MPI-Based High-Performance UVFITS Data Synthesis: Research and Application (Postprint)

Authors: Tairan Chen, Wang Wei, Wang Feng, Deng Hui, Liu Yingbo, Mei Ying

Date: 2017-10-20T00:00:00+00:00

Abstract

The Mingantu Ultrawide Spectral Radioheliograph (MUSER) in China generates approximately 100 kB of data per frame every 3 ms since entering operational status, with daily raw observation data totaling approximately 3.5 TB. As the radioheliograph' s raw data employs a custom format, conversion to commonly used astronomical file formats is necessary to meet data storage requirements for subsequent analysis and sharing. Previous work has implemented conversion from the raw data format to UVFITS files; building upon this, we investigated performance optimization of the UVFITS synthesis system in an MPI-based cluster parallel environment. Experimental results demonstrate that in the improved parallel environment, the UVFITS synthesis system achieves performance 2.5 times the requirement, enabling effective processing of the radioheliograph' s massive observation data for both current and future needs. Additionally, the improved system exhibits good horizontal scalability, providing a valuable reference for data processing in related projects.

Full Text

Abstract

The Mingantu Ultrawide Spectral Radioheliograph (MUSER) generates approximately 100 kilobytes of raw observational data every 3 milliseconds, totaling more than 3.5 terabytes per day. Since the raw data employs a custom format, conversion to standard file formats commonly used in radio astronomy is necessary for subsequent data analysis and sharing. Previous work successfully implemented conversion from the raw data format to UVFITS files. This paper presents performance optimization of the UVFITS assembly system based on a cluster parallel environment. Experimental verification demonstrates that the improved parallel system can assemble a UVFITS file in approximately 1.2 milliseconds—2.5 times faster than the data acquisition rate—effectively handling

MUSER' s massive observational data volumes for the foreseeable future. Moreover, the optimized system exhibits excellent horizontal scalability and can serve as a reference for data processing in related projects.

Keywords: UVFITS; Massive data; MPI; Parallel computing; High-performance computing

1. Introduction

The Mingantu Ultrawide Spectral Radioheliograph (MUSER) is a solar radio imaging instrument with high temporal, spatial, and frequency resolution [1]. As the system enters integrated testing and trial observation phases, there is an urgent need to store trial observation results and apply current radio astronomy software packages, such as the Common Astronomy Software Applications (CASA) [2], to process and analyze trial data for equipment error correction and calibration. Converting observational results into formats compatible with such software packages has become a critical challenge.

In radio astronomy, the random groups data storage format is widely adopted, with UVFITS being the primary file type [3]. MUSER-I generates data at a rate of 25 milliseconds per frame, producing 2 GB files. To meet real-time imaging requirements, UVFITS files must be generated at a rate of one frame per minute. Similar international projects include the Nobeyama Radioheliograph in Japan and the Nancay Decimetric Radio Telescope (NRT) in France. These facilities generate one large data file after accumulating 60 frames ($8 \times 40 \times 60 =$ 19,200 frames), utilizing the 1-minute interval for data transfer. This approach allows integration of data from the same time period to improve accuracy while conserving storage space [4].

However, MUSER-I' s data scale significantly exceeds that of existing international instruments: its data volume is approximately 200 times that of the Nobeyama Radioheliograph and 60 times that of the Nancay Radio Telescope, with different imaging processing requirements. Consequently, foreign radioheliograph processing software cannot be directly adopted. Designing and implementing a high-performance UVFITS assembly system tailored for MUSER is therefore a pressing priority.

2. UVFITS Assembly Process

2.1 Data Format and Processing Requirements

MUSER' s raw data files contain quantization level values, thresholds, delay adjustment switches, fringe rotation switches, sub-band operation modes, timestamps, signal center frequency selection switches, timing information, antenna parameters, and numerous delay corrections. To meet storage requirements, this information must be preserved when converting to UVFITS format. For improved accuracy, the system performs integration operations on data according to specified start and end times, saving processed data as UVFITS files for convenient retrieval.

The UVFITS file structure consists of several key components: PRIMARY, ANTENNA, FREQUENCY, and SOURCE. The assembly process involves reading raw data into memory, extracting timestamps, calculating current antenna positions and solar positions, reading antenna polarization and reception frequency information into AIPS AN and AIPS SU tables, processing visibility data from the raw files into AIPS FR and PRIMARY extensions, and adding necessary UVFITS keywords.

2.2 Data Flow

Radio noise signals received by antennas are amplified by low-noise amplifiers, converted to optical signals for fiber transmission to the equipment room [6], and processed through analog and digital receivers to generate data files in a predefined format [7]. The assembly system reads these stored raw data files into memory. For each frame, the system processes timestamp, position, and visibility information. In multi-frame synthesis, integration operations combine data from multiple frames to enhance precision. The complete UVFITS assembly flow is illustrated in [Figure 1: see original paper].

Visibility image data originates from the visibility measurements in raw data. According to aperture synthesis principles, signals from two antennas must reach the correlator simultaneously and in-phase [8]. The system compensates for system delays and phases using delay compensation and fringe-stopping techniques [9]. During implementation, the system processes visibility data based on input timestamps and parameters, adding essential UVFITS keywords to each extension (AIPS AN, AIPS SU, AIPS FR, PRIMARY).

3. Parallelization Analysis

3.1 Performance Bottlenecks

Due to limited single-machine performance, multi-machine parallelization was adopted to maximize program performance and provide room for further improvement. The entire program execution can be summarized as: input parameters, process data (analysis and integration), and output results. Parallelizable components include data reading, processing, and output.

Testing revealed that program time overhead concentrates in data reading and data integration processes. Data reading accounts for approximately 60% of total runtime, while data integration accounts for about 30%. Initial parallelization attempts for data integration proved ineffective due to communication overhead approaching processing time, resulting in negligible performance gains from additional nodes. Bus speed bottlenecks also caused single-core and multi-core performance to be nearly identical.

3.2 Parallel Strategy

Given these constraints, parallelization was applied to data reading—the frame reading portion of raw data input. The system employs a master-slave model in cluster environments for parallel visibility data processing. The master node divides raw data into frame-based tasks, distributing them sequentially to slave nodes. Each slave node processes received tasks and writes results directly to the master node' s storage medium without inter-node communication.

Data Partitioning: Each frame serves as a processing unit, distributed evenly across nodes based on node count. Each slave node maintains complete frame header information and further partitions data internally according to configured process numbers. Results are written directly to master node storage, eliminating merge overhead.

Result Merging: Since each node processes complete frames with independent timestamps, result files are named according to time ranges. Task independence ensures merge overhead depends only on master node I/O performance, not computational complexity.

The parallel execution flow is illustrated in [Figure 2: see original paper].

4. Performance Analysis

4.1 Evaluation Metrics

Performance analysis uses speedup ratio as the primary metric, calculated from the time between raw data loading into memory and program completion. The speedup ratio (S_p) compares single-node serial execution time (T_serial) to multi-node parallel execution time (T_parallel):

$$S_p = \frac{T_{\text{serial}}}{T_{\text{parallel}}}$$

Parallel execution time includes node communication overhead and is determined by the slowest node:

$$T_{\text{parallel}} = T_{\text{bcast}} + T_{\text{calculate}} + T_{\text{reduce}}$$

where T_bcast is broadcast time, T_calculate is computation time, and T_reduce is reduction/merge time.

4.2 Experimental Setup

Test hardware environment: - Servers: Sugon A620r-G with AMD Opteron 6128 CPUs (2.0 GHz) - Memory: 8 GB per node - Hard drives: Seagate ST3250310AS (250 GB) - Network: 1 Gbps - Operating system: Linux - MPI implementation: MPICH-2

Tests used randomly selected observation data from file MUSER_20140122-131903_342668765, evaluating performance under non-integrated and 735-frame integration scenarios with data volumes of 1,055 and 1,375 frames respectively.

4.3 Results

Serial Execution: Single-machine serial execution time increases linearly with frame count but achieves only half the target efficiency, reaching approximately 25 ms per frame—far below MUSER's storage requirement of 12 ms per UVFITS file.

Single-Machine Parallelism: Using 2, 4, and 8 processes on a single server reduces execution time by more than half with each doubling of process count, achieving stable speedup ratios. Detailed timing data is provided in .

Multi-Machine Parallelism: In a four-node cluster, execution times increase slightly compared to single-machine parallelism due to inter-machine communication overhead exceeding intra-machine bus communication. However, time costs still decrease by nearly half with each node doubling, meeting expectations. Performance data for 2, 4, and 8 processes across four nodes is shown in and .

The optimized parallel UVFITS assembly system processes one frame in 1.2 ms under 8-process configuration—2.5 times faster than MUSER' s 3 ms data acquisition rate—satisfying current requirements with excellent scalability for future higher-performance needs.

5. Future Work

Future research will address: 1. **File Synchronization:** Implementing proactive multi-machine file synchronization to achieve pipeline effects and reduce communication time for master node storage access. 2. **Process Optimization:** Further optimizing data processing workflows to increase parallelism. 3. **API Development:** Providing Application Programming Interfaces (APIs) for data processing and archival query systems to meet data access requirements.

References

[1] Yan Y, Zhang J, Wang W, et al. The Chinese Spectral Radioheliograph. Earth and Planetary Physics, 97-100.

[2] Wells D C, Greisen E W, Harten R H. FITS—a flexible image transport system. Astronomical Data Analysis Software and Systems XVI.

[3] Gao Jiaojiao, Wang Feng, Dai Wei, et al. Analysis of random-group data format and file design for solar radio heliograph. Astronomical Research & Technology—Publications of National Astronomical Observatories of China, 365-371.

[4] Jaeger S. The Common Astronomy Software Applications. Astronomical Data Analysis Software and Systems XVII.

[5] Nakajima H, Nishio M, Enome S. The Nobeyama radioheliograph. Proceedings of the IEEE, 705-713.

[6] Wang Wei, Chen Zhijun, Ji Guoshu, et al. Optical fiber transmission analysis for CSRH. Astronomical Research & Technology—Publications of National Astronomical Observatories of China, 143-147.

[7] Ji Guoshu, Dou Yujiang, Wang Wei, et al. Research & Technology, 135-142.

[8] Zhang Jian, Yan Yihua, Liu Fei, et al. Publications of National Astronomical Observatories of China, 148-153.

[9] Liu Donghao, Yan Yihua, Zhao An, et al. RF receiver design for CSRH. Acta Electronica Sinica, 570-574.

Note: Figure translations are in progress. See original paper for figures.