물리탐사
Vol. 3, No. 2, 2000, p. 53~60

네트워크에 기반한 MT자료의 처리기술 개발 연구

이희순[1] · 권병두[2] · 정호준[3] · 오석훈[2]

Development of Network Based MT Data Processing System

Heuisoon Lee, Byung-Doo Kwon, Hojoon Chung and Seokhoon Oh

요 약 : 본 연구에서는 급격히 증가하는 인터넷망 및 분산(distribution) 컴퓨팅 환경을 이용한 서버/클라이언트(server/client) MT 자료 처리 시스템의 구축을 위한 여러 가지 기술적 사항에 대해 논의하였다. 이러한 시스템은 표준적인 처리 방식의 도입과 인증된 자료 처리 서버에서의 해석 수행을 통해 일관성과 안정성을 동시에 제공할 수 있을 것이다. 또한 인터넷망을 이용하여 현장에서의 자료해석이 가능해지므로 탐사 시간, 경비의 감소 및 추가 탐사 계획 수립에도 도움을 줄 것이다. 각종 자바 기술(pure java와 enterprised java)은 네트워크 프로그램을 손쉽게 개발할 수 있는 많은 방법들을 제공한다. 본 연구에서는 이를 이용하여 웹(web)에 의한 서버/클라이언트 모델과, 소켓(Socket) 및 원격 함수 호출(RMI: Remote Method Invocation) 에 의한 처리 기법을 MT 자료의 해석에 적용하기 위한 방법에 대해 논하였다. 또한 MT 자료의 특성상, 그 해석은 고성능의 컴퓨터를 이용하였을 때에도 상당한 시간을 필요로 하므로 이를 극복하기 위해 서버 프로그램에 MPI(Message Passing Interface) 병렬처리 기술을 적용하고자 한다. 이는 고가의 병렬 처리 컴퓨터를 대체할 수 있으며, 표준적인 코딩이 제시되었으므로 관리 및 유지, 보수에 있어 효율성을 제공할 것이다.

Abstract : The server/client systems using the web protocol and distribution computing environment by network was applied to the MT data processing based on the Java technology. Using this network based system, users can get consistent and stable results because the system has standard analysing methods and has been tested from many users through the internet. Users can check the MT data processing at any time and get results during exploration to reduce the exploration time and money. The pure/enterprised Java technology provides facilities to develop the network based MT data processing system. Web based socket communication and RMI technology are tested respectively to produce the effective and practical client application. Intrinsically, the interpretation of MT data performing the inversion and data process requires heavy computational ability. Therefore we adopt the MPI parallel processing technique to fit the desire of in situ users and expect the effectiveness for the control and upgrade of programing codes.

Keywords : 네트워크, MT 자료처리, Java, MPI

서 론

지난 몇 년간 인터넷과 웹(WWW)의 확산은 정보의 교환 및 공유에 있어 중요한 저반 시설(infrastructure)로서 자리 잡았다. 이에 동반하여 인터넷을 이용하여 여러 과학 분야의 자료를 다루고, 주고 받으며, 처리하는 각종 연구들이 시작되고 있으며(Lin *et al.*, 1999; Templeton and Gough, 1999; Huang and Lin, 1999) 이를 지원하기 위한 다양한 컴퓨팅 도구들도 제공되고 있다(Carey and Bell, 1997; Hibbard, 1998). 하지만 대개의 관련 연구들이 관측 자료의 실시간 도시(visualization)에 의한 자료의 공유나 지리 정보 자료(GIS)의 복합 도시 등과 같은 자료의 표현(presentation)에 그치고 있어, 인터넷의 기능을 제대로 이용하지 못하고 있다.

인터넷, 즉 네트워크에 연동되어 있는 컴퓨터들은 쓰기에 따라, 각기 독립적으로 행동할 수도 있고, 함께 힘을 모아 작업에 투입(parallel computing)될 수도 있으며, 필요에 따라 자기에게 부여된 작업을 수행(distribution computing)할 수도 있다. 이는 지구물리 자료의 처리와 같은 고비용과 많은 시간의 컴퓨팅을 요구하는 경우에 효율적으로 적용될 수 있다. 2, 3차원 역산과 같은 많은 컴퓨팅시간을 요하는 문제의 해결을 위해 최적으로 인터넷을 활용하는 방안의 한 가지는, 탐사 현장에서 노트북을 이용하여 작성된 야외 자료를 디지틀 전화기를 통해

*2000년 5월 16일 접수
1) 인천교육대학교 과학교육과 (Dept. of Science Education, Inchon National University of Education)
2) 서울대학교 지구과학교육과 (Dept. of Earth Science Education, Seoul National University)
3) 한국자원연구소 (Korea Institute of Geology, Mining and Materials)

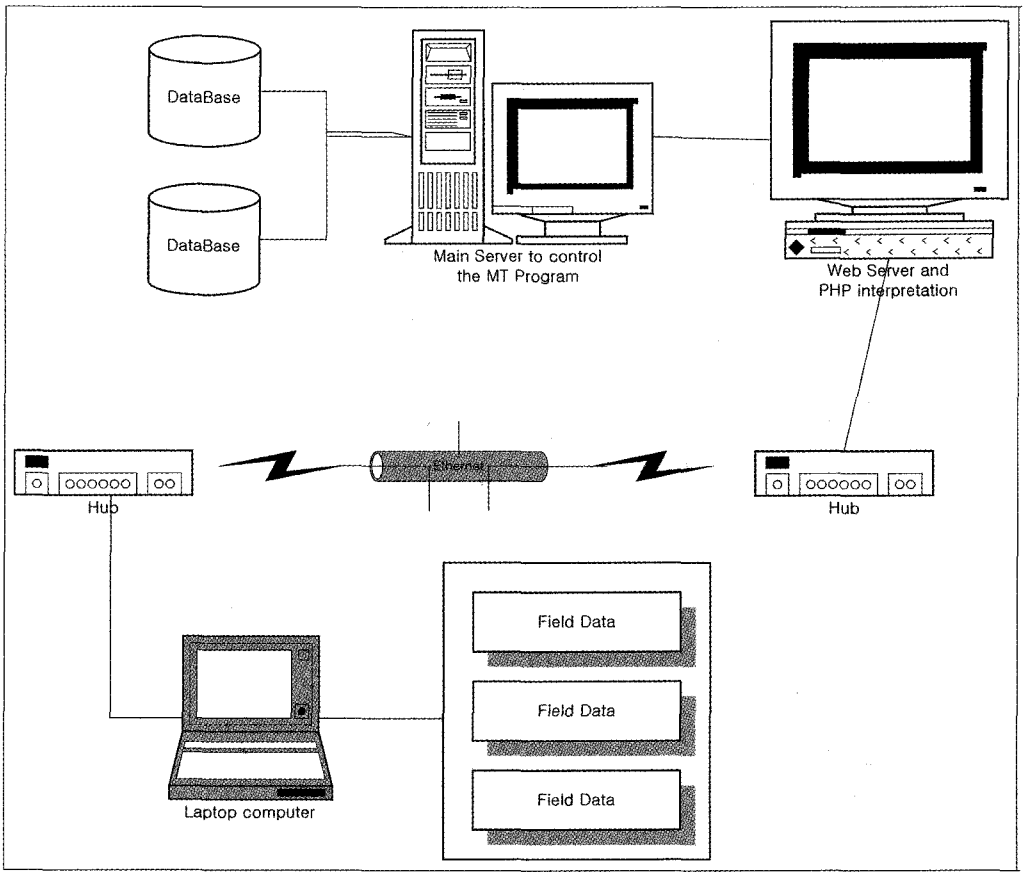

Fig. 1. Network based MT data processing system.

인터넷에 연결하고 이를 적절한 방식으로 서버에 접속하여 자료를 넘기고 서버에서 계산을 수행하고 그 결과를 다시 노트북으로 넘겨받아 해석하는 것이다. Fig. 1은 이러한 시스템의 모식도를 보여준다.

주 계산을 수행하는 서버는 MPI(message passing interface, 1997)기술에 의해 프로그램되어 계산속도를 최대한 높일 수 있는 구조를 가지게 된다. 이상과 같은 시스템이 완성될 경우, 고가의 서버를 한 곳에만 설치하면 인터넷을 통해 어느 곳에서나 접속하여 원격으로 손쉽게 자료를 처리하여, 경제적 이득을 얻는 것은 물론이고, 탐사자료의 일관성 및 안정성을 유지할 수 있으므로 자료의 교환과 공유, 데이터베이스화 등에 대한 문제를 해결할 수 있다. 또한 교육 및 실습에 효과적으로 이용되므로서, 실제적 자료처리 기술의 습득 및 관련 종사자의 수준 향상에 도움을 제공할 것으로 생각된다.

본 연구에서는 위에서 언급한 인터넷 활용 과정에서의 각각의 기술을 고찰하고 이를 구현하는 알고리즘을 제시하고자 한다.

네트워크 모델

대부분의 네트워크 관련 소프트웨어는 중점적으로 자료를 처리, 관리하는 서버와 이에 접근하여 자신의 문제를 해결하는 클라이언트의 형식으로 구성되어 있다. 이를 지구물리 자료의 처리에 적용하면, 현장에서 취득된 자료를 일괄적으로 서버에 전송하여 그 처리 결과를 다시 수신받는 방식이 된다. 이는 서버에 전적으로 의존하여 자료를 처리함으로써 네트워크 의존률을 낮출 수 있으므로, 서버 애플리케이션의 개발과 관리가 비교적 용이한 면이 있으나, 처리 과정상의 문제가 발생할 경우나, 계속적인 통신이 이루어져야 할 경우 불가능한 면이 있다.

최근 대중화된 월드 와이드 웹에 의한 서버/클라이언트 프로그램은 자체적으로 운영체제 독립적인 웹 브라우져들을 이용하여 통신하므로, 그 처리가 수월하다. 하지만 보다 완벽하고 다양한 사전처리가 요구되는 지구물리 자료의 특성 상, 소켓(socket) 방식에 의해 자체 통신이 가능한 모델의 개발이 필요하다. 이와 같이 처리되는 자료들은 사후 관리 등을 위해 데이터베이스화 될 수 있으며, 이를 위한 기술로 JDBC(Java DataBase Connectivity)가 제시된다.

한편 인터넷 및 인트라넷의 보급이 확산되면서 네트워크는 컴퓨팅 환경의 기반을 이루게 되었으며 이 기종 컴퓨터간 분산 컴퓨팅의 중요성이 부각되고 있다. 그러나 다른 기종 컴퓨터간의 분산 컴퓨팅은 각 컴퓨팅 환경을 구성하는 플랫폼의

다양성 때문에 매우 복잡하고 어려운 작업으로 여겨지고 있다. 관련 업체 및 표준화 단체에서 분산 컴퓨팅을 위한 표준을 제안하고 있지만 분산 응용 프로그램의 요구가 점차 다양화됨에 따라 보다 일관적이고 플랫폼에 독립적인 개발 환경을 필요로 하게 되었다. 썬마이크로시스템즈에 의해 개발된 Java가 등장하면서 분산 응용프로그램 개발자들은 이러한 개발 환경을 제공받을 수 있는 기회를 갖게 되었다. Java가 제시하는 엔터프라이즈 플랫폼은 분산 컴퓨팅을 위한 다양한 API(Application Program Interface)를 제공하며 응용 프로그램의 생산성 향상 및 플랫폼 독립성을 제공한다.

분산 컴퓨팅은 서버/클라이언트의 일방적 방향의 자료의 송수신에 의한 교환이 아닌, 동등하게 서로의 메소드를 교환하는 방법이다. 특히 Java 기술과 함께 도입된 원격함수호출(Remote Method Invocation: RMI) 기술은 객체를 주고받음으로써 보다 능률적으로 컴퓨터 네트워크의 능력을 이용할 수 있는 기법이다.

Web에 의한 서버/클라이언트 모델

인터넷 익스플로러나 넷스케이프와 같은 웹 브라우져를 통해 탐사자료와 각종 파라미터를 전송한 후 서버 처리 결과를 제공받는 방식은 가장 간단하게 인터넷을 지구물리 자료처리에 이용할 수 있는 방법이다. Fig. 2는 이 과정을 간단히 도시한다.

Fig. 2에서 보는 바와 같이 자료의 처리를 위해 사용자 인증과정을 마친 후 HTML문서상에서 파라미터를 전달하기 위해 form tag로 작성된 웹 페이지에 측정자료의 이름과, 수행 제목 등을 적어 넣는다. 시계열 자료와 스펙트럼 자료는 추후 정밀처리를 위해 선택사항으로 제공될 수 있다. 그림에서 패스워드는 차후 결과를 보고자 할 때 이용되므로, 따로 기억해 두어야 한다. 서버 측에서 처리가 완료된 후 클라이언트 측은 다시 웹 페이지에 접속하여 결과를 관찰하는 페이지로 옮겨가게 되고, 적당한 인증키와 패스워드를 삽입하면 Fig. 3과 같은 결과를 얻을 수 있다. 이후 수행결과를 삭제하거나 전송받기를 원하는 메뉴 페이지로 전환되고 일단락된다.

한편 서버측에서는 form tag로부터 전달받은 파라미터를 PHP 스크립트에 의해 처리하여, MT 역산 프로그램을 가동하는 shell 스크립트를 작성하게 된다. PHP 스크립트는 각종 데이터베이스 서버와 연동하여 처리 자료를 관리할 수 있다.

소켓(socket) 방식에 의한 서버/클라이언트 모델

Web에 의한 서버/클라이언트 모델은 그 작성이 간단하여 쉽게 구축할 수 있으나, 한 번에 다수의 인원이 접속하는 경우의 혼란 가능성이 있다. 또한 서버에 전송할 자료로의 변환, 시계열 자료처리, 서버에서 처리된 결과의 도시 등에 있어서 다른 프로그램을 사용해야 하기 때문에 효율성과 일관성에 문제가 있으며, 서버에서의 자료처리 상태나 예상 시간 등의 정보를 제공하는데도 어려움이 있다. 따라서 기본적인 전처리 및 후처리 과정과 더불어 서버와의 자료통신을 담당하는 전용 클라이언트 프로그램의 개발을 통하여 이상의 문제점을 효과적으로 극복할 수 있다. 이와 같은 전용 클라이언트를 개발하기 위해서는 네트워크 프로그래밍에 적합한 언어를 선택하여, 소켓을 이용하여 송수신하여야 한다. 본 연구에서는 Java에 의한 MT 처리 프로그램의 서버/클라이언트 모델을 개발하고자 한다.

Java를 애플릿(applet)이 아닌 완성된 어플리케이션(application)으로 사용하는 큰 이유 중의 하나는 네트워크 프로그램 작성의 용이함과 그와 관련된 유지 및 보수가 뛰어나기 때문이다. 여기에 다중 처리를 뒷받침할 수 있는 간단 명료하고 뛰어난 다중쓰레드(multithread) 기능은 많은 양의 접속을 동시에 처리할 수 있도록 한다.

소켓은 포트(port)보다는 소프트웨어적인 개념으로, 데이터가 이동을 시작하거나 도달할 수 있는 창구 역할을 하는데 포트를 이용한 데이터 전송을 추상화한 것으로 보면 된다. 클라

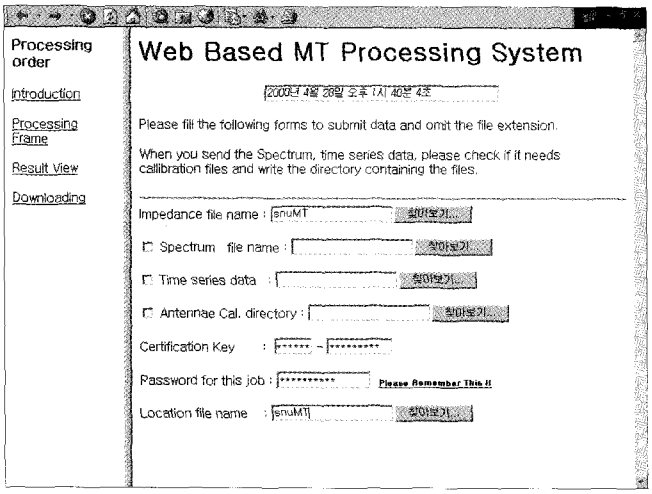

Fig. 2. MT data transfer and analysis through Web.

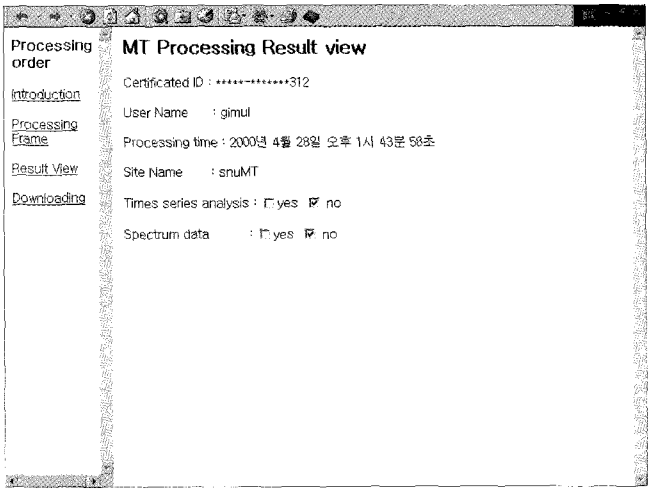

Fig. 3. Result of MT data analyzed from server.

이언트와 서버에 존재하는 소켓은 한 쌍(socket pair)을 이루어 클라이언트에서 서버로의 데이터 통로를 만든다. 특정 TCP 연결을 의미하는 소켓 쌍은 인터넷 상에서 오직 하나로 유일하다(이현우 외, 1999). Java에서의 소켓 프로그램은 서버 소켓(Server Socket)과 소켓 클래스로서 이루어진다. 기존의 언어들이 복잡하고 에러를 유발하기 쉬운 수많은 함정을 포함하고 있는데 반해, Java는 2개의 클래스에 소켓의 설정과 연결에 관한 모든 변수와 함수를 담고 있어서, 일단 선언이 된 후 연결이 이루어지면 기존의 프로토콜을 이용하듯이 사용하면 된다.

소켓에 의한 모델은 개발자가 자신의 고유 프로토콜을 정의할 수 있으므로 필요한 정보를 주고받는 데 있어 기존의 프로토콜보다 수준 높은 방법을 제공하며, 보안이나 다중 접속 시의 처리가 용이하다. 또한 사용자의 자료에 대해 보다 세밀한 처리와 정보를 제공하므로 보다 완벽한 서버/클라이언트 모델을 제공한다.

JDBC에 의한 서버 측의 자료 관리

JDBC는 Java 클라이언트와 관계형 데이터베이스 서버와의 연동을 위한 해결책을 제공한다(Fig. 4). 다시 말해서 Java를 이용한 데이터베이스 접속과 SQL(Structured Query Language) 문장의 실행 그리고 실행 결과로 얻어진 데이터를 다루는 방법과 절차를 제공한다. JDBC의 큰 특징은 통합적인 SQL 데이터베이스를 이용할 수 있는 프레임워크로서 데이터베이스에 비종속적인 분산 응용프로그램을 구현할 수 있도록 해준다는 것이다. 즉, JDBC와 연동 가능한 드라이버 및 모듈을 모두 사용하여 데이터베이스에 접근할 수 있는데, 순수 Java로 작성된 드라이버와 호스트 시스템에서 제공하는 드라이버를 모두 사용할 수 있는 것이다. Java 클래스 형태로 제공되는 JDBC API는 ODBC(Open DataBase Connectivity)나 기존의 데이터베이스 접근을 위한 인터페이스의 상위 계층을 구현할 목적으로 고안되었다. 또한 웹 브라우저상의 보안에 위배되지 않는 조건 하에 다양한 데이터베이스 접근 방식을 제공

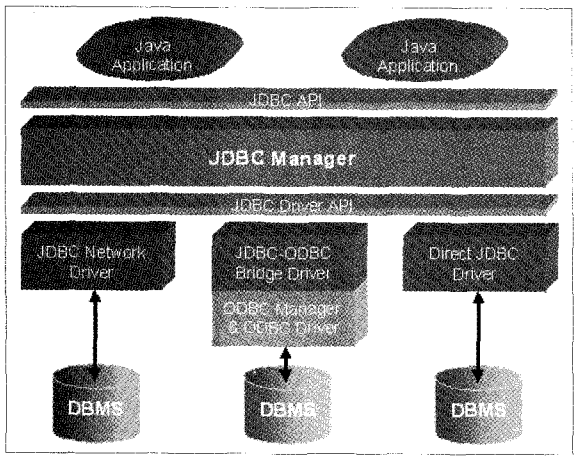

Fig. 4. Structure of JDBC.

함으로써 CGI(Common Gateway Interface)를 이용한 수동적인 방식을 벗어나 보다 능동적인 데이터베이스의 이용이 가능하다.

각종 탐사자료는 바로 이와 같은 JDBC에 의해 관리되고, 클라이언트 내의 검색 루틴을 통해 참조, 탐색할 수 있다.

분산 컴퓨팅 – RMI

분산 컴퓨팅 환경 하에서 Java는 응용프로그램 개발자에게 서버 독립성 뿐 아니라 트랜잭션 시스템 및 데이터베이스 등 하부구조와의 독립성을 제공한다. Java 엔터프라이즈 API를 통하여 개발자는 기존의 다양한 분산 컴퓨팅 플랫폼과 프로토콜을 위한 일관된 개발 환경을 제공받는다. CORBA(Common Object Request Broker Architecture)와 Java의 연동을 위한 JavaIDL(Java Interface Definition Language), 원격 Java 함수 호출을 위한 RMI, Java와 관계형 데이터베이스와의 연결을 위한 JDBC 등의 API를 이용하여 개발자는 Java라는 단일 플랫폼 하에서 분산 응용프로그램을 작성할 수 있다.

분산 환경은 네트워크에 기반한 지구물리 자료의 처리에도 효율적으로 이용될 수 있다. 즉, 서버와의 계속적인 연결이 가능한 경우, 각종 파라미터의 통신에 있어서 복잡한 헤더와 양식(protocol)을 요구하는 소켓방식을 벗어나서, 간단히 지역 내(local)의 함수를 부르듯이 사용할 수 있다. 단 이 과정에서, 서버와 클라이언트가 모두 Java일 경우는 매우 간단히 RMI를 이용하면 되지만, 그렇지 않을 경우에는 위에서 언급한 CORBA라는 기술을 바탕으로 Java 인터페이스를 작성하여야 한다.

대부분의 지구물리 관련 프로그램들은 C나 Fortran으로 작성되어 있는 경우가 많고, 이를 호출하기 위해서는 이 기종간의 통신을 원활히 할 수 있는 CORBA 기술을 사용해야 한다. 하지만, 기존의 코드들이 대개 독립적으로 수행 될 수 있는 경우가 많으므로, 이 들의 인터페이스를 약간 개선하여 자바에서 호출할 수 있는 형태로 변경하고 이후에 설명할 RMI를 이용하는 것이 더욱 간명하다.

RMI 기술은 자바 개발자 도구(JDK) 1.1에 등장한 이래, 네트워크 프로그래밍 기술을 한 단계 위로 끌어올린 것으로 평가받는다.

RMI의 구조는 Fig. 5와 같이 Stub 및 Skeleton 계층, 원격 참조(Remote Reference) 계층, 전송(Transport) 계층으로 이루어져 있다. RMI를 구성하는 요소의 하나로서 Stub 과 Skeleton 계층은 JavaIDL의 그것과 유사한 기능을 제공한다. Stub 및 Skeleton 계층과 전송 계층을 연결해주는 원격 참조 계층은 두 가지 기능을 수행하는데, 첫 번째는 Stub과 Skeleton 계층으로부터의 전송 요청을 호스트 시스템의 전송 호출로 변환해 주는 것이며, 두 번째는 원격 객체를 호출할 수 있게 하는 원격 참조 프로토콜을 수행하는 기능이다. 가장 하단에 존재하는 전송 계층은 원격 객체간의 호출을 위한 연결의 설정, 해제 및

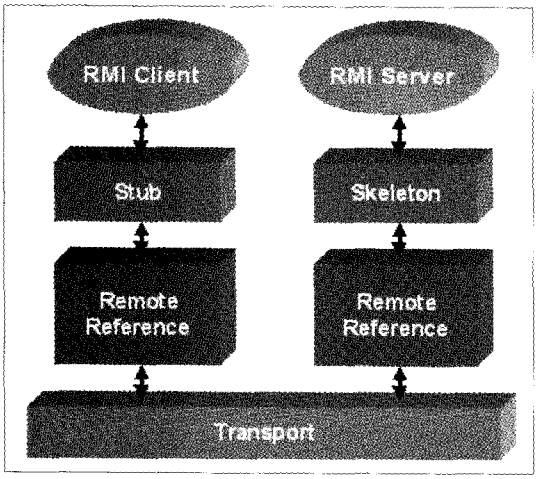

Fig. 5. Structure of RMI.

관리를 수행한다.

통신 접속 상태를 계속 유지할 수 없는 야외 현장에서는, 대개 한번의 접속을 통해 자료의 송수신과 각종 파라미터의 전송을 완료하여야 한다. 그러나 이와 같은 즉각적인 처리가 완료된 후, 인터넷 접속을 계속 유지 할 수 있는 곳에서는 보다 원활하고 다양한 처리를 위해 계속적으로 서버와의 접속을 유지하고 세부적인 조절이 가능하게 작업하는 것이 능률적이다. RMI는 이와 같은 클라이언트 프로그래밍에 효율적으로 이용될 수 있으며, 소켓 프로그래밍보다 고급의 처리를 제공한다. 즉, 하위 수준에서 일일이 규정하고 처리하는 소켓 방식과 달리, 원격 함수를 마치 자기 컴퓨터에 있는 함수를 호출하듯이 이용하는 것은 훨씬 고급의 사용자 인터페이스를 제공하고 프로그램 작성상의 문법적 오류나 버그를 줄일 수 있다. 또한 앞에서 언급한 데이터베이스의 접근이나 서버 자료의 이용에 있어 강력한 효율성을 제공하며, 차후 관리 유지, 보수에도 많은 이점을 제공한다.

MPI 병렬처리 기술

앞에서도 언급한 바와 같이 많은 양의 계산을 요하는 MT 자료의 역산 프로그램은 MPI를 이용하여 병렬 처리함으로써 짧은 시간 안에 계산을 역산 결과를 얻을 수 있도록 할 것이다.

Message passing은 분산메모리(distributed memory) 환경의 다양한 형태의 병렬 플랫폼에서 작업의 병렬화를 위한 패러다임으로 메시지의 교환을 통해서 각 프로세서간의 정보를 공유하는 방법을 의미된다. MPI는 분산메모리 환경의 메시지 패싱 병렬화의 표준으로 1992년부터 미국 및 유럽의 40개 기관들이 참가하여 1994년 MPI-1(Message Passing Interface 1) (Hubertus *et al.*, 1994)이 완성되었고 이후 1997년에 MPI-2(MPI Forum, 1997)가 발표되었으며 Fortran77, Fortran90, C, C++ 등의 프로그래밍 언어들을 지원한다. 현재 고가의 수퍼컴퓨터를 이용하지 않더라도 네트웍으로 연결된 컴퓨터들, 또는 저비용으로 구현할 수 있는 Beowulf와 같은 시스템을 이용하여 고효율의 message passing 병렬처리를 하는 것이 가능하다.

Message passing을 통한 병렬처리의 핵심은 작은 규모로 분할된 작업들의 각 프로세서로의 할당, 각 프로세서에서 계산된 결과들의 취합이라 할 수 있다. 프로그램의 효율적인 병렬화에 있어서 고려해야 할 점은 전체 프로그램 중에서 병렬화가 불가능한 부분의 비율, 병렬화에 적합한 알고리듬의 선택, 계산시간/통신시간 비, 각 노드간의 load balancing 등이다. 이러한 관점 하에 MT 탐사자료의 역산프로그램의 병렬화에 대해 알아보면 다음과 같다.

MT 탐사자료의 역산문제에서 계산량의 대부분은 모델반응 계산(forward modeling), 민감도행렬(sensitivity matrix), 모델 update를 위한 모델미소변화량 계산 등에 집중된다. 민감도행렬의 계산은 상반성(reciprocity) 원리를 이용하면 (측정 주파수의 수)×(관측점의 수) 만큼의 모델반응 계산이 필요하다. 결국 MT 역산 문제에서 대부분의 계산량은 주어진 지하모델의 모델반응 계산에 집중되어 있음을 알 수 있다. 부록은 이러한 모델반응 계산을 master-slave 모델로 병렬화 한 예로 작업의 할당 및 자료의 통신부분만을 주로 하여 나타낸 것이다.

구성된 MPI 시스템에서 master 노드는 작업의 할당 및 계산된 자료를 취합하는 일을 담당하며 나머지 노드들은 master 노드에서 할당된 작업을 수행하고 그 결과를 master 노드에 반환하게 된다. 이 프로그램을 각 단계별로 살펴보면 다음과 같다. 먼저 맥스웰 방정식을 근사하는 선형시스템의 계수행렬을 계산하게 되는데 주파수가 들어가는 부분을 제외하고 나머지 부분들을 2개의 행렬로 나누어서 계산할 수 있다. 이 부분 또한 병렬화가 가능한데 여기서는 제외하였다. 실제 필요한 행렬은 이 두 행렬과 계산하고자 하는 주파수를 이용하여 구성하게 된다. 이 행렬들은 MPI_BCAST에 의해 각 slave 노드에 전해진다. Slave 노드들은 master 노드에서 프로세스 종료 신호가 오기 전까지는 계속 실행되고 있으므로 이 행렬들은 한 번만 전해지면 된다. Master 노드 부분은 계산하고자 하는 주파수의 수만큼 반복하는 루프로 구성되며 작업이 끝난 노드로부터 계산 결과를 받고 바로 새로운 계산 작업을 할당하도록 구성되어 있다. 서브루틴 recv_dat, send_dat는 MPI 서브루틴들을 이용하여 작성되어 있으며 모델반응 계산에 필요한 값들의 전송 및 결과 값의 반환 등의 작업들을 수행하는데 자세한 코드는 여기서는 생략한다. 이와 같은 master 노드의 코드는 각 노드들간의 load balancing 까지 고려된 것이다. 이때 루프가 끝나게 되면 slave 노드 수만큼의 작업이 수행되고 있으므로 이들의 계산 결과를 받고 나서 프로그램이 종료하게 된다. 다음으로 slave 노드 부분은 무한 루프로 이루어져 있는데 이는 하나의 작업 결과를 반환하고 다시 새로운 작업을 받기 위함이다. 만약 모든 작업이 끝나게 되면 루프를 탈출하여 프로세스가 종료되고 이는 task_code 인수를 통해 제어된다. Slave

노드는 주파수 및 기타 필요한 모든 정보를 master 노드로부터 받고 실제 모델 반응을 계산하여 master 노드에 되돌려 준다.

본 예는 매우 간단한 구조를 가지지만 load balancing, 작업의 크기를 주파수 별로 크게 나눔으로써 계산/통신 비의 향상 등의 문제가 고려된 것이다. 그리고 추가적으로 non-blocking send, receive를 사용하여 작업효율을 더 증대시킬 수도 있다.

이러한 예와 같은 구조를 바탕으로 비선형 CG(nonlinear conjugate gradients)방법, 민감도 행렬의 Broyden update, 웨이블릿(wavelet) 변환 등의 기법들을 결합함으로써 대량의 MT 자료의 2차원, 3차원 역산문제를 매우 신속하게 풀 수 있는 프로그램을 개발할 수 있을 것이다.

MT 자료 처리 서버/클라이언트 시스템의 구성

MT 자료의 처리는 크게 클라이언트 프로그램과 서버프로그램으로 나눌 수 있다. 클라이언트 프로그램은 현장에서 측정한 전장 및 자장의 시계열 자료를 처리하고, 각종 현장 지표들을 즉각 파악할 수 있는 다양한 정보(tipper, skew, strike angle 등)들을 분석하는 기능과 서버와의 자료 통신 기능을 제공하게 된다. Fig. 6은 클라이언트 프로그램의 수행 예를 보여준다. 클라이언트 프로그램은 비교적 간단한 처리를 수행할 수 있는 기능도 포함한다(Table 1 참조). 클라이언트 프로그램은 서버에 측정된 임피던스 및 측정 정보 등을 송신하고 역산 작업을

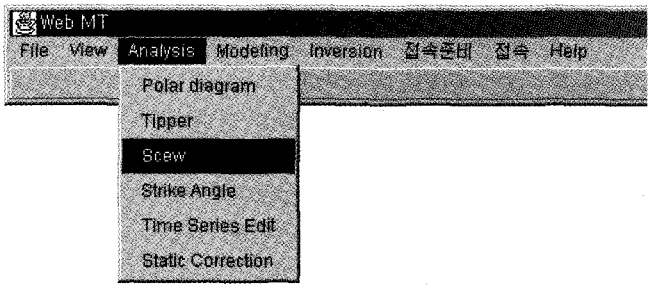

Fig. 6. MT client program.

Table 1. *Main functions of server/client side program*

		전처리과정	시계열의 재편집
클라이언트		strike,TE,TM	strike를 구하고 이에 따른 TE, TM mode 계산
		tipper, skew	tipper와 skew 계산
		impedance polar diagram	impedance polar diagram의 도시
		static correction	EMAP 등의 방법에 의한 static correction
서버	순산	1차원	임의로 주어진 1차원 모델의 순산
		2차원	임의로 주어진 2차원 모델의 순산
		3차원	임의로 주어진 3차원 모델의 순산
	역산	1차원	1차원자료의 역산
		2차원	2차원자료의 역산

수행시킨 후 서버와의 접속을 끊을 수 있다. 이는 PCS 등을 이용해서 클라이언트 프로그램과 서버프로그램을 연결할 경우 접속 시간의 과다에 따른 비용 문제에 도움이 된다. 역산이 끝나면 클라이언트 프로그램에서 다시 서버 프로그램에 접속해서 역산이 끝난 후의 결과들을 전송 받게 된다.

서버프로그램은 클라이언트프로그램에서 전송받은 임피던스 파일 및 측점 정보를 바탕으로 역산을 수행하여 그 결과를 클라이언트의 요청에 의해 되돌려주는 역할을 하며 또한 임의의 지하 모델에 대한 모델반응을 계산하는 역할도 하게 된다. 서버프로그램이 가져야 할 기본적인 기능에 대하여 Table 1에 간단히 정리를 하였다.

결과 및 고찰

본 연구에서는 네트워크에 기반한 MT자료 처리 및 해석 시스템의 구축을 위한 여러 가지 기술적인 사항들에 대하여 논의하였다. 자료 처리에 있어 네트워크를 이용하는 것은 1대 이상의 컴퓨터를 이용하여 고성능의 서버를 만들 수 있다는 점 이외에도 일관성 및 체계성 등 여러 장점을 가지고 있다. 또한 한국에서는 광범위하고도 고수준의 인터넷 망이 구성되어있으므로, 이의 적절한 활용은 각종 비용의 감소와 효율성의 증대를 가져올 것으로 예상된다.

기존의 web에 기반하여 클라이언트/서버 모델을 개발하는 것은 상대적으로 쉽고 효율적이어서 많은 비용을 들이지 않고 완성할 수 있는 간단한 네트워크 처리 시스템이다. 그러나 다중 사용자의 지원이 쉽지 않고, 지구물리 자료가 그 특성상 많은 사전, 사후 처리를 요구하므로, 복잡하고 다양한 인터페이스를 지원하지 못하는 웹에만 의존하는 것은 현실적으로 어렵다. 따라서 자유롭게 서버와 통신할 수 있는 독립적인 모델이 필요하며, 이를 위해 본 연구에서는 소켓과 분산 처리 기술의 일종인 원격 함수 호출(RMI) 기술을 적용, 검토하였다. 소켓 처리의 경우 개발자의 의도에 알맞게 프로토콜을 재 정의함으로써, 보다 효율적으로 서버/클라이언트 통신을 구성할 수 있으며, RMI는 소캣 프로그래밍을 보다 고급화, 단순화함으로써 관리 및 유지 보수에 있어 획기적 안정성과 효율성을 제시한다. 이 두 기술은 클라이언트 사용자의 인터넷 연결 환경과도 밀접한 관련이 있으므로, 현장용과 인터넷 환경이 갖추어진 환경에 따라 다르게 이용되어야 할 것이다.

또한 본질적으로 대량의 계산을 요구하는 MT자료 해석의 특성을 극복하기 위해 MPI를 이용한 MT 역산 프로그램의 병렬화에 대해서도 검토하였다. 역산 과정의 거의 대부분을 차지하는 MT 순산 모델링 과정을 각각의 주파수별로 각각의 병렬 노드에 배정하도록 병렬화 함으로써 빠른 시간 안에 역산을 수행할 수 있으며 이러한 병렬 계산은 고가의 수퍼컴퓨터를 이용하지 않더라도 네트웍으로 연결된 컴퓨터들을 이용하거나 저가의 Beowulf 시스템을 이용하여 효율적으로 수행 가능할

것이다.

이처럼 하나의 서버에 여러 사용자에 의해 검증된 MT 자료 처리 및 해석 프로그램이 탑재되어 있으면 매우 안정적이고 빠른 해석결과를 얻을 수 있는 길이 된다. 따라서 이 프로그램은 개발이 완료된 후에도 많은 사용자들에 의해 검증을 거쳐야하고 또한 새로운 요구들에 의해 기능을 보완하는 방향으로 발전해나가야 할 것이다.

사 사

본 연구는 한국과학재단에서 시행하는 해외박사후과정 지원 프로그램에 의하여 일부가 지원되었다. 이 기관에 감사한다.

참고문헌

이현우, 김형국, 홍성민, 1999, *Java programming bible v.2 : 영진출판사*. 627-674.

Carey, R., and Bell G., 1997, *The annotated VRML 2.0 reference manual*: Addison-Wesley Developers Press, Reading, Massachusetts, 501.

Hibbard, B., 1998, VisAD: connecting people to computations and people to people, *IEEE Computer Graphics 3*, **32**, 10-12.

Huang, B. and Lin, H., 1999, GeoVR: a web-based tool for virtual reality presentation from 2D GIS data, *Computer & Geoscience*, **25**, 1167-1175.

Hubertus, F., Hochschild, P., Pattnaik, P., and Snir, M., 1994, An efficient implementation of MPI. *In 1994 International Conference on Parallel Processing*.

Lin, H., Gong, J., and Wang, F., 1999, Web-based three-dimensional geo-referenced visualization, *Computer & Geosciences*, **25**, 1177-1185.

Message Passing Interface Forum, 1997, *MPI-2: a Message Passing Interface Standard*: University of Tennessee.

Templeton, M. E. and Gough, C. A., 1999, Web seismic Un*x: making seismic reflection processing more accessible, *Computer & Geoscience*, **25**, 421-430.

Appendix A. An example of MT forward modeling by MPI parallelization

```
PROGRAM MPIMTFWD
!
! EXAMPLE OF PARALLEL MT FORWARD MODELING
! Programming Laguage : Fortran 90

! Include MPI definitions
USE mpi

! Include other modules
USE ...

! Define Variables
  . . .

! Reading Inputs
   CALL mtfwd_input( ... )

! Initialize MPI system
   CALL MPI_INIT(rc)

! Get the ID number of current task
   CALL MPI_COMM_RANK(..., no_id , ..)

! Get the number of total nodes in MPI system
   CALL MPI_COMM_SIZE(...,no_pc , ...)

! Number of slave nodes
   no_slave = no_pc - 1

! Calculate System matrix : Can be parallelized
   CALL mak_system(..., CA , CB , ...)

! Broadcast CA , CB
   CALL MPI_BCAST(CA , ... )
   CALL MPI_BCAST(CB , ... )

! Forward calculation over required frequencies
send_count = 0
if(no_id == 0 )  then   ! Master node
   DO i = 1 , no_freq
      if(send_count >= no_slave) then
         ! receive MT forward result
         CALL recv_dat(..... , no_msgsource)
         ! node ID waiting new job
         no_destid = no_msgsource
      else
         no_destid = send_count + 1
      end if
      ! Assign new job
      CALL send_data(no_destid, send_count, ...)
   END DO

   ! Receive remaining data
   DO i = 1 , no_slave
      CALL recv_dat(.... , no_msgsource)
   END DO

   ! Send stop signal to slave processes
   do i = 1 , no_slave
      CALL MPI_SEND(stop_task_code , .. , i , .. )
   end do
else                    ! Slave node
   no_masterID = 0
   DO
      CALL MPI_RECV(task_code , .. , no_masterID , ..)
      if(task_code == stop_task_code) EXIT
      CALL MPI_RECV(freq , .. , no_masterID , .. )
         . . .
      CALL mtfwd( ... )
```

```
        ! Send Calculated results to master node
        CALL MPI_SEND( ..., no_masterID , .. )
        ! Loop for waiting new job
    END DO
end if

CALL MPI_FINALIZE(rc)
END PROGRAM MPIMTFWD
```