NorthWestNet

User Services Internet Resource Guide (NUSIRG)

Third Edition

March,1992

Section 6: Empowering Scientific Research

On-Line PostScript Version, March 28, 1992

Author: Jonathan Kochmer

Executive Editor: Dr. Eric S. Hood

Technical Editor: Dan L. Jordt

Production Manager: Schele Gislason

Copyright (c) 1991 by the NorthWestNet Academic Computing Consortium, Inc.

A product of NorthWestNet and the Northwest Academic Computing Consortium, Inc.

All rights reserved. NWACC permits the copying of NUSIRG or parts of NUSIRG for research and educational purposes only, provided that appropriate attribution is given to both NWACC and NorthWestNet. The requirement for appropriate attribution will be satisfied by the inclusion of this copyright page with every copy. Resale of NUSIRG is expressly prohibited without written permission from NWACC, Inc. NWACC, Inc. makes no warranty of any kind, expressed or implied, with regard to the contents of this Guide, nor shall be liable in any event for incidental or consequential damages, direct or indirect, resulting from the use of this document.

How To Get NUSIRG

THE NORTHWESTNET USER SERVICES INTERNET RESOURCE GUIDE

You can either get NUSIRG at no charge by the Internet "File Transfer Protocol" (FTP), or you can purchase a bound copy from NorthWestNet. If you have a connection to the Internet, you are encouraged to follow these instructions for using FTP to obtain NUSIRG. If you can't use FTP (or don't know how yet!), instructions on how to order a printed and bound copy from NorthWestNet are included on the next page.

Free Copies of NUSIRG By File Transfer Protocol (FTP)

NUSIRG is available at no charge, by FTP, as text-only or PostScript files. The text-only files are in a simple format which can be printed by most kinds of printers or viewed on-screen with most text editors. The PostScript files produce output which is essentially identical to the bound version of NUSIRG.

If you have access to a computer connected to the Internet, all you have to do is follow the simple instructions below.

Note: commands you type are shown in the left hand column in lower case (don't type the #'s!); UPPER CASE and ">" indicates prompts from the computer that will appear on your screen.

What you type (or what you'll see)	Explanation
1) ftp ftphost.nwnet.net	Connect to the NorthWestNet FTP host.
2) LOGIN: anonymous	At the "LOGIN:" prompt enter "anonymous".
3) PASSWORD: your-user-id@address	At the "PASSWORD:" prompt, please enter your user-id@your.full.internet.address (e.g., robessin@uwvvd.derrg.edu).
4) FTP> cd nic/nwnet/user-guide	Change directories (cd) to the directory containing FTP copies of NUSIRG.
5) FTP> get README.nusirg	Use the "get" command followed by a filename from the list on the next page to copy a file; get "README.nusirg" first, which contains the most up-to-date information about NUSIRG files.
6) FTP> ls	The "ls" command will produce a listing of the names of the files in the NUSIRG directory (see below).
7) FTP> quit	The "quit" command will end your FTP session.

The file "README.nusirg" contains more detailed information about the files in the NUSIRG directory.

Files in the NUSIRG directory	Description of file's contents
README.nusirg	A detailed description of NUSIRG files.
nusirg,Index.ps	Index for all of NUSIRG
nusirg.Index.txt	
nusirg.about.nwnet.and.aup.ps	NWNET & NSFNET backbone information and policies
nusirg.about.nwnet.and.aup.ps	
nusirg.beginner.overview.ps	Overview of NUSIRG and the Internet.
nusirg.beginner.overview.txt	
nusirg.email.ftp.telnet.ps	Basic Internet tools: e-mail, FTP, TELNET and Archie
nusirg.email.ftp.telnet.txt	
nusirg.listserv.usenet.ps	Using USENET and LISTSERV.
nusirg.listserv.usenet.txt	
nusirg.online.info.resources.ps	On-line books, libraries, databases, WAIS, and directory services
nusirg.online.info.resources.txt	
nusirg.supercomputers.ps	Supercomputer use and access information.
nusirg.supercomputers.txt	
nusirg.teach.k-12.ps	The Internet and K-12 education.
nusirg.teach.k-12.txt	
nusirg.whole-guide.ps.tar.Z	All of NUSIRG in compressed and archived format.
nusirg.whole-guide.txt.tar.Z	

Notice that the file names above occur in pairs differing only in the last parts of their names. Files ending with "ps" are in PostScript format and cannot be printed unless you have access to a PostScript printer. You should not copy the files whose names end with "tar.Z" unless you know how to handle compressed and archived files, which is explained in the file "README.nusirg".

Printed Copies From NorthWestNet

If you would prefer to purchase an already printed and bound copy (or if you are unable to use FTP), NUSIRG is available for $20.00 per copy from NorthWestNet.

postal mail:	NorthWestNet NUSIRG Orders 15400 SE 30th Place, Suite 202 Bellevue, WA 98007
e-mail:	nusirg-orders@nwnet.net
phone:	(206) 562-3000
fax:	(206) 562-4822

Please make your check or money order payable to NorthWestNet.

If you have questions, comments, or suggestions about NUSIRG, please send e-mail to nusirg@nwnet.net

Trademarks and Copyrights

Apollo is a trademark of Apollo Computer, Inc.

Apple is a trademark of Apple Computer, Inc.

BRS/Search is a copyright of BRS Software Products

ConneXions and Interop are trademarks of the Advanced Computing Environments.

CP/M is a trademark of Digital Research, Inc.

Cray X-MP, Cray Y-MP, and Cray 2 are trademarks of Cray Research, Inc.

DEC, DECNET, TOP-20, Ultrix, VAX, and VMS are registered trademarks of the Digital Equipment Corporation.

DYNIX is a trademark of Sequent Computer Systems Inc.

Dow Jones News/Retrieval is a registered trademark of Dow Jones and Company, Inc.

Fido and FidoNet are trademarks of Tom Jennings.

Geographic Nameserver is a copyright of Merit, Inc.

IBM, IBM PC, PC/XT. PC/AT and OS/2 are registered trademarks of International Business Machines Corporation.

IEEE is a trademark of the Institute of Electrical and Electronics Engineers, Inc.

Knowbot is a trademark of the Corporation for National Research Initiatives

Macintosh is a trademark licensed to Apple Computer, Inc.

Microsoft and MS-DOS are trademarks of Microsoft Corporation.

PostScript is a trademark of Adobe Systems Inc.

QUIPU is a trademark of NYSERNet, Inc.

SAS is a trademark of SAS Institute, Inc.

Telenet and Tele-mail are trademarks of Telenet Communications Corporation.

Unix is a registered trademark of AT&T Bell Laboratories.

Word Perfect is a trademark of the Word Perfect Corporation.

Xerox is a registered trademark of Xerox Corporation

SECTION 6: EMPOWERING SCIENTIFIC RESEARCH

Chapter 16: Using Supercomputers on the Internet 217
Using Supercomputers 217
Supercomputers: Growth and Future 217
Supercomputers--Always a Need 218
Supercomputer Sites 218
The NSF Backbone 218
Supercomputer Hardware 219
Supercomputer Software 221
Supercomputer Performance Gains 221
Performance Gains Through Specialized Hardware 222
Interaction with Optimizing Compilers 225
Migrating Program to Supercomputers 226
Administrative Access 227
Supercomputer Glossary 227
Supercomputer References 228
Chapter 17: Supercomputer Site Descriptions 230
CORNELL NATIONAL SUPERCOMPUTER FACILITY (CNSF) 231
NATIONAL CENTER FOR ATMOSPHERIC RESEARCH (NCAR) 236
NATIONAL CENTER FOR SUPERCOMPUTER APPLICATIONS (NCSA) 240
NATIONAL ENERGY RESEARCH SUPERCOMPUTER CENTER (NERSC) 247
PITTSBURGH SUPERCOMPUTING CENTER (PSC) 251
SAN DIEGO SUPERCOMPUTER CENTER (SDSC) 259

Chapter 16: Using Supercomputers on the Internet

USING SUPERCOMPUTERS

One of the primary purposes of many parts of the American portion of the Internet (particularly NSFNET) is to provide researchers easy access to supercomputing resources at a number of centralized supercomputer sites. This chapter provides an introduction to the basic principles of supercomputer use and includes detailed site descriptions for a number of the NSF-sponsored supercomputer facilities accessible through the Internet.

SUPERCOMPUTERS: GROWTH AND FUTURE

One of the most amazing phenomena of recent times has been the growth of the computer industry from a laboratory curiosity at the end of World War II to a major player in the economies of the world today, followed by rapid penetration into nearly all areas of life. The growth has been accomplished through extraordinary reductions in size and cost, and improvements in speed, reliability, and usability.

Not long ago a new class of computers has emerged: supercomputers. By one definition supercomputers are "At any given time, that class of general-purpose computers that are both faster than their commercial competitors and have sufficient central memory to store the problem sets for which they are designed" [1]. By this definition, it might appear that supercomputers have been with us from the start, and differ only quantitatively from other computers. In fact, supercomputers are qualitatively different in that they employ different architectures which result in major differences in programming and use.

According to the report "A National Computing Initiative" [2] the American research community will require a thousand-fold increase in computational power over the next few years. This growth is propelled by factors which are likely to exist for a long time to come.

One reason is the dramatic change in international economic competition. Since World War II many countries, especially in Eastern Asia, have entered into the economic mainstream armed with a powerful combination of Western technology and indigenous cultural backgrounds. These countries are redefining many economic and industrial roles and are putting severe pressure on the American economy. Technological innovation, especially in leading-edge fields such as computers, is viewed as a major way to help our competitive stature.

Another reason is that computers are assuming new roles. Computation is now recognized as the third mode of science, co- equal with theory and laboratory research [2]. Simulation of complex real world phenomena is increasingly possible and indeed necessary for future advance: investigations of very large (galaxies), very small (electron orbits), very fast, very slow, very complex, dangerous, or highly energetic phenomena are often possible only through simulation. Computers can also provide "impossible views" of objects under conditions which can never be obtained physically.

Finally, human activities are causing changes in the natural world: developments such as pollution, the destruction of rain forests, ozone depletion, and the greenhouse effect, require extensive monitoring and modelling which would be impossible without the computational and data management capabilities of supercomputers.

SUPERCOMPUTERS--ALWAYS A NEED

The enormous progress in computer hardware will result in a "desk-top-CRAY" in the near future. With such powerful personal computers will we still need supercomputers? The answer appears to be yes. However powerful PC's become, there will always be problems which strain the leading edge of whatever computational powers we develop. The objects of study of disciplines such as mathematics (especially combinatorial problems), and the physical, biological and social sciences are infinitely complex, and computer simulations can never exhaust analytical possibilities. Many simulations today scrape by with the coarsest possible resolution in the hope that important details are not neglected. Increased resolution can come only at great computational cost: for example, a ten-fold increase in spatial resolution in each of 3 dimensions translates into a thousand-fold increase in computational operations. Naturally, many-dimensional problems would require much greater increases. The combined requirements of computational power, data storage, and data flow should insure the continued need for supercomputer sites.

SUPERCOMPUTER SITES

By their very nature, supercomputers are expensive and require extensive hardware, software, documentation, and personnel support in order to function. Vast computational performance must be matched by massive data storage capacities, large data flows, high resolution output devices, an extensive communications interface, as well as a wealth of software applications, libraries, documentation, and expert consulting.

A number of supercomputer sites are currently promoted as national research centers. They are focal points for the collection and distribution of software, documentation, and extensive sets of examples; for training programs, seminars, and symposiums for high-performance graphics laboratories for visualization; and communication facilities in the form of e-mail, lists, and bulletin boards for specialized user groups.

The wealth of different services available at supercomputer sites implies that you should shop carefully in order to locate a site which best matches your needs. The types of software (editors, languages, packages, and operating systems), hardware (memory size and structure, vector and parallel hardware), and services (graphic output, type of consulting) affect the appropriateness of a site to your intended uses. For example, some systems couple high performance vectorization with somewhat limited memories, others feature very large memories with limited vectorization capabilities, whereas others specialize in massively parallel systems.

THE NSF BACKBONE

Supercomputers were born in the U.S. but were generally unavailable to university scientists because of their high costs. Some American scientists actually went to Europe to obtain time on U.S.-built supercomputers. Recognizing the importance of computing access, the NSF established five supercomputer centers across the nation in 1984. Time on these machines is obtained mainly through several grant mechanisms and is otherwise free of charge.

The NSF also helped establish a communications network, called the NSF backbone, which connects these centers and links them to regional networks.

Originally, the centers were deliberately designed to differ from one another in order to make make more options available to researchers, and to test out a variety of novel machine architectures, operating systems, and support strategies under real-life conditions. Recently, however, the systems have converged around several models of CRAY computers, IBM 3090s, and the new massively parallel Connection Machines, mostly running slight variants of Unix.

SUPERCOMPUTER HARDWARE

Supercomputers are not just big, fast machines that use state-of the-art hardware; they also feature special architectures. While these architectures can greatly boost performance, their full power can only be exploited under certain conditions and often require that the program and/or problem itself be restructured by someone who understands how these architectures function.

The most common supercomputer architectures include special memory organization, such as cache memories in the IBM 3090, or independent memory banks in the CRAYs, vector hardware (also called pipeline hardware) present in nearly all supercomputers, and parallel hardware, usually in the form of multiple processors. All of these features have recently started to appear on non-supercomputers, but their implementation and support are especially well developed on supercomputers.

Cache Memories

For a given price, memory can either be big and slow or small and fast. A cache is a small and very fast memory which serves as a buffer to a large slow memory; therefore, cache memories can, under certain conditions, improve overall performance by reducing the time to access main memory. When an operand is fetched from main memory, both it and a number of its nearest neighbors are placed in the high speed cache. If the next reference is to a neighbor, it will be taken from the cache instead of the main memory. If memory usage in the program tends to be highly clustered, then the program operates mainly out of the cache and performance increases greatly. On the other hand, if memory usage is highly scattered, then the cache memory can actually degrade performance.

Independent Memory Banks

Independent memory banks are another way of speeding memory access and storage. A reference to a given bank of memory cells ties up the bank for a number of machine cycles, and no further access is possible until the bank is free. Some computers arrange memory into a number (64 on the CRAY) of independent banks so that most memory references address different banks and do not interfere with one another.

Typically the banks are arranged so that adjacent words, such as X(1), X(2), X(3), reside in different banks. For example, the loop

```
DO 10 I=1,N
```

runs faster with an independent memory bank architecture since the storage times for X(1), X(2), X(3), X(4) etc., are overlapped. However, array structures or patterns of array access which are discordant with the memory architecture can defeat the architecture and slow execution considerably. For example, on a 64 bank memory the following code:

```
DIMENSION X(64,50)
```

```
      DO 10 J = 1,50
10    X(1,J) = 0.0
```

would nail the same bank each and every time through the "J" loop, since X(1,1), X(1,2), X(1,3) etc. reside in the same bank. In this case the problem can easily be solved by redimensioning X as X(65,50) in order to change the memory layout.

Vector Hardware

Vector hardware is currently the most widespread and mature of the special architectures found in supercomputers. Vector units consist of CPU components such as add or multiply units which are divided into a number of sequential stages or segments. The units typically accept two vectors of numbers which proceed through the segments and emerge out the other end as a vector of results. As long as the vectors continue to flow, the process is efficient, and speedups by a factor of 5 or 10 are not uncommon, but starting and stopping the units for short vectors can actually increase execution time. Efficient vector operation often requires some restructuring of code by the programmer as well as the use of compiler directives to monitor and control the process. A more complete description of vector operations appears later in this document.

Parallel Hardware

Parallel architecture distributes the processing of a single program over a number of different physical processing units. It offers the greatest potential for vast increases in performance of any known architectural layout. For example, a machine with 10,000 processors might solve some problems almost 10,000 times more quickly than a single processor machine. However, most programs cannot be partitioned so completely into independent units, and many difficult hardware and software issues remain unsolved and are the subject of intense research. At present, limited parallelism can be exploited on the IBM 3090 and CRAYs, and massive parallelism is possible with the Connection Machine. The effort needed to restructure code ranges from about the same as for vectorization to considerably more. A more complete description of parallel operations appears later in this document.

Impact on Debugging and Graphics

Supercomputers are so powerful and generate such vast amounts of information that they force major changes in both debugging techniques and in the presentation of output. Ad hoc debugging techniques are woefully inadequate for dealing with, say, an error in the 471,000-th iteration of a 500 variable program operating on 18 million separate values. Special debugging software is required which lets you examine values, restart code at various points, observe the effects of changes in values, run controlled experiments, etc.

Similarly, traditional printed and even standard graphical output is often inadequate to represent the millions or even billions of separate data values that can be provided by a supercomputer run. The term "visualization" is the latest buzzword used to describe the sets of advanced graphical techniques which address this problem. Visualization often involves high resolution displays, extensive use of color, 3D representations, and extensive interactive control of the image in order to control the viewing angle, collapse dimensions in a multi-dimensional data set, or perform image processing functions in order to extract or highlight desired information. Existing graphics packages are often insufficient for the task as they emphasize the display of 2- and 3-dimensional XY, XYZ, or contour plots, whereas the representations of complex simulated phenomena require some combination of image processing

approaches along with more standard graphic techniques. In addition, graphics place severe demands on data transmission as well as real time interaction.

Other Hardware Issues

Other hardware issues include memory size and use of high speed peripherals, such as the Solid State Disk (SSD), available on some CRAYs.

The hardware available at different supercomputer sites should be a major consideration in your selection. For example, a CRAY Y-MP is extremely good for highly optimized vector operations, but is somewhat limited in terms of memory, an IBM 3090 has an immense memory but only modest vector performance, and a Connection Machine is much superior for highly parallel problems.

SUPERCOMPUTER SOFTWARE

Supercomputer sites are often endowed with a rich collection of applications software in the areas of Chemistry, Physics, Engineering, Mathematics, and Graphics. Computer languages, however, are confined mainly to FORTRAN, Pascal, and C, and FORTRAN is the only language that is extensively optimized for vector/parallel operations at all sites.

Operating Systems

While the early sites featured a variety of operating systems, most have converted to variants of Unix (UNICOS on the Crays, AIX of the IBM 3090, and Ultrix on some front ends), and CMS which shares the 3090 with AIX. A few sites also offer VMS font ends.

Workstation Support

Some sites provide extensive support for Workstation and PC software to be used in conjunction with the supercomputer or its front end. NCSA at Illinois, for example, provides free downloadable graphics and communications software for PCs, SUNs, and Macintoshes.

Software is probably the single most important consideration in site selection. Unfamiliar languages or editors can greatly increase learning time; programs, data files, and data formats may have to be restructured to move to a particular environment, and future portability must be kept in mind; non-FORTRAN programs may have to be rewritten or run in a non- or sub- optimized mode; and major applications programs, libraries, or graphics libraries may be unavailable or at different revision levels.

SUPERCOMPUTER PERFORMANCE GAINS

Despite the many "fringe" benefits of supercomputer sites, such as access to software, training, graphics, and collaborative research, the main purpose in using supercomputers is to exploit their size and speed in order to handle otherwise intractable problems. The performance of supercomputers is due mainly to two factors:

- Supercomputers are generally big, fast machines which employ high performance technology. As a result, all computer programs automatically benefit.

- Hardware architectures, such as cache memory, independent banks, instruction stacks, high speed I/O, vector, and parallel operations which, although not unique to supercomputers, are usually more highly developed. The exploitation of these features is partly automatic but can also benefit significantly from modifications by the programmer.

Vector and parallel architectures are most often emphasized in optimization, but attention to the other architectures mentioned above can bring about major performance gains as well. One should also note that (a) extensive optimization should only be done in conjunction with special timing tools which locate those portions of the program which can benefit most, and (b) optimization should only be carried to a certain degree--smarter compilers in the near future will be able to optimize well structured code much better than the "spaghetti" code that often results from overzealous hand optimization.

PERFORMANCE GAINS THROUGH SPECIALIZED HARDWARE

The most significant performance gains are obtainable through vector processing and parallel processing hardware.

Vector Processing

Vector processing can best be explained through analogy. Imagine a factory which assembles 6-bladed airplane propellers. It has 6 people, p1,p2,...,p6, at 6 workstations who attach each blade in turn to the hub. Person p1 receives a tray with the parts and assembles the first blade to the hub. When done, he passes the tray along with his partial assembly to p2. P2 attaches the 2nd blade and when done passes the tray and his partial assembly to p3, and so on.

Of course, when p1 has passed his partial assembly to p2 he does not just quit for the day. Instead he immediately gets a new tray and starts a new assembly. Similarly, p2 no sooner finishes his assembly when he gets the tray and the partial assembly from p1 and begins attaching the second blade.

After 6 time periods, the above pipeline is filled with 6 different stages of assembly. The important point to note is that while it still takes 6 time periods to assemble any given propeller, a new finished propeller emerges from the assembly pipeline each and every time period. We have speeded up the assembly process by a factor of 6 by using 6 people in a quasi-parallel manner. Note that even if a product required 10,000 stages to assemble, the products would still emerge at the rate of 1 per time period. In other words, production speed is independent of the complexity or length of the assembly process. On the other hand, if the pipeline is interrupted for any reason, a very long pipeline (or assembly process) will take much longer to restart than a short one.

In supercomputers, the vector or pipeline process is usually applied to arithmetic units, such as add, multiply, or divide units which "assemble" new numbers from the vectors of input numbers.

For example, a multiply unit on the CRAY contains 7 stages. The FORTRAN code

```
      DO 10 I = 1,60
10    C(I) = A(I)*B(I)
```

is converted to vector instructions which start loading the vectors A(1), A(2),... and B(1), B(2),... into special vector registers. As soon as the first pair is available, A(1) and B(1) enter the multiply unit to begin the first stage of multiplication.

At the end of one clock cycle, the partial multiplication of A(1)*B(1) moves to the 2nd stage in the multiply unit, and A(2), B(2) enter stage 1; at the end of the 2nd cycle, A(1)*B(1) moves to stage 3, A(2)*B(2) moves to stage 2, and A(3), B(3) enter stage 1. This process continues and at cycle 7 all seven stages are being utilized, the fully multiplied value of A(1)*B(1) emerges and is stored in C(1), and A(8), B(8) get ready to enter stage 1. From then on, a new multiply is finished every clock cycle, even though it takes 7 cycles to complete one multiply. We have increased the speed almost by a factor of 7 (except for initiating the pipeline).

Vectorization only applies to explicit DO-loops in FORTRAN and their equivalents in other languages. Only one loop within a nest of loops can be vectorized, but the software automatically selects the loop unless that choice is overridden by directives from the programmer.

There exist many obstacles to vectorization, some of which are easily overcome (for example, changing the order of statements), some which require use of compiler directives to modify the vectorization process, and some which require major restructuring of the code. On high performance vector machines, such as the CRAY, speedups of a factor of 5 or 10 for that portion of the code that can be vectorized are not uncommon.

Parallel Processing

The basic idea of parallel processing is easier to grasp than vector processing, but the usage, implementation, and many of the concepts are more difficult. On the other hand, the rewards may be much greater since the potential speedup is proportional to the number of processors available, whereas vector speedup can never be much more than a factor of 10.

Parallel systems distribute the work of a program or subroutine among a number of processors. In some implementations, such as CRAY Macrotasking, the number and operation of parallel processors must be predetermined; in other implementations, such as CRAY Microtasking, the number, identity, or order of execution of the processors is unknown and variable (this has many consequences which influence the structure of a parallel program and hence the modifications needed to convert an existing program to parallel operation). Most of the following discussion applies to CRAY Microtasking.

Some of the concepts of parallel processing are illustrated by the execution profile of subroutine ABC:

```
SUBROUTINE ABC

    (Block B1)      a block of code executed by one processor
    (Block B2)      another block executed by a processor
    (Block B3)      another block executed by a processor

RETURN
END
```

The subroutine is entered at the top under the control of a single processor, after which control is transferred to one, two, or three other processors for blocks B1, B2, and B3. When all of the blocks are finished, control is returned to a single processor which terminates the subroutine. The execution profile might look like this:

Time	Processor 1	Processor 6	Processor 4
1	Start ABC	x	x
2	Run B2	Run B1	Run B3
3	x	x	Exit ABC

where "x" means the processor is idle or engaged with another program. The profile is generally unpredictable and the actual number and identity of processors depends on many factors. Because of this unpredictability, the parallel operation is often referred to as a "fray."

A DO-loop is a special case of the above structure:

```
      SUBROUTINE ABC

      DO 10 I = 1,3 (the i-th iteration of the loop is block B(i)

10    CONTINUE

      RETURN
      END
```

in which some iterations of the loop may be handled by different processors.

Some obvious and not-so-obvious consequences derive from such an implementation. The different code blocks must be totally independent of the sequence in which they are processed: Block 3, for example, cannot depend on the results of, say, Block 1. It is less obvious that local variables assigned values inside a block can have no validity outside the block, that values associated with loop iterations must be explicitly tied to the loop index, and that global variables which are not indexed by a loop index must be "guarded" from simultaneous access. Consider the following section of code:

```
      K = 0
      DO 10 I = 1, 1000
      K = K + 1

10    CONTINUE
```

The operation of the above loop depends on whether K is local or global. Local variables in a parallel block of code are replicated for each processor that joins in the "fray" whereas global variables occupy a single location which is accessed by all processors.

Suppose that K is a local variable and the 1000 iterations of the loop are randomly distributed among, say, 64 processors. Then processor 17 might handle 23 iterations of the loop, processor 39 might handle 11 iterations, and so on. At the end of the loop, if control is assigned solely to processor 17 then K=23, if assigned to processor 39 then K=11, whereas in a non-parallel environment K always equals 1000.

Inside the loop, if K is used as a substitute for the loop index, then its value depends only on how many times that processor has handled the loop, and so will generally have little relation to the value of the loop index itself. Thus A(K) rarely is the same as A(I), whereas in a non-parallel environment it is always so.

If K is a global variable then it must be guarded from simultaneous access by more than 1 processor. For example, suppose that K=0 and processors 13 and 19 begin execution of RK=K+1S at slightly different

times. If 19 accesses the expression before 13 has finished, then K will be reset to 1 instead of being set to 2 as it should be.

This type of code is called a "critical section" and must be guarded by explicit directives which guarantee that another processor can only enter that section of code when the current processor has finished.

Parallel and vector processing are quite compatible and can be used not only in the same program but often in the same nest of loops. For example, some compilers analyze a DO-loop nest and select one loop for vectorization and another for parallel processing. On the CRAY, a single loop can be partitioned into vectorized and parallelized sections.

INTERACTION WITH OPTIMIZING COMPILERS

The goal of an optimizing compiler is to speed the execution of a program without changing the results. It scans the user's source code and identifies structures which can utilize the special hardware found in supercomputers. Since the conditions under which these structures can be safely optimized are very limited, it must analyze much of the surrounding code to determine if, and to what degree, the code can be optimized.

The compiler is subject to several important constraints which limit its effectiveness. Since the optimization is done at compile time it can have no knowledge of the run-time structure of the job. Thus, input data values which may change the flow of execution and the degree of optimization cannot be considered, so the compiler has to assume the worst case and do the least amount of optimization. Another constraint is that optimization is often limited to the scope of a single routine, and interactions between routines which might affect optimization cannot be taken into account. In addition, the technology of optimizing compilers is continually advancing, and any given compiler may be unable to optimize code which later versions can handle.

Because of these limitations it is often necessary for the programmer to supply auxiliary information which eliminates ambiguity and resolves problems which the compiler cannot handle. This is especially important in parallel processing, but is also often required in vectorization. The programmer supplies global directives on the compile statement or embeds local directives in the source code, or both, which guide the optimization process. For example, the following loop would be rejected by any vectorizing FORTRAN because of a "recurrence" relation in the loop:

```
      K = IVAL
      DO 10 I = 1,N
10    A(I) = A(I-K) + 1.0
```

While "recurrence" and other inhibitors of vectorization are beyond the scope of this report (see [4] and [5] for more information), the above code can be vectorized on a CRAY provided that (a) K is negative, or (b) K is greater than 64. If the programmer knows this he can insert compiler directives which force the compiler to vectorize. Thus the code

```
CDIR$ IVDEP                    ("ignore vector dependency")
      DO 10 I = 1,N
10    A(I) = A(I-K) + 1.0
```

in effect, tells the compiler that "I know more than you, so forget your inhibitions and vectorize anyway."

Parallel optimization usually requires more embedded directives than vectorization. It is frequently necessary to explicitly prevent multiple processors from accessing code until all processors have finished operating on a previous section. For example, the CRAY compiler directives (see [6])

```
CMIC$ DO GLOBAL
      DO 10 J = 1,N
                                    (operation on the J-th column of A)
10    CONTINUE
CMIC$ DO GLOBAL
DO 20 I = 1,M
                                    (operation on the I-th row of A)
20    CONTINUE
```

would distribute the operations of the N columns of A and the M rows of A to different processors, but would assure that all columns were finished before any rows were processed. The directives are signaled by "CMIC$" (for Microtasking) in columns 1-5 and apply to the next DO-loop. Thus the start of the second "CMIC$" terminates the first "CMIC$" and tells the compiler that the two loops are to be processed in sequence.

MIGRATING PROGRAM TO SUPERCOMPUTERS

The main point in using supercomputers is to achieve dramatic increases in processing speeds. The increases are partially due to the fact that supercomputers are simply big and fast computers, but much of the increase comes from the special architectures that supercomputers employ. Achieving a high degree of optimization on supercomputers requires a well thought-out strategy as well as realistic expectations as to what can actually be achieved.

The maximum gains that can be obtained through parallel processing or vectorization are controlled by "Amdahl's" law [1]: P = 1/(1-X+X/a) where P is the performance gain, X is that ratio of the code which can be optimized (vectorized or processed in parallel), and "a" is the speedup ratio for vector or parallel code. The important part of the law is that the performance is much more controlled by X, the proportion of the code that can be optimized, than by "a." For example, suppose that X = 0.80, so that 80% of the code can be vectorized or processed in parallel. Then the absolute maximum speedup possible is a factor of 5, even if vector or parallel operations are infinitely fast (that is, even if "a" is infinity, then P = 1/(1-0.8 + 0.8/infinity) = 1/(0.2) = 5).

Note that it is rare for X to exceed 0.8 (factor of 5 speedup) for vector processing applications, whereas X may often exceed 0.99 (factor of 100 speedup) for parallel processing applications.

The first step in migrating code to a supercomputer (after making any adjustments to insure that the FORTRAN code will actually compile and run), is to instrument the program in order to identify "hotspots," that is, code which is extremely CPU intensive. A good candidate program for optimization should have an execution profile such that nearly all of the CPU time is spent in just a few percent of the code (if the execution profile is fairly uniform, then hand optimization may be a waste of time). Most supercomputer sites provide easily used compiler directives or other tools which will show you a histogram of CPU time spent versus source code.

If the program has identifiable hotspots, the next step is to read and understand the compiler messages which identify obstacles to optimization. For example, the IBM 3090 compiler provides an in depth analysis of each loop, which identifies not only the obstacles to vectorization, but also the results of an "economic" analysis to determine which loop in a nest of loops yields the best results.

At this point you should analyze the loops highlighted by the compiler to see if any can be replaced by routines from existing optimized libraries, such as the ESSL routines on the IBM 3090. Many such routines are coded in assembly language and are optimized to a very high degree. After this, you should attempt to remove inhibitors to optimization by making local modifications to loops. In many cases this can be done simply by reordering statements, splitting loops, or adding compiler directives. For example, the loops on the left will not vectorize, while the loops on the right (which produce exactly the same results) will:

```
      DO 10 I=2,N                      DO 10 I=2,N
      B(I) = A(I-1)                    A(I)=C(I)
10    A(I) = C(I)                10    B(I) = A(I-1)

      DO 10 I=1,N                      DO 10 J=1,N
      DO 10 J=1,N                      DO 10 I=1,N
10    A(I,J)=A(I,J-1)*B(I,J)     10    A(I,J)=A(I,J-1)*B(I,J)
```

In the above examples, a "recurrence" relation existed which prevented vectorization. In the following example, the recurrence relation could not be avoided for one expression, but by splitting the loop into two parts, one of the two expressions on the right can be vectorized:

```
      DO 10 I=1,60                     DO 10 I=1,60
      A(I)=B(I)*C(I)             10    A(I)=B(I)*C(I)
10    D(I)=D(I-1)*E(I)           20    D(I)=D(I-1)*E(I)
```

When these steps have been taken, further optimization can be achieved by paying careful attention to memory layout or by restructuring the program to obtain the best match between program structure and the architecture of the machine. The latter is, of course, a very time consuming operation and should only be undertaken when the benefits and the costs are fully considered.

ADMINISTRATIVE ACCESS

Resources on all of the NSF sites and some of the other sites are mainly allocated on the basis of grants, rather than by money. Three main grant types are commonly available:

- Large allocations which are subject to formal peer review boards which meet at scheduled times during the year. These boards are comprised of personnel which may be from the site, jointly from several sites, or from NSF.
- Small allocations for startup or familiarization purposes. These are usually available at any time throughout a year.
- Block grants awarded to Universities. The Universities allocate sub blocks of time through their own review boards.

SUPERCOMPUTER GLOSSARY

The following glossary defines some of the terms used in this report. A much more complete glossary of supercomputer terms appears in reference [1].

CGM:	an acronym for Computer Graphics Metafile. It is a GKS standard metafile which contains generalized plot instructions which can then be converted to drive a specific graphics device.
ETHERNET:	a network scheme which uses the Carrier Sense, Multiple Access, and Collision Detect (CSMACD) method of attaching hosts to it. The Ethernet can be used with a number of protocols including TCP/IP and DECNET.
FILE SERVER:	a computer system that permits network users to create or access files.
FTP:	an acronym for File Transfer Protocol. This is a commonly used program that reliably transfers files between two computers on a network.
GBYTES:	a billion bytes of information.
GKS:	an acronym for Graphics Kernel System. An internationally recognized standard for creating graphic images on a computer.
HPGL:	a plot format specifically designed for certain models of Hewlett-Packard plotters.
MAC:	short for Macintosh computer.
MBYTES:	a million bytes of information.
MASS:	a mass storage facility commonly used at supercomputer sites to store large amounts of semipermanent data for each user.
METAFILE:	a generic graphics file which is not designed to drive any existing graphics device but which can be converted to any specific graphics device.
MEMORY BANK:	a region of memory in a computer which has a single access path to it. Any access to a bank blocks all other accesses until it is complete.
MFLOPS:	millions of floating point operations per second.
MWORDS:	millions of computer words.
NS:	a nano-second, or one billion-th of a second.
POSTSCRIPT:	a text and graphics language designed especially for "Desk Top Publishing" environments.
SSD:	an acronym for Solid State Disk, a very large high speed semiconductor memory that serves in place of a rotating disk, but is much faster.
TELNET:	a program that permits a user at one computer to login to another computer.
T1:	a telecommunications medium that allows data to be transmitted at a speed of 1.544 megabytes per second. The NSF backbone and much of NorthWestNet operate at T1 speeds.
VISUALIZATION:	a collection of techniques which make it possible to display the results of calculations. It includes, but it not confined to, standard graphical displays such as line or contour plots. It is especially useful to compress the vast amount of information generated by supercomputers into a comprehensible form.

SUPERCOMPUTER REFERENCES

[1] "Supercomputing: An Informal Glossary of Terms," IEEE Washington Office, 1111 Nineteenth Street, NW, Washington D.C. 20036. 1987.

[2] H.J. Raveche, Duncan H. Lawrie, Alvain Despain, "A National Computing Initiative-An Agenda for Leadership." Society of Industrial and Applied Mathematics, 1400 Architects Building, 117 So. 17th Street, Philadelphia, PA, 19103-5052. 1987.

[3] Norris P. Smith, "Of Supers and Minisupers," Computer Graphics World, August 1988.

[4] Helen M. Doerr, Francesca Verdier, "Improving Vector Performance," and "Introduction to Vectorization," Cornell National Supercomputer Facility, Cornell University, Ithaca, New York, September 1987.

[5] David D. Soll, "Vectorization and Vector Migration Techniques," IBM Technical Bulletin, SR20-4966-0.

Chapter 17: Supercomputer Site Descriptions

The following section provides detailed information about six NSF-sponsored sites which provide supercomputer services for NSF sponsored researchers. The purpose of these site descriptions is to allow you to do some preliminary comparisons of each site's hardware, software, and training programs so you can select the supercomputer site which would be most appropriate for your particular research needs.

All of the site descriptions were reviewed or contributed by representatives of each site's user services staff in September, 1991. We'd like to thank the following individuals for their participation:

Supercomputer Site	Reviewed By
Cornell National Supercomputer Facility	Blaise Barney
National Energy Research Supercomputer Center	Jean Schuler
Pittsburgh Supercomputer Center	Deb Nigra
National Center for Atmospheric Research	Juliana Rew
National Center for Supercomputer Applications	Lyle Rigdon
National Center for Supercomputer Applications	Melissa Johnson
San Diego Supercomputer Center	Mark Sheddon

CORNELL NATIONAL SUPERCOMPUTER FACILITY (CNSF)

Computer Hardware

- ❏ Two IBM ES/3090 600J, with 6 vector facilities each
- ❏ 512 Mbytes main memory each
- ❏ 827.6 Mflops maximum speed
- ❏ 266 Gbytes of disk storage
- ❏ 896 MBytes of memory available to each user application
- ❏ Both vector and parallel processing hardware
- ❏ 2 Sun workstations
- ❏ Stellar Graphics Supercomputer
- ❏ Graphics hubs across Cornell campus

Software

- ❏ AIX operating system for both 3090s (CMS still available as well)
- ❏ Software support for vectorization, including vectorizing compiler and vector libraries
- ❏ Interactive and batch modes
- ❏ Editor: IBM Xedit
- ❏ Text Processors: SCRIPT/VS (includes MFF), TeX
- ❏ FORTRAN: VS FORTRAN, PTOOL FORTRAN aids
- ❏ Other languages: Assembler, C, EXEC, EXEC2, LISP, PASCAL, PL/I, PROLOG, REXX, APL2 (vectorized)
- ❏ PUBLIC: public domain user contributed and supported software

Chemistry

ALCHEMY II	AMBER	AMPAC	CHARMm
DISCOVER 2.5.1	GAMESS	GAUSSIAN 88	HONDO 7PF & 8
KGNMOL	MELDF	MOPAC	VENUS

Computational Fluid Dynamics

FIDAP

Electronics

HSPICE 8907

Engineering

ABAQUS	ESSL	DYNA3D	HSPICE
SUPREM III	VSPICE		

FORTRAN Utilities & Debuggers

FORTRAN IAD	ROAFP	FORTAUX	UTILITY

Graphics

CONPLOT	DI-3000	DISSPLA 11.0	GDDM
GDQF	graPHIGS 2.1	GRAPHIT	NCAR
SAS/GRAPH	TELL-A-GRAF	VIEW	

Mathematics and Numerical Libraries

ACMALG	ACRITH 1.3	BSPLINE	CERNLIB
EISPACK 3	ELLPACK	ESSL 3 & 4	FFTPACK
FISHPACK	FUNPACK 2	GAMS/MINOS	IMSL 11
LAPACK	LINPACK	LLSQ	MAPLE 4.0
MATHADVANTAGE	NAG 13	ODEPACK	QUADPACK
RECIPES	REDUCE 3.3	SCRATCHPAD II	SLATEC 3.2
TOEPLITZ			

Mechanics

ABAQUS 4.8.4	ANSYS 4.4

Programming Languages

APL2	ASSEMBLE	C/370 1.1	EXEC
EXEC2	FORTRAN VS 2.3	FORTRAN VS 2.4	HASM
LISP VM	PARALLEL FORTRAN	PASCAL VS 1.2	PL/I OPT. 2.3
REXX			

Simulation

GPSS/H 1.99	SIMSCRIPT II.5	SLAM II 4.03

Statistics

BMDP88	EDA	GLIM 3.77	NTSYS 4.6
SAS 5.18	SAS/IML	TSP	

Utilities

PASSTHRU	PDF	PIPELINES	PS INTERPRETER
PTOOL	SyncSort	TOOLS	VM BATCH

Training

- ❑ Introduction to systems, access, languages, and optimization tools for new users
- ❑ Summer institutes for researchers working on code
- ❑ Undergraduate/Faculty workshops
- ❑ Workshops on parallel processing and vectorization
- ❑ Computational Chemistry workshops
- ❑ Visualization and remote graphics workshops

Collaborative Research

- ❑ User groups and Research Interest Groups (Computational Chemistry and Statistics)
- ❑ Interdisciplinary research groups open to all researchers
- ❑ Corporate Research Institute

Documentation

- ❑ Online documentation: BOOKMGR, CNSFINFO, CUINFO
- ❑ Newsletter: ForeFronts
- ❑ Training notebooks of presentations from classes and lab exercises
- ❑ Online instruction Program: TUTOR
- ❑ Online help program and man pages

Consulting

- ❑ E-mail consulting
- ❑ Large scale computing consulting
- ❑ Computational Research Associate Staff in Chemistry, Statistics, Operations Research, Astrophysics, Mathematics, Engineering, and Agricultural and Biological Engineering
- ❑ Visualization consulting
- ❑ Strategic User Program for those using parallel computing in production work
- ❑ Corporate Research Consulting

Graphics Support

- ❑ Support for remote X-Windows (including xim with WaveFront) Tektronix 4105, 4010; Versaterm for Mac; GRAF for PC; VT100 emulators; gnuPLOT and a version of xmovie
- ❑ An extensive graphics lab with high speed fiber optic links, an image processing system, facilities for videotape production, a digital camera for 35 mm slides, printers for producing camera ready copy, plotters and color PostScript printers, and a number of IBM and Unix workstations

Strengths and Limitations

Strength

parallel processing and very large memory

Limitation

limited vector processing (as compared with a CRAY)

Administrative

Allocations are approved at Theory Center at CNSF by a peer review process; small startup grants may be made before peer review process is complete.

Contact Summaries

Consulting:	(607) 254-8686 eagle@eagle.tc.cornell.edu
Technical and visualization consulting:	(607) 254-8686 (607) 255-3985 psfy@cornellf.tc.cornell.edu
Allocations:	Pat Colasurdo (607) 255-3985 pat@cornellf.tc.cornell.edu
Training:	(607) 254-8686
System Status:	(607) 255-7138
Network help:	Network Management Center (24 hrs/day, 7 days/week) (607) 255-9900
IBM 3090 addresses:	cornelle.tn.cornell.edu (128.84.252.7) cornellf.tn.cornell.edu (128.84.252.21) eagle.tc.cornell.edu (AIX) (128.84.201.1)

Login Script

The following script shows how to connect to the CNSF IBM, login, compile, load and run a program, and then logout. It assumes that you connect from some local machine which supports the TCP/IP TELNET program. Note that if you are login in from a VMS machine, you will need to use TN3270 to access CNSF.

Prompts are represented by "user>>" for you, "CNSF>" for the IBM, "TELN>>" for TELNET, and "CNus>>" for IBM responses followed by user entries on the same line. Most responses are abbreviated. Comments are enclosed in parentheses.

Note that this dialog has been tested.

Prompt	Dialog	Comments
user>>	telnet 128.84.252.21	(TELNET to the IBM)
TELN>>	...	(TELNET messages)
CNSF>>	LOGON user-id	
CNSF>>	ENTER PASSWORD OR LOGOFF (IT WILL NOT APPEAR WHEN TYPED): password	
		system messages
CNus>>	fortvs hello	(Compile the program)
CNSF>>	...	(Compiler messages)
CNus>>	fortlibs	(Access libraries required to load and run program)
CNus>>	load hello	(Load the program)
CNus>>	start *	(Run the program)
CNSF>>	Execution begins...	
CNSF>>	HELLO WORLD	
CNSF>>	logoff	

NATIONAL CENTER FOR ATMOSPHERIC RESEARCH (NCAR)

Computer Hardware

- ❏ CRAY Y-MP8/864 with 8 processors; CRAY Y-MP2D with 2 processors
- ❏ 64 Mwords of main memory for the Y-MP8, 16 Mwords for the Y-MP2
- ❏ 48 Mword maximum memory per job for batch; 16 Mwords for interactive
- ❏ 6.0 ns clock for both CRAYs; a measured peak rate of over 1 Gflops has been achieved with a multitasked ocean model
- ❏ Vector processing hardware for both CRAYs
- ❏ 256 million word SSD for Y-MP8 (1000 Mbyte/sec channel), 128 million word SSD for Y-MP2
- ❏ 60 gigabytes of local storage for the Y-MP8, 20 gigabytes of local storage for the Y-MP2
- ❏ Unix front end
- ❏ NCAR Mass Storage System (MSS) is a hierarchical file storage system with no limit per user. The MSS has a 120 gigabyte disk farm and over 108,000 tape cartridges. It features a StoragTek 4400 Automated Cartridge System capable of holding 6,000 cartridges, or a terabyte of information. Currently, MSS stores over 19 terabytes of data. Extremely high data transfer to the CRAYs; accessible from front-end
- ❏ Connection Machine 2 (8,000 processors and floating point hardware)
- ❏ Sun 4/490 used as a gateway for NCAR's Internet Remote Job Entry System (IRJE). Allows users to create and submit CRAY jobs from their local hosts connected to the Internet. Output is automatically shipped back to the remote host

Software

- ❏ UNICOS 6.1.4 operating system on CRAY Y-MP8/864; UNICOS 5.1.11 onCRAY Y-MP2D
- ❏ FORTRAN: CFT 77 version 4.0 (default) or 5.0
- ❏ Other languages: vectorizing C and Pascal compilers

Application Libraries

ALFPACK	AMOSLIB	ECMFFT	EDA
EISPACK	EISPKD	FISHPAK	FITPACK
FUNPACK	IMSL	CRAYFISH	LINPACK
FFTPACK	MINPACK	MUDPACK	NAG
ODEPACK	SPHERE	SCI	SLATEC
SSDLIN	STARPAC	NCARG	NCARG-GKS
NCARG-LOC	NCARM	NCARO	

Graphics

NCAR/GKS Version 3.1.2

Training

- New user orientation class: One-day class to introduce Unix/UNICOS programming tools and the NCAR computing environment.
- Unix Basics Class: Two-day introduction to the basics of Unix
- Remote site visits are granted to sites with a significant number of users
- User conference held every 18 months to update users on new developments and new directions in computing at NCAR
- Site liaison workshops held biennially: intensive, in-depth presentations on specific aspects of the NCAR computing facility
- Graphics and optimization workshops are planned for the future

Collaborative Research

- User groups
- Real-time computing during field programs
- Classroom grants of computer time

Documentation

- Online documentation available for most of the utilities in the supported libraries via the Distributed Software Libraries (dsl) utility
- Scientific Computing Division (SCD) hardcopy documentation available free except for the GKS Graphics manual
- User Documentation Catalog containing references both to SCD supported and vendor documentation. Available online via anonymous FTP to ftp.ucar.edu with the filename /docs/catalog/userdoc.catalog
- NCAR UNICOS Primer: contains necessary basic information to begin computing at NCAR. Uses step by step examples to create, submit, and receive output from the CRAYs
- Daily Bulletin: Online source of up to date information on the computing systems
- Monthly newsletter: SCD Computing News
- Annual planning report: Supercomputing: The View From NCAR
- NCAR Annual Report
- NCAR Annual Scientific Report

Consulting

- Phone, 8-5 MST weekdays; walk-in consulting available for visitors and local users. Extended consulting available by appointment
- E-mail consulting available from Internet, BITNET, Span, and Omnet
- Specialized consulting for software libraries, networking and data communications, optimization, multi-tasking, CRAY I/O optimization, segmentation (overlaying) of large applications, math algorithms, NCAR Graphics, IBM PC/AT and Macintosh, and Unix and UNICOS
- Visitor assistance includes access 24 hrs/day, 7 days/week, IBM PCs and Macintoshes for terminal or stand-alone use, color graphics terminals, Canon microfilm/fiche reader/printers,

community telephone for business use, documentation library (including vendor documentation)

Graphics Support

- ❑ DICOMED film recorder for 16 mm film and fiche (black and white)
- ❑ DICOMED color film recorder (35 mm color slides)
- ❑ Xerox 4050 laser printers for paper graphical output
- ❑ CGM metafile support
- ❑ Remote graphics support includes: Tektronix 4010, 4105, 4107, 4025, 4115; VT100, VT125, VT220, VT330; ADM; AED512; HP2648a, HP7475a, HP150; PC-Plot for IBM/PC; Versaterm for Macintosh.
- ❑ Output mailed to users at remote sites

Strengths and Limitations

Strengths

- ❑ High performance vector processing
- ❑ Extensive user support services
- ❑ SSD for large I/O applications
- ❑ Extensive data archives for atmospheric and oceanographic research
- ❑ High volume mass storage system
- ❑ Extremely high volume graphics output capabilities
- ❑ 16 mm movie making and video support
- ❑ Remote job entry system

Limitations

- ❑ Limited memory
- ❑ Few applications outside atmospheric sciences

Administrative

- ❑ Allocations are open to university researchers with NSF grants in atmospheric, oceanographic, and related sciences; grant requests are reviewed by a peer review board composed of NCAR staff and university researchers
- ❑ Allocations are also available to government agencies on a cost recovery basis

Contact Summaries

Consulting:	(303) 497-1278 consult1@ncar.ucar.edu
Visitor/User Information:	(303) 497-1225 scdinfo@ncar.ucar.edu
Allocations:	(303) 497-1207 knudson@ncar.ucar.edu

Documentation Orders:	(303) 497-1232 maryb@ncar.ucar.edu
IBM 4381 front end address:	evans.ucar.edu (128.117.8.18)
IRJE (remote job entry):	rje@windom.ucar.edu

NATIONAL CENTER FOR SUPERCOMPUTER APPLICATIONS (NCSA)

Computer Hardware

- CRAY Y-MP4/464 with 4 processors; CRAY-2S/4-128 with 4 processors; Connection Machine Model 2 (CM-2) with 32,000 parallel processors and 32 bit floating point hardware; Convex C240 system (used for memory intensive real-time visualization)
- 64 Mwords of main memory for the Y-MP; 128 Mwords of main memory for the CRAY-2; 256 MByte main memory for CM-2; 1 Gbyte of physical memory for the Convex
- A maximum of 6.5 Mwords of main memory per user on the X-MP (see HELP MEMORY on the CRAYs for details)
- 6.0 ns clock speed for the CRAY Y-MP; 4.1 ns clock speed for the CRAY-2
- Vector, scalar, and parallel processing
- High speed 128 Mword Solid State Disk (SSD) for the CRAY-Y/MP
- 16 Gbytes temporary storage for Crays; 10 GByte DataVault for the CM-2
- Common File System (CFS) with 120 GBytes of permanent file storage running on an Amdahl 5860
- Potentially unlimited magnetic tape storage through an IBM 3480
- Plans for significant increases in mass storage capabilities
- Three IBM RISC System 6000 model 550's (high-end superscalar machines for predominantly scalar applications); there will be a total six of 550's and one RS/6000 model 950 Front-end / File Server by the end of 1991.
- Three Sun 4/490 front ends running SunOS for the CM-2
- Numerous high end machines from Silicon Graphics, Apple, IBM, and SUN in the Numerical Laboratory, accessible over the Internet
- VAX/Ultrix front end for BITNET connectivity
- New network hardware and software being installed in a testbed research project (BLANCA) to prepare for the national gigabyte transcontinental network (NREN) ; by early 1992 BLANCA should include a 622 Mbit/sec trunk to University of Wisconsin, and 45 Mbit/sec links to Berkeley and Bell labs

Software

- UNICOS is the operating system on both CRAYs. It provides interactive and batch access to a Unix operating system (based on AT&T Unix System V)
- Multitasking is supported on the CRAYs. Dedicated time is available to allow access to all 4 CPUs
- Editors: vi on the CRAYs
- FORTRAN: CFT, CFT77 and CF77; cdbx is the debugger of choice on both CRAYs, though other debuggers are available
- Other languages: vectorizing C compiler on both CRAYs
- cfs utility to move files between CFS and the CRAYs and Many example programs for vectorization and graphics on CFS
- Extensive in-house software development applications staff
- Public domain software for workstations available via anonymous FTP-connect to ftp.ncsa.uiuc.edu (or 141.142.20.50) with user-id "anonymous" and your user-id and e-mail address as the password--get the file "README.FIRST" for an informative overview of current holdings and FTP instructions. Each directory, including the root, contains a file

named "INDEX" which contains a recursive listing of the files of that directory and of all subdirectories within it.

FTP'able software currently available includes: DataScope, Image, Layout, PalEdit, TELNET, Contours, GelReader, Hypercard Scientific Animation Package, Import2HDF, Reformat/XReformat, SDIFF, PC Show, ImageTool, DataSlice, X Image, PolyView, Isosurface Visualizer, and CompositeToolarchive.

Software On the CRAYs

Astronomy

MIRIAD

Chemical Engineering

ASPEN PLUS

Chemistry

AMBER	AMPAC	BROOHKAVEN PDb	CADPAC
CHAARMm	DISCO	DISCOVER	GAMESS
GAUSSIAN90	GPRLSA	MACROMODEL	MINP
MM2	MM2(87)	MOLECULE-SWEDEN	
MOPAC	RANFOLD	RNAFOLD	RPAC
XPLOR			

Computational Fluid Dynamics

FIDAP	FLOW3D	FLUENT	KIVA II
PHOENICS	PLOT3d	SALE3D	TURB3D

Electrical Engineering

CAzM	EMAS	HSPICE	MSC/EMAS
PISCES IIB	SIGVIEW	SPICE	SUPREM 4

Graphics

Blaze	GNUPLOT	GPLOT	MOVIE.BYU
MPGS	NCAR (GKS)	PVI Contouring	PVI DI-3000
PVI GK-2000	PVI GRAFMAKER	PVI Metafile	Zetavu

Mathematics and Statistics

ACM-Algorithms	BCS	BCSEXT	CALMATH
ECMFFT	EDA	EISPACK3	ELLPACK
FFT	GLIM	HSML	IMSL
ITPACK	LAPACK	LASO2	MINOS
MINPACK	NAG	NUMERICAL RECIPES	
ODEPACK	PCGPAK	PORT	POSSOL
SCILIB	SLAM II	SLATEC	SMPAK

SPARSE	SPECFUN	SPSS-X	SSDLIN
TOEPLITZ	XMP	ZOOM	

Solid Mechanics and General Engineering

ABAQUS	ANSYS	DADS	DYNA2D
DYNA3D	INGRID	MAZE	MSC/NASTRAN
NIKE2D	NIKE3D	ORION	PATRAN II
TAURUS	TOPAZ2D	TOPAZ3D	

Systems and Utilities

AS	BENCHLIB	CDBX	CF77
CFT	CFT77	HDF	KAP/Cray
PASCAL	PCC	SCC	

Training

- ❑ Education program features supercomputer use for users from kindergarten to advanced researchers; a strong emphasis also on incorporating scientific visualization techniques into education.
- ❑ Academic Affiliates Program with members at more than 100 institutions; features special training and block grants of computer time for members.
- ❑ Visitors program for faculty, postdoctoral, and student researchers to support working visits to NCSA.
- ❑ Renaissance Experimental Laboratory for teaching visualization skills and efforts to incorporate scientific visualization into course curricula.
- ❑ Monthly training workshops on introductory and advanced topics including use of the Crays, the Connection Machine, applications packages, and visualization techniques.
- ❑ Off-site classes available covering topics of interest to particular remote sites.
- ❑ Workshops and seminars on specialized topics in various disciplines.
- ❑ Summer institutes and workshops.
- ❑ Many training video tapes: vectorization, CRAY architecture, Multi-tasking, etc.

Collaborative Research

- ❑ Association with the Beckman Institute, an interdisciplinary program for artists, computer scientists, and researchers in many intellectual disciplines.
- ❑ Unix mail on the CRAYs (users are encouraged to forward mail to their home systems).

Documentation

- ❑ Unix man pages for technical information
- ❑ Documentation system (originally from CTSS) ported to UNICOS for access to complete manuals
- ❑ On-line HELP facilities on CRAYs
- ❑ NCSA Technical Resources Catalog lists documentation available from NCSA or third parties

- ❏ Innovative video magazine (NCSA RealTime)
- ❏ New users provided with an extensive startup kit and kept informed via frequent online and printed materials.
- ❏ Many documents are available for free or for purchase
- ❏ Online help is available for both the CRAY and the VAXes
- ❏ Extensive sets of example programs are available on CFS
- ❏ Bimonthly newsletters: ACCESS and DATALINK
- ❏ Annual report, entitled "SCIENCE: The State of Knowing," which summarizes NCSA's activities in the previous year

Consulting

- ❏ Consulting available via e-mail (24 hours a day) and phone (8am to 5pm central time (see Contact Summaries below)
- ❏ On-line HELP and man pages on the CRAYs, help files on the VAXes

Graphics Support

- ❏ Strong emphasis on local and remote use of scientific visualization
- ❏ Workstation tools for graphics analysis
- ❏ Scientific visualization programs, to which users may apply to have their data produced for use in high quality graphics presentations
- ❏ CGM metafile support
- ❏ Support for remote graphics terminals includes Tektronix 4014, 4105, 4113, 4115/4125, PostScript, HPGL, Macintosh
- ❏ Output is mailed to remote users
- ❏ Media Services facility features video and audio resources, including Silicon Graphics Inc. frame buffers, Abekas digital disk recorders, D1 format digital VCRs, Macintosh-based digital audio recording, Betacam SP format analog VCRs, computer-controlled editing and routing, and over 150 other components. Activities supported include transfer of images from computers to video, audio, hard copy, and film, as well as between the different formats within each of these media; media processing such as video editing, text generation, electronic paintbox, digital effects, and compositing; audio recording and editing to further supplement basic imagery; and video field production for documentary programming and special projects.

Strengths

Strengths:

- ❏ High performance vector processing
- ❏ Solid State Disk for fast internal I/O
- ❏ Scientific visualization
- ❏ State-of-the-art graphics laboratories

Administrative

- ❏ Requests for 100 or more service units of CPU time are reviewed by a joint NCSA/Pittsburgh Peer Review Board quarterly.
- ❏ Requests for less than 100 service units are made eight times a year.
- ❏ Applications for less than six service units are reviewed internally by NCSA.
- ❏ Researchers at affiliate institutions may apply for start-up grants for 1-5 hours on the CRAY Y/MP4/464 through their campus affiliate representatives.
- ❏ The Supercomputing Education Project allocates blocks of five service units on the CRAY Y/MP4/464 for faculty who use high-performance computing in their classes.

(note: service units are CPU or wall clock hours weighted by different factors for the various computers)

Contact Summaries

Academic Affiliates:
Joe Blackmon
(217) 244-1100
affiliat@ncsa.uiuc.edu

Academic & Industrial Relations Program:
Scott Lathrop
(217) 244-1099
slathrop@ncsa.uiuc.edu
u13006@ncsagate (BITNET)

Accounts / Client Administration:
Judy Olson
(217) 244-0074
uadmin@ncsa.uiuc.edu
uadmin@ncsagate (BITNET)

Consulting:
(217) 244-1144
consult@ncsa.uiuc.edu
consult@ncsagate (BITNET)

Education Program:
Nora Sabelli
(217) 244-0644
nsabelli@ncsa.uiuc.edu
nsabelli@ncsagate (BITNET)

Faculty Program:
Melanie Loots
(217) 244-2921
mloots@ncsa,uiuc.edu
u26400@ncsagate (BITNET)

Media Services:
Vincent Jurgens
(217) 244-1543
vjurgens@ncsa.uiuc.edu
u14998@ncsagate (BITNET)

Orders for software, publications, videos:	Debbie Shirley (217) 244-4130 docorder@ncsa.uiuc.edu docorder@ncsagate (BITNET)
Training:	Lyle Rigdon (information) (217) 244-3659 lrigdon@ncsa.uiuc.edu Karen Gobble (registration) (217) 244-4198 kgobble@ncsa.uiuc.edu

Hardware Status: (217) 244-0710

Network status: (617) 873-3400

Cray Operations: (217) 244-0710

CRAY Y-MP address:	uy.ncsa.uiuc.edu (141.142.101.1)
CRAY-2 address:	u2.ncsa.uiuc.edu (141.142.102.2)
CM Sun Front-Ends:	cmsun1.ncsa.uiuc.edu (141.142.220.4)
	cmsun2.ncsa.uiuc.edu (141.142.220.7)
	cmsun3.ncsa.uiuc.edu (141.142.220.8)
VAX/Ultrix gateway	ncsagate.ncsa.uiuc.edu (141.142.121.38)
	ncsagate (BITNET)
Dial-up Access	(217) 244-0664 (2400 Baud or less)
	(217) 244-0662 (9600 Baud or less)
Anonymous FTP:	ftp.ncsa.uiuc.edu (141.142.20.50)

Login Script

The following script shows how to TELNET to the NCSA CRAY, list files, compile and run a job, and logout. The script assumes that you connect to the CRAY from a local machine which supports the TCP/IP program TELNET. Prompts are represented by "user>>" for you, "CRAY>>" for the CRAY, "TELN>>" for TELNET, and "CRus>>" for CRAY responses followed by user entries on the same line. Most CRAY responses are abbreviated. Comments are preceded by parentheses.

Prompt	Dialog	Comments
user>>	telnet u2.ncsa.uiuc.edu	(TELNET to the CRAY-2)
TELN>>	...	(TELNET messages)
CRAY>>	CRAY-2 UNICOS 5.0.7 (u2)	
CRAY>>	National Center for Supercomputing Applications	
CRus>>	login: your-CRAY-login	(login to the CRAY)
CRus>>	Password:your-password	
CRus>>	account:your-account	

```
CRAY>>                                                  (CRAY login messages)
CRus>>      u2 1% ls -la                                (ask to list your files)
CRAY>>      .                   ..          .cshrc
            .login              .profile    hello.f
CRus>>      u2 2% cft77 hello.f                         (compile program called
                                                        "hello.f", generate a
                                                        binary object file
                                                        called "hello.o")
CRus>>      u2 3% segldr -o hello hello.o               (load the object file and
                                                        create an executable
                                                        file called "hello")
CRus>>      u2 4% hello                                 (submit executable file)
CRus>>      u2 5% cat fort.6                            (display contents of
                                                        output file)
CRAY>>      Hello world, goodbye world
CRus>>      u2 6% logout                                (logoff CRAY and return
                                                        to local machine)
```

NATIONAL ENERGY RESEARCH SUPERCOMPUTER CENTER (NERSC)

Computer Hardware

- One CRAY 2/8, two CRAY 2/4s, and an XMP 1/8 for High School Supercomputer Honors Program (one of the CRAY 2/4s will be removed sometime in early 1992).
- Memory sizes are 128 Mwords on the CRAY 2's
- Vector and parallel hardware; 2 processors on the CRAY X-MP/24, 8 processors on one of the CRAY 2s, and 4 processors on each of the CRAY-2s
- Permanent storage is implemented on the IBM-based CFS storage system
- 1500 Gbytes total mass storage
- Auxiliary computers include VAXs and other network computers

Software

- UNICOS 6.1 on the CRAY 2/8 and the CRAY XMP 1/8; The CRAY 2/4 using CRAY Time Sharing System (CTSS) will be discontinued early 1992; the other CRAY 2/4 is physically located in Colorado Springs, and runs CSOS (Colorado Springs Operating System, a derivative of UNICOS 5.1 developed by the Cray Computer Corporation).
- Auxiliary VAXs use VMS operating system
- Editors on the CRAYs include vi, EMACS, and standard Unix editors
- FORTRAN compilers: CF77 on UNICOS and F77 on CSOS
- Other languages: C, Pascal, and LISP compilers, and CRAY Assembler
- CFS is the software used for mass storage
- Example sets, sample programs, and extensive documentation can be viewed online through the DOCVIEW program
- Disks with the NERSC CTSSLINK software for IBM PCs and PC-compatibles are available free to users
- Many Unix utilities are available in the CTSS and UNICOS environments

Math

IMSL NAG SCILIB

Graphics

DISSPLA GKS GRAFLIB NCARGKS
TV80LIB

Training

- New user workshops are available both on site and for export
- Monthly on-site UNICOS classes
- Classes are offered on supercomputer related topics
- Work in progress to develop an online tutorial service
- A high school Supercomputing Honors Program is offered each summer

Collaborative Research

- E-mail is performed with MAIL and MAILX
- Bulletin Board Server and a Mail Server machine being developed
- MFENET/ESNET networks supported by this center provide for remote logins, e-mail, and file transfers to and from nodes on these networks and on the Internet

Documentation

- Monthly newsletter, "The BUFFER"
- News is supplied by a site-supported bulletin board called "news"
- A VMS-like HELP is available

Consulting

- Phone consulting Mondays through Fridays, 8:00--11:45 a.m., 12:45-4:45 p.m.
- Online e-mail consulting via mail to consultants@nersc.gov

Graphics Support

- A DICOMED camera is available to remote users for film, slides, and fiche
- Remote graphics terminals supported include most Tektronix displays and IBM PCs and compatibles
- Output is mailed free of charge to users at remote sites
- Facilities for PostScript output are being explored

Strengths

- Highly optimized vector processing
- Large memory machines
- A network connecting energy research sites provides for collaboration and exchange of information between researchers

Administrative

- Most allocations are made by applying to the Department of Energy (DOE)

Contact Summaries

Consulting:	(510) 422-1544 (800) 66NERSC
Accounts:(passwords and ID's):	(510) 422-2888
Allocations:	(510) 422-1544

Training:	(510) 422-1544
Hardware status:	(510) 422-4283
Network status:	(510) 422-4283
Direct CRAY connection:	f.nersc.gov (128.55.64.120)

Login Script

The following script shows how to TELNET to the NERSC CRAYs, login, transfer files from a local account to the CRAY using FTP (NERSC currently has a limited number of FTP functions installed), list files, compile/load/execute a program, and then logout. It assumes that you connect to the CRAY from some local machine which supports the TCP/IP TELNET program.

Prompts are represented by "user>>" for you, "TELN>>" for TELNET, "CRAY>>" for the CRAY, and "CRus>>" for CRAY responses followed by user entries on the same line. Comments are preceded by parentheses. Most CRAY responses are abbreviated using the "..." ellipsis notation. This script has been tested from the University of Washington:

Prompt	Dialog	Comments
user>>	telnet f.nersc.gov	(TELNET to the CRAY)
TELN>>	...	(TELNET messages)
CRus>>	ID: usernumber	(enter CRAY usernumber)
CRus>>	Password: password	(enter CRAY password)
CRAY>>	...	(CRAY login messages)
user>>	ftp	(start FTP)
CRus>>	FTP> connect 128.95.137.4	(make connection with FTP to your local account)
CRAY>>	...	(FTP messages)
CRus>>	MAX.U.WASHINGTON.EDU> login user-id ctrl-z password	
		(note that the user-id and password must be entered on the same line, the ctrl-z is required to mask the password)
CRus>>	MAX.U.WASHINGTON.EDU> get hello.for hello.f	
		(now get your program with FTP your local account and send to the CRAY)
CRAY>>	...	file transfer messages from FTP
CRus>>	MAX.U.WASHINGTON.EDU> quit	(quit FTP and return to CRAY session)
CRAY>>	...	(more FTP messages)
user>>	ls	(ask to list your files)
CRAY>>	hello.f	
user>>	CF77 hello	(compile and load your FORTRAN program)
user>>	ctrl-d	(control-d logs you off the CRAY)

PITTSBURGH SUPERCOMPUTING CENTER (PSC)

Hardware

Cray Y-MP/832 with 8 processors, 32 million words of memory, and a high speed (156 Mword/second) 128 million word Solid-State Storage Device (SSD)

- A maximum of 25 Mwords per user
- 6.0 ns clock; maximum speed 2.6 gigaflops
- High performance parallel and vector hardware
- Very large temporary storage per user
- Unlimited permanent storage in IBM based Common File System using hierarchical directories
- Users can request dedicated time.

Connection Machine CM-2 with 32,768 processors, 1 gigabyte of memory, and 10 gigabytes of high speed storage on the DataVault. The processors work in parallel. Each processor has 32 Kbytes of memory associated with it, for a total of 1 gigabyte for the machine.

- Users may use 8K, 16K, or 32K processors at once
- 1,024 64-bit floating point accelerator chips
- Maximum speed in the gigaflop range
- Temporary space and permanent storage space available on the Andrew File System (afs) through the CM-2's front-ends (see later for more information on afs)
- Driven by three front-ends: CMVAX, a VAX running Ultrix, and two SUN 4-470s running Unix, SUNA and SUNB. CM applications are executed on the front-ends. Those parts of the application which benefit from parallel processing are executed on the CM-2, while those which are serial remain on the front-end and are processed there.
- Users can request dedicated time.
- Three front-ends which drive the CM-2 (see section on CM-2 above)
- Two Vax 6420s (CPWSCA and CPWSCB) running VMS from which you can reach either the Y-MP or the CM-2's front ends. Users have 10,000 blocks of permanent storage and a large amount of temporary storage.
- Two Vaxstation 5100s (PSCUXA and PSCUXB) running Ultrix from which you can reach either the Y-MP or the CM-2's front-ends. These front-ends use afs as their file system. Users have 10,000 1024-byte blocks of permanent storage on afs and a limited amount of temporary space.

Software:

Y-MP

- UNICOS
- Editors include EMACS, vi, ed and LSEDIT (Language sensitive editor for CFT77)
- FORTRAN: CFT77, and FORGE (an aid to vectorization)
- CF77 compiling system includes CFT77, automatic optimization and autotasking, and loader.
- Other languages: C (with vectorization support), Pascal

- ❏ Mass storage: Common File System (CFS) with automatic tape storage and retrieval
- ❏ A large examples system, containing sample jobs, data and the resulting output are publicly available .
- ❏ Downloadable public domain software

CM-2

- ❏ Front-ends run either Ultrix or Unix.
- ❏ Editors on front-ends include EMACS, vi and ed
- ❏ Languages: FORTRAN (cmf), C and Lisp
- ❏ Mass storage: AFS (see below)
- ❏ Examples system containing sample jobs, data and the resulting output are publicly available

Biology

ALIGN	BROOKHAVEN	CLUSTER	CONSENSUS
CONSORT	ECD	EMBL	ENZYME GCG
GENBANK	IB_TO_GB	MAKSEQ	MAXSEGS
MDPP	MSA	MSC	NWGAP
PICKEMBL	PICKGEN	PROFILE	REBASE
SEQA	SEQH	SEQHP	XRFORM
SEQ_FILE	SEQUIN	XPLOR	

Chemistry

AMBER	AMPAC	AMPS	CADPAC40
CHARMM	CHELP	DISTAN	GAMESS90
GAUSSIAN	GPRLSA	GROMOS	MOPAC
MULFOLD	NBRF	POLYRATE	PROSITE
PROTIN	RANFOLD	RNAFOLD	SN
SP	ST	SWEDEN	SWISS

Computational Fluid Dynamics

FIDAP	FLOW3D	STEAM

Electrical

SPICE	SPLICE

Engineering

KIVA

FORTRAN Utilities

BENCHM	CHRPAK	CLEAN77	FORCHEK
FORTOOLS	FSPLIT	FTREF	LOOPMARK

RECIPES	TOKENS	TOOLPACK	V77

Graphics

ANIMATOR	CODEBOOK	DI3000	DISSPLA11
DRAW	DRAWCGM	DRAWP3D	EDA
GPLOT	GPLOTM	GRAPHX	GTEX
IMAGETOOL	LVR	MACGPLOT	MARCHP3D
MOVIE6	MOVIETOP3D	NAVIEW	NCARDSL
NCARGKS2	NCARGKS3	ORION	PAL2RGB
PDBMODEL	PLT2	PNMTOCGM	POLYP3D
REQUESTAPE	RLECAT	RLETOCGM	SGML
SLIDEMAKING	SMDLIB	SMONGO	TAURUS
TEXTMAKER	TIFFTORLE	TITLEMAKER	

Mathematics

ALFPACK	AMOS	AMOSLIB	APML
AUTO	BAILEY	BANDIT	BCSLIB
BCSLIBEXT	BDMG	BIHAR	BIVAR
BLAS	BORMAT	BSMP	BVPSOL
CALMATH	CGLIB	CMLIB	COLNEW
COMPAR	CYCLE	DASSL	DIERCKX
DRIV	DVERK	DYNA2D	DYNA3D
EFUN	EISPACK	EISPACKC	ELEFUNT
ELLPACK	ELLPACK	FFTPACK	FFTPACK
FISHPAK32	FITPACK	FN	FUNPACK
GAMS	GBSOL	GQOPT	HARWELL
HBSMC	HSSXEV	IMSLDOC	IMSL11
INGRID	INTLIB	ITPACK2C	ITPACKV2D
JAKEF	LAPACK	LASO	LBENCH
LBENCHC	LINPACK	LINWOOD	LLSQ
LOPSI	LSOLVE	LSQR	LUSP
MAPLE	MATHLIB	MATMUL	MFFT
MINPACK	MP	MUDPACK	NAG
NAPACK	NL2SOL	NMS	ODE
ODEPACK	PCHIP	PDECOL	PDEONE
PITCON	PLTMG	POLPAK	QFEPDE
QUADPACK	RANPACK	RECIPESC	REDUCE
RKF45	SCILIB	SGEFAC	SLAP
SLATEC31	SPARBLAS	SPARSE	SPARSKIT
SPARSPAK	SPECFUN	SPLPAK	SSDLIN
SUBSET	SYMPAK	TESTLP	TESTMAT
TESTNLS	TESTNON	TOEPLITZ	TOMS
UNCMIN	VCFT	VE1	VFFTPK
Y12M	YSMP	ZERO	

Mechanics

ABAQUS	ANSYS	BOPACE	FACET
MAZE	NIKE2D	NIKE3D	SAP4
TOPAZ2D	TOPAZ3D		

Programming Languages

ALISP	C	C++	CAL
FORTRAN	GCC	LISP	PASCAL

Simulation

CASCADE

Statistics

APSTAT	ODRPACK	ODRPACKX	STARPAC
STATLIB	TLS		

Utilities--Benchmarks/Debugging/Optimization/Tracing/Vectorization

BOUNDS	CDBX	CM_TIMER	DEBUG
DHRYSTONEC	EXPORTS	FLOWTRACE	FORGE
HPM	LIVERMORE	MACHAR	MACHINE
NASKER	PARANOIA	PERFECT	PERFTRACE
PROCSTAT	PROF	PSR	SCOUNT
SLALOM	TESTUCON	TESTVECTOR	WHETSTONE

Utilities--File Conversion / Processing

COMPRESS	CONVERT	D2S	DETAB
ETAPE	LINES	PAD	SGREP
STRIPPER	TAIL	TAR2VMS	UUENCODE
VMS2TAR	XTRACT		

Utilities--File Systems

AFS	CFS

Utilities--Job handling, Accounting

BENCHLIB	CMNQS	DEDCPU	DHSC
HELPER	NQS	PSCSUB	YJANUS
YSUB			

Utilities--Miscellaneous

BISON	CDECL	CONDENSE	CSORT
SETUP	SPR		

Andrew File System:

The Andrew File System (afs) is a distributed network file system used on PSC front-ends PSCUXA, PSCUXB, CMVAX, SUNA, and SUNB. PSC plans to make afs available on all of its machines. Users

therefore maintain one set of files, accessible from all machines. It is not necessary to keep multiple copies of files or to transfer files back and forth between the various front-ends. Automatic daily backups are done. PSC maintains documentation, examples, libraries, executables and source code on afs in a single location which is accessible from all machines. Each user has a quota of 10,000 1024-byte blocks of permanent storage. A limited amount of temporary storage is also available.

The Biomedical Initiative:

PSC received a grant from the National Institutes of Health's (NIH) Division of Research Resources' Biomedical Research Technology program to provide the biomedical community with supercomputing resources, training, and user support. Two biomedical specialists, with large-scale computational backgrounds, are available to support the supercomputing activities.

Distributed High Speed Computing:

The Distributed High Speed Computing (DHSC) project at PSC aims to provide an easy to use interface to the HiPPI (High Performance Parallel Interface) link between the Center's Cray YMP/832 and 32K processor Connection Machine. This HiPPI network link between the Cray and the CM is capable of transmitting data at rates of up to 100 MBytes/Sec. Through the use of Distributed High Speed Computing, users are able to partition applications between the Cray and the CM, thus choosing the most appropriate type of supercomputer to solve different portions of the application. Because of the vastly different kinds of problems which can be solved on the Cray and the Connection Machine, some applications will see significant speedups in overall execution time via this approach.

The DHSC project is embodied in a set of library routines, callable from CFT77 and Standard C on the Cray, and CMFortran, C*, and C-Paris on the Connection Machine. These library calls provide a simple interface to transfer data between programs running in parallel on the two machines. Control issues (such as synchronization of flow) and data conversion issues (such as parallel to serial and IEEE to Cray floating point conversions) are handled internally by the libraries, freeing the user from these low level concerns.

Training:

New User Workshops

- ❏ Cray Techniques workshop (5 days)
- ❏ Connection Machine Techniques workshop (5 days)
- ❏ Summer Institute (2 weeks)

Advanced-User Workshops

- ❏ Code Optimization (one day)
- ❏ Multitasking (one day)
- ❏ Visualization (one day)
- ❏ Optimization of FORTRAN Code for Parallel Shared-Memory Systems (4 days)

Discipline-Specific Workshops

- Introduction to Gaussian: Theory and Practice (5 days)
- National Institutes of Health Workshops

Through grants from the National Institutes of Health, the PSC offers a series of workshops dedicated to the biomedical research community, including an overview of biomedical software and databases available at PSC. Topics presented have included molecular mechanics and dynamics, macromolecular structure refinement, nucleic acid and protein sequence analysis, fluid dynamics with immersed flexible structures, and image reconstruction from Electron Microscopy. Topic-specific workshops are provided on an "as-needed" basis and new workshops are under continual development.

Documentation:

- Many vendor and hundreds of PSC publications online, including policy, allocations and user guides
- CRAY manuals mailed via UPS
- List of documents is available online
- Hot tips and news appear in BULLETIN, and rn. BULLETIN and rn are bulletin board systems on the VMS front-ends and the Ultrix front-ends, respectively. Notices are also posted on 'news' on the Y-MP.
- Online help includes man pages on every PSC system.
- PSC Users Guide mailed to all users.
- Online EXAMPLES system contains complete job setups.
- An annual report "Projects in Scientific Computing" highlights the scientific projects completed at PSC

Consulting:

- Phone consulting available M-F 9 a.m. to 8 p.m., Saturday 9 a.m. to 4 p.m. (Eastern time)
- E-mail consulting via user-id REMARKS
- Online man pages
- Discipline-specific consulting
- "life-cycle" consulting: starting with algorithm selection through data layout, programming, and program optimization for CM users.

Graphics Support:

- CGM metafile support
- CGM-based video animation system for VHS tape and videodisk
- Support for remote graphics terminals includes many devices; also PSC is willing to acquire or develop drivers for nearly any modern device on request.
- VHS tape output mailed to remote users
- On-site graphics labs with Silicon Graphics IRIS, SUN Sparcstation, IBM RS6000, and DECStation 5000.
- Capability to make 35mm color slides using Matrix Instruments SlideWriter
- Color output from Cannon CLC-500 color copier/printer

Administrative:

- ❑ Small allocations (10 hours or less) are reviewed quickly by PSC staff. Larger allocations (over 100 hours) are reviewed by the joint PSC/NCSA Peer Review Board. Intermediate size allocations (11-99 units) are reviewed internally subject to confirmation by outside reviews via U.S. Postal mail.
- ❑ Starter grants are available for any researcher who wishes to explore the use of the PSC.

Contact summaries:

Consulting	(412) 286-4960
Accounts	(412) 286-4960
Allocations	Allocations Coordinator (412) 286-4960 or stock@cpwsca.psc.edu
Training	(412) 286-4960
Hardware status	(412) 286-4960
Network status	(412) 286-4960
Front-end Internet addresses	a.psc.edu (VMS) b.psc.edu (VMS) pscuxa.psc.edu (Ultrix) pscuxb.psc.edu (Ultrix)
CM-2 front-end Internet addresses	cmvax.psc.edu (Ultrix) suna.psc.edu (Ultrix) sunb.psc.edu (Ultrix)
Cray Internet address	pscymp.psc.edu

Login Script

The following script shows how to connect to the font end VMS VAX, login, create and a job which submits 'hello.f' to the CRAY, and then logout. It assumes that you connect to the front end from some local machine which supports the TCP/IP TELNET program.

Prompts are represented by "user>>" for you, "PSC>" for the frontend, "TELN>>" for TELNET, and "PSCus>>" for frontend responses followed by user entries on the same line. Most responses are abbreviated. Comments are preceded by parentheses.

Prompt	Dialog	Comments
user>>	TELNET a.psc.edu	(TELNET to the front end)
TELN>>	...	(TELNET messages)
PSCus>>	Username: user-id	
PSCus>>	Password: password	
PSC>>	...	(PSC login messages)
user>>	emacs hello.job	(create the following job (to submitted to the CRAY)
	#USER=user-id PW=password	(CRAY from the VAX later
	#QSUB -lM 0.8 Mw	with the csubmit command)
	#QSUB -lT 5	
	ja	

```
                date
                set -x
                cd $TMP
                fetch hello.f -t 'usr$root0:[user-id]hello.f
                cf77 hello.f limsl11
                mv a.out hello.exe
                hello.exe > results
                cfs -r5 store results
                rm hello.exe hello.f
                date
                ja -st

user>>          cray
PSC>>           The Cray Station is available
user>>          csubmit hello.job
PSC>>           %CX-S-SUB_OK, Job: Hello  queued for submission
PSC>>           VAX TO CRAY:%SYSTEM-S-NORMAL, normal successful
                completion
PSC>>           VAX TO CRAY: FILE=HELLO
PSC>>           VAX TO CRAY: 1215 BYTES TRANSFERRED
PSC>>           CRAY TO VAX: %RMS-S-NORMAL, normal successful completion
PSC>>           CRAY TO VAX: FILE=$1$DUA102:[USR2.USERID]HELLO.OUT;1
PSC>>           CRAY TO VAX: 34 BYTES TRANSFERRED
user>>          logout
```

SAN DIEGO SUPERCOMPUTER CENTER (SDSC)

Computer Hardware

- ❏ CRAY Y-MP8/864 running UNICOS
- ❏ Up to 32 Mwords main memory per user
- ❏ 6 ns clock; a maximum of 2.7 Gflops for the entire system; up to 337 maximum Mflops per CPU
- ❏ Vector processing hardware
- ❏ 65 Gbytes online temporary storage
- ❏ 128 MWord SSD
- ❏ Intel iPSC/860, 64 nodes
- ❏ 8 MBytes of memory per node totaling 512 MBytes
- ❏ Peak speed of 2.6 Gflops (for 32-bit arithmetic)
- ❏ nCUBE 2, 128 nodes
- ❏ 64 nodes with 16 MByes of memory each
- ❏ 64 nodes with 4 Mbytes of memory each
- ❏ Total memory 1.3 Gbytes
- ❏ Peak speed of 420 Mflops (for 32-bit arithmetic)
- ❏ DataTree for file storage on an Amdahl 5860

Software

- ❏ UNICOS on the CRAY Y-MP
- ❏ Editor: vi, emacs, and standard Unix line editors on the CRAY Y/MP
- ❏ FORTRAN: CFT77 vectorizing compiler
- ❏ Other languages: Pascal (vectorizing), CC (ANSI C vectorizing compiler), CRAY C, AS (CRAY Assembly language)
- ❏ Extensive example programs are available.

Biology

DRAW	FASTA	GenBank DB	PIR DB
RANFOLD	RNAFOLD	SN	

Chemistry

AMBER	AMPAC	BIGSTRN-3	CADPAC
CAMBGIDGE DB	CHARM	CHELP	CORMA
DGEOM	DISCOVER	DMOL	ECEPP2
GAMESS	GAUSSIAN 88	GAUSSIAN 90	GROMOS
MM2	MMTOOLS	MMX	MOPAC
MPLOT	POLYRATE	PROLSQ	PSI77
QCFF/PI	VENUS	X-PLOR	

Electrical

HSPICE	NETSCAT	PRECISE	SPICE
SUPREM	UMSPICE		

Mechanics

ABAQUS	ADINA	ANSYS	CONTINUSYS
CSQ	DYNA2D	DYNA3D	FACET
FIDAP	FLOTRAN	FLOW-3D	FLUENT
FLUENT/BFC	INGRID	INS3D	MAZE
MOLDFLOW	MPGS	MSC/NASTRAN	NIKE2D
NIKE3D	ORION	STARDYNE	TAURUS
TOPAZ2D	TOPAZ3D		

Math

BCS	BCS-EXT	DASSL	ELLPACK
HARWELL	HOMPACK	IMSLMATH	IMSLSFUN
IMSLSTAT	ITPACKV	LAPACK	LASO
LIBSCI	MATLAB	MINOS	MINPACK
MUDPACK	NAG	NREG77	NSPCG
ODEPACK	OMNI	PLTMG	SLAP
SLATEC	SMPAK	SPARSPAK	SPSS-X
TOEPLITZ	TOMS	VECTFFT	

Graphics

DISSPLA	GKSNCAR	GKSUL	GNUPLOT
HDF	MOVIE.BYU	NCARGRAPHICS	PLOTXY
VOGLE	XDATASLICE	XIMAGE	

Nuclear Engineering

DIF3D	DOS	ENDF DB	MCNP
NJOY	REBUS	SABRINA	TRANSX
TWOHEX			

Training

- ❑ Two-day workshops for new users are held at the beginning of each quarter and at remote sites when requested.
- ❑ Advanced workshops are available on specialized topics, such as vectorization and visualization, both on-site and for export.
- ❑ Summer Institute for those with developed code: the applications are reviewed by an SDSC committee and are open to faculty, graduate students, undergraduates, and industry.

Collaborative Research

- Visiting scientists program
- Research Fellows program for undergrads, postdoctoral, and senior staff

Documentation

- Over 150 complete user documents are available online for viewing and/or printing.
- Over 3,000 man pages are online including over 300 locally developed.
- Hot tips and news is available in NEWS.
- The bi-monthly newsletter is called "Gather/Scatter."
- Users Guide is given to all users.
- Extensive examples and instructions on their use is available online.
- Annual report titled "Science at the San Diego Supercomputer Center 19xx"
- An overview brochure that includes an application for computing resources

Consulting

- Phone consulting is available 8am-5pm Monday-Friday Pacific time.
- Online consulting is through MAIL CONSULT@Y1.SDSC.EDU
- Discipline-specific consulting is available in most applications.

Graphics Support

- A Management Graphics Solitaire-8 digital film recorder
- Matrix 6264 analog film recorder
- Lasergraphics LFR Macintosh film recorder
- Hewlett Packard Paintwriter color Macintosh printer
- Complete audio and video post-production facility for animations
- Complete set of image manipulation and conversion software
- Rendering software including Alias, Wavefront, and Renderman
- Output mailed to your site

Strengths

- MIMD parallel computing
- Networking--infrastructure and research
- Visualization
- Large collection of applications and math software
- High performance vectorization

Administrative

- ❑ Allocations for large blocks of time are reviewed by an allocation committee and must be submitted 60 days prior to each quarter
- ❑ Small allocations (1-20 hours) may be submitted any time

Contact Summaries

Consulting:	(619) 534-5100 CONSULT@Y1.SDSC.EDU
Accounts:	(619) 534-5100
Allocations:	Dr. Rozeanne Steckler (619) 534-5120 steckler@sds.sdsc.edu
Training:	Ms. Jayne Waggoner (619) 534-5124 waggoner@sds.sdsc.edu
Hardware status:	(619) 534-5100
Network status:	(619) 534-5100
CRAY Internet address:	y1.sdsc.edu (132.249.10.1)

Login Script

The following script shows how to TELNET to the San Diego CRAY, login, list files, compile and execute a simple FORTRAN program, and then logout. It assumes that you connect to the CRAY from some local machine which supports the TCP/IP TELNET program.

Prompts are represented by "user>>" for you, "TELN>>" for TELNET, "CRAY>>" for the CRAY, and "CRus>>" for CRAY responses followed by user entries on the same line. Comments are preceded by parentheses. Most CRAY responses are abbreviated using the "..." ellipsis notation. Note that the system prompts on the SDSC CRAY Y/MP currently have the form "y1-n%" where "n" is the line number

Prompt	Dialog	Comments
user>>	telnet y1.sdsc.edu	(TELNET to the CRAY)
TELN>>	...	(TELNET messages)
CRus>>	login: your-CRAY-login	(login to the CRAY)
CRus>>	Password: your-password	
CRAY>>	...	(CRAY login messages)
CRus>>	y1-1% ls hello.f	(ask to list your files)
CRus>>	y1-2% cf77 -o hello hello.f	(compile "hello.f" code and generate executable file called "hello"
CRus>>	y1-3% hello	(execute "hello" program)
CRus>>	y1-4% logout	(logoff CRAY, return to your local machine)