A Study on Application of Grid Computing to Medical Data Analysis

（医用データ解析へのグリッドコンピューティングの応用に関する研究）

2002

Susumu Date

A Study on Application of Grid Computing to Medical Data Analysis

（医用データ解析へのグリッドコンピューティングの応用に関する研究）

2002

Susumu Date

A dissertation
submitted to the Graduate School of Engineering
of Osaka University
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Engineering.

Summary

Recently, the lack of computational power for analyzing scientific data has become a major problem in all scientific fields. The rapid development of measurement technologies provides scientists and researchers with a large amount of scientific data. These technologies enhance the possibility of humankind finding new theories through the analysis of scientific data. However, no effective way of analyzing scientific data within a realistic time exists. In reality, it takes a long time to adequately analyze scientific data, which prevents science from proceeding to the next step.

In the last few decades, processor technology has dramatically advanced. As a result, the amount of computational tasks a single processor can complete in a certain time has remarkably improved. However, even current cutting-edge processing technologies cannot meet the computational needs common in science because scientific data must be investigated on the order of Giga Bytes, even Peta Bytes in such fields as Physics, Astronomy, Bioinformatics and Medicine.

Researchers have exploited a variety of parallel processing techniques for solving computationally-intensive scientific problems. Some researchers use a supercomputer, or high-performance Symmetric Multiple Processors (SMP) to simulate the behavior of molecules under an assumed circumstance. Other researchers use a Linux box composed of multiple personal computers connected to a dedicated fast network such as Myrinet to generate visualization data volume. Presently, the latter form of parallel processing, that is, parallel processing with network and multiple computers, is the major trend for parallel processing. This trend can be explained with respect to the ratio of performance to cost. However, even these parallel processing technologies cannot satisfy the demands on computational power for analyzing the data acquired from scientific devices such as Compact Muon Solenoids (CMS), Magnetoencephalography (MEG), Ultra High Voltage Electron Microscopy (UHVEM). Therefore, a computational platform that can solve this serious problem of computational power is increasingly demanded.

In addition, the geographical distribution of knowledge (by scientists) and technologies (for advanced scientific devices) is also a serious problem in all scientific fields. Science, although aiming for the simple and elegant, tends to become complicated. As a result, more than one specialist in more than one field of science is often needed to solve a scientific problem. However, such specialists are rarely at the same location but are often geographically distributed. Additionally, advanced scientific devices are also geographically distributed because of the costs for their purchase and maintenance. Such scientific devices need to be shared among scientists and researchers for the advance of science.

Emerging Grid computing is categorized into a different class of high performance computing from the class of conventional parallel computing such as computing with a supercomputer and a network of workstations (NOW). The difference between the two can be clarified in terms of the following two points; integration of heterogeneity and pervasive and flexible access to a variety of computational resources. Until recently, parallel computing assumed a homogeneous element in computers participating in the computation. For example, the processor architecture, operating system, and administrative domain needed to be the same. However, the new emerging Grid enables flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions and resources, without assuming a homogeneity. In addition, the Grid aims to integrate not just high-performance computers but also advanced scientific devices. This integration of high-performance computers with advanced scientific devices will solve common problems in many fields of science.

Brain science, for example, has become an increasingly important field of science in the rapid development of the sciences. In most developed countries, an aging society has appeared. A variety of brain diseases have also increased. For the effective treatment of some brain diseases and disorders, early detection and therapy is reported to be clinically effective. For this purpose, doctors need to adequately analyze functional brain data as quickly as possible. However, doctors cannot always detect early symptoms of brain diseases despite today's marriage of science and technology because the brain's mechanisms are not easily revealed. Thus, to be able to perfectly reveal brain functions is an important goal in the brain sciences as well as for humankind in general.

Magnetoencephalography is a highly sophisticated medical device that dynamically captures neural activities inside the brain. It allows a doctor to investigate

a patient's brain activity non-invasively and accurately. Until today, both non-invasiveness and accuracy in the measurement of brain activities were unachievable. This is why MEG is a promising brain imaging device. However, the measurement of brain activities with MEG is not often performed despite its possitive features because of the large computational time necessary and the diverse geographical distribution of knowledge and technologies, problems common in many scienific fields.

In this dissertation, the application of Grid computing to MEG data analysis is explored. The goals of this research are to support and contribute to advances in brain science and to testify to the effectiveness and usefulness of Grid computing to the brain science. For the purpose, the application of the Grid to MEG data analysis is performed from two different aspects. In the first aspect, the Grid is viewed as a building block for a medical infrastructure. In the second aspect, the Grid is viewed as a software technology for building a specific application system. More specifically, the first aspect targets the fundamental problems in Grid computing for medical data analysis. An example includes security problems related to a patient's privacy. On the other hand, the second aspect solves problems specific to brain data analysis including MEG data analysis. Therefore, in this dissertation MEG data analysis is explained in detail.

Furthermore, this dissertation proposes a new programming model for Grid computing. Presently, two major programming models are used for building application systems with Grid computing. The author considers the problems in the two existent programming models, by actually building the MEG data analysis system with each model. The proposed programming model based on consideration of the two existent models allows a user to develop a Grid-enabled system. The evaluation of the proposed programming model is presented from both qualitative and quantative viewpoints. The author shows that the proposed programming model dramatically reduces the user's development costs.

This dissertation is organized as follows: In Chapter 1, research backgrounds and purposes from both medical and engineering aspects are examined. In Chapter 2, first the problems involved in building the infrastructure for medical data analysis with the Grid are considered and then, the following three problems are clarified and solved; 1) the lack of an encrypted environment, 2) the shortage of IP addresses, and 3) the lack of data access structure with location-transparency.

In Chapter 3, the MEG data analysis is described. This chapter begins with a description of what MEG is and continues with explanation of why the Grid is

necessary to MEG data analysis. Next, wavelet cross-correlation analysis which has been adopted in this research is introduced to the readers. Next, the goals to be achieved for supporting the MEG data analysis are clarified. To perform wavelet cross-correlation analysis efficiently, efficient and effective parallel processing methods for wavelet cross-correlation analysis are discussed. In addition, a visualization software for wavelet cross-correlation analysis is described.

In Chapter 4, the application of Grid computing to the MEG data analysis system is presented. First, two existent programming models for Grid computing are used for the development of the MEG data analysis system. Through the building of the system, the effectiveness of the Grid is verified. Next, a new programming model, or *hybrid programming model* is proposed. Through the building of the MEG data analysis system with the hybrid programming model, the effectiveness of the programming model is also verified. In the hybrid programming model, the reduction of users' development costs is also discussed.

Chapter 5 concludes the present study and suggestions are offered for future research. The overall goal of this dissertation is to verify that the application of Grid computing to medical data analysis is highly effective from the viewpoint of application building.

Contents

1 Introduction **1**

1.1 Medical Background 1

1.2 Engineering Background 3

1.3 Outline of the Dissertation 4

2 Grid Computing as Medical Infrastructure **7**

2.1 Introduction 7

2.2 Encrypted Data Environment 9

2.2.1 Authentication Infrastructure with Private Certificate Authority 9

2.2.2 IPv6/IPSec-enabled Globus 11

2.3 Implementation of Data Access Structure with Location-transparency 16

2.3.1 Self-Certifying File System 17

2.3.2 Problem in Introducing the Self-Certifying File System 18

2.3.3 Solution 18

2.4 Concluding Remarks 19

3 MEG Data Analysis **21**

3.1 Introduction 21

3.2 Magnetoencephalography (MEG) 21

3.3 Problems with MEG Data Analysis 22

3.4 Wavelet Cross-correlation Analysis 24

3.5 Issues in MEG Data Analysis with Wavelet Cross-correlation Analysis 27

3.6 Parallel Computing for Wavelet Cross-correlation Analysis 29

3.7 Visualization Software 31

3.7.1 Visualization for Wavelet Analysis 32

3.7.2 Visualization for Cross-correlation Analysis 35

3.8 Concluding Remarks 37

4 An Application of the Grid to MEG Data Analysis **39**
4.1 Introduction 39
4.2 System Overview 40
4.3 An Approach with the Globus Programming Model 42
4.3.1 Simulated Grid Environment 42
4.3.2 A Design Strategy for Efficient Computation 43
4.3.3 Implementation 46
4.3.4 Programming Difficulties 49
4.3.5 Performance Evaluation 53
4.4 An Approach with MPICH-G Programming Model 55
4.4.1 Simulated Grid Environment 55
4.4.2 A Design Strategy for Efficient Computation 56
4.4.3 Implementation 57
4.4.4 Performance Evaluation 59
4.5 An Approach with Hybrid Programming Model 63
4.5.1 Programming Template 63
4.5.2 Software Architecture 66
4.5.3 Communication between Modules 67
4.5.4 A Design Strategy for Efficient Computation 68
4.5.5 Implementation 69
4.5.6 Evaluation 72
4.6 Concluding Remarks 78

5 Conclusion **79**
5.1 Concluding Remarks 79
5.2 Future Directions 82

References **87**

List of Publications by the Author **92**

Chapter 1

Introduction

1.1 Medical Background

Brain science has recently become an increasingly important scientific field. In most developed countries, the transition to an aging society has rapidly proceeded. In such societies, a variety of diseases associated with aging can be easily predicted to increase in number. Examples of such diseases include forms of dementia such as Alzheimer and celebrovascular diseases. In general, for the diseases caused by a disorder of brain function, detecting a symptom of the brain disease early has great importance from a clinical standpoint. Also, early therapy is important. In order to realize early detection and therapy for brain diseases and disorders, a precise understanding of brain functions and an efficient analysis of brain functional data are indispensable.

However, there are two major problems preventing doctors from efficiently detecting the symptoms of diseases caused by functional brain disorders:

1. Geographical distribution of both knowledge and technologies regarding the analysis and diagnosis of brain functions, and

2. Long computational time for the analysis of brain functions.

In general, the analysis of brain functions is performed in the following three stages; data acquisition stage, data analysis stage, and implementation stage of the analysis result. Each stage requires a specialist who has been engaged in the correspondent stage, or an advanced scientific device, or both. For example, in the stage of data acquisition, a brain imaging device such as ECoG (Electroencephalography), EEG (Electrocorticography), fMRI (functional Magnetic Resonance Instrument) or

MEG (Magnetoencephalography) is required as well as specialists who have the knowledge of how to use these devices. In the second stage, specialists of signal processing technique are required. Also, high-performance computers are required for the computation of their analysis methods. In the final stage, specialists of diagnosis, or doctors are required. Unfortunately, these stages are rarely performed in the same location. In other words, these three stages are often geographically distributed.

In the current scientific environment, the analyses of brain functions are often performed as follows. First, brain data is acquired from a brain imaging device. Next, the acquired brain data is passed to a specialist of signal processing. Often, the specialist works at a scientific organization other than the organization where the data acquisition stage is performed. After receiving the data, the specialist analyzes the brain data with his or her own software on a single personal computer or workstation. After the computation of the analysis is completed, the specialist gives the results of the analysis and the data acquired from the brain imaging device to a doctor who implements the results of the analysis. This common situation makes the time taken for the analysis of brain functions long and inefficient, which results in the situation that doctors miss the opportunity of early detection of brain diseases.

Additionally, the amount of medical data to be acquired from a medical device tends to increase. The recent development of measurement technologies has improved both temporal and spatial resolution. These technologies enhance the possibility of understanding the body of human being. However, the increase in data acquired from scientific devices simultaneously has led to a long computational time for the analysis of the acquired data. The reason for this lengthy computational time can be explained by the fact that most analyses of brain data are often performed on a single-processor basis even though that parallel computing techniques are mature and available. In fact, analyses are rarely finished within a realistic time. Therefore, the analyses of brain functions are often performed inadequately.

In order to improve this inefficient situation in brain science, we have to solve the two problems of geographical distribution of both knowledge and technologies and long computational time. To this purpose, this research realizes an MEG data analysis system, or a kind of brain data analysis system that allows a doctor to analyze the brain data acquired from the MEG with ease on the Internet. In the MEG data analysis system on the Internet, the geographical distribution of both

knowledge and technologies can be solved.

The use of the Internet, furthermore, provides us with a chance of reducing the analysis time of brain functions. Presently, an infinite number of various high-performance computers are connected on the Internet. By making maximum use of these computers, this research shows how to considerably reduce analysis time.

The medical purpose of this research is to support advancements in brain science using the MEG data analysis system, which has been designed and implemented on the Internet. More specifically, the author proposes the application of a new emerging Grid to medical data analysis. Through the building of the MEG data analysis system, the effectiveness of Grid computing for medical data analyses is verified.

1.2 Engineering Background

The form of high performance computing has changed dramatically over time. Since mainframe computers emerged in the middle of 1960s, the main form of high performance computing had been parallel computing with a single big computer such as SMP (Symmetric Multiple Processors) computers and supercomputers. However, from the middle of 1980s, the form of high performance computing changed to high performance computing with multiple computers connected on the Internet. The transition of the high-performance computing form has been accelerated by the development of network technologies. Network of workstations (NoW) is one example.

From the 1990s until now, researchers have developed various forms of network distributed computing. Peer-to-peer computing and Internet computing like seti@home [1], folding@home [2] and aids@home [3] are examples. Also, Java-based object-oriented computing like Common Object Resource Broker Architecture (CORBA) is another example. In such high-performance computing backgrounds, a new emerging computing concept, or *Grid* has emerged.

In a recent paper by Dr. Ian Foster, who is a leader in the world-wide Grid project, the Grid has been described as enabling flexible, secure, coordinated resource sharing among dynamic collections of individuals, institutions, and resources [4]. Furthermore, the Grid also allows academic and industrial communities to share geographically distributed resources for their common purposes. In other words, the Grid is expected to realize a metacomputer on the Internet and to enable scientists

and researchers to work together in a virtual laboratory with the metacomputer. The application of the Grid to "Big Science" such as astronomy, physics, and medicine is promising. The common feature of such sciences is the large amount of data to be analyzed [5]. The size of the data must be investigated with Giga Bytes, or most likely, even Peta Bytes.

There are two major differences between the new emerging Grid and traditional high-performance computing. One difference is heterogeneity. Traditional high performance computing has assumed homogeneous elements such as operating system, processor architecture, and a single administrative domain. For example, cluster computing such as NoW is designed to gain high performance when the computing is performed among computers with the same architecture. In another example, internet computing such as seti@home makes full use of thousands of ordinary personal computers on the Internet but does not consider robust security well, assuming that the users of software are a flat and homogeneous relatively careful people. On the other hand, the Grid integrates a diversity of heterogeneous computational resources such as storage systems, high-performance computers, and scientific devices and provides a method of accessing such devices in a uniform manner. This method is designed to allow users to access such devices with ease.

The other difference is that the Grid shares not only high-performance computers (from ordinary personal computers to even supercomputers), but also highly sophisticated scientific devices like Compact Muon Solenoids (CMS), Magnetoencephalography (MEG), and Ultra High Voltage Electron Microscopy (UHVEM).

The overall engineering purpose of this research is to find and solve problems in new emerging Grid computing through the building of an application system using the Grid, which will hopefully also contribute to on-going world-wide Grid research. In this particular application of Grid computing to medical data analysis, the MEG data analysis system that can help doctors efficiently diagnose brain diseases.

1.3 Outline of the Dissertation

The remainder of this dissertation is organized as follows. In Chapter 2, first the problems involved in building the infrastructure for medical data analysis with the Grid are considered and then, the following three problems are clarified and solved; 1) the lack of an encrypted environment, 2) the shortage of IP addresses, and 3) the lack of data access structure with location-transparency.

In Chapter 3, the MEG data analysis is described. This chapter begins with a description of what MEG is and continues with explanation of why the grid is necessary to MEG data analysis. Next, wavelet cross-correlation analysis which has been adopted in this research is introduced to the readers. Finally, the goals to be achieved for supporting the MEG data analysis are clarified.

In Chapter 4, the application of Grid computing to the MEG data analysis system is presented. First, two existent programming models for grid computing are used for the development of the MEG data analysis system. Through the building of the system, the effectiveness of the Grid is verified. Next, a new programming model, or *hybrid programming model* is proposed. Through the building of the MEG data analysis system with the hybrid programming model, the effectiveness of the programming model is also verified. In the hybrid programming model, the reduction of users' development costs is also discussed.

Chapter 5 concludes the present study and suggestions are offered for future research. The overall goal of this dissertation is to verify that the application of Grid computing to medical data analysis is highly effective from the viewpoint of application building.

Chapter 2

Grid Computing as Medical Infrastructure

This research approach has been viewed from two different aspects. The approach from the first aspect has been performed from the viewpoint of establishing a medical infrastructure. On the other hand, the approach from the second aspect has been explored from the viewpoint of building an actual application system on the Grid environment. In this chapter, the first approach will be detailed.

2.1 Introduction

For this research, the Globus grid toolkit [6, 7] has been adopted as an implementation of the Grid, although Legion [8], Netsolve [9, 10], and so on exist as such implemantations. The Globus grid toolkit is a middleware that embodies the concept of Grid computing [11]. Generally, the Grid refers to recent, advanced technology that allows scientists to easily integrate a diversity of computational resources geographically distributed on the Internet. These resources include supercomputers and scientific instruments such as Compact Muon Solenoids. Clearly, we can gain far more computational power through the use of Grid technology than through the use of traditional parallel computing technology. Also, remote-control of scientific instruments is possible using the Grid.

In order to allow end-users without special knowledge of Internet technologies to easily utilize the Globus, the Globus provides multiple complex services essential for widely distributed parallel computing. Examples of these services include communication, security, and information management, to name a few.

More specifically, Grid Security Infrastructure (GSI) [12, 13] is the core authentication infrastructure of the Globus system. GSI allows users to use a variety of computers, even if these computers are located on different administrative domains, by allowing users to simply identifying themselves with their own credentials, used like passphrases. Meta Directory Service (MDS) [14] manages the static and dynamic information on the Grid environment. The static information manages the architecture type of processor and operating system, the amount of main memory, and so on, while the dynamic information manages the information varied over time such as the load average of processors. Dynamically-Updated Request Online Coallocator (DUROC) allows users to invoke the programs on remote computers. Global Access to Secondary Storage (GASS) [15] provides a method of accessing the file system on a remote computer. Globus I/O (Nexus) [16] [17] provides users with communication services on the Grid environment. By using these communication APIs, the users can easily build application systems with complex communication models on the Internet.

These services are extremely difficult to implement on the Internet. This difficulty is easy to understand if you imagine that there are many different administrative policies on the Internet. However, the Globus has a design strategy that offers users uniform and integrated methods to access these resources in an API-(Application Programming Interface) based manner. Accordingly, we can build a parallelized application over organizations such as clinics, hospitals and scientific institutions on the Internet with relative ease.

Although Globus provides many services essential for realizing a Grid environment, Globus has no capability of realizing a secure line on the Internet. The lack of this functionality is fatal in building any medical data analysis system on the Internet. The reason can be easily explained from the viewpoint of protecting patients' privacy.

Furthermore, Globus provides no network file system on the Grid environment that the Globus can realize. The lack of a network file system makes it difficult for doctors without detailed knowledge of the Grid and Globus grid toolkit to find medical data files of their interest on the Internet. This situation makes the analysis of brain functions inefficient.

In this chapter, these two problems will be solved. In other words, the following issues have been targeted in order to build the infrastructure for medical data analysis.

1. Data Security, and

2. A file system which provides doctors with a single disk image on the Grid environment

2.2 Encrypted Data Environment

Recently, the transmission of confidential data like medical data to the public network, or to the Internet remains a serious problem, while we increasingly demand use of the Internet for the transfer of such data. In order to transfer medical data privately and securely, we need to achieve the following two goals:

1. Mutual authentication, and

2. Protection of medical data on the network.

The first goal is required so that confidential data is not sold or transferred to malicious crackers. The second goal is required for protecting data from wire tapping and so forth.

2.2.1 Authentication Infrastructure with Private Certificate Authority

The GSI that the Globus provides as a security service has been developed based on Public Key Infrastructure (PKI) technology and the mutual authentication has been realized through the X.509v3-based certificates which Globus CA (Certificate Authority) issues [12, 13]. The certificate contains the information on the issuer, the subject, the subject's public key encrypted with the private key of the issuer's and so forth. By processing the information with public key cryptographic technique, we can mutually authenticate our partner. Moreover, the best feature of GSI is a single sign-on function, namely, a function that allows us to utilize a variety of computational resources geographically distributed on the Internet just by identifying ourselves with our own certificates. This feature meets our system's needs in that the system should hide the existence of the network from doctors.

In this research project, in order to facilitate the sharing of data and softwares, the author has exploited the GSI as a building block of the medical infrastructure. Furthermore, in order to make the system more practical for the current medical

situation, the author has established our project's CA and made the certificates which the project's CA issues valid in the GSI. In general, medical organizations such as hospitals, clinics, and other medical institutes form a group to communicate with each other for a higher quality of medical service as well as for general business considerations. Each of such groups has established its own security policy regarding the management of data and authentication. From this viewpoint, the establishment and management of such a private CA is suitable for practical medical situations.

The project's CA has been established with SSLeay-0.9.0b. The SSLeay provides the libraries with a function of encryption necessary for Secure Socket Layer (SSL) and Transport Layer Security (TLS) and also facilitates the establishment and operation of the private CA [18]. Until today, the certificates which our project's CA issued have been deployed to twenty-nine computers. Twelve computers of these computers are located in the Nara Institute of Science and Technology (NAIST), and the other seventeen at the Cybermedia Center, Osaka University (Fig. 2.1).

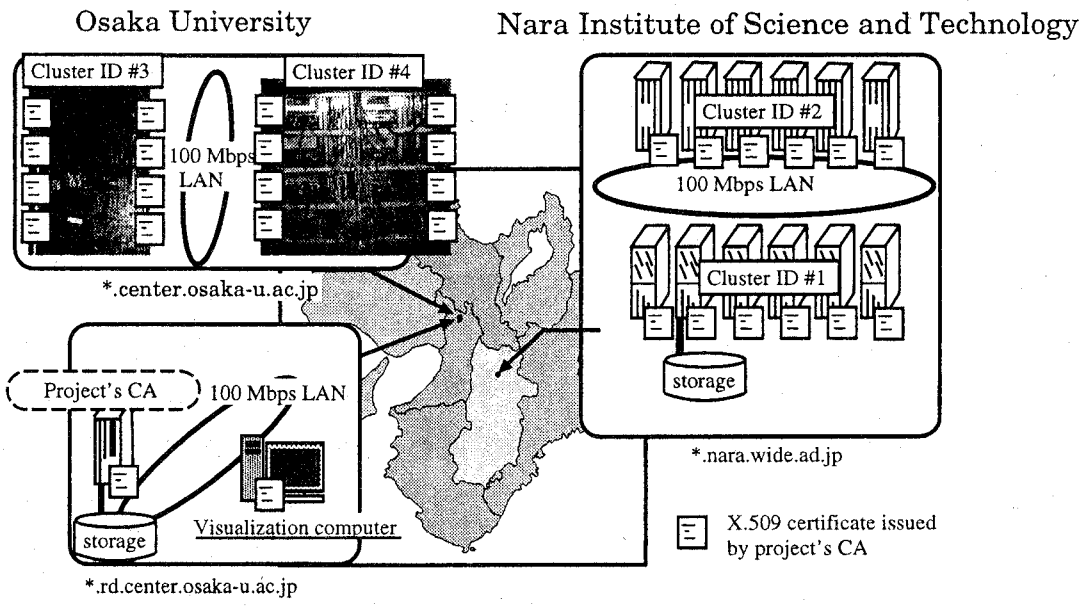


Figure 2.1: The testbed for our research project

2.2.2 IPv6/IPSec-enabled Globus

2.2.2.1 The Design Strategy

To guarantee the confidentiality of medical data on the Internet, the author has adopted the most recent to date level of the Internet protocol, that is to say, IP version 6 (IPv6). As of writing this dissertation, the IPv6 is included as part of IP support in many products including major computer operating systems. IPv6 has also been called "IPng" (IP Next Generation). IPv6 was designed as an evolutionary set of improvements to the current IP Version 4.

The following two features of the IPv6 protocol motivated its adoptionin this research:

1. The number of IP addresses available is virtually infinite, and

2. The use of Internet Security (IPSec) Protocol is considered in the IPv6 protocol.

The most prominent feature in IPv6 over IPv4 is that IP addresses are lengthened from 32 bits to 128 bits. At present, the number of IP addresses available under the current major internet protocol, or IPv4 has already exceeded the number of the global population. In the near future, wireless and mobile computing will have a role of more importance than now. Under such a situation, it is easy to predict that the shortage of IP addresses available will become a serious problem.

Network Address Translation (NAT) technology was developed in hope that it could solve the shortage of IP addresses. NAT technology provides a method of translating an IPaddress outside the network to multiple private IP addresses inside the network. This NAT technology has been used in many enterprises, universities, and scientific institutions.

However, NAT technology brings us a serious security problem when two correspondents want to mutually authenticate with each other on the Internet. For example, assume the network in Fig. 2.2. In the network, *host B* mediates the communication *host A* and *host C*, *host D*, *host E*. Then, *host B* translates the IP address outside the private network of *host B* to the corresponding IP address inside the private network. In such a network environment, *host A* has no way of specifying which *host A* is now communicating. This means that NAT technology makes the mutual authentication between two correspondents quite difficult. Without mutual

authentication, doctors could send confidential medical data to the wrong correspondent. NAT technology is, therefore, not suitable for building medical data analysis system on a public network, or the Internet.

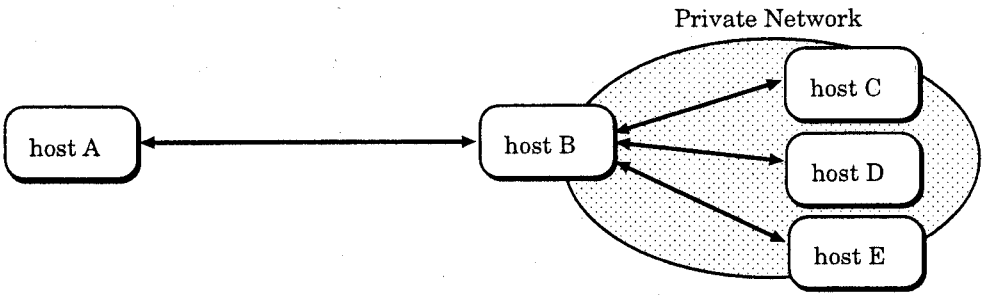


Figure 2.2: Network address translation: Host B mediates the communication between the external computer and all internal computers. This mechanism makes the mutual authentication between two computers more difficult.

On the other hand, the IPv6 protocol provides us with the solution for not only the shortage problem of IP addresses but also for the protection of confidential data on the Internet including mutual authentication. For the protection of confidential data on the Internet, the IPv6 protocol offers the IPSec Protocol. The IPSec [19] is a developing standard for security at the network or packet processing layer of network communication and enables users to encrypt their confidential data like medical data without being aware of special programming techniques. In other words, the IPSec protocol allows users to protect confidential data on the Internet on the third layer of the Open Systems Interconnection (OSI) reference model. This feature helps application developers to write codes.

Alternatively, SSL (Secure Socket Layer), TLS (Transport Layer Security) and SSH (Secure Shell) can be also used for protecting confidential data. However, these cryptographic technologies affect the application. In short, users need to modify and rewrite their existent codes in order to use these technologies, which makes it difficult to reuse existent codes. For this reason, these technologies have not been adopted in this research.

2.2.2.2 Implementation

In order to utilize the functionality of IPSec, FreeBSD operating system has been adopted as a platform for development in this research. Currently, a diversity of

Unix-based operating system exists. Examples include Solaris, Linux, NetBSD and so on. However, for the initial stage of development, BSD-based operating systems like FreeBSD, OpenBSD and NetBSD have the most developed architecture for implementation of the IPv6 protocol [20]. Before describing the implementation of the IPv6/IPSec-enabled Globus, the author explains in detail what elements the IPSec protocol is composed of and how the IPSec works on FreeBSD.

The IPSec protocol guarantees the confidentiality and integrity of the data transferred over a public network and further enables users to mutually authenticate with each other. Also, non-repudiation, essential for E-commerce, can be achieved. However, the IPSec protocol does not provide the functionalities for realizing a secure line on the Internet, but only the framework for realizing a secure line. In other words, the algorithm for authentication and encryption is not defined in the IPSec protocol suite. Thus, network and system administrators can freely select the algorithm for authentication and encryption on a connection basis.

A secure line is realized based on the following two databases; the Security Policy Database (SPD) and the Security Association Database (SAD). The SPD controls whether to apply to an outgoing or incoming packet or not based on the security policy contained in itself. On the other hand, the SAD controls what kind of transfer mode (such as Authentication Header (AH) and Encapsulating Security Payload (ESP)) should be applied to the packet, which the SPD is determined to apply to the IPSec, what kind of algorithm for cryptography and authentication should be used, and so forth. In other words, the SAD holds the information necessary for realizing a secure line on the Internet. Figure 2.3 diagrams how the IPSec is applied to an incoming or outgoing packet. After the kernel of the operating system detects an incoming or outgoing packet, the kernel decides whether the IPSec should be applied or not by looking up the SPD. If the kernel makes the decision to apply the IPSec to the packet, then the kernel investigates the packet header to obtain a Security Parameter Index (SPI) marked in the packet header. Next, the kernel applies the IPSec to the packet based on Security Association (SA) correspondent to the SPI in the SAD. The SA is the information necessary for realizing a secure line on the Internet.

In the IPSec protocol, the sharing of this SA is essential between the packet sender and the receiver. Furthermore, the sharing of the SA needs to be performed securely over the Internet. For this purpose, mutual authentication between the sender and the receiver is required. In the most cases, currently, the sharing of the

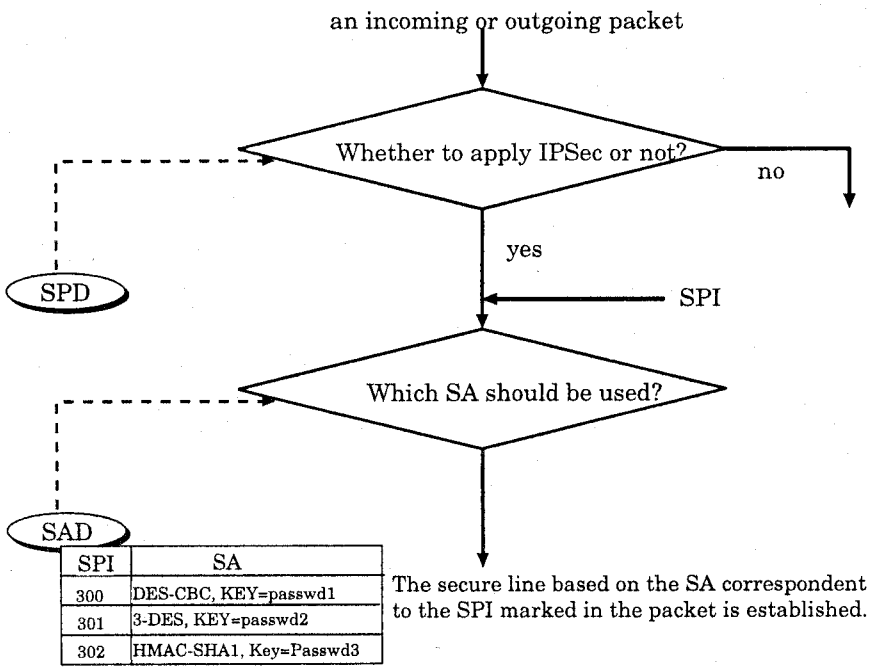


Figure 2.3: IPSec protocol in action: A secure line based on the SA correspondent to the SPI marked in the packet header is realized.

SA is manually performed by network and system administrators. However, the way of manually managing the SA becomes a problem when end-users want to realize secure lines among multiple entities, or network and hosts. The more the number of nodes increase, the more the management costs exponentially increases. If endusers want to realize secure lines over the network composed of n nodes, developers needs to exchange $\frac{n(n-1)}{2}$ SAs on the network in a secure way.

Internet Key Exchange (IKE) protocol [21] is a promising technology that provides a method of exchanging the SA of IPSec protocol in a secure and dynamic way. Also, the IKE protocol performs the mutual authentication necessary for exchanging the SA instead of the IPSec protocol. However, a problem exists in that the exchange of the SA is performed based on the assumption that mutual authentication is already performed before the exchange. In short, mutual authentication problems still remain even if the IKE is used with the IPSec.

In order to address this mutual authentication problem, I have integrated the mutual authentication of the IKE protocol to the GSI as a core authentication infrastructure, without doubling the authentication mechanism on the Grid environment. The GSI provides an authentication infrastructure based on Public Key Infrastructure (PKI) technology, and allows users to identify computing resources as well as

users with X.509v3-based certificates. As described in section 2.2.1, in this research testbed, the certificates the private project CA issues have been used instead of the certificates which Globus CA issues.

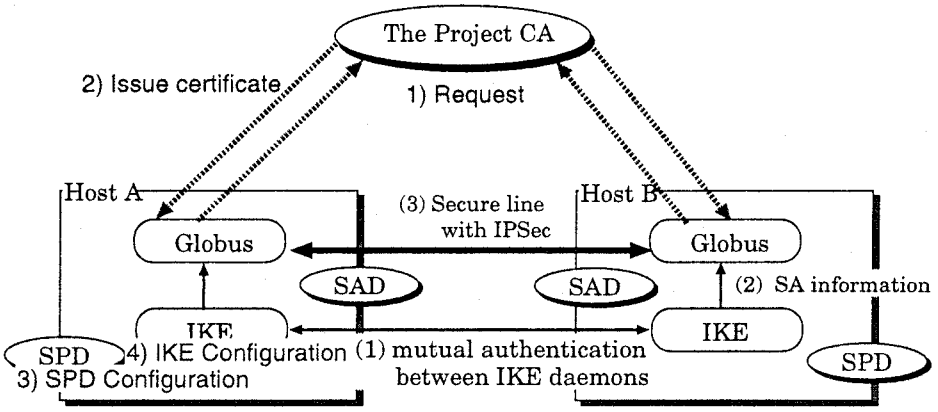


Figure 2.4: IPSec-enabled Globus

Figure 2.4 illustrates how an IPSec-enabled Globus is configured and works. The following configurations need to be performed once by the network and system administrators.

1. System administrators ask the administrator of the project's CA to issue the certificate for the nodes between which the secure line needs to be established.
2. The system administrators deploy the certificate the project's CA issued into the appropriate directory.
3. The system administrators configure the SPDs after discussion about IPSec policy with each other.
4. The system administrators configure the IKE daemons so that the IKE daemon can mutually authenticate the other daemon with the certificates issued by the private projects CA.

Based on this configuration, a secure line is automatically established as follows. The following procedures are performed when a Globus job starts.

1. Both IKE daemons mutually authenticate with the project's CA X.509 certificates.
2. Both IKE daemons exchange the SA necessary for realizing a secure line with the IPSec.

3. Each node obtains the SA from the correspondent IKE daemon.

4. Based on the obtained SAs, the secure line with the IPSec is established.

Next, the author discusses the IPv6-enabled Globus. As described in the previous section 2.1, the Globus is composed of many services essential for Grid computing. These services tightly interact with each other to produce a variety of grid services. Therefore, the transition of IPv4 Globus to IPv6-enabled Globus, requires the transition not to affect the mechanism regarding Globus interaction.

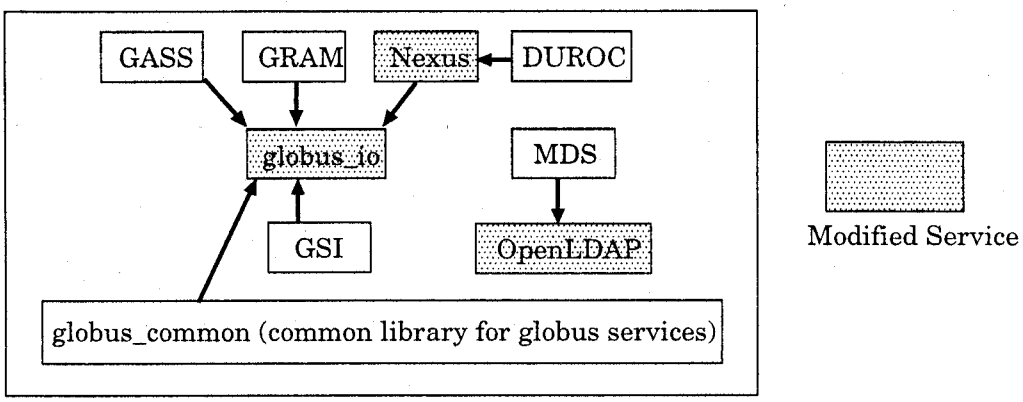


Figure 2.5: Modified services in IPv6-enabled Globus

Figure 2.5 shows the interaction mechanism in terms of communication. In the Globus grid toolit, most communication functions are aggregated into two services; Globus I/O and Nexus. A part of the information service, or MDS uses OpenLDAP software to manage the information on the Grid environment. Thus, in this research, by changing these services to IPv6-enabled services, the whole Globus has been modified to an IPv6-enabled Globus without affecting the interaction mechanisms in the Globus services.

2.3 Implementation of Data Access Structure with Location-transparency

In this research, a strategy that provides doctors with a single disk image on the Internet has been adopted. This strategy allows doctors to access medical data files located on a remote computer without having to be aware of the location of the data access protocol and other information concerned with accessing files.

2.3.1 Self-Certifying File System

In order to provide doctors with a single disk image, Self-Certifying File system has been introduced into the medical infrastructure. The SFS shares files between computers using cryptographically protected communication, and allows doctors and researchers to access files from anywhere in the world and share them with anyone, anywhere [22]. The namespace of the SFS is global and therefore the SFS allows users to access arbitrary files in a uniformed format */sfs/hostname/* [23]. These features also allow doctors to access a medical database from a patient's house through the Internet with ease.

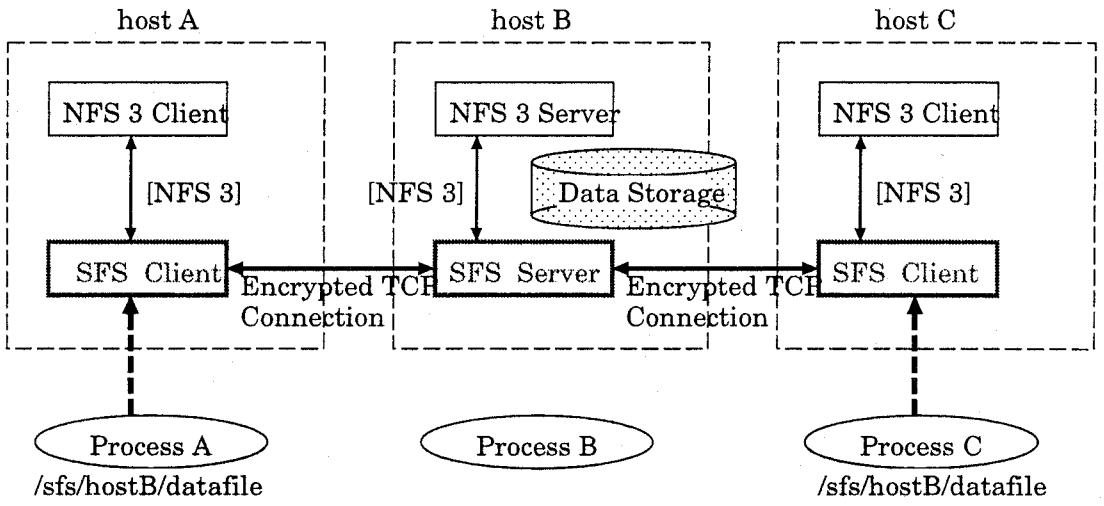


Figure 2.6: Overview of the Self-Certifying File System

Figure 2.6 illustrates an overview of the SFS. The SFS server and client mediate communication between the NFS server and the NFS client and then provide an encrypted TCP connection on a user-level basis. More specifically, the SFS client behaves as a NFS server for the NFS client, while the SFS server behaves as a NFS client for the NFS server. Furthermore, the SFS server and the SFS client provide an authentication mechanism similar to the public key cryptography method which the SSH protocol has utilized for the users and computers [24].

Medical data storage with the SFS has been built over the network at NAIST (The Nara Institute of Science and Technology) and Osaka University for preliminary research (Fig. 2.1). The medical data storage is composed of two SFS servers, each of which is located at NAIST and Osaka University. In this system, the file systems on these servers are mounted statically from the visualization computer which doctors

use in their laboratories and houses. Thus, the system has the capability of providing medical data the doctors want to access without the doctors' being aware of the existence of the network and the access protocol.

2.3.2 Problem in Introducing the Self-Certifying File System

In this research, the author assume that the SFS is used only when doctors search a medical data file of their interest on the Grid environment. Therefore, for the analysis of medical data, parallel computation raises a serious problem.

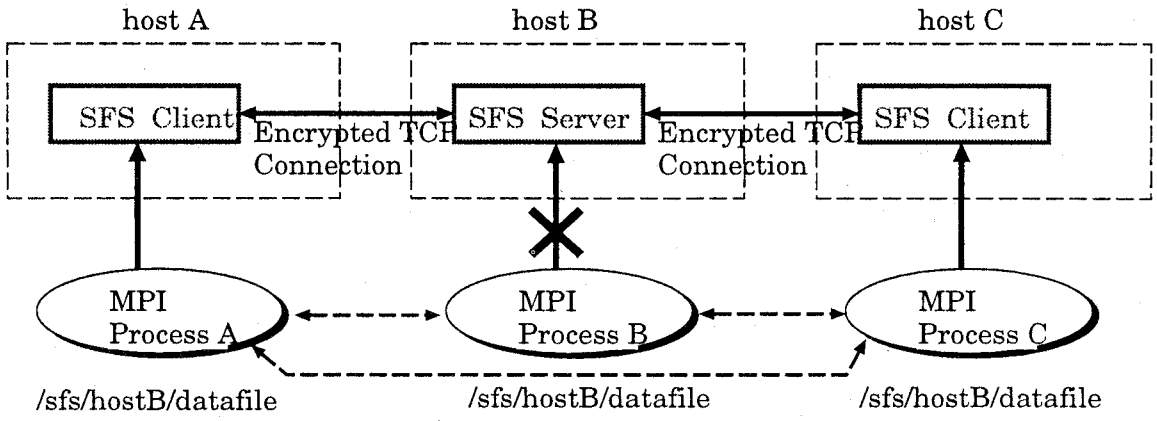


Figure 2.7: A problem during parallel computing on the Grid

Consider the situation shown in Fig. 2.7. For the computation of the analysis of medical data, assume *host A*, *host B*, and *host C* are used. In general, the developer writes the application code, assuming that all processes which run in parallel can access data files with the same file path like Uniform Resource Locator (URL). However, in the environment shown in Fig. 2.7, the process B on *host B* cannot access the data file which doctors specified in the same file path as *host A* and *host C* because *host B* has no SFS client functionality. This situation forces developers to write their application codes without being aware of the difference of file paths among computers, which results in a long development period.

2.3.3 Solution

In order to solve the problem that a process on the computer with no SFS client functionality cannot access a file during the computation of analysis, described in

section 2.3.2, the author has exploited the file path converter that converts the SFS file path which a doctor specifies to the actual file path.

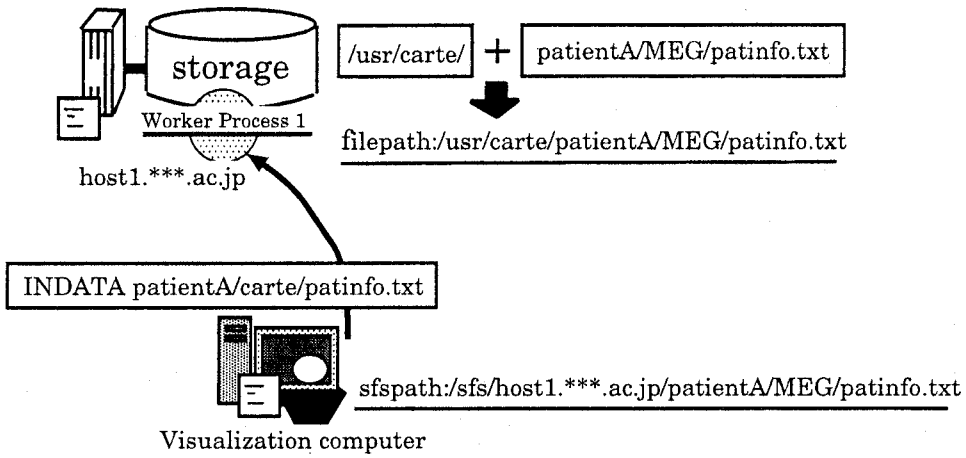


Figure 2.8: Filepath conversion

Figure 2.8 illustrates how the file path converter works. The idea of the file path converter is simple. The file path converter is run only on computers which have no SFS client functionality. In Fig. 2.8, it is assumed that *visualization computer* has SFS client functionality, while *host A* does not. Also, the medical data file is assumed to be located on the data storage connected to *host A*. The doctor in front of the visualization computer thinks that the medical data is located in the local disk because the SFS provides the doctor with a single disk image. The specified SFS file path by the doctor is converted to the actual file path by the file path converter in *host A*.

2.4 Concluding Remarks

In this chapter, the author described the infrastructure for medical data analyses. The infrastructure offers doctors an encrypted secure line. This feature is essential for exchanging medical data sensitive to security problems. In this research, this feature has been realized with the IPSec protocol.

Next, in order to solve the mutual authentication problem between the sender and receiver of medical data on the Internet, the authentication infrastructure with the GSI provided by Globus has been exploited. Mutual authentication is necessary to realize a secure line. Furthermore, in order to enhance the security level regarding

authentication in the infrastructure for medical data analyses, the author has established a private certificate authority for this research project and merged the CA into the GSI in order to issue certificates only to the people involved in this project. The establishment and management of such a private CA is suitable and necessary for the medical infrastructure. This fact can be explained from practical medical situations where medical organizations such as hospitals and medical schools form a group so as to communicate with each other for a higher quality of medical services and cares.

Furthermore, the latest Internet protocol, or IPv6 protocol, has been introduced into the infrastructure together with the IPSec Protocol. In other words, a IPv6/IPSec-enabled Globus has been developed. This feature of the IPv6 protocol will solve the problem of a shortage of IP addresses, which will become a serious problem in the near future global computing era. Also, the infrastructure built with the IPv6/IPSec-enabled Globus will make it easy for the mutual authentication essential for medical data analyses on the Internet is performed.

The data access structure with location-transparency has been developed with the existent network file system, the Self-Certifying File System. The data access structure can provide doctors with a single disk image. The feature of offering a single disk image allows doctors to find the medical data file on the Grid environment without doctors having to be aware of something special regarding data access such as access protocols.

In sum, the author have arranged and developed a medical infrastructure for medical data analyses. The designing and implementing strategy has been performed in a complementary way to the Globus services. The author believes that medical infrastructure for medical data analyses with the Globus grid toolkit will be useful and promising in actual medical data analyses. In the following chapters, the author aims to verify the usefulness of this infrastructure through the building of an actual medical data analysis system.

Chapter 3

MEG Data Analysis

3.1 Introduction

In this research, the MEG data analysis problem has been adopted as an actual scientific problem to which the Grid is applied. First, this chapter details not only what the MEG is, but also what MEG data analysis is. Next, specific MEG data analysis problems, which prevent doctors from efficiently diagnosing and analyzing brain functions, are considered. After that, the author describes *Wavelet Cross-correlation Analysis*, the analysis method used in this research.

Next, through an investigation of how wavelet cross-correlation analysis is used, efficient and effective parallel processing methods for wavelet cross-correlation analysis are discussed. In addition, a visualization software for wavelet cross-correlation analysis has been developed in this chapter. The author shows that the visualization software was designed and implemented so that doctors and researchers can perform wavelet cross-correlation analysis efficiently.

3.2 Magnetoencephalography (MEG)

A diversity of technologies that enable visualization of brain activities can be classified into two classes. One class is morphological examination and the other is functional examination. Functional magnetic resonance instrument (fMRI) and positron emission tomography (PET) are categorized as the former class. These technologies visualize neural activation, by focusing on the cerebral blood flow. On the other hand, electroencephalography (EEG) and magnetoencephalography (MEG) [25] visualizes neural activity, by focusing on the electric potential and the magnetic field

change generated from brain activity.

The former class is, to some degree, more available than the latter one for the purpose of localizing brain activities, since it gives doctors and researchers an intuitive macroscopic view of brain activities. However, two major problems of the modalities exist in this class; low temporal resolution and invasivessness in the measurement.

For revealing brain functions, the pursuit of dynamic neural activity is essential. In other words, we need to investigate from and to where brain signals propagate. In contrast, the temporal resolution in fMRI and PET is fundamentally limited by the time constants of the hemodynamic effects that result from neural activation and which are exploited to produce these activation images [26]. This limitation means that fMRI and PET are not suitable for the source localization of brain signals. Also, the invasiveness of the measurement makes it almost impossible for brain scientists to use fMRI and PET for scientific purposes.

On the other hand, EEG and MEG realize non-invasive measurement. MEG especially, achieved measurement accuracy. Until recently, there was, in general, a trade-off problem between non-invasiveness and measurement accuracy. For example, Electrocorticography (ECoG) achieves measurement accuracy by directly measuring the electrical potential on the cerebral cortex. Therefore, measurement with ECoG is used just for patients whom surgery must be performed. Nonetheless, MEG have achieved both of non-invasiveness and measurement accuracy due to the development of a super-conducting quantum interference device (SQUID) magnetometer. In contrast, MEG allows researchers to detect subtle magnetic field change resulting from brain activities on the order of 10^{-8} Gauss (G) or 10^{-12} (T), compared with the terrestrial magnetic field of about 0.5 G. This measurement is performed from multiple measurement points around the head. Measurement using MEG is non-invasive for a subject and patient. Thus, MEG is a very promising modality in the field of brain research from both the clinic and scientific points of view.

3.3 Problems with MEG Data Analysis

Today, the large amount of MEG data is a major problem in MEG data analysis, although MEG has many positive features. The recent development of measurement technologies increases the number of MEG sensors. Currently, the MEG is equipped

with more than 200 sensors [27]. This situation has improved spatial resolution in the measurement, but it also results in the increase of MEG data to be analyzed.

The MEG, in general, samples brain signals from multiple measurement points at an interval from 1 to 4 milliseconds. The sampling value is recorded as **float** type in computer language. This means that MEG data increases by (the number of MEG sensors) $\times$ (the byte size of float type) bytes per a sampling. If a 64-sensor MEG and a 4-byte floating type is used, the data size reaches 256 bytes.

MEG measurement for clinical purposes is usually performed for more than an hour. If the MEG measurement using a 64-sensor MEG is performed for one hour under the condition that a sampling interval is 1 millisecond, the amount of MEG data reaches approximately 0.9 Giga bytes. This amount of data is quite large. This increase in MEG data that must be analyzed results in the increase of calculations necessary for the MEG data analysis, which prevents efficient diagnosis and analysis.

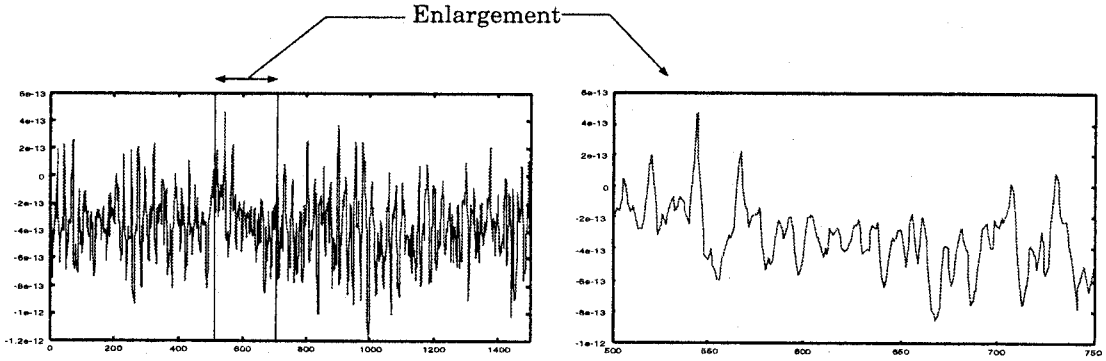


Figure 3.1: MEG data

The complexity of MEG data analysis makes efficient diagnosis and analysis difficult. In the analysis of brain function, the investigation of frequency components in the brain data is of great importance because each brain signal with a certain frequency component is supposed to convey a certain function. However, even specialists find it difficult to understand the frequency components precisely or to detect brain functional disorders, just by observing the MEG data like the type shown in Fig. 3.1. Therefore, a variety of signal processing techniques have been proposed and developed for the analysis of brain function. Most of these techniques are very effective, but can be classified into computationally-intensive problems.

Furthermore, the analysis of brain functions is a kind of inverse problem. An inverse problem is a problem where the cause is investigated from the observed ef-

fect. This type of problem has many possible solutions with which one can explain the observed effect. Thus, in order to find the real solution from all these possible solutions, doctors must suspect all possibilities. Obviously, suspecting all possibilities takes much analysis time.

Currently, to efficiently solve the inverse problem, scientists have introduced a variety of brain activity models. For example, some scientists use the brain activity models assuming the number of brain signal source is a specific number to reduce the long computational time for analysis. Other scientists use the brain activity models assuming the abnormal region in brain. However, even exploring the analysis space reduced by the assumption needs a long computational time because the analysis result is very sensitive to the parameters of frequency analysis.

In MEG data analysis, all the possible brain activity models must be ideally investigated for the purpose of revealing brain functions. Nonetheless, at present, MEG data analysis has been performed with only one model. This medical situation prevents doctors from detecting a symptom of brain disease.

The large amount of computational power is essential for the advancements in MEG data analysis. In this chapter, the author first discusses the MEG data analysis method adopted in this research and then considers issues to supporting the use of MEG data analysis.

3.4 Wavelet Cross-correlation Analysis

In this research, wavelet cross-correlation analysis [28, 29] has been adopted as a signal processing technique. This analysis has the capability of investigating frequency components contained in MEG data without losing original time information. Traditional Fourier-based analyses have been unable to achieve this feature. This prominent feature is suitable for the analysis of non-stationary data like MEG data.

Wavelet cross-correlation analysis is composed of two types of analyses. One is wavelet analysis, and the other is cross-correlation analysis. Wavelet analysis investigates frequency components contained in MEG data. Cross-correlation analysis quantifies the result of wavelet analysis.

Figure 3.2 illustrates the overview of wavelet cross-correlation analysis. First, wavelet analysis is performed for MEG data acquired from a single sensor. The

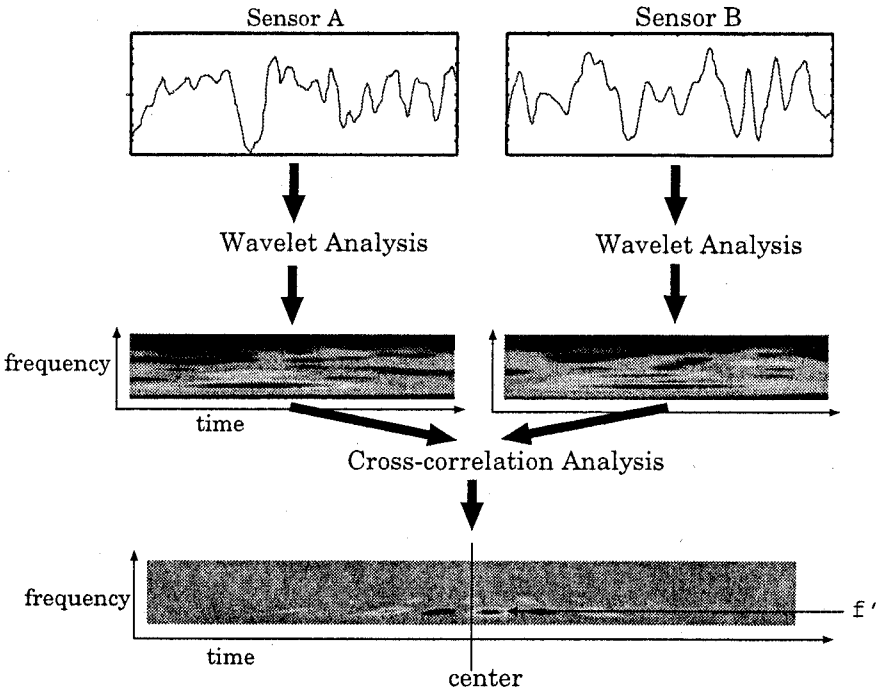


Figure 3.2: Overview of wavelet cross-correlation analysis

wavelet analysis is performed based on the following equations (3.1) – (3.3).

$$Wf(a,b) = \int_{-\infty}^{\infty} \overline{g_{a,b}(t)} f(t) dt \tag{3.1}$$

$$g_{a,b}(t) = \frac{1}{\sqrt{a}} g(\frac{t-b}{a}) \tag{3.2}$$

$$g(t) = e^{-\frac{t^2}{2}}(e^{j\Omega t} - e^{-\frac{\Omega^2}{2}}), \Omega = 2\pi \tag{3.3}$$

The function $f(t)$ represents MEG data acquired from a single sensor. The function $g(t)$ is a Gaussian basis which is a kind of mother wavelet (analyzing wavelet). The Gaussian basis has been adopted in this research because it is effective for non-stationary data analysis such as MEG data [30]. The parameter "a" controls the frequency context in the mother wavelet, while the parameter "b" shifts the mother wavelet along the time axis. Figure 3.3 shows a variety of Gaussian bases dilated by the parameter "a" in the equation (3.3). The upper images in Fig. 3.2 are the visualized results of wavelet analysis. These images allow doctors to intuitively understand a frequency distribution map of the corresponding MEG data.

Next, cross-correlation analysis is performed for each pair of the results of wavelet analysis. This analysis is performed based on the equation (3.4).

$$WC_{1,2}(a,\tau) = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} \overline{Wf_1(b,a)} Wf_2(b+\tau, a) db \tag{3.4}$$

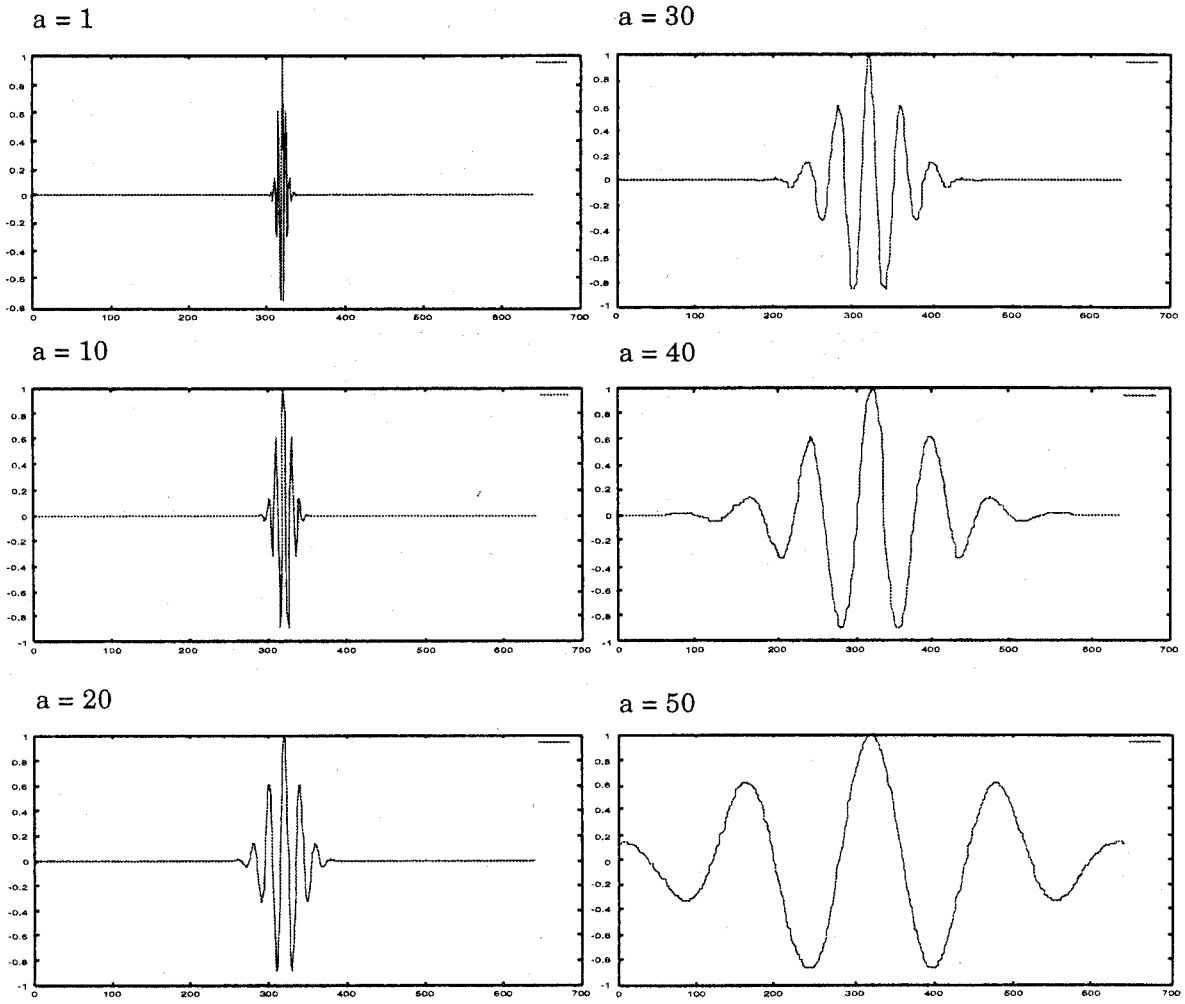


Figure 3.3: Gaussian basis with a variety of frequency response

Wf_1 and Wf_2 are the results of the wavelet analysis. This analysis provides information on the correlation of frequency distribution between the corresponding two sensors. The bottom image in Fig. 3.2 is the visualized result of cross-correlation analysis. This image indicates that a brain signal with a frequency f' component is detected in sensor B earlier than in sensor A. This feature of the analysis allows doctors and scientists to localize the source of a brain signal.

However, the wavelet cross-correlation analysis space becomes large because of the existence of many program parameters and the large amount of data. Therefore, wavelet cross-correlation analysis is used as follows. First, doctors select the temporal region of the MEG data containing a brain signal of interest to them. Next, using wavelet analysis, doctors investigate whether or not MEG data contains a frequency component of interest. Next, using cross-correlation analysis, doctors

investigate the correlations in frequency components contained in MEG data among all pairs of sensors. Finally, they investigate the time-lag of the signal's emergence and localize the source of the signal.

The computational workload for wavelet cross-correlation analysis increases intensively with the increase of MEG sensors. The use of a 64-sensor MEG is assumed in this research. In this case, then, cross-correlation analysis should be ideally performed 2016 times. Recently, the number of MEG sensors has increased. The MEG with more than 200 sensors exists. In the case of a 200-sensor MEG, a cross-correlation analysis should be ideally performed 19900 times. In this research, the computational workload for this analysis is distributed on the Grid for the reduction of the analysis time.

3.5 Issues in MEG Data Analysis with Wavelet Cross-correlation Analysis

Diagnosis of brain disorders is an extremely difficult and time-consuming task. In general, it has been performed as the following three stages:

1. Brain data measurement with various brain imaging technology,
2. Data analysis, and
3. Evaluation of brain function based on the analysis results.

These three stages require specialized knowledge, or specialists or specialized resources such as high-performance computers and MEG. Hence, these three stages are often locally gathered, or distributed, which results in inefficient diagnosis.

In order to realize efficient diagnosis, the integration of these three stages is indispensable. In other words, diagnosis of brain disorders requires seamless processing from data collection to evaluation of brain function. At present, nevertheless, brain data has been recorded on optical disks and passed from person to person by hand. Sharing data by hand not only wastes time but may also cause loss of data consistency or loss of data itself.

Furthermore, the data analysis in stage 2 is also another factor that prevents doctors from diagnosing brain disorders. In general, after brain data are measured at multiple points, data analysis is performed by comparing and investigating the correlation among brain data collected with brain imaging technologies. For the

comparison and investigation between brain data, frequency analysis such as the Fourier transform is often used. This analysis requires far more time than the other two stages. Hence, for the seamless processing of the three stages, the computational time for the analysis must be reduced dramatically.

In addition, the evaluation of brain function based on the analysis results in stage 3 has an another factor leading to inefficient analysis and diagnosis. Each result of the signal processing methods performed in stage 2 appears as a sequence of numerical number. Trained specialists of dianogsis and signal processing could not understand the meaning of the analysis results at a glance without a visualization software for the complex sequence of numerical number. Sometimes, the results appears in a unreadable form for human beings. In reality, the lack of visualization software is a big problem in rather brain science than the whole brain science. Actually, wavelet cross-correlation analysis adopted in this research has no effective software for providing an intuitive understanding of the meaning of the results, because wavelet cross-correlation analysis method is a relatively new signal processing method. The lack of visualization software for wavelet cross-correlation analysis leads to inefficient diagnostic and analytic situation. Therefore, the development of a visualization software for wavelet cross-correlation analysis is indispensable for improving such inefficient diagnostic and analytic situation.

To summarize, the following three goals need to be achieved in order for doctors to proceed efficiently their analyses:

1. Seamless integration of the three stages essential for the analyses of brain functions,

2. Dramatic reduction of the computational time for wavelet cross-correlation analysis, and

3. The development of a visualization software that provides doctors and scientists with an intuitive understanding of analysis results.

The first two goals are described in detail in the following chapter. Again, the Grid, as described in chapter 1, is a promising technology that integrate an infinite number of computataional resources geographically distributed on the Internet. The Grid's feature of integrating computational resources on the Internet has the potential to achieve the first two goals. The following chapter will show that the Grid can achieve the first two goals and verify effectiveness and usefulness of the Grid. In the

following section, 3.6, the author explores efficient and effective parallel computing methods for wavelet cross-correlation analysis.

Next, for the third goal, the author describes the explore of visualization software for wavelet cross-correlation analysis in the following section. To realize such a visualization software, the visualization requirements for wavelet cross-correlation analysis are first considered. Subsequently, a visualization software based on a consideration is shown.

3.6 Parallel Computing for Wavelet Cross-correlation Analysis

In order to consider an efficient parallel processing method for wavelet cross-correlation analysis, investigating how the wavelet cross-correlation analysis is used and the most efficient way to use it is necessary. After understanding the computing requirements regarding wavelet cross-correlation analysis, the author considers effective parallel processing for wavelet cross-correlation analysis.

In wavelet cross-correlation analysis, the wavelet analysis needs to be repeatedly performed. In general, brain signals that are of interest to doctors, such as epileptic wave signals, last for 4 to 10 seconds and such signals appear many times. Furthermore, the emergence of such signals is non-stationary. Doctors need to find the data regions which contain such signals in a large MEG data space. To do this, doctors often perform wavelet analysis many times, finding arbitrary temporal regions of MEG data. This work is time and labor -consuming. Thus, quickly finding the pertinent brain signals is an important key to success in diagnosing various brain diseases.

Doctors and scientists use this wavelet analysis to find brain signals by focusing on two different regions in MEG data space. The two regions are as follows (Fig. 3.4):

1. The region along the temporal axis, and

2. The region along the spatial axis.

The first region is used where the source of the brain signal is, to some degree, localized by a medical test performed in advance. In this case, the doctors' main concern is how the frequency components in the MEG data vary over time, since doctors want to precisely localize the source of brain signals by comparing them

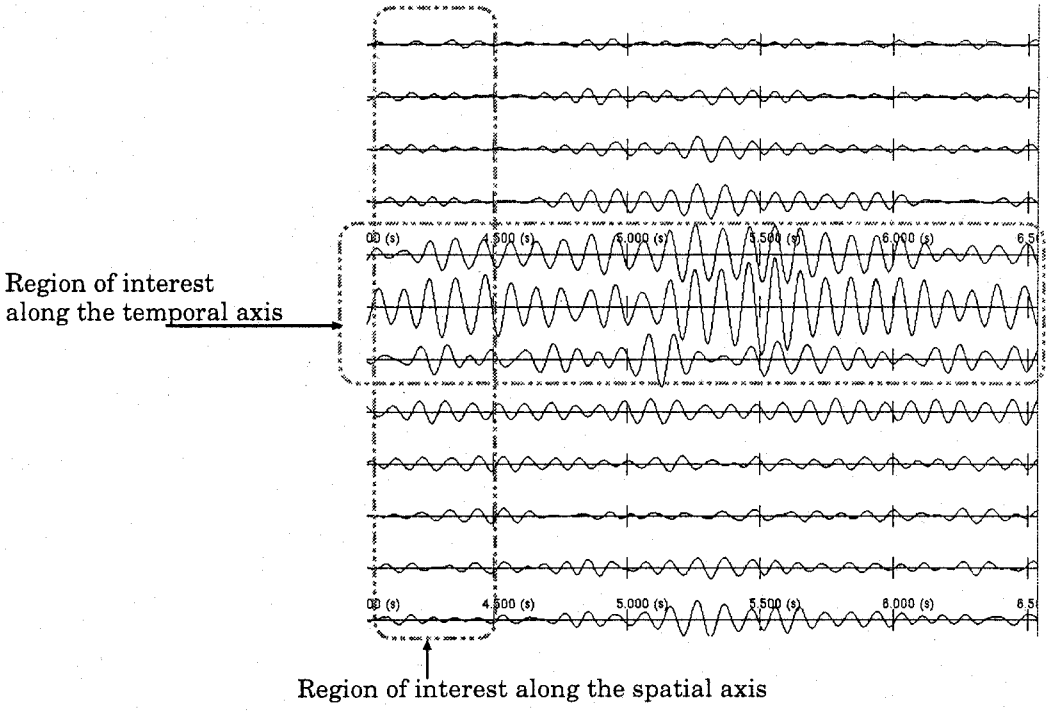


Figure 3.4: Two regions of interest to doctors in the MEG data space

among sensors. Hence, long MEG data such as 60 second data, which are acquired from only a few sensors around the source, are analyzed.

The second region is used when doctors have no information on the source of the brain signal which interests them. In this case, in order to observe which MEG data contains the frequency components of the brain signal, doctors need to investigate the MEG data from all sensors. Here, short MEG data ranging from 1 to 4 seconds acquired from all sensors are analyzed.

In this research, two types of parallel processing methods for wavelet analysis are proposed on our grid environment, based on the previously described doctors' needs. For problems focusing on the region along the temporal axis in the MEG data space, the workload of wavelet analysis itself is distributed to multiple computers. In other words, the equation (3.1) is computed in parallel.

Figure 3.5 shows the parallel processing method which focuses on the region along the temporal axis. This wavelet analysis result provides a 2-dimensional time-frequency map for a single MEG data. Each row shows the result of a convolution operation between MEG data $f(t)$ and a Gaussian basis $g_{a,b}(t)$ with a certain frequency response. The convolution operation for each row can be easily performed in parallel. This research utilizes this computational locality to distribute the workload

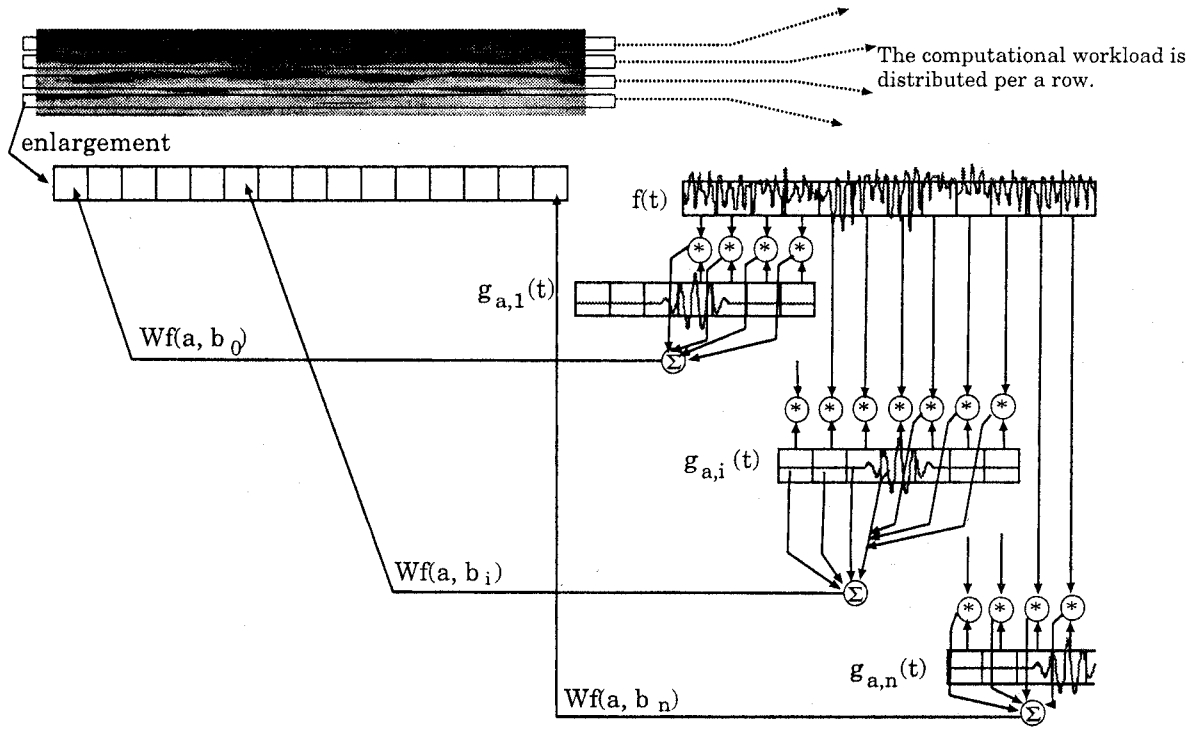


Figure 3.5: A parallel processing method which focuses on the region along the temporal axis

for a single wavelet analysis.

For the region along the spatial axis, the distribution method is simple. In this case, as MEG data acquired from all sensors are analyzed, the author distributes these multiple tasks of wavelet analysis to multiple computers. In our system, 64 data at most are simultaneously investigated with wavelet analysis.

After wavelet analysis, cross-correlation analysis is performed for each pair of the wavelet analysis results. When a 64-sensor MEG is used, the number of such pairs reaches 2016. In our system, this computational workload for 2016 pairs of MEG data at most is distributed between multiple computers.

3.7 Visualization Software

As described in section 3.4, wavelet cross-correlation analysis is composed of two different analyses. These different analyses have different requirements for visualization. First, the author considers visualization for wavelet analysis. Next, visualization for cross-correlation analysis is considered.

3.7.1 Visualization for Wavelet Analysis

Wavelet cross-correlation analysis plays a role of great importance in providing doctors and researchers with clues when they try to find a brain signal such as epileptic waves. Doctors and researchers find brain signals of their interest based on the temporal change in the frequency component contained in MEG data. To realize efficient diagnostic and analytic situations for wavelet cross-correlation analysis, therefore, visualization software for wavelet cross-correlation analysis must have the capability of visualizing the change in frequency components in MEG data over time. Furthermore, visualization software can be used in an interactive manner for doctors and researchers. In short, visualization software also needs to provide a user-friendly interface.

To help doctors and researchers find brain signals of their interest, visualization software needs to provide the functionality of enabling them to select an arbitrary temporal region of MEG data. The wavelet analysis is performed based on the equations (3.1) – (3.3) described in section 3.4. The function $f(t)$ in these equations corresponds to the MEG data acquired from a single MEG sensor. Although the time parameter t in the equation (3.1) moves from minus infinity to infinity in mathematical theory, in reality, it is impossible to move the time parameter t in such a way. Therefore, doctors and researchers need to select an arbitrary temporal region where they want to perform wavelet analysis. From the above considerations, the author has developed a raw data viewer that allows doctors and researchers to select the temporal region of their interest from MEG data space.

Figure 3.6 shows the raw data viewer. This raw data viewer is composed of the following five principal components; amplification control buttons, backword buttons, forward buttons, file menu, and plugin menu. The amplification control buttons offer the functionality of changing the amplification of MEG data. The functionality emphasizes the changes in wave form, and as a result helps doctors and researchers find brain signals of their interest. The backward and forward buttons control the temporal region shown in the window-frame. This functionality is useful for providing an overview of MEG data space. To improve the switching speed of regions of MEG data space, the author has used a frame buffer technique. This technique stores whole MEG data in the form of *off-buffer screen*, a kind of visualization expression, into main memory as much as possible and switches the regions of MEG data space quickly. This visualization technique works effectively

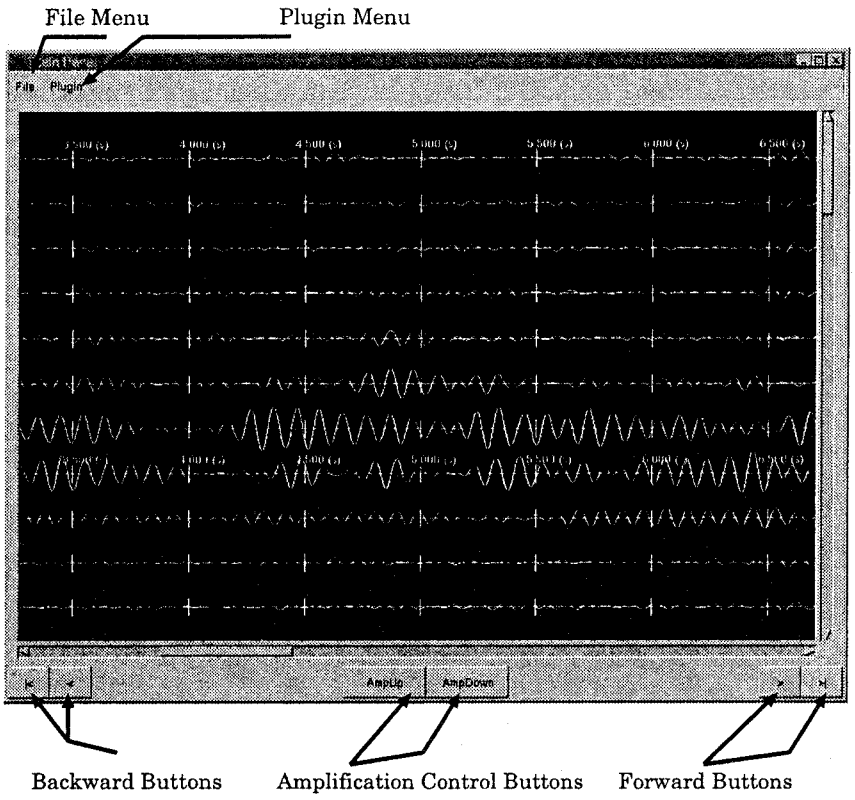


Figure 3.6: Snapshot of Raw Data Viewer

in case interactive processing is demanded, although it consumes a large amount of main memory. The file menu provides a set of functionalities regarding file operation such as opening files and saving files. Through this menu, doctors and researchers can have the software read as a medical data file located on a remote computer via the data access structure with the location transparency described in chapter 2. Finally, the plugin menu has a role of great importance in the use of the Grid environment. Doctors and researchers can utilize computational power with the Grid via this plug-in menu. More specifically, doctors and researchers can choose an analytic module from a variety of analytic modules developed on the Grid environment in a plug-in manner. In sum, this menu is designed to allow doctors and researchers to use the benefits of the Grid without having to be aware of the complex Grid technologies.

The result of this analysis provides doctors and researchers with the time-frequency map of the correspondent MEG data. The time-frequency map helps doctors to find the brain signal of interest. Providing this time-frequency map in an intuitive way is a key to success for realizing efficient diagnosis and evaluation of

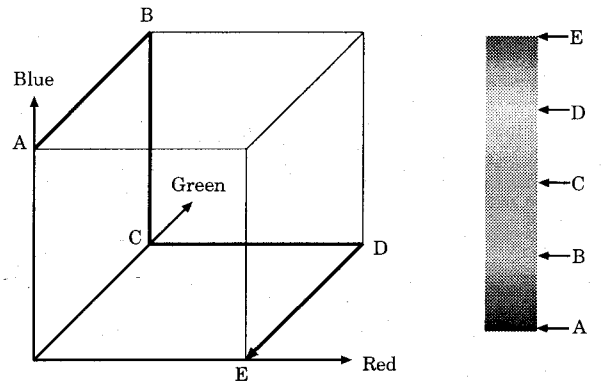


Figure 3.7: Visualization for wavelet analysis

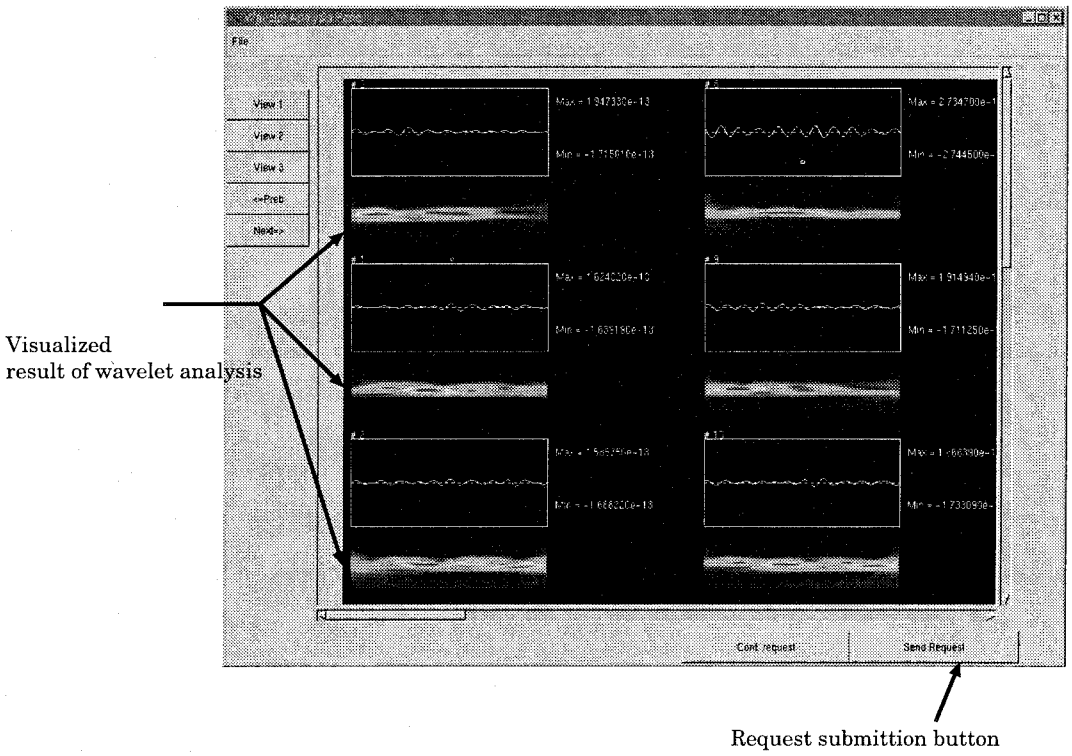


Figure 3.8: Snapshot of the wavelet viewer

brain functions. To achieve the provision of the time-frequency map, the author has developed a wavelet viewer that makes full use of RGB color space. This wavelet viewer maps a numerical value in a sequence of wavelet analysis result to a coordinate in the RGB color space. Figure 3.7 shows how the result data is visualized. The value of $Wf(a,b)$ in the equation (3.1) is mapped on the arrow in Fig. 3.7 so that the minimum value is mapped to the coordinate A and the maxmimu value is mapped to the coordinate E.

Figure 3.8 shows a snapshot of the wavelet viewer. The wavelet viewer offers multiple time-frequency maps of the temporal region in MEG data analysis, selected by doctors and researchers. With this viewer, doctors and researchers can compare analysis results of wavelet analysis sensor by sensor. This provision of multiple time-frequency maps realizes an efficient diagnotic and analytic situation.

3.7.2 Visualization for Cross-correlation Analysis

Cross-correlation analysis quantifies the result of wavelet analysis in detail. A result of cross-correlation analysis can also be visualized as a time-frequency map. The time-frequency map can provide an intuitive understanding of the result of cross-correlation analysis for MEG data acquired from a pair of MEG sensors. In cross-correlation analysis, the time-frequency map to be treated increases exponentially. In our case of a 64-sensor MEG, the number of the maps is 2016, compared to 64 in wavelet analysis. The more the MEG sensors increase, the greater the number of the time-frequency maps doctors and researchers have to investigate. In the current situation where the number of MEG sensors tends to increase, a more effective visualization technique is demanded.

So far, various approaches to visualization have been attempted [31]. Typically, most of such approaches are based on two-dimensional visualization. Two-dimensional visualization helps our understanding to a certain degree but approaches based on two-dimensional visualization have a limitation when the amount of data to be investigated is quite large. In this research, in order to give doctors and researchers more intuitive understanding, visualization with three-dimensional animation has been attempted [32]. Visualization with three-dimensional animation can represent time-variant brain activities. For the visualization, OpenGL has been used in this research. OpenGL is a graphic library that provides us with basic primitives for three-dimensional computer graphics [33].

Furthermore, interactiveness is one of the most important features for visualization of cross-correlation analysis as well as for wavelet analysis. Interactiveness in visualization enables specialists to arbitrarily operate three-dimensional computer graphics. Figure 3.9 shows a three-dimensional visualization of the analysis results. This visualization software has been developed to make maximum use of animation techniques and therefore provides doctors with an intuitive understanding of the analysis results over time. In addition, doctors and researchers can investigate this

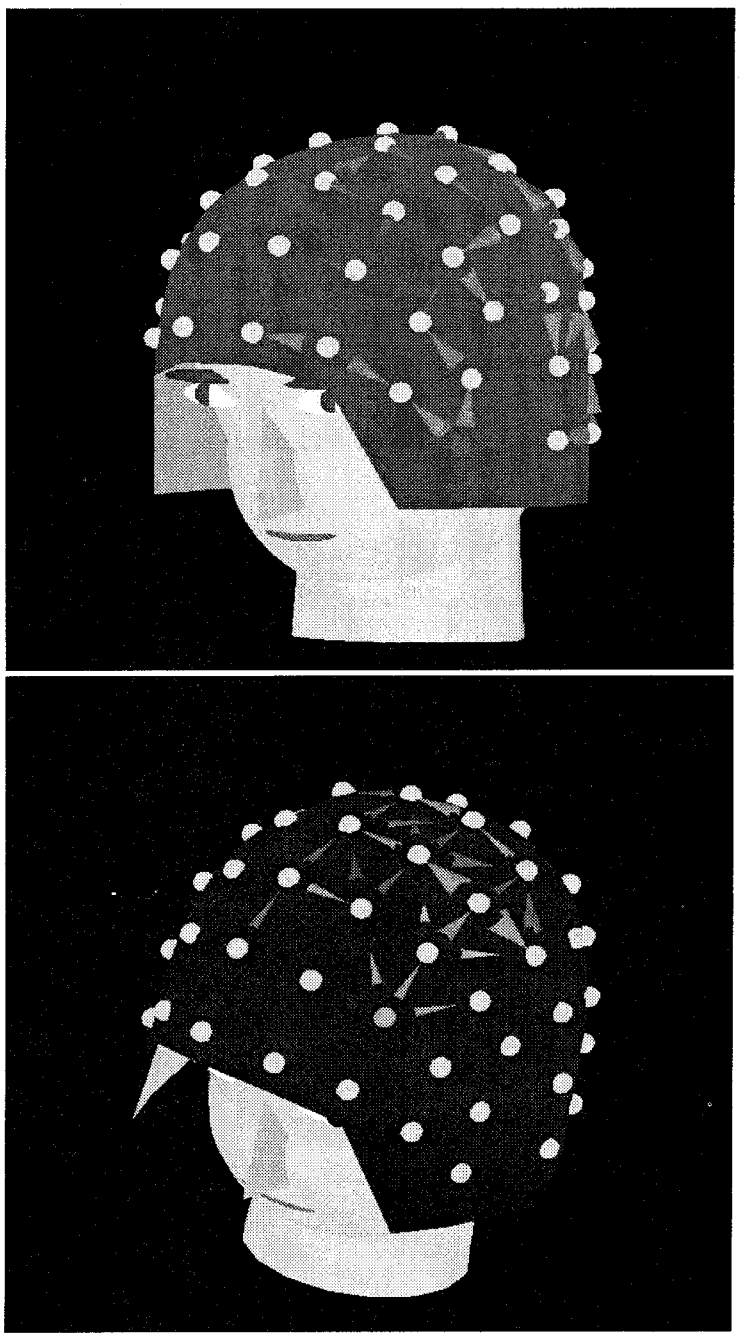


Figure 3.9: The three-dimensional visualization of the analysis results

result from an arbitrary aspect as shown in Fig. 3.9. Therefore, with this three-dimensional, doctors and researchers may investigate the source of a brain signal of interest in an interactive manner.

3.8 Concluding Remarks

In this chapter, MEG data analysis has been discussed. Through a consideration of MEG data analysis, problems in MEG data analysis were clarified. Furthermore, through a consideration of wavelet cross-correlation analysis, which has been adopted in this research, the goal of computational power required for the analysis of brain functions was described. In addition, the lack of visualization software for wavelet cross-correlation analysis was described.

To reduce the computational time for wavelet cross-correlation analysis, the author investigated how doctors and scientists perform wavelet cross-correlation analysis. Based on the investigation, the author proposed parallel processing methods for wavelet cross-correlation analysis. These parallel processing methods are used in chapter 4.

Furthermore, to realize an efficient diagnostic and analytic situation, the author has developed a visualization software for wavelet cross-correlation analysis. This visualization software is composed of the following main components: a raw data viewer, a wavelet viewer, and a three-dimensional visualization tool. Each component was designed and implemented to offer an intuitive understanding of analysis results based on a consideration of the visualization requirements. Furthermore, the visualization software allows doctors and researchers to investigate analysis results for wavelet cross-correlation analysis in an interactive and plug-in manner. More specifically, the visualization software was designed and implemented so that doctors and researchers can make full use of the Grid without being aware of complex Grid technology.

In the next chapter, the following two problems with using the Grid environment will be discussed:

1. Seamless integration of three stages essential for the analyses of brain functions, and

2. Dramatic reduction of the computational time for wavelet cross-correlation analysis.

Likewise, MEG data analysis systems that can solve the two problems outlined above are described. The systems have been built with different programming models.

Chapter 4

An Application of the Grid to MEG Data Analysis

4.1 Introduction

The Grid is a newly emerged technology being explored in the hope that it can be realized on a virtual computer on the Internet. Researchers have developed and used various kinds of programming models for Grid computing. The major programming models are the Globus and MPICH-G at the time of writing this dissertation, although there are some attempts to construct other programming models effective for Grid computing.

The Globus programming model allows researchers to use a suite of Application Programming Interfaces (APIs) provided by the Globus grid toolkit. The Globus APIs are designed to enable researchers with no knowledge of complex computer technologies hidden behind these APIs to develop their own application system such as a MEG data analysis system. The number of these APIs is quite large. Also, the APIs are difficult to use. The reason for the programming difficulties can be explained as follows. The Globus APIs provide researchers with a high flexibility for the development of a system. Clearly, researchers can write their own computer programs freely without being aware of the limitations of the Globus programming model. However, the flexibility for development simultaneously leads to certain difficulties in programming.

In contrast, MPICH-G is designed and implemented to make it easier for researchers to develop their own programs. Because MPICH-G provides reserchers with familiar APIs defined by the MPI Forum [34], researchers can shift conventional

distributed parallel computing, such as NoW computing, to new Grid computing. In fact, programs with MPI APIs which are developed for conventional distributed parallel computing are reusable and therefore researchers can use such programs by simply re-compiling the programs for Grid computing. The programming familiarity enhances the possibility of researchers using Grid computing. In contrast, however, the MPICH-G programming model cannot provide programming flexibility. In other words, researchers find it difficult to make full use of the Grid functionalities with the MPICH-G programming model. This situation is against the vision researchers of the Grid envisage.

In this chapter, the author proposes *hybrid programming model*, a new programming model for Grid computing. As the name of the proposed programming model indicates, the author has designed a programming model that balances the trade-off between flexibility and familiarity in programming. Furthermore, the proposed programming model provides a programming template based on the data flow in Grid-enabled systems researchers want to develop. The programming template is effective for reducing development cost.

To verify the proposed programming model, the author has applied the Grid to MEG data analysis. Through the application of the Grid, the two goals clarified in chapter 3 were targeted: seamless integration of three stages essential for the analyses of brain functions, and dramatic reduction of the computational time for wavelet cross-correlation analysis.

This chapter begins with an overview of the MEG data analysis system. Next, to investigate the problems of the actual development of Globus programming model, two proto-typic MEG data analysis systems are developed on a Grid environment. These systems are simulated on the Local Area Network (LAN) in the Cybermedia Center, Osaka University, Japan. A MEG data analysis system is developed with the proposed programming model on the wide-area network over NAIST and Osaka University and the effectiveness and usefulness of the proposed programming model for Grid computing is verified.

4.2 System Overview

Figure 4.1 illustrates an overview of the MEG data analysis system. This system envisages the seamless integration of an MEG (brain database), high performance computers such as supercomputers and cluster systems, and a real-time rendering

system on the Internet.

The system shown in Fig. 4.1 had never been achieved until the emergence of the Grid. A factor that prevents the development of such a system may be explained by the fact that even though a scientific organization or university possesses such valuable gears as a supercomputer, a highly sophisticated medical device and large data storage, no technology that could integrate such gears in a uniform way existed until the Grid appeared.

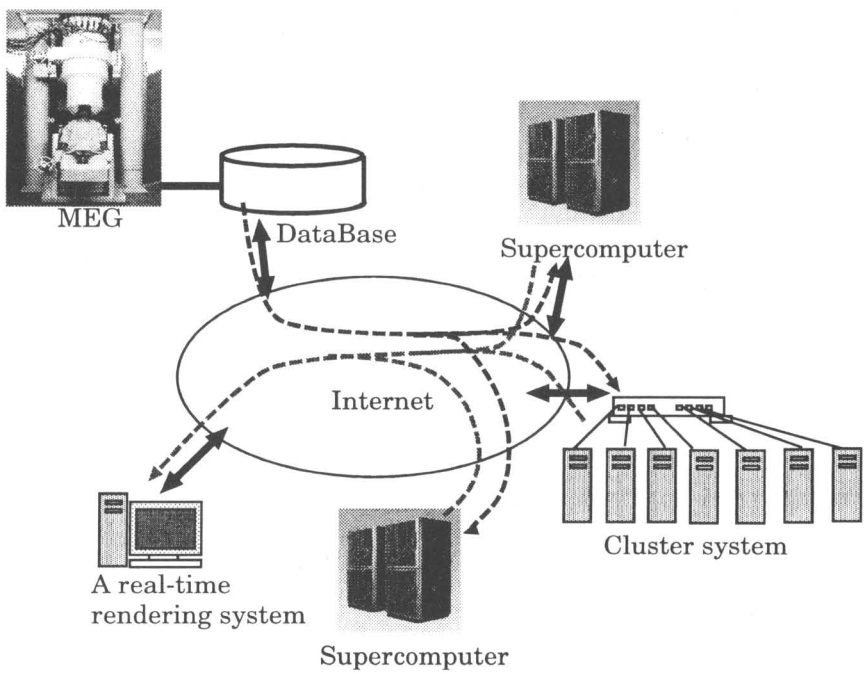


Figure 4.1: Overview of MEG data analysis (The figure of MEG is cited from [35].)

In this overview, the system integrates not only physical resources but also the three stages essential for MEG data analysis; the data acquisition stage, the data analysis stage, and the implementation stage of the analysis result. The author aims to achieve semless data transfer among these three stages with the Grid.

Furthermore, the system shown in Fig. 4.1 aims to dramatically reduce the computational time for wavelet cross-correlation analysis adopted in this research. More specifically, the author performs Grid computing on the Grid environment realized with multiple computers, which range from commercial personal computers to a supercomputer.

In sum, the MEG data analysis system based on this overview achieves the two goals targeted in Chapter 3:

1. Seamless integration of the three stages essential for the analyses of brain functions, and

2. Dramatic reduction of the computational time for wavelet cross-correlation analysis.

4.3 An Approach with the Globus Programming Model

In this section, a proto-typic MEG data analysis system is described. The MEG data analysis system is designed to use the APIs provided by the Globus grid toolkit and is implemented on a Grid environment simulated on the LAN in the Cybermedia Center, Osaka University, Japan.

The purpose of this section is to illustrate the difficulties and flexibilities of the Globus programming model. Simultaneously, the author aims to show researchers who are about to start building their own Grid-enabled systems how to build such a system with the following Globus programming model.

4.3.1 Simulated Grid Environment

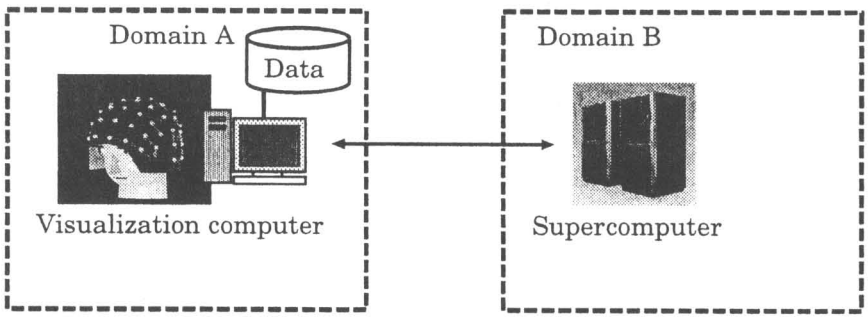


Figure 4.2: Overview of the first simulated Grid environme nt

Figure 4.2 shows the simulated Grid environment. The simulated Grid environment spanned over two different administrative domains. The simulated Grid environment was simple and was composed of two computers. As shown in Fig. 4.2, a commercial personal computer, which had a single Pentium III 500 MHz processor and 512 MB main memory, located on domain A was used as a visualization computer. This visualization computer was controlled by FreeBSD operating system. In

this environment, researchers and doctors were supposed to work in administrative domain A.

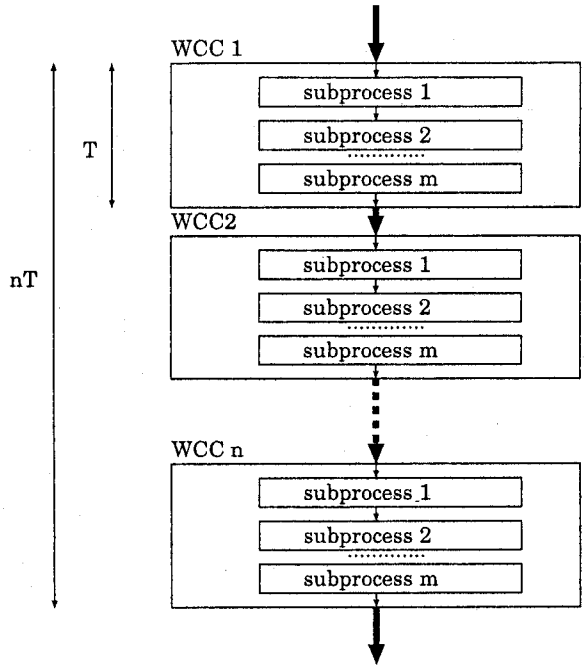
On the other hand, HP-UX Exemplar V2200/N, a supercomputer composed of two nodes, each of which had 16 PA8200 200 MHz processors and 16GB main memory, was used as a high-performance computer. Each node of the Exemplar could be viewed as two different computers because each node was controlled by a Unix-based operating system proprietarily developed by Hewlett-Packard (HP).

To realize the Grid environment on such an environment, the Globus grid toolkit was installed into the visualization computer, and two nodes of the Exemplar. The Globus provides four kinds of installation configurations called *flavor* with system administrators. In other words, system administrators can choose flavors when they install the Globus grid toolkit into their computers. The following two kinds of flavors, nothreads and pthreads, were selected for the visualization computer. In contrast, beside these two flavors, the remaining two flavors, mpi and nompi were selected so that researchers could decide whether or not they wanted to use MPI functionalities proprietarily developed by HP simultaneously with Globus APIs.

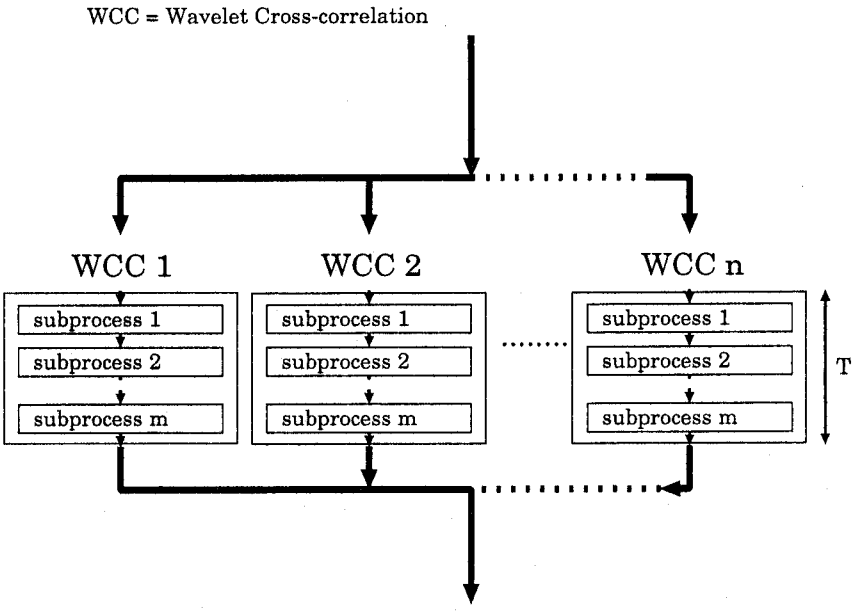
This simulated Grid environment allowed doctors and researchers to transfer medical data seamlessly to the visualization computer and the two nodes of the Exemplar. At the writing of this dissertation, no MEG has connected to a public network or the Internet because of security problems. This situation may be partly due to conventional practices in medicine. However, the connection of the MEG to the Internet is technically feasible. The author believes that the connection of MEG to the Internet will be realized. Therefore, in this stage of development, MEG data are expected to be located in a visualization computer. In the following sections, how the computational time for wavelet cross-correlation analysis was reduced on the simulated Grid environment is described.

4.3.2 A Design Strategy for Efficient Computation

To reduce the computational time for wavelet cross-correlation analysis in the environment shown in Fig. 4.2, the author designed a two-stage parallelism composed of coarse-grained and fine-grained parallelism. Coarse-grained parallelism performs load balancing with the Globus. Fine-grained parallelism performs load balancing with the Exemplar's MPI functionalities. More specifically, the visualization com-



Traditional sequential computation



Coarse-grained parallel processing

WCC = Wavelet Cross-correlation

Figure 4.3: Sequential computing and Coarse-grained parallelism

puter sends the computational workload to the Exemplar. The distributed computational workload on the Exemplar is first distributed between two nodes in the

coarse-grained parallelism. Next, the computational workload is further distributed among 16 processors on each node in fine-grained parallelism.

In this environment, the author designed an MEG data analysis sytem using Grid computing to distribute the computational workload for wavelet cross-correlation analysis. As described in chapter 3, wavelet cross-correlation analysis needs to be performed for 2016 pairs of MEG sensors at most in the case of using a 64-sensor MEG. In the simulated Grid environment, an MEG data analysis system was designed to distribute this computational workload for this "bag of tasks" in order to dramatically reduce the computational time for wavelet cross-correlation analysis.

Figure 4.3 shows a diagram of coarse-grained parallelism. If it takes T to perform one pair of wavelet cross-correlation analysis, the total analysis time of n pairs of wavelet cross-correlation analysis reaches at least nT in traditional sequential computation. In contrast, with coarse-grained parallel processing, if the workload is distributed to different n sites, the total analysis time can be reduced to at most T.

Next, in order to further shorten analysis time T, wavelet cross-correlation analysis itself is decomposed into subprocesses. These subprocesses can be performed in parallel.

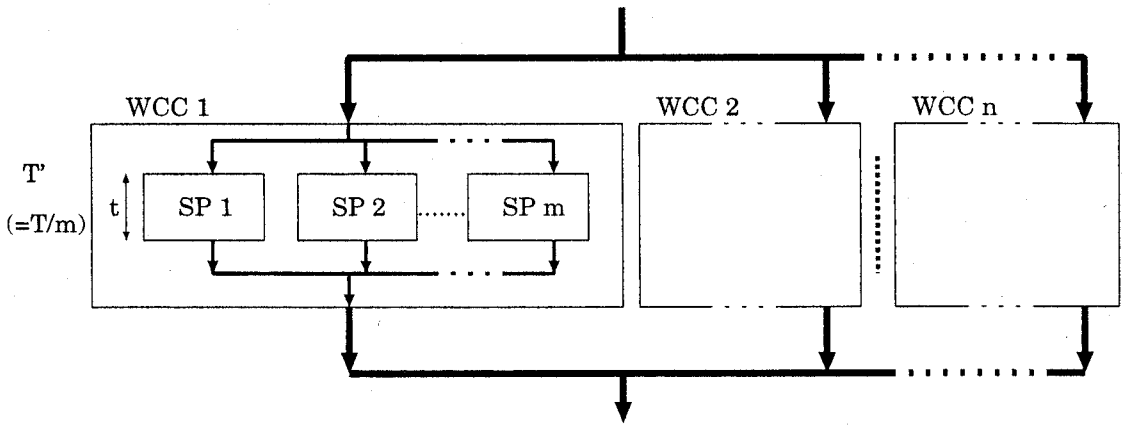


Figure 4.4: Coarse-grained and Fine-grained parallelism

Figure 4.4 illustrates the effect of this fine-grained parallelism. Wavelet cross-correlation analysis subprocesses are spread to multiple processors within a single computer. After these subprocesses are computed separately, the computation results are gathered to a certain processor. In this case, if m processors are used

for fine-grained parallelism, the total analysis time can be reduced to $T' = (T/m)$, which is the theoretical upper bound.

4.3.3 Implementation

The Globus provides for Nexus, which is a service for communication on wide-area network. Nexus allows users to express diverse communication models, ranging from point-to-point message passing to unreliable multicast communication, despite network heterogeneity [11]. A communication link is formed by binding two basic abstractions, that is, a startpoint and an endpoint.

In addition, this communication link supports a RSR (Remote Service Request) mechanism [17, 36]. This mechanism enables a certain program to execute an arbitrary module in a remote program. In other words, the RSR mechanism provides a mechanism that is similar to RPC (Remote Procedure Call).

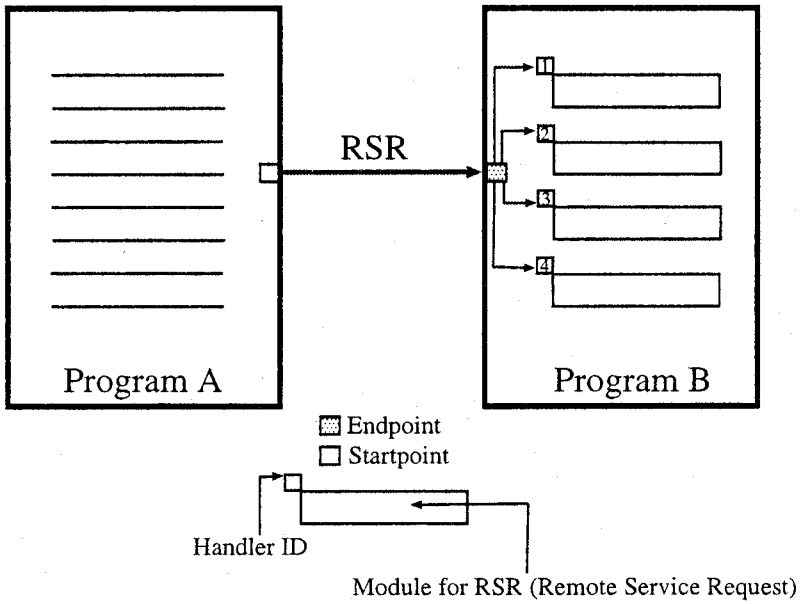


Figure 4.5: The Nexus RSR mechanism

Figure 4.5 illustrates the RSR mechanism. Program A has the startpoint bound to the endpoint on program B. Likewise, program B has the endpoint linked to the starpoint on program A. Program B is composed of some RSR modules, which are identified by handler ID. This handler ID is defined per endpoint. If the RSR with handler ID 1 is sent from a startpoint on program A, program B invokes the RSR module with handler ID 1.

The MEG data analysis system built on the simulated Grid environment was designed to take advantage of this RSR mechanism. Figure 4.5 shows the RSR mechanism adopted in the MEG data analysis system. This RSR mechanism is composed of the following two parts: the computation part and the visualization part. The computation part is located on each node of the Exemplar, and a visualization part is located on the visualization computer shown in Fig. 4.2.

These two parts exchange RSR messages with each other in order to share the computational workload for wavelet cross-correlation analysis. To realize this sharing, the visualization part has a pair of startpoints and endpoints to each computation part. Computation parts are mainly composed of the following three Nexus modules: the load data module, the calculation module, and the finalizing program module. The load data module has the role of loading MEG data from a data storage connected to the visualization computer. The computational module performs wavelet cross-correlation analysis for loaded MEG data and then sends computational results to the result visualization module in the visualization part. The finalizing program module performs the termination process.

On the other hand, the visualization module is mainly composed of the following two modules: the load result receiver module, and the result visualization module. The load result receiver module is responsible for loading MEG data. The result visualization module sends a visualization request to the visualization software described in chapter 3. Figure 4.6 shows how this RSR mechanism adopted in the MEG data analysis system works. This RSR mechanism allows researchers and scientists to repeatedly perform wavelet cross-correlation analysis in an interactive way.

Finally, to invoke a computational part on a remote computer, the MEG data analysis system used DUROC and GRAM APIs. A request with DUROC APIs are decomposed into individual GRAM requests and each request is submitted through GRAM service. In sum, DUROC services are substantially the same service as the GRAM service.

A job executed through GRAM API is specified with the RSL (Resource Specification Language). The GRAM service has the capability of implementing the RSL file written by researchers and then sending resource allocation requests to a gatekeeper located on the corresponding remote computer. The gatekeeper waits for computational requests while listening to the TCP 754 port. After receiving the request, the gatekeeper creates a job manager. The job manager is responsible

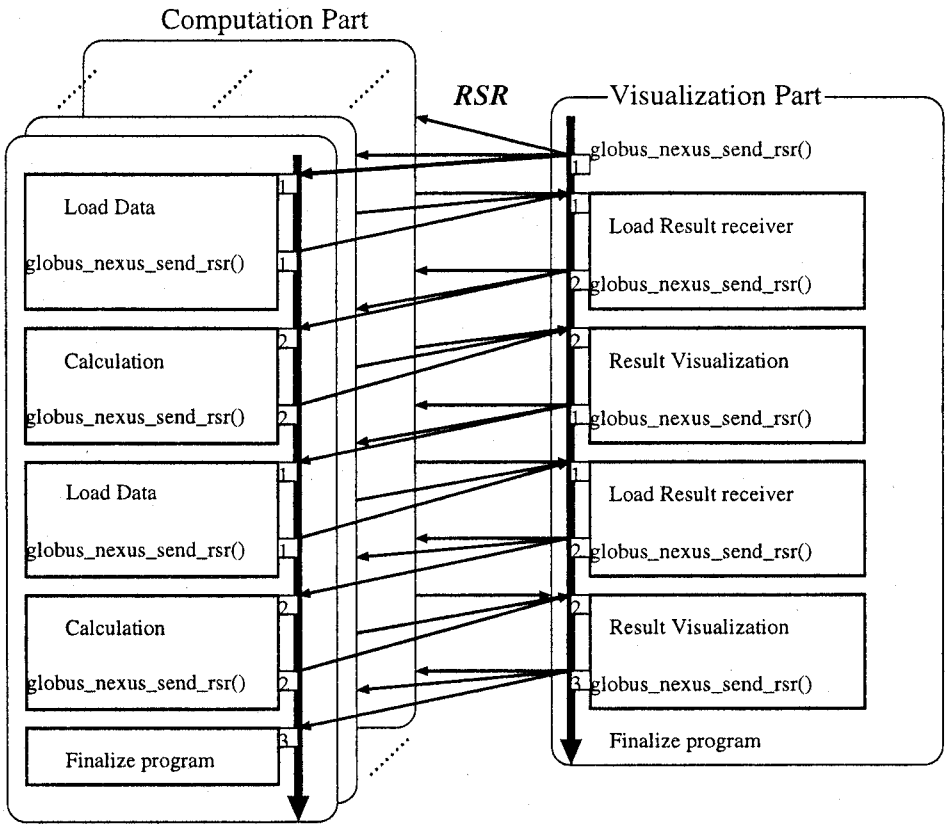


Figure 4.6: Interaction between the two parts

for the completion of the job and this manager generates a set of processes for the requested job (Fig. 4.7).

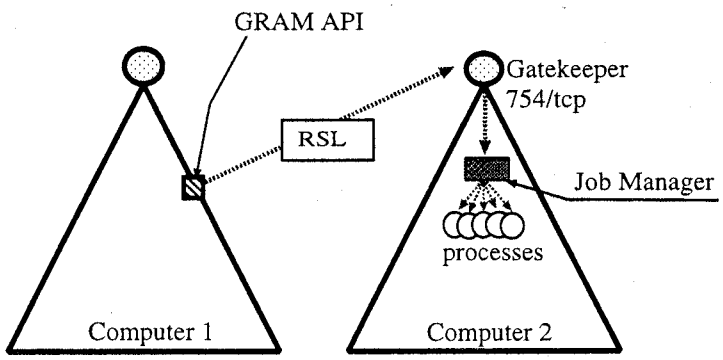


Figure 4.7: Resource allocation mechanism based on GRAM

4.3.4 Programming Difficulties

In this section, the author shows how to build the MEG data analysis system based on the design in the previous section 4.3.3. Furthermore, the author illustrates programming difficulties with the Globus programming model with actual programming codes. The author believes that this section is useful for researchers who want to develop their own Grid-enabled systems with the Globus programming model.

4.3.4.1 Visualization Coding

To construct the RSR mechanism shown in Fig. 4.5, the visualization part needs to perform the following three steps:

1. make an endpoint,
2. prepare a listening TCP port, and
3. invoke a computational part.

At the first step, an endpoint linked to server's startpoint is created. This endpoint must be linked to procedures invoked by server RSRs. This part is written as the following codes.

```
globus_nexus_endpointattr_init(&EpAttr);
globus_nexus_endpointattr_set_handler_table(&EpAttr,handlers,
                        sizeof(handlers)/sizeof(nexus_handler_t));
globus_nexus_endpoint_init(&GlobalEndpoint,&EpAttr);
```

Figure 4.8 illustrates what this code produces. First, in this code, the structure `handlers`, which has pointers to the RSR module, is set to the structure `EpAttr`. Next, the structure `EpAttr` is linked to the structure `GlobalEndpoint`. The `GlobalEndpoint` is the endpoint in the visualization part. This two-stage linking design enables users to reuse the structure `EpAttr`. In other words, this design enables more than two endpoints to share identical endpoint attributes.

Next, the structure `handlers`, which is linked to multiple RSR modules, is described as follows.

```
static globus_nexus_handler_t handlers[] = {
  {NEXUS_HANDLER_TYPE_NON_THREADED,attach_handler},
  {NEXUS_HANDLER_TYPE_NON_THREADED,receive_gaussian_report_handler},
```

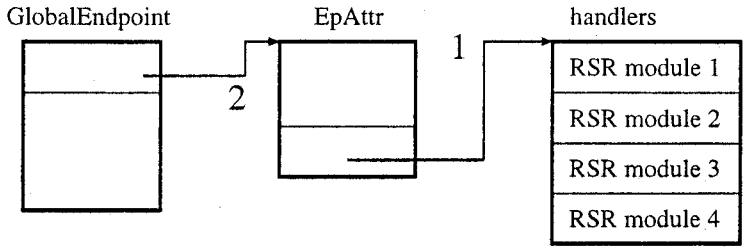


Figure 4.8: Making an endpoint

```
  {NEXUS_HANDLER_TYPE_NON_THREADED,receive_result_print_handler},
  {NEXUS_HANDLER_TYPE_NON_THREADED,receive_wavelet_result_handler}
};
```

In the above case, four procedures are registered as RSR modules. They can be identified by handler ID 0, 1, 2, and 3, in the order that they are defined. (The procedure `attach_handler` has handler ID 0.) These procedures have the following template.

```
void RSR_MODULE(globus_nexus_endpoint_t *endpoint,
                globus_nexus_buffer_t *buffer,
                globus_bool_t is_non_threaded_handler){
  /*
Here, developers can insert an arbitrary code.
  */
}
```

Here, three parameters in `RSR_Module` are defined by Nexus specification. The first and second parameters store a pointer to the endpoint, a pointer to a buffer containing the same data as sent by the computational part's RSR, respectively. The third parameter determines whether the RSR module uses a thread-based handler or not.

At the second step, a TCP port for listening to the computational part's attachment must be created before the visualization part invokes the computational part. In this system, the information on the visualization part listening to the TCP port is passed to the computational parts as a parameter at the computational part's startup. The TCP port is created as the following code.

```
  globus_nexus_allow_attach(&port,&host,
                                  attach_requested,GLOBUS_NULL);
```

The API `globus_nexus_allow_attach` creates the TCP port for listening to the computational part's attachment and returns the information on the TCP port into the first two parameters (port and host). The TCP port number is stored in the parameter port, and the hostname is stored in the parameter host.

The information on visualization part's listening TCP port needs to be received by the computational parts. The visualization part's makes a URL concerning the TCP port with the previous two parameters and passes the URL to the computational parts as a startup parameter. The URL is expressed in the following way:

```
x-nexus://host:port
```

A computational part attempts to attach the TCP port where the URL is identified.

As to the third parameter, a pointer to callback function must be specified. The specified callback function is invoked when the visualization part detects any computational part's attachment. Figure 4.9 illustrates the interaction.

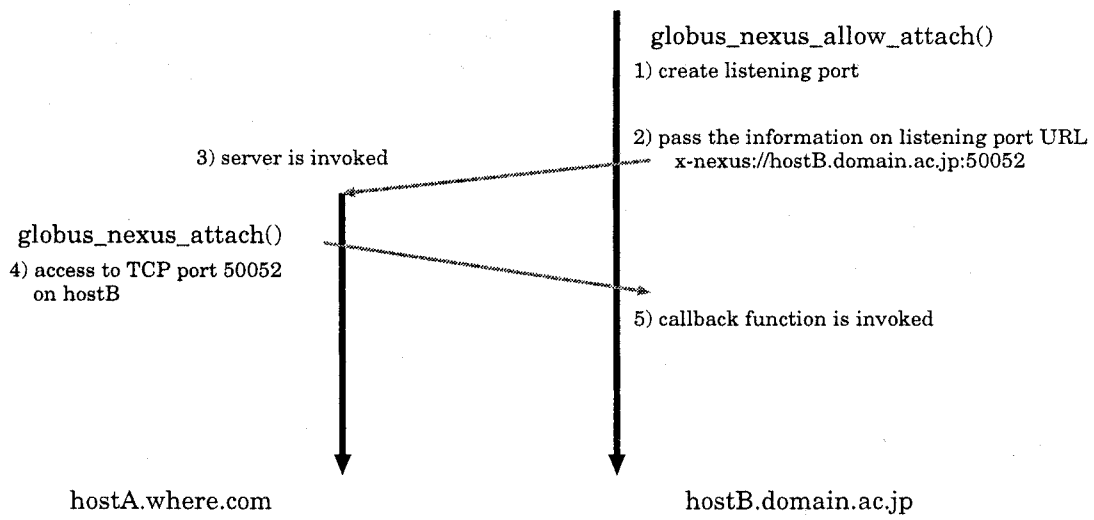


Figure 4.9: Nexus attachment mechanism

One final concern is how the computational part is invoked. As described in section 4.3.3, in order to invoke the computational part, DUROC and GRAM API were attempted. These two services implement the RSL (Resource Specification Language) and allocate or coallocate processes to computational resources and management processes. The core syntax of the RSL syntax is the relation. Relations associate an attribute name with a value. In this simulation, the RSL for the computational part of the invocation is written as follows.

```
& (directory=/home/.../bin/)(executable=server2)
      (argument=x-nexus://host:port param1 param2)
```

This RSL instructs a GRAM gatekeeper to invoke the program servers in directory /home/.../bin/ with parameter "x-nexus://host:port", param1, and param2.

4.3.4.2 Computational Part Coding

A server program primarily consists of the following four steps:

1. make an endpoint,
2. attach to a visualization part,
3. transfer the startpoint to the visualization part, and
4. send the RSR to the visualization part.

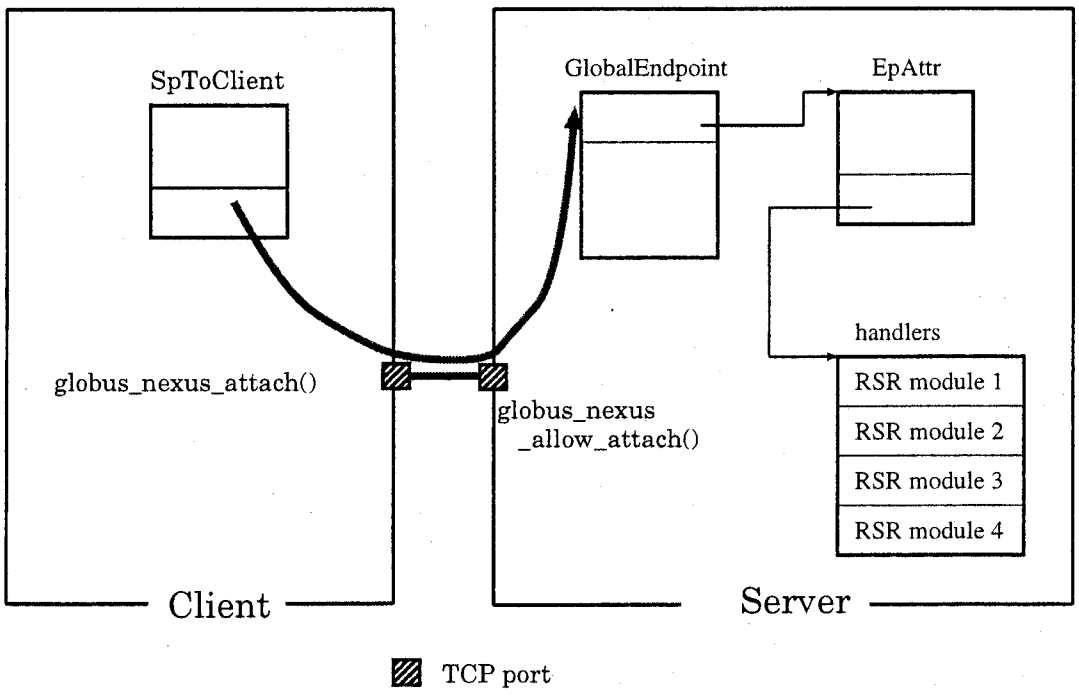


Figure 4.10: Attach to the client process

The first step is the same as the visualization coding. This step initializes and creates an endpoint linked from the corresponding computational part's startpoint. At the second step, a computational part attempts to attach the TCP port where the visulization part is waiting for the computational part to attach using

`globus_nexus_allow_attach()` the API. The computational part's attachment to the client is performed using `globus_nexus_attach()` the API as follows:

```
globus_nexus_attach(attach_url, &SpToClient);
```

The first parameter in this example is the URL expression of the TCP port where the client is listening to the server's attachment. We can use this API to link the startpoint in the server to the endpoint in client (Fig. 4.10).

4.3.5 Performance Evaluation

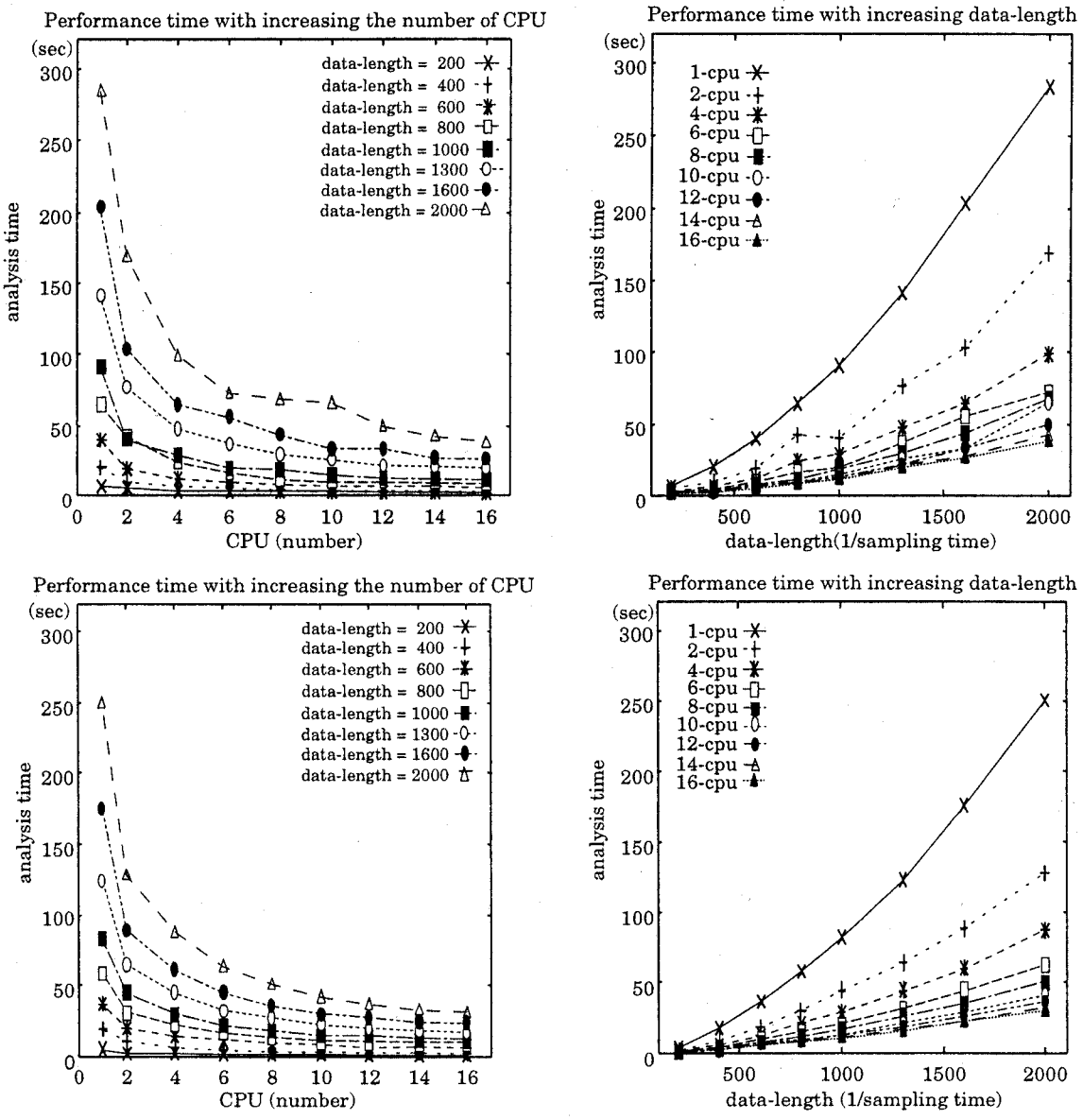


Figure 4.11: Real analysis time on Exemplar

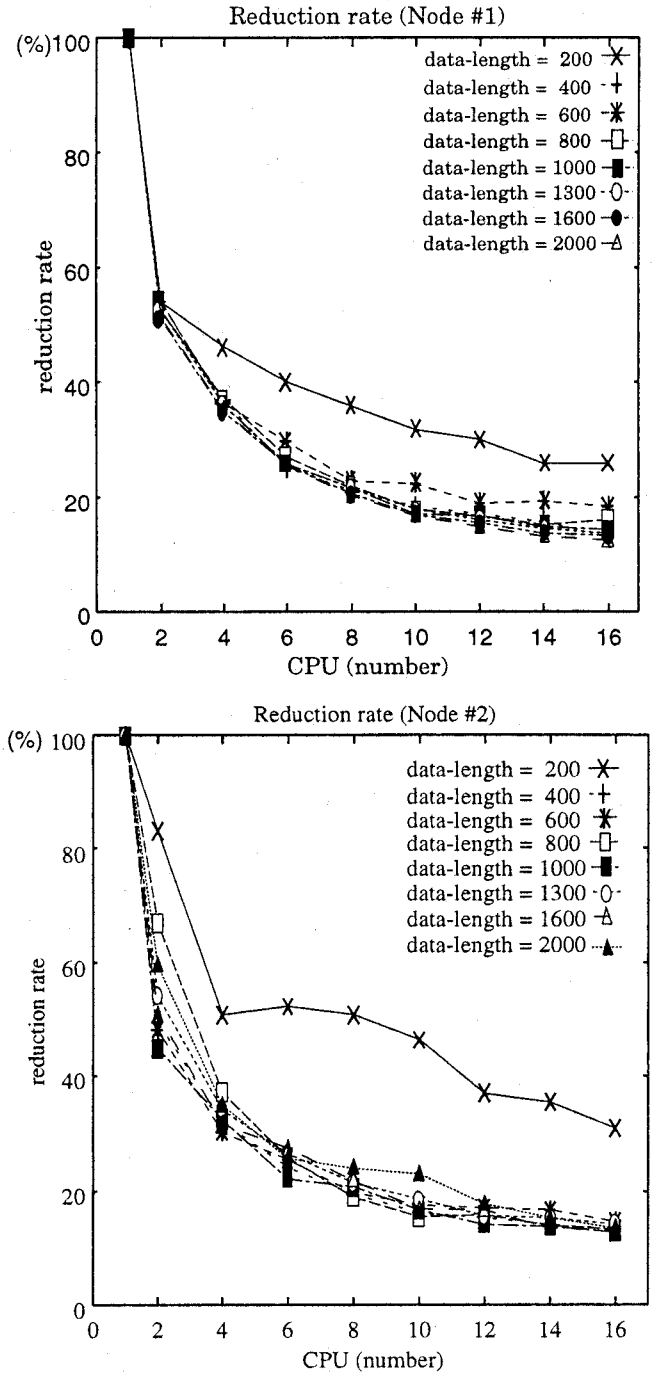


Figure 4.12: Reduction rate

In this evaluation, the analysis time needed for one pair of wavelet cross-correlation analyses was investigated by changing the data-length and the number of processors. The data-length is defined in terms of a sampling point. (For example, the data-length of 3.0-second data collected with a sampling interval of 4 ms is 750.)

Figure 4.11 shows the analysis time measured on each node in this experiment.

The analysis time in these graphs is the averaged number in 20 measurements. The left graph indicates that the analysis time decreases with an increase of the processors used for the fine-grained parallelism, despite data-length. The right graph indicates that the analysis time increases with an increase of the data-length, despite the number of processors.

Figure 4.12 shows the descent of analysis time, denoting the analysis time of a single processor as 100. According to this graph, with an increase in the numbers of the processors, the effect of this parallelism gradually descends. However, this figure also shows that at most approximately 8 times the acceleration of analysis time was obtained. Accordingly, through the use of both coarse-grained parallelism with two nodes and fine-grained parallelism, the analysis time has been accelerated approximately 16 times from a single processor computation.

4.4 An Approach with MPICH-G Programming Model

In this section, another proto-typic MEG data analysis system is described. The MEG data analysis system was built on a different Grid environment from the environment in section 4.3. The difference of these two systems is that the MEG data analysis system is designed to use MPICH-G APIs. MPICH-G is implemented with the APIs provided by the Globus grid toolkit and provides researchers with almost the same set of APIs as MPI. Therefore, researchers can obtain the powerful functionalities the Globus provides via familiar MPI APIs. Today, MPI and Parallel Virtual machine (PVM) are being used as a library for conventional parallel computing [37]. In fact, many computer vendors have offered MPI APIs for parallel computing, especially for parallel computing on a computer design based on distributed memory architecture.

4.4.1 Simulated Grid Environment

For the building of the MEG data analysis system, the author built a simulated Grid environment. Figure 4.13 shows the simulated Grid environment. The simulated Grid environment is composed of a visualization computer and a cluster system composed of seven nodes, each of which is connected to the Gigabit Ethernet, that is, 1 Gbps network. The visualization computer is the same as the computer shown

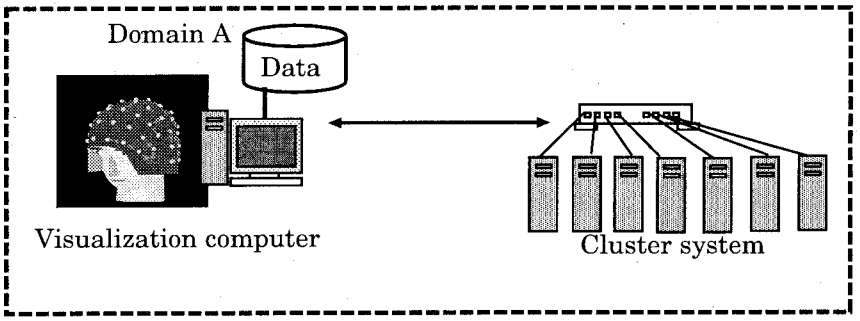


Figure 4.13: Overview of the second simulated Grid environment

in Fig. 4.13. Each node of the cluster system has a single Alpha EV56 500 MHz processor and 1 GB main memory. A file system on each node is not shared with other nodes. Each node can be viewed as a single computer.

Next, as to the software environment, the Globus grid toolkit and MPICH-G were installed into the visualization computer and a node of the cluster system. MPICH is installed into all nodes of the cluster system. The purpose of this configuration is described in section 4.4.2.

4.4.2 A Design Strategy for Efficient Computation

The MPICH-G is supposed to be a Grid-enabled version of the MPICH, which is a software for conventional parallel computing with a message passing method. The Grid-enabled MPI allows a user to run MPI programs across multiple computers at different sites using the same commands that would be used on a parallel computer [38, 39, 40]. Moreover, MPICH-G enables a user to run existing MPI programs on a heterogeneous wide-area network just by recompiling it for the Grid environment.

To distribute the computational workload for wavelet cross-correlation analysis, the author configured the cluster system so that the cluster system can perform a two-stage parallelism. In the two-stage parallelism, MPICH and MPICH-G are simultaneously used. The difference of MPICH and MPICH-G exists at the level of parallelism. The first stage distribution of the computational workload is performed with MPICH-G functionalities, more precisely, the Globus functionalities. Next, the second stage distribution is performed with MPICH. In short, in the case of the environment shown in Fig. 4.13, the computational workload for wavelet cross-correlation analysis is transferred to the cluster system. Next, the distributed workload is shared by all nodes of the cluster system (Fig. 4.14).

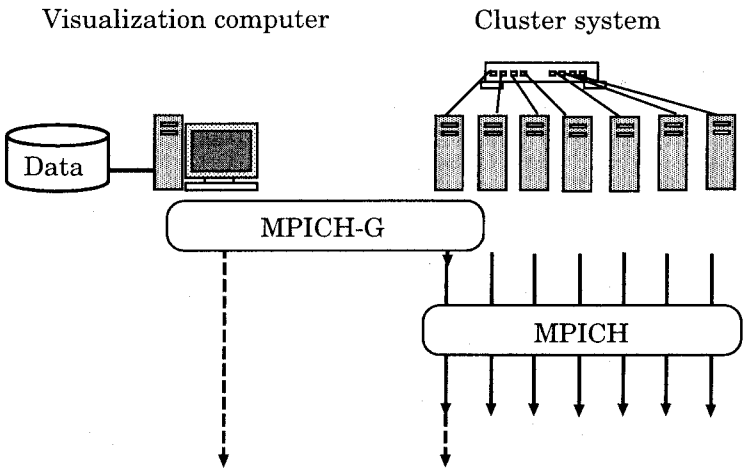


Figure 4.14: Task distribution mechanism

First, two MPICH-G processes are generated on the visualization computer and a node of the cluster system as follows:

```
mpirun -np 2 -machinefile machines myprog
```

In this case, the processes of a program, **myprog** are generated on the computers whose hostnames are listed in a file **machines**. This method of generating MPICH-G processes is the same as conventional parallel computing with MPICH. Therefore, researchers are familiar with this method. This familiarity is the most beneficial point in the MPICH-G programming model. This familiarity is never achieved in the Globus programming model. Conversely, however, in the MPICH-G programming model, researchers cannot dynamically generate processes on demand. Therefore, researchers need to preserve the maximum number of computers in advance. This fact means that the MPICH-G programming model does not provide researchers with flexibility. Researchers have to balance the trade-off between familiarity and flexibility in programming.

Next, in the cluster system, second-stage parallelism is performed. The MPICH-G process on the node of the cluster system generates the MPICH processes on all nodes of the cluster system.

4.4.3 Implementation

In the system with the MPICH-G programming model, the computational part and visualization part use identical programs. Each process in this system is identified by rank and is programmed per rank.

Unlike the RSR mechanism in the system based on Nexus, there is no interaction between two organizations during computation. In this system, MEG data analysis results are transferred in a network transparent manner, just before and after computation.

First, the visualization part must send MEG data to the computation part. In the case where MPI_Send and MPI_Recv are used, this is implemented as follows.

```
MPI_Comm_rank(MPI_COMM_WORLD,&myid);
MPI_Comm_size(MPI_COMM_WORLD,&numprocs);

if ( myid == numprocs - 1 ) {
  for (p=0;p<numprocs-1;p++){
    MPI_Send(&braindata[data_length*p],data_length,MPI_FLOAT,
             p,tagid+p,MPI_COMM_WORLD);
} else {
  for (p=0;p<numprocs-1;p++){
    MPI_Recv(MEGdata,data_length,MPI_FLOAT,numprocs-1,
             tagid+p,MPI_COMM_WORLD);
  }
}
```

Rank N - 1 process sends the content of `data_length` succeeding entries of the type indicated by MPI_FLOAT from the address where `&braindata[data_lengthp]` points to rank p process. On the other hand, rank $0 \cdots$ N - 1 processes receive the message and store the message into `MEGdata` in rank.

Alternatively, in the case where MPI_Bcast is used, the code is more easily written as follows.

```
MPI_Comm_size(MPI_COMM_WORLD,&numprocs);

MPI_Bcast(braindata,data_length*numprocs,MPI_FLOAT,
          numprocs-1,MPI_COMM_WORLD);
```

Rank N - 1 process broadcasts the content of `data_lengthnumprocs` succeeding entries of the type indicated by MPI_FLOAT from the address `braindata` which points to other processes.

There is a difference between these two approaches. The former approach distributes MEG data to an appropriate site, whereas the latter approach sends all

brain data to all sites. In other words, unnecessary data is also sent to various sites. However, all MEG data can be visualized in the visualization part.

After computation, analysis results are gathered to the visualization part and visualized there. In this system, MPI_Gather is used for this purpose.

4.4.4 Performance Evaluation

In this evaluation, the analysis time was investigated in terms of both real time and system time. For measurement of real time, Time command provided by UNIX-based operating systems was used. For the measurement of system time, Jumpshot was utilized. The Jumpshot is a tracing tool written by Java. It can analyze a CLOG binary file, and provide users with the information when an event occurred and where it occurred in terms of the location in the process that executed it. Furthermore, this information can be visualized (Fig. 4.15).

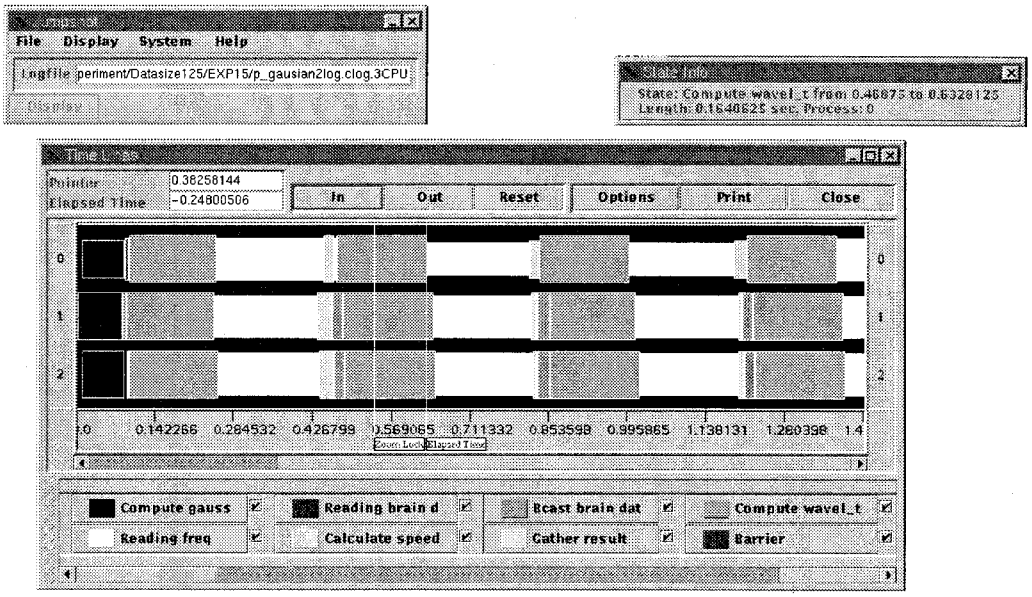


Figure 4.15: Jumpshot snapshot

In this evaluation, to generate a CLOG binary file, MPE (Multi-Programming Extensions) library [41] was used. This library provides APIs for tracing an MPI program. We can easily trace parallel programs using MPE APIs. For example, for measuring the system time of a certain region in the source program, all we have to do is to compile our program with the MPE library after inserting the following code.

```
main(){
    MPE_Init_log();
    MPE_Describe_state(1, 2, "Bcast pair",      "yellow:gray0");
    MPE_Start_log();

    MPE_Log_event(1,0,"Start Log");
/*
        a certain region
*/
    MPE_Log_event(2,0,"End Log");

        ..................

    MPE_Finish_log("output.clog");
}
```

This code generates a CLOG binary file after being executed.

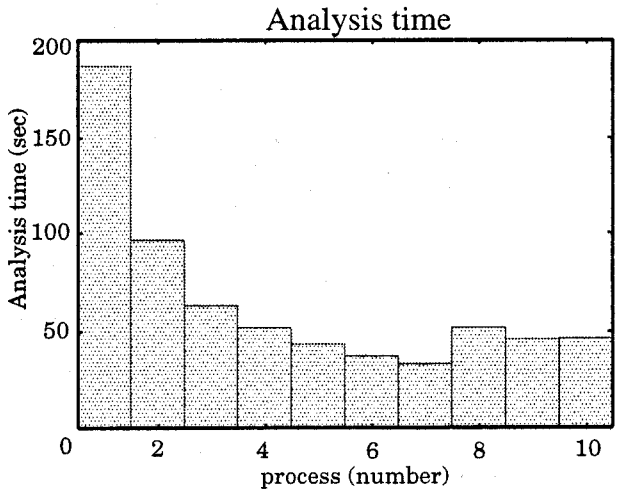


Figure 4.16: Real analysis time on an alpha system

First, the author considers the result of analysis time. In this measurement, it is assumed that 50 pairs of wavelet cross-correlation analysis are passed to this Alpha machine for coarse-grained parallelism and 1.0-second brain data (sampling interval 4ms), that is, brain data with data-length of 250 are analyzed.

Figure 4.16 shows the measurement result. According to this graph, with an increase of process, the real analysis time decreases. When the degree of parallelism

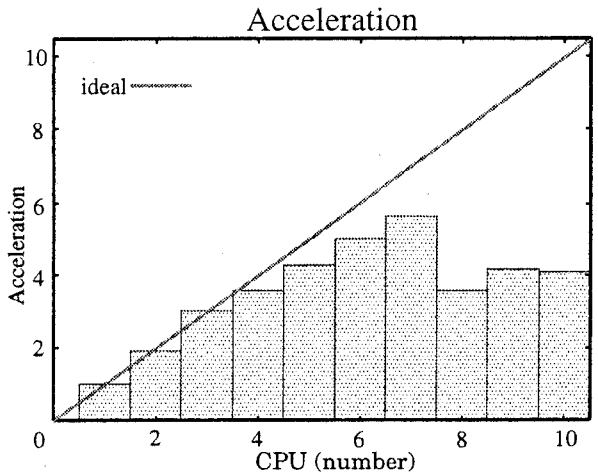


Figure 4.17: Acceleration on an alpha system

reaches 8, the analysis time jumps up. Clearly, this is explained from the fact that the number of processes exceeds the number of processors. In short, the waiting time for processor allocation increases.

In order to observe the effect of fine-grained parallelism, the analysis time in Fig. 4.17 is normalized, denoting analysis time with one process as 1.0. This graph shows that approximately ideal acceleration is obtained until the degree of parallelism reaches seven.

Next, let me illustrate the system time in this wavelet cross-correlation analysis. As is described in section 3, wavelet cross-correlation analysis has three time-consuming parts: the mother wavelet generation part, the wavelet transformation part and the cross-correlation part. In this measurement, the system time needed for each of these three nodes was measured (Fig. 4.18).

Each graph in Fig. 4.18 plots the system time needed for the part. This system time was measured per process. (When this parallelism was performed with N processes, N measure values were obtained.) The curve in each graph is the one estimated from analysis time with one process.

These three graphs indicate that for any part, the ideal workload distribution to each process has been achieved. Also, the variance of the system time increases when using more than 8 processors.

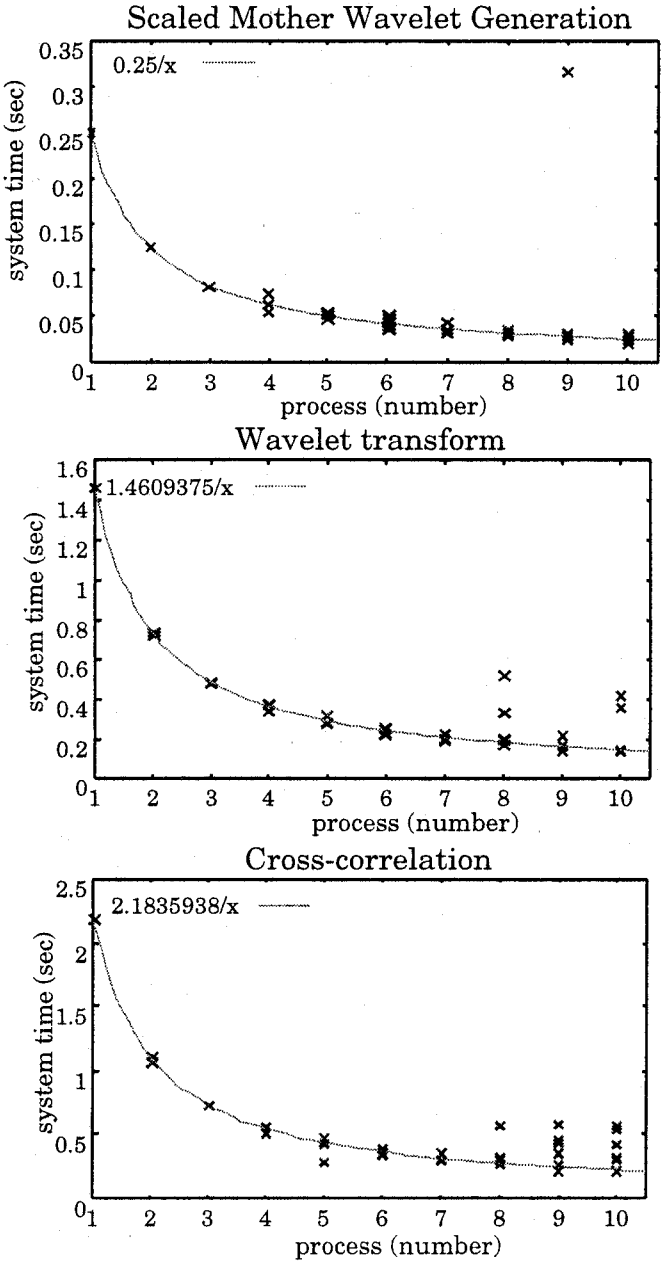


Figure 4.18: System time on the alpha system

4.5 An Approach with Hybrid Programming Model

In the previous sections 4.3 and 4.4, the author described how to build two MEG data analysis systems on the simulated Grid environments shown in Fig. 4.2 and Fig. 4.13. The development with Globus programming model is difficult without detailed knowledge of the Grid and the Globus grid toolkit, although the Globus grid toolkit hides its complexity with unformed APIs. This situation of development prevents researchers and scientists from efficiently developing their own Grid-enabled systems. On the other hand, the previous section 4.3 showed with actual programming codes that building a Grid-enabled system with MPICH-G programming model is easier than with Globus programming model because of the familiarity of the developer. However, the development with MPICH-G programming model has the problem that programming flexibility is lost. Thus, a new programming model that provides researchers both flexibility and ease in programming for Grid computing is demanded.

In this section, the MEG data analysis system based on the design and implementation of the *hybrid programming model*, or the new programming model, is described. First, the hybrid programming model is detailed. Next, the MEG data analysis system which has been built on the Grid environment simulated on the LAN at Osaka University is described. Finally, the MEG data analysis system which has been built on the Grid environment on the network over NAIST and Cybermedia Center, Osaka University is detailed.

4.5.1 Programming Template

The Globus grid toolkit is supposed to be extremely promising for the application of scientific problems. Currently, however, few effective programming models exist because the Grid is an emerging technology. This situation prevents developers from efficiently writing their own application codes with the Grid.

To make it easier for researchers and doctors to write their own application codes with the Grid, this research proposes a new programming model which provides developers with the programming template for Grid programming. The development cost is reduced through the use of the programming template.

However, providing the programming template for Grid programming could rob developers of programming flexibility. This results in a situation where developers have difficulty realizing what developers want to build on the Grid environment.

In this research, by focusing on the data flow in the Grid application, the author aims to propose a new programming model providing a programming template without robbing developers of programming flexibility.

In order to provide a programming template to fit the requirements of developers, understanding how the Grid is applied for a variety of scientific problems has great importance.

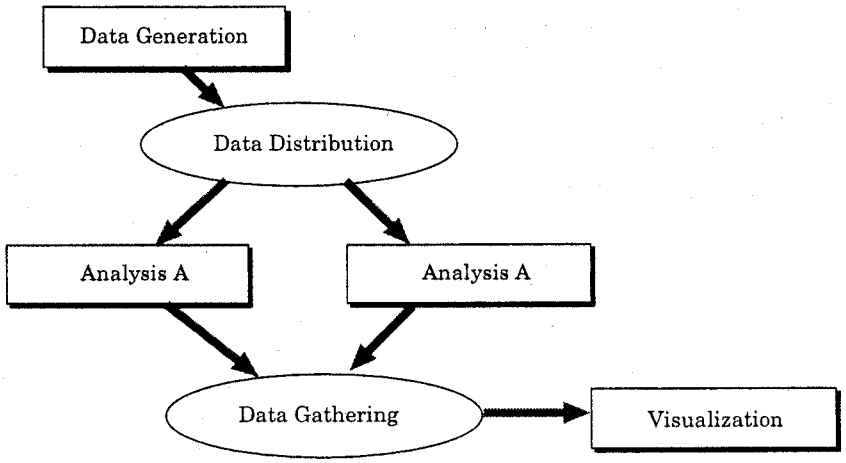


Figure 4.19: General data flow model on the Grid

As mentioned previously, the Grid has been developed to solve geographical distribution problems relating to computational resources. In other words, the Grid aims for a seamless integration of the procedures necessary for solving scientific problems on the Internet. Figure 4.19 illustrates the general data flow predicted to occur in the application of the Grid. The large amount of data generated from a scientific device is distributed to multiple computers on the Internet. The distributed data is analyzed on each computer. After computation, the analysis results on each computer are gathered for the purpose of visualization.

This research has also clarified an extended data flow model on the Grid. The prominent feature of the Grid is the ability to use virtually infinite computational resources to gain high-performance. Therefore, easily imagined is the situation where scientists simultaneously perform multiple analyses for the acquired scientific data (Fig. 4.20).

Based on the data flow model shown in Fig. 4.20, the author proposes the programming template composed of the following three modules: the analysis module, the communication module, and the visualization module (Fig. 4.21). The communication module provides programming codes regarding the communication module

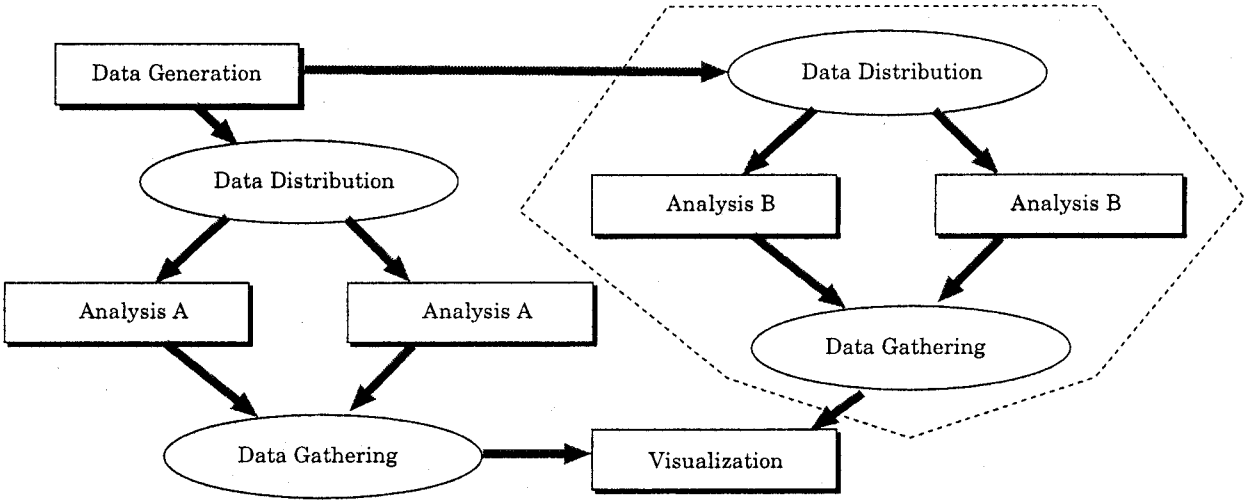


Figure 4.20: Extended data flow model on the Grid

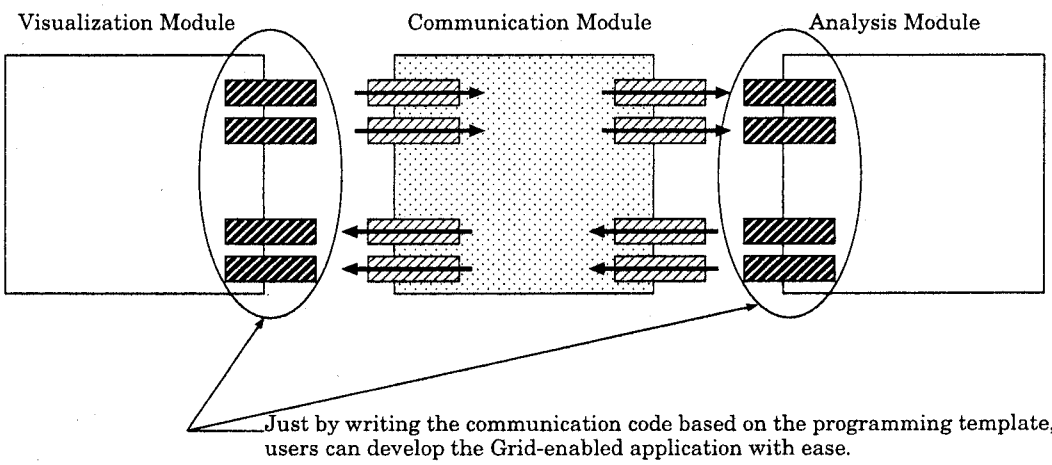


Figure 4.21: The programming template on the Grid

based on the data flow in the Grid application. Thus, the programming template allows users to develop their own visualization and analysis modules without being aware of the complex communication module. For example, users can easily develop a new analysis module just by being aware of the communication protocol provided by the programming model.

Furthermore, the programming template composed of these three modules enables users to develop the analysis module with either Globus APIs or MPICH-G APIs, as far as the users follow the communication protocol provided by the programming template. From this viewpoint, the proposed programming model is named a hybrid programming model. The flexibility with which users can select program-

ming APIs in developing the Grid-enabled application promotes the reusability of existent codes. This point is important in terms of the reduction of development costs.

Below, the author details the MEG data analysis system with the proposed hybrid programming model.

4.5.2 Software Architecture

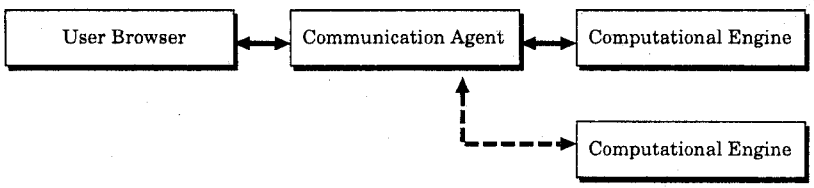


Figure 4.22: The modular structure of the system

Figure 4.22 diagrams the modular structure of the system. The system is composed of three major modules: the user browser module, the communication agent module and the computational engine module.

The reason for adopting the modular structure can be explained primarily from the flexibility and scalability in developing the system. Systems with monolithic structures can be designed and implemented more easily than modular systems, since the developer can build the system without considering complex matters such as the timing of communication between modules. However, we need to consider a situation where we might want to append a new function and slightly modify the system. Monolithic systems make that situation more difficult to solve than modular systems because the modification may affect the whole system structure. On the other hand, in the modular system, we have only to modify the correspondent module. In this research, the author plans to append new computationally-intensive medical analysis softwares such as Independent Component Analysis (ICA) [42] to the system. The modular system allows us to not only modify the system, but also to share the analysis softwares with other research groups. This sharing of knowledge is important to the system.

To realize the sharing of knowledge, the author has designed a system composed of three modules. The first module accepts a doctor's request for analyzing the data, and then passes the request to the second module. The second module mediates

communication between the user browser module and the computational engine. The second module is capable of communicating with more than two computational engines. The computational engine module has the responsibility of analyzing medical data quickly.

4.5.3 Communication between Modules

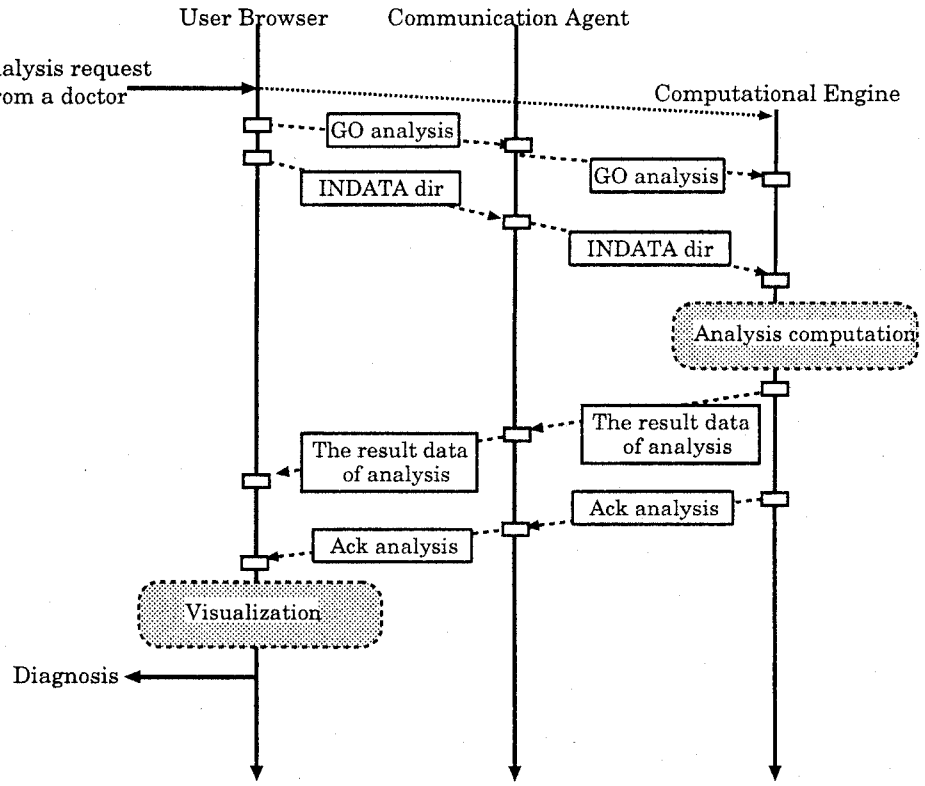


Figure 4.23: A communication mechanism between modules

Figure 4.23 illustrates the communication flow between three modules. The communication flow is simple and is based on the data movement model in general parallel computing on the Internet. First, the user browser module accepts an analysis request from a doctor and transfers the request to the communication agent. Second, the user browser invokes the computational engine corresponding to the analysis request. Third, the communication agent sends the analysis request to the engine. Fourth, the computational engine invoked is informed of the location of the medical data necessary for the analysis. Fifth, after the computationof analysis, the computational engine module sends the result of the analysis back to the user agent

module via the communication agent. Finally, the user browser agent visualizes the result data received.

The communication flow just described by the author can be performed repeatedly, except for the invocation of the computational engine module, until the doctor sends a request for termination to the communication agent. This communication mechanism introducing the communication agent as a mediator allows doctors to exchange the computational engine modules on demand in a plug-in manner.

4.5.4 A Design Strategy for Efficient Computation

The system has been designed based on *Client-Manager-Worker Model* [43]. The user browser module, which is a client, asks the manager of the computational module for the computation of medical analysis, via the communication agent. The manager requires all workers to complete their computations.

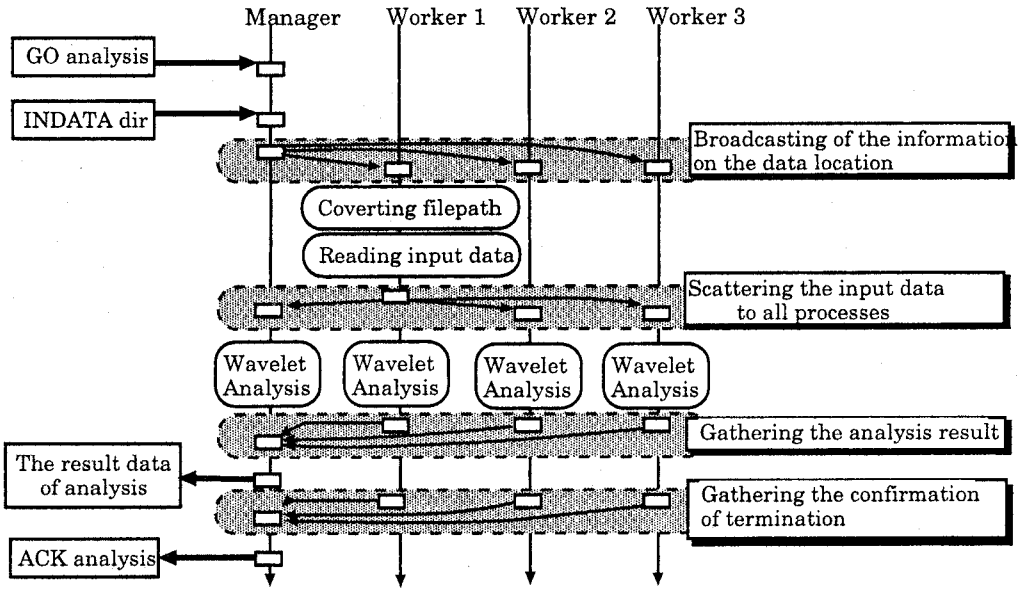


Figure 4.24: Internal communication mechanism in the computational engine module

Figure 4.24 illustrates the internal communication flow in the computational engine module for wavelet analysis. The computational engine module is based on the manager-worker parallel computing model and is composed of two entities: manager and worker. The principal communication framework for the computational engine module is as follows. The manager of the computational engine module accepts the computational request and the information on the location of the medical

data necessary for the computation from the client. Next, the manager and workers perform the computation in a way optimized for the requested analysis. After the computation, the manager gathers the results of analysis from all workers and then transfers that result data to the communication agent. Finally, all entities confirm with each other the termination of the computation and the manager reports the termination of the mission.

For wavelet analysis, a Single-Program Multiple-Data (SPMD) paradigm has been adopted for the development of the computational engine module. In this research, the analysis of Magnetoencephalography (MEG) data has been attempted. The MEG has the ability to measure the strength of the magnetic field generated by the neural activities inside the brain. Our MEG measurement is performed at sixty-four points around a head. The wavelet analysis needs to be simultaneously performed for sixty-four series of MEG data, since the result of the wavelet analysis offers doctors clues for finding brain signals of interest. These two computing requirements of wavelet analysis are suitable for the SPMD paradigm.

4.5.5 Implementation

The author has implemented the computational engine module for the wavelet analysis with MPICH-G2 [38, 40, 44]. MPICH-G2 is a Grid-enabled version of MPI, which operates on top of the Globus grid toolkit. The Globus grid toolkit provides a middleware that allows users to build an application over a wide-area network by handling distributed, heterogeneous computing resources. The middleware consists of the services essential for Grid computing [4].

In this system, the Globus and MPICH-G2 software have been installed onto all computers in Fig. 2.1. This software environment allows us to run an MPICH-G2 program over the network between NAIST and Osaka University.

Figure 4.25 diagrams the computational engine module. The computational engine module can be triggered with a RSL (Resource Specification Language) specification [45] transmitted by the user browser agent on the visualization computer. The Globus system interpretes the RSL specification, and then invokes an arbitrary group of MPICH-G processes based on the RSL specification. The RSL specification allows users to specify the computational resources necessary for the computation.

In the system, the Meta-Directory Service (MDS), which is the information management service the Globus provides can be used for the submission of jobs.

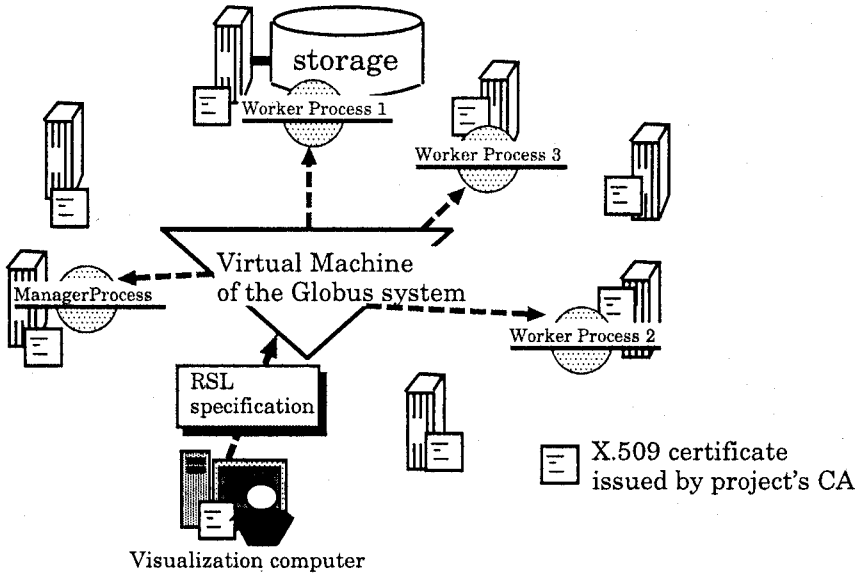


Figure 4.25: Computational engine module

At present, the author has investigated the light-weighted computers from the computers in our Grid environment by using UNIX command such as *grid-ldap-info*. In order to take maximum advantage of the information services, the author has provided some sets of optimized templates of RSL specifications that should benefit doctors. In near future, the author plans to design and implement an automatic generation mechanism of RSL specifications.

Figure 4.24 illustrates how the worker 1 process plays an important role in the reading and scattering of MEG data to the other workers and managers. In this system, the worker 1 process loads a medical data file using the information on the file location notified by manager process and then distributes the data to all processes in the computational engine module. However, before reading a data file, the worker 1 process needs to convert the SFS filepath provided by doctors to the filepath on the computer where the data is actually located. Doctors in front of the visualization computer assumes that the MEG data is located in the local disk, because the data access structure with location-transparency described in chapter 2 allows doctors and reseachers to access the data in the directory */sfs/hostname/*. However, as described in chapter 2, the medical data file of their interest may be located on a remote computer. In the case, the conversion of the sfspath to the actual file path where the MEG data is acutally located needs to be performed.

In this system, the author has adopted a strategy that automatically generates

the RSL specification from the SFS path which the doctor inputs so that the worker 1 process runs on the storage computer. This strategy allows the worker 1 process to read a medical data file from a native disk on the computer where the worker 1 process runs. The hostname where the worker 1 should be run can be obtained from the hostname part of the SFS path. Next, the remaining portion of the SFS path after removal */sfs/hostname/* is sent to the worker 1. The worker 1 process generates the actual filepath on the native computer by interpreting the SFS configuration file (Fig. 4.25).

As another strategy, it is possible that all computers statically mount the SFS server's filesystem in advance. In this case, the conversion is not necessary. However, the analysis time becomes longer than in the case where worker 1 performs the conversion, since the encryption time of SFS make the transfer time of data long. Therefore, the author has adopted the policy of protecting not all information regarding medical data but only the medical data with information specifying a patient's name. More specifically, the system provides doctors with the SFS filepath so as to confirm the information regarding the MEG data, while the access to the MEG data with the actual filepath has been used in the computational engine module for performance reasons.

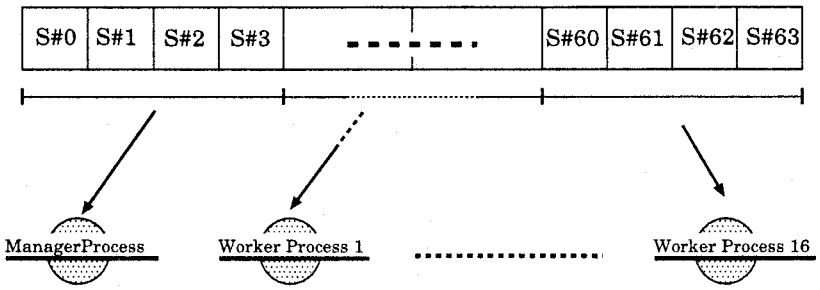


Figure 4.26: Data scatter

After the conversion, the worker 1 process needs to distribute the MEG data it reads. Figure 4.26 shows how the MEG data is scattered in the computational engine module. In order to maximize the efficiency in the computation for wavelet analysis, the author has assumed the number of processes in the computational engine module is one of common divisors for sixty-four, except for one. This assumption has been observed with the RSL specification generated in submitting jobs to the globus system. By observing this assumption, researchers and doctors can can maximize computational performance for the wavelet analysis based on SPMD paradigm.

After the computation of the wavelet analysis, the manager process gathers the analysis results from all processes in order to send back the analysis results data to the user browser module via the communication agent. The gathering of MEG data is performed in the reverse manner of scattering.

4.5.6 Evaluation

In this section the author evaluates the system performance. For this purpose, the author first measured the total time consumed in the computational engine module. Next, the author considered the overhead incurred by the SPMD paradigm for wavelet analysis. The measurement was performed with Jumpshot-3 [46].

4.5.6.1 Measurement Environment

In order to precisely evaluate system performance, the measurement environment needs to be described first.

ID	Location	architecture	#
1	NAIST	Pentium III 866MHz	6
2	NAIST	Pentium III 450MHz	6
3	Osaka	Pentium III 700MHz	8
4	Osaka	Alpha 751MHz	8
5	Osaka	Pentium III 500MHz	1

Table 4.1: Clusters in the project testbed

Table 4.1 shows our computational environment. The IDs correspondent to the counterparts in Fig. 2.1 in the chapter 2 of the dissertation. For measurement, the author used three clusters; namely 1, 2 and 3, and the visualization computer with ID 5. The clusters with ID 4 were not used because the cluster was administered as a service computer and therefore the cluster was heavily load-weighted.

Next, the author needs to detail the information on the network between NAIST and Osaka University in order to clarify system performance. The network connects these two scientific institutions via WIDE network [47] at low latency and high bandwidth. The slant distance of the network is nearly 40 Kilometers. The average round-trip time between NAIST and Osaka University was 5.755 milliseconds, compared with 0.902 milliseconds at the LAN between the visualization computer

and the CA computer. These figures were obtained with *ping*. In addition, the bandwidth on the network was also obtained with *pathchar*. The result shows the bandwidth at the bottleneck link on the network path was approximately 25 Mb/s, compared with 47 Mb/s at the LAN.

4.5.6.2 The Total Time Consumed in the Computational Engine Module

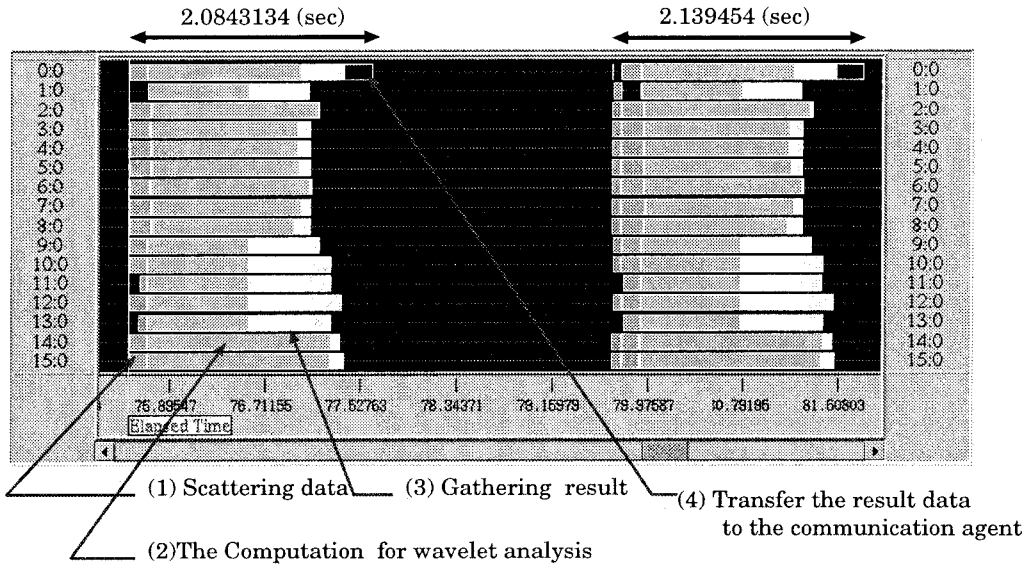


Figure 4.27: Measurement result with 16 computers

Figure 4.27 shows the measurement result of Jumpshot-3 when 16 computers were used for the computational engine module. The jumpshot result allows us to investigate each process's behavior in detail.

In the Client-Manager-Worker paradigm which the author has adopted for the wavelet analysis of MEG data, the descent of the worker processes' computational performance finally results in the descent of the manager processe's computational performance. This descent of the manager processe's computational performance causes the descent of the entire system. Thus, investigating the computational performance of the manager process is essential for the evaluation of the system's performance.

From the above viewpoint, the author measured the manager processe's total time consumed in the computational engine module. The total time is mainly composed of the following four items as shown in Fig. 4.27:

1. the time for scattering input MEG data,

2. the computational time for wavelet analysis,

3. the time for gathering the analysis result time from each process, and

4. the time for transferring the gathered result data to the communication agent.

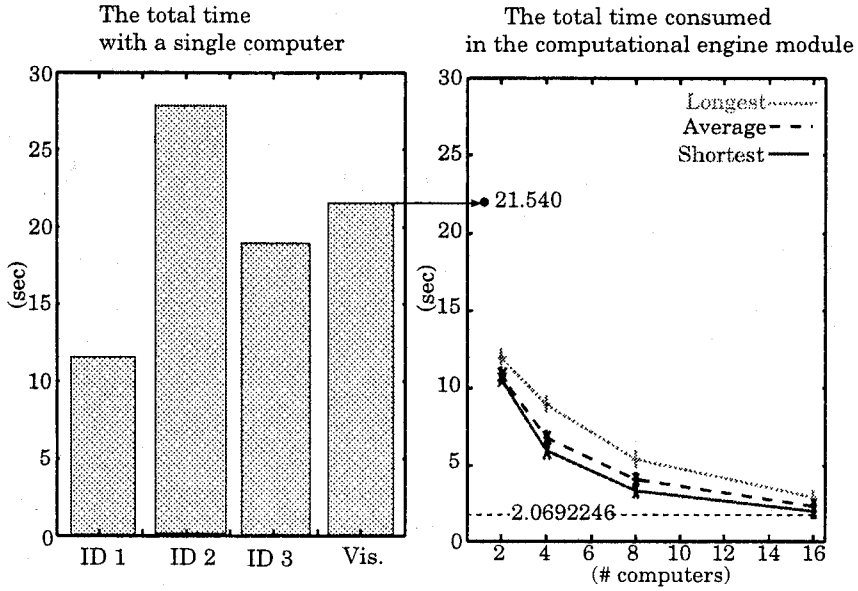


Figure 4.28: The total time consumed in the computational engine module

The left graph in Fig. 4.28 shows the performance when a single computer of the clusters located in NAIST and Osaka University performed wavelet analysis. The right graph shows the effect of the Grid computing for wavelet analysis. Since the manager process's total time varies over the time due to factors such as network parameters and the load status of the computers, the author has plotted the shortest, longest and average line regarding the total time of the manager process into the graph.

The right graph indicates that the total time consumed in the computational engine module was reduced regardless of the three lines in proportion to the increase of computers. Note that the total time with a single computer in the right graph was measured on the visualization computer. In particular, in the shortest total time line, the total time with 16 computers was reduced to 2.0692246 seconds while the total time was 21.540 seconds on the visualization computer. This fact indicates that the author has achieved approximately 11 times faster wavelet analysis with the 16 computers located in NAIST and Osaka University in comparison with the

use of the local visualization computers. In addition, these graphs indicate that we can gain more computational power through the integration of remote computers with the Grid than through the use of a local visualization computer.

4.5.6.3 The Overhead Time

In this section, the author considers the overhead time incurred by Grid computing based on the SPMD paradigm for wavelet analysis. For this purpose, the author first evaluates (2) the computational time for wavelet analysis, shown in section 4.5.6.2, in the manager process. Next, the author investigates the overhead time, that is, the data transfer time (1), (3) and (4) listed in section 4.5.6.2.

The left graph in Fig. 4.29 shows how the computational time for wavelet analysis decreases as the number of computers available for the computational engine module increases. The computational time was measured in the manager process. In order to compare the total time measured in section 4.5.6.2 with the computational time for wavelet analysis, the author has also plotted the shortest, longest and average lines regarding the computational time for wavelet analysis into the left graph in Fig. 4.29. Here, note that each computational time line corresponds to the total time line in the right graph in Fig. 4.28. The graph shows that all lines are plotted along almost the same orbit and that the computational time was reduced to 1.3147799 seconds with 16 computers, compared to 9.5646780 seconds with 2 computers.

In order to evaluate the improvement of the SPMD paradigm for wavelet analysis, the author has normalized the computational time for wavelet analysis by denoting 1.0 as the computational time measured when 16 computers were used. The right graph shows the result of the performance normalization. These graphs indicate that we have achieved approximately 7.3-fold higher computational performance through the use of 16 computers than through the use of 2 computers. Therefore, the SPMD paradigm appears highly effective for the wavelet analysis of the MEG data. In addition, these graphs indicate that the difference between the total time lines in the right graph in Fig. 4.28 is not caused from the computational time for wavelet analysis.

Next, the author evaluates the data transfer time necessary in the SPMD paradigm for wavelet analysis. In this measurement, the worker 1 entity distributed 64 KB MEG data to each entity in the computational engine module. After the computation, the manager entity gathered a total of 2.4 MB result data from each

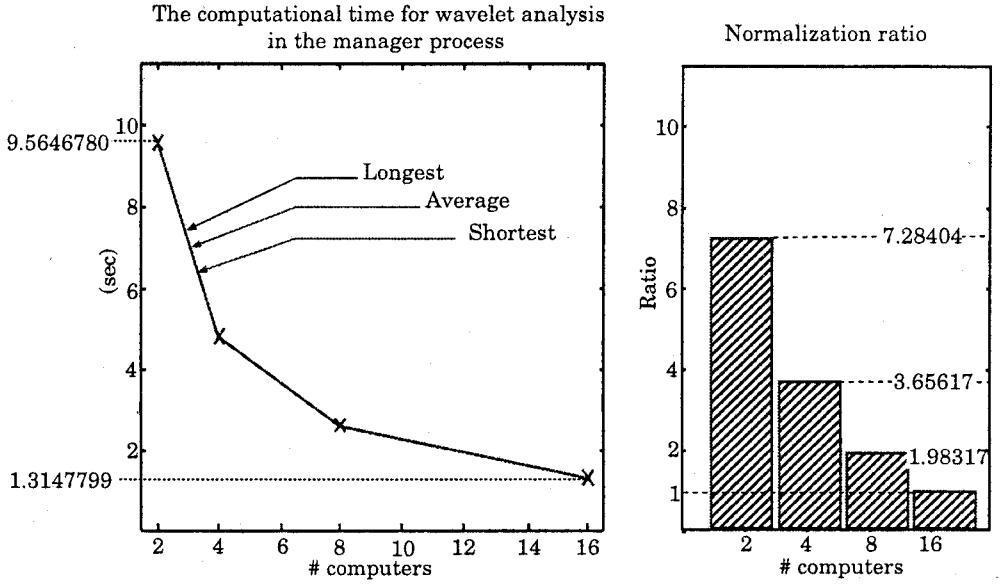


Figure 4.29: The computational time for wavelet analysis in the manager process

entity and then transfered the gathered result data to the communication agent. In the evaluation, the time required for the data transfer operation pertaining to this computational procedure was measured. The data transfer operation includes the time (1), (3), and (4), listed in section 4.2.

For the evaluation of the data transfer time, the author has plotted the time required for the data transfer operation into the left graph in Fig. 4.30. The worst, average and best lines in the left graph correspond to the longest, average and shortest lines in the right graph, respectively. In addition, these lines also correspond to the counterparts in the left graph in Fig. 4.28.

The left graph shows that the average data transfer was performed within less than 2.0 seconds and that the shortest data transfer time was performed within less than 1.1 seconds. On the other hand, the left graph shows that the longest data transfer time reached approximately 4.1 seconds when 4 computers were used.

In order to consider how this data transfer time affects the system performace, the author has shown the ratio of the data transfer time to the total time consumed in the computational engine module in the right graph. The ratio of the data transfer time to the total time consumed in the computational engine module gradually increases when accompanied with an increase in computers. In particular, the graph indicates that the system incurred 55.76 % data transfer time of total time when 16 computers were used. This fact means that the manager entity requires approx-

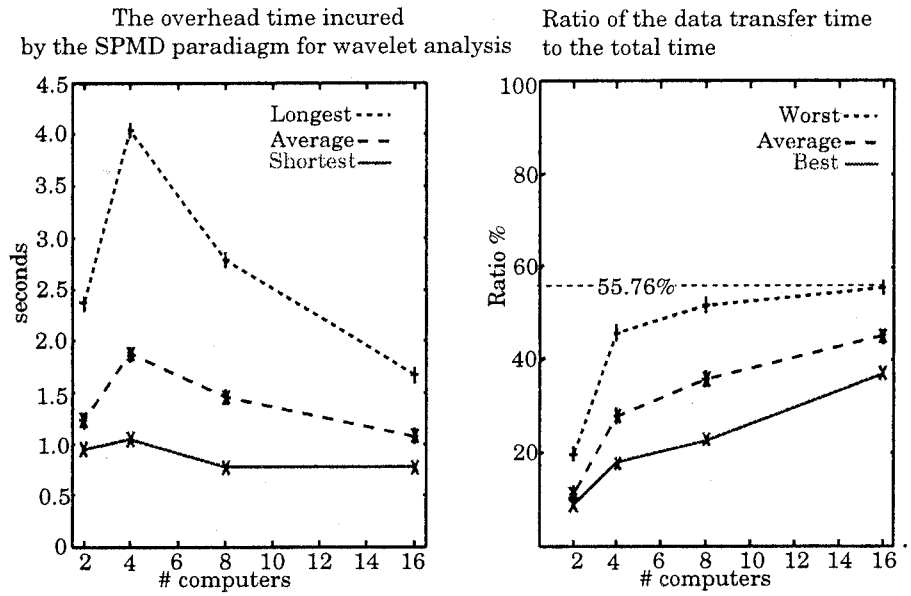


Figure 4.30: Data transfer time in the manager entity

imately twice as much time as the real computational time for wavelet analysis. However and also importantly, in this case, the total time consumed in the manager entity was 2.0692246 seconds, compared with 21.540 seconds when only the visualization computer was used. In other words, remarkably, approximately 10 times faster system performance was achieved.

4.5.6.4 Practical Diagnotic Viewpoint

Traditionally, doctors and researchers often analyzed MEG data on a single-processor. Analyzing MEG data acquired from 64 sensors takes a long time. The measurement result that the author has just summarized shows that it takes 21.540 seconds to analyze just 1-second epoch of MEG data. Taking into consideration the fact that the amount of MEG data to be analyzed often reaches more than one hour, investigating this amount of MEG data with wavelet analysis within a realistic time is impossible.

However, measurement results in Fig. 4.28 clarifies that the MEG data analysis system built over the network between NAIST and Osaka University allows doctors to simultaneously investigate 1-second epoch of the MEG data acquired from 64 sensors with wavelet analysis within approximately 2.1 seconds. Therefore, doctors can investigate the MEG data in an interactive manner without feeling frustrated. Moreover, the MEG data analysis system is capable of providing a transparency of

data location through the use of SFS and the conversion of the sfs filepath. This transparency allows doctors and researchers to seamlessly search for MEG data of interest without special data transfer operations such as FTPs and HTTPs. These features will dramatically improve the efficiency of doctors' diagnostic procedures.

4.6 Concluding Remarks

In this chapter, the author has described MEG data analysis systems developed with three different programming models. First, the MEG data analysis system built with the programming model of Globus APIs was described. In the MEG data analysis system, the functionality of the Nexus RSR, which the Globus provides, has been utilized as a building block. The Nexus RSR offers users a method of generating processes on a remote computer, even if the computer is located on different administrative domains. The MEG data analysis system built with the supercomputer HP-UX Exemplar showed that the system has the capabilities of dramatically reducing the computation time for wavelet cross-correlation analysis by seamlessly integrating data acquisition, data analysis, and visualization. Next, the MEG data analysis system built with the programming model with the MPICH-G APIs was described. This system also showed the capabilities of reducing the computational time and seamlessly integrating three parts essential for the analysis of brain functions.

In addition, based on the actual building of an application system on the Grid, the author proposed a new programming model: *the hybrid programming model.* The hybrid programming model provides a user with a programming template. Because the programming template is designed based on the data flow generally occurring in the Grid application, users can develop their own Grid applications easily by using the template. Also, the template never robs users of the flexibility of programming. Furthermore, the evaluation in this chapter showed that the MEG data analysis system with the hybrid programming model can reduce computational time. This reduction in computational time gives practitioners (doctors and researchers) an effective system to aid in the advancement of the brain sciences.

Chapter 5

Conclusion

5.1 Concluding Remarks

The Grid has increasingly attracted attention from scientists and researchers in many fields since it appeared in the middle of the 1990s. This new technology is considered to be an essential tool for research in almost all areas of science. This is particularly evident in those fields frequently requiring the process of large amounts of data, where the demands on new applications of the Grid increase rapidly.

In this dissertation, the author described the application of the Grid computing to an actual medical data analysis, MEG data analysis. Through building the MEG data analysis system on the Grid, the author believes that the Grid is effective and useful for actual scientific problems, especially for sciences that require heavy computational time. Most importantly, the MEG data analysis systems built in this research illustrated that the system built on the Grid has the potential of solving geographical distribution problems. In other words, the Grid has the potential of integrating knowledge and technologies, which in reality are often dispersed. Finally, the author believes that the Grid is essential in the further development of various fields of science and medicine and the author hopes that this research will become a forerunner for these fields.

Currently, brain science is one of the most important research fields in the 21st century. The brain is the most complex organ in the human body and it remains largely elusive despite enormous advances in science and technology. Nonetheless, rapid developments in imaging technologies enable brain functions to be revealed more accurately than ever before and promise better diagnosis and treatments for many neurological disorders. However, raw data, which needs to be processed by

computers have considerably increased both in amount and in size, and have reached the limit of existing computing technology.

The present research has investigated the application of the Grid in the brain sciences. The primary goals of this dissertation are to support and contribute to advances in brain sciences and to testify to the effectiveness and usefulness of Grid computing to the brain sciences. To achieve these goals, the author has applied the Grid to an actual scientific problem, MEG data analysis. More specifically, the development of an MEG data analysis system enabling doctors and researchers to efficiently perform analysis of brain function was explored in this dissertation. To embody such an MEG data analysis system, the author has approached the development of the MEG data analysis system from two different aspects. First, the author discussed the Grid as a building block for a medical infrastructure. In this approach, the fundamental problems in Grid computing were targeted. Next, the author discussed the Grid as a software technology for building a specific application system. In this approach, problems specific to brain data analysis including MEG data analysis were solved. In this dissertation, problems in the first approach were described in Chapter 2, and problems in the second approach were solved in Chapter 3 and 4. This dissertation, therefore, discussed the following issues, problems and solutions in the following chapters:

In Chapter 1, motivations for this research were described from both the medical and engineering points of view. This chapter verified that the application of the Grid to scientific problems is now demanded.

In Chapter 2, an infrastructure with Globus grid toolkit for medical data analyses was described. First, the author described the security problems that arise when doctors and researchers transfer any medical data, especially medical data which specifies a patient's name on a public network such as the Internet. To solve these problems, Internet Security Protocol (IPSec) was integrated into the GSI provided by the Globus. To further enhance the security level in the infrastructure, a private CA was established and integrated into the GSI mechanism. In this dissertation the author also illustrated how IPSec is useful for assuring the confidentiality and integrity of medical data transferred on the Internet. In addition, the Internet Protocol version 6 (IPv6), which defines the use of IPSec protocol, was integrated into the Globus in order to address the shortage problem of IP addresses in near term world-wide global computing. Finally, in order to offer doctors a single disk image from the data structure on the Grid, the author has developed a data access

structure with location-transparency, by using Self-Certifying File System.

In Chapter 3, the author targeted MEG data analysis as an actual scientific problem and detailed not only what the MEG is, but also what MEG data analysis is and then explained wavelet cross-correlation analysis. Next, problems in MEG data analysis, which prevent doctors from efficiently diagnosing and analyzing brain functions, were considered. As a result, to realize efficient analytic and diagnostic situations, the author identified and clarified the following issues:

1. Seamless integration of the three stages essential for the analyses of brain functions,

2. Dramatic reduction of the computational time for wavelet cross-correlation analysis, and

3. The development of a visualization software that provides doctors and scientists with an intuitive understanding of analysis results.

Subsequently, to achieve the second issue, efficient and effective parallel processing methods for wavelet cross-correlation analysis were explored through a consideration of how wavelet cross-correlation analysis is performed by doctors and researchers. Finally, to achieve the third issue, a visualization software composed of the following main components: a raw data viewer, a wavelet viewer, and a three-dimensional visualization tool was developed. This visualization software is capable of providing doctors and researchers with an intuitive understanding of complex analysis results.

In Chapter 4, the author has built three MEG data analysis systems. The first two systems were designed and implemented on Grid environments simulated on the LAN in the Cybermedia Center, Osaka University, Japan. The first system was built on the simulated Grid environment. It was composed of a commercial personal computer and a supercomputer, the HP Exemplar V2200/N, which was developed with the Globus programming model. The second system was also built on the simulated Grid environment. This system was composed of a commercial personal computer and a Linux box with seven computers, which was developed with the MPICH-G programming model.

Through the building of these two systems, the author has illustrated how the Globus programming model provides doctors and researchers with programming flexibility, although programming difficulty becomes a serious problem in the Globus programming model. Also, conversely, the author has illustrated that the MPICH-G

programming model provides doctors and researchers with programming familiarity, although the MPICH-G programming model robs doctors and researchers of programming flexibility. To balance this trade-off between flexibility and familiarity, the author proposed a new programming model, or the hybrid programming model. The hybrid programming model provides a programming template which allows doctors and researchers to write their own application programming codes with ease. The programming template was designed based on the data flow in the Grid application and provides doctors and researchers with as much programming flexibility as possible. By following the programming template, doctors and researchers can develop their own application system with either MPICH-G APIs or Globus APIs. In short, the hybrid programming model allows doctors and researchers to write and reuse their own application programming codes with familiar MPICH-G APIs. To verify the effectiveness and usefulness of the hybrid programming model, the author has developed the third MEG data analysis system with the hybrid programming model. Through the development of the MEG data analysis system, this hybrid programming model was shown to have the ability to help users develop their own applications on the Grid more easily than the existent two programming models: the MPICH-G programming model and Globus programming model.

5.2 Future Directions

The author believes the following issues remain to be answered in future research on the Grid. At the writing of this dissertation, the following issues are being investigated by the author.

(1) Quality of Service

The evaluation in this research showed MEG data analysis systems have the potential to dramatically reduce computational time for MEG data analysis with the wavelet cross-correlation analysis method. However, the evaluation also indicated that the data transfer time heavily affected computational performance. From this point of view, a mechanism that can control network parameters, known as, Quality of Service (QoS) technology is essential for the construction of further effective MEG data analysis system.

(2) Load Balancing for Grid Computing

A load balancing mechanism that can integrate a variety of existent scheduling systems such as LSF, PBS [48], and Condor [49] is essential. In this research, LSF was utilized for balancing the computational workload among a supercomputer composed of two nodes. The evaluation showed that these scheduling systems worked well on the supercomputer. However, this research does not consider the simultaneous use of these scheduling systems yet. In this case, balancing the computational workload will be a very difficult problem.

At present, the Globus grid toolkit exploited in this research enables a user to submit jobs to light-weighted computers via an existent scheduling system such as LSF, PBS and Condor. However, the interoperability among multiple such existent scheduling systems has not been considered yet. If two different institutions administer different scheduling systems for load balancing, users need to submit their jobs after considering the difference of the scheduling systems. This way of submitting jobs is troublesome and not realistic. Therefore, the development of scheduling system on the Grid that can optimize load balancing among the existent scheduling systems is indispensable.

(3) Utilization of Grid Information

Finally, a mechanism that can utilize the information of the Grid environment efficiently is essential. As described in this dissertation, the Globus grid toolkit provides MDS as an information service for users. The MDS allows users to easily retrieve status information of computational resources with APIs. Such information includes computer architecture, CPU load average, and operating systems. However, making maximum use of such information is difficult. For example, if users have a computer with a sigle Pentium III 1 GHz processor and a computer with two AMD Athlon 500 MHz processors, and if the load average of these two computers are both 1.80, then how should users decide which computer provides better computational performance? At the time of writing this dissertation, users still have much difficulty in understanding how information provided by MDS should be treated. Therefore, a semi-automatic mechanism that provides clues for users for the utilization of the Grid information is demanded. These are difficult concepts.

Moreover, such a semi-automatic mechanism is essential for developing a scheduling system on the Grid environment that optimizes load balancing among existent

scheduling systems. Also, the mechanism is useful for the development of QoS technology. If such a scheduling system with a semi-automatic mechanism for utilization of Grid information can be realized in the future, computational performance should dramatically improve.

Acknowledgements

I would like to express my appreciation to Professor Shinji Shimojo of Osaka University, my supervisor, for his countless suggestions and continuous support. His advice was at all times and in all aspects invaluable. Without him, I would not be able to complete this research. I am also heartily grateful to Professor Shojiro Nishio and Professor Kenzo Akazawa of Osaka University for serving as referees on my dissertation committee. Their expertise and insightful comments have been most beneficial.

Deep gratitude is also due to Professor Koso Murakami, Professor Isao Shirakawa, Professor Hiromu Fujioka, and Professor Norihisa Komoda, Department of Information Systems Engineering, Graduate School of Osaka University, for serving as members on my dissertation committee.

I would also like to express my deep gratitude to Associate Professor Youki Kadobayashi of the Nara Institute of Science and Technology for his countless suggestions and continuous support. His technical advice and support helped to accelerate my research. He is ultimo Thule for me.

I also would like to express my gratitude to my advisor, Associate Professor Yuko Mizuno-Matsumoto, of Osaka Jonan Women's College, for her countless hours of advice and sincere support. Her medical background opened the door to my research. Her mental support as a psychiatrist always motivated me to work harder. Without her mental support and advice to my research, this dissertation would not have been written.

I am also heartily grateful to Associate Professor Ken-ichi Baba, Assistant Professor Hiroki Nogawa, Assitant Professor Kaname Harumoto, Research Associate Toyokazu Akiyama, and Research Associate Yutaka Nakamura for hours of countless technical advice and help with my research. I would like to express my gratitude to Shimojo Laboratory secretary Ms. Kae Morita and Ms. Hiroko Tada, and Technical Staff. Ms. Fumi Takahashi for supporting my research. I would also like to thank

the alumni and the students in Shimojo Lab: Mr. Masahiro Fujisawa, Mr. Makoto Kitani, Mr. Satoshi Kamio, Mr. Hikaru Yagi, Mr. Yuji Hosokawa, Mr. Kengo Hara, Mr. Koji Kobayashi, Mr. Takeshi Kaishima, Mr. Weigeng Zhao, Mr. Tomohito Sakai, Mr. Hongyu Shi, and Mr. Shoichi Motohisa.

I also would like to express my sincerely appreciation to Associate Professor. Francis Lee, and Dr. Bertil Schmidt, of the Nanyang Technological University, Singapore, as well as Mr. Uvaraj Periyathamby, of the Institute of High Performance Computing, Singapore as well as the staff of the Singapore Grid team.

In addition, I would like to show appreciation to Professor Shin-ichi Tamura, and Associate Professor Yoshinobu Sato, of the Graduate School of Medicine, Osaka University, as well as Mr. Yuji Yabuchi, KDDI, and Dr. Jian Chen. And, I would like to express my gratitude to my seniors Dr. Tetsuya Kobota and Dr. Hisashi Taketani. Without their support, this research would never be achieved.

I would like to express my gratitude to my friends: Mr. Shuji Wakita, Mr. Kei Nakahodo, Mr. Hidenori Shikata, Mr. Hiraku Tuchiya, and Ms. Miho Ikai for encouraging me.

My deepest gratitude also goes to my father and mother Tsuyoshi and Yuriko for their economic support and sincere encouragement. And I also would like to say "thank you very much" to my younger brother Masashi for his understanding of my situation. Finally, I would like to devote this dissertation to my uncle Tsutomu Date who died due to a sudden marine accident on 10th, August, 2001. I would like to say to him, "I appreciate you very much for your sincere encouragement. Without your encouragement, this dissertation would not have appeared.".

References

[1] SETI@home, *http://setiathome.ssl.berkley.edu.*

[2] Folding@home, *http://folding.standord.edu.*

[3] AIDS@home, *http://www.fightaidsathome.org.*

[4] Foster, I., Kesselman, C., and Tuecke, S.: "The anatomy of the Grid: Enabling scalable virtual organizations", *Intl. J. Supercomputer Applications*, Vol. 15, No. 3, (printing), 2001.

[5] Louys, M., Starck, J. L., Mei, S., Bonnarel, F., and Murtagh F.: "Astronomical image compression", *Astronomy and Astrophysics*, Suppl., Ser. 136, pp. 579–590, 1999.

[6] The Globus Project, *http://www.globus.org.*

[7] Foster, I., and Kesselman, C.: "Globus: A metacomputing infrastructure toolkit", *Intl. J. Supercomputer Applications*, Vol. 2, pp. 115–128, 1997.

[8] Grimshaw, A., Ferrari, A., Knabe, F., and Humphrey, M.: "Legion: An operating system for wide-area computing", *Technical Report CS-99-12*, Department of Computer Science, University of Virginia, 1999.

[9] Netsolve, *http://icl.cs.utk.edu/netsolve.*

[10] Casanova, H., and Dongarra, J.: "NetSolve: A network server for solving computational science problems", *Intl. J. Supercomputer Applications and High Performance Computing*, Vol. 11, No. 3, pp. 212–223, 1997.

[11] Foster, I., and Kesselman, C.: "The Globus Project: A status report", *Proc. IPPS/SPDP'98 Heterogeneous Computing Workshop*, pp. 4–18, Mar. 1998.

[12] Foster, I., Kesselman, C., Tsudik, G., and Tuecke, S.: "A security architecture for computational Grids", *Proc. 5th ACM Conf. on Computer and Communications Security*, pp. 83–92, Nov. 1998.

[13] Butler, R., Engert, D., Foster, I., Kesselman, C., Tuecke, S., Volmer, J., and Welch, V.: "A national-scale authentication infrastructure", *IEEE Computer*, Vol. 33, No. 12, pp. 60–66, 2000.

[14] Fitzgerald, S., Foster, I., Kesselman, C., Laszewski, G. von, Smith, W., and Tuecke, S.: "A directory service for configuring high-performance distributed computations", *Proc. 6th IEEE Symp. on High-Performance Distributed Computing*, pp. 365–375, Aug. 1997.

[15] Bester, J., Foster, I., Kesselman, C., Tedesco, J., and Tuecke, S.: "GASS: A data movement and access service for wide area computing systems", *Proc. 6th Workshop on I/O in Parallel and Distributed Systems*, May 1999.

[16] Foster, I., Kohr, D., Krishnaiyer, R., and Mogill, J.: "Remote I/O: Fast access to distant storage", *Proc. Workshop on I/O in Parallel and Distributed Systems (IOPADS)*, pp. 14–25, Nov. 1997.

[17] Foster, I., Kesselman, C., and Tuecke, S.: "Nexus: An interoperability toolkit for parallel and distributed computer systems", *Technical Report ANL/MCS-TM-189*, Argonne National Laboratory, 1994.

[18] OpenSSL, *http://www.openssl.org*.

[19] Kent, S., and Atkinson, R.: "Security architecture for the Internet Protocol", *RFC 2409*, IETF Network Working Group, 1998.

[20] KAME project, *http://www.kame.net*.

[21] Harkins, D., and Carrel, D.: "The Internet Key Exchange (IKE)" , *RFC 2409*, IETF Network Working Group, 1998.

[22] Self-certifying File System, *http://www.fs.net*.

[23] Mazieres, D., Kaminsky, M., Kaashoek, M. F., and Witchel, E.: "Separating key management from file system security", *Proc. 17th ADM Symposium on Operating Systems Principles (SOSP'99)*, pp. 124–139, Dec. 1999.

[24] Mazieres, D.: "Self-certifying File System", *Ph. D Thesis*, MIT, 2000.

[25] Sato, S.: "Magnetoencephalography", Raven Press New York, *Advances in Neurology*, Vol. 54, 1990.

[26] Phillips, J. W., Leahy, R. M., Mosher, J. C., and Timsari, B.: "Imaging neural activity using MEG and EEG", *IEEE Engineering in Medicine and Biology Magazine*, Vol. 16, No. 3, pp. 34–42, 1997.

[27] Neuromag, *http://www.neuromag.com.*

[28] Li, H., and Nozaki, T.: "Wavelet Cross-Correlation Analysis (Analysis of Vortical Structures in a Turbulent Plane Jet)", *JSME Intl. J.*, Ser. B, Vol. 40, No. 1, pp. 58–66, 1997.

[29] Mizuno-Matsumoto, Y., Tamura, S., Sato, Y., Zoroofi, R. A., Yoshimine, T., Kato, A., Taniguchi, M., Takeda, M., Inouye, T., Tatsumi, H., Shimojo, S., and Miyahara, H.: "Propagating process of epileptiform discharges using waveletcrosscorrelation analysis in MEG", Tohoku University Press, *Advances in Biomagnetism*, pp. 782–785, 1999.

[30] Kikuchi, H., Nakashizuka, M., Watanabe, H., Watanabe, S., and Tomisawa, F.: "Fast wavelet transform and its application to detecting detonation", *IEICE Trans. Fundamentals.*, Vol. E75-A, No. 8, pp. 980-987, 1992.

[31] Mizuno-Matsumoto, Y., Okazaki, K., Kato, A., Yoshimine, T., Sato, Y., Tamura, S., and Hayakawa, T.: "Visualization of epileptogenic phenomena using crosscorrelation analysis: Localization of epileptic foci and propagation of epileptiform discharges", *IEEE Trans. Biomed. Eng.*, Vol. 46, No. 3, pp. 271–279, 1999.

[32] Mizuno-Matsumoto, Y., Date, S., Tamura, S., Sato, Y., Zoroofi, R. A., Tabuchi, Y., Shimojo, S., Kadobayashi, Y., Tatsumi, H., Nogawa, H., Shinosaki, K., Takeda, M., Inouye, K. and Miyahara, H.: "Integration of signal processing and medical image for evaluation of brain function on Globus", *Proceeding of Internet Workshop'99 (IWS'99)*, pp. 241–246, Feb. 1999.

[33] OpenGL High Performance 2D/3D Graphics, *http://www.opengl.com.*

[34] Message Passing Interface Forum, *http://www.mpi-forum.org.*

[35] CTF. *http://www.ctf.com.*

[36] Foster, I., Kesselman, C., and Tuecke, S.: "The Nexus task-parallel runtime system", Tata McGraw Hill, *Proc. 1st Intl. Workshop on Parallel Processing*, pp. 457–462, 1994.

[37] Geist, G. A., and Kohl, J. A.: "PVM and MPI: a comparison of features", *Calculateurs Paralleles*, Vol. 8, No. 2, 1996.

[38] Foster, I., and Karonis, N.: "A Grid-enabled MPI: Message passing in heterogeneous distributed computing systems", *Proceeding CDROM of Supercomputing 98*, Nov. 1998.

[39] Foster, I., Geisler, J., Gropp, W., Karonis, N., Lusk, E., Thiruvathukal, G., and Tuecke, S.: "Wide-area implementation of the Message Passing Interface", *Parallel Computing*, Vol. 24, No. 12, pp. 1735–1749, 1998.

[40] Gropp, W., and Lusk, E.: "A high-performance, portable implementation of the MPI message passing interface standard", *Parallel Computing*, Vol. 22, No. 6, pp. 789–828, 1996.

[41] Gropp, W., and Lusk, W.: "User's Guide for MPE: Extensions for MPI Programs", *http://www-unix.mcs.anl.gov/perfvis/download*

[42] Vigario, R., Sarela, J., Jousmaki, V., Hamalainen, M., and Oja, E.: "Independent component approach to the analysis of EEG and MEG recordings", *IEEE Trans Biomed Eng2000*, Vol. 47, No. 5, pp. 589–593, 2000.

[43] McKendall, R., and Bajcsy, R.: "Scalable parallel computing for real-time telepresence in medical imaging: Preliminary demonstration", *Proceedings of Supercomputing'96*, Nov. 1996.

[44] MPICH-G2, *http://www.hpclab.niu.edu/mpi.*

[45] Czajkowski, K., Foster, I., Karonis, N., Kesselman, C., Martin, S., Smith, W., and Tuecke, S.: "A resource management architecture for metacomputing systems", *Proc. IPPS/SPDP '98 Workshop on Job Scheduling Strategies for Parallel Processing*, Mar. 1998.

[46] Zaki, O., Lusk, E., Glopp, W., and Swider, D.: "Toward scalable performance visualization with jumpshot", *High-Performance Computing Applications*, Vol. 13, No. 2, pp. 277–288, 1999.

[47] WIDE project, *http://www.wide.ad.jp.*

[48] Portable Batch System, *http://www.openpbs.org.*

[49] Condor Project, *http://www.cs.wisc.edu/condor.*

List of Publications by the Author

A. Journal Papers

1. Date, S., Mizuno-Matsumoto, Y., Kadobayashi, Y., and Shimojo, S.: "An MEG data analysis system using Grid technology", *Journal of Information Processing Society of Japan*, Vol. 42, No. 12, pp. 2952–2962, Dec. 2001.

2. Mizuno-Matsumoto, Y., Ishijima, M., Shinosaki, K., Nishikawa, T., Ukai, S., Ikejiri, Y., Nakagawa, Y., Ishii, R., Tokunaga, H., Tamura, S., Date, S., Inouye, T., Shimojo, S., and Takeda, M.: "Transient Global Amnesia (TGA) in an MEG study", *Brain Topography*, Vol. 13, No. 4, pp. 269–274, Nov. 2001.

3. Mizuno-Matsumoto, Y., Yoshimine, T., Nii, Y., Kato, A., Taniguchi, M., Lee, J. K., Ko, T.S., Date, S., Tamura, S., Shimojo, S., Shinosaki, K., Inouye, T., and Takeda, M.: "Landau-Kleffner syndrome: localization of epileptogenic lesion using wavelet-crosscorrelation analysis", *Epilepsy and Behavior*, Vol. 2, No. 3, pp. 288–294, Jun. 2001.

4. Mizuno-Matsumoto, Y., Date, S., Tabuchi, Y., Tamura, S., Sato, Y., Zoroofi, R. A., Shimojo, S., Kadobayashi, Y., Tatsumi, H., Nogawa, H., Shinosaki, K., Takeda, M., Inouye, T., and Miyahara, H.: "Telemedicine for evaluation of brain function by a metacomputer", *IEEE Transactions of Information Technology in Biomedicine*, Vol. 4, No. 2, pp. 165–172, Jun. 2000.

5. Mizuno-Matsumoto, Y., Tamura, S., Sato, Y., Zoroofi, R. A., Date, S., Tabuchi, Y., Shinosaki, K., Ukai, S., Ishii, R., Inouye, K., Tatsumi, H., Kadobayashi, Y., Shimojo, S., Takeda M., and Miyahara, H.: "Propagation of epileptiform discharges using wavelet-crosscorrelation analysis in MEG", *Medical Imaging Technology*, Vol. 18, No. 1, pp. 61–70, Jan. 2000.

6. Date, S., Mizuno-Matsumoto, Y., Tamura, S., Sato, Y., Zoroofi, R. A., Tabuchi, Y., Shimojo, S., Kadobayashi, Y., Tatsumi, H., Nogawa, H., Shinosaki, K., Takeda, M., Inouye, K., and Miyahara, H.: "Metacomputing environment for magnetoencephalography", *Medical Imaging Technology*, Vol. 18, No. 1, pp. 47–59, Jan. 2000.

B. International Conference Papers

1. Mizuno-Matsumoto, Y., Date, S., Kaishima, T., Hara, K., Shinosaki, K., Tamura, S., and Shimojo, S.: " The development of a high performance computing for MEG data using an Independent Component Analysis on Globus", *The International Conference on Bioinformatics 2002*, Feb. 2002 (in press).

2. Date, S., Lee, B. S., Shimojo, S., Prekumar, P. A., Kadobayashi, Y., Bertil, S., and Uvaraj, P.: "MEG image analysis and visualization", *Proceedings of The Seminar on Integrated Engineering*, pp. 365–370, Dec. 2000.

3. Mizuno-Matsumoto, Y., Lesser, R. P., Webber, R. S., Motamedi, G. K., Shimojo, S., Kadobayashi, Y., Date, S., Tamura, S., Shinosaki, K., and Takeda, M.: "Wavelet-crosscorrelation maping can reveal how the brain conditions change with brief pulses of stimulation", *54th Annual Meeting of the American Epilepsy Society (AES) Epilepsia*, Vol. 14, Suppl. 7, p. 211, Dec. 2000.

4. Mizuno-Matsumoto, Y., Date, S., Kadobayashi, Y., Shimojo, S., Tatsumi, H., Shinosaki, K., Takeda, M., Inouye, K., Tamura, S., and Miyahara, H.: "Evaluation system for brain function using MEG, Internet and high performance computing", *Proceedings of The 12th International Conference on Biomagnetism (Biomag2000)*, p. 206, Aug. 2000.

5. Date, S., Mizuno-Matsumoto, Y., Shinosaki, K., Tamura, S., Kadobayashi, Y., and Shimojo, S.: "Diagnosis supporting system for MEG using two-stage parallel processing on computational grids", *Proceeding CDROM of INET 2000*, Jul. 2000.

6. Mizuno-Matsumoto, Y., Date, S., Tamura, S., Sato, Y., Zoroofi, R. A., Tabuchi, Y., Shimojo, S., Kadobayashi, Y., Tatsumi, H., Nogawa, H., Shinosaki, K., Takeda, M., Inouye, T. and Miyahara, H.: "Integration of signal processing

and medical image for evaluation of brain function on Globus", *Proceedings of Internet Workshop '99 (IWS99)*, pp. 297–302, Feb. 1999.

C. Domestic Conference Papers

1. Kaishima, T., Mizuno-Matsumoto, Y., Date, S., Shinosaki, K., and Shimojo, S.: "Development of fast analysis system for MEG data using Independent Component Analysis", *Technical Report of IEICE*, MI2001–37, pp. 13–20, Sep. 2001.

2. Mizuno-Matsumoto, Y., Lesser, R. P., Webber, R. S., Motamedi, G. K., Shinosaki, K., Ukai, S., Ishii, R., Inouye, K., Date, S., Shimojo, S., and Takeda, M.: "Quantifying the changes of ECoG from the patients with Epilepsy using the electrical stimulation", *The 97th Annual Conference of The Japanese Society of Psychiatry and Neurology*, May. 2001.

3. Hara, K., Mizuno-Matsumoto, Y., Date, S., Nogawa, H., Shinosaki, K., and Shimojo, S.: "Development of MEG Data Analysis System by Independent Component Analysis using Parallel Computing", *Technical Report of IEICE*, MI2000–72, pp. 127–132, Jan. 2001.

4. Mizuno-Matsumoto, Y., Lesser, R. P., Webber, R. S., Motamedi, G. K., Inouye, K., Shinosaki, K., Ukai, S., Ishii, R., Date, S., Shimojo, S., Tamura, S., and Takeda, M.: "Wavelet cross-correlation analysis can predict whether BPS will stop AD of ECoG", *The 30th Annual Congress of the Japanese Society of Clinical Neurophysiology*, Dec. 2000.

5. Date, S., Kadobayashi, Y., Kamio, S., Mizuno-Matsumoto, Y., and Shimojo, S.: "Problems in Implementing Distributed Parallel Computing", *The 3rd Workshop on Internet Technology of Japan Society for Software and Technology (WIT2000)*, Sep. 2000.

6. Date, S., Kadobayashi, Y., Shimojo, S., Tamura, S., Mizuno-Matsumoto, Y.: "MEG Data Analysis System on Metacomputing Environment", *Technical Report of IEICE*, MI99–42, pp. 73–78, Nov. 1999.

7. Tabuchi, Y., Mizuno-Matsumoto, Y., Tamura, S., Sato, Y., Date, S., Takeda, M., Shinosaki, K., Tatsumi, H., Shimojo, S., Kadobayashi, Y., Miyahara, H.:

"The development of visualization system for evaluating the brain function on Globus using high-speed network and parallel computing", *Proc. The Conference of Japanese Society of Medical Imaging Technology (JAMIT Frontier '99)*, pp. 213–218, Jan. 1999.

8. Date, S., Mizuno-Matsumoto, Y., Shimojo, S., Kadobayashi, Y., Tabuchi, Y., Takeda, M., Sato, Y., Tamura, S., Nogawa, H., Tatsumi, H., "The Acceleration of Telemedicine with a Highly Sophisticated Medical Device and Globus", *ITRC Technical Report*, No. 3, pp. 85–90, Nov. 1998.

D. Invited Talks

1. Date, S., and Mizuno-Matsumoto, Y.: "Telemedicine work bench for evaluation of brain function on GRID: I. Engineering Aspect", *Seminar on Grid: The infrastructure on the Internet*, Singpore, Nov. 2001.

2. Mizuno-Matsumoto, Y. and Date, S.: "Telemedicine work bench for evaluation of brain function on GRID: II. Medical Aspect", *Seminar on Grid: The infrastructure on the Internet*, Singapore, Nov. 2001.

3. Date, S.: "Application of Grid Technology to Medical Data Analysis", *The 3rd Workshop on Large Data Management for Creative Research*, Nara, Oct. 2001.

4. Date, S.: "The Advance in Abroad Grid Activity", *The Second Workshop of BioGrid*, Osaka, Sep. 2001.

E. Magazines, etc

1. Kadobayashi, Y., and Date, S.: "Global Computing (6)", SAIENSU-SHA Co., Ltd. Publishers, *Computer Today*, No. 102, pp. 56–62, Mar. 2001.

2. Mizuno-Matsumoto, Y., Date, S., Shinosaki, K., and Shimojo, S.: "The development of a diagnosis system for brain disease on telemedicine", *The Bulletin of Osaka Jonan Women's College*, Vol. 35, pp. 59–64, Feb. 2001.

3. Shimojo, S., Youki, K., Date, S., and Tamura, S.: "Next-Generation Internet and Medicine", Electronic Publishing Company Japan, *Medical Care and Computer*, Vol. 10, No. 10, pp. 19–23, Nov. 1999.