Replica Scheduling Strategy for Streaming Data Mining

Shufan Li[1], Siyuan Yu[2], Fang Xiao[3]
Computer Science and Artificial Intelligence School, Wuhan University of Technology, Wuhan, China[1, 2]
Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, China[3]

***Abstract*—In a distributed storage and computing framework, traditional streaming data mining techniques are inefficient when processing massive amounts of data. In this paper, we take the copy in cloud storage as an allocatable resource for scheduling and propose a RepRM strategy to improve the efficiency of data mining and analysis. The key idea of this work is to take the data copy as the resource to be allocated, and use the backward inference method of dynamic programming to solve the data copy ratio, the optimal number of copies is obtained. Experiments and observations have proved that compared with the traditional scheduling method of Hadoop, after adopting the RepRM strategy scheduling, the memory resources of the homogeneous cluster are saved by about 40-50% during parallel mining of streaming data, and the throughput rate is increased by 20% to 30%.**

Keywords*—Streaming data mining; dynamic programming; replica scheduling strategy; cloud computing

I. INTRODUCTION

The continuous development of computer science has resulted in more and more tasks and data needed to be processed, and the computer processing capacity and processing speed have been difficult to meet the needs of users. As a product of virtual technology, cloud computing can process massive data and tasks. However, due to the huge amount of computing, the cloud platform needs to allocate the system resources to each task in the computing process reasonably. What's more, more and more streaming data are stored in the cloud, especially large data analysis systems store a great deal of data, such as data generated by sensors, generated by network management equipment, and generated by core switches, such as log data, audio and video data, etc. and these data are continuously generated in a data stream according to time. Consequently, an effective strategy to allocate the system resources to each task in the computing process is vital.

The mining of massive streaming data is applied to various fields and it produces valuable analytical results for it. Taking marketing as an example, by mining and analyzing the market behavior of a large number of user data, we can guide the further market working by getting the consumption habits of users. For example, according to the consumption situation of users' credit cards, we can directly know the main consumer needs, shopping interests and consumption concepts, which is very valuable information for marketing and product promotion, and is helpful to guide the next market planning.

Because massive streaming data has the characteristics of big data, and for the upper application of cloud storage, the processing and analysis of massive streaming data are very different from the previous processing. Taking data query and mining as examples, users need to query and analyze the accumulated data for a long time when analyzing the data, which has high requirements for the data searching and comparison performance in cloud storage, while the past data query only queries and compare for a certain data. So the traditional streaming data mining technology is inefficient when processing huge amounts of data. Uncontrollable and continuous surge of large volumes of data has exacerbated the trend of "the explosion of data and the lack of knowledge". At this time, the distributed computing platform has become a research hotspot as an effective means to solve the problems [1,2].

Take Hadoop as an example, the homogeneous or heterogeneous distributed computing platform built by it can meet the needs of large-scale data processing technology in scientific research, engineering and other fields, it has become one of the mainstream data processing frameworks of large Internet companies at home and abroad, such as the data processing applications of Yahoo, Twitter, and other companies. With the increasing scale of the Hadoop cluster and the increasing fields of use, the management and usage of resources are increasingly valued. Many researchers take YARN's resource scheduling algorithm as their research direction, through researching and designing reasonable resource allocation algorithms to achieve higher resource utilization. But among the numerous studies, according to the characteristics of streaming data, it is rare to design and implement the resource scheduling strategy of the data mining algorithm, so that the current mining model for streaming data in the cloud platform cannot effectively allocate the resources in the cluster, or complete the mining of streaming data efficiently.

In practical applications, most streaming data mining algorithms are aimed at a certain type of streaming data [3,5]. The operating parameters of the algorithm are determined in the program initialization stage and cannot be dynamically modified, which is called a "static algorithm" [4]. Although some algorithms can be regulated through parameter adjustment to adapt to the dynamic application environment, these dynamic algorithms are relatively rigid and have no learning and adaptive capabilities. For example, Franke pointed out that streaming data is usually a sequence of data objects with time as the latitude. Therefore, the execution process of

more streaming data mining algorithms is sorted according to the time when the data arrives sequentially, and a single linear scan method is used to analyze the data. Analyze and compare [4].

Alternatively, the relationship between steaming data and environmental differences has been considered by Knorr and other professors. Different environments can lead to differences in the expression form of streaming data, which can then lead to differences in data mining algorithms. For example, the resource allocation of a computer system is dynamic, and the data mining algorithms may allocate more or fewer resources (such as CUP, memory, etc.) because of the actual load situation of the computer system, and the number of resources will directly affect the response time and processing time of the streaming data mining algorithm. Therefore, an adaptive resource scheduling method is needed so that the resources can be adjusted according to the different streaming data.

The work done in this paper is as follows: First, verify the impact of replicas on the throughput of streaming data mining. After analyzing the relationship between the copy and the throughput rate of streaming data mining, the resource allocation model of streaming data mining is established. Secondly, according to the characteristics of streaming data, the resource scheduling model of the cloud platform incorporates the copy data resources and the resource (replica) scheduling model for the cluster. Once again, build a model to analyze and solve. Finally, based on the Hadoop big data platform, the scheduling model was implemented by improving the YARN component. What is more, in order to verify the effectiveness of the copy-based resource scheduling model, we use two types of test data sets and real network traffic data sets to test the improved resource scheduling strategy.

II. Related Work

In order to propose an efficient resource scheduling model, we should consider the challenges which come from several aspects. Then we will investigate the recent research about these challenges as follow to build a proper model [6].

Dominant Resource Fairness (DRF) is a multi-resource allocation algorithm of max-min fairness [7]. The authors introduced that dominant share is the largest share of any resources that distribute to the user, and DRF conceived that the number of resources assigned to the user should be determined by its dominant share. After all, the purpose of DRF is to seek the maximum of the minimum share among all users. While DRF figured out the demand heterogeneity of multiple resources, however, DRF neglects the heterogeneity, it based on an assumption that computing resources are generally assumed to be isomorphic. In order to resolve more problems, many researchers have optimized and expanded DRF. DRFH [8] is a multi-resource allocation mechanism. For DRFH, the resources are pooled by a large number of heterogeneous servers; they are representing processing, memory, storage and so on. DRFH equalizes the global dominant share allocated to each user. However, it fails to consider users' placement constraints. PS-DSF [9], an extension of DRF is applicable for heterogeneous resource-pools in the presence of placement constraints. Its solution is defining a virtual dominant share.

For further study, the current schedulers such as DRF which only for the fairness can't meet our needs, so professionals investigated other schedulers which consider the counterbalance between performance and fairness [6]. Gemini [10] considers two scheduling policies during runtime, the former maximizes the utilization of resources by balancing the remaining capacity of the resources of the node, while the latter is to fairly allocate resources to users in a system containing multiple types of resources. During the adjustment process, the strategy will be selected by estimating the loss of performance and fairness.

The cost of a great deal of energy consumption has taken up a considerable part of the total cost of the data center. When we not only need to reduce this part of the cost, but also meet the QoS requirements of users, it's been a main research topic of resource scheduling strategy. EFS [11] is an Energy-aware Fair Scheduling framework based on YARN, EFS uses dynamic node management. It can meet the users' QoS requirements and reduce the energy consumption, because of the energy-aware resources scheduling and the strategy of turning off the unused nodes for a specified duration. However, EFS does not consider the data distribution and replication dependencies.

In recent years, cloud storage systems have emerged as a promising technology for storing data blocks on various cloud servers [12], and replica is the basic means of tolerance and availability of the distributed systems, so the distribution of data copy is significant to the systems [13], many problems can be resolved by data replication algorithms. However, data replication also produces energy consumption and costs, so it is very important to reasonably schedule copy resources in resource scheduling, there are also many kinds of research about this topic.

In order to solve the problem about the placement of data replica, Cui et al. [14] build a tripartite graph-based model, and propose a genetic algorithm-based data replica placement strategy. The dynamic multi-objective optimized replica placement and migration strategies for SaaS applications in edge cloud are proposed by Chunlin et al. [15]. According to the result of the experiment, this strategy can improve the utilization rate of network resources. Huang and Wu [16] not only proposed an optimization model for data replication and placement problem, but also designed hybrid genetic algorithm based on data support degree to solve the model. The algorithm is found to have good performance by using real data set. Khojant et al. [17] proposed Predictive Frzzy Replication (PFR). The new algorithm can replicate the historical usage of files, files size, the level of the sites and free available space for replication in advance and decide which replications should be deleted through forecasting of future demand and the relevant file of the replications to save cost. Salem et al. [18] created a new algorithm derived from a combination of ABC and Multi-Objective Optimization. The proposed MPABC algorithm enables fast access to the data and selects the best copy location closest to the user.

In these present studies, the purpose of most resource scheduling models is to allocate resources such as CPU, memory and bandwidth, they don't consider the data replica, CPU and memory simultaneously, unified scheduling. So we incorporate data copy as a scheduling resource when build a new resource scheduling model. And from the perspective of the whole process of data processing, the bottleneck of the big data processing performance in the distributed computing platform lies in the data transmission consumption, not the CPU computing power [19]. Therefore, an effective resource scheduling method is needed to select appropriate nodes for data processing, that is, considering the location of the current copy, data mining is performed on the node where the copy is located, so that the mining speed is greatly improved. Based on this, this article starts from the perspective of copy selection, takes the copy as a data resource and considers it as the computing resource at the same time, and uses the dynamic programming model to design the resource scheduling model of the big data platform to improve the data mining throughput rate of the cloud platform.

III. Materials and Methods

A. Distributed Application and Copy Selection

In order to take Hadoop and OpenStack as examples, the copy in the cloud platform is generally set to 3, which means that the same data may only exist in a few nodes of the cloud platform. Therefore, when a distributed application applies for the use of resources such as CPU and memory, the positional relationship between the node where the resource is located and the copy is extremely important.

1) Verification of the impact of replicas on distributed mining: For the purpose of verifying and testing the impact on data copies on distributed mining, we use the KDD CUP 2000 data set for verification, and use Hadoop clusters and stand-alone machines to analyze and mine streaming data respectively. Table I shows that the Hadoop and single configuration used for the experiment. The experimental process is to extract NetFlow seven-tuples from KDD CUP 2000. What is more, The Hadoop test uses MapReduce to submit a task which is NetFlow analysis to Hadoop, and the stand-alone test uses the network interactive analysis tool set SILK for NetFlow seven-tuple analysis.

The results of the experiment are shown in Fig. 1. It can be found from the experiment that when the total amount of data is small (less than 400G), in contrast to the Hadoop distributed platform, in data analysis ability, a single machine has apparent advantages, which is just 70%. When the total amount of data is large (greater than 400G), the Hadoop distributed platform shows its advantages progressively. When the amount of data is 500G, the analysis time is only 84% of the single machine. But 5 servers make up this distributed platform, and the single platform uses just one server. It can be seen that using the single platform used in the experiment can process 400GB stream data within 15 minutes, which can meet the seven-tuple analysis of about 3.5Gpbs network traffic.

TABLE I. Server Configuration of the Experiment

Hadoop	**Stand-alone**
4 slave hosts: CPU: Two-way four-core 2.6G RAM: 48G Disk: 40TB	CPU: Two-way eight-core 2.6G RAM: 60G Disk: 20TB
1 master host: CPU: Two-way eight-core 2.6G RAM: 60G Disk: 20TB	

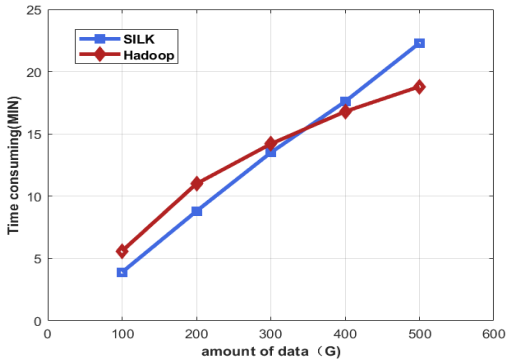


Fig. 1. Time Consuming (MIN).

In situation of a bit of data, the location of the replica in the Hadoop cloud platform will affect the throughput rate of the distributed platform for traffic analysis. Hadoop's three-copy strategy requires some computing nodes to wait for the completion of the network transmission of data, thus increasing the time consumption of network transmission; and the MapReduce task allocation mechanism has a certain time loss at the beginning and end of the task. Therefore, in the case of a relatively small amount of data, low throughput for traffic analysis on distributed platforms.

2) The optimization problem of data copy in data mining: Through the verification of *1)*, disposable resources such as replicas are important factors that apparently affect the capability of data mining algorithms. Both the results and throughput of data mining algorithms not only depend on the allocation of computing hardware resources such as CPU time and available memory, but are also closely related to the number of copies. In the case of a small number of copies and limited computing resources, there will be a disparity between data mining algorithm results and the optimal results.

The mining model can adjust itself by modifying its copy resources and computing resources, thereby increasing the throughput of the mining model. Therefore, to make the throughtput of the mining algorithm higher, we need to adjust the computing resources's allocation and data resources. Especially for real-time streaming data mining, it is more necessary to dynamically adjust how computing resources and data resources are allocated, so that the real-time mining model is resource-adaptive [4].

Take the WEB server as an example, the WEB server needs to process a large number of incoming and outgoing data packets in the network in real-time. This is the typical stream of data. Within a period of time, the WEB server will receive data packets from the Internet and send a certain number of data packets to the Internet in chronological order. These data packets are usually organized in the form of IP data packets. Each IP data packet constitutes a stream data object, which together constitutes a data stream on the WEB server. When the network traffic is large enough, due to the limited resources of the host, online analysis, offline analysis, and initial filtering of stream data will all cause conflicts in resource usage.

For the contradiction between resources and speed, the solution is to effectively allocate limited data resources and computing resources [20, 21]. Taking the traditional mining model as an example, it will affect the four modules of the model: data filtering module, online mining module, offline mining module and resource detection module. Assuming that the process of processing a stream data object includes data online mining module, resource detection module, offline processing module and data filtering module, the total amount of available resources is R, of which data online mining module, resource detection module, offline processing module and the resources consumed by the data filtering module are R_1, R_2, R_3 and R_4. Among these four modules, the online mining module and the data filtering module need to detect and process the data in real-time, so the resources consumed are increased or decreased at the same time; on the other hand, the offline mining module processes the data offline, the resources which are consumed do not increase or decrease at the same time as the first two. According to this situation, the resource consumption can be adjusted into three parts: the overall consumption R_1+R_4 of the online mining module and the data filtering module, and the consumption R_2 and R_3 of the resource detection module and the offline mining module. So when the total amount of resources R is certain, how to effectively allocate the resources of R_1+R_4, R_2 and R_3, so that the throughput rate of the online mining module, resource detection module, offline mining module and data filtering module can be increased to V_1, V_2, V_3 and V_4, which maximizes the throughput rate, is a dynamic programming problem that optimizes resource allocation. To resolve this problem, the following modeling is required:

$$R = \sum_{i=1}^{n} R\left(p, z\left(p\right)\right) \tag{1}$$

In order to deal with the problem about dynamic programming, the resource allocation process is first divided into n different stages to allocate resources. In the pth stage, the pth module is allocated resources by the system. After the allocation is completed, the number of resources remaining in the system is q(p+1). Then the system allocates the remaining resources q(p+1) to the p+1, p+2, ..., nth modules. The optimal function value that can be obtained in the p+1 stage is set to t(q(p+1), p+1), that is, under the premise of resource q(p+1), the final p+1 to n is completed. The maximum mining throughput rate that can be obtained by the allocation of each module.

Then, for dynamic programming problem, the basic equation is:

$$\begin{aligned} &t\left(p, q\left(p\right)\right) = \max_{z(p)\in M\left(q(p)\right)} \left\{R\left(p, z\left(p\right)\right) + t\left(q\left(p+1\right), p+1\right)\right\}, \\ &p = n-1, ..., 1 \\ &t\left(n, q\left(n\right)\right) = R\left(p, z\left(n\right)\right) \end{aligned} \tag{2}$$

The corresponding state transition equation is:

$$q\left(p+1\right) = q\left(p\right) - z\left(p\right) \tag{3}$$

For the resource optimization problem, according to Formula 2 and Formula 3, a related mathematical model is established. To solve the model, the optimal decision sequence $(z'(1), z'(2), \cdots, z'(n))$ can be obtained through the inverse method, and then the maximum throughput rate R(q(1)) of the mining algorithm can be obtained. Specifically, in the data mining model, n is the 4 modules in the model, that is, n=4, and the total resource number S is the number of copies of the data resource.

In a distributed system, it is supposed that the resources of S units need to be allocated to n modules, where S represents the number of stream data objects that can be processed and is a positive integer. Assuming that the throughput of data mining can be increased to R(k, z(k)) after z(k) resource units are allocated to the kth module, then the overall goal of resource allocation is to allocate S to each module. The resources of each unit finally make the total throughput of mining reach the highest, that is, the R value in Formula 1 reaches the maximum.

In this paper, we know that in the distributed framework, data mining is usually carried out in parallel, so it is usually multiple mining algorithms to mine massive data. Taking a typical parallel mining model as an example, suppose that there are S units of resources need to be allocated by distributed system to n parallel mining algorithms, and *S* is the number of data copies that can be allocated. Assuming that the throughput rate of data mining can be increased to R(p, z(p)) after allocating z(p) units of resources to the p-th mining algorithm, then the overall goal of resource allocation is to be reasonable for the data mining algorithm to allocate *S* data copies, and finally maximize the mining throughput rate, in other words, the *R* value in Formula 1 reaches the maximum.

Similarly, in order to deal with the optimization problem, the allocation replica is grouped into n distinct phases in the resource allocation process in a distributed system. The pth algorithm is allocated data copy resources by the system in the pth stage. After the allocation is completed, the number of copies remaining in the system is q(p+1). Similarly, the optimal function value that can be obtained in the q(p+1) stage is set to t(q(p+1), p+1). According to Formula 2 and the related theory of dynamic programming, the basic equation of the mathematical model can be determined at first:

$$t(p,q(p)) = \max_{z(p)\in M(q(p))}\{R(p,z(p)) + t(q(p+1),p+1)\},$$
$$p = n-1,...,1$$
$$t(n,q(n)) = R(p,z(n)) \quad (4)$$

According to the above Formula 4, the state transition equation in the mathematical model can be determined as:

$$q(p+1) = q(p) - z(p) \quad (5)$$

According to the established mathematical model, the solution of this model can be used to obtain the optimal decision sequence $(z'(1), z'(2), \cdots, z'(n))$ through the inverse method, so as to the mining throughput reaches the maximum value R(q(1)).

B. Replica Scheduling Strategy RepRM for Streaming Data

For big data systems (such as Hadoop systems), the distributed resource scheduling problem is an NP-hard problem. Mass flow data has a certain degree of no aftereffect, so the traditional scheduling algorithm can't allocate system resources well. In the whole process of mining algorithm operation, if adjustment of resources is a decision, the data mining between two decisions is a stage. Therefore, the resource allocation problem in distributed mining of streaming data is a dynamic programming problem. When performing distributed mining on streaming data, the resource allocation for it can be described as: how to allocate limited resources to multiple mining algorithms, so that the mining model can mine the streaming data to the greatest extent.

For various algorithms of parallel mining, take K-Means, KNN and Apriori algorithms as examples. It is assumed that each algorithm is performing streaming data mining, we need to consider how to build a model which can allocate limited resources to these algorithms to make the entire parallel mining model throughput rate be the largest. The problem comes down to how to allocate data copy resources to maximize the throughput of the parallel mining model, which is also a resource scheduling problem.

1) Replica scheduling based on the copy: Assuming that there are R assignable data copy resources in the distributed computing platform, n data mining algorithms are running on the platform at the same time, and the throughput of the algorithm in mining is related to the amount of replica resources put into use. Assuming that the i-th data mining algorithm are allocated by r_i data copy resources, and the throughput rate of the i-th mining algorithm is $f_i(r_i), i = 1,2,3\ldots,n$. At this time, the whole throughput rate of the platform's n algorithms on mining is $f_i(r_i), i = 1,2,3\ldots,n$.

Then the problem boils down to how to allocate R data copy resources: for n data mining algorithms, in order to maximize the total throughput rate, the total throughput rate reaches $g\ g_n(r_n)$. The programming model is as below:

$$\text{Max}: Z = f_1(r_1) + f_2(r_2) + \cdots + f_n(r_n)$$
$$s.t.\begin{cases} r_1 + r_2 + \cdots + r_n = R \\ r_i \geq 0, i = 1,2,\ldots,n \end{cases} \quad (6)$$

This depends entirely on the throughput function $f_i(r_i)$ of each mining algorithm. When $f_i(r_i)$ is a linear function. It is a linear programming problem; when $f_i(r_i)$ is a non-linear function, it is a nonlinear programming problem. Especially when n is relatively large, the solution process is extremely troublesome. However, due to the inefficiency of massive data, the problem is to solve a parallel resource allocation model, which can be solved by using the inverse relationship of dynamic programming.

Let the state variable of the number of data replication resources allocated to algorithm k to algorithm n be s_k, and the decision variable u_k represents the number of data copy resources allocated to algorithm k. After the data copy resource allocation decision of algorithm k is completed, let the number of data copy resources obtained by algorithm k be r_k, that is, $u_k=r_k$ after the allocation is completed. At this time, the number of replica resources r_k allocated to algorithm k satisfies:

$$s_{k+1} = s_k - r_k \quad (7)$$

And because $u_k=r_k$, you can get:

$$s_{k+1} = s_k - r_k = s_k - u_k \quad (8)$$

The allowed decision set is:

$$D_k(s_k) = \{u_k | 0 \leq u_k = r_k \leq s_k\} \quad (9)$$

When the number of data copy resources of sk is allocated to the k-th to n-th algorithms, the platform can get the maximum mining throughput rate of $g_k(s_k)$, and the inverse relationship of dynamic programming can be obtained:

$$\begin{cases} g_k(s_k) = \max\limits_{0\leq r_k \leq s_k}\{f_k(r_k) + g_{k+1}(s_k - r_k)\}, k = n-1,\ldots,1 \\ g_n(s_n) = \max\limits_{r_n = s_n} f_n(r_n) \end{cases} \quad (10)$$

Using Formula 10, we can calculate one by one algorithm, and finally get $g_1(s_1)$.

2) Replica scheduling model RepRM: Distributable data copy resources and changes in data characteristics are two important factors that affect the throughput of data mining algorithms. In distributed streaming data mining, the available resources can be reasonably allocated to the mining model according to the number of copies and the characteristics of the streaming data, and the throughput rate of the model can be improved.

If only considered the computing resources, the CPU and memory contained in each server are almost the same, and the allocatable resources used by each node for streaming data mining are also almost the same. Then the problem can be simplified to the configuration of data copy resources. At this point, the problem turned into a resource allocation problem.

Take a Hadoop cluster as an example. The cluster is composed of R identical servers (for example, DELL 820), which are used for network traffic analysis and mining. When multiple different users submit n network traffic data mining algorithm requests, the mining throughput rate is $f_i(r_i), i=1,2,3\ldots,n$, and the inverse method can be used to solve the problem. Assume that in this example, the cluster is composed of 5 homogeneous servers. At present, the Apriori algorithm is already running for online mining of streaming data. At the same time, two users request to use the KNN algorithm and the K-Means algorithm for offline mining of streaming data. And according to the number of data copies allocated to the mining algorithm, the sampling size during mining is also inconsistent. The specific sampling size is shown in Table II.

TABLE II. DATA COPY RESOURCE ALLOCATION AND SAMPLING SIZE OF HOMOGENEOUS CLUSTER

Number of servers	**0**	**1**	**2**	**3**	**4**	**5**
K-Means	0 Gbps	3 Gbps	7 Gbps	9 Gbps	12 Gbps	13 Gbps
KNN	0 Gbps	5 Gbps	10 Gbps	11 Gbps	11 Gbps	11 Gbps
Aprior	0 Gbps	4 Gbps	6 Gbps	11 Gbps	12 Gbps	12 Gbps

At this time, according to Formula 10, the inverse method can be used to obtain the optimal solution, and the solution process is as follows.

Solution:

$$\begin{cases} g_k(s_k) = \max\limits_{0\le r_k\le s_k}\{f_k(r_k)+g_{k+1}(s_k-r_k)\}, k=n\text{-}1,\ldots,1 \\ g_n(s_n) = \max\limits_{r_n=s_n} f_n(r_n) \end{cases} \quad (11)$$

Stage 3: When assigning s_3 data copies (s_3=0,1,2,3,4,5) to the Apriori algorithm, the sampling size is:

$$g_3(s_3) = \max_{r_3}[f_3(r_3)], r_3 = s_3 \quad (12)$$

Because only the Apriori algorithm is mining at this time, all data copies in the cluster can be allocated to it, so its sampling size is the maximum sampling size at this stage, as shown in Table III, in the table r_3^* means that $g_3(s_3)$ is the optimal decision at the maximum value.

Stage 2: Assuming that s_2 data copies (s_2=0,1,2,3,4,5) are allocated to the Apriori algorithm and the KNN algorithm, then for each s_2 value, the available sample size is:

$$g_2(s_2) = \max_{r_2}[f_2(r_2)+g_3(s_2-r_2)], r_2 = 0,1,2,3,4,5 \quad (13)$$

TABLE III. RESOURCE DYNAMIC DECISION OF HOMOGENEOUS CLUSTER STAGE 3

r_3	$f_3(r_3)$						$g_3(s_3)$	r_3^*
	0	**1**	**2**	**3**	**4**	**5**		
0	0						0	0
1		4					4	1
2			6				6	2
3				11			11	3
4					12		12	4
5						12	12	5

Because r_2 data copies of the KNN algorithm are allocated, the mining throughput rate is $f_2(r_2)$, and the remaining $s_2 - r_2$ data copies are used in the Apriori algorithm, so the Apriori throughput rate is $g_3(s_2-r_2)$. The sample size is $f_2(r_2)+g_3(s_2-r_2)$ at this time. Therefore, it is necessary to select an appropriate value of r_2 to maximize the function. At this time, the numerical calculation table is shown in Table IV.

Stage 1: In order to obtain the maximum mining throughput rate, 5 data copies need to be allocated to the algorithms for calculation. Therefore, when s_1=5 data copies are allocated to the Apriori, KNN and K-Means algorithms, the mining throughput rate is:

$$g_1(5) = \max_{r_1}[f_1(r_1)+g_2(5-r_1)], r_1 = 0,1,2,3,4,5 \quad (14)$$

At this time, the maximum value of this function is the maximum mining throughput rate. The specific numerical calculation table is shown in Table V.

According to the numerical calculation table, there are two executable solutions:

1) When r_1^*=0, look up Table V and Table IV to know that the allocation plan at this time is: r_1=0, r_2=2, and r_3=3.

TABLE IV. RESOURCE DYNAMIC DECISION OF HOMOGENEOUS CLUSTER STAGE 2

r_2	$f_2(r2)+g_3(s_2-r_2)$						G_2 (s_2)	r_2
	0	**1**	**2**	**3**	**4**	**5**		
0	0						0	0
1	0+4	5+0					5	1
2	0+6	5+4	10+0				10	2
3	0+11	5+6	10+4	11+0			14	2
4	0+12	5+11	10+6	11+4	11+0		16	1,2
5	0+12	5+12	10+11	11+6	11+4	11+0	21	2

TABLE V. RESOURCE DYNAMIC DECISION OF HOMOGENEOUS CLUSTER STAGE 1

r_1	$f_1(r_1)+g_2(5-r_1)$						$G_1(5)$	r_1
	0	**1**	**2**	**3**	**4**	**5**		
5	0+21	3+16	7+14	9+10	12+5	13+0	21	0,2

That is, the K-Means algorithm does not allocate data copies, the KNN algorithm allocates 2 data copies, and the Apriori algorithm allocates 3 data copies. At this time, the maximum throughput rate of the mining algorithm is 21Gbps.

2) When r_1^* =2, look up Table V and Table IV to know that the allocation plan at this time is: r_1=2, r_2=2, and r_3=1.

That is, the K-Means algorithm allocates 2 data copies, the KNN algorithm allocates 2 data copies, and the Apriori algorithm allocates 1 data copy. At this time, the maximum throughput rate of the mining algorithm is 21Gbps.

IV. Result

A. Experiments and Observations

This article implements the RepRM model by revising the resource management and scheduling component (YARN) in Hadoop and modifying the Scheduler component in the Resource Manager module. By default, YARN only supports memory scheduling (such as Capacity strategy, Fair strategy). It uses "containers" to encapsulate memory and CPU. When tasks have resource requirements, they apply to YARN for the CPU and memory "containers" required by the task.

In the testing session, we conducted experiments and observations on the RepRM replica scheduling method. For the parallel streaming data mining model, we used KNN, K-Means and Apriori algorithms to simultaneously mine online. Then observe the differences between YARN's built-in Capacity strategy, Fair strategy and the improved RepRM strategy.

1) Experimental environment: In the experiment and observation, the Hadoop cluster used for testing is a 2U rack-mounted DELL PowerEdge FX2 server with a convergent architecture, which contains 4 nodes, and the operating system uses CentOS. The hardware configuration of each node is shown in Table VI. The experimental data includes two test data sets: the test data set generated by the IBM synthesizer and the WEB access traffic data.

a) Experimental data set: IBM synthetic data: The data set in this experiment is the T10-I5-D1000K data set produced by the IBM synthesizer [10], where T, I, and D mean the average length of the transaction, the average length of the pattern, and the number of transactions.

b) Experimental data set: WEB access traffic: The WEB access traffic used for testing was collected at the network exit of the library of Huazhong University of Science and Technology. The collection program is connected to the egress gateway through the bypass, and the program filters and saves the traffic of the WEB service.

c) Evaluation index: In the comparative experiment, YARN's built-in Capacity strategy, Fair strategy, and dynamic planning improved RepRM scheduling strategy are used for resource scheduling, and parallel data mining using K-Means, KNN, and Apriori algorithms. We observe the performance of RepRM from multiple angles: For the mining of test data sets, we compare the throughput and memory usage of parallel mining. By comparing the data obtained from these two sets of experiments, we observe the different performance of the Capacity strategy, Fair strategy and RepRM strategy.

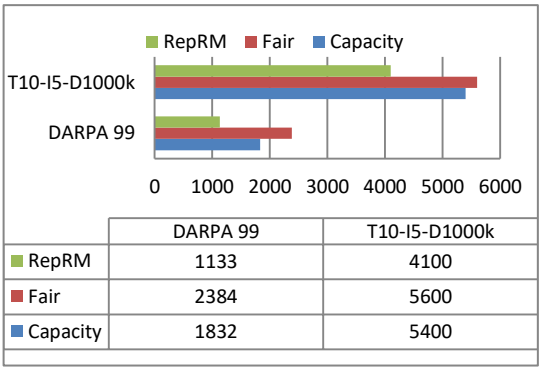
2) Mining experiment of IBM synthetic data set: In experiments based on data sets, we will pay attention to analyze the time complexity and space complexity of the algorithm. When comparing the two data, because the DARPA 99 data collective is at the GB level, and the T10-I5-D1000K is only at the MB level, in order to visualize it on the chart, we will use the test result value of the T10-I5-D1000K data set. It is 1000 times larger, so that it can be compared with the test results of the GB-level DARPA 99 data set.

In the time complexity test, the execution time of the mining model is compared. Fig. 2 shows the parallel mining results of the Capacity strategy, the Fair strategy and the RepRM strategy. It can be seen from Fig. 2 that the execution time of the RepRM strategy is lower than the Capacity strategy, what is more, it is also lower than the Fair strategy that Hadoop comes with. According to the data shown in the experimental results, compared with the Capacity strategy, the execution time of the RepRM strategy is about 70%, and compared with the Fair strategy, the execution time is about 65%. Therefore, the RepRM strategy increases the mining throughput rate by 25% compared with the Fair strategy and 30% compared with the Capacity strategy.

In the comparison of space complexity, the comparison is still based on the memory footprint of the algorithm. The specific data and comparison are shown in Fig. 3. The RepRM strategy has the least memory footprint, the Fair strategy has the middle memory footprint, and the Capacity strategy has the largest memory footprint. The experimental results show us the memory container consumption of the RepRM strategy is only 60%, and only about 53% compared to the Fair strategy.

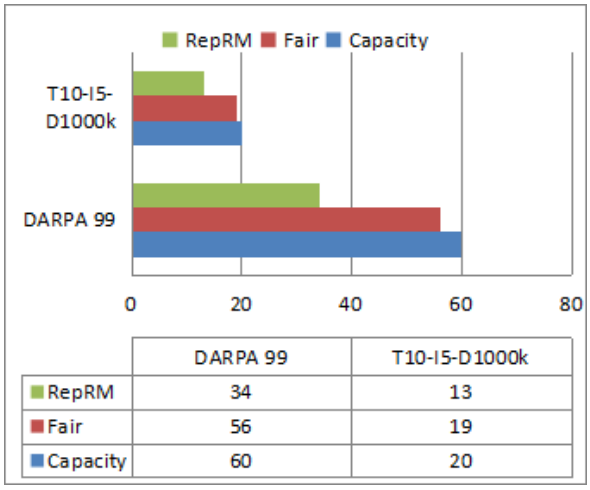
TABLE VI. Server Configuration of the Experiment

Model	**DELL FX2(Including 4 blade servers)**
Master	128GBRAM/ 2TBDisk/Operating System CentOS 6.5
Slave1-Slave3	64GBRAM/ 2TBDisk/Operating System CentOS 6.5
Hadoop version	Cloudera 2.2
Network environment	1000M



	DARPA 99	T10-I5-D1000k
RepRM	1133	4100
Fair	2384	5600
Capacity	1832	5400

Fig. 2. Execution Time Comparison for the Three Strategies in the Parallel Model.



	DARPA 99	T10-I5-D1000k
RepRM	34	13
Fair	56	19
Capacity	60	20

Fig. 3. Memory Footprint Comparison for the Three Strategies in the Parallel Model.

Through four sets of experimental data, it can be seen that the RepRM strategy improves the mining throughput rate by about 25-30% for the resource scheduling strategy that comes with YARN, and saves about 40% of resources in the consumption of memory containers. The reason is:

a) RepRM strategy is resource-adaptive. When the Capacity strategy and Fair strategy allocate tasks in Map-Reduce, they only allocate resources based on the existing remaining resources, rather than allocating tasks based on all running tasks. RepRM is based on the throughput of all tasks for dynamic scheduling and appropriate allocation of resources.

b) YARN's own strategy does not consider the impact of copies on mining throughput. For distributed mining under the condition of multiple copies, after the task is allocated by Map-Reduce, when there are more computing resources such as CPU and memory, it is necessary to wait for the network transmission of the copy. RepRM allocates computing resources based on the number of data copies, saving network transmission waiting time.

3) Mining test of network traffic: Before the start of the experiment, first, we use a data capture tool to capture and save the throughput network traffic from a certain WEB server within 240 hours; then we use the traffic replay toolkit to reproduce the saved traffic file; at the same time, the mining model in the cluster captures and mines traffic data, so that the stream data-parallel mining model can mine WEB traffic data.

Under normal circumstances, the traffic of the WEB server should be IP packets following HTTP, HTTPS and other protocols. If there is a network attack or intrusion (such as DDOS and TearDrop, etc.), the network traffic data in a certain period of time has certain abnormal characteristics, such as the statistical count of the source address in the seven-tuple. This abnormal characteristic is normal. There have a big difference in flow.

Due to the small traffic of the WEB server (KB-MB level), the simulation of a large traffic and large concurrent environment cannot be completed, so that the mining algorithm does not consume all the resources that can be allocated. Therefore, in specific processing, by collecting network traffic data within 240 hours, and replaying the traffic to the network within a few hours, considering the carrying capacity of the cluster, the memory usage of the mining model in large-traffic and large-concurrency environment is usually relatively large. In order to avoid the emergence of a deadlock, the maximum memory that can be allocated by each scheduling strategy is limited to 20GB.

Fig. 4 records the value of the mining throughput rate under the Capacity strategy, Fair strategy and RepRM strategy scheduling of the parallel mining model at different flow rates. Fig. 5 records the value of the memory usage. In Fig. 4, when the network data flow rate is low, the Capacity strategy, the Fair strategy and the RepRM strategy can effectively mine the data. With the increase of network data traffic, the throughput rates of the three resource scheduling strategies have gradually increased. When the maximum allocable memory condition of 20G memory is reached, the maximum throughput rate of the Capacity strategy, the Fair strategy and the RepRM strategy is about 3Gbps, 4Gbps and 6Gbps, respectively.

Fig. 5 shows the memory occupancy of the parallel mining model under the Capacity strategy, Fair strategy and RepRM strategy scheduling under different network data flow rates. As shown in Fig. 5, when the network data flow rate is low, the memory footprint of the three resource scheduling strategies is roughly the same, and the value is relatively low. At the same time, it can be seen that the Capacity strategy and the Fair strategy are similar, and slightly larger than the memory occupied by the RepRM strategy. In addition, with the increase of traffic, the memory consumption of the three strategies has gradually increased. When the flow rate reaches 4Gbps, the Capacity strategy and the Fair strategy have exhausted 20G of memory, and when the flow rate reaches 7Gbps, the RepRM strategy has exhausted 20G of memory.

The RepRM resource scheduling strategy considers the location of the data copy in the memory allocation, which can better eliminate the network transmission time of the data copy, so that the mining model can effectively mine. Based on the experimental results in Fig. 4 and Fig. 5, compared to the Capacity strategy and the Fair strategy, RepRM's resource utilization for parallel data mining has increased by about 40%.

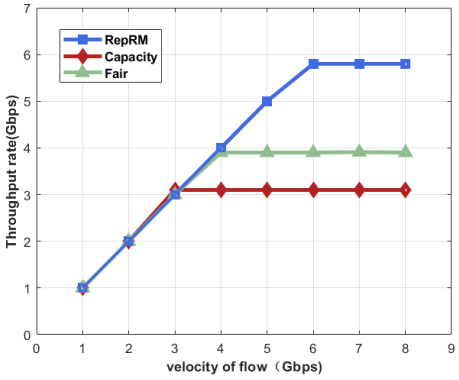


Fig. 4. The Effect of Flow Rate on Throughput.

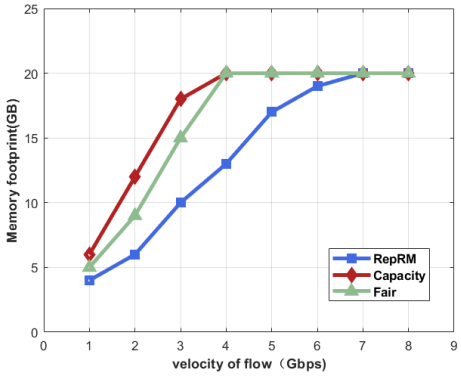


Fig. 5. The Effect of Flow Rate on Memory Consumption.

Table VII shows the accuracy and window size of parallel mining when using the RepRM strategy to schedule resources under different traffic conditions. According to Table VII, when the traffic reaches 6Gbps, because the RepRM resource scheduling strategy has exhausted 20GB of available memory, the network data packet mining capability of the mining model cannot be increased anymore. After that the sampling frequency of the window can only be modified. Mining is carried out, but the sampling method of streaming data seriously affects the accuracy of mining, which makes the accuracy rate continue to decline.

TABLE VII. PARALLEL MINING EXPERIMENTS USING REPRM RESOURCE SCHEDULING

Model	**DELL FX2(Including 4 blade servers)**
Master	128GBRAM/ 2TBDisk/Operating System CentOS 6.5
Slave1-Slave3	64GBRAM/ 2TBDisk/Operating System CentOS 6.5
Hadoop version	Cloudera 2.2
Network environment	1000M

V. CONCLUSION AND FUTURE WORK

For the mining of massive streams of data, resource allocation has always been a hot research topic. Researchers use various models to schedule CPU, memory, bandwidth, etc., in order to achieve ideal mining results. Especially when performing distributed mining of massive flow data, reasonable resource scheduling can achieve better mining results. However, most researchers did not consider the impact of the location of the data copy on the mining effect, and did not use the data copy as a resource for scheduling.

According to this, for distributed data mining, this paper takes the data copy in the cloud platform as a resource and incorporates it in the resource scheduling strategy for consideration, and realizes a copy-aware resource scheduling strategy RepRM. The RepRM strategy uses data copies as data resources, memory as computing resources, and uses a dynamic programming method to uniformly schedule data resources and computing resources, to improve the adaptability of the mining model to the data and computing resources in the cloud platform. In RepRM, data copies are regarded as resources that need to be allocated. At the same time, in order to solve the problem of data copy ratio, this paper adopts the dynamic programming method to achieve the maximum mining throughput of the cluster.

Then, this paper conducts simulation tests on parallel mining of streaming data through experiments. The test results prove that the RepRM resource scheduling strategy proposed in this paper has obviously advantages compared to the original resource scheduling strategy of Hadoop itself. After the homogeneous cluster is scheduled through the RepRM strategy, memory resources are saved by about 40-50% during parallel mining of streaming data, and the throughput rate is increased by 20%-30%. After the heterogeneous cluster is scheduled through dynamic planning, the throughput of parallel mining of streaming data is increased by 30-40%, saving about 40% of memory resources.

We aim to extend the RepEM strategy to the use of heterogeneous clusters by taking both the data copy and memory resource as resources and use Lattice point method to solve the problem. What'more, we plan to add more kinds of datasets for further research.

REFERENCES

[1] Li Qiao, Zheng Xiao. Research Survey of Cloud Computing. Computer Science 2011, 38, 32-37.

[2] Chi Xuebin, Gu Beibei, Wu Hong, et al. Analysis of the development status of high-performance computer systems and platforms[J]. Computer Engineering and Science, 2013, 35(11): 6-13.

[3] Barddal J P, Bifet A. A Survey on Ensemble Learning for Data Stream Classification. Acm Computing Surveys 2017, 50, 23. 10.1145/3054925

[4] C. Franke. Adaptivity in Data Stream Mining[D]. University of California at Davis, 2009.

[5] Gomes H M, Bifet A, Read J, et al. Adaptive random forests for evolving data stream classification. Machine Learning 2017, 1-27. 10.1007/s10994-017-5642-8.

[6] Wael Khallouli, Jingwei Huang. Cluster resource scheduling in cloud computing:literature review and research challenges. The Journal of supercomputing 2021. 10.1007/s11227-021-04138-z.

[7] Ghodsi A, Zaharia M, Hindman B, Konwinski A, Shenker S, Stoica I. Dominant resource fairness: fair allocation of multiple resource types. Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation 2011, pp 323–336. 10.5555/1972457.1972490.

[8] Wang W, Li B, Liang B. Dominant resource fairness in cloud computing systems with heterogeneous servers. INFOCOM, 2014 Proceedings IEEE. IEEE 2014, pp 583–591. 10.1109/INFOCOM.2014.6847983.

[9] Khamse-Ashari J, Lambadaris I, Kesidis G, Urgaonkar B, Zhao Y, Per-server dominant-share fairness (ps-dsf): a multi-resource fair allocation mechanism for heterogeneous servers. 2017 IEEE International Conference on Communications (ICC). IEEE 2017, pp 1–7. 10.5555/1972457.1972490.

[10] Niu Z, Tang S, He B. Gemini: An adaptive performance-fairness scheduler for data-intensive cluster computing. 2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom). IEEE 2015, pp 66–73.10.1109/CloudCom.2015.52.

[11] Shao Y, Li C, Gu J, Zhang J, Luo Y. Efficient jobs scheduling approach for big data applications. Comput Indus Eng 117 2018. 249–261;10.1016/j.cie.2018.02.006.

[12] Ali Shakarami, Mostafa Ghobaei-Arani, Ali Shahidinejad, Mohammad Masdari, Hamid Shakarami. Data replication schemes in cloud computing: a survey. Cluster Computing 2021, pp 2545-2579. 10.1007/s10586-021-03283-7.

[13] Najme Mansouri, Mohammad Masoud Javidi. A review of data replication based on meta-heuristics approach in cloud computing and data grid. Soft Computing 2020, pp 14503-14530. 10.1007/s00500-020-04802-1.

[14] Cui L, Zhang J, Yue L, Shi Y, Li H, Yuan D. A genetic algorithm based data replica placement strategy for scientific applications in clouds. IEEE Trans Serv Comput 2018, 11(4):727–739. 10.1109/TSC.2015. 2481421.

[15] Chunlin L, Ping WY, Hengliang T, Youlong L. Dynamic multi-objective optimized replica placement and migration strategies for SaaS applications in edge cloud. Future Gener Comput Syst 2019, 100:921–937;10.1016/j.future.2019.05.003.

[16] Huang X, Wu F. A cost-effective data replica placement strategy based on hybrid genetic algorithm for cloud services. International conference on research and practical issues of enterprise information systems 2018, pp 43-56. 10.1007/978-3-319=99040=8_4.

[17] Khojand M, Fatan Serj M, Ashrafi S, Namaki V. Predicting dynamic replication based on fuzzy system in data grid 2018. arXiv:1804.02963

[18] Salem R, Salam MA, Abdelkader H, Awad A, Arafa A, An artificial bee colony algorithm for data replication optimization in cloud environments. IEEE Access 2019, 7:1–12. 10.1109/ACCESS. 2019.2957436.

[19] Christiansen H. A survey of adaptable grammars. Sigplan Notices 1990, 25,35-44. 10.1145/101356.101357.

[20] Li Zhilin, Ou Yigui. Mathematical modeling and analysis of typical cases. Beijing: Chemical Industry Press, 2007.10.3969/j.issn.1000-4076.2006.05.035.

[21] Ming, Sun Lingyu, Zhu Ping. Research on the Load Balancing Task Scheduling Algorithm Based on Cellullar Automata in Cloud Computing. Small Microcomputer System 2016, 37, 2212-2216.