Research on Multi-task Scheduling Algorithm for Integrated Energy Metering Acquisition System

Xia Xiong*, Xiaofeng Tao and Xueliang Yang
Nari Group Co., Ltd /State Grid Electric Power Research Institute, Nanjing 210000, China

*Corresponding author e-mail: xiongxia0105@163.com

Abstract. With the development of the energy Internet and the construction of a new energy measurement system, it is particularly important to promote the remote automatic collection and copying of water, gas, heat and electricity. Based on the electricity information collection system, this paper designs an integrated acquisition technology architecture system for electricity, water, gas and thermal energy measurement, and clarifies the technical route of the energy measurement system. Then, through the analysis of the system task classification, a multi-task adaptive real-time scheduling method based on Spark cluster is proposed, which solves the problem that the collection system has uneven task allocation, long overall task completion time and high priority based on the existing scheduling mechanism of the Spark cluster. The task timeout response causes problems such as poor user experience. Finally, the experimental results show that the total number of 2 million collection tasks are divided into six categories, and the default scheduling methods in the text and Spark are used for comparison. The time-based and priority-critical tasks are completed earlier than the default method. The response speed greatly improves the system user experience and effectively improves the level of social public services.

1. Introduction
Electricity, water, gas and heat are indispensable public service energy products for people's daily life. The measuring equipments involved mainly include electric energy meters, water meters, gas meters and heat meters, collectively referred to as “civil four watches”[1]. With the National Development and Reform Commission, the United Nations Energy Bureau and the Ministry of Industry and Information Technology issued the "Guiding Opinions on Promoting the Development of "Internet +" Smart Energy" in February 2016 (Development and Reform Energy [2016] No. 392), and proposed the development of intelligent terminals for the energy Internet. The measurement system and its supporting equipment realize real-time measurement, information interaction and active control of energy consumption such as electric energy, heat and refrigeration. Enrich the implementation function of the advanced measurement system of intelligent terminals, and promote the remote automatic collection and copying of electricity, water, gas and heat to achieve multi-table integration.

With a large number of water, gas, and heat metering points connected to the system, how to use the more flexible and reliable acquisition pre-task scheduling algorithm based on the task scheduling of the current power collection system, especially when the system scales [2] When it is getting bigger and
bigger, the efficiency of task scheduling directly affects the system’s acquisition success rate. Based on the functional requirements of the integrated energy integrated acquisition system architecture, this paper analyzes the multi-task structure features in detail, proposes a new multi-task real-time scheduling algorithm principle and implementation flow, and gives the results of the scheduling simulation experiment of 2 million terminal tasks [3]. The feasibility of the algorithm improves the cluster service of dozens of acquisition servers, realizes multi-task adaptive real-time scheduling strategy, improves the task response speed with high priority, and greatly improves the user experience of the system[4] [5].

2. System architecture overall design
The integrated energy integrated acquisition system consists of six subsystems: communication pre-configuration, real-time business analysis, background statistical service, intelligent analysis, visual display and external interface services [6]. As shown in Figure 1.

Figure 1. System architecture diagram

The specific functions of each subsystem are described as follows:
The communication front-end subsystem adopts flexible architecture design to quickly collect data from millions of terminals and billion-level water, electricity, gas, heat and other metering devices from different communication channels, and supports more than 1 billion records per day. The communication channel is mainly a 230MHz wireless private network built by the power grid company,
a wireless public network such as GPRS, and a fiber-optic private network; the collection device is a collection of on-site metering equipment and collection terminals, mainly including an electric energy meter, a concentrator, and a special-purpose terminal. For other devices, the communication methods used are mainly narrowband carrier, wideband carrier, micro power wireless, 485 bus, etc.

The real-time business analysis subsystem applies stream processing and caching technology, analyzes all kinds of uploaded information in time, realizes monitoring of smart meters, terminals, distribution transformers and lines in the field, and timely discovers energy supply abnormalities and faults. Water, electricity, gas, and heat failures are quickly repaired. The background statistical analysis module performs efficient calculation and rapid processing by hourly and daily data.

Based on the statistics of each dimension, the intelligent analysis subsystem builds user and equipment tag libraries, builds models of energy behavior analysis, equipment health analysis and spare parts prediction, and analyzes the overall operational status of the metering device, fault prediction and health management.

The visual display subsystem is based on the application of different organizations to the system and the individualized needs of the data, and the self-customized visual data display platform realizes panoramic and visual display of all analysis results and operational data through the visualization module.

The external interface service subsystem establishes an interaction model and business process with the power marketing and other municipal public service systems through the external interface service, and is used for accessing the file information and using the data that can collect and analyze the results.

After the integrated energy integrated acquisition system is put into operation, it can greatly improve the collection, calculation and analysis performance of the main station system, expand the business scenario application, enrich the user experience, realize the transformation from experience-driven to data-driven, and support business innovation and process optimization. Provide data collection for government agencies and water supply and gas companies, support the government's macroeconomic regulation and control, energy conservation and emission reduction policies, and meet the needs of public institutions. However, with the deepening of system construction and the expansion of user acquisition coverage, the number of energy measurement points has doubled, the volume of collected data is huge, the frequency of collection is increased, and the external shared interface is increased. Both of them have proposed the concurrent processing capability of the acquisition system. The new challenge, the next section focuses on the multi-tasking structure of the system, proposes a multi-task real-time scheduling method, and greatly reduces the collection pressure of the system.

3. System multitask structure
The integrated energy integrated collection system realizes remote automatic collection of various energy users' measurement information, energy use information, abnormal information, energy quality information, and efficient data analysis and management. The system task management cluster module is responsible for remote scheduling of various terminal tasks and is responsible for protocol parsing. The pre-task module angles fall into two categories: pre-automatic tasks and external tasks. The pre-automatic task refers to the data calling task of the internal running of the collection server itself, including real-time, history and remediation tasks. External tasks include web call tasks, interface tasks, and background tasks. System pre-task scheduling architecture as shown:
Figure 2. System pre task scheduling architecture diagram

The pre-configuration task scheduling strategy is to uniformly schedule external tasks by the scheduler. After the task is generated, it is placed in the distributed cache task queue (that is, the shared task queue in the distributed cache), and each acquisition server is scheduled. Get the tasks of their own machines and execute them. The pre-automatic task has the characteristics of high acquisition frequency. The pre-cluster itself generates tasks and assigns them to each node of the collection cluster. It does not go through the shared task queue in the distributed cache and directly puts into the task queue of the node. This node is responsible for implementation. The terminal status queue records the task execution status of each terminal, including whether the terminal is performing tasks, execution time, which predecessor node, communication mode, and task clear flag, and each node shares the queue, when the node executes Or complete a task, the terminal status list needs to update the task status and clear the task after the task is completed.

The pre-automatic tasks are highly concurrent and high, and the task response priority, the load of the collection server node, and the resource utilization are changing with time. The collection server has uneven task assignment based on the existing scheduling mechanism of the Spark cluster. A task with a long overall task completion time and a high priority task has a poor user experience. Therefore, a new scheduling method is proposed to solve this problem, and the elastic scheduling of the collection server is realized. The purpose of task scheduling is to assign system tasks to appropriate acquisition server nodes based on priority [7] [8]. Usually task scheduling indicators have the shortest scheduling length, load balancing, economic principles and quality of service [9] [10] [11]. In the energy metering and integration elastic collection architecture, task scheduling mainly considers the shortest scheduling length and load balancing, and can be re-allocated in the next scheduling when a new acquisition server is online or offline, thus achieving integration. Flexible acquisition architecture plug and play.

4. Spark cluster task scheduling mechanism

4.1. Spark platform architecture
A Spark cluster mainly includes a master node and a worker node. The Master node is mainly responsible for managing the cluster, and the Worker node is mainly responsible for the execution of tasks. Currently, Spark supports different deployment methods. Because Spark is a memory-based
iterative computing framework, it is ideal for applications that require multiple operations on specific data sets [12] [13].

4.2. Spark task scheduling strategy
When the Spark system executes multiple tasks at the same time, the task scheduling needs to assign multiple tasks to the appropriate nodes for execution. The default task scheduling policy in Spark is as follows:

(1) The Spark task scheduler randomly selects some nodes from the registered Worker nodes.
(2) Traversing the nodes selected in the first step and assigning tasks to the highest localized task execution node.

4.3. Spark problem
The above Spark task scheduling strategy assumes that the cluster nodes are isomorphic, That is, the node resources are basically the same. However, in a deployment scenario such as a data center, as time goes by and the amount of traffic increases, the cluster is likely to have greater heterogeneity. At the same time, the load and resource utilization of each node is dynamically changing. In a heterogeneous Spark cluster, the current communication capabilities of each node are very inconsistent. The task scheduling strategy of the Master node adopts FIFO, that is, first-come-first-served and FAIR fairness strategy, without considering the difference between the worker node and the resources required by the task[14] [15] [16], this scheduling strategy will bring the following 2 questions:

(1) Nodes calculate the relative imbalance of task assignments, that is, high-configuration nodes are often starved, while nodes with lower configurations are often at full load.
(2) Limited overall system communication capacity and long overall task completion time, low performance.
(3) Tasks initiated by the user manually manipulating the page are usually queued for execution, causing a timeout response, and the user experience is poor.

According to the actual application in a long period of time, combined with the various tasks of water, electricity, gas and heat, the success rate of each task, the success rate of each data item, the benchmarking of the industry, the stability of the system and the adjustment of the operating parameters of the system. According to the reference standard, it is found that the time-sensitive task of the pre-communication module of the energy integrated acquisition system is not responded due to timeout, resulting in poor user experience [17]. Therefore, the paper proposes to adopt multi-task adaptive on the basis of Spark cluster. The scheduling policy automatically adjusts the execution order of the tasks, optimizes the execution of the collection and setting tasks, and implements the tasks with higher priority in time to further improve the execution efficiency of the tasks.

5. Multi-task adaptive real-time scheduling strategy

5.1. Multi-task adaptive real-time scheduling design idea and system scheduling architecture
Considering the situation of heterogeneous nodes in the cluster, the main design ideas of the multi-task adaptive real-time scheduling strategy proposed in the paper are as follows: (1) In a running Spark cluster, each worker node can change according to its own resources and load. Regularly adjust the weight of the task that is selected to perform it. Select CPU utilization, memory utilization, and single-core average queue length as indicators to measure the load and resources of the node to calculate the weight. (2) When the master node performs task scheduling, the weights of each node are read, and the nodes with larger weights are preferentially selected. The specific process is shown in the figure:
Each worker node periodically detects its own load and resources through the node monitoring module, and reports it to the weight adjustment module. The weight adjustment module determines whether the weight needs to be adjusted within a period of length N. When the master node needs to perform task adjustment, first read the current weight of the Worker node, and preferentially select some nodes with larger weights for task assignment.

5.2. Intelligent scheduling strategy
According to the terminal protocol, communication mode, user category and collected data items, different types of tasks are generated for the energy integration acquisition system. According to the execution of the entire system task, the task execution time is reasonably scheduled, and the system channel resource usage is balanced. At the same time, the task status is monitored, and the task execution strategy is dynamically scheduled to ensure efficient execution of the task. Multi-task adaptive intelligent scheduling architecture is shown in Figure 4.

1) The intelligent task scheduling service adopts distributed architecture design, and uses message middleware, scheduling management, caching and other technologies to achieve loose coupling with big data platform, relational database, pre-location, web and interface, and solves the problem of efficient execution of millions of tasks. Problem.
Figure 4. Multitask adaptive intelligent scheduling architecture

2) Use ZooKeeper to manage nodes, use message channels to parallelize node tasks, support elastic dynamic expansion, and implement high-performance task distribution scheduling strategy.

3) According to the real-time tasks of the Web and interfaces, automatic collection tasks, replenishment tasks, transparent task importance and priority levels, intelligent generation of task execution strategies, intelligent task management, efficient generation and rapid distribution execution, supporting energy integrated acquisition system Intelligent collection management and control functions.

5.3. Acquisition multi-task adaptive real-time scheduling algorithm implementation process

The process of collecting multi-task adaptive real-time scheduling algorithm is shown in Figure 5.

Detailed steps are as follows:

Step1: When the system front communication module receives the current new joining task, the initial parameters of the task are calculated, and the task is expressed by a formula a:

$$ T_i = \langle a_i, d_i, V_i, E_i, ved_i \rangle $$

\( i \) is task number, \( a_i \) indicates the arrival time of the task, that is, the time to enter the queue. \( d_i \) indicates the validity period of the task, which means that the task must be completed at the specified time and output the result. Otherwise, the task is invalid and discarded;
Figure 5. Multitask adaptive real-time scheduling algorithm process

$V_i$ Indicates the importance level of the task, and divides the importance of the task into five levels: very important, important, general, unimportant, and very unimportant. The values 5 to 1 respectively indicate the degree of urgency, 5 are very urgent, and then decrease in turn; The priority of the charge control task, parameter task, interface data call test task and automatic call test task in the acquisition system is from high to low;

$E_i$ Indicates the urgency of the task. The urgency of the task is divided into five levels: very urgent, urgent, general, non-emergency, and very urgent. The available values are 5 to 1 for the degree of urgency, and 5 for the emergency. In general, the interface data call task initiated by the user through the page is closely related to the user experience. The level is classified as very urgent, and the automatic call test task has the longest time interval, and only needs to be completed at the deadline, and the emergency level is divided. Very urgent;
Indicates the value density of the task, which is the ratio of the value of the task to the time required for the task to be executed, that is, the value of the task in the unit time. The value of the task is the product of the urgency and importance of the task. The value density is calculated as:

\[\text{ved}_i = \frac{(V_i \times E_i)}{d_i}\]  (2)

Step2: According to the validity period and value density of the task, a two-dimensional priority table of multi-task scheduling is established, as shown in Fig. 6, the establishment method is: the horizontal axis represents the value density of the task \(\text{ved}_i\), from left to right, the value density gradually decreases, corresponding to the priority descending; The vertical axis indicates the validity period of the task \(d_i\). From top to bottom, the validity period gradually increases, corresponding to the priority descending order; from Figure 6, it can be seen that the task priority depends on the value density and validity period.

Step3: Scheduling the two-dimensional priority table by multitasking and calculating the priority \(P\) of all tasks in the queue according to formula (3):

\[p = \frac{(m+n-1)(m+n-2)}{2} + m\]  (3)

Where \(m\) and \(n\) represent the coordinate positions of the corresponding validity period and value density of the task in the two-dimensional priority table (Fig. 6), respectively.

Step4: After the priority pre-processing of the task is completed, the tasks are sorted according to the priority. When the tasks have the same priority, according to the arrival time of the task \(a_i\), the first-time task first joins the task queue of the pre-communication module; the current communication module If the number of queues to be processed has reached the predetermined value, enter the SPARK cluster master node scheduling center, otherwise continue to loop steps 1 - 4;

Step5: The SPARK cluster task scheduling center adopts an adaptive task scheduling strategy. Each worker node can dynamically adjust the node weights based on CPU utilization and memory utilization.
as a measure of node load and resources. The master node has a higher priority. Worker node performs tasks.

CPU utilization is calculated by two consecutive CPU sample values, the definition formula is as follows:

\[ CU_k = \frac{CPU_{time_{k2}} - CPU_{time_{k1}}}{CPU_{Total_{k2}} - CPU_{Total_{k1}}}, k = 1, 2, \ldots \] (4)

\( CU_k \) indicates the CPU utilization of the kth real-time monitoring, \( CPU_{time_{k2}} \) represents the CPU working time of the second sampling at the kth monitoring, \( CPU_{time_{k1}} \) represents the CPU working time of the first sampling at the kth monitoring, \( CPU_{Total_{k2}} \) indicates the sum of CPU working time, idle mode, disk I/O wait, and hardware interrupt time for the second sampling at the kth monitoring; \( CPU_{Total_{k1}} \) indicates the sum of CPU working time, idle mode, disk I/O wait, and hardware interrupt time for the first sampling at the kth monitoring.

The memory utilization definition formula is as follows:

\[ MU_k = \frac{Total_k - Free_k - Buffer_k - Cache_k}{Total_k}, k = 1, 2, \ldots \] (5)

\( MU_k \) Represents the memory utilization of the kth real-time monitoring. \( Total_k \) is the total memory value of the kth real-time monitoring. \( Free_k \) is the free memory value of the kth real-time monitoring. \( Buffer_k \) is the buffer size of the k-th real-time monitoring Buffer, \( Cache_k \) is the Cache cache size for the kth real-time monitoring.

The weight calculation formula of the worker node of the kth real-time monitoring is as follows:

\[ Capacity_k = \frac{CU_k + MU_k}{2} \] (6)

By judging the weights of all the Worker nodes in the SPARK cluster, sorting by weight, the Master node selects a predetermined number of Worker nodes to perform the task.

Step 6: Assign the task newly added to the SPARK cluster master node scheduling center to the worker node selected in step 5, and the task enters the corresponding node for execution.

### 6. Experiment and result analysis

In order to verify that the multi-task adaptive real-time scheduling method can meet the functional requirements of the integrated energy integrated acquisition system, this method is compared with the Spark default scheduling strategy, and the comparison performance is analyzed under the real environment.

This article uses Spark-0.8.0 version, Kafka stream processing platform. Kafka is a high-throughput distributed publish and subscribe messaging system that handles all action flow data in a consumer-sized website. Simulated 2 million acquisition tasks on the Kafka platform, the statistics of the collection task types are as follows:
Table 1. Task type quantity statistics

| Task type       | Web task | Interface task | Background task | Real-time automated task | Historical automatic task | Historical revocation task |
|-----------------|----------|----------------|-----------------|--------------------------|--------------------------|---------------------------|
| Number          | 337821   | 341874         | 325460          | 335471                   | 331343                   | 328031                    |
| Priority        | 5        | 4              | 4               | 3                        | 2                        | 1                         |

A heterogeneous Spark cluster is built using servers of different configurations, each server serving as a node of the cluster with a total of 10 servers. One of the servers acts as the Master, nine servers act as the Worker, and the Worker node has two types of configurations. The experiment used an open source cluster monitoring project, Ganglia, initiated by the University of California at Berkeley to monitor the cluster status and sort the data of different data volumes to measure the performance of the method.

Experiment 1 The purpose of this group of experiments is to verify that the method in this paper is compared with the default scheduling strategy, and whether different priority tasks have a significant improvement in the completion time of the job. The experiment will prioritize different data sets, which are Web task, interface task, background, real-time automatic task, historical automatic task, and historical call task randomly interspersed. At the same time, two methods are used for scheduling execution. In order to ensure the reliability of the experimental results, the task assignment experiment is operated 5 times, and the time difference obtained by subtracting the default method from the method completion time is used for statistics. The statistical information of the execution time difference of each group of tasks is shown in Table 2 below:

Table 2. Data statistics of task completion time difference experiment

| Experiment type and acceptance number | NO.1(s) | NO.2(s) | NO.3(s) | NO.4(s) | NO.5(s) |
|--------------------------------------|---------|---------|---------|---------|---------|
| Web task                             | -1809   | -2087   | -1980   | -2103   | -2306   |
| Interface task                       | -1221   | -1830   | -1364   | -1876   | -1901   |
| Background task                      | -2763   | -1965   | -2034   | -2127   | -2225   |
| Real-time automated task             | +763    | +1657   | -922    | +1645   | +2441   |
| Historical automatic task            | +1870   | +1818   | +3382   | +2012   | +1767   |
| Historical revocation task           | +3160   | +2398   | +2918   | +2449   | +2224   |

As can be seen from the table, the real-time scheduling method proposed in this paper is for the higher priority tasks, the task time of the interface task and the background task are earlier than the default method, and for the lower-priority real-time automatic task, historical automatic The completion time of the task and historical revocation tasks is later than the default method, which indicates that the method of this method is preferentially executed for the higher priority tasks, and the corresponding speed is obtained, which greatly improves the system user experience.

Experiment 2 the purpose of this group of experiments is to verify the time and distribution rate of the 2 million task allocation in comparison with the default scheduling strategy. Tasks are randomly interspersed, and two methods are used for scheduling execution. To ensure the reliability of the experimental results, the task assignment experiment was performed 5 times. The experimental calculation calculates the distribution rate of each experiment as shown in Table 3 below:

Table 3. Task scheduling rate statistics

| Experiment NO. | 1     | 2     | 3     | 4     | 5     |
|----------------|-------|-------|-------|-------|-------|
| Allocation time(ms) | 20271 | 21900 | 19835 | 20491 | 20769 |
| Distribution rate(num/s) | 97461 | 103928| 103811| 97819 | 92214 |
It can be seen from Table 3 that the average allocation rate of the multi-task adaptive real-time scheduling method is 99,046 per second. At this rate, the integration of 80 million metering tasks can be completed in 14 minutes.

Through the above simulation experiments, the feasibility of the multi-task adaptive real-time scheduling algorithm proposed in the energy metering integrated task scheduling mechanism is proved. By monitoring the task situation, reasonably dynamically scheduling the task execution strategy and balancing the system channel resource usage to ensure the task is efficient. Carried out.

7. Conclusion
Carry out the research and application of the integrated energy integrated collection system, clarify the technical route of the collection system, effectively promote the intelligent development of water, gas and heat metering products, and realize the simultaneous improvement of the energy measurement industry. In view of the integration of multi-type heterogeneous energy metering, this paper mainly made the following research:

(1) Based on the electricity information collection system, through the overall design of the system architecture, six subsystems of communication pre-configuration, real-time business analysis, background statistical services, intelligent analysis, visual display and external interface services are defined to clarify the technical route of the energy metering system. Provide technical support for the efficient and socialized services of public utilities such as water, gas and heat.

(2) Through the analysis and introduction of the multi-task structure of the system and the scheduling mechanism of the default Spark cluster task, the existing scheduling strategy has the tasks of uneven task allocation, limited overall computing power, long overall task completion time and high priority. Timeout response causes problems such as poor user experience.

(3) A multi-task adaptive real-time scheduling strategy is proposed, and it is proved by experiments that the method is implemented preferentially for time-sensitive tasks, which greatly improves the user experience and makes the system cluster architecture have a good distribution average.

The architecture design and task scheduling algorithm of the integrated energy integrated acquisition system studied in this paper proposes a multi-task adaptive real-time scheduling method, which can effectively solve the collection problem of large-scale heterogeneous energy, and the traditional power collection system when the scale of the acquisition node is expanded. The task timeout response with uneven allocation and high timeliness in the collection of task assignments leads to poor user experience.

References
[1] SUNG Kai, LV Xiaoqiang. Research on Four-in-One Technology of Electricity Information Acquisition System [J]. China Plant Engineering, 2017 (5): 141 - 142.
[2] YU Bo. The Design and Implementation of Electric Information Acquisition System [D]. School of Automation Engineering, 2016.
[3] HU Jiangyi, ZHU Enguo. Application Status and Development Trend of Power Consumption Information Collection System [J]. Automation of Electric Power Systems, 2014 (2): 131 - 135.
[4] PEI Zhongai. Design of remote meter reading system for water, electricity, gas and heating [D]. Shandong University of Science and Technology, 2004.
[5] Xu Qing, Liu Jian. Research and Application on Data Acquisition System about Water, Heat, Gas, and Electricity Meters in One [J]. Computer Measurement & Control, 2017, 25 (3): 217 - 221.
[6] KANG Liyan, WANG Tianbo, etc Design and Implementation on Distributed Elastic Structure of Electric Customers Information Acquisition System [J]. Northeast Electric Power Technology, 2018: 10: 42 - 45.
[7] LV Kunming, JIAO Yanan, WANG Dong, etc. Research on Key Technologies of Cloud Computing Based on Associated Task Scheduling [J]. Industry and Technology Forum, 2018 (4).
[8] ZHAO Chunyan. Research and Implementation of Job Scheduling Algorithm in Cloud Environment [D]. Beijing Jiaotong University, 2009.

[9] Ghanbari S, Othman M. A Priority Based Job Scheduling Algorithm in Cloud Computing [J]. Procedia Engineering, 2012, 50 (9): 778 - 785.

[10] Xu B, Zhao C, Hu E, et al. Job scheduling algorithm based on Berger model in cloud environment [J]. Advances in Engineering Software, 2011, 42 (7): 419 - 425.

[11] Anitha A. Job Scheduling Using Coupling in Grid [J]. Journal of Computer & Communications, 2015, 3 (10): 1 - 12.

[12] Apache Spark. Spark Overview [EB/OL]. http://spark. Apache. org, 2015-01-21/2015-03 - 18.

[13] The Spark Software Foundation. Spark [EB/OL]. [2015-01-08]. http://spark. apache. Org.

[14] BIAN Chen, YU Jiong, etc. Self-Adaptive Strategy for Cache Management in Spark [J]. Acta Electronica Sinica, 2017: 2 (45): 278 - 283.

[15] YANG Zhiwei, ZHENG Quan, etc. Adaptive Task Scheduling Strategy for Heterogeneous Spark Cluster [J]. Computer Engineering, 2016: 1 (42): 31 - 37.

[16] BIAN Chen, YU Jiong, etc. Progressive filling partitioning and mapping algorithm for Spark based on allocation fitness degree [J]. Journal on Communications, 2017: 9 (38): 133 - 147.

[17] FENG Lin. Research and implementation of Memory optimization based on parallel computing engine spark. [D]. Beijing: Tsinghua.