4-2 Applications of Advanced Information and Communications Technology

4-2-1 Distributed Parallel Processing Based on Grid Datafarm Architecture

YAMAMOTO Kazunori, KIMURA Eizen, MURATA Ken T., TATEBE Osamu, MATSUOKA Daisuke, and MIYACHI Hideo

In the Solar-Terrestrial Physics field, satellite observation data and computer simulation data have been tremendous increased. Since most of data files and computer resources are distributed over the Internet, analysis environments for data intensive processing are required. In this study, we propose a parallel distributed processing system with meta-data system and Grid Datafarm. A testing system is constructed with 8 filesystem nodes. As a result of small-data processing of observation data on the system, parallel processing is found effective using metadata file at local disk and hierarchical Gfarm file. As for parallel visualizations of simulation data, it was achieved high parallelization efficiency of 97.6% when using FIFO-type scheduling.

Keywords
Meta-database, Grid datafarm, Distributed parallel processing, Satellite observation date, Computer simulation data

1 Introduction

The field of solar-terrestrial physics is a multidisciplinary branch of research activity intended to shed light on the physical process in which space weather and other solar activities affect the terrestrial magnetosphere, ionosphere, and other aspects of the earth's atmosphere. Advanced satellite equipment continues to deliver more granular observation data of a larger scale than ever before, while the volume of that data continues growing with impetus from the implementation of such international projects as the electronic Geophysical Year (eGY)[1] and International Solar Terrestrial Physics (ISTP) program[2]. In addition, improvements in the performance of supercomputers (including the Earth Simulator) have accelerated ongoing leaps in the granularity and scale of computing models used for computer simulation. There are also growing expectations in terms of data analysis and computer simulation focus on statistical analyses and the manipulation of multi-point, long-term observation data[3] delivered from multiple satellites, and on large-scale data processing, such as for 3D visualization processing in long-time steps[4].

Large-scale data processing in the field of solar-terrestrial physics more often than not involves data-intensive processing, or a series of processing steps where the same set of processing steps are executed on a large number of data files, thereby calling for a method of managing and sharing relevant data files, as well as distributed parallel processing to balance the file I/O load.

This paper proposes the constructing of a distributed parallel data processing system for executing data-intensive processing on satellite observation data and computer simulation data using Gfarm (Grid Datafarm)[5]—a reference implementation of the grid datafarm architecture—to evaluate its effectivity and practical usefulness in the field of solar-terrestrial physics. Gfarm is middleware designed for data grids that control file transfer between networks and distribute file I/O by using a number of nodes that serve as storage in computing[6]. Gfarm has thus far been successfully used in implementing astronomical data analyses[7], the GEO Grid project[8] and more, with the effectivity of its file I/O scalability and features to balance CPU performance load having been verified. Yet, constructing a data-intensive environment in the field of solar-terrestrial physics in which data files are managed, distributed, is such a laborious task that the effectivity of distributed parallel processing by data grids has yet to be verified. Moreover, because the data size and data workload vary among different batches of analysis data and different analysis periods, the scheme for load balancing by simply splitting data files may not prove useful at all times. Hence, a scheduling method not targeted by Gfarm is needed.

2 Status quo and problems of the integrated data processing environment in the field of solar-terrestrial physics, and the proposed system

2.1 Status quo and problems of the data processing environment

As described in Section **1**, the field of solar-terrestrial physics is a multidisciplinary branch of research activity, for which observation data have been managed and publicized, and distributed among universities and research institutes both at home and abroad. Because data analyzers download all data files needed for their own data processing computers, the tasks of downloading, analyzing and visualizing these files become more complex as more types of data and more files are involved.

When data is to be analyzed and visualized on a data analyzer's own terminal, CPU performance and disk I/O would constrain a scale of processing involving huge sizes or files of data. For this reason, many data centers offer online services for large-scale online data processing. For example, SSCWeb[9] operated by the National Aeronautics and Space Administration (NASA) and DARTS[10] operated by the Japan Aerospace Exploration Agency (JAXA) offer services for searching through their satellite observation data and plotting the data in a simple sequence. Moreover, 3D visualization films of simulation data have been released based on real-time terrestrial magnetosphere simulation[11] conducted by the National Institute of Information and Communications Technology (NICT). In these environments, data analyzers can retrieve processing results without having to download data files. Data processing, however, does not go beyond simple searches under limited conditions or primitive tasks such as data plotting, since the data sites prescribe the details thereof. Many data centers offer no service for creating an integrated analysis environment that encompasses data from other data centers as well.

2.2 Summary of the proposed distributed parallel processing system

As a solution to address the problems outlined in Section **2.1**, this paper proposes a data processing system that builds on the meta-data utilization system (STARS: Solar-Terrestrial data Analysis and Reference System)[12][13] and Gfarm, a reference implementation of the grid datafarm architecture. STARS provides a data utilization environment that offers access transparency and location transparency to satellite observation data files and computer simulation data files kept under distributed management. The proposed system uses STARS to search for and retrieve data, and executes large-scale distributed parallel processing on the data thus collected on Gfarm. Gfarm v1.4.1 was used to implement the system.

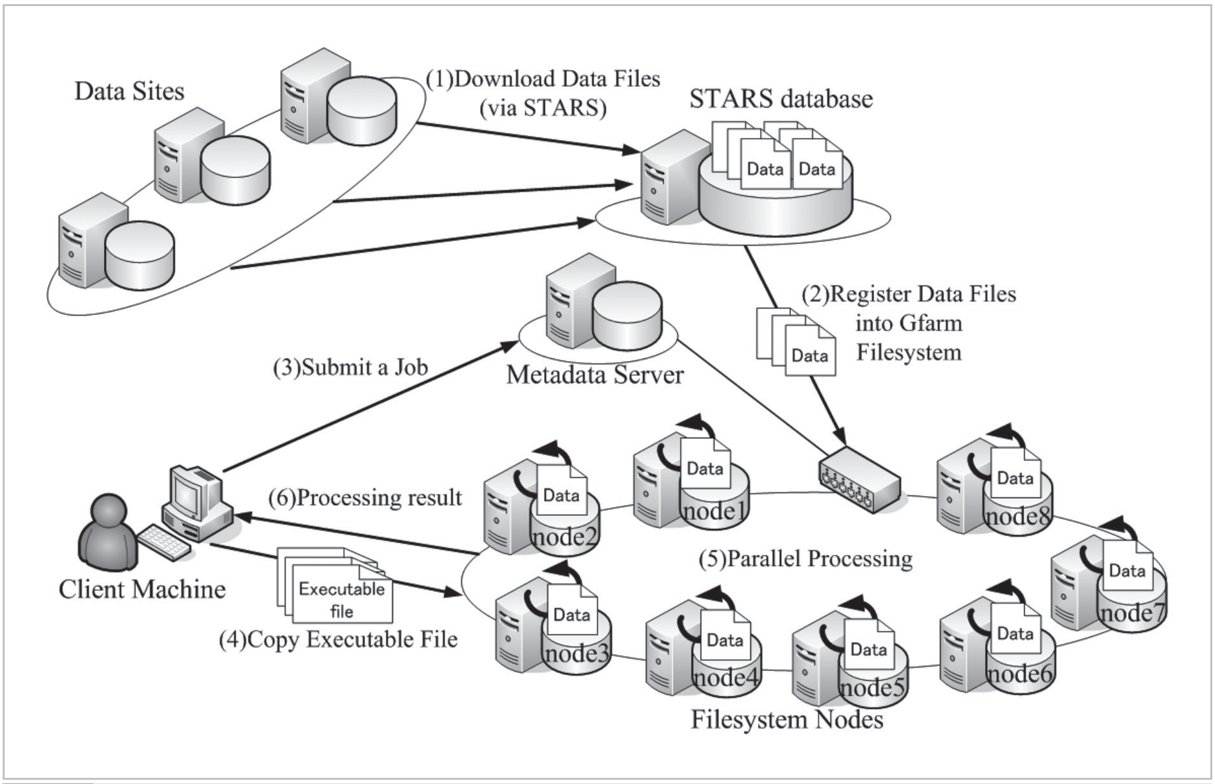

Fig. 1 Parallel distributed processing system running on STARS and Gfarm

Table 1 Specifications of the computer used for system implementation

	Filesystem Node ×8	Metadata Server ×1 Client Machine ×1
CPU	Athlon 64 ×2 Dual Core 2GHz	Dual Core AMD Opteron 1.81GHz
Memory	2GB	1GB
Disk	1.2TB(using RAID0)	232GB
OS	Fedora Core 5	Fedora Core 5

Figure 1 shows the configuration of the proposed system built of eight file system nodes; Table 1 summarizes its computer specifications. To begin with, the user retrieves data files from a data site by way of STARS (Fig. 1-(1)) and then registers those files with the Gfarm file system (Fig. 1-(2)). The nature of processing may require replicating the files on each file system upon their registration, in order to ensure the efficient load balancing described in Section **4.4**. When a distributed parallel processing job is later submitted (Fig. 1-(3)), an executable program is copied from the client machine to each file system node (Fig. 1-(4)) to launch distributed parallel processing (Fig. 1-(5)). Lastly, the integrity of the processing results displayed is verified before those results are returned to the user (Fig. 1-(6)).

3 Distributed parallel processing of long-term satellite observation data

3.1 Comparison of distributed parallel processing with sequential processing

This section compares the tested performance of distributed parallel processing on Gfarm with sequential processing by using the system shown in Fig. 1. One file system node is used to execute sequential processing, with Gfarm assumed to generate no overhead. Parallel distributed processing is executed with two to eight nodes running in parallel.

Data-intensive processing was used in an experiment where each file of data would be processed in one process. Data processing was executed by loading total time-series data sequentially from the two kinds of satellite observation data varying in file size as listed in Table 2, with the results of all data records being directed to standard output. Assuming M data files and N file system nodes running in parallel, each node processes M data files (Fig. 2 (a) by itself in sequential processing, as compared to M/N data files in distributed parallel processing (Fig. 2 (b)). As a preparatory step to the experiment, all data files were registered with the Gfarm file system beforehand.

Figure 3 shows the sequence of distributed parallel processing among the client machine, meta-data server, and file system nodes. Processing is primarily broken down into the scheduling shown in Fig. 3-(1) to (8) (file system node selection and job submission), the file loading in Fig. 3-(9) to (11) (meta-database access for referencing the entity files of Gfarm files and for data processing), and standard output in Fig. 3-(12) (client machine access).

Table 2 *Kinds of satellite observation data*

Mission / Data	File Size[KB/file]
(a)GEOTAIL/LEP(Key parameter)	450 - 995
(b)GEOTAIL/Orbit(Level 2)	44 - 58

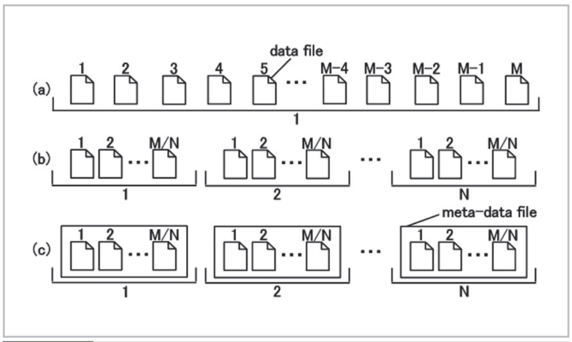

Fig.2 *Satellite observation data processing scheme*

(a) Sequential processing
(b) Parallel distributed processing
(c) Parallel distributed processing where Gfarm files are structured hierarchically

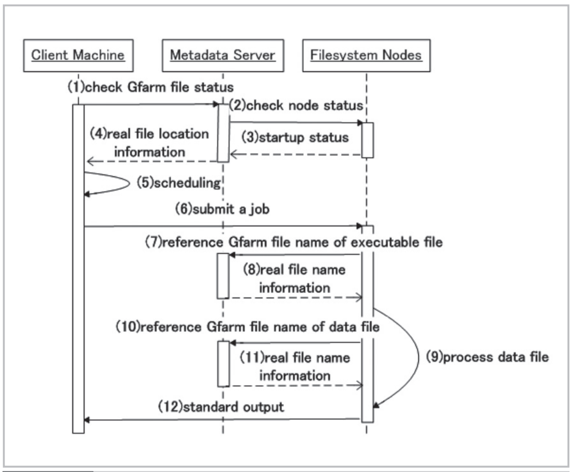

Fig.3 *Flow of distributed parallel processing on Gfarm*

3.2 Results

Figures 4 and 5 (a) present the results of processing both kinds of data listed in Table 2 (relation between the number of data files and execution time). Table 3 lists the values of parallelization efficiency $\eta(n) = (T_1/T_n)/n$ for processing 1,000 files, with T_1 denoting the processing time per computer and T_n the processing time spent by n computers. As can be seen from Fig. 4, GEOTAIL/LEP data having larger file sizes took less time to process in distributed parallel processing than in sequen-

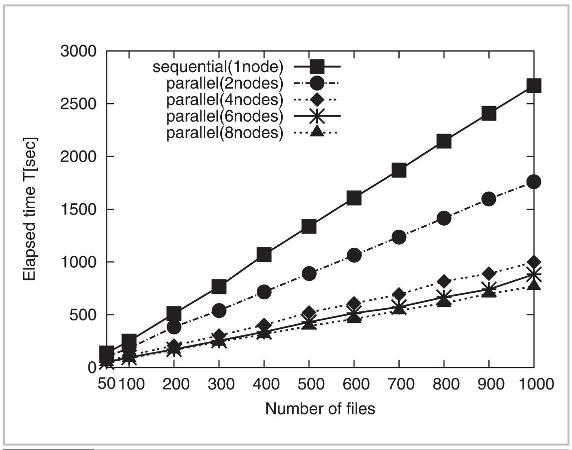

Fig.4 *Comparison of processing time for distributed parallel processing of GEOTAIL/LEP data with sequential processing*

(■ denotes sequential processing, ● the parallel processing on two nodes, ◆ the parallel processing on four nodes, ✳ the parallel processing on six nodes, and ▲ the parallel processing on eight nodes.)

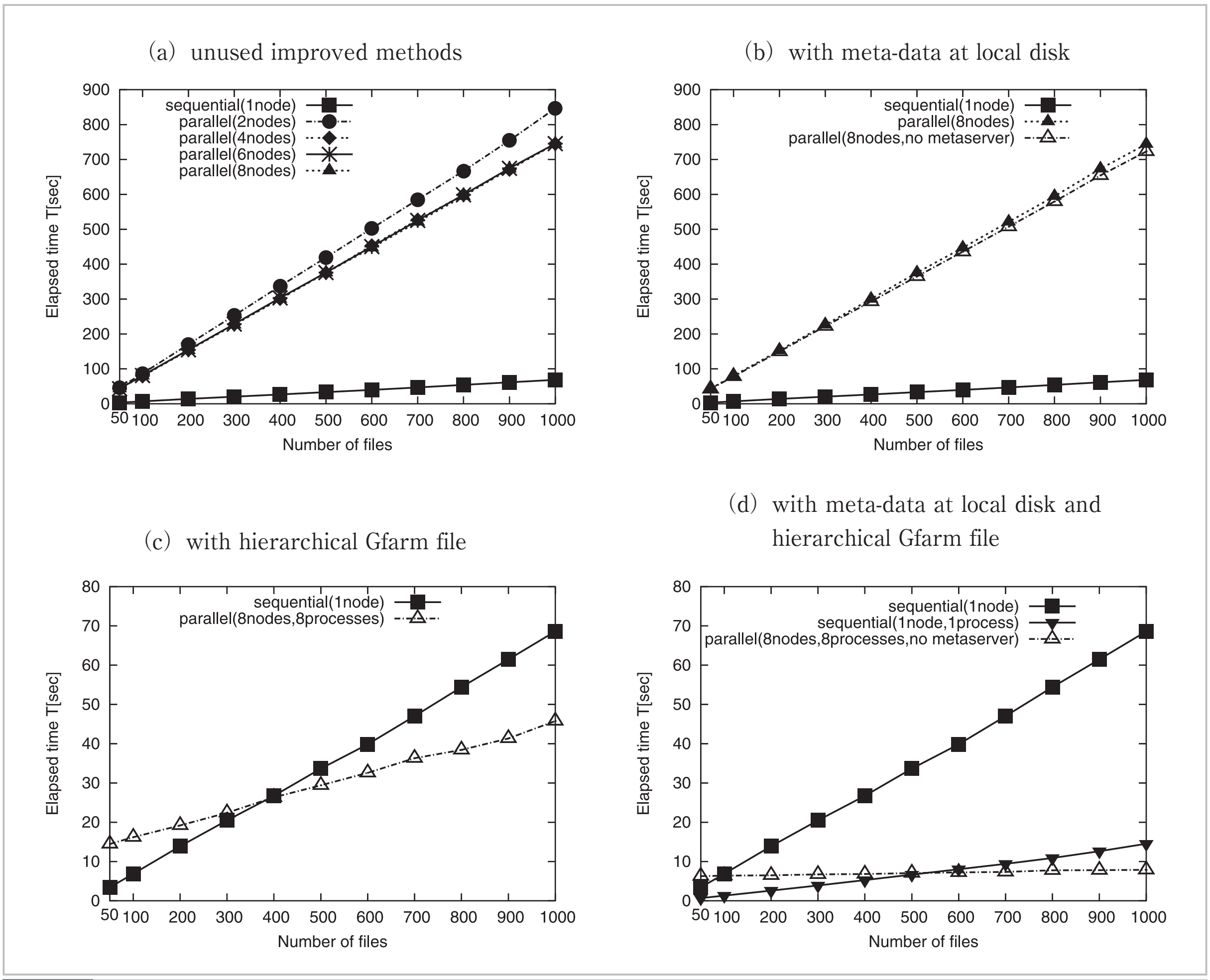

Fig.5 *Comparison of processing time for distributed parallel processing of GEOTAIL/orbit data with sequential processing*

(■ denotes sequential processing, ● the parallel processing on two nodes, ◆ the parallel processing on four nodes, ✳ the parallel processing on six nodes, ▲ the parallel processing on eight nodes, △ the parallel processing on eight nodes (using improved method) and ▼ the sequential processing (of multiple files in a single process): (a) not using improved method, (b) using meta-data local caching, (c) hierarchically structured segment files, and (d) using meta-data local caching and hierarchical segment files.

Table 3 *Parallelization efficiency η (%) for processing 1,000 files*

Number of nodes (n)	2	4	6	8
(a)GEOTAIL/LEP	75.90	66.85	50.43	43.39
(b)GEOTAIL/Orbit	4.05	2.30	1.53	1.15

tial processing, registering parallelization efficiency $\eta(8)$ of about 43.4% with eight nodes running in parallel.

In contrast, GEOTAIL/orbit data having smaller file sizes took less time to process in sequential processing than in distributed parallel processing, registering lower paralleliza-

Table 4 *Time spent processing each file of GEOTAIL/orbit*

(A) scheduling, (B) file loading, (C) output processing

Process	Elapsed Time[sec]		Time Lag
	Sequential (Variance)	Parallel (Variance)	Average [sec]
(A)Scheduling Phase (Fig. 3-(1)～(8))	0.0122 (0.0000138)	5.3317 (0.0608658)	-5.3195
(B)Reading File Phase (Fig. 3-(9)～(11))	0.0615 (0.0000619)	0.9784 (0.0031524)	-0.9169
(C)Output Phase (Fig. 3-(12))	0.0002 (0.0000002)	0.0772 (0.0001900)	-0.0770
Total Time[sec]	0.0739 (0.0000861)	6.3873 (0.0724450)	-6.3134

tion efficiency as listed in Table 3 (b). This is the result of overhead during distributed parallel processing exceeding the cuts in processing time derived from parallelization.

The overhead is associated with (A) scheduling time (Fig. 3-(1) to (8)), (B) meta-database access time in loading files (Fig. 3-(9) to (11)), and (C) output processing time (Fig. 3-(12)) as mentioned in Section **3.1**. Table 4 lists the time spent processing each file of GEOTAIL/orbit data for (A) through (C) in sequential and distributed parallel processing. Time lags indicated in the table represent overhead that never occurs in sequential processing.

Each process runs independently in distributed parallel processing on Gfarm. The more nodes running in parallel, the worse parallelization efficiency becomes as can be seen from Table 3. This is because the higher the number of nodes running in parallel, the more congested access to the meta-database becomes when referencing the entity files of Gfarm files, thereby resulting in degraded server response.

3.3 Improvement in file name reference time by caching meta-data

This section describes attempts to cut the meta-database access time (Fig. 3-(9) to (11)) involved in loading the files (B) having the second largest effects as listed in Table 4. The overhead in loading the files (B) in Table 4 arises from communication with the meta-data server, despite the presence of the entity files of Gfarm files on the local disk of each file system node. When files are available on the local disk, performance can be improved by resolving the file names through internal caching. More specifically, a list of files to be processed on each file system node is placed as a meta-data file in text format on the local disk beforehand, and then loaded into the file system node. Figure 5 (b) shows the results of the same experiment described in Section **3.1** as conducted on GEOTAIL/orbit data using the improved method. The results of sequential and parallel processing on eight nodes shown in Fig. 5 (a) are reprinted in Fig. 5 (b) as "sequential (1 node)" and "parallel (8 nodes)." Evidently from Fig. 5 (b), the meta-database access time in loading files has been cut by using the improved method, but with a small degree of improvement from "parallel (8 nodes)."

3.4 Configuring Gfarm files in hierarchical structure for better scheduling time

This section describes attempts to cut the meta-database access time (Fig. 3-(1) to (8)) involved in loading the files (A) having the most significant effects as listed in Table 4. The number of processes that launch upon the scheduling on each file system node matches the number of segment files that comprise the Gfarm files. Therefore, when segment files are structured hierarchically to simplify their process configuration as shown in Fig. 6, the need to launch as many processes as there are data files would be eliminated, thereby reducing the time required for scheduling as indicated in Table 4 (A). A function for maintaining Gfarm files in their segment files is scheduled for implementation in Gfarm v2[14], but since no function for hierarchically structuring segment files had yet to be implemented in Gfarm when this paper was prepared, multiple observation data files were combined into a single meta-data file so that all of these files could be processed in a single process (Fig. 2 (c)).

Figure 5 (c) shows the results of the same experiment described in Section **3.1** as conducted on GEOTAIL/orbit data using the improved method. "Sequential (1 node)" in the diagram has been reprinted from the

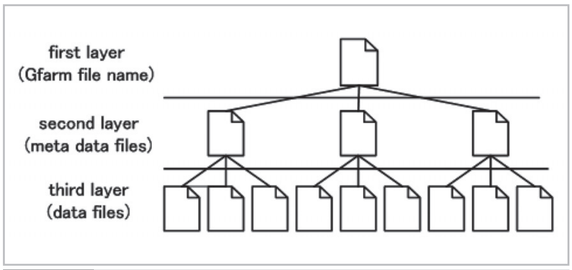

Fig.6 *Hierarchical Gfarm file structure*

results of sequential processing in Fig. 5 (a). With the improved method, larger overhead occurs at commissioning when compared with “sequential (1 node)” since jobs are submitted from a client machine, though load balancing helps reduce the slope of the graph, thereby offering higher efficiency than in sequential processing with more files being processed.

3.5 Improvements derived from combined use of meta-data caching with hierarchically structured Gfarm files

Figure 5 (d) shows the results of applying both improved methods described in Figs. 5 (b) and (c), in “parallel (8 nodes, 8 processes, no meta-data server.” In the diagram, “sequential (1 node)” has been reprinted from the results of sequential processing given in Fig. 5 (a), where “sequential (1 node)” has been processed in a single process to equalize the relation conditions with “parallel (8 nodes, 8 processes, no meta-data server).”

With the improved methods, larger overhead occurs at commissioning when compared with “sequential (1 node, 1 process)” since jobs are submitted from a client machine, as in the case of Fig. 5 (c), though load balancing offers higher efficiency than in “sequential (1 node, 1 process)” when more files are processed.

Figure 7 (a) shows the results of an experiment similar to that in Fig. 5 (d) as conducted on 5,000 to 50,000 files, with parallelization efficiency for the varying number of nodes running in parallel in “sequential (1 node, 1 process)” in (a) being plotted in Fig. 7 (b). As can be seen from Fig. 7 (b), parallelization efficacy tends to rise in line with more files, regardless of the number of nodes running in parallel. This is due to the narrowing ratio of overhead occurring at commissioning associated with increases in data processing time as shown in Fig. 5(d). High parallelization efficiencies above 95% are derived from processing 50,000 files on a varying number of nodes running in parallel.

3.6 Discussions

Parallel distributed processing on Gfarm promises high parallelization efficiency without depending on the number of nodes running in parallel, because data-intensive processing keeps the individual processes independent of

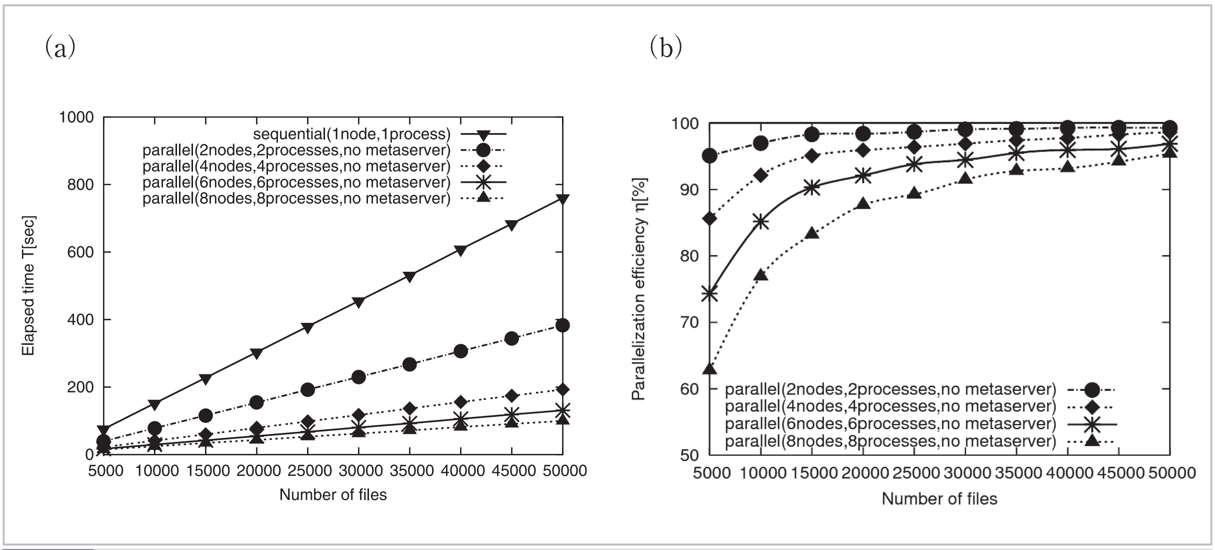

Fig.7 *Comparison of distributed parallel processing of GEOTAIL/orbit data (using meta-data local caching and hierarchical segment files) with sequential processing*

(▼ denotes the processing of multiple files in a single process, ● the parallel processing on two nodes, ◆ the parallel processing on four nodes, ✳ the parallel processing on six nodes, and ▲ the parallel processing on eight nodes) : (a) relation between the number of files and execution time, (b) parallelization efficiency one another, without involving inter-process communication. The parallel distributed processing of small files where multiple nodes running in parallel share a single meta-data server is likely to increase the proportion of overhead, however, as shown in Fig. 5 (a), making sequential processing a more efficient choice. Even though the scheduling time—a dominant aspect of overhead in distributed parallel processing—and the meta-database access time in loading files are cut, there is still the overhead of submitting jobs; therefore, the sequential processing of multiple files in a single process would prove more efficient when the number of files involved is less than a certain level as shown in Fig. 5 (d). This is one reason why Gfarm targets large sizes of data, such as write-at-once data.

Assuming the availability of one file of satellite-specific data per day, the 50,000 data files summarized in Fig. 7 represent the size of data collected from 14 satellites over 10 years, suggesting that enhancements to meta-data collection performance might make distributed parallel processing on Gfarm suitable for multi-point, long-tem satellite observation data in the field of solar-terrestrial physics. Ongoing enhancements have evolved from the findings of work conducted with Gfarm v1, including on-memory database processing on Gfarm v2 and minimized meta-data reference counts[15].

4 Parallel 3D visualization processing of computer simulation data

4.1 Status quo of large-scale parallel visualization

The methods for implementing large-scale parallel visualization proposed thus far include splitting space regions and processing data distributed by a visualization method[16]-[18]. These methods work on the visualization of grid sizes in excess of per-node computer performance. Yet, these methods require an individual to collaborate in visualizing single time steps. Depending on the computer configuration and visualization region, variations in visualization processing time might arise from node to node, making these methods unfit for use in performing data-intensive processing tasks, such as multi-time-step long-term visualization. This section applies the system depicted in Fig. 1 to the 3D visualization of data yielded from STARS-managed Real-time Magnetosphere Simulation (hereinafter “real-time simulation”)[11] conducted by NICT to examine the usefulness of parallel visualization methodology.

4.2 Parallel visualization in the time-series direction

One numeric data file is generated from each time step in real-time simulation. In this experiment, simulation data from 150 time steps is visualized in parallel on the system shown in Fig. 1. The simulation data used for visualization was magnetic field three-component data calculated by real-time simulation, measuring about 80 MB per time step (one file) (12 GB in total). The 3D object file generated through visualization is about 1.2 MB per time step (180 MB in total). The general-purpose visualization application AVS/Express 7.1.1[19] was used to conduct visualization in magnetic lines of force.

Figure 8 shows the parallel visualization method used for this work. First, data files are allocated to individual system nodes by exe-

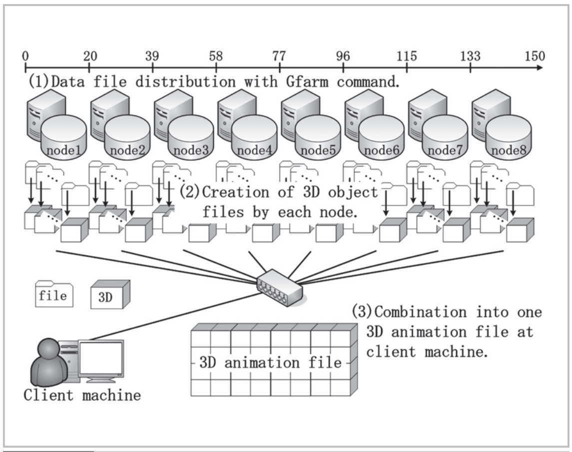

Fig.8 Distributed parallel processing for visualization

cuting a Gfarm command (Fig. 8-(1)). In the experiment, 150 files were divided into eight groups and allocated equally to each node. Next, each system node visualizes an allocated data file on the local disk to create a 3D object as a Gfarm file (Fig. 8-(2)). This sequence of processing is repeated for each additional file allocated to the system node. Lastly, the resultant 3D objects are merged into one animation file on the client machine (Fig. 8-(3)).

4.3 Results

Figure 9 shows the results of visualization conducted in Section **4.2**, with the visualization processing time spent for each time step given in Fig. 10 (a), and the total visualization processing time in Fig. 11 (a). The longest time in Fig. 11 (a) was 10,962 seconds (about 183 minutes) or equal to the total execution time of parallel visualization processing. Similar visualization processing executed as sequential processing on a single file system node took about 4.5 times longer, or 49,726 seconds (about 829 minutes). Though these findings attest to the usefulness of parallel visualization processing, the gap of 8,160 seconds (about 136 minutes) recorded between the shortest and longest visualization processing times in Fig. 11 (a) with a low parallelization efficacy of 56.7% suggests room for further improvement.

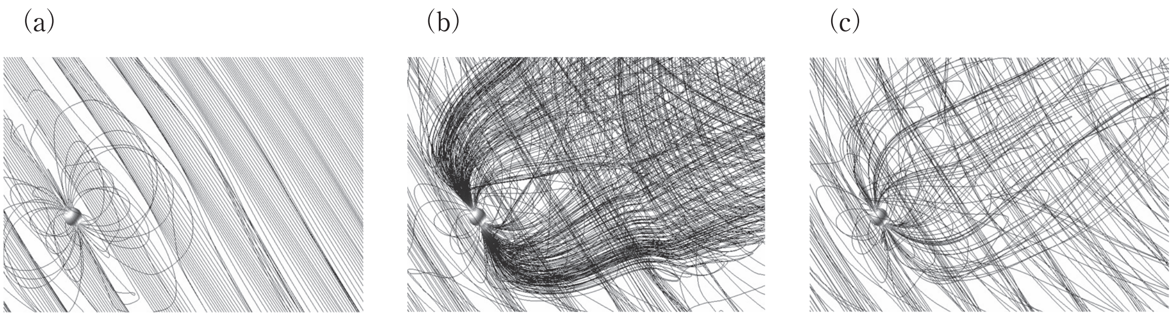

Fig.9 *3D visualization of the terrestrial magnetosphere*

(a) first step, (b) 82nd step, (c) 100th steps

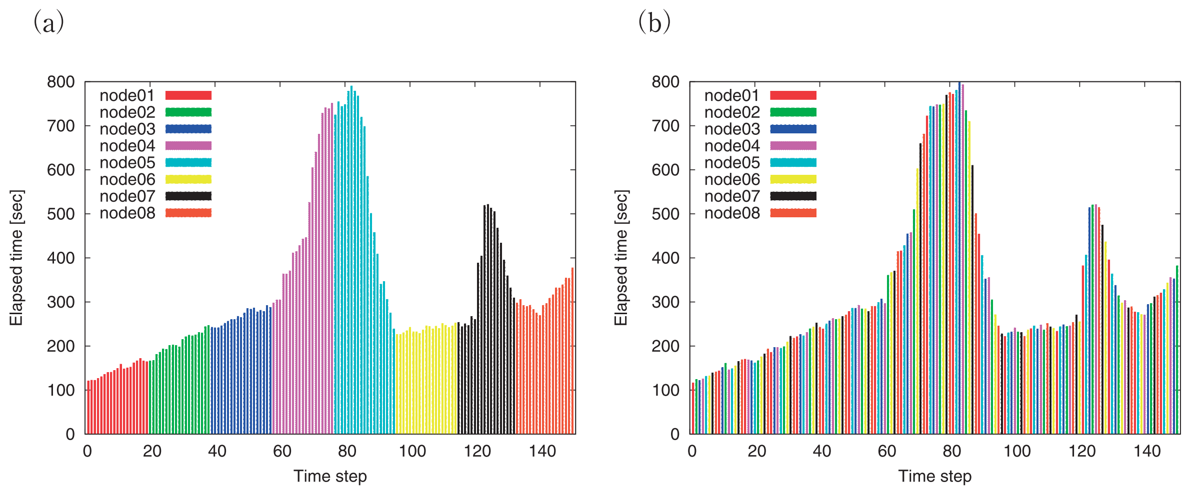

Fig.10 *Execution times by time step*

(a) files allocated uniformly to nodes, (b) files allocated by FIFO scheduling

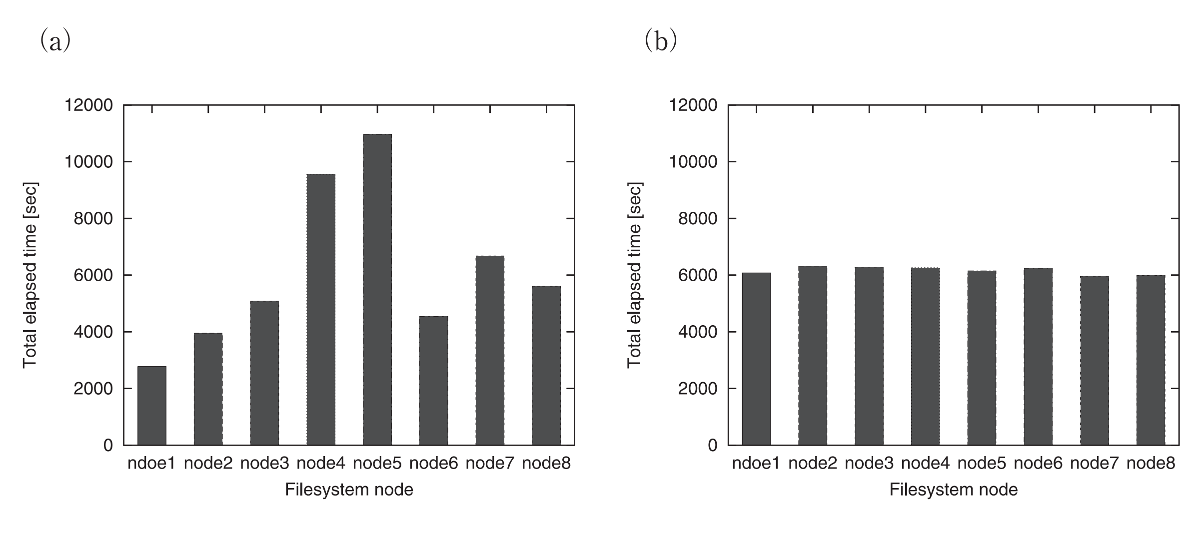

Fig.11 *Total of visualization processing times on individual file nodes*
(a) files allocated uniformly to nodes, (b) files allocated by FIFO scheduling

4.4 Review of load balancing optimization

When many magnetic lines of force are involved, the visualization processing time required to plot those lines increases as shown in Fig. 9 (b). This in turn increases the processing times on nodes 4 and 5 in Fig. 11 (a), thereby disrupting the load balance. This occurs because the Gfarm scheduler initially allocates jobs to all segment files in a batch (Fig. 12 (a)). This method, however, would not allow load balancing to be optimized in case of non-uniform file processing times. Batch scheduling at job submission has therefore been improved to the FIFO scheduling solution (Fig. 12 (b)), whereby jobs are placed in a queue, and then dequeued and allocated in sequence to the job ending nodes. With this method, the files allocated to individual nodes are unpredictable, and if an allocated file does not reside on the local disk, the data transfer time would be added to the data processing time. In this experiment, all data files are replicated on the individual nodes beforehand by using the Gfarm gfrep command.

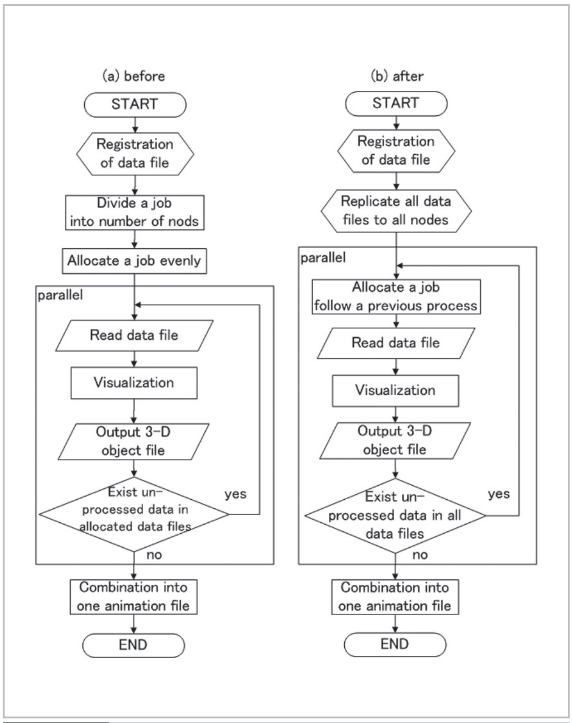

Fig.12 *Visualization scheduling flowchart*
(a) files allocated uniformly to nodes,
(b) files allocated by FIFO scheduling

Figure 10 (b) shows the visualization processing time per step based on the improved method; Figure 11 (b) shows the total visualization time of all file system nodes. The gap between the shortest and longest visualization processing times has been narrowed to 360 seconds (6 minutes) in Fig. 11 (b), with load balancing being more optimized than in Fig. 11 (a). The total execution time for parallel visualization processing has also been cut to 6,360 seconds (106 minutes). High parallelization efficiency is thus promising even in distributed parallel processing where data files vary in terms of data processing granularity.

4.5 Discussions

Because the method proposed by this paper optimizes load balancing regardless of the number of nodes running in parallel, a scalable distributed parameter is made possible. Given the need to replicate all data files on individual nodes, however, a tradeoff exists between parallelization efficacy and data file duplication time. In this experiment, the process of duplicating all data files measuring 12 GB on all nodes took about 77 minutes to complete. The two scheduling methods did not show noticeable differences in processing time for the first visualization session. Since the replication process is bypassed in the second and subsequent visualization sessions, the proposed method would prove more efficient when visualizing the same set of data by varying the visualization parameters. The evolving task is how to resolve problems by pipelining the visualization process and the process of transferring data files to the file system nodes.

5 Conclusions

Observation data continues being increasingly digitized in many fields of geoscience, resulting in growing volumes of all kinds of data, including computer simulation data. This paper proposed a distributed data-type/data-intensive processing system that builds on STARS and Gfarm as a solution to large-scale data processing in the context of distributed data management and data processing environments in the field of solar-terrestrial physics. The usefulness of a system built of eight file system nodes was verified using satellite observation and computer simulation data. Distributed parallel processing proved useful even when processing small files of data, given the combined use of meta-data local disk caching with hierarchical segment files. In the parallel 3D visualization of computer simulation data varying in terms of data processing granularity, optimized load balancing through FIFO scheduling yielded parallelization efficacy as high as 97.6% in processing on eight modules running in parallel.

Satellite observation data and computer simulation data have a complementary relation in the field of solar-terrestrial physics, calling for the implementation of a multidisciplinary integrated data analysis environment migrating from existing goal-specific data analysis environments. The research group formed by the authors has attempted to merge and analyze these two different sources of data[20]. The authors sought to develop research findings to realize an integrated, multi-purpose data processing environment operating on the principles of integral-intensive processing involving datasets of heterogeneous data varying in file size or data processing granularity. The authors also hope that file system nodes will be installed at data sites that provide meta-data, in order to standardize the schemes of STARS meta-damage management in a virtual directory that can be shared among virtual organizations (VOs) in the field of solar-terrestrial physics, thereby creating a data-intensive processing environment.

The authors are deeply indebted to Associate Professor Iku Shinohara at the Japan Aerospace Exploration Agency (JAXA) for his valuable assistance extended to their work. This work was supported by the Grant-in-Aid for Creative Scientific Research, “The Basic Study of Space Weather Prediction” (17GS0208, Head Investigator: K. Shibata), from Japan’s Ministry of Education, Culture, Sports, Science, and Technology. The work was conducted by using satellite observation data made available from the JAXA Center for Science-satellite Operation and Data Archive, and the Research Institute for Sustainable Humanosphere, Kyoto University. NICT real-time terrestrial magnetosphere simulation data was computed on SX-8R (NICT).

References

1 eGY: The electronic Geophysical Year, http://www.egy.org/, 2010.

2 NASA Polar, Wind, and Geotail Projects, http://www-istp.gsfc.nasa.gov/, 2010.

3 SCOPEWG, http://www.stp.isas.ac.jp/scope/index.html, 2010.

4 D. Matsuoka, Ken T. Murata, S. Fujita, T. Tanaka, K. Yamamoto, and E. Kimura, “Analyses of 3D Structure of Magnetic Flux Ropes via Global MHD Simulations,” Transaction of the Visualization Society of Japan, Vol. 28, No. 6, pp. 38–46, 2008.

5 Grid Datafarm - Gfarm file system, http://datafarm.apgrid.org/, 2010.

6 O. Tatebe, Y. Morita, S. Matsuoka, S. Sekiguchi, and N. Soda, “Grid Datafarm Architecture for Global Petascale Data Intensive Computing,” High Performance Computing Symposium, pp. 89–96, 2002.

7 N. Yamamoto, O. Tatebe, and S. Sekiguchi, “Performance Evaluation of Astronomical Data Analysis Infrastructure on Grid Datafarm Architecture,” Symposium on Advanced Computing Systems and Infrastructures 2004, pp. 233–240, 2004.

8 GEO Grid, http://www.geogrid.org/, 2010.

9 SPDF - Satellite Situation Center Web (SSCWeb), http://sscweb.gsfc.nasa.gov/, 2010.

10 DARTS at ISAS/JAXA, http://darts.isas.jaxa.jp/, 2010.

11 NICT Real-Time Magnetosphere Simulation,
http://www2.nict.go.jp/y/y223/simulation/realtime/home.html, 2010.

12 Ken T. Murata, M. Okada, F. Abe, T. Araki, and H. Matsumoto, “A Design and Estimation of Distributed Meta-database for Solar-terrestrial Physics Observation Data,” Transactions of Information Processing Society of Japan, Vol. 43, No. SIG12(TOD16), pp. 115–130, 2002.

13 Ken T. Murata, “A World-wide Distributed Database System for ISTP Project,” The Transactions of the Institute of Electronics, Information and Communication Engineers B, Vol. J86-B, No. 7, pp. 1331–1343, 2003.

14 O. Tatebe, N. Soda, and S. Sekiguchi, “Gfarm v2: Design and Implementation of Global Virtual File System,” Special Interest Group Notes of IPSJ, 2004-HPC-99, SWoPP2004, pp. 145–150, 2004.

15 O. Tatebe and N. Soda, “Implementation and Evaluation of Gfarm v2 Global Distributed File System,” The Special Interest Group Notes of IPSJ, 2007-HPC-113, pp. 7–12, 2007.

16 Y. Suzuki, “Large-Scale Visualization System for Grid Environment,” The 56th Japan National Congress for Theoretical and Applied Mechanics, pp. 33–34, 2007.

17 K. Nakajima and L. Chen, “Parallel visualization method for large-scale distributed data sets in scientific simulations with background voxel's,” The Special Intesrest Group Notes of IPSJ, Vol. 2006-HPC-107, No. 87, pp. 91–96, 2006.

18 T. W. Crockett, “An introduction to parallel rendering,” In Parallel Computing, p. 23 (7): 819L843, 1997.

19 AVS/Express, http://www.avs.com/software/soft_t/avsxps.html, 2010.

20 Ken T. Murata, K. Yamamoto, D. Matsuoka, E. Kimura, H. Matsumoto, M. Okada, T. Mukai, J. B. Sigwarth, S. Fujita, T. Tanaka, K. Yumoto, T. Ogino, K. Shiokawa, N. A. Tsyganenko, J. L. Green, and T. Nagai, “Development of the Virtual Earth’s Magnetosphere System (VEMS),” Advances in Polar Upper Atmosphere Research, Vol. 19, pp. 135–151, 2005.

YAMAMOTO Kazunori

Research Student, Fuclty of Engineering, Ehime University

Virtual Observation, Semantic Web

***KIMURA Eizen**, Ph.D.*

Associate Professor, School of Medicine, Ehime University

Medical Informatics, Health Economics

***MURATA Ken T.**, Ph.D.*

Group Leader, Space Environment Group, Applied Electromagnetic Research Center

Information Communication Technology

***TATEBE Osamu**, Ph.D.*

Associate Professor, Graduate School of Systems and Information Engineering, University of Tsukuba

Parallel and Distributed Computing

***MATSUOKA Daisuke**, Ph.D.*

Research Scientist, Earth Simulator Center, Japan Agency for Marine-Earth Science and Technology

Solar-Terrestrial Physics, Scientific Visualization

***MIYACHI Hideo**, Dr. Eng.*

General Manager, Technical Department, Visualization Division, KGT Inc.

Visualization Method