pressing industrial and scientific problems.

Recently, the use of high-performance computing techniques has led to the development of methods for building and accessing VLKBs with the faster processing and large memories enabled by modern supercomputers. Whereas entering the knowledge base for CYC has taken tens of person-years, these new techniques permit the automatic generation of VLKBs in much shorter times. In addition, new accessing techniques provide searches that are several orders of magnitude faster than serial algorithms for matching complex patterns on relatively unstructured data. Although these techniques may not obviate the need for CYC-like projects (after all, common sense is hard to find, even among people), they open many intriguing possibilities. Some examples are mentioned here:

1) *Large case-based systems*. Instead of solving problems from scratch, systems can solve new problems by analogy to previous solutions (2). Such "case-based" reasoning requires very large memories of previous problem solutions. Building these memories by hand can be an enormously difficult knowledge engineering task. Recent work has shown that large case bases can be automatically generated with AI techniques (3) and accessed extremely efficiently by parallel inferencing techniques (4).

2) *Hybrid knowledge and databases*. Many large corporate and scientific databases can be used to create AI knowledge bases. First, specific information about the domain of discourse is used to encode knowledge about the underlying characteristics and functions of objects in that domain. Following this, traditional database queries are used to create a knowledge base relating specific instances from the database to the more generic AI knowledge. The resulting hybrid knowledge and database can be used to combine searching and inferencing, with supercomputing techniques again providing efficient pattern-matching capabilities that are difficult to encode and inefficient to run in the unaugmented database. This technique is particularly important in applications where old data must be explored in novel ways to see if recently discovered patterns were previously existent in the database (examples include epidemiology and pharmaceutical research).

3) *Software agents*. A recent innovation in AI technology is the creation of intelligent agents to help users explore complex unstructured information, such as that in the millions of documents distributed across the Internet. Although not yet a commercial technology, software agents are expected to become an important mechanism in providing access and navigation aids to the large amounts of information stored in so-called cyberspace. Current Internet agents, for example, provide knowledge-based interface tools for making the net more user friendly (5) and use AI techniques to help filter out vast amounts of irrelevant information (6). A recently started project in my laboratory, for example, focuses on the use of parallel inferencing techniques to provide a basis for creating agents that wander through the network, explore the information residing therein, and process it to create a large, knowledge-based memory for use by interface, search, and filtering agents.

It is still early in the development stage of this exciting new technology, and much important research remains to be done. High-performance computing support for massive knowledge bases requires the exploration of several areas, which are currently receiving only minimal funding (despite the vast budget resources being put into the national information infrastructure). To continue scaling AI, we must explore the use of secondary storage in ways that are amenable to the irregular memory usage patterns of AI's knowledge-base technology (very different from the more regular accesses used in most scientific supercomputing). New inference classes and access mechanisms must be developed for efficiently exploring the ever larger knowledge repositories available to users. New techniques must also be found for mining the data stored in scientific and medical databases, and reasoning mechanisms must be developed for extending software agents technology, tailoring it to the creation and use of large knowledge-based memories. As these techniques become more commercially viable, we can expect to see the next great stride in the use of AI technology in industrial, government, and scientific applications.

References

1. D. Lenat and E. Feigenbaum, *Artif. Intell.* **47**, 1 (1991).
2. J. Kolodner, *Case-Based Reasoning* (Morgan Kaufman, San Francisco, CA, 1993).
3. B. Kettler, J. Hendler, W. Andersen, M. Evett, *IEEE Expert*, **9** (no. 1), 8 (1994).
4. M. Evett, J. Hendler, L. Spector, *J. Parallel Distrib. Comput.*, in press.
5. O. Etzioni and D. Weld, *Commun. ACM* **37**, 72 (1994).
6. P. Maes, *ibid.*, p. 30.

Enterprise-Wide Computing

Andrew S. Grimshaw

For over 30 years, science fiction writers have spun yarns featuring worldwide networks of interconnected computers that behave as a single entity. Until recently, such fantasies have been just that. Technological changes are now occurring that may expand computational power just as the invention of desktop calculators and personal computers did. In the near future, computationally demanding applications will no longer be executed primarily on supercomputers and single workstations dependent on local data sources. Instead, enterprise-wide systems, and someday nationwide systems, will be used that consist of workstations, vector supercomputers, and parallel supercomputers connected by local and wide-area networks. Users will be presented the illusion of a single, very powerful computer, rather than a collection of disparate machines. The system will schedule application components on processors, manage data transfer, and provide communication and synchronization so as to dramatically improve application performance. Further, boundaries between computers will be invisible, as will the location of data and the failure of processors.

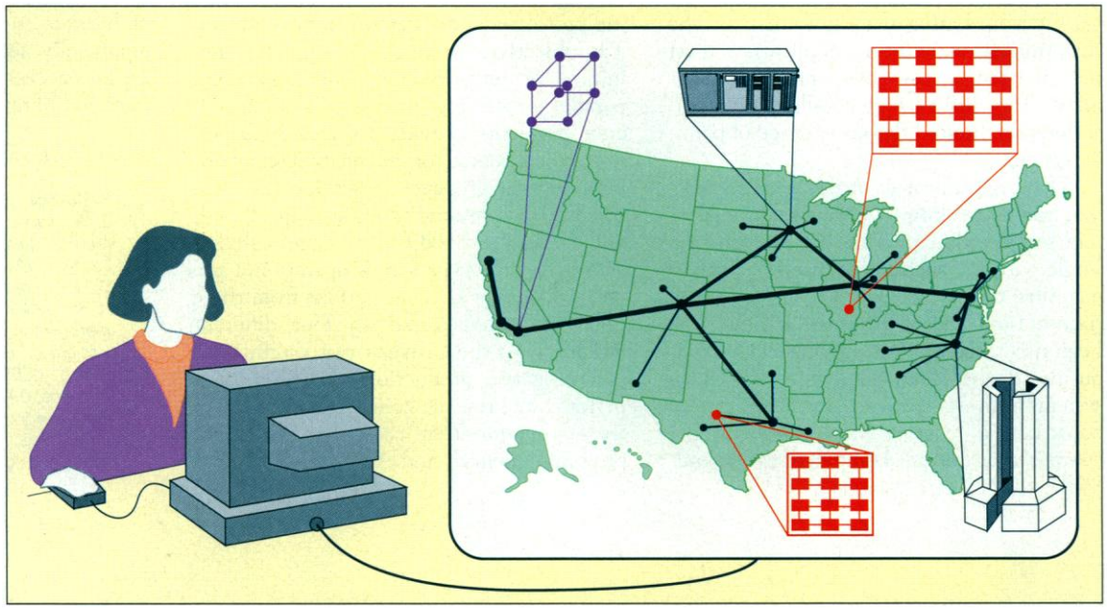
To illustrate the concept of an enterprise-wide system, first consider the workstation or personal computer on your desk. By itself it can execute applications at a rate that is loosely a function of its cost, manipulate local data stored on local disks, and make printouts on local printers. Sharing of resources with other users is minimal and difficult. If your workstation is attached to a department-wide local area network (LAN), not only are the resources of your workstation available to you but so are the network file system and network printers. This allows expensive hardware such as disks and printers to be shared and allows data to be shared among users on the LAN. With department-wide systems, processor resources can be shared in a primitive fashion by remote log-in to other machines. To realize an enterprise-wide system, many department-wide systems within a larger organization, such as a university, company, or national lab, are connected, as are more powerful resources such as vector supercomputers and parallel machines. However, connection alone does not make an enterprise-wide system. If it did, then we would

The author is in the Department of Computer Science, University of Virginia, Charlottesville, VA 22903, USA.

have enterprise-wide systems today. To convert a collection of machines into an enterprise-wide system requires software that makes the sharing of resources such as databases and processor cycles as easy as sharing printers and files on a LAN; it is just that software that is now being developed.

The potential benefits of enterprise-wide systems include more effective collaboration by putting co-workers in the same virtual workplace, higher application performance as a result of parallel execution and exploitation of off-site resources, improved access to data, improved productivity resulting from more effective collaboration, and a considerably simpler programming environment for applications programmers.

Three key technological changes make enterprise-wide computing possible. The first is the much heralded information superhighway or national information infrastructure (NII) and the gigabit (10^9 bits per second) networks that are its backbone. These networks can carry orders of magnitude more data than current systems. The effect is to "shrink the distance" between computers connected by the network. This, in turn, lowers the cost of computer-to-computer communication, enabling computers to more easily exchange both information and work to be performed.



Sharing resources. The user's view of an ideal enterprise-wide system is one of a very powerful monolithic virtual machine that provides computational and data storage services. The user need not be concerned with the details of machine and processor type, data representation and physical location of processors and data, or the existence of other competing users. The user does not see system boundaries, only objects, both application objects, for example, executables and data objects, such as files. Object location and representation are hidden.

The second technological change is the development and maturation of parallelizing compiler technology for distributed-memory parallel computers. Distributed-memory parallel computers consist of many processors, each with its own memory and capable of running a different program, connected together by a network. Parallelizing compilers are programs that take source programs in a language such as High-Performance Fortran and generate programs that execute in parallel across multiple processors, reducing the time required to perform the computation. Depending on the application and the equipment used, the performance improvement can be from a modest factor of 2 or 3 to as much as two orders of magnitude. Most distributed-memory parallel computers to date have been tightly coupled, where all of the processors are in one cabinet, connected by a special purpose, high-performance network. Loosely coupled systems, constructed of high-performance workstations and LANs, are now competitive with tightly coupled distributed-memory parallel computers on some applications. These workstation farms (*1*, *2*) have become increasingly popular as cost-effective alternatives to expensive parallel computers.

The third technological change is the maturation of heterogeneous distributed systems technology (*3*). A heterogeneous distributed system consists of multiple computers, called hosts, connected by a network. The distinguishing feature is that the hosts have different processors (80486 versus 68040), different operating systems (Unix versus VMS), and different available resources (memory or disk). These differences and the distributed nature of the system introduce complications not present in traditional, single-processor mainframe systems. After 20 years of research, solutions have been found to many of the difficulties that arise in heterogeneous distributed systems.

The combination of mature parallelizing compiler technology and gigabit networks means that it is possible for applications to run in parallel on an enterprise-wide system. The gigabit networks also permit applications to more readily manipulate data regardless of its location because they will provide sufficient bandwidth to either move the data to the application or to move the application to the data. The addition of heterogeneous distributed system technology to the mix means that issues such as data representation and alignment, processor faults, and operating system differences can be managed.

Today enterprise-wide computing is just beginning. As of yet, these technologies have not been fully integrated. However, projects are under way at the University of Virginia and elsewhere (*4*) that if successful will lead to operational enterprise-wide systems. For the moment, systems available for use with networked workstations fall into three non–mutually exclusive categories, (i) heterogeneous distributed systems, (ii) throughput-oriented parallel systems, and (iii) response time–oriented parallel systems. Heterogeneous distributed systems are ubiquitous in research labs today. Such systems allow different computers to interoperate and exchange data. The most significant feature is the shared file system, which permits users to see the same file system, and thus share files, regardless of which machine they are using or its type. The single file-naming environment significantly reduces the barriers to collaboration and increases productivity. Throughput-oriented systems focus on exploiting available resources in order to service the largest number of jobs, where a job is a single program that does not communicate with other jobs. The benefit of these systems is that available, otherwise idle, processor resources within an organization can be exploited. Although no single job runs any faster than it would on the owner's workstation, the total number of jobs executed in the organization can be significantly increased. For example, in such a

system I could submit five jobs at the same time in a manner reminiscent of old-style batch systems. The system would then select five idle processors on which to execute my jobs. If insufficient resources were available, then some of the jobs would be queued for execution at a later time. Response time–oriented systems are concerned with minimizing the execution time of a single application, that is, with harnessing the available workstations to act as a virtual parallel machine. The purpose is to more quickly solve larger problems than would otherwise be possible on a single workstation. Unfortunately, to achieve the performance benefits an application must be rewritten to use the parallel environment. The difficulty of parallelizing applications has limited the acceptance of parallel systems.

The Legion project at the University of Virginia is working toward providing system services that provide the illusion of a single virtual machine to users, a virtual machine that provides both improved response time and greater throughput (5). Legion is targeted toward nationwide computing. Rather than construct a full-scale system from scratch, we have chosen to construct a campus-wide test-bed, the campus-wide virtual computer, by extending Mentat, an existing parallel processing system. Even though the campus-wide system is smaller, and the components much closer together, than a full-scale nationwide system, it presents many of the same challenges. The processors are heterogeneous; the interconnection network is irregular, with orders of magnitude differences in bandwidth and latency; and the machines are currently in use for on-site applications that must not be hampered. Further, each department operates essentially as an island of service, with its own file system.

The campus-wide system is both a working prototype and a demonstration project. The objectives are to demonstrate the usefulness of network-based, heterogeneous, parallel processing to computational science problems; provide a shared high-performance resource for university researchers; provide a given level of service (as measured by turn-around time) at reduced cost; and act as a test-bed for the large-scale Legion. The prototype is now operational and consists of over 60 workstations from three different manufacturers in four different buildings. At the University of Virginia, we are using two production applications for performance testing: complib, a biochemistry application that compares DNA and protein sequences, and ATPG, an electrical engineering application that generates test patterns for very large scale integrated circuits. Early results are encouraging. Other production applications are planned.

I believe that projects such as Legion will lead to the widespread availability and use of enterprise-wide systems. Although significant challenges remain, there is no reason to doubt that such systems can be built. The introduction of enterprise-wide systems will result in another leap forward in the usefulness of computers. Productivity will increase owing to the more powerful computing environment, the tearing down of barriers to collaboration between geographically separated researchers, and increased access to remote information such as digital libraries and databases.

References and Notes

1. B. Buzbee, *Science* **261**, 852 (1993).
2. J. A. Kaplan and M. L. Nelson, *NASA Tech. Memo. 109025* (NASA Langley Research Center, Langley, VA, 1993).
3. D. Notkin *et al.*, *Commun. ACM* **30**, 132 (1987).
4. R. Rouselle *et al.*, *The Virtual Computing Environment: Proceedings of the Third International Symposium on High Performance Distributed Computing* (IEEE Computer Society Press, Los Alamitos, CA, in press).
5. For more information on heterogeneous parallel processing, use Mosaic to access http://uvacs.cs.virginia.edu:/~mentat/science.html.