ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

Research on Data Engine of Space Command and Display System Based on Flink Task Scheduling Strategy

Wenhao Zhang*, Gan Zhou, Zhifeng Yin, Kunlun Hu

The 6th Research Institute of China Electronics Corporation, Beijing, 100083, China
**Corresponding author: 13869693368@163.com*

Abstract: *Integrated Command and display system is the command hub of space launch center. In recent years, with the rapid development of China's space industry, the number and types of launch missions are increasing, and the data become diverse and complicated. In the face of the huge ocean of information, this paper designs a data acceptance and processing engine, adopts the microservice architecture, and designs a task scheduling algorithm different from the default scheduling strategy of flink, including the pre-scheduling prediction algorithm of WSVM and the resource.*

Keywords: *Stream Data Processing, Data engine, Flink, Analytic hierarchy process, WSVM, Task scheduling, Resource scheduling*

1. Introduction

With the development of the space industry, space missions with new payload types continue to appear, and the data that need to be processed is also increasing. It is necessary to collect the real-time monitoring data of various subsystems of the launch site, such as measurement and control, measurement, meteorology, communication and service, and convert it into graphics, images, words, curves and other more intuitive visual information for the staff of the command center. Therefore, the integrated development of space launch command and display system is put on the agenda [1-5].

First, the Command and display system version is not unified, and it is difficult to upgrade and maintain. Second, the amount of data is large, resulting in data processing delay. Third, the original single-threaded architecture has become a performance bottleneck.

A new command and display system is designed and implemented according to the principles of real-time, ease of use, reliability, maintainability and expansibility. The system can display data information in text, table, curve, two-dimensional and three-dimensional situation and other ways, and add a variety of page navigation to improve the man-machine experience effect. It provides a source of information for commanders and operators to make decisions.

2. Design of Command and display system data engine

2.1 Cluster architecture

The monolithic service architecture was divided into multiple services and deployed to multiple servers, and the high availability of the overall command and display system is guaranteed through the connection and management between the services. Each microservice can be independently deployed to the production environment and test environment, which enhances the scalability of the service.

At the same time, containerization technology is used to coordinate configuration with microservice architecture, which realizes application isolation, reduces the configuration of each service, and improves the efficiency of testing and operation and maintenance. This article uses Docker to deploy the service. Kubernetes is used as a container orchestration tool to realize container startup, deployment, online upgrade and other functions, and make full use of system resources.

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

2.2 System architecture

2.2.1 Command and display system architecture

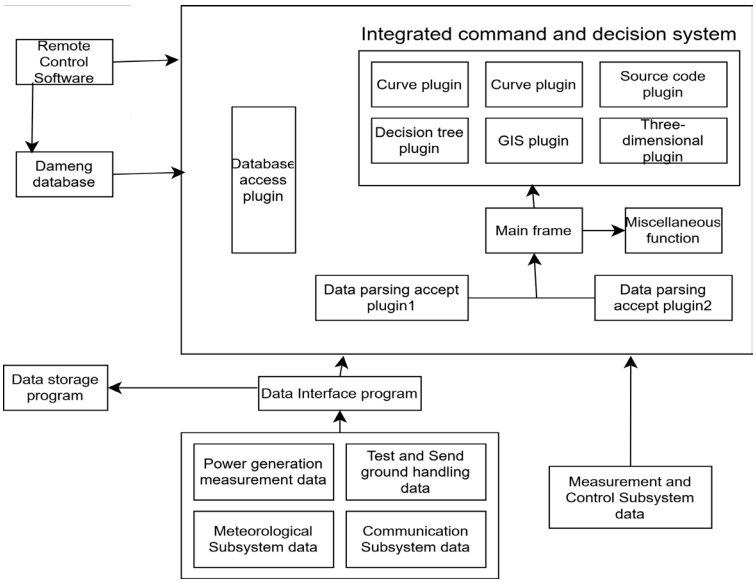

Figure 1. Command and display system architecture

Based on the above limitations of the command and display system, this paper designs the overall framework of the aerospace integrated command and display system, and focuses on the design and implementation of a data engine for the basic space launch integrated command and display system. The command and display system architecture is shown in Figure 1.

(1)Data engine layer: The main functions include data access, processing, computing, storage, analysis and distribution. Through close cooperation with the interaction layer, external data can be received and parsed effectively. Distributed computing and parallel processing technology are used to realize efficient data processing and calculation, and the results and raw data are stored in a reliable and easily scalable data storage layer.

(2)Data receiving layer: In the input part of the whole data engine system, all kinds of external data are stored to the database through the data interface program, and the data is transmitted to the data acceptance and analysis plugin. The main framework can obtain data from the data acceptance and analysis plugin, and can also obtain data from the database access plugin.

(3)Main frame: It is the central hub of the command and display system function, receives real-time data and historical data, monitors each plugin, warns in time and warns the problem plug-in, manages all kinds of plugins, and finally passes all kinds of processed data to the display layer.

(4)Display layer: Mainly responsible for visualizing the processed data, including curve plugins, table plugins, source code plugins, equipment status tracking plugins, two-dimensional GIS plugins, three-dimensional situation components, decision tree components, etc.

(5)Service layer: The remote control software can make service calls through the permission information configured in the database. It can also achieve timely communication, fast navigation, fast navigation and other functions.

2.2.2 Data Engine Architecture

(1)The software has the ability to receive various types of data that can access the major bases of the finger display system. Data from different bases, different tasks, and different systems generally come from different network transmission methods, different network addresses, or different port numbers. The software supports users to add, delete and modify interface configuration information in a flexible and extensible way according to their different requirements for data sources. Users can name new interface information (weather system), so that users can intuitively see which system and task the received data comes from (Figure 2).

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

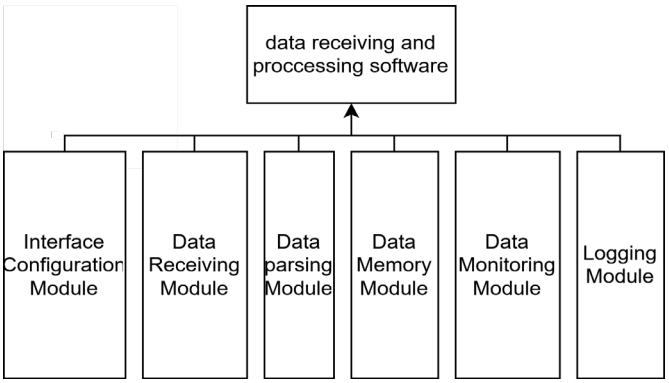

Figure 2. Data Engine Architecture

(2)Data validation, protocol conversion. Since the data sources and data formats of external systems are different, after the software accesses the data of major systems to the finger display system, it needs to perform protocol adaptation and convert it into the data format that can be directly used by the internal subsystems of the command and display system. Data access is the initial data interface of space launch command and display system, which gathers multi-source data information such as client request information, mission process, wireless and wired test, telemetry, physiological monitoring, weather, support facilities and communication system working status, and video of key areas, and provides original input for subsequent data processing flow.

Network packets. Through network monitoring, the software can obtain all kinds of special data such as measurement and generation measurement, measurement and development ground service, measurement and control, communication, weather, service, astronaut, command and dispatch from external systems, and the frame header format of each type of data is different. Some may not be uniform frame header formats. Through the custom development of different protocol conversion plugins, the heterogeneous data of various situations can be converted into the frame header information of the unified formal uniform format after the protocol adaptation, which is convenient for each internal subsystem to use the data.

Web page data information. The command and display software can not only access the webpage data through the form of embedded webpage, but also receive the webpage data forwarded by the processing software by receiving data, and display the webpage data in the form of self-developed display components. Through the custom web data protocol conversion plugin, the software captures the web data package information needed in the network, and converts the Web information into uniform format, so that the internal subsystems can directly use the converted Web data.

Text files. The software can parse various data formats in the text file, load the data in the file into memory, and convert it into a unified uniform frame header format and forward it to the internal subsystems of the command and display.

(1)Data forwarding. Through the configured forwarding information of each interface (including sending multicast address, sending multicast port number, local sending address, local sending port number.), the software forwards the data packets after protocol conversion and adaptation to each internal subsystem.

(2)Data storage. The software can classify all received data by date, task name, task phase, data source, data types and other dimensions, the data is stored in the specified source code file or the specified database table, which is convenient for users to query and playback historical data afterwards.

(3)Data monitoring. The software can monitor the status information of each interface receiving and storing data in real time, including the cumulative number of received packets, the cumulative number of sent packets, the receiving bandwidth, the sending bandwidth, the cumulative amount of stored data, link status and other information, and update and display in the form of table.It is convenient for users to understand the data receiving, sending and storage status of each interface in real time, and to have an intuitive and efficient understanding of the data throughput of the whole system.

(4)Data querying. The software supports users to query all received data in terms of data frame number and time period. The software can also perform fuzzy search according to parameter code, parameter name, parameter index number.

(5)Logging. The software can store the software running process, user operation records and other

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

information into the specified log file according to the different abnormal levels, and display the log record information in real time for users to read at any time.

(6)Traffic alerts. The software can monitor the flow of data received by each interface in real time through the upper limit of the flow threshold configured by each interface. When the traffic is close to or exceeds the upper limit of the threshold, the alarm information is given in time, which is convenient for users to query the problem.

(7)Duplex management. The data receiving and processing software supports dual hot backup. The software is deployed on two machines respectively, forming a relationship between master and backup, receiving and processing network information at the same time, but only the host sends information at the same time.

Dual machine switching principle: in automatic mode, the secondary machine does not switch when the fault, the main machine switch when the fault. In the manual mode, the main machine can be switched to the secondary machine, and the original secondary machine automatically becomes the main machine. The secondary machine can switch to the main machine, and the original host machine automatically becomes the secondary machine.

(8)Network verification. The software can periodically receive the time series information sent by the time server and compare it with the local time. If the time difference is greater than a preset value, the software will use the time information sent by the time server to correct the local time actively.

External interface: The data receiving and processing software receives all kinds of data information sent by various data sources of external systems (including C3I system, central computer system, communication system, meteorological system, astronaut system and other third-party systems, etc.), or simulation data and playback data information sent by data auxiliary support software. After a series of processing, the received data is forwarded to each command display software, Web display software and LED screen display control software in a relatively uniform format. The software can receive the command and dispatch information sent by the command display software and forward it to other external systems. The software realized the functions of starting and closing the software by receiving the control instructions sent by the remote control software. The software realizes online deployment, version update and other functions of data receiving and processing software by receiving deployment and upgrade instructions sent by system management and maintenance software.

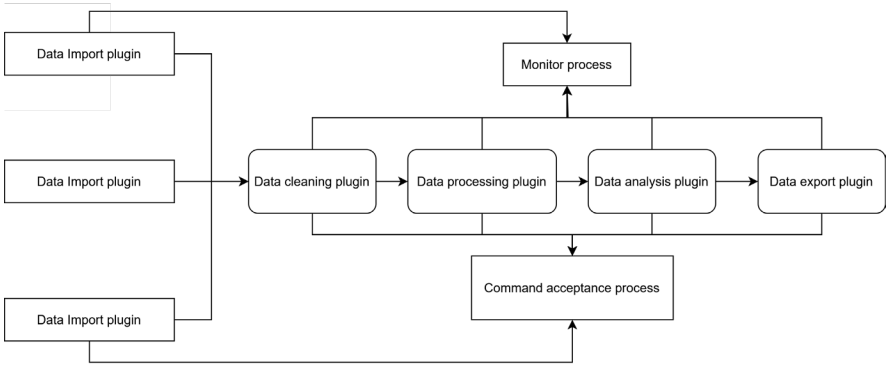

Figure 3. Data Engine Architecture

Internal interface: Through the interface configuration module, the software sends the receiving configuration information, protocol conversion plugin information and forwarding configuration information to the data receiving module, data parsing module and data forwarding module respectively. The data receiving module sends the received data to the data storage module for data storage, and the data query module waits for the user to perform data query operations. The data receiving module pushes the received original data to the specified data parsing module, executes the data protocol adaptation conversion, and sends the converted data to the internal subsystems through the data forwarding module. Data receiving, data parsing, data forwarding and other modules can push various status monitoring information to the data monitoring module, and generate log information through the log recording module to store in the log file. The network calibration module realizes the calibration function of the local computer by receiving the time series information(Figure 3).

When building the data engine, this study adopt the chain of responsibility pattern to abstract its diverse functions. As shown in the figure, the chain of responsibility pattern significantly improves the flexibility and scalability of the system by decoupling the sender and receiver of the request. In this mode,

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

the sender does not need to know which specific receiver the request will be processed by, and similarly, the receiver does not need to care about the origin of the request. This design allows the system to dynamically combine and adjust the processing order of objects, giving the flexibility to add, remove, or reorder objects in the chain as needed without modifying the existing code base. Such flexibility makes it possible to build diverse processing chains to meet different business requirements.

2.2.3 Flink architecture

This paper is based on the flink architecture for real-time data processing, continuous stream processing systems such as Storm and Flink model streaming applications as a topology. In a topological graph, vertices are called processing units and edges are called data streams. Each operator receives a sequence of tuples from the input queue, performs the computation logic, and places the result on the output queue. Edges represent the flow of data from one operator to the next. In the process of operator deployment, each operator is extended to multiple parallel tasks, and tuples are distributed between upstream and downstream operators through the stream grouping mechanism to achieve load balancing. A Flink cluster consists of three types of components: Flink clients, job manager, and task manager. The Flink client receives and analyzes the application code, converts it into a dataflow graph, optimizes the graph accordingly, and submits the dataflow graph representing the application to the job manager. The job manager is mainly responsible for coordinating the distributed execution of the dataflow graph. It parses the logic of operators and the communication relationship between operators from the dataflow graph, constructs the data structure of tasks and intermediate results, and sends the tasks to the scheduler for scheduling and dispatch.

At the same time, the job manager tracks the status and progress of operators, handles checkpoints, recovery failures. The Task manager executes operator instances in tasksslots and reports the status of operators to the job manager. It also maintains and manages flow buffers, which are responsible for data transfer and network communication between upstream and downstream operators.

3. Task scheduling strategy for distributed computing of aerospace test data

3.1 Main idea

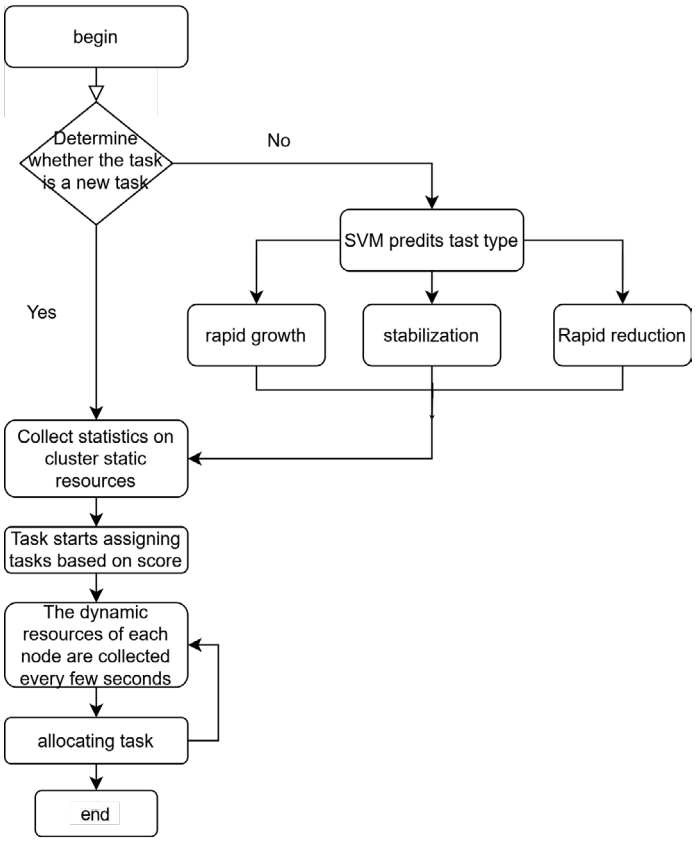

Figure 4. Route map

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

Data stream is the core of data engine, which runs through the whole business process. In the big data computing engine, the task flow is often intricate. In order to maximize resource utilization and improve the efficiency of task scheduling, the key is the design of scheduling algorithm. In the Flink system, the default round poll scheduling algorithm fails to fully consider the performance differences between cluster nodes, which leads to the fact that resource nodes with poor performance need more time to complete tasks, thus affecting the execution efficiency of the whole computing system. Since the types of space tasks include new tasks and routine tasks, it is particularly important to design a general scheduling algorithm that can adapt to various types of tasks. The algorithm needs to be able to propose corresponding strategies for different types of tasks.

In this paper, we first propose a method to decide the scheduling policy by judging whether the task is novel or not. For the new type of tasks, the analytic hierarchy process is used to summarize the information of each node by combining static and dynamic indicators. Analytic Hierarchy process (AHP) can quantify multiple indicators, so as to provide scientific basis for task allocation. Finally, the task will be assigned to the node with the highest score to optimize resource allocation and improve the overall scheduling efficiency. For routine tasks, the prediction algorithm can be used to train and predict historical resources, so as to rationalize the allocation of resource nodes in the early stage(Figure 4).

3.2 Model implementation process

3.2.1 Predictive scheduling algorithms

Firstly, for routine tasks, the law of resource utilization can be analyzed, and a predictive scheduling scheme can be adopted. The resource demand and its influencing factors are not generally linear, so as to save resources as much as possible and improve the running speed of the overall task to make up for the shortcomings of the round-robin algorithm. The algorithm predicts the future resource usage based on the historical time resource usage, and divides the future resource demand into three categories of scenarios, including rapid increase, stable and rapid decrease, to manage the utilization of cluster nodes. The WSVM is used to fit the classification set. For the three-class classification model, the classification expression is

$$\min_{\omega,B,\varepsilon_i} \frac{1}{2}||\omega||^2 + C\sum_{i=1}^{m}\varepsilon_i, \qquad s.t.\, y_i(<\omega, \emptyset(\overrightarrow{x_i}) > \overrightarrow{x_i} + b) \geq 1 - \varepsilon_i, \qquad \varepsilon_i \geq 0$$
$$i = 1,2,\ldots,m \tag{1}$$

3.2.2 Analytic hierarchy process scheduling

For different tasks, the utilization of cpu and memory will be different, and the number of cpu cores, cpu speed and memory size of each machine will also be different. For the machine that has just been started, this paper mainly uses the static resource usage analytic hierarchy process to quantify. Static resources include cpu speed, number of cpu cores, memory size, disk capacity, and the priority used by that machine. As the task starts to execute, the remaining tasks should be gradually tiled to all nodes, so the dynamic resources are used to quantify at this time, which include cpu utilization, memory utilization, disk I/O, and network bandwidth. Finally, it constantly adjusted according to the dynamic resources to ensure the load balance of the whole cluster.

The weight coefficient of the performance index in the above performance evaluation function reflects the impact of the performance index on the cluster performance. The weight coefficient cannot be determined by experience, but needs to be analyzed and predicted by scientific and effective methods. The performance indexes selected in this paper are relatively independent, so the analytic hierarchyprocess (AHP) is used to determine the weight coefficients of performance indexes.

For multi-objective decision problems, AHP mainly constructs the judgment matrix according to the importance degree of the given index, and follows the top-down sequential hierarchical decision goals and decision criteria to establish a hierarchical model. According to the comparison of the importance of different indicators, the weight order of the feature vector of the judgment matrix in each level is calculated. Finally, the final weight of the decision target is sorted by using the weighted method, and the maximum final weight is the optimal scheme.

In the running process, the dynamic indicators need to be obtained periodically, the static indicators do not change, and the dynamic indicators are dynamically adjusted with the running of the task. The

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

definitions of each dynamic performance metric are described below.

(1)Cpu utilization

As more tasks are allocated, the cpu load becomes higher and higher, that is, the task will be blocked. Define the formula

$$R_{\text{cpu}}(i) = \frac{idle}{idel + other} \tag{2}$$

Rcpu(i) describes the cpu utilization of the ith node, idle represents the cpu usage time of the node for processing the current task, and other represents the cpu usage time for performing other tasks.

(2)Memory utilization

Memory utilization represents the proportion of memory used by a node to the total memory.

$$R_{\text{mem}}(i) = 1 - \frac{MemFree}{MemTotal} \tag{3}$$

Rmem(i) represents the memory utilization of the ith node, MemFree represents the remaining memory, and MemTotal represents the total memory of the node.

(1)Disk utilization

$$R_{\text{dis}}(i) = \frac{UsedNum}{DisTotal} \tag{4}$$

Rdis (i) is the disk utilization of the ith server node, Used Num is the disk resources used to execute the current task per unit time, and disTotal is the total disk storage available.

(2)Network broadband utilization

$$R_{\text{net}}(i) = \frac{\Delta Byte\,\text{Re}ceived + \Delta ByteTransmited}{\Delta T \cdot Speed} \tag{5}$$

Rnet(i) represents the network bandwidth utilization of the ith server node,

ByteReceived and ByteTransmited represent the total number of bytes received and sent by the heterogeneous cluster node in a certain time interval, and Speed represents the network bandwidth of the heterogeneous cluster located at the server.

$$WS_{\text{i}} = \delta S_{\text{i}} + (1-\delta)(\alpha_{cpu}\frac{U_{cpu}(i)}{T_{cpu}} + \alpha_{mem}\frac{U_{mem}(i)}{T_{mem}} + \alpha_{dis}\frac{U_{dis}(i)}{T_{dis}} + \alpha_{net}\frac{U_{net}(i)}{T_{net}}) \tag{6}$$

δ represents the weight corresponding to the index, Si is the sum of the static performance index data calculated , and αcpu, αmem, αdis, αnet are the weight coefficients of the performance index calculated.

4. Test results

In this paper, the data engine of space command and display system is designed and constructed, and a number of tests are carried out on the data engine, and Kylin V10 is used as the basic test environment. With the architecture proposed in this paper, the image library can be used as a middleware for version synchronization and history acquisition, which greatly reduces the configuration work time.

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

4.1 Prediction results

Using a space test mission, the node growth status is shown in the following Tables 1 and 2.

Table 1. The node growth status (1)

Method	Precision			Recall		
	stabilization	Rapid growth	Rapid reduction	stability	Rapid growth	Rapid reduction
KNN	0.64	0.41	0.53	0.53	0.55	0.39
GaussianNB	0.34	0.39	0.23	0.92	0.04	0.04
CNN	0.52	0.39	0.62	0.48	0.46	0.52
LSTM	0.67	0.60	0.57	0.43	0.53	0.55
WSVM	0.88	0.45	0.60	0.34	0.71	0.51

Table 2. The node growth status (2)

Method	F1-score			Accuracy
	stabilization	Rapid growth	Rapid reduction	
KNN	0.59	0.67	0.44	0.49
GaussianNB	0.49	0.11	0.16	0.40
CNN	0.51	0.43	0.54	0.52
LSTM	0.53	0.50	0.56	0.52
WSVM	0.50	0.55	0.58	0.55

Finally, by comparing the Precision, Recall, F1-score and Accuracy of each algorithm, the WSVM algorithm is generally better than other algorithms, and performs better in stability and fast reduction.

4.2 Round robin scheduling algorithm

Using a certain space mission, the results of the scheduling strategy are as follows, the results table name, compared to the default scheduling algorithm, the time adopted in this paper is shorter(Table 3).

Table 3. Results of the scheduling strategy

Data/GB	Default round robin scheduling algorithm/s	Pre-scheduling algorithm/s
2	150	130
5	230	190
8	440	400
11	620	520

In the face of large-scale aerospace experimental data, it is obvious that the scheduling algorithm designed in this paper is significantly higher than the default. In the conducted tests, the overall efficiency was optimized by 15%.

5. Conclusion

In this paper, a set of task scheduling algorithms are designed for the requirements of aerospace command and display system, aiming at optimizing the processing of aerospace experimental data. The algorithm not only ensures the availability, high performance and high scalability of the system architecture, but also facilitates monitoring and management, which has achieved remarkable results in the test task.

This paper will focus on two research directions. Firstly, the pre-scheduling algorithm will deeply investigate machine learning and deep learning techniques to improve the accuracy of the algorithm. This will not only be applied to conventional missions, but also when the amount of tasks is huge, the designed prediction model can summarize the overall law, and then extend to all space missions. Secondly, the subjective weight assignment method used in the current scheduling algorithm will be optimized. As the amount of data increases, objective assignment methods, such as analytic hierarchy process (AHP) and entropy weight method, can be introduced to dig deeper into the inherent laws of space missions and construct a set of more general models.

ISSN 2522-3488 Vol. 9, Issue 1: 97-105, DOI: 10.25236/IJNDES.2025.090113

References

[1] Liu Yugui. Design and Realization of the Integratde Command and Display Software System [D]. Xidian University, 2012.
[2] Zhou Yu, Yin Zhifeng, Li Linfeng, Zhou Gan. General command display data engine architecture and application design [J]. Cyber security and data governance, 2021, 40(12):65-70.
[3] Dan Peng; He Xiaosong; Wang Dan. Design and Algorithm of a Real-Time Data Processing Software of Exterior Measurement for Spacecraft [J]. Radar Science and Technology, 2023, 21(05):568-574.
[4] Yunjun L , Jufang L I , Yingwu C ,et al. Research on the Uplink Task Scheduling Strategy of Satellite Navigation System[C]//SpaceOps Symposium.2012.
[5] Mobile Computing W C A. Retracted: Research on Optimization Strategy of Task Scheduling Software Based on Genetic Algorithm in Cloud Computing Environment [J]. Wireless Communications & Mobile Computing, 2023.DOI:10.1155/2023/9834952.