Utilization of Grid and Modern Artificial Intelligence Technologies in Agriculture

Péter SALGA

University of Debrecen,
Department of Business- and Agricultural Informatics

Abstract

Agricultural decisions and rural demands frequently require that several different types of data be processed, accessed and integrated using enormous computing power. Emerging Grid technology offers a powerful mechanism for assembling and processing the requisite data for different applications e.g. decision support, knowledge sharing or monitoring systems.
The Grid is expected to bring together geographically and organizationally dispersed computational resources, such as CPUs, storage systems, communication systems, real-time data sources and instruments, and human collaborators.
The application of artificial neural networks and fuzzy logic is wide spread in several disciplines. These tools borrow great efficiency to traditional scientific methods.
This paper overviews the basic concepts associated with Grid technology and Artificial Intelligence, with a focus on Grid computing. It presents a set of possible application in the topic of Agriculture. Also, a computational Grid facility built on the NorduGrid ARC middleware at the University of Debrecen, Department of Business- and Agricultural Informatics is described.

Keywords

GRID, neural networks, fuzzy logic, cluster, cluster programming, Java

Introduction

The last decade has seen a substantial increase in commodity computer and network performance, mainly as a result of faster hardware and more sophisticated software (*Figure 1*). Nevertheless, there are still problems, in the fields of science, engineering, and business, which cannot be effectively dealt with using the current generation of supercomputers. In fact, due to their size and complexity, these problems are often very numerically and/or data intensive and consequently require a variety of heterogeneous resources that are not available on a single machine. A number of teams have conducted experimental studies on the cooperative use of geographically distributed resources unified to act as a single powerful computer.
This new approach is known by several names, such as metacomputing, scalable computing, global computing, Internet computing, and more recently peer-to-peer or Grid computing. The early efforts in Grid computing started as a project to link supercomputing sites, but have now grown far beyond their original intent. In fact, many applications can benefit from the Grid infrastructure, including collaborative engineering, data exploration, high-throughput computing, and of course distributed supercomputing.
The stated goal of Grid computing is to create a worldwide network of computers interconnected so well and so fast that they act as one.

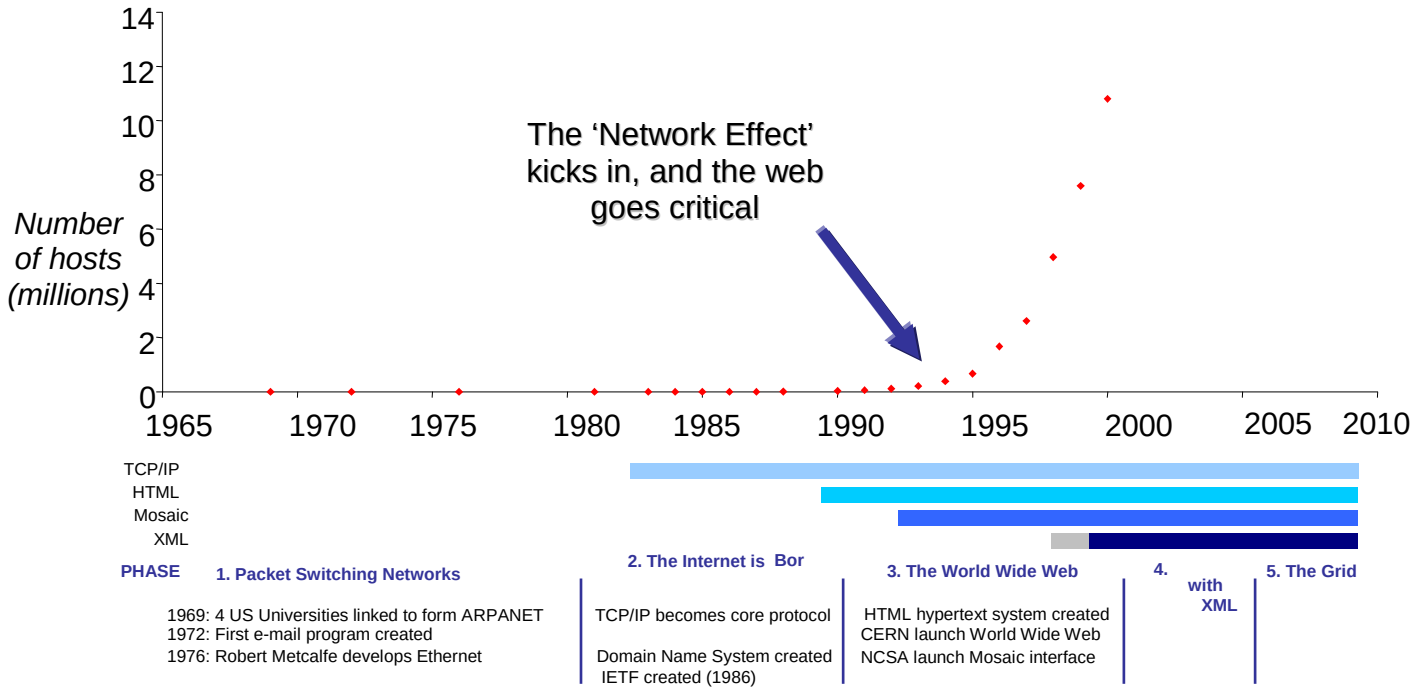

Figure 1. Past, Present, Future [2]

The decision to build a distributed computing system to deal with this deluge of data predates the hype about Grid technology and is purely pragmatic: it would be difficult to fund the necessary computational power and storage capacity if it were concentrated on one site. If, on the other hand, the computations are distributed among the hundreds of institutes worldwide that are involved in the Grid, each institute can tap into national or regional funding sources to raise cash, spreading the pain.

What is Grid?

Computer systems using multiple processors have existed for decades in various forms, the most common of which have been multiprocessing servers and mainframes. The advent of inexpensive, high-performance processors has provided impetus to the development of multiprocessor designs.

The terms 'Parallel processing', 'parallelization' or 'distributed programming' all refer to the system where a complex task is broken up into many subtasks that are to be run in parallel. Each subtask is then assigned to a CPU on the network and the results are combined.

Distributed programming uses a collection of computers connected over a network to solve a single problem. Programming multi-computers requires models which are different from normal systems. The programmer must be able to transfer data between different parts of the program through a shared memory space and to coordinate efforts through an inter-process communications system capable of communication between interconnected CPUs.

Distributed programs achieve the following:

- Increased processing speed by using more than one computer at a time.
- Potential for improved reliability when additional computers can compensate for the failure of one
- Allowance for some problems, like remote data acquisition, to succeed in a distributed environment

A *cluster* is a group of individual, stand-alone computers that work together in parallel way and that outside systems view as a single computing resource. The individual systems (*nodes*) that make up the cluster communicate with each other via high-speed connections such as Gigabit Ethernet, ATM or a proprietary link. For easier management, clusters use special software to coordinate and manage their activities, depending on how they are used. Clusters are particularly well suited to meeting the needs of high-availability, load balancing and scientific computing.

Cluster computing using machines connected in a finite physical network. These are PCs similar in both hardware and software. In order to solve massive computational problems most networks are not big enough. The Grid computing is the answer to this problem.

While clusters are groups of computers tied together as a single device, *Grid*s consist of multiple systems that work together while maintaining their distinct identities.

Figure 2. shows the TeraGrid, a large high-performance computing project headed by the National Center for Supercomputing Applications at the University of Illinois at Urbana-Champaign. The TeraGrid uses thousands of Intel Itanium 2 processors located at four sites in the United States.

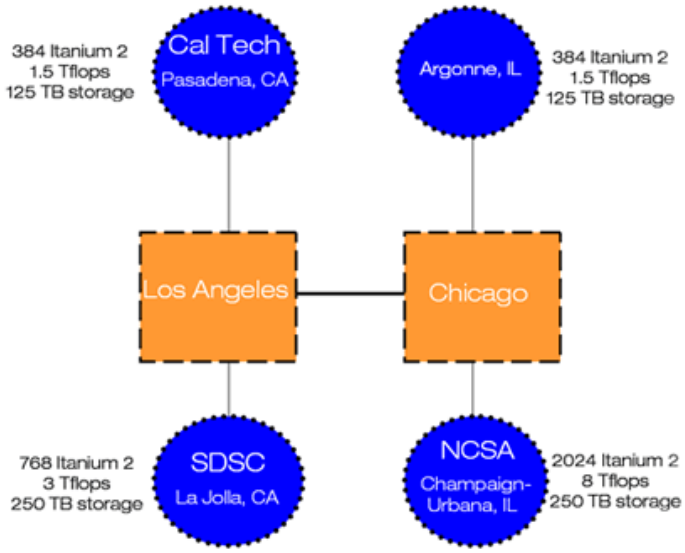

Figure 2. The TeraGrid project uses multiprocessing systems at four sites. [8]

Grids virtualize resources, such that a user logging on at Caltech in *Figure 2.* would view the processors at the other sites as though they were part of the local system. In fact, of course, they are not. They are in a separate administrative domain but available as a virtual local resource to any node on the Grid. T*he constituent systems are administered separately and are physically distinct from each other.* This is not true for any of the previous designs discussed so far.

Like clusters, Grids use high-speed interconnects to link the various systems. Because of the latencies inherent in long distances, however, Grids often partition tasks to minimize the need for long-distance messaging. Grids enjoy common benefits with clusters: they can be built from inexpensive, off-the-shelf parts, and they can be expanded nearly endlessly, that is why often called the poor man's Supercomputer.

The simplicity of expanding a Grid is emerging as a compelling proposition for businesses. In a scenario promoted by several vendors, all the computers at a business would be part of one Grid. When PC and servers are not otherwise engaged in work, they can be dedicated to attacking grid-level computing tasks.

Application of Grid technology

Grid computing is made up of computational and data intensive problems. The computational aspect focuses on reducing execution time of applications that require large amounts of computer processing cycles. Data intensive problems require large scale data management methods to transfer the data needed for solving the problem to the machine assigned to solve it. Data intensive applications such as High Energy Physics and Bioinformatics require both computational and data management solutions to be present in Grid computing solutions. Another power of Grid that it can provides a uniform interface for researchers to common use instruments and infrastructure.

Grid offers a way to solve Grand Challenge problems like:

- Protein folding
- Drug discovery
- Financial modelling

- Earthquake simulation
- Climate / weather modelling
- Optimization studies

Grid computing can use the Internet to borrow unused CPU cycles and storage from millions of systems across a worldwide network. This flexible, readily accessible pool can then be harnessed by anyone who needs it, much as power companies and their users share the electrical grid. Grid computing leans more to dedicated tasks, such as single large medical and engineering problems, rather than for general, everyday jobs. Sun defines a computational grid as "a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to computational capabilities." The computers on a Grid can be of many different OS and hardware platforms.

Generally Grids are classified by function:

- Computational Grids (including CPU scavenging grids) referred - Computational Grids typically gain and lose machines at unpredictable times as interactive users start or stop using their machines, new machines are purchased, machines are removed from the network, or break down. Cycle-scavengers move jobs from machine to machine as necessary to allow the smooth running of the job and the network being scavenged
- Data Grids - Cycle-scavenging systems use machines purchased for other purposes to run batch jobs at night, weekends, and other idle times. A data grid is a Grid computing system that considers access to distributed data as important as access to distributed computational resources. Many distributed scientific and engineering applications require access to large amounts of data -- often terabytes or even petabytes of data

It’s expected that in the future applications will require even more widely distributed access to data. Data Grids will have to support scientific collaboration in a virtual environment allowing access around the world by many people. A Grid is a distributed collection of computer and storage resources maintained in a Virtual Organization (VO). Any of the authorized users within that VO has access to all or some of these resources, and is able to submit jobs to the Grid and expect responses.

Why Nordugrid?

The main Grid systems in Europe were observed by experts of our workgroup. Our main aspects were the international connectivity, functionality and functioning, and the user- and admin-friendly configuration and management. The *NorduGrid* project and the *NorduGrid ARC* (Advanced Resource Connector) fulfils all of the conditions, and we found it is the only well-working international Production Grid in Europe.

The NorduGrid architecture’s basic components are the user interface, information system, computing cluster, storage element, and replica catalog. The NorduGrid’s user interface includes high-level functionality namely resource discovery and brokering, Grid job submission and job status querying. NorduGrid thus does not require a centralized resource broker. The user interface communicates with the NorduGrid grid manager and queries the information system and replica catalog. Users can install the user interface client package on any machine, using as many interfaces as they need.

The information system is a distributed service that serves information for other components, such as monitors and user interfaces. The information system consists of a dynamic set of distributed databases that are coupled to computing and storage resources to provide

information on a specific resource's status and operates on a pull model: when queried, it generates requested information on the resource locally (optionally caching it afterward). Local databases register to a global set of indexing services via a soft-state registration mechanism. For direct queries, the user interface or monitoring agents contact the indexing registries to find contact information for local databases. The computing cluster consists of a front-end node that manages several back-end nodes, typically through a private closed network.

The software component is a standard batch system, with an extra layer that acts as a grid interface and includes the grid manager, the GridFTP server and the local information service. Although Linux is the operating system of choice, Unix-like systems, including Hewlett-Packard's UX and Tru64 Unix, can be used as well. The NorduGrid does not dictate batch system configuration; its goal is to be an add-on component that hooks local resources onto the Grid and lets Grid jobs run along with conventional jobs, respecting local setup and configuration policies. The cluster has no specific requirements beyond a shared file system (such as Network File system) between the front- and back-end nodes. The back-end nodes are managed entirely through the local batch system; no grid middleware is required on them. To register and locate data sources, the Globus project's replica catalog was modified to improve its functionality. The catalog's records are primarily entered and used by the grid manager and the user interface. The user interface can also use the records for resource brokering.

The development objectives:

- enabling an efficient and scalable distributed data-management systém;
- further developing resource discovery and brokering to allow interoperability with complex local scheduling policie;
- improving the authorization and authentication system, while maintaining its flexibility;
- modifying the information system to accommodate heterogeneous clusters, providing more sophisticated caching mechanisms; and• developing and implementing bookkeeping and accounting services, as well as a user-friendly Grid portal.

The NorduGrid Toolkit is freely available at www.nordugrid.org as RPM distributions, source tar-balls, and as CVS snapshots and nightly builds.

Figure 3. – the Grid Monitor - shows the actual activity of clusters involved in NorduGrid and other technical data:

- 40 clusters
- ~4000 CPUs
- 10 countries
- storage: 42 TB total, 28 TB free
- more than 1000 users

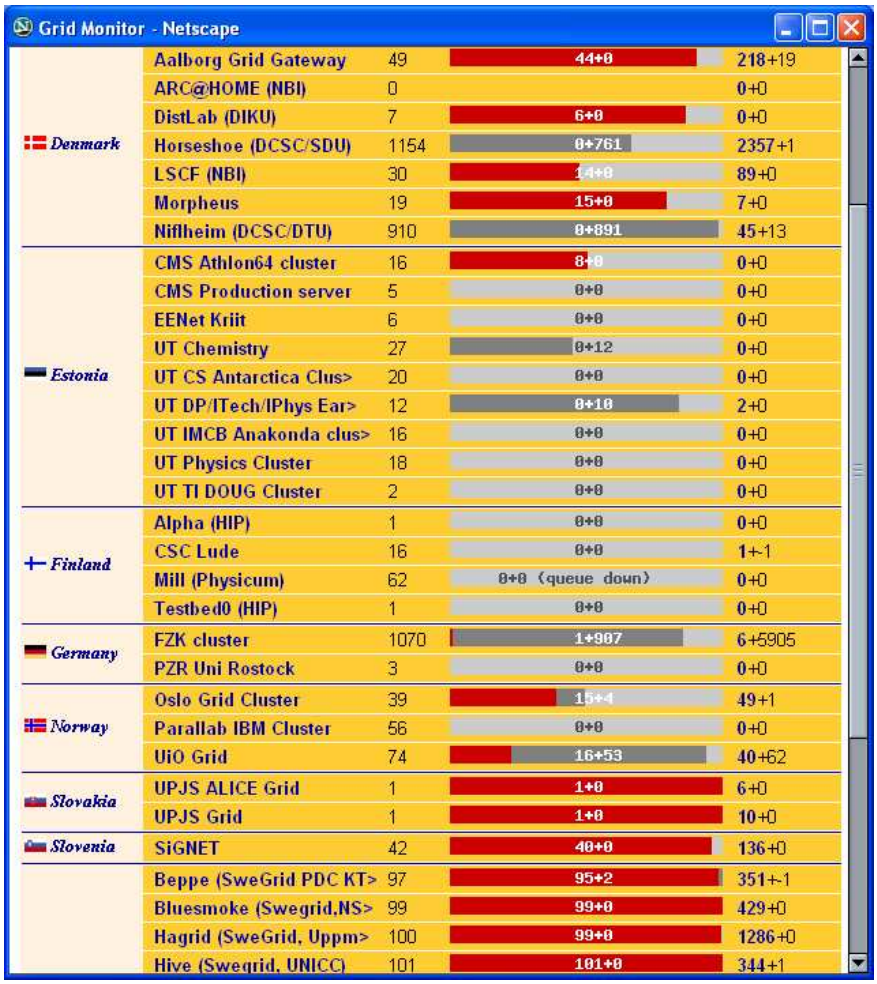

Figure 3. The NorduGrid Grid Monitor

Artificial Intelligence – what good is it?

In this paper we used the concept of Artificial Intelligence for Neural Networks and Fuzzy Logic Systems.

Neural networks can represent complex nonlinear relationships, and they are very good at classification of phenomena into preselected categories used in the training process.

Fuzzy logic systems address the imprecision of the input and output variables directly by defining then with fuzzy numbers that can be expressed in linguistic terms (e.g., cold, warm, and hot). Furthermore, they allow far greater flexibility in formulating system descriptions at the appropriate level of detail.

The efficiency of an enterprise depends on its working parameters. A good expert can advise on production-efficiency based on few important data. Obviously, this derives from the depending of parameters, and if we know every condition, we can almost perfectly determine the efficiency. In the head of the expert, in place of thorny, mathematical and exact formulas, there is a simple deduction-chain crystallized from his experience. Neural network uses sophisticated optimization procedures to speed up the “thinking process”.

Some modules can connect to the data of the Enterprise Resource Planning (ERP) system and can extract the usable information for the analysis. The system works as a knowledge engineering system, but the supplied basic artificial intelligence analog system can be useful on different areas of business and production:

- Data processing (fraud and fault detection, speeding up the data-processing)
- Evaluation (optimization of rules, speeding up the evaluation)
- General analysis (continual observation of parameters and tracking of theirs changes).

The system can find the main reasons for increasing or decreasing production because it can also study the most extreme analogous signals. Similar to human thinking, the system analyzes the processes based on patterns, so it is capable of observing nonanalytical objects. [10]

Application of Grid technology and Artificial Intelligence in Agriculture

There are lot of decision situations in the life of a farm. Examples for *strategic* decisions are:

- Which crop or variety to plant?
- Whether to dam a river?
- What land use is appropriate?

Operational decisions:

- Whether to spray a crop to protect from disease?
- When to harvest?
- How much to irrigate?
- Where will flooding occur?

Not an easy mission to support similar type of decisions because the necessary data stored in heterogeneous databases managed by several different ownerships. No chance whatever to centralize data-storage units but there is a strong necessity of integration of various databases. Another effect in this topic is the commercial competition which is an additive difficulty for information-support.

The Grid is a plausible way for data integration because it serves:

- Uniform interface for queries
- High storage capacity
- Processing power for optimization programs (neural networks, genetic algorithms)
- Accessibility rights and policies to data
- Accounting system for data-sellers and costumers
- Sharing of instruments

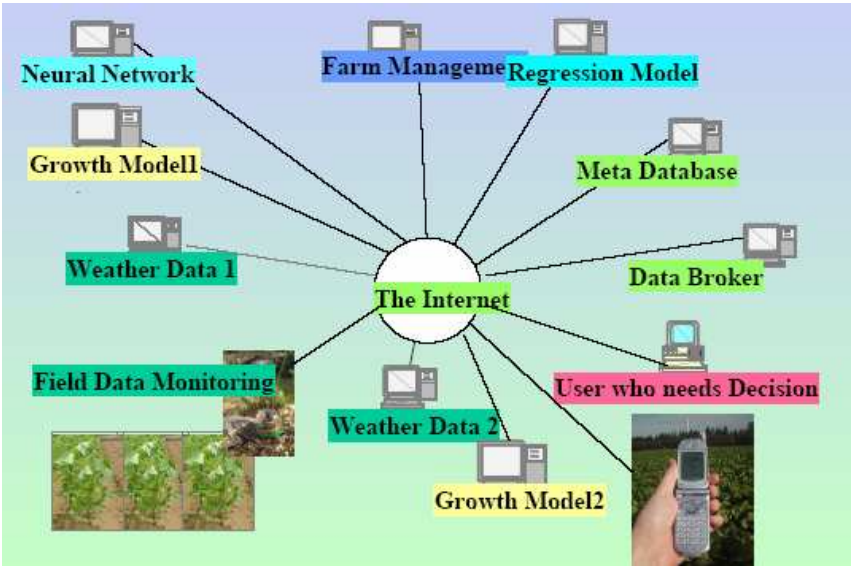

Figure 4. Model of AgriGrid [4]

As *Figure 4.* shows, with the help of AgriGrid system the farmer can call information for irrigation or spraying based on images taken by his mobile phone and the geographical position of area. This system adds the following advantages to the Grid:

- Maximize data and application usage without centralization
- Easy Maintenance
- Easy updates of data and applications
- Easy addition of new data and applications
- Flexible and dynamic integration of data and applications

This model can help to integrate the decision data for agriculture and usable also in development phase, in this way we can reduce the costs of development.

The evaluation of data based on the pattern recognition ability of neural networks, the fuzzy logic serves a very important role in the processing of incorrect and often incomplete data.

Initiative of Hungarian AgriGrid

At the University of Debrecen, Department of Business- and Agricultural Informatics a Fedora C2 Linux cluster was built. We use the PBS based *Torque* cluster management tool, and the connection to the NorduGrid system is under development (at this moment Hungary doesn't have yet Grid certificate which allows of official connection to international systems). In the cluster multiple programming environments were installed: C, C++, MPI and Java. The distributed treatment of Java programs is managed by *J2EE* and *JBoss* application servers.

The following is a clustering feature overview for JBoss 3.0:

- Automatic cluster membership discovery.
- Fail-over and load-balancing features for JNDI, RMI, Entity Beans, Stateful Session Beans with in-memory state replication, and Stateless Session Beans.
- Pluggable load-balance policie.
- HTTP session replication with Tomcat and Jetty (CVS HEAD only).
- Dynamic JNDI discovery. JNDI clients can automatically discover the JNDI InitialContext.
- Cluster-wide replicated JNDI tree.

- Network Boot.
- Farming: Distributed cluster-wide hot-deployment.

These features make the JBoss 3.0 application server to a very useful cluster programming tool and enable the application of portable code in distributed environments.

We planned and designed the architecture to satisfy the needs of future users. These needs constitute a general guiding philosophy:

- Start with something simple that works.
- Avoid single points of failure.
- Give resource owners full control over their resources.
- Ensure that developed software can use the existing system and, eventually, other preinstalled versions.

Conclusions

The potential of computer science has always been hampered by the inability to adequately address massive processing and data volume issues. No matter how fast a CPU is or the data throughput rate, our imaginations come up with new applications that exceed the existing technology or budget.

Grid computing technology has the potential to alleviate processing capacity and cost barriers. A Grid can solve problems that can't be approached without an enormous computing power. Computers will collaborate rather than being directed by one managing computer. Ultimately, the future may bring pervasive computing; computers will be saturating our environment without our direct awareness. Recent Internet over power grid developments may further increase Grid computing use by making high speed connection ubiquitous.

The Java programming language successfully addresses several key issues that accelerate the development of Grid environments, such as heterogeneity and security. It also removes the need to install programs remotely; the minimum execution environment is a Java-enabled Web browser. Java, with its related technologies and growing repository of tools and utilities, is having a huge impact on the growth and development of Grid environments. From a relatively slow start, the developments in Grid computing are accelerating fast with the advent of these new and emerging technologies. It is very hard to ignore the presence of the Common Object Request Broker Architecture (CORBA) in the background. We believe that frameworks incorporating CORBA services will be very influential on the design of future Grid environments. The two other emerging Java technologies for Grid and P2P computing are Jini and JXTA. The Jini architecture exemplifies a network-centric service-based approach to computer systems. Jini replaces the notions of peripherals, devices, and applications with that of network-available services. Jini helps break down the conventional view of what a computer is, while including new classes of services that work together in a federated architecture. The ability to move code from the server to its client is the core difference between the Jini environment and other distributed systems, such as CORBA and the Distributed Common Object Model (DCOM).

The main application areas of today’s Grids are the biotechnology, high energy physics, but the structure and demands of agriculture makes it a very appealing field for Grid technology. The enormous processing power of clusters is not the most applicable property in agriculture, the main advantages are the common interface, the sharing of instruments and the data integration facility of datagrid technology. The soft programs (neural networks, fuzzy

systems) developed in java environment can run under appropriate cluster to make very easy the deployment of decision support applications.

Literature

[1] Buyya, R. ed. (1999). High Performance Cluster Computing: Architectures and Systems. Volume 1, Prentice Hall, Old Tappan.

[2] Buyya, R. (April 2002). Economic-based distributed resource management and scheduling for Grid computing. PhD Thesis, Monash University, Melbourne, Australia.

[3] Laurenson, M. R., Yamakawa, A., Meng, H., Kiura, T., Wang, J. and Ninomiya, S. (August 2004). Integration of Data Broker Web Services for Agricultural Grid. AFITA 2004, Bangkok, Thailand.

[4] Ninomiya, S., Laurenson, M. (March 2003). A Grid for Efficient Decision Support in Agriculture. International Symponium Grid Computing, Taipei, Taiwan.

[5] Kónya, B. (Nov 2004). Advanced Resource Connector (ARC) – The Grid Middleware of the NorduGrid. Lecture Notes in Computer Science, Volume 3241, Page 10.

[6] Kacsuk, P., Kónya, B., Stefán, P. (Nov 2004). Production Grid Systems and Their Programming. Lecture Notes in Computer Science, Volume 3241, Page 13.

[7] Burger, T. W. Trends in Distributed Computing, http://www.intel.com/cd/ids/developer/asmo-na/eng/95223.htm

[8] Binstock, A. Multiprocessors, Clusters, Grids, and Parallel Computing: What's the Difference? http://www.intel.com/cd/ids/developer/asmo-na/eng/95581.htm

[9] (Oct 2004). One Grid to Rule Them All, Economist Magazine.

[10] P. Salga, R. Szilagyi, T. Toth, A. Pokol: Application of CNN-GRID technology, Information Systems for Agriculture, Forestry and Rural Areas - enlargement of EU 10th International Conference 2004, Seč u Chrudimi, Czech Republic