ant Colony Optimization [8] based on Simulated Annealing (ACOSA), its principle is to find the local optimal task scheduling solution for task T_i and node N_j through ACO, and then uses SA to perform local optimization, thereby reducing the task allocate to the appropriate heterogeneous resource node for execution. This paper takes the ACO to set the initial

pheromone concentration to a constant before the ant searches, thereby the ant's search space can be increased:

$$\tau_j(0) = c$$

With the increase of time, the concentration of pheromone becomes higher and higher, and the ant can choose the appropriate path according to the concentration of the pheromone on the path that the ant walked last time.

When the task is scheduled to be executed on the resource node, the trustworthiness reflects the reliability of the service provided by the target resource node. The ACOSA algorithm fully takes into account the heterogeneous characteristics of resource nodes, but does not consider the influence of the trustworthiness of resource nodes on the task scheduling results. For this reason, this paper takes the maximization of credibility and the minimization of time cost as the objective function, and the Function as a heuristic function of ACO.

$$\eta_{ij} = \frac{1}{-x\ln(K_{ij}) + RT_{ij}}$$

In the actual cloud environment, when a node fails, the tasks assigned to the node will often be re-executed. With the introduction of a failure recovery mechanism, for those failures that can be recovered, the node can recover the stopped tasks by running the failure recovery program.
Execution steps based on reliability task scheduling strategy are as followed:

(1) Initialization parameters. Set the initial temperature T_{max}, the maximum number of iterations $Iter_{max}$, and the initial pheromone τ_{ij}.
(2) Construct a feasible solution. Ant_j selects the appropriate node according to the selection migration rule, adds the selected node to the taboo table, and instructs all tasks to be allocated to the appropriate node resources.
(3) Update pheromone. When all ants complete the path search, a local optimal solution is generated, and the local optimal solution is used to update the local pheromone.
(4) SA performs partial optimization. According to the local optimal solution obtained by ACO, SA optimizes the local optimal solution and a new solution is can be obtained.
(5) Metropolis guidelines. According to Metropolis criterion, the new solution constructed by SA is judged whether it will be accepted.
(6) Termination criteria. Cool down the current temperature and determine whether the termination criterion is met. If it is met, execute step(7), otherwise return to step(4).
(7) Global pheromone update. Updating the global pheromone according to the candidate solution generated by SA, the number of iterations is increased by 1. If the number of iterations is greater than Itermax, all steps are terminated, otherwise, return to step(2).

4. SIMULATION

Aiming at the simulation experiment of the reliability task scheduling strategy with the introduction of the failure recovery mechanism, the reliability of the strategy is mainly determined by the trustworthiness of the node and the trustworthiness of the communication link. According to the algorithm performance proposed in the literature [9], the size of the application task is related to the communication / computation ratio R_{cc} (Communication to Computation Ratio). Therefore, the task type will be determined by the communication / computation ratio. $R_{cc}>1$ indicates that the task is communication-intensive, and $0<R_{cc}<1$ indicates that the task is computationally intensive.

The simulation experiment focuses on the effect of failure recovery mechanism and its parameters on the evaluation of the trustworthiness of resource nodes, and the performance of algorithms under different nodes and tasks will be compared.

This paper firstly verifies the effectiveness of the failure recovery mechanism. For different types of tasks, it is discussed that the impact of the node failure recovery rate u_j on the probability of successful task execution and task completion time under the premise of introducing the failure recovery mechanism.

The experimental environment parameters are set as follows: task R_{cc} is 0.1, 1, 10 respectively; the number of tasks is 100, the number of nodes is 20, and the number of communication links is 20. Set up two kinds of nodes with low trustworthiness, which account for 20% and 30% of the total number of nodes respectively, and the probability of execution failure of the two types of nodes is 80% and 50% respectively. The experimental results are shown in Figure 1.

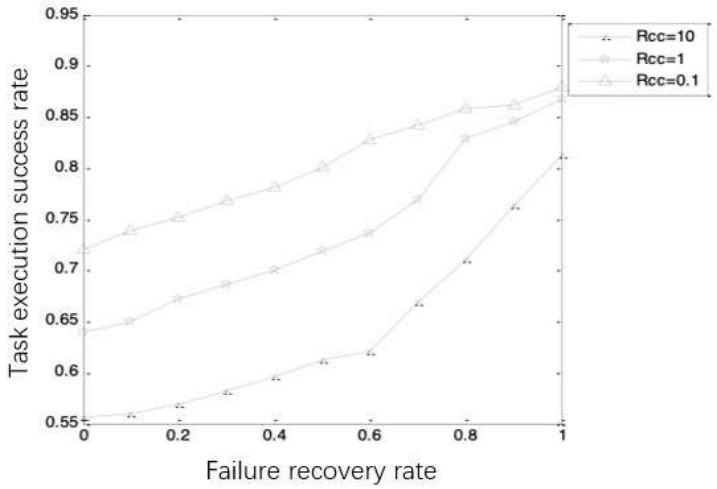

Figure 1. Task execution success rate under different failure recovery rates (without limiting the maximum number of recovery times)

It can be seen from Figure 1 that when there is no failure recovery mechanism (that is, u_k=0), the probability of successful completion of the three types of tasks is relatively low. As the failure recovery rate u_k increases, the probability of successful execution of the three types of tasks slowly increases. When u_k is 1, the probability of successful task execution does not reach 100%. Because although recoverable failures can be recovered through the application, some failures are not recoverable. The two curves with $R_{cc}$of 0.1 and 10 in Figure 1 represent computation-intensive tasks and communication-intensive tasks respectively. Communicationintensive tasks use the communication link for a relatively long time, and the probability of communication link failure is higher than that of computationally intensive tasks. Since the failure of the communication link is unrecoverable, the probability of successful completion of communication-intensive tasks is greater than that of computationally intensive tasks.

In order to study the effect of failure recovery rate on task execution time, this paper conducted the experiments on three different types of tasks. The experimental results are shown in Figure 2.

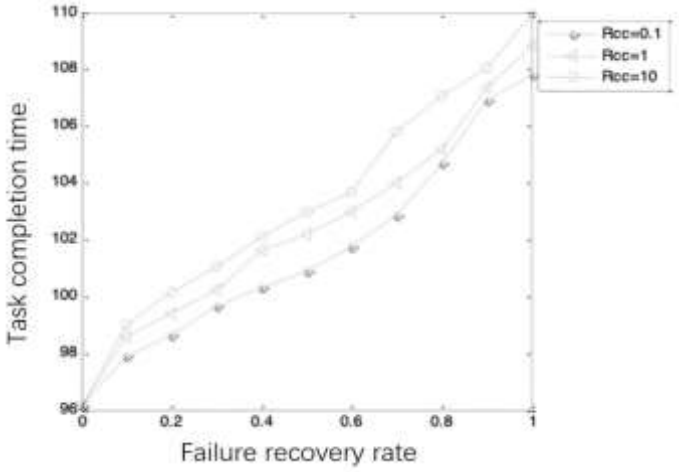

Figure 2. Task completion time corresponding to different failure rates

It can be seen from Figure 2 that different failure recovery rates depend on different task execution times. As the failure recovery rate increases, the completion time of the task is gradually extended. This phenomenon is caused by the service overhead generated during the failure recovery process with the task execution time to increase. In an ideal situation, the node has no failure u_k=0, all tasks can be executed successfully according to the assigned node, and the task completion time is the shortest. In Figure 2, when the failure recovery rate is less than 0.6, the task completion time increases slowly. But when the failure recovery rate exceeds 0.6, the task completion time increases sharply. This phenomenon is shown the failure recovery time will increase when the node fails to recover. Therefore, frequent failure recovery will increase the task completion time. Choosing an appropriate failure recovery rate has an important impact on the task completion time.

In the case of considering the number of different tasks, how the task execution success rate and the value of the objective function are compared between the FCFS algorithm (First Come First Service) and the ACOSA algorithm, where the failure recovery rate u_k is set to 0.6.
Figure 3 shows that different task numbers correspond to different task execution success rates and target function values corresponding to different tasks.

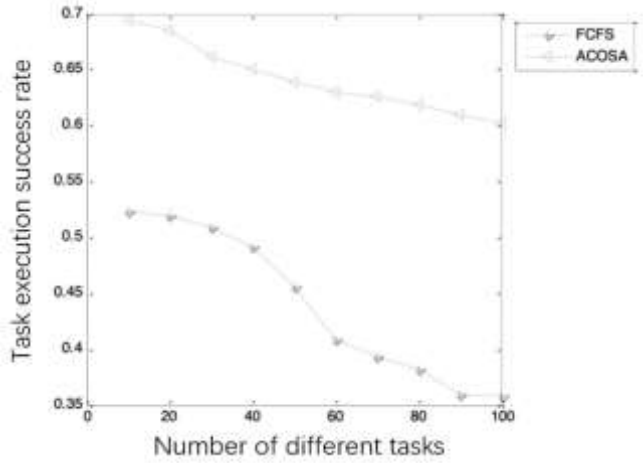

Figure 3. Different task numbers correspond to different task execution success rates

It can be seen from Figure 3 that when the number of tasks gradually increases, the success rate of FCFS algorithm and ACOSA algorithm task execution is gradually decreasing. The task execution success rate of the ACOSA algorithm with a failure recovery mechanism is significantly higher than that of the FCFS algorithm.

It can be seen from Figure 4, that the task completion time of the ACOSA algorithm with the introduction of the failure recovery mechanism is shorter than the task completion time of the FCFS algorithm. It can be proved that the introduction of failure recovery mechanism can not only improve the success rate of task execution. When an appropriate failure recovery rate is selected, the performance of the ACOSA algorithm that introduces the failure recovery mechanism is better than the FCFS algorithm.

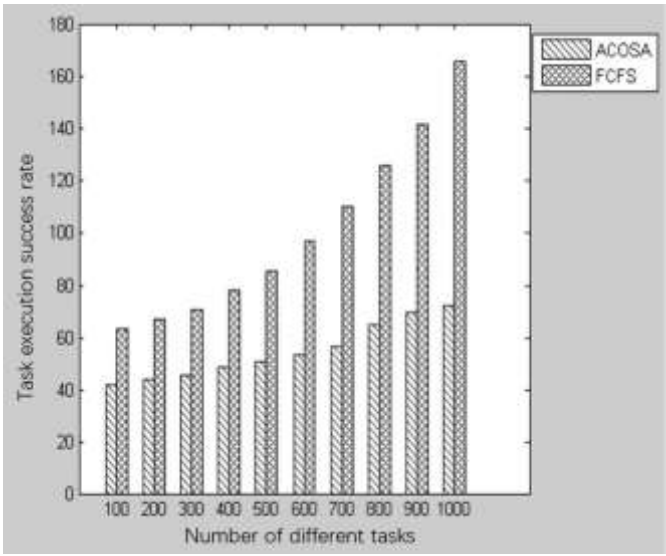

Figure 4. shows the objective function values corresponding to different tasks

5. CONCLUSION

MapReduce distributes the tasks of large-scale data set operations to each sub-node under the management of the master node to complete it, and then obtains the result by integrating the intermediate results of each sub-node in a cloud environment with many resource nodes. Therefore, the event of node failure is inevitable. Aiming at the defects of MapReducefaulttolerant mechanism, this paper proposes a task scheduling strategy based on trustworthiness. Based on the analysis of trustworthiness metrics, a trustworthiness model is constructed, and a failure recovery mechanism is introduced for recoverable faulty nodes. The scheduling strategy that combines the failure recovery mechanism and the trustworthiness model provides a reliable guarantee for the successful execution of the task. Finally the simulation platform CloudSim verifies the validity and stability of the task scheduling algorithm and scheduling model proposed in this paper.

ACKNOWLEDGEMENTS

The research work of this paper is approved by General Research Project of Natural Science Foundation of Universities of Jiangsu Provincial Department of Education with 19KJD520007.

REFERENCES

[1] Tian Zhuojing , Huang Zhenchun , Zhang Yinong. Review of Task Scheduling Methods in Cloud Computing Environment[J]. Computer Engineering and Applications. 2021,57(02):1-11.

[2] Zhang Hang, Zhang Xin, Zhang Pingkang, Li Qi. Parallel Weighted FIUT Algorithm Based on Mapreduce[J]. Microelectronics & Computer. 2018,35(07):41-44.

[3] Ye Haiqin, MengCaixia, Wang Yifeng, Zhang Ailing. Frequent pattern mining algorithm based on MapReduce[J]. Journal of Nanjing University of Science and Technology. 2018,42(01):62-67.

[4] Ren Gang, Deng P, Yang C, Wu C. Mapreduce Back Propagation Algorithm Based on Structure Parallelism. Journal Computer Research and Development. 2018,55(06):1308-1319.

[5] Zhang Jie. The Trust of Networked Software Measurement Model to Optimize The Simulation Analysis[J]. Computer Simulation. 2016,33(10):278-281+299.

[6] Jiang Jing, Yu Yonghong, Zhao Weibin. Research on A Trust Model Based on the QoS and Malicious Node Deletion[J]. Computer & Digital Engineering. 2020,48(01):98-105.

[7] Wu Chun, You Xiaojian, Lyu Tao. Research on multipath secure routing based on confidence degree[J]. Journal of Northeast Normal University (Natural Science Edition). 2018,50(04): 66-72.

[8] Jiang Qiang, Yi Chunlin, Zhang Wei, Gao Sheng. The Multi-objective Path Planning for Mobile Robot Based on Ant Colony Algorithm[J]. Computer Simulation. 2021,38(02):318-325.

[9] Luo W, Yang F M, Pang L P, et al. A real-time fault-tolerant scheduling algorithm of periodic tasks in heterogeneous distributed systems[J]. Chinese Journal of Computer. 2007, 30(10):17401749.

AUTHORS

Qin Jun, Professor, graduated from Nanjing University of Posts and telecommunications. He has presided over and participated in more than 30 scientific research projects, won more than 10 awards, published more than 80 academic papers in academic journals and international conferences, and published 5 teaching materials.