A Distributed Transmission Method For Satellite Data Based On Streaming

Guo Peng, Manyun Lin, Xiangang Zhao, Cunqun Fan*, Lizi Xie

National Satellite Meteorological Centre, Beijing, China
{guopeng, linmy, zhaoxg, fancq, xielz}@cma.gov.cn

Keywords: distributed transmission; satellite data; streaming.

Abstract

The transmission and collection of satellite data will directly affect the success or failure of the subsequent satellite products. So the data transmission, the stability and efficiency of the collection, has been highly valued. This paper presents a distributed data transmission architecture, through the data transmission, and the establishment of monitoring module to monitor the data transmission process. On this basis, this paper also expands the system's scalability, fault tolerance and reliability.

1 Introduction

Data transmission, aggregation is a very important part in satellite data processing system. The transmission and collection of satellite data will directly affect the success or failure of the subsequent satellite products. So the data transmission, the stability and efficiency of the collection, has been highly valued.

There have been some research results on distributed data transmission. In paper [1], authors address only the latter concern. At first they assume that each sensor is tasked to communicate exactly one of its observations to a Fusion Center (FC) for a global estimate, and they work in one dimension. Via order statistics they show that, surprisingly, the nearest neighbor (NN) is not always the most appropriate measurement to share. They also expand our bandwidth to allow for transmission of multiple measurements, for example the nearest and third-nearest: it turns out that a single-measurement transmission is more bandwidth efficient than multiple. Paper [2] proposed a multimedia data transmission method, Multitrack, that achieves high-speed and efficient data transmission between multiple mobile hosts and the internet. Users of the hosts in a cluster can see a high quality streaming video by sharing the received multimedia data cooperatively with the other hosts. In Multitrack, it is important to disperse traffic according to the quality of multiple links in order to offer a stable multimedia data transmission. They also proposed an effective traffic dispersion method for this environment, and evaluate this method and other basic methods by simulations. To overcome the problems from past research, paper [3] proposed a data recovery model based on the parallel transmission strategy, Poisson distribution theory, and the mechanism of a data exchange center. The model introduced the non-linear speed and function, cancelling the constraints and assumption for parallel transmission speed of a physical network. The paper theoretically analyzed the influence of all parameters on the performance of the data recovery model. In paper [4], the authors therefore use local measurements of the instantaneous channel conditions to select the best relay pair from a set of N available relays, which both come from the same cluster or different clusters, and then use these best relays for cooperation between the source and the destination. The authors also show that the best relay pair selection scheme has robustness against feedback error and outperforms a scheme based on selecting only the best single relay. In paper [5], a waveform relaxation algorithm is presented for efficient transient analysis of large transmission-line networks. The proposed methodology represents lossy transmission lines as a cascade of lumped circuit elements alternating with lossless line segments, where the lossless line segments are modeled using the method of characteristics. Partitioning the transmission lines at the natural interfaces provided by the method of characteristics allows the resulting subcircuits to be weakly coupled by construction. The subcircuits are solved independently using a proposed hybrid iterative technique that combines the advantages of both traditional Gauss–Seidel and Gauss–Jacobi algorithms. The weight combination of the neighboring nodes play a crucial role in adaptation and tracking ability of the network. The paper [6] investigates for general adaptive diffusion algorithm, in presence of various sources of imperfect information exchange, then moves on to investigate for BLMS diffusion method, which plays a crucial role in reducing the burden of communication by a factor of block length. Paper [7] proposed a versatile and high-fidelity multi vital signs compression method using adaptive Fourier decomposition to decompose signal beat into weighted orthogonal components. An intelligent signal type detection and automatic parameter adjustment scheme is designed and implemented, thus compress different bio-signals with compression ratio (CR) > 5.6 and normalized percentage relative difference (PRDN) < 7.3%. In paper [8], authors introduce a computation model called distributed two-phase model, in which the process of a task can be divided into two independent phases: data transmission and data computation. Suppose an upper bound of relative computation time is given, they show how to schedule data transmission with minimum resources, such as data transmission time and occupied bandwidth, to meet the demand. And they proposed a novel algorithm to minimize data transmission time and network bandwidth usage in the data transmission phase, with the conditions that an upper bound of relative computation time of data computation phase is given. Moreover, the number of nodes that participate in data computation phase is also reduced, in this way, the computational resources are saved. The remainder of this paper is organized as follows. This is followed by our proposed distributed transmission system in section II. Streaming data processing is explained in section III. The paper concludes with the ending remarks in section IV.

2 Distributed transmission system

The main steps of distributed data transmission are data acquisition and transmission, and real-time monitoring of its main indicators. The main business process is shown in Figure 1.

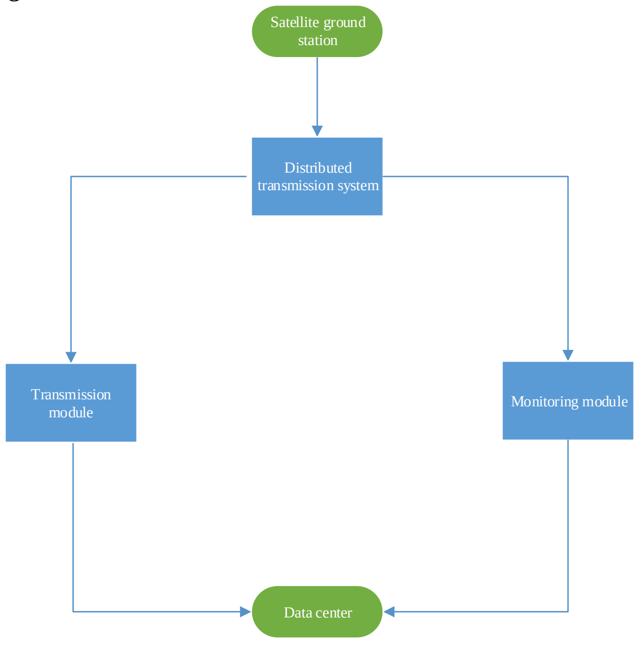

Figure 1: Distributed transport business process analysis.

2.1 System operating mode

Uninterrupted scan monitoring directory, the directory to be transferred to the file to be split into bytes stream for transmission at the receiving end of the file stream will be aggregated into a file block, after the file block will be aggregated into a complete file, in this process will be transmitted bytes Stream, file block, the number of files and other information statistics passed to the monitoring page by the report display. The process is shown in Figure 2.

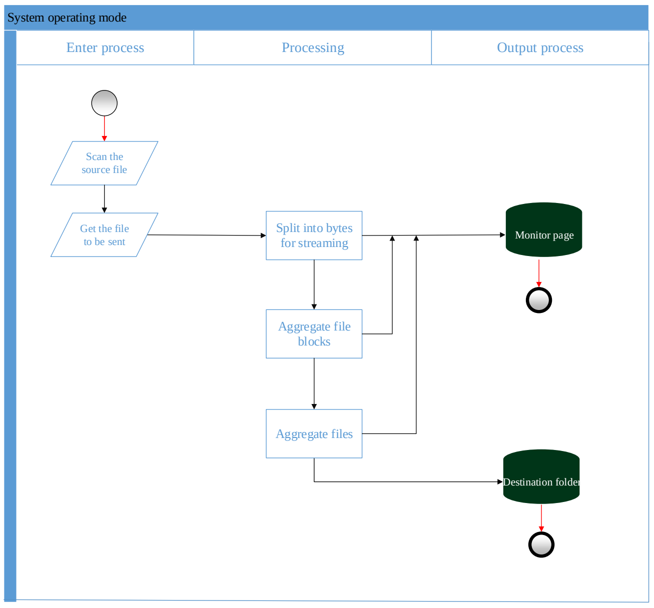

Figure 2: System operation mode flow chart.

2.2 Normal data transmission mode

The system scans the folder to fetch the file to be transferred, splits it into a node that flows to the cluster, aggregates the byte stream into the file block at the target node, and finally aggregates the file block into a complete file. After the file has passed the integrity check End of transmission. The process is shown in Figure 3.

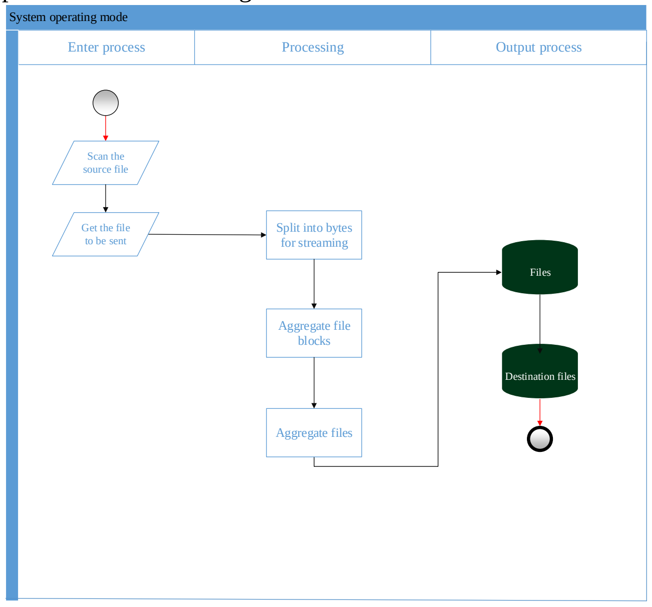

Figure 3: Normal data transfer mode flow chart.

2.3 Data transmission mode under device failure

When the data is being transferred to the task node encountered an unexpected failure caused by downtime, the master node will try to restore the transmission process, if not, the master process will switch the transmission node, the fault node of the transmission task to switch to other nodes to re-transfer the file. The process is shown in Figure 4.

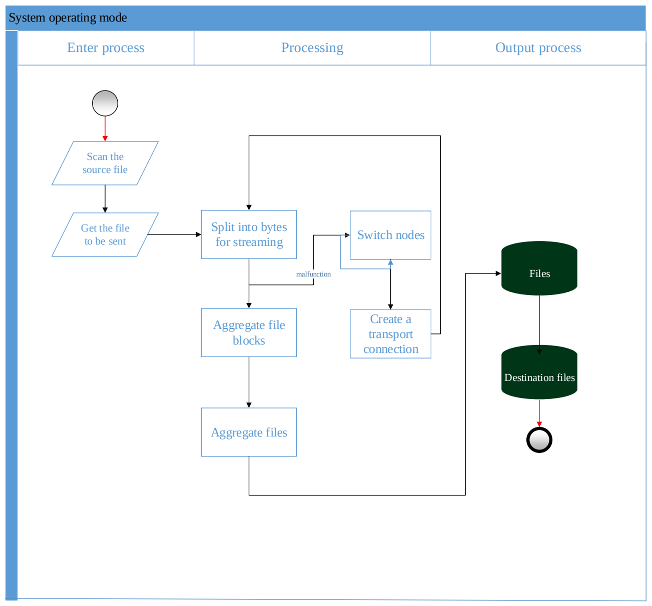

Figure 4: Flow chart of data transmission mode under device failure.

3 Streaming data processing

The streaming framework has a simple and easy-to-use API. When programming with a streaming framework, the user can use the corresponding API to manipulate and transform the tuple's stream. A tuple is a list of named values. A tuple can contain any type of object, and if you want to use a type that the streaming framework does not know, you can register the type by serializing it and then applying it in the program.
The streaming framework has only three abstract types: the stream sender, the stream sender, and the topology. The stream sender is the source of the calculated stream. Usually the streaming sender in the system reads from the message queues such as Kestrel, RabbitMQ and Kafka, but the stream sender can also generate its own stream or read from somewhere in the Twitter stream API. The streaming sender implements most of the existing queue systems.
The stream sender processes any number of input streams and generates any number of new output streams. Most of the logic to enter the flow of the sender, such as function, filtering, flow connection and database interaction.
Topology is a network consisting of a number of streamers and streams. Each edge on the network represents a stream of data sent by a streaming sender, which includes data streams from the streaming sender or from the streaming sender. A Topology is actually an arbitrary multilevel flow calculation process. When Topology is deployed on the server, it will run until the user disables the corresponding process.
The streaming framework has a "local mode" where the user can simulate a streaming framework cluster in the process and then perform development work on a similar physical cluster. This model is useful for development and testing. When the user is ready to submit Topology on a real cluster, you can use the Streaming Framework command line to easily run a Topology from the client to run on the cluster.

3.1 System scalability

The Topology of the streaming framework is computed in parallel and runs on a cluster host. You can adjust their parallel extensions individually for different parts of Topology. Streaming Frames The command line client's rebalance command can adjust the topology running in parallel.
The parallelism of the streaming framework means that it can handle high-throughput messages at low latency. According to the profile of the streaming website, a node in the streaming framework (Intel E5645@2.4Ghz CPU, 24 GB of memory) can handle 1 million 100-byte messages in 1 second.

3.2 System fault tolerance

The streaming frame is fault tolerant. When Worker is dead, the streaming framework automatically restarts them. If a node dies, the Worker will restart on another node.
Streaming the framework of the daemon Nimbus and Supervisor are designed to be stateless and fast fail, so if they die, they will reboot, just like nothing has happened. This means that you can use the kill -9 command to force the kill frame to protect the daemon without affecting the health of the cluster or Topology.

3.3 System reliability

Like a butterfly effect, a tuple from the streaming sender can trigger thousands of tuples based on what it creates.
This topology reads sentences from a Kestrel queue, divides sentences into words, and then counts each word before emit.
A tuple from the stream sender triggers a lot of tuples based on it: one is the tuple for each word in the sentence, and one is the tuple for each word update count. As shown in Figure 5, for the sentence of the cow jumped over the moon tuple tree.

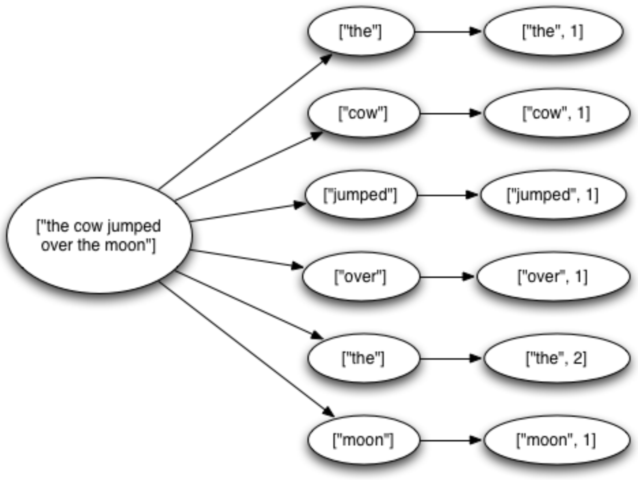

Figure 5: WordCount a tuple tree for a sentence.
When the tree is created and every message in the tree has been processed, the streaming framework considers the tuple from the stream sender to be "" completely processed. " When a tuple message tree can not be completely processed within the specified timeout range, the tuple is considered to be failing. The default value for timeout is 30 seconds. For a

specific topology, you can use Config.TOPOLOGY_ MESSAGE_TIMEOUT_SECS to configure the changes.

4 Conclusion

In this paper, satellite data transmission, collection of research. The transmission and collection of satellite data will directly affect the success or failure of the subsequent satellite products. So the data transmission, the stability and efficiency of the collection, has been highly valued. This paper presents a distributed data transmission architecture, through the data transmission, and the establishment of monitoring module to monitor the data transmission process. On this basis, this paper also expands the system's scalability, fault tolerance and reliability.

Acknowledgements

The work presented in this study is supported by National High-tech R&D Program (2011AA12A104).

References

[1] Braca P, Guerriero M, Marano S, et al. “Selective measurement transmission in distributed estimation with data association”. *IEEE Transactions on Signal Processing,* vol. 58, pp. 4311-4321, (2010).
[2] Saito Y, Ishihara S, Mineno H, et al. “Evaluation of traffic dispersion methods for synchronous distributed multimedia data transmission on multiple links for group of mobile hosts”. *International Journal of Applied Systemic Studies*, vol. 3, pp. 23-34, (2010).
[3] Jiang C, Zhang G, Mingcheng Q U. “A highly-reliable data recovery model based on parallel transmission for a P2P distributed system”. *Journal of Harbin Engineering University*, vol. 33, pp. 347-354, (2012).
[4] Chen G, Chambers J. “Outage probability in distributed transmission based on best relay pair selection”, *Iet Communications*, vol. 6, pp. 1829-1836, (2012).
[5] Roy S, Dounavis A, Beygi A. “Longitudinal-Partitioning-Based Waveform Relaxation Algorithm for Efficient Analysis of Distributed Transmission-Line Networks”. *IEEE Transactions on Microwave Theory & Techniques*, vol. 60, pp. 451-463, (2012).
[6] Kumar S, Sahoo A K, Acharya D P. “Block diffusion adaptation over distributed adaptive networks under imperfect data transmission”. *IEEE India Conference*, pp. 1-5, (2014).
[7] Ma J L, Chen M B, Dong M C. “High-fidelity data transmission of multi vital signs for distributed e-health applications”. *IEEE International Symposium on Bioelectronics and Bioinformatics*, pp.1-4, (2014).
[8] Zhang X, Jiang J, Zhang X, et al. “A data transmission algorithm for distributed computing system based on maximum flow”. *Cluster Computing,* vol. 18, pp. 1-13, (2015).