

An Internet Accessible Grid Computing System: Grid-ELIMINO

Y. Wu¹, W. Liao², P. Wang², D. Lin³, G. Yang¹

¹*Department of Computer Science and Technology,
Tsinghua University, Beijing 100080, China.*

²*Institute of Computational Mathematics,* Kent State University,
Kent, Ohio 44242, U.S.A.*

³*State Key Laboratory of Information Security, Institute of Software,
Chinese Academy of Sciences, Beijing 100080, China.*

Abstract

The Grid-ELIMINO system employs and interacts with multiple slave ELIMINOs to achieve a distributed/parallel computing environment, using an IAMC approach. ELIMINO is a research system developed to support Wu's method for computing characteristic sets of polynomials and for other related operations. Grid-ELIMINO combines grid computing and the IAMC framework to deliver its computing powers to users through the Internet. The overall IAMC framework, grid technology/GT, ELIMINO and OMEI, as well as the implementation of Grid-ELIMINO based on GT and OMEI are presented. The way to access and apply Grid-ELIMINO is also discussed.

1 Introduction

ELIMINO [9] is a new symbolic computation system being developed at the Institute of Systems Science of the Chinese Academy of Sciences. Capabilities of ELIMINO include manipulation of multi-precision numbers and polynomials, computation of characteristic sets in Wu's method [18], polynomial equation solving, geometric theorem proving etc. As a universal system for a broad class of problems, ELIMINO is very computation intensive.

Polynomial characteristic sets are especially very computation intensive. Even medium-sized characteristic sets problems can take a very long time to solve. Consequently,

*Work reported herein has been supported in part by the National Science Foundation under Grant CCR-0201772 and in part by the Ohio Board of Regents Computer Science Enhancement Funds.

ELIMINO can benefit greatly through parallelization. Grid-ELIMINO is a distributed/parallel computing system using multiple ELIMINOs over a grid to speed up the computation.

The grid [1, 2] technology uses high-speed networks to integrate heterogeneous computers distributed over a network to form a virtual supercomputer. Grid computing is an important and current research area and it promises to supply supercomputing powers by combining a grid of networked workstations. By using grid technology, *Globus Toolkit* (GT) [3], multiple ELIMINOs distributed over a grid can provide high performance computing services for the on-grid users as an integrated system: Grid-ELIMINO.

The *Internet Accessible Mathematical Computation* (IAMC) [12, 13, 14] framework is an effort to establish a protocol-based, platform, programming language, and mathematical encoding independent, solution for serving mathematical computation over the Web/Internet. It makes math-oriented services easily and widely accessible on the Internet in many contexts (Web vs Email, small vs large bandwidth, etc.). The IAMC framework can be used to deliver the computing powers of Grid-ELIMINO to off-grid users anywhere on the Internet.

In the IAMC framework, connectivity between client and server is defined by the *mathematical computation protocol* (MCP) [15] while connectivity between IAMC server and external compute engines follows the *Open Mathematical Engine Interface* (OMEI) [7, 8] API specification. We apply OMEI to help build Grid-ELIMINO. By implementing OMEI as ELIMINO's application programming interface through GT, we can easily turn the ELIMINO system into a grid-based server. Grid-ELIMINO contains a master program controlling a number of slave ELIMINO servers. Also, IAMC already has an OMEI-ready server prototype (Starfish) [5], and a client prototype (Dragonfly) [6] supplying a nice GUI for back-end mathematical systems.

By adopting GT and IAMC technologies, Grid-ELIMINO achieves the following specific results with minimal effort.

- Parallelizing GCD, factorization and characteristic-sets based computations.
- Making Grid-ELIMINO a powerful and IAMC compliant compute engine through the IAMC server prototype Starfish.
- Delivering the power of Grid-ELIMINO to on-grid users and remote users on the Internet through the IAMC client prototype Dragonfly.
- Demonstrating grid computing as a way to speed up compute engines for IAMC servers
- By complying with the OMEI interface definition, achieving interoperability with other OMEI compliant tools, such as editors and word processing systems.

We begin with an overview of the IAMC framework, the ELIMINO system, and the OMEI specification. We then introduce the *Open Grid Service Architecture* (OGSA) and GT. These pave the way for describing the architecture and implementation of Grid-ELIMINO. Access and application of Grid-ELIMINO are then discussed. A working Grid-ELIMINO demo will be demonstrated at the IAMC'03 workshop.

2 IAMC Framework

The IAMC framework is designed to make mathematical computing easily accessible and usable on the Web/Internet. Figure 1 shows the overall IAMC framework architecture.

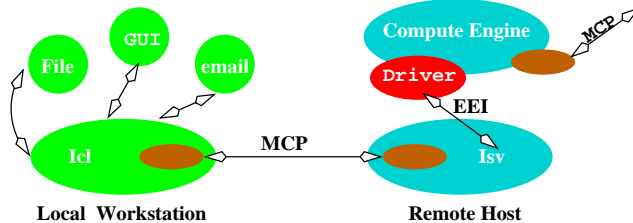


Figure 1: IAMC Architecture

The IAMC framework consists of these components:

1. IAMC client (Icl)—An end-user agent for accessing services provided by any IAMC server.
2. IAMC server (Isv)—A program to provide mathematical computation powers through the MCP protocol. An Isv may or may not employ an external compute engine to perform mathematical computations.
3. Protocol—IAMC clients and servers are connected by the Mathematical Computation Protocol [13]). MCP aims to be a simple and effective session oriented protocol to support one-time transactions and interactive sessions.
4. Mathematical Data Encoding—Standard and user-defined mathematical data encodings can be used.
5. External Engine Interface (EEI)—A specification and API implementation for binding existing compute engines to IAMC servers.

3 ELIMINO with OMEI

ELIMINO is a new computer-mathematics research system developed at the Mathematics Mechanization Research Center(MMRC), Institute of Systems Science, Chinese Academy of Sciences, as part of the “Mathematics Mechanization and its Applications” project. In ELIMINO, many different kinds of mathematical objects and data structures are provided. As an interactive system, ELIMINO is designed to focus on the implementation of Wu’s method for researchers to perform sophisticated mathematical computations. It has very general capabilities for treating numbers, polynomials and characteristic sets [18].

To facilitate mathematical research, ELIMINO is kept open and flexible. The architecture of ELIMINO consists of three parts (see Figure 2):

- **Kernel part** is the soul of the system, it contains implementation of number system, polynomial manipulation system, characteristic sets method. The kernel part can be viewed as a powerful algebraic compute engine.
- **Applications** are packages or programs developed using the ELIMINO library. Examples include the polynomial system solver and the geometry theorem prover. A package may be built-in or loaded into ELIMINO on demand.
- **Front-end** is the interface between the system and users. The front end handles the interaction between the user and the system.

Open Mathematical Engine Interface (OMEI) is an application programming interface (API) specification that aims to be an interface general enough to work for most mathematical packages and systems. It specifies a set of function prototypes together with their syntax and semantics to give a unified programming level interface for heterogeneous mathematical engines. These function prototypes supports all operations that are necessary for server-engine interaction, including connecting to/disconnecting from a compute engine, querying engine capabilities, creating and executing commands, etc..

As an attempt in standardizing programming interface for compute engines, OMEI can achieve several objectives:

- **Making Compute Engine IAMC-Capable**
OMEI and IAMC framework together can help making compute engine internet accessible. The OMEI can be used to connect an IAMC server and external engines. Once an OMEI driver of a compute engine has been developed according to the OMEI specification, the IAMC server will be able to forward the computation requests to the engine and results to IAMC clients.
- **Application Portability**
An application or user interface developed using any OMEI-compatible interface would be portable among different compute engines, as long as those compute engines have OMEI drivers available. That is, the compute engine and its user interface or applications can be developed separately with the help of OMEI specification.
- **Integration of Heterogeneous Compute Engine**
Since an application can access multiple engines by loading multiple OMEI drivers, an integrated compute engine with combined capabilities can be accomplished under OMEI programming model. A parallel/distributed problem solving environment would be easily achievable over the OMEI programming paradigm.
- **3-tier/multi-tier mathematical system**
Through OMEI, 3-tier/multi-tier mathematical system can be arrived easily.

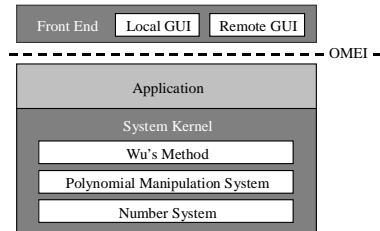


Figure 2: ELIMINO system Architecture

By implementing OMEI as an application programming interface above the application layer of ELIMINO (see Figure 2), ELIMINO makes itself a compute server interoperable with any OMEI compliant tool, such as an IAMC server to access the graphical user interface Drangonfly locally and remotely. It also allows us to easily integrate multiple ELIMINOs together to provide more powerful computing service for Internet users.

4 Open Grid Service Architecture and GT

The grid [1, 2] is a virtual supercomputer consisting of heterogeneous computers (nodes) distributed over a network. Grid computing is a research area about how to combine networked workstations and harness their computation powers.

The Open Grid Service Architecture (OGSA) [2] uses key grid technologies and Web services mechanism [21] to create an integrated, distributed system framework. It specifies a uniform exposed service semantics (the Grid service), defines standard mechanisms for creating, naming, and discovering transient Grid service instances, provides location transparency and multiple protocol bindings for service instances, and thus supports integration with underlying native platform facilities. GT3 (Globus Toolkit 3) is a reference implementation of the Open Grid Service Infrastructure (OGSI). It provides a development environment including programming models for exposing and accessing grid service implementations.

The GT3 provides a uniform Java programming model for programmers to build and deploy their own grid services. Figure 3 shows the architecture of the globus platform and the way users access the grid service. To a globus platform, computing and data resources of a single node are considered *grid services*. A grid service is a network service that provides a set of well-defined interfaces that follow specific conventions [2].

The GT3 *Service Container* (Figure 3) listens for incoming service requests. For a create-service request, the service container first performs security checks. It then calls the *Grid Service Factory* class to create a new service instance. A *Uniform Resource Identifier* URI for this service instance is returned to the requester. This URI is known as a *Grid Service Handle* (GSH). With the GSH, the service client can use and control the service instance. At the end of computations, the service instance can be destroyed.

Each node that provides grid services has a service container that manages all grid services in that node. A grid service factory acts as a service resource provider. It man-

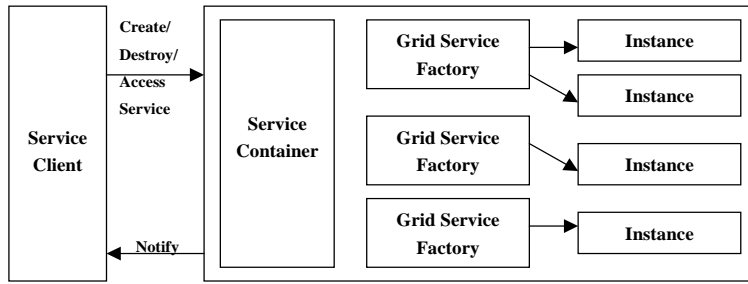


Figure 3: Architecture of GT3

ages all service instances of a specialized grid service.

Through OGSA, multiple ELIMINO engines distributed over a grid can provide powerful computing services for on-grid users as a virtual supercomputer.

5 Grid-ELIMINO Architecture and Implementation

Grid-ELIMINO is a distributed/parallel computing environment built over GT3. Figure 4 shows the architecture of Grid-ELIMINO. It is a master-slave arrangement. The master program, on the client side, instantiates and controls multiple slave ELIMINO servers, each with an OMEI front end. The master runs a *control pool* of threads. Each control thread is in charge of the interaction with one particular remote ELIMINO engine. The *control pool* loads the task class and allocates server resources for the required tasks. For each task, the control pool first creates a service instance (a slave ELIMINO) in the allocated server node and then creates a new control thread for the task. Because the actual task is executed in the ELIMINO server, the only responsibility of the control thread is to supply service control.

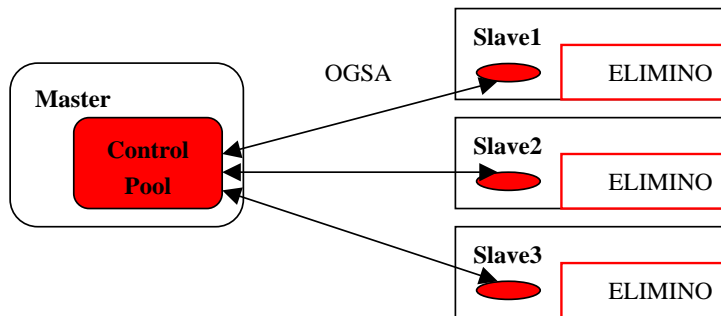


Figure 4: Grid-ELIMINO Architecture

Developers can easily create and access the ELIMINO computing services following the OMEI compliant API Just like writing an MPI [22] program, developers simply write a Java class that describes the task for each ELIMINO server and send this class to

the control pool. The control pool accesses the remote ELIMINO servers through OMEI drivers. The OMEI drivers in turn access computing service through a grid service locator. Figure 5 shows the control flow of a computing thread in the pool.

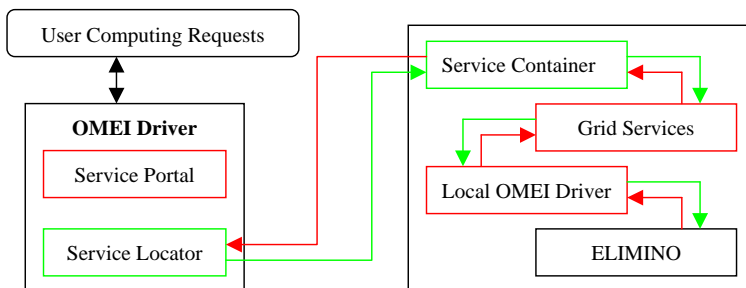


Figure 5: Control Flow of One Computing Thread

As shown in Figure 5, an *ELIMINO server* is an ELIMINO deployed as a grid service through a local OMEI driver. This service is mapped to a GSH (service locator in Figure 5) held by the Grid-ELIMINO master through the GT3 service container, Grid-ELIMINO can deliver the high performance computing power to on-grid users. The IAMC framework can be used to make the power Internet accessible.

6 Use and Example Application of Grid-ELIMINO

Grid-ELIMINO provides an easy-to-setup platform for experimenting with coarse-grain parallelism in computer algebra. As an example, let's see how to perform the distributed/parallel polynomial GCD computation $gcd(P_1(x, y, z), P_2(x, y, z))$ in Grid-ELIMINO using three slave ELIMINOs, where

$$P_1(x, y, z) = (-37z + 13y + 11x)^3 (27yz - 113xz + 174xy)^2;$$

$$P_2(x, y, z) = (-37z + 13y + 11x)^2 (27yz - 113xz + 174xy)^3.$$

The following is the Java control program fragment for this GCD computation.

```
int ELIMINO_Num;
int Task_Num;
DefaultListModel Node_List; //Node list;
// It consists of three node;
DefaultListModel Task_List; //Computing task list;
// It consists of three GCD computation over a finite field.
// pmod_gcd(P1,P2,2003); pmod_gcd(P1,P2,2011);
// pmod_gcd(P1,P2,2017);
public class ELIMINO;
//a java class describing ELIMINO computing task
```

```

ELIMINO[] Pool = new ELIMINO[Task_Num+1];
// Creat a control pool of threads.
for (int i=0;i<Task_Num;i++)
{ int j=i%ELIMINO_Num;
  Pool[i].Connect( (String) Node_List.get(j),
                  "UserID", "PassWord");
  // Creat grid service for $i_{th}$ ELIMINO.
  Pool[i].Execute( (String) TaskList.get(j), 0);
  // Submit the $i_{th}$ task to $j_{th}$ ELIMINO. }
String GCDlift="GCDlift([2003,2011,2017],[";
int num=0;
string result;
while(num<Task_Num)
{ for (int i = 0; i < Task_Num; i++)
  { if (Pool[i] != null)
    { Pool[i].waitforeresult(1);
      result=Pool[i].GetResult(); // Get each result ;
      // Disconnect from Grid_ELIMINO server.
      Pool[i].Disconnect();
      Pool[i].Dispose(); //Destroy the grid service;
      GCDlift=GCDlift+(result.substring(
                          0,result.length()-1))+", ";
      Pool[i] = null;    num++;
    } } }
Pool[Task_Num] = new ELIMINO();
Pool[Task_Num].Connect(
  (String)Pool[Task_Num].get((Task_Num+1)%cpu_Num),
  "UserID", "PassWD");
// GCD lifting computation
Pool[Task_Num].Execute( GCDlift.substring(
                        0,result.length()-1]);", 0);
result=Pool[Task_Num].GetResult(); //final GCD
Pool[Task_Num].Disconnect();
Pool[Task_Num].Dispose();

```

In this experiment, three slave ELIMINOs are assigned for three initial computing requests and each of them computes a GCD mod a small integer prime first. Then the results are lifted with sequential processing by another slave ELIMINO.

Experiments have been conducted on a network of workstations with several polynomial GCD computation examples. The workstation cluster consists of four PIII XEON workstations connected through a local area network. In figure 6, we give the timing data for parallel polynomial GCD computation. All timings are given in CPU seconds from task dispatch to combination of the final computing results.

Example No.	Serial Time	2 ELIMINOs		3 ELIMINOs		4 ELIMINOs	
		Time	Speedup	Time	speedup	Time	speedup
1	4.906	3.656	1.36	2.704	1.83	2.812	1.74
2	9.034	6.712	1.35	6.804	1.33	7.326	1.23
3	64.32	43.26	1.49	42.83	1.50	38.47	1.67

Figure 6: Timings for GCD Computations over Grid-ELIMINO

From the figure 6, we find that we do not get a stable speedup when the number of computing nodes increase. It is because that the load of network and computing resources is dynamic. Even for the same example and the same computing nodes, there is still a gap between the timing data. But we can get an accelerated trend when the scale of computing problem becomes larger.

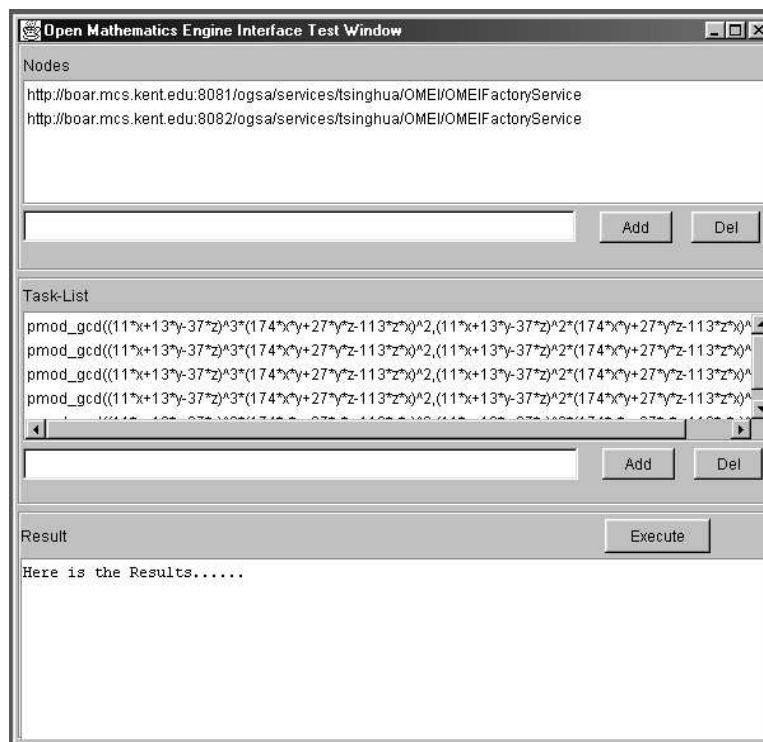


Figure 7: Grid-ELIMINO User Interface

Figure 7 shows the user interface we developed to access Grid-ELIMINO. First, the Nodes box displays all the available grid nodes that can provide computational services. You can also add or delete nodes from this interface. The computation tasks can be seen in the Task Lists box. This list is editable by adding or deleting tasks. Once the node list and task list have been set up, you can click on the Execute button and the tasks will be assigned and submitted to grid nodes for computing. The ongoing status

and results sent back from grid nodes will be displayed in the `Result` box.

7 Internet Accessible Grid-Elimino

The IAMC framework prototypes include the IAMC client prototype Drangonfly and the IAMC server prototype Starfish. These prototypes can also be used to access Grid-ELIMINO environment. Specifically, Grid-ELIMINO can be integrated into Starfish to form a high-performance IAMC server, as shown in Figure 8. We then will be able to use IAMC clients, such as Drangonfly to access Grid-ELIMINO, locally or remotely in the same way as we describe in [19].

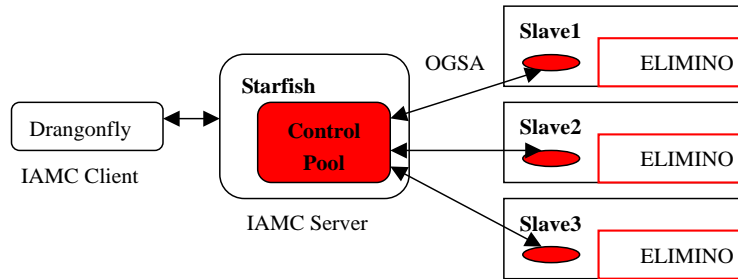


Figure 8: Grid-ELIMINO and Starfish

8 Conclusion and Future Work

By using grid technology, Grid-ELIMINO integrates multiple ELIMINOs distributed over the grid to provide high performance computing services for the grid users through OMEI. It provides parallel GCD, factorization and characteristic-set based computations. IAMC framework is used to deliver the power of Grid-ELIMINO to on-grid users and remote users on the Internet through the IAMC client prototype Drangonfly. At the same time, Grid-ELIMINO achieves interoperability with other OMEI compliant tools, such as other computing tools, editors and word processing systems.

We are implementing the data communication and synchronous control among ELIMINOs. When this work is completed, Grid-ELIMINO will support the parallel computation more widely, and more, provide one MPI-like parallel programming environment for on-grid users and remote users on the Internet through the IAMC client.

References

- [1] I. Foster, C. Kesselman, S. Tuecke, The Anatomy of the Grid: Enabling Scalable Virtual Organization, International J. Supercomputer Applications, 15(3), 2001

- [2] I. Foster, C. Kesselman, *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*, J. Nick, S. Tuecke, 2002
- [3] I. Foster, C. Kesselman, *Globus: A Metacomputing Infrastructure Toolkit*, International J. Supercomputer Application, 1997, 11(2), 115-128
- [4] ION, P., MINER, R., BUSWELL, S., S. DEVITT, A. D., POPPELIER, N., SMITH, B., SOIFFER, N., SUTOR, R., AND WATT, S. *Mathematical Markup Language (MathML) 1.0 Specification*. (www.w3.org/TR/1998/REC-MathML-19980407), Apr. 1998.
- [5] LIAO, W. and WANG, P. S. *Building IAMC: A Layered Approach*. Proc. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'00). pp. 1509-1516.
- [6] LIAO, W. and WANG, P. S. *Dragonfly: A Java-based IAMC Client Prototype*. Lecture Notes on Computing Vol.8. (Proceedings of ASCM 2000.) World Scientific Press, pp. 281-290.
- [7] LIAO, W. and WANG, P. S. *Specification of OMEI: Open Mathematical Engine Interface*. ICM Technical Report. 2001.
<http://icm.mcs.kent.edu/reports/index.html>.
- [8] LIAO W., LIN D. and WANG P. S. *OMEI: Open Mathematical Engine Interface*. Proceedings of ASCM'2001, pp 83-91, Matsuyama, Japan, September 26-28, 2001. Lecture Notes Series on Computing Vol. 9, World Scientific.
- [9] LIN D., LIU J. and LIU Z. *Mathematical Research Software: ELIMINO*. Proceedings of ASCM'98. pp. 107 - 116, Lanzhou Univ., China, 1998.
- [10] Simple Object Access Protocol (SOAP) 1.1. W3C Note. 08 May 2000.
<http://www.w3.org/TR/SOAP/>
- [11] Universal Description Discovery and Integration (UDDI) Version 3.0. Published Specification, 19 July 2002. <http://uddi.org/pubs/uddi-v3.00-published-20020719.htm>
- [12] WANG, P S. *Internet Accessible Mathematical Computation*. In the 3rd Asian Symp. on Computer Mathematics (ASCM'98), pp 1-13, Lanzhou Univ., China, 1998.
- [13] WANG, P. S., GRAY, S., KAJLER N., LIN D., LIAO W. etc. *IAMC Architecture and Prototyping: A Progress Report*. Proceedings of ACM ISSAC'01, University of Western Ontario, London, Ontario, Canada, July 22-25, 2001.
- [14] WANG, P. S. *Design and Protocol for Internet Accessible Mathematical Computation*. In Proc. ISSAC'99 (1999), ACM Press, pp. 291-298.

- [15] WANG, P. S., GUO, Q. and LIAO, W. *Mathematics over the Internet/Web: A Protocol-based Approach*. ICM Report No. ICM-200206-0007. <http://icm.mcs.kent.edu/reports/index.html>.
- [16] Web Service Activity inside W3C. <http://www.w3.org/2002/ws/>.
- [17] Web Services Description Language (WSDL) 1.1. W3C Note. 15 March 2001. <http://www.w3.org/TR/wsdl>.
- [18] WU, W. T. *Basic Principle of Mechanical Theorem Proving in Elementary Geometries*, J. Syst. Sci. Math. Sci. 4, 207-235 (1984).
- [19] WU, Y., LIAO, W., LIN, D., WANG, P. S., *Local and Remote User Interface for ELIMINO through OMEI*. Proceedings of International Congress on Mathematical Software (ICMS 2002). World Scientific Press. Aug. 2002.
- [20] ZOU, X. *XMEC: An Extensible Mathematical Encoding Converter*. ICM Technical Report. 2001. <http://icm.mcs.kent.edu/reports/index.html>.
- [21] Graham, S., Simeonov, S., Boubez, T, Daniels, G., Davis, D., Nakamura, Y. and Neyama, R. *Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI*. Sams, 2001.
- [22] W Gropp, E. Lusk, *User's Guide for MPICH, a Portable implementation of mpi*, Argonne National Laboratory, University of Chicago, 1996.