# DisCAS: A Distributed-Parallel Computer Algebra System*

Yongwei Wu[1], Guangwen Yang[1], Weimin Zheng[1], and Dongdai Lin[2]

[1] Department of Computer Science and Technology,
Tsinghua University, Beijing, 100084, China
[2] State Key Laboratory of Information Security, Institute of Software,
Chinese Academy of Sciences, Beijing, 100080, China.

**Abstract.** The DisCAS system employs and interacts with multiple ELIMINOs distributed over the Internet to achieve a distributed-parallel computing environment. ELIMINO is a computer algebra system developed to support Wu's method for computing characteristic sets of polynomials and for other related operations. GridPPI, an MPI-like interface for grid computing, could couple multiple computing tools distributed over grid to run complex computing problems. DisCAS combines grid technology, GridPPI and ELIMINOs to deliver high performance computing to Internet users. The overall ELIMINO, GridPPI, and grid technology, as well as the DisCAS architecture are presented. The way to access and apply DisCAS and related works are also discussed at last.

## 1 Introduction

ELIMINO [7] is a new computer algebra system being developed at the Key Laboratory of Mathematics Mechanization, Chinese Academy of Sciences. Capabilities of ELIMINO include manipulation of multi-precision numbers and polynomials, computation of characteristic sets in Wu's method [9], 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 set problems can take a very long time to solve. Consequently,it is reasonable and promising to use many ELIMINOs distributed over Internet to improve the computing performance. DisCAS, a distributed-parallel computer algebra system, aims to use multiple ELIMINOs over Internet 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

* This Work is supported by NSFC (60373004,60373005) and China Postdoctoral Foundation

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3039, pp. 295–302, 2004.

grid can provide high performance symbolic computing services for users as an integrated system: DisCAS.

GridPPI[17] is a coarse-grained distributed parallel programming interface (PPI) for grid computing. As a MPI-like programming model[12], GridPPI provides a group of generic and abstract function prototypes with well-specified semantics. It supports high-level dynamic parallel tasking over grid too. Through GridPPI, users could couple multiple ELIMINOs distributed over multiple heterogeneous machines to run practical complex computing applications.

By adopting GT and GridPPI, DisCAS achieves the following specific results with minimal effort.

- Delivering the powerful distributed-parallel symbolic computation to Internet users.
- Demonstrating grid computing as a way to speed up computer algebra systems.
- Parallelizing GCD, factorization and characteristic-sets based computations over the Internet.
- By implementing GridPPI for DisCAS, providing one MPI-like programming model for users.
- Achieving interoperability with other GridPPI compliant systems, such as grid operation system *TsingHua University Grid* (THUG)[18].

We begin with an overview of the ELIMINO system, and the GridPPI specification. We then introduce the *Open Grid Service Architecture* (OGSA) and GT. These pave the way for describing the architecture and implementation of DisCAS. Use of DisCAS and related works are then discussed.

## 2 ELIMINO

IELIMINO is a new computer-mathematics research system developed at the Key Laboratory of Mathematics Mechanization, Institute of Systems Science, Chinese Academy of Sciences, as part of the "Mathematics Mechanization and its Applications" project. A long-standing goal at MMRC is to automate Wu's method independent of existing computer algebra systems. 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.

To facilitate mathematical research, ELIMINO is kept open and flexible. The architecture of ELIMINO consists of three parts (see Figure 1):

- **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.

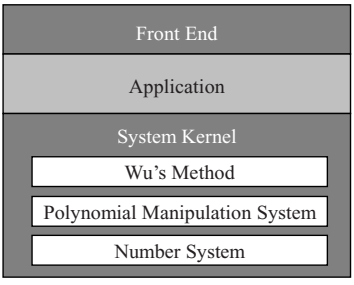


**Fig. 1.** ELIMINO system Architecture

- **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.

## 3 GridPPI Specification

GridPPI[17] is extension of *Open Mathematical Engine Interface*OMEI[5,6]. It aims to be an *application programming interface* (API) general enough to work for most grid computing environments. It specifies a set of function prototypes together with their syntax and semantics to give a MPI-like programming level interface for computing engines. These function prototypes supports all operations that are necessary for secure access and coordinated use of multiple computing tools, including service discovering and selecting, task submitting and reporting, communication between subtasks, atom task executing and status report, etc..

GridPPI supports task-level dynamic parallel tasking over grid too. Through GridPPI, users could couple multiple computing tools distributed over multiple heterogeneous machines to run practical complex computing applications. As an attempt in standardizing programming interface for grid computing, GridPPI achieve several objectives:

– **Achieving Cooperative Use of Multiple Compute Engines**
GT framework can make heterogeneous machines internet accessible. Compute engines over these machines can be called through GridPPI easily, and more, can be cooperatively used.
– **Providing a MPI-like Programming Model**
MPI[12] programming model is easy accepted for most high performance computing requirers. GridPPI provides one distributed-parallel computing interface available over the Internet.

- **Application Portability**
  An application or user interface developed using any GridPPI-compatible interface would be portable among different grid systems, as long as those systems have GridPPI drivers available.
- **Integration of Different Grid Systems**
  Since an application can access multiple grid system by loading multiple GridPPI drivers, an integrated grid system with more powerful and combined capabilities can be accomplished under GridPPI programming model.

## 4 Open Grid Service Architecture and GT3

The grid [1,2] is a virtual supercompter 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 [11] 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)[14]. 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 2 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].

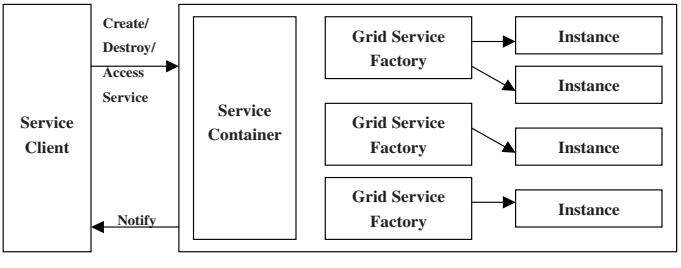


**Fig. 2.** Architecture of GT3

The GT3 *Service Container* (Figure 2) listens for incoming service requests. For a create-service request, the service container first performs security checks.

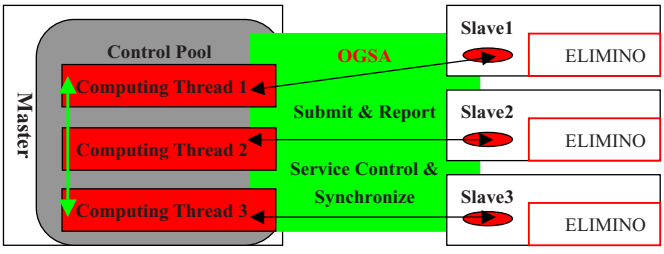


**Fig. 3.** DisCAS Architecture

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 manages all service instances of a specialized grid service.

## 5 DisCAS Architecture and Implementation

Through OGSA, multiple ELIMINO engines distributed over a grid can provide powerful computing services for Internet users as a virtual supercomputer. Figure 3 shows the architecture of DisCAS. It is a master-slave arrangement. The master program, on the client side, instantiates and controls multiple slave ELIMINO servers, each with a 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 responsibility of the control thread is to supply service control and synchronization, communication between threads.

Developers can easily create and access the ELIMINO computing services following the GridPPI compliant API Just like writing an MPI [12] 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 *remote drivers*. The remote drivers in turn access computing service through a grid service locator. Figure 4 shows the control flow of a computing thread in the pool.

As shown in Figure 4, an *ELIMINO server* is an ELIMINO deployed as a grid service through a local driver. This service is mapped to a GSH (service locator

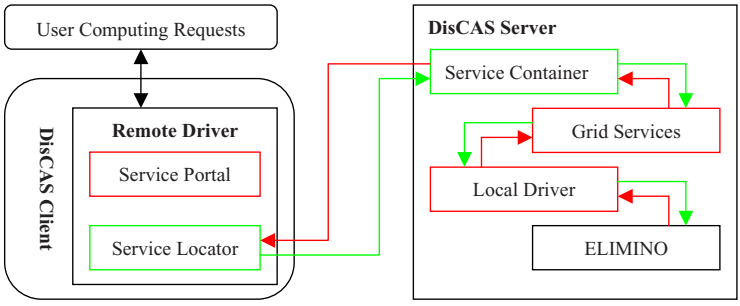


**Fig. 4.** Control Flow of One Computing Thread

in Figure 4) held by the DisCAS master through the GT3 service container, DisCAS can deliver the high performance computing power to remote users.

## 6 Use of DisCAS

Figure 5 shows the user interface we developed to access DisCAS. 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.

Another way is Java programming through GridPPI. This way is much more flexible and practical for users. Developers could use the implementation of GridPPI we provided to couple multiple ELIMINOs over the Internet to complete the complex symbolic computation.

## 7 Related Works

By all means there have been various attempts to provide a distribute-parallel computing environment through grid technology. Many such efforts have been collected and catalogued by the grid application research group of the Global Grid Forum. Among the most famous and similar with our DisCAS are PSE and IAMC. PSE also provides an API for grid-based computing. Users could couple multiple computing nodes, potentially of different architectures, to run chemical computational problems through PSE[13]. IAMC[8,4] aims to make mathematical computations accessible easily and widely over Internet.

PSE (Problem Solving Environments)[13] inherits some interesting solutions exploited in Charlotte [16] and NetSolve[15]. It is designed to provide all the computational facilities needed to solve a target class of problems over a grid.

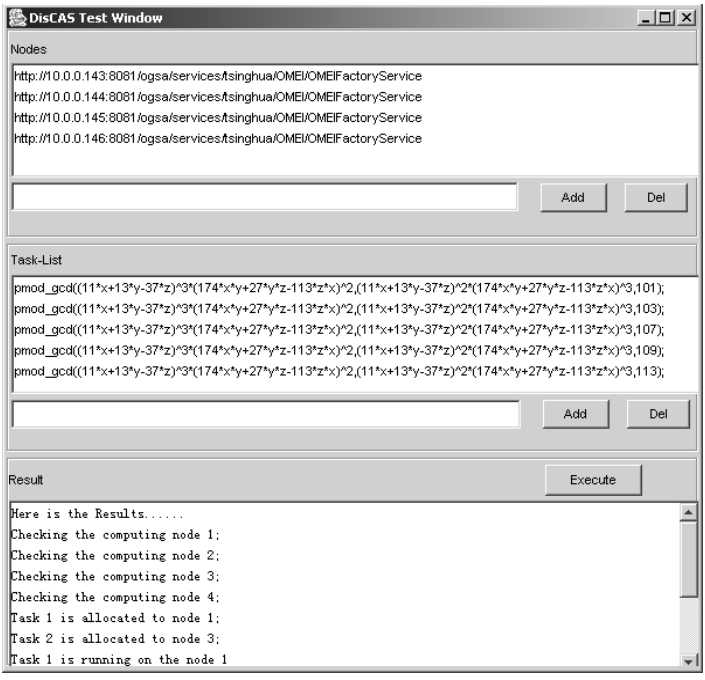


**Fig. 5.** DisCAS User Interface

As one chemical application grid, PSE also supply a completely transparent support to the user who does not have to care about the localization and the allocation of computing resources.

Internet Accessible Mathematical Computation (IAMC)[8] is a research project at the Institute of Computational Mathematics (ICM) at Kent State University. The goal of IAMC is to make mathematical computations Internet accessible easily and widely. By loading multiple OMEI [6] drivers, it can also access multiple compute engines. IAMC is an interactive computing environment over Internet. It gets the users' single computing request from the IAMC client and sends the request to one compute engine one by one.

## 8 Conclusion and Future Work

By using grid technology, DisCAS integrates multiple ELIMINOs distributed over the Internet to provide high performance computing services for remote users. It provides parallel GCD, factorization and characteristic-set based computations. By implementing GridPPI, DisCAS provides one MPI-like programming model for users. At the same time, DisCAS could achieve interoperability with other GridPPI compliant systems, such as grid operation system THUG.

The design and implementation of DisCAS is not final. THUG provides one testbed for our DisCAS. On-going work on DisCAS include design refinements, an efficient grid task manager, reliability and performance test.

## 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. LIAO, W. and WANG, P. S. *Building IAMC: A Layered Approach*, Proc. International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'00), (2000), 1509-1516.
5. 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.
6. LIAO W., LIN D. and WANG P. S. *OMEI: Open Mathematical Engine Interface*, Proceedings of ASCM'2001, pp 83-91, Matsuyama, Japan, (2001)
7. LIN D., LIU J. and LIU Z. *Mathematical Research Software: ELIMINO.* Proceedings of ASCM'98. Lanzhou Univ., China, (1998), 107-116
8. WANG, P. S. *Design and Protocol for Internet Accessible Mathematical Computation.* In Proc. ISSAC'99, ACM Press, (1999), 291-298.
9. WU, W. T. *Basic Principle of Mechanical Theorem Proving in Elementary Geometries,* J. Syst. Sci. Math. Sci. 4, (1984), 207-235
10. 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)
11. 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)
12. W Gropp, E. Lusk, *User's Guide for MPICH, a Portable implementation of mpI*, Argonne National Laboratory, University of Chicago,(1996)
13. Baraglia, R., Laforenza, D., Lagana, A., *A Web-based Metacomputing Problem-Solving Environment for Complex Applications*, Proceedings of Grid Computing 2000, (2000), 111–122
14. Tuecke, S., Czajkowski, K., Foster, I. , et.al.: *Open Grid Services Infrastructure (OGSI) Version 1.0*, Global Grid Forum Draft Recommendation. (2003).
15. Casanova H., Donfarra, J., *NetSolve: A network Server for Solving Computational Science Problems*, Intl. Journal of Supercomputing Application and High Performance Computing, **11(3)** (1998)
16. Baratloo, A., Karaul, M., *Charlotte: Metacomputing on the Web, Special Issue on Metacomputing*, Future Generation Computer Systems, (2001) 559-570.
17. Guangwen Yang, Yongwei Wu, Qing Wang, Weiming Zheng, *GridPPI: Task-level Parallel Programming Interface for Grid Computing*, Accepted by International Journal of Grid and Utility Computing, (2003)
18. Dazheng Huang, Fei Xie, Guangwen Yang, *T.G.: a Market-oriented Computing System with Fine-grained Parallelism*, 9th Workshop on Job Scheduling Strategies for Parallel Processing Seattle, Washington, (2002)