Innovative Computing Laboratory

ICL UT 2001 REPORT

COMPUTER SCIENCE DEPARTMENT
UNIVERSITY OF TENNESSEE

Innovative Computing Laboratory 2001 REPORT

The University of Tennessee - Knoxville
EEO/Title IX/Section 504/ADA Statement

The University of Tennessee does not discriminate on the basis of race, sex, color, religion, national origin, handicap, or veteran status in provision of educational opportunities and benefits. This policy extends both to employment by and admission to t he University.

UT does not discriminate on the basis of race, sex, or disability in the education programs and activities which it operates, pursuant to requirements of Title VI of the Civil Rights Act of 1964, Title IX of the Educational Amendments of 1972, section 504 of the Rehabilitation Act of 1973, and the Americans with Disabilities Act (ADA) of 1990.

Edited by Scott Wells
Designed by David Rogers

Inquiries and charges of violation concerning Title VI, Title IX, Section 504, ADA or the Age Discrimination in Employment Ace (ADEA), or any of the other above referenced policies should be directed to the Office of Diversity Resources and Educational Services (DRES), 1818 Lake Avenue, Knoxville, TN 37996-3560, telephone (865) 974-2498 (TTY available). Requests for accommodation of a disability should be directed to the ADA Coordinator at the Office of Human Resources Management, 600 Henley Street, Knoxville, TN 37996-4125.

ICL 2001 REPORT
CONTENTS

INTRODUCTION

MISSION STATEMENT

The guiding mission of the Innovative Computing Laboratory (ICL) is to push back the frontiers of discovery in high performance and distributed computing in the 21st Century and to infuse the innovations flowing from those discoveries into the leading edge of scientific and real-world applications. The pace and potential for progress of scientific inquiry across a vast spectrum of disciplines is now intimately linked with advances in scientific computing and with the creation of new software tools that puts this power into the hands of domain scientists and engineers. ICL, and the richly talented group of people who have brought about its ongoing success, stands committed to leadership in this new era of scientific simulation, in which the cooperative efforts of Computer and Computational Scientists propel scientific research to new and unparalleled knowledge of the world around us.

ICL aspires to be a world leader in enabling technologies and software for scientific computing. Our vision is to provide high performance tools to tackle science's most challenging problems and to play a major role in the development of standards for scientific computing in general.

BACKGROUND

ICL was established in the fall of 1989 when Dr. Jack Dongarra came to the University of Tennessee (UT) from Argonne National Laboratory (ANL). Dr. Dongarra was given a dual appointment as Distinguished Professor in the Computer Science Department at the university and as Distinguished Scientist at Oak Ridge National Laboratory (ORNL). This dual position was established by the UT/ORNL Science Alliance, Tennessee's oldest and largest Center of Excellence, as a means for attracting top research scientists from around the country and the world to visit the university and collaborate. Subsequently, many post-doctoral researchers and professors from various research backgrounds such as mathematics, geology, chemistry, etc. visited the university. Many of these scientists have passed through UT as post-doctoral researchers and worked with Dr. Dongarra. Subsequently, these scientists were vital in helping him attract additional researchers as well as top graduate students. Below is a list of some of the original researchers who were instrumental in helping Dr. Dongarra with the establishment and growth of ICL.

- Zhaojun Bai, University of California Davis
- Adam Beguelin, NeoPyx
- Susan Blackford, Myricom
- Jaeyoung Choi, Soongsil University (Korea)
- Andy Cleary, Lawrence Livermore National Laboratory
- Frederic Desprez, ÉNS Lyon
- Robert van de Geijn, University of Texas Austin
- Robert Manchek, Mediaone
- Roldan Pozo, National Institute of Standards and Technology
- Françoise Tisseur, Manchester University
- Bernard Tourancheau, Université Claude Bernard de Lyon and Sun Microsystems, France

Through interactions with colleagues at Rice University, ICL became an integral part of the Center for Research on Parallel Computation (CRPC), a National Science Foundation (NSF) Science and Technology Center established in 1989 and lead by Rice University. CRPC worked to make parallel computation accessible to industry, government, and academia and to educate a new generation of technical professionals. Through the 1990s, ICL worked on a number of efforts that have since become part of the basic fabric of scientific computing in the world. Our enabling technology efforts include the BLAS, LAPACK, ScaLAPACK, PVM, MPI, Netlib, NHSE, and the TOP500. These successes are continuing along with current ICL efforts such as ATLAS, PAPI, HARNESS, DSI, IBP, RIB, and NetSolve. In 1999, two of our projects were awarded R&D 100 awards: ATLAS and NetSolve. This year, another of our projects, PAPI, was recognized with an R&D 100 award.

Having linear algebra as its original focus, the group has evolved and expanded to focus on many progressive areas of high performance computing research such as distributed computing, software repositories, and performance evaluation. Currently a University Distinguished Professor, Dr. Dongarra continues as director of ICL. As such, he not only serves as principal investigator (PI) for many of our projects, but he also maintains some level of participation in all projects.

PROFILE

Residing at the heart of the UT campus in Knoxville, ICL is an internationally recognized academic leader in high performance computing (HPC) research. We have recently moved into new facilities at the university, due in large part to our incredible growth over the last few years. Located in the recently built Claxton building, ICL and the Computer Science Department occupy nearly ¾ths of the approximately 70,000 square foot facility.

Claxton Building - Home of ICL

Operating under grants totaling nearly $5 million annually, ICL is also recognized by senior UT administration as one of UT's top three research centers. According to Dr. T. Dwayne McCay, Vice President of Research and Information Technology, "Jack Dongarra and the students and staff of the Innovative Computing Laboratory have been leading our University and our nation in high performance computing and information technology research for more than a decade now. Looking out at the decade to come, we're really excited by the prospect of what their creativity and their new discoveries will bring us. It is the work of Dr. Dongarra's team that exemplifies why we are so determined to be a top 25 research university, the efficient solution to modern problems requires the most modern tools and the best trained minds and they are attracted to the great research universities by the opportunity to work with someone like Jack Dongarra. These universities in turn fuel the engine that drives the economy which raises the standard of living for all our citizens."

FROM THE DIRECTOR

In 2001, the ICL is celebrating 12 years of leadership in enabling technologies for high performance computing. Looking back over the 12-year period, the evolution and growth of the technology for computing has been truly astonishing. In an environment where technology changes every 18 months, ICL cannot afford to stand still. In 1989 the speed of a supercomputer was measured in gigaflops and in gigabytes. Today our measures are teraflops for speed and terabytes for memory, a thousand-fold increase over the standards of a decade ago. The research that ICL has undertaken in the past decade has followed a natural progression and growth from our original tread of numerical linear algebra to performance evaluation, to software repositories, and to distributed computing.

The ICL staff's ongoing ability to apply the latest technologies to provide advanced services and solutions for the scientific computing community underscores the ICL's leadership role. Standards and efforts such as PVM, MPI, LAPACK, ScaLAPACK, BLAS, Netlib, NHSE, TOP500, and the LINPACK Benchmark have all left their mark on the scientific community. We are continuing these efforts with ATLAS, PAPI, NetSolve, RIB, TOP100 Clusters, HARNESS, and Self-Adapting Numerical Software Effort (SANS-Effort), as well as other innovative computing projects.

There have also been a number of changes in the location and personnel at ICL. As we settle in to our first year in the Claxton building things are beginning to seem not so strange and awkward. We continue to grow in terms of the resources we have at our disposal. We have expanded our research portfolio with DOE's SciDAC, NSF's NGS, and DoD PET-II efforts. We have ongoing efforts to strengthen our organization and to ensure the proper balance and integration of research and projects. We have also had a number of modifications to our organizational structure with the spin off of the Logistical Computing and Internetworking (LoCI) Lab and the formation of our Research Center of Excellence, Center for Information Technology Research (CITR). The pace of change will continue to accelerate in the coming years.

During these exciting times, I am grateful to our sponsors for their continued endorsement of our efforts. My special thanks and congratulations go to the ICL staff and students for their skill, dedication, and tireless efforts in making the ICL one of the best centers for enabling technology in the world.

-Jack Dongarra

RESEARCH PROJECTS

INTRODUCTION

As ICL has grown over the years, the range and diversity of the research and development carried out by our staff and students has increased in parallel. In the past year alone, we supported or participated in more than 20 significant projects. Our large and wide-ranging portfolio of research projects has evolved over the course of a decade, beginning from a narrow but solid foundation.

The original focus of ICL was Dr. Dongarra's work in numerical linear algebra and the numerical libraries that encode its operations in software. But driven by the relentless demand for higher performance in the computational science community, ICL built upon its successes in the area of numerical libraries and the growing strength of its personnel to break new ground in the areas of high performance parallel and distributed computing. Similarly, our work with numerical libraries created a strong area of expertise in performance evaluation and benchmarking for high-end computers. The enormous investments by both government and private industry in high performance computing have made our ability to do research in this area correspondingly important. Finally, as a by-product of a long tradition of delivering high quality software produced from our research, we have helped to lead the movement to build robust, comprehensive, and well-organized software repositories.

With the phenomenal growth over the last several years in parallel computing technology and the demands placed on such technology by government and private business, we are consistently challenged to apply expert-level understanding to each of our research efforts. The areas of distributed and network computing are no exception as we've learned to harness enormous computing power to quickly and efficiently solve mathematical problems that would take humans years or decades to solve by hand.

ACKNOWLEDGEMENT

On the following pages, many of the projects in our main research areas - numerical libraries, high performance distributed computing, performance evaluation and benchmarking, and software repositories - are described in detail. Evidence of the perceived value of this work is apparent in the range of agencies and organizations that have funded these efforts. The main source of support has been federal agencies that are charged with investing the nation's research funding: the National Science Foundation (NSF), Department of Energy (DOE), Department of Defense (DoD), and the National Aeronautics and Space Administration (NASA). However, strong support from private industry has also played a significant role. Some companies have targeted specific ICL projects. But others have made contributions to our work that are more general and open-ended. We gratefully acknowledge the following for their generosity and their significance to our success:

AMD
Compaq
Hewlett-Packard
IBM
Intel
MathWorks
Microsoft
Myricom
Scali
Sun Microsystems

RESEARCH PROJECTS
AREAS OF EXPERTISE

NUMERICAL LINEAR ALGEBRA

ATLAS
BLAST/LAPACK/SCALAPACK
JLAPACK/F2J
SPARSE MATRIX ALGORITHMS AND SOFTWARE

ICL has long been a leader in the area of numerical linear algebra algorithms and software for high performance computers. Linear algebra operations form the core of an overwhelming number of scientific applications. Having efficient algorithms and implementations for these operations is of utmost importance in achieving good performance for these applications. In collaboration with other researchers and with industry, ICL has led successful efforts to standardize library interfaces for the Basic Linear Algebra Subroutines (BLAS) as well as for higher level dense linear algebra routines such as those for solving linear systems. ICL's groundbreaking research in efficient algorithms and implementations for these routines has been widely adopted and refined by industry to produce highly efficient linear algebra implementations for most architectures, with the result that applications that rely on the standard libraries can achieve excellent performance while remaining portable across multiple platforms. Dense linear algebra software produced by ICL research has included LINPACK for vector supercomputers, LAPACK for shared memory multiprocessors, and ScaLAPACK for distributed memory multiprocessors. More recently, the Automatically Tuned Linear Algebra Software (ATLAS) system has been developed for the automatic generation and optimization of linear algebra software. ATLAS has been widely adopted by vendors to produce efficient BLAS for their machines in a fraction of the time required for hand coding. Recent ICL linear algebra research has focused on sparse linear algebra in the areas of iterative methods and parallel preconditioning and on smart libraries for partial automation of the choice of method and preconditioner.

DISTRIBUTED COMPUTING

FT-MPI
GRADS
HARNESS
I2-DSI
IBP
MPI_CONNECT
NETSOLVE
SINRG
TORC

Distributed computing is another major area of research for ICL. ICL researchers have been involved in a number of distributed computing projects over the past decade. To allow scientific applications to run in parallel across networks of workstation, ICL researchers, in collaboration with Oak Ridge National Laboratory and Emory University, developed the highly successful Parallel Virtual Machine (PVM) system. Although originally intended for cluster computing, most major high performance computing vendors adopted PVM as a *de facto* standard for message passing on distributed memory multiprocessors. The PVM system transparently handles message routing, data conversion for incompatible architectures, and other tasks necessary for operation in a heterogeneous, network environment. Although largely surpassed by the industry-standard Message Passing Interface (MPI), PVM remains widely used and is particularly effective for heterogeneous applications that exploit specific strengths of individual machines on a network.

ICL helped lead the effort to standardize message passing in the form of MPI. In addition, the MPI-Connect and FT-MPI projects have addressed the needs for MPI interoperability and fault tolerance, respectively, in a networked heterogeneous environment. FT-MPI is part of a broader project called HARNESS, which is based on the concept of a distributed virtual machine and provides a plug-in interface for dynamically customizing, adapting, and extending a heterogeneous network-computing environment.

The NetSolve client-server system is another ICL distributed computing project. The goal of NetSolve is to provide users with easy access to remote hardware and software computational resources in a networked environment. NetSolve consists of computational servers, agents that handle scheduling decisions, and a variety of client interfaces. The NetSolve team is collaborating with other researchers in the area of grid computing to provide users with access to a growing network of computational services.

BENCHMARKING AND PERFORMANCE EVALUATION

LINPACK BENCHMARK
PAPI
TOP500 SUPERCOMPUTERS/
TOP100 CLUSTERS

In addition to producing software that helps achieve high performance on parallel computers, ICL has been a leader in benchmarking and performance evaluation efforts that measure and report performance of these machines. ICL researchers have developed a number of benchmark codes. The LINPACK Benchmark is a numerically intensive test that has been used for years to measure the floating-point performance of computers. Performance on this benchmark is the basis of the semi-annual TOP500 list that ranks the fastest 500 computers in the world. HPL is a portable high-performance implementation of the LINPACK Benchmark for distributed memory computers. SparseBench uses common iterative methods, preconditioners, and storage schemes to evaluate machine performance on typical sparse operations.

In addition to benchmarks, ICL researchers have developed a portable library interface, called PAPI, for access to hardware performance counters on most modern microprocessors. PAPI not only provides a standard set of routines for accessing counter data, but also defines a common set of performance metrics considered relevant and useful for application performance tuning. The PAPI library interface and reference implementations are becoming widely used by performance tool developers. By using PAPI, tool developers can devote most of their effort to tool design rather than to re-implementing access to the counters.

SOFTWARE REPOSITORIES

NA-NET/NA-DIGEST
NHSE
NETBUILD
NETLIB
RIB

ICL and other academic researchers have produced a wealth of numerical and high performance computing software. To preserve this software past the lifetime of the projects that produced it, and to provide a central location containing a moderated collection of high-quality software, a software repository effort was begun in the 1980s. The original Netlib collection of mathematical software has grown to include software and technical reports in a variety of areas, including dense and sparse linear algebra, differential equations, optimization, and parallel tools. The Netlib collection is widely used by academic and government researchers as well as by industry and has received over 90,000,000 requests over its 15-year lifetime. The National High-performance Software Exchange (NHSE) project, which started in 1994, sought to enable other organizations and high performance computing areas to replicate Netlib's success in sharing software resources. The Repository in a Box (RIB) toolkit developed by the NHSE has been used by a number of government agencies to set up repositories in the areas of parallel tools, computational chemistry, signal image processing, and grand challenge applications. RIB's interoperation capabilities allow these repositories to share software information. Netlib and the NHSE and RIB repositories make a vast amount of high quality software easily available to users via the web, thus accomplishing technology transfer on a large scale. Together with the NetSolve system described above, Netlib and RIB are also being used to provide an active inquiry-based numerical software collection for engineering education.

ATLAS

AUTOMATICALLY TUNED LINEAR ALGEBRA SOFTWARE

R&D 100
1999
WINNER

DEVELOPMENT TEAM
Clint Whaley
Jack Dongarra

WEB SITE
http://icl.cs.utk.edu/atlas/

E-MAIL
atlas@cs.utk.edu

AGENCY FUNDING
Department of Energy
National Science Foundation

ATLAS (Automatically Tuned Linear Algebra Software) is an instantiation of a new paradigm in high performance library production and maintenance, which we term AEOS (Automated Empirical Optimization of Software). This style of library management has been created in order to allow software to keep pace with the incredible rate of hardware advancement inherent in Moore's Law. ATLAS is the application of this new paradigm to linear algebra software, with the present emphasis on the Basic Linear Algebra Subprograms (BLAS), a widely used, performance-critical, linear algebra kernel library.

Linear algebra routines are widely used in the computational sciences in general, and scientific modeling in particular. In many of these applications, the performance of the linear algebra operations is the main constraint preventing the scientist from modeling more complex problems, which would then more closely match reality. This then dictates an ongoing need for highly efficient routines; as more compute power becomes available the scientist typically increases the complexity/accuracy of the model until the limits of the computational power is reached. Therefore, since many applications have no practical limit of enough accuracy, it is important that each generation of increasingly powerful computers have optimized linear algebra routines available.

Linear algebra is rich in operations, which are highly optimizable, in the sense that a highly tuned code may run multiple orders of magnitude faster than a naively coded routine. However, these optimizations are platform specific, such that an optimization for a given computer architecture will actually cause a slow-down on another architecture. The traditional method of handling this problem has been to produce hand-optimized routines for a given machine. This is a painstaking process, typically requiring many man-months of highly trained (both in linear algebra and computational optimization) personnel. The incredible pace of hardware evolution makes this technique untenable in the long run, particularly so when considering that there are many software layers (e.g., operating systems, compilers, etc), which also affect these kinds of optimizations, that are changing at similar, but independent rates.

Therefore, a new paradigm is needed for the production of highly efficient routines in the modern age of computing, and ATLAS represents an implementation of such a set of new techniques. In an AEOS-enabled package such as ATLAS, the package provides many ways of doing the required operations and uses empirical timings in order to choose the best method for a given architecture. Thus, if written generally enough, an AEOS-aware package can automatically adapt to a new computer architecture in a matter of hours, rather than requiring months or even years of highly trained professionals' time, as dictated by traditional methods.

ATLAS typically uses code generators (i.e., programs that write other programs) in order to provide the many different ways of performing a given operation and has sophisticated search scripts and robust timing mechanisms in order to find the best ways of performing the operation for a given architecture.

The BLAS application programming interface (API) is supported by hand-tuned efforts of many hardware vendors and thus provides a good first target for ATLAS. There is a large audience for this API, and on those platforms where vendor-supplied BLAS exist, there is an easy way to determine if ATLAS can provide the required level of performance. Figure 1 shows a performance comparison chart of ATLAS, the vendor-supplied BLAS, and the Fortran77 reference BLAS across various architectures.

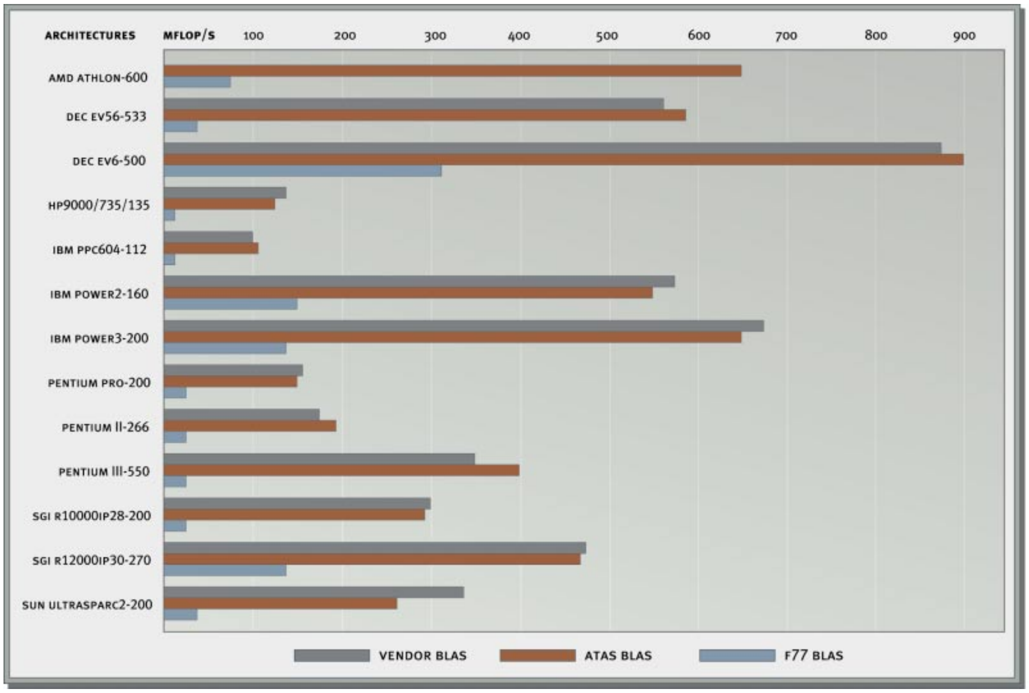


FIGURE 1
Chart Depicting the performance of 500 x 500 DGEMM across multiple architectures

ATLAS currently provides the following:

- Complete, optimized, and portable BLAS
- Optimized LU and Cholesky factorization and solve routines that are compatible with LAPACK
- Standard C and Fortran77 APIs for all routines
- Testers and timers for above
- Support for most ISA extensions including SSE, SSE2, 3DNow!, and AltiVec

Excepting the Fortran77 interface routines, ATLAS is written purely in strict ISO/ANSI C. Compiling the package requires access to a Unix-like command environment (e.g., Unix make, /bin/sh, etc.). ATLAS is optimized for hierarchical memory systems, and will perform best on machines with registers and at least one level of cache. ATLAS runs on any Unix OS possessing an ANSI/ISO C compiler as well as Windows 9x/NT/2000.

Anyone needing optimized BLAS/LAPACK benefits from ATLAS. There are a large number ATLAS users who call the BLAS directly from their applications. Several computer vendors, in creating architecture-specific versions of the BLAS, also use ATLAS.

In addition, ATLAS is used by several linear algebra problem-solving environments (MAPLE, MATLAB, Octave, and is being considered for use in Mathematica), by many projects (HPL, LAPACK, MatLisp, The R Project, Scilab, etc), and is also included in many Linux distributions (e.g., Debian, Scyld Beowulf and SuSE).

RECENT PUBLICATIONS

Whaley, C., Petitet, A., Dongarra, J. "Automated Empirical Optimization of Software and the ATLAS Project," *Parallel Computing*, May/June 2001, 22-29.

RELATED URLS

BLAS - http://www.netlib.org/blas/
HPL - http://www.netlib.org/hpl/
LAPACK - http://www.netlib.org/lapack/
Maple - http://www.maplesoft.com/
Mathematica - http://www.mathematica.com/
Matlab - http://www.mathworks.com/
MatLisp - http://matlisp.sourceforge.net/
Octave - http://www.che.wisc.edu/octave/
R Project - http://www.r-project.org/
Scilab - http://www.hammersmith-consulting.com/scilab-us-g.html

BLAST/LAPACK/ScaLAPACK

BLAS TECHNICAL FORUM LINEAR ALGEBRA PACKAGE SCALABLE LAPACK

DEVELOPMENT TEAM
ALPHABETICAL

Susan Blackford

Jack Dongarra

Clint Whaley

WEB SITE

http://icl.cs.utk.edu/blast/

E-MAIL

blast@cs.utk.edu

COLLABORATORS

Compaq

Cornell Theory Center

Cray

Florida Institute of Technology

Hewlett-Packard

Hitachi

IBM

Intel

Lucent Technologies

Mississippi State University

Myricom

National Institute of Standards and Technology

NEC

Numerical Algorithms Group,Ltd.

RAL/CERFACS

Rice University

SGI

Sun Microsystems - France

Texas Instruments

University of California, Berkeley

University of Houston

University of Illinois Urbana - Champaigne

University of Minnesota

University of Notre Dame

University of Tennessee

University of Texas at Austin

Visual Numerics, Inc.

BLAST

The BLAS Technical Forum was established to consider expanding the specification of a set of kernel routines for linear algebra (historically called the Basic Linear Algebra Subprograms and commonly known as the BLAS) in a number of directions in light of modern software, language, and hardware developments. The BLAS Technical Forum meetings were conducted in the spirit of the earlier BLAS meetings and the standardization efforts of the MPI and HPF forums, and began with a workshop in November 1995 at the University of Tennessee. Additional meetings were hosted by universities, government institutions, and software and hardware vendors. The final meeting was held in March 1999.

Various working groups were established at the meetings to consider issues such as

- the overall functionality,
- language interfaces,
- sparse BLAS,
- distributed-memory dense BLAS,
- extended and mixed precision BLAS,
- interval BLAS, and
- extensions to the existing BLAS.

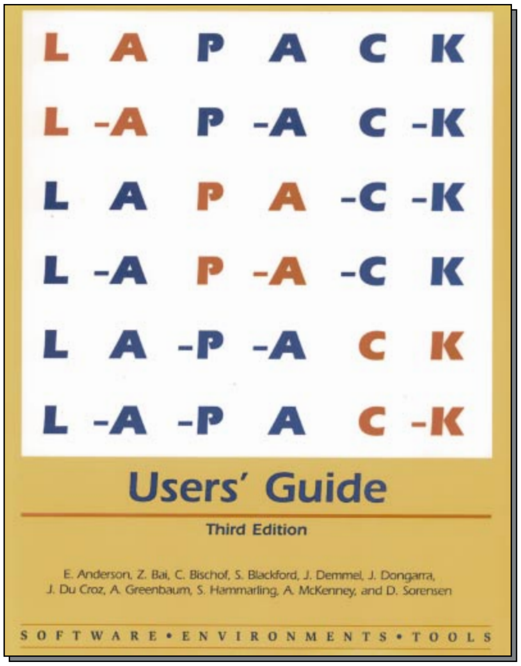
The rules of the forum were adopted from those used for the MPI and HPF forums. The minutes from all of the meetings are contained on the BLAST Forum website. Virtual meetings and voting on chapters were also conducted via the website. The efforts of these working groups are summarized in the BLAST document and are also available on the BLAST Forum website.

The BLAS Technical Forum standards document has been accepted for publication in the *International Journal for High Performance Computing Applications*, and will appear in the Spring and Summer issue of 2002.

A major aim of the standards defined in the document is to enable linear algebra libraries (both public domain and commercial) to interoperate efficiently, reliably, and easily. We believe that hardware and software vendors, higher level library writers, and application programmers all benefit from the efforts of this forum and are the intended end users of these standards.

The first major concerted effort to achieve agreement on the specification of a set of linear algebra kernels resulted in the Level 1 BLAS and associated test suite. The Level 1 BLAS are the specification and implementation in Fortran of subprograms for scalar and vector operations. This was the result of a collaborative project in 1973-77. Following the distribution of the initial version of the specifications to people active in the development of numerical linear algebra software, a series of open meetings were held at conferences, and as a result, extensive modifications were made in an effort to improve the design and make the subprograms more robust. The Level 1 BLAS were extensively and successfully exploited by LINPACK, a software package for the solution of dense and banded linear equation and linear least squares problems.

With the advent of vector machines, hierarchical memory machines, and shared memory parallel machines, specifications for the Level 2 and 3 BLAS (concerned with matrix-vector and matrix-matrix operations, respectively) were drawn up in 1984-86 and 1987-88. These specifications made it possible to construct new software to more effectively utilize the memory hierarchy of modern computers.


L A P A C K
Users' Guide
Third Edition
E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen
SOFTWARE • ENVIRONMENTS • TOOLS

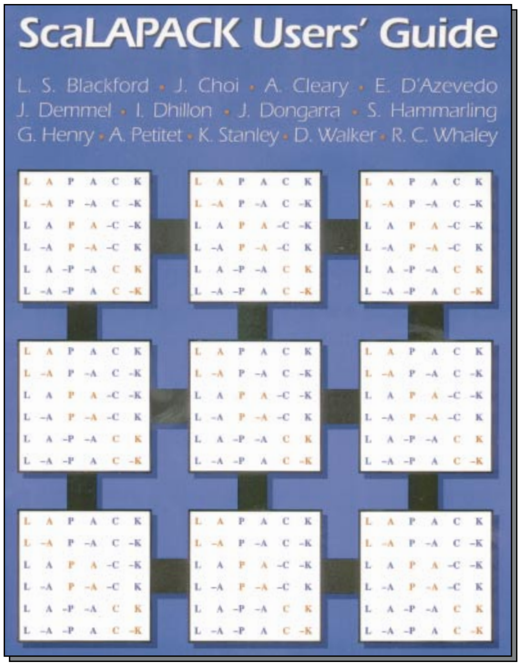

ScaLAPACK Users' Guide
L. S. Blackford • J. Choi • A. Cleary • E. D'Azevedo
J. Demmel • I. Dhillon • J. Dongarra • S. Hammarling
G. Henry • A. Petitet • K. Stanley • D. Walker • R. C. Whaley

FIGURE 1
Cover of LAPACK Users' Guide

FIGURE 2
Cover of ScaLAPACK Users' Guide

Images courtesy of Society for Industrial and Applied Mathematics (SIAM)

In particular, the Level 3 BLAS allowed the construction of software based upon block-partitioned algorithms, typified by the linear algebra software package LAPACK. LAPACK is state-of-the-art software for the solution of dense and banded linear equations, linear least squares, eigenvalue problems, and singular value problems. The software makes extensive use of all levels of BLAS and particularly utilizes the Level 2 and 3 BLAS for portable performance. LAPACK is widely used in application software and supported by a number of hardware and software vendors.

To a great extent, the user community embraced the BLAS, not only for performance reasons, but also because developing software around a core of common routines like the BLAS is good software engineering practice. Highly efficient, machine-specific implementations of the BLAS are available for most modern high-performance computers. The BLAS have enabled software to achieve high performance with portable code.

The original BLAS concentrated on dense and banded operations, but many applications require the solution of problems involving sparse matrices, and there have also been efforts to specify computational kernels for sparse vector and matrix operations.

The original efforts to specify sparse Level 2 and 3 BLAS took considerably longer than the corresponding efforts for the dense and banded BLAS, principally because of the need to obtain consensus on the way to represent sparse matrices. The lessons learned from those efforts have provided vital background to the specifications given in the BLAST document.

The original Level 2 BLAS included, as an appendix, the specification of extended precision subprograms. With the widespread adoption of hardware supporting the IEEE extended arithmetic format, other forms of extended precision arithmetic, and increased understanding of algorithms to successfully exploit such arithmetic, it was felt to be timely to include a complete specification for a set of extra precise BLAS.

The specification of the original BLAS was given in the form of Fortran66 and subsequently Fortran77 subprograms. In the BLAST document we provide specifications for Fortran95, Fortran77, and C. Reference implementations are also provided. Alternative language bindings for C++ and Java were also discussed during the meetings of the forum, but the specifications for these bindings were postponed for a future series of meetings.

RECENT PUBLICATIONS

"Basic Linear Algebra Subprograms Technical (BLAST) Forum Standard", to appear in *International Journal of High Performance Computing Applications*, Vol. 16 (May and November 2002).

RELATED URLS

BLAS - http://www.netlib.org/blas/

LAPACK - http://www.netlib.org/lapack/index.html

ScaLAPCK - http://icl.cs.utk.edu/scalapack/

BLAST/LAPACK/ScaLAPACK

CONTINUED

LAPACK

DEVELOPMENT TEAM
ALPHABETICAL

Susan Blackford

Jack Dongarra

WEB SITE

http://icl.cs.utk.edu/lapack/

E-MAIL

lapack@cs.utk.edu

COLLABORATORS

Myricom

Numerical Algorithms Group, Ltd.

University of California, Berkeley

University of California, Davis

Rice University

Sun Microsystems - France

AGENCY FUNDING

Department of Energy

National Science Foundation

LAPACK (Linear Algebra PACKage) is a library of Fortran77 subroutines for solving the most commonly occurring problems in numerical linear algebra. It has been designed to provide high efficiency on vector processors, high-performance "super-scalar'' workstations, and shared memory multiprocessors. It can also be used satisfactorily on all types of scalar machines (PC's, workstations, mainframes). A distributed-memory version of LAPACK, called ScaLAPACK, has been developed for other types of parallel architectures (for example, massively parallel SIMD machines, or distributed memory machines).

LAPACK can solve systems of linear equations, linear least squares problems, eigenvalue problems, and singular value problems. LAPACK can also handle many associated computations such as matrix factorizations or estimating condition numbers. Dense and band matrices are provided for, but not general sparse matrices. In all areas, similar functionality is provided for real and complex matrices.

LAPACK has been designed to supersede LINPACK and EISPACK, principally by restructuring the software to achieve much greater efficiency, where possible, on modern high-performance computers; also by adding extra functionality, by using some new or improved algorithms, and by integrating the two sets of algorithms into a unified package.

LAPACK routines are written so that as much as possible of the computation is performed by calls to the Basic Linear Algebra Subprograms (BLAS). Highly efficient, machine-specific implementations of the BLAS are available for many modern high-performance computers. The BLAS enable LAPACK routines to achieve high performance with portable code.

The current version of LAPACK (v 3.0, June 1999) is freely available on Netlib and can be obtained via the World Wide Web or anonymous ftp. FAQ and Errata files are also available. Alternative language interfaces to LAPACK (or translations/conversions of LAPACK) are available in Fortran95, C, and Java. The LAPACK Users' Guide provides an informal introduction to the design of the package, a detailed description of its contents, and a reference manual for the leading comments of the source code.

RECENT PUBLICATIONS

Barker, V., Blackford, S., Dongarra, S., Du Croz, J., Hammarling, S., Marinova, M., Wasniewski, J., Yalamov, P. *LAPACK95 Users' Guide* (Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM) Publications), 2001.

RELATED URLS

ATLAS - http://www.netlib.org/atlas/

BLAS - http://www.netlib.org/blas/faq.html

CLAPACK - http://www.netlib.org/clapack/

JavaLAPACK - http://www.netlib.org/java/f2j/

LAPACK 95 - http://www.netlib.org/lapack95/

ScaLAPCK - http://icl.cs.utk.edu/scalapack/

Users' Guide - http://www.netlib.org/lapack/lug/

FAQ

http://www.netlib.org/lapack/faq.html

ScaLAPACK

DEVELOPMENT TEAM
ALPHABETICAL

Susan Blackford

Jack Dongarra

Clint Whaley

WEB SITE

http://icl.cs.utk.edu/scalapack/

E-MAIL

scalapack@cs.utk.edu

COLLABORATORS

Intel

Myricom

Numerical Algorithms Group, Ltd.

Oak Ridge National Laboratory

Soongsil University - Seoul, South Korea

Sun Microsystems - France

University of California, Berkeley

AGENCY FUNDING

Department of Energy

National Science Foundation

ScaLAPACK is a library of high-performance linear algebra routines for distributed-memory message-passing MIMD computers and heterogeneous or homogeneous networks of PCs or workstations supporting MPI and/or PVM. The name ScaLAPACK is an acronym for Scalable Linear Algebra PACKage, or Scalable LAPACK. Like LAPACK, the ScaLAPACK routines are based on block-partitioned algorithms in order to minimize the frequency of data movement between different levels of the memory hierarchy. The fundamental building blocks of the ScaLAPACK library are distributed memory versions of the Level 1, 2 and 3 BLAS (called the Parallel BLAS or PBLAS) and a set of Basic Linear Algebra Communication Subprograms (BLACS) for communication tasks that arise frequently in parallel linear algebra computations. In the ScaLAPACK routines, the majority of interprocessor communication occurs within the PBLAS, so the source code of the top software layer of ScaLAPACK looks similar to that of LAPACK.

ScaLAPACK includes routines for the solving the following:

- linear systems of equations,
- general and symmetric positive definite band linear systems of equations, general and symmetric positive definite tridiagonal linear systems of equations,
- condition estimation and iterative refinement for LU and Cholesky factorization,
- matrix inversion,
- full-rank linear least squares problems,
- orthogonal and generalized orthogonal factorizations,
- orthogonal transformation routines,
- reductions to upper Hessenberg, bidiagonal and tridiagonal form,
- reduction of a symmetric-definite generalized eigenproblem to standard form,
- the symmetric/Hermitian eigenproblem,
- the generalized symmetric/Hermitian eigenproblem,
- the nonsymmetric eigenproblem,
- the singular value decomposition.

The current version of ScaLAPACK (v 1.7) was released in August of this year and is freely available on Netlib. It can be obtained via the World Wide Web or anonymous ftp. FAQ and Errata files are also available. The ScaLAPACK Users' Guide provides an informal introduction to the design of the package, a detailed description of its contents, and a reference manual for the leading comments of the source code. ScaLAPACK continues to expand into an even wider community effort.

RELATED URLS

DOE ASCI Red - http://www.sandia.gov/ASCI/Red/

IBM Parallel ESSL - http://www.rs6000.ibm.com/software/sp_products/esslpara.html

NAG Parallel Library - http://www.nag.co.uk/numeric/fd/FDdescription.asp

SGI Cray Scientific Software Library - http://www.sgi.com/software/scsl.html

VNI IMSL Numerical Library - http://www.vni.com/products/imsl/index.html

FAQ

http://www.netlib.org/scalapack/faq.html

FT-MPI

FAULT-TOLERANT MESSAGE PASSING INTERFACE

DEVELOPMENT TEAM
ALPHABETICAL

Antonin Bukovsky

Jack Dongarra

Graham Fagg

Jeremy Millar

Farial Shahnaz U

Sathish Vadhiyar G

U = UNDERGRADUATE STUDENT
G = GRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/ftmpi/

E-MAIL

ftmpi@cs.utk.edu

AGENCY FUNDING

Department of Energy

The initial version of the Message Passing Interface (MPI) standard was designed to work efficiently on Massively Parallel multi-Processors (MPPs), which had very little job control and thus a static process model. Later versions of the MPI standard incorporated some dynamic process control, but did not allow for node/process failures to be tolerated. As high performance computing (HPC) systems increase in size with higher potential levels of individual node failure, the need for new fault tolerant applications to be developed increases. Currently, fault tolerant applications cannot be built with MPI because MPI is unable to handle failures gracefully. FT-MPI is a developmental implementation of MPI that allows fault tolerant applications to be built. Under FT-MPI, applications control how failures are handled by either the message passing layer or the application itself.

The initial version of the MPI standard specified that, if a process failed, the communicator (of which the process belonged) became invalid and could no longer be used for communication. The only recovery method from such a failure was to abort the rest of the application regardless of how long it had been running. Check-pointing and regular saving of the application's state could alleviate these problems, and for some application classes, it still can.

Such semantics for failure are suitable for highly stable small to medium sized systems where failures are infrequent. Beyond these sizes, the mean time between failures (MTBF) of nodes becomes a factor. As attempts to build the next generation Petaflop systems advance, this situation is likely to become worse. Individual node reliability decreases with an increase in node numbers, which leads to higher possibilities of node failures.

Although FT-MPI can allow for automatic restart of applications in the case of failures by co-operating with checkpoint libraries, it is really meant to allow development of algorithm level, fault tolerant applications. In other words, some applications adapt to failures by changing their algorithms, such as applications that allow for reduced grids in numeric solvers. Building such applications using present implementations of MPI is not currently possible. FT-MPI supports the execution of current MPI applications without modification by providing a subset of the MPI-1 and MPI-2 application programming interfaces (APIs).

Current semantics of MPI indicate that a failure of an MPI process or communication causes all communicators associated with them to become invalid. Since the standard provides no method to reinstate the communicators (it is even unclear if they can be freed), MPI_COMM_WORLD itself becomes invalid, thus causing the entire MPI application to shut down.

FT-MPI extends the MPI communicator states from {valid, invalid} to a range {FT_OK, FT_DETECTED, FT_RECOVER, FT_RECOVERED, FT_FAILED}. In effect, this becomes {OK, PROBLEM, FAILED} with the other states mainly of interest to the internal fault recovery algorithm of FT_MPI. Processes also have typical states of {OK, FAILED}, which FT-MPI replaces with {OK, UNAVAILABLE, JOINING, FAILED}. The UNAVAILABLE state includes unknown, unreachable, or "we have not voted to remove it yet" substates. A communicator changes its state when either an MPI process changes its state or a communication within that communicator fails for some reason. The typical MPI semantics are from OK to FAILED, which then causes an application to abort. By allowing the communicator to be in an intermediate state, we allow the application the ability to decide how to alter the communicator and its state, as well as how communication within the intermediate state behaves.

When a failure occurs, the user application can tell the message-passing layer how to handle it or it can handle the failure itself. The application can specify failure modes that determine whether communicators are reformed or destroyed in the event of a failure, and whether communications can continue until the application reaches a state in which it can more fully address the failure.

Failure modes for communicators are as follows:

- SHRINK: The communicator is shrunk so that there are no holes in its data structures. The ranks of the processes are changed, forcing the application to recall MPI__COMM__RANK.
- BLANK: This is the same as SHRINK, except that the communicator can now contain gaps to be filled in later. Communicating with a gap will cause an invalid rank error. Note also that calling MPI__COMM__SIZE will return the size of the communicator and not the number of valid processes within it.
- REBUILD: It is the most complex because it forces the creation of new processes to fill any gaps. The new processes can either be placed into the empty ranks or the communicator can be shrunk and the processes appended to the end. This is used by applications that require a certain size to execute, i.e., power of two FFT solvers.
- ABORT: This is a mode where the application (on error detection) forces a graceful abort. The user can not trap this, and the only option is to change the communicator mode to one of the above modes. Communications within the communicator are controlled by a message mode for the communicator, which can be either of the following:
  - NOP: No operations allowed on error, i.e., no user level message operation is allowed and all simply return an error code. This is used to allow an application to return from any point in the code to a state where it can take appropriate action as soon as possible.
  - CONT: All communication that is *not* directed to the effected/failed node can continue as normal. Attempts to communicate with a failed node will return errors until the communicator state is reset.

Typical usage of FT-MPI would be in the form of an error check and then some corrective action such as a communicator rebuild. A typical code fragment is shown below, where in the event of an error the communicator is simply rebuilt and reused:

```
rc= MPI_Send (----, com);
If (rc==MPI_ERR_OTHER)
        MPI_Comm_dup (com, newcom);
        com = newcom;        /* continue.. */
```

Some types of computations such as SPMD master-slave codes only need the error checking in the master code if the user is willing to accept the master as the only point of failure.

The FT-MPI system was developed from scratch as a high performance MPI implementation that is designed to execute on top of the HARNESS MetaComputing environment. The system includes an advanced buffer management scheme for handling complex, user derived data types (DDTs), a self-tuning collective communications library, and a complex, multi-threaded message passing subsystem that supports TCP/UDP, Myricom GM, and SystemV shared memory concurrently.

FT-MPI allows message passing applications to be built that can survive node failures without the need to continuously make expensive state and checkpoint dumps to disk. It also allows for new classes of algorithms to be developed that can adapt to failures, which is currently not possible with other implementations of MPI.

FT-MPI has been used by the developers of the Parallel Spectral Transformation Shallow Water Model (PSTSWM) at Oak Ridge National Laboratory (ORNL) to implement a new version of this complex parallel code that automatically repairs itself during failures without external intervention. Developers of the widely used Parallel Climate Community Model (PCCM) are considering modifying their code to use FT-MPI to implement a reduced grid algorithm for handling failures.

RECENT PUBLICATIONS

Fagg, G., Bukovsky, A., Dongarra, J. "Fault Tolerant MPI for the HARNESS Meta-Computing System," Alexandrov, V., Dongarra, J., Juliano, B., Renner, R., Tan, K. (Eds.) In *Proceedings of International Conference of Computational Science - ICCS 2001*, San Francisco, CA. *Lecture Notes in Computer Science*, Vol. 2073 (Berlin: Springer Verlag), 2001: 355-366.

RELATED URLS

HARNESS - http://www.epm.ornl.gov/harness/

Parallel Climate Community Model (PCCM) -
http://www.epm.ornl.gov/chammp/pccm2.1/

Parallel Spectral Transformation Shallow Water Model (PSTSWM) -
http://www.epm.ornl.gov/chammp/pstswm/

GrADS

GRID APPLICATION DEVELOPMENT SYSTEM

DEVELOPMENT TEAM
ALPHABETICAL

Susan Blackford

Jack Dongarra

Brett Ellis

Ken Roche G

Sathish Vadhiyar G

Asim YarKhan

G = GRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/grads/

E-MAIL

grads@cs.utk.edu

COLLABORATORS

Information Sciences Institute

Rice University

University of California, San Diego

University of Chicago

University of Houston

University of Illinois at Urbana-Champaign

University of Indiana

University of Southern California

AGENCY FUNDING

National Science Foundation

Advances in networking technologies will soon make it possible to use the global information infrastructure in a qualitatively different way—as a computational resource as well as an information resource. This idea for an integrated computation and information resource called the Computational Power Grid has been described in the recent book entitled *The Grid: Blueprint for a New Computing Infrastructure*. The Grid will connect the nation's computers, databases, instruments, and people in a seamless web of computing and distributed intelligence that can be used as a problem-solving resource in many fields of human endeavor, particularly for science and engineering. To realize this vision, we must overcome significant scientific and technical obstacles. Principal among these is usability. Because the Grid will be inherently more complex than existing compute systems, programs that execute on the Grid will reflect some of this complexity. Hence, making Grid resources useful and accessible to scientists and engineers will require new software tools that embody major advances in both the theory and practice of building Grid applications.

The goal of the Grid Application Development System (GrADS) is to simplify distributed heterogeneous computing in the same way that the World Wide Web simplified information sharing over the Internet. GrADS will exploit the scientific and technical problems that must be solved to make it relatively easy to develop Grid applications for real problems and to tune their performance. This will require research in the following five key areas, each validated in a prototype infrastructure:

- Grid software architectures that facilitate information flow and resource negotiation among applications, libraries, compilers, linkers, and runtime systems
- Base software technologies, such as scheduling, resource discovery, and communication, to support development and execution of performance-efficient Grid applications
- Languages, compilers, environments, and tools to support the creation of applications and problem-solving environments for the Grid
- Mathematical and data structure libraries for Grid applications, including numerical methods for control of accuracy and latency tolerance
- Innovative new science and engineering applications that can take advantage of these new technologies to run effectively in a Grid environment

A team of thirteen principal investigators from the following universities is conducting the research:

- Rice University,
- University of California San Barbara,
- University of California San Diego,
- University of Chicago,
- University of Houston,
- University of Illinois at Urbana-Champaign,
- University of Indiana,
- University of Southern California, and
- University of Tennessee.

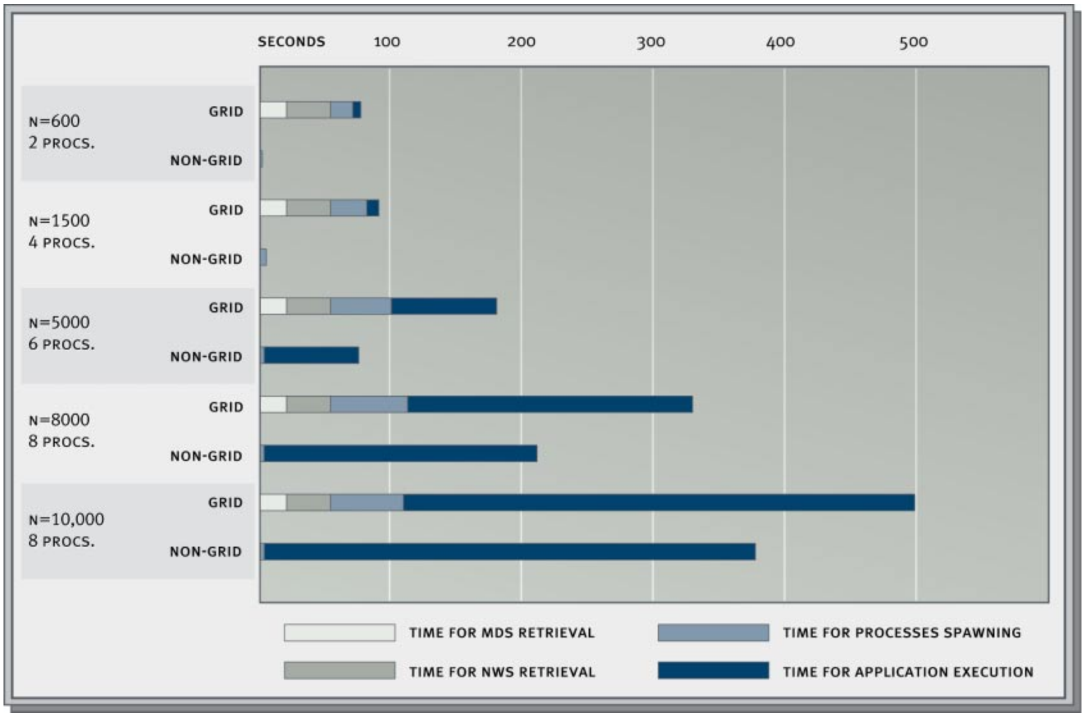


FIGURE 1
Grid ScaLAPACK vs. Non-Grid ScaLAPACK, dedicated TORC machines

Technology transfer in GrADS will be via two principal mechanisms. First, we are working closely with a group of industrial collaborators to encourage the adoption and standardization of system software technologies that arise from the research. Second, we are working directly with application developers through the two NSF PACIs (NPACI and the Alliance) and through the NASA IPG and ASCI ASAP programs. GrADS is fostering research, education, and technology transfer programs that are contributing to evolutionary new ways of utilizing the global information infrastructure as a platform for computation, changing the way scientists and engineers solve their everyday problems.

In addition, we are developing a prototype system designed specifically for the use of numerical libraries in the grid setting and experimenting with routines from the ScaLAPACK library on the Grid.

RECENT PUBLICATIONS

Berman, F., Chien, A., Cooper, K., Dongarra, J., Foster, I., Gannon, D., Johnsson, L., Kennedy, K., Kesselman, C., Mellor-Crummey, J., Reed, D., Torczon, L., Wolski, R. “The GrADS Project: Software Support for High-Level Grid Application Development,” *International Journal of High Performance Applications and Supercomputing*, Vol. 15, Number 4 (Winter 2001): 327-344.

Petitet, A., Blackford, S., Dongarra, J., Ellis, B., Fagg, G., Roche, K., Vadhiyar, S. “Numerical Libraries and The Grid: The Grads Experiments with ScaLAPACK, “ *Journal of High Performance Applications and Supercomputing*, Vol. 15, Number 4 (Winter 2001): 359-374.

RELATED URLS

GrADS - http://www.hipersoft.rice.edu/projects/grads.html

NSF Alliance - http://www.ncsa.edu/

NSF NPACI - http://www.npaci.edu/

HARNESS

HETEROGENOUS ADAPTABLE RECONFIGURABLE NETWORKED SYSTEMS

DEVELOPMENT TEAM
ALPHABETICAL

Antonin Bukovsky

Jack Dongarra

Graham Fagg

Jeremy Millar

Keith Moore

WEB SITE

http://icl.cs.utk.edu/harness/

E-MAIL

harness@cs.utk.edu

COLLABORATORS

Emory University

Oak Ridge National Laboratory

AGENCY FUNDING

Department of Energy

HARNESS (Heterogeneous Adaptable Reconfigurable Networked SystemS) is an experimental metacomputing framework built around the services of a highly customizable and reconfigurable distributed virtual machine (DVM). A DVM is a tightly coupled computation and resource grid that provides a flexible environment to manage and coordinate parallel application execution.

The system is designed to support a wide range of DVM sizes, from users building personal DVMs to enterprise and widely distributed DVMs. Collaboration and resource sharing between different entities is performed by the temporary merging and splitting of different DVMs.

HARNESS seeks to remove the limitations discovered in the Parallel Virtual Machine (PVM) system and create a completely different approach to building, modifying, and using multiple DVMs to facilitate a new generation of collaborative and computation environments.

Virtual machine (VM) terminology, borrowed from PVM, refers to a system where the computing resources on that system can be viewed as a single, large, distributed memory computing resource. This abstraction was once very powerful and widely accepted, but the PVM system itself only allowed for a single VM, which limited collaboration, and the code was monolithic, which made it difficult to add new functionality. PVM's single VM also had a single point of failure through which it maintained a master database that inevitably limited its scalability.

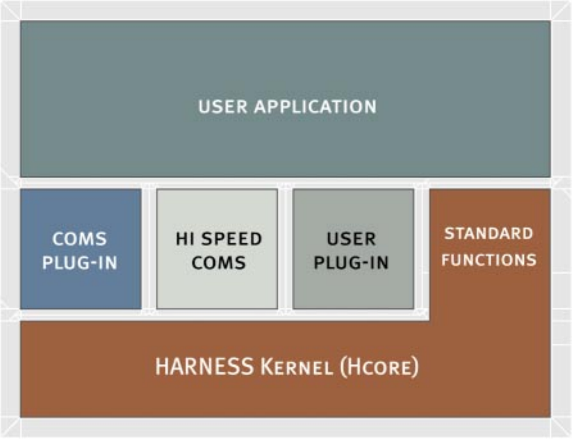
HARNESS was designed to support

- multiple DVMs
- no single point of failure
- scalable infrastructure though replicated state
- flexible/reconfigurable functionality (by allowing new components or plug-ins to be added and removed dynamically) and
- Legacy message passing code

The architecture is built on kernels, daemons, DVMs, and services provided by standard components. The kernel is implemented as a set of core functions for loading and running components either locally or via remote requests. A HARNESS daemon is composed of a kernel (the HARNESS Core or Hcore) and a minimal set of required components to provide basic services. These services include maintaining state, the ability to communicate between components, and remote invocation of components and new daemons. A HARNESS DVM is composed of a set of co-operating daemons that together present the basic services of communication, process control, resource management, and fault detection. Important goals of HARNESS are that it should be robust and reliable. All information and control within HARNESS are to be built on a Symmetric Peer-to-Peer Distributed Control (SPDC) algorithm; thus, HARNESS will not have a single point of failure, but instead a user configurable level of fault tolerance.

Figures 1 and 2 show the design of a daemon built on a kernel (Hcore), then how the daemons interconnect to form a DVM. A HARNESS DVM is made from multiple HARNESS daemons executing on multiple hosts.

Emory University has produced a Java-based implementation using JNI/RMI and dynamic loading. Both ICL and Oak Ridge National Laboratory (ORNL) have produced C-based HARNESS cores



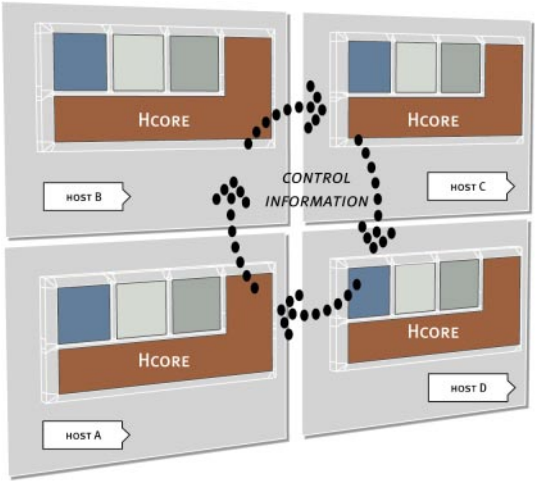


FIGURE 1
Illustration of a daemon built on a kernel

FIGURE 2
Illustration of daemons interconnecting to form a DVM

that support dynamic libraries and shared objects as component plug-ins. Support for inter-operation between the Java and C-based systems has also been developed.

Two plug-ins have been developed to allow standard message passing codes to be used directly, and they support both the PVM 3.X and a subset of MPI-2 APIs. These plug-ins allow users of existing applications to run on HARNESS without any code modification. The MPI plug-in, known as FT-MPI, provides additional functionality to support fault-tolerant applications.

Using the PVM and MPI plug-ins allows for thousands of existing message passing applications to execute under the HARNESS system. The HARNESS system itself is expected to be used as a natural upgrade path for existing PVM users as well as users wishing to build their own personal computational grids without having to buy into some global grid framework.

The HARNESS system provides a number of benefits over existing VM-based systems, such as reliability, scalability, ease of adding new functionality, and the sharing of resources by merging DVMs.

RECENT PUBLICATIONS

Fagg, G., Bukovsky, A., Dongarra, J. “HARNESS and Fault Tolerant MPI,” *Parallel Computing*, Vol. 27, Number 11 (October 2001): 1479-1496.

RELATED URLS

FT-MPI - http://icl.cs.utk.edu/ftmpi/
PVM - http://www.epm.ornl.gov/pvm/

I2-DSI

INTERNET2 DISTRIBUTED STORAGE INFRASTRUCTURE

DEVELOPMENT TEAM
ALPHABETICAL

Micah Beck
Ying Ding G
Leon Dong G
Hunter Hagewood G
Jeremy Millar
Terry Moore

G = GRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/dsi/

E-MAIL

dsi@cs.utk.edu

COLLABORATORS

EROS Data Center
IBM
Lokomo Systems
North Carolina Supercomputing Center
University of Hawaii at Manoa
University of Indiana
University of North Carolina at Chapel Hill
Texas A & M University
Virginia Tech University

AGENCY FUNDING

National Science Foundation

INDUSTRY SUPPORT

Cisco Systems, Inc.
Ericcson
IBM
Internet2
Novell
Starburst Multicast

The Internet2 Distributed Storage Infrastructure (I2-DSI) project is a research effort designed to explore innovative uses of network storage in next generation applications and wide area information systems. The project began in the fall of 1998 with the cooperation of the University Corporation for Advanced Internet Development (UCAID), the support of industry partners and the participation of the Internet2 community. The ultimate goal of I2-DSI is to develop a next generation service platform that can help both the research and education communities fulfill their traditional missions during the next decade. To this end, I2-DSI aims to develop a reliable, scalable, high performance storage infrastructure for use by advanced applications to exploit the power of the Internet2 campus and backbone networks and overcome limitations inherent in the current World Wide Web architecture.

The underlying idea of I2-DSI's storage-based approach is to combine *intelligent replication* of content and services with the *transparent resolution* of client requests to the nearest virtual host, I2-DSI can give the I2 community easy, high-performance access to types of content and modes of service delivery that are now available only in proof of concept form. In some important respects, I2-DSI's approach is similar to (but predates) recent commercial efforts from companies like Akamai and Digital Island, which have capitalized on the idea of using advanced forms of traditional web caching to reduce bandwidth consumption, distribute server load, and improve performance in the distribution of typical Web content. But I2-DSI seeks to generalize content distribution beyond the current cache-based model, using replication in a way that allows objects like databases and executable content to be staged on and served from a variety of different platforms.

With sponsorship from Internet2 corporate partners and the cooperation of several universities in the Internet2 community, I2-DSI has established a national testbed to experiment with this new approach. Building on a major donation of large storage servers from IBM, along with contributions from companies such as Novell, Cisco Systems, Sun Microsystems, this testbed now has servers in six different locations in the USA. Current locations include the following:

- Univ. of Tennessee (Knoxville, TN),
- Univ. of North Carolina (Chapel Hill, NC),
- Univ. of Indiana (Indianapolis, IN),
- Texas A & M (Houston, TX),
- Univ. of Hawaii (Honolulu, HI),
- USGS EROS Data Center (Sioux Falls, SD).

Digital content and services are distributed on I2-DSI in the form of channels. Channels are I2-DSI's units of replication. A channel in this sense is a collection of content and associated services that can be transparently delivered to end user communities at a chosen cost/performance point through a flexible, policy-based application of resources. What such a channel can contain is a superset of what can be accessed via Web protocols. This includes text, graphics, Java applets, and multimedia, as well as services that utilize non-Web protocols. Experimental channels already deployed on the current testbed include

- Open Video: An expanding collection of public domain video content for the information retrieval and digital library research communities - http://openvideo.dsi.internet2.edu
- Docsouth: *Documenting the American South* is a collection of sources on Southern history, literature and culture from the colonial period through the first decades of the 20th century - http://docsouth.dsi.internet2.edu

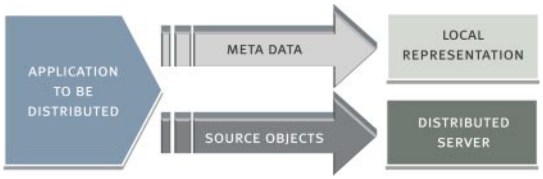


FIGURE 1
The PCR Distribution Process - Meta data and source objects

- MetaLab Linux: The comprehensive Linux Archives maintained by the UNC MetaLab (formerly the original SunSITE) - http://linux.dsi.internet2.edu
- Mandrake Linux: Mandrake Linux repository from Virginia Tech - http://linux-mandrake.dsi.internet2.edu

To make it possible to replicate such channels in a scalable way, the project has had to address the technical challenge of automating the replication process even when the underlying hardware/software platforms on the different system nodes vary. The current lack of standards for Web servers makes such push-button replication of Web sites nearly impossible. In response to this problem, we have developed a data model for channels, called the Portable Channel Representation (PCR), which explicitly encodes all the technical metadata necessary to automate the process of installing and configuring a given channel on a given type of hardware/software platform. There are two key elements:

- *Server-Independent Specification of Behavior*: A PCR description is an encoding of metadata in the eXtensible Markup Language (XML) that specifies the behavior of the server in response to a set of requests, collectively known as a "channel." In order to avoid dependence on configuration files specific to particular server software, PCR specifies in a platform-independent manner the source object and the method of interpretation that should be invoked on any particular request.
- *File-system-Independent Specification of Source Objects*: References to source objects from within a PCR channel description do not name them through their installed location in a file system directory structure, but instead use local names defined by a "file store".

When the channel is replicated to other locations, both the PCR metadata and the source objects in the file store, which are the content of the channel, are replicated (fig. 1). The PCR description and the file store together define a complete channel description that can be correctly implemented on any platform that correctly interprets both elements.

PCR will be promulgated as an open standard for creating replicable content channels, and I2-DSI is collaborating with private industry to develop tools that implement it and make it easy to use. A start-up company, Lokomo Systems, has already been formed to develop software for creating advanced content distribution networks based on PCR. The design and development of PCR is just one example of the kind of collaboration among academic organizations and between academia and industry that have marked I2-DSI from its inception.

The project has received government support as well. In the summer of 2000, the National Science Foundation's Advanced Network Infrastructure program added its support to this collective effort, awarding a three-year grant for almost one million dollars to the I2-DSI team.

This fall the I2-DSI project and its members will be leaving ICL to join the IBP project in a new CS research laboratory formed around the *logistical networking* research of Dr. Micah Beck, who has led I2-DSI from its inception, and Dr. Jim Plank, a co-leader of the IBP project. In a first effort to exploit the common elements of the two projects, this new group will deploy IBP depots on the I2-DSI infrastructure to create an experimental, wide-area test bed called the *Logistical Backbone* (*L-Bone*). With support from the Center for Information Technology Research (CITR), headed by Dr. Dongarra, this new laboratory will pursue new lines of research in the area of Logistical Computing and Internetworking, while continuing or expanding its fruitful collaborations with other ICL projects.

RECENT PUBLICATIONS

Abrahamsson, L., Achouiantz, C., Beck, M., Johansson, P., Moore, T. "Enabling Full Service Surrogates Using the Portable Channel Representation," *Proceedings of the 10th Intl. World Wide Web Conference*, Hong Kong, May 1 - 5, 2001.

RELATED URLS

Akamai - http://www.akamai.com/
Digital Island - http://www.digitalisland.com/
Lokomo Systems - http://www.lokomo.com/index.shtml
UCAID - http://www.ucaid.edu/ucaid/

IBP
INTERNET BACKPLANE PROTOCOL

DEVELOPMENT TEAM
ALPHABETICAL

Alex Bassi
Micah Beck
Xiang Li G
Terry Moore
Susan Wo G
Yong Zheng G

G = GRADUATE STUDENT

WEB SITE
http://icl.cs.utk.edu/ibp/

E-MAIL
ibp@cs.utk.edu

COLLABORATORS
UNIVERSITY OF TENNESSEE

Scott Atchly G
Jim Plank
Stephen Soltez U
Martin Swany G
Rich Wolski

U = UNDERGRADUATE STUDENT
G = GRADUATE STUDENT

AGENCY FUNDING
Department of Energy
National Science Foundation

The Internet Backplane Protocol (IBP) is middleware for managing and using remote storage. It was developed to support an approach to building large scale, distributed applications called logistical networking. Logistical networking is the global scheduling and optimization of data movement, storage, and computation, using a model that takes into account all the network's underlying physical resources. It contrasts with more traditional approaches to networking, which do not explicitly model storage or computation as shared network resources. We call this approach "logistical" because of the analogy it bears with the systems of warehouses, depots, and distribution channels commonly used in the logistics of military and industrial activities. IBP provides a mechanism for using distributed storage for logistical purposes.

IBP represents the kind of middleware needed to overcome the current balkanization of storage management capabilities on the Internet. By providing a uniform, application-independent interface to storage in the network, IBP makes it possible for applications of all kinds to use logistical networking to exploit data locality and more effectively manage buffer resources. This allows any application that needs to manage distributed state to benefit from the kind of standardization, interoperability, and scalability that have made the Internet such a powerful communication tool.

By itself, IBP is a very primitive enhancement to the functionality of the network, playing a role for data storage analogous to the role played by Internet Protocol (IP) for data transmission. As a primitive layer of storage functionality, IBP adds just enough additional abstraction to the underlying storage resource (disk, tape, etc.) to allow it to be utilized at the next higher level, but it does not add logic beyond what is necessary for the most common and indispensable storage functionality. For example, IBP provides no built in support for fault tolerance or location-independent naming, other than what is already provided by the underlying server platform on which it is implemented. In order to support such useful features, it is necessary to layer additional protocols and data structures on top of IBP, just as the Transmission Control Protocol (TCP) is layered on top of IP to provide for error correction and ordered delivery of packets.

The most intuitive and universal of these is the file abstraction, and filed generally have strong properties (e.g., unbounded size and duration) that are not modeled by native IBP. In order to apply the principle of aggregation to IBP's exposed storage services, it is necessary to maintain state that represents an aggregation of storage allocations. In the traditional Unix file system, a file is implemented as a tree of disk blocks with data blocks as the leaves, and the intermediate nodes of this tree, called the inodes, are data structures used to implement the aggregation of the underlying disk blocks on a single disk volume. Working by analogy with the inode, the IBP group has implemented a single generalized data structure, called an exNode (external node) to represent information known about the Internet storage resources that have been aggregated to create a single file (see Figure 1).

The exNode is basically a set of declarations and assertions that together describe the state of (i.e., the information known about the storage resources implementing) a single file. ExNode libraries will support generic requirements for files such as large size (through fragmentation), fast access (through caching), and reliability (through replication). Since it is intended for use in a number of different applications that are interoperable between heterogeneous nodes on the Internet, an exNode is expressed concretely as an encoding of storage resources (URLs or IBP capabilities) and associated metadata in XML. The use of the exNode by varying applications will provide interoperability similar to being attached to the same network file system. It is important to note, however, that the flexibility of a file implemented by the exNode is a function of the flexibility of the underlying storage resources. IBP's value consists of the fact that it overwhelmingly creates the most flexible and most easily deployable network storage resources.

Applications for IBP and the exNode include the following:

- When the exNode is populated exclusively with network-accessible storage resources, its portable XML encoding allows it to become highly mobile. IBP-Mail, for example, is a simple application that uses IBP depots to store and move mail attachments that are much larger than can normally be sent as an SMTP payload using MIME encoding. IBP-Mail builds an exNode to represent the attached file and then sends the XML serialization of that file in the SMTP payload. The recipient then rebuilds the exNode and uses it to access the content of the stored attachment.
- A simple distributed file system can be built by storing serialized exNodes in the host file system and using them like Unix soft links. Programs that would normally access a file instead find the exNode serialization, build an exNode data structure, and use it to access the file.
- A mobile agent that uses IBP depots to store part of its state can carry that state between hosts in the form of a serialized exNode. If the hosts understand the exNode serialization, then they can perform file system tasks for the agent while it is resident, returning the updated exNode to the agent when it migrates.

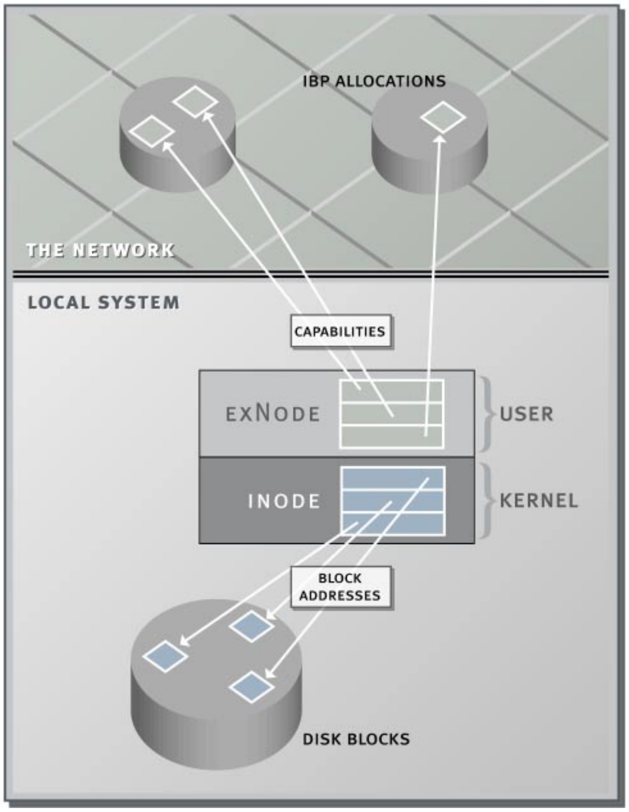


FIGURE 1
The exNode (external node): a generalized data structure representing the Internet storage resources used to implement a single file. It is the network equivalent of the Unix inode.

From its inception, the IBP project has been co-lead by Micah Beck, who worked as research faculty within ICL until this fall, and Jim Plank of the Computer Science Department. To promulgate the use of logistical networking by the research community, their team is now using the Internet2 Distributed Storage Infrastructure (I2-DSI) to deploy IBP depots in an experimental, wide-area test bed called the Logistical Backbone (L-Bone). The L-Bone consists of a set of IBP servers and some basic associated resources and protocols, such as directory service and network proximity measurement, which will offer higher-level services (e.g., resource discovery) to the research community. Having achieved critical mass with this substantial collection of overlapping projects, Drs. Beck and Plank, with the support of the Center for Information Technology Research (CITR), are now forming a separate research group as a peer of ICL that focuses on logistical computing and internetworking.

RECENT PUBLICATIONS

Plank, J., Bassi, A., Beck, M., Moore, T., Swany, M., and Wolski, R. "The Internet Backplane Protocol: Storage in the Network," *Internet Computing*, Vol. 5, Number 5, 2001.

Beck, M., Moore, T., Plank, J. "Exposed vs. Encapsulated Approaches to Grid Service Architecture," *2nd International Workshop on Grid Computing*, Denver, CO, Nov. 12, 2001.

RELATED URLS

I2-DSI - http://icl.cs.utk.edu/projects/I2DSI/
Lbone - http:www.cs.utk.edu/lbone/
NetSolve - http://icl.cs.utk.edu/projects/netsolve/
SInRG - http://icl.cs.utk.edu/projects/SInRG/

JLAPACK/f2j

JAVA LAPACK FORTRAN TO JAVA

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra

Andrew Downey G

Keith Seymour

G = GRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/f2j/

E-MAIL

f2j@cs.utk.edu

The goal of the JLAPACK project is to provide Application Programming Interfaces (APIs) to numerical libraries from Java programs. A Fortran-to-Java translator, f2j, will distribute the numerical libraries as class files produced. The f2j translator is a formal compiler that translates programs written using a subset of Fortran77 into a form that may be executed on Java virtual machines. The first priority is to translate the BLAS and LAPACK numerical libraries from their Fortran77 reference source code to Java class files. The subset of Fortran77 translated by f2j roughly matches the Fortran source used by BLAS and LAPACK. These libraries are established, reliable, and widely used linear algebra packages and are therefore a reasonable first test bed for f2j. Many other libraries of interest are expected to use a very similar subset of Fortran77.

The current release of JLAPACK (version 0.5) consists of 346 double-precision routines translated from LAPACK and 33 double-precision routines from BLAS, totaling 137,406 lines of Java source. Translated versions of the BLAS and LAPACK testers -- totaling over 100,000 lines of Java source code -- are also available for download.

There are two primary differences in the latest release of JLAPACK compared to the last public release. First, the current release is based on version 3.0 of LAPACK, whereas prior versions of JLAPACK were based on LAPACK 2.0. Second, the current release was generated using a new code generator that translates the original source directly from Fortran77 to JVM class files.

Since the nature of numerical computing demands fast execution, we have also begun to implement an optimization phase in the f2j compiler to produce faster bytecode. Given the goal of producing a Java implementation of JLAPACK, there are three options:

- Wrap the native routines in Java interfaces
- Rewrite the routines in Java from scratch
- Develop a tool to automate the translation

We avoided the first method because we wanted the Java version of LAPACK to be used by applets as well as applications, thus requiring a pure Java implementation. The second option would have required hand translating, testing, and debugging hundreds of routines. Given the large amount of code in LAPACK, the second option could be very time-consuming and error-prone. We chose the third option because it allows us to generate pure Java code in a consistent and reliable way from the original Fortran source. In addition, after pursuing the third option, we have a tool that could be applied successfully to other numerical libraries and eventually to a wide range of Fortran code.

The f2j compiler operates in four stages:

LEXING/PARSING

In this stage the lexer separates the Fortran source code into tokens and the parser builds a complete AST and symbol tables for each program unit. Subsequent compilation stages obtain all information about the program structure from the AST built during parsing.

OPTIMIZING THE USE OF SCALAR WRAPPERS

In Fortran, values are passed to functions and subroutines by reference. This implies that if a Fortran subroutine modifies one of its parameters, then that modification also takes effect in the calling routine. However, Java uses pass-by-value, which implies that modifications would not take effect in the caller. In order to simulate pass-by-reference in Java, we must wrap the scalar in an object. Then instead of passing the integer value, we would pass the object wrapper whose scalar field may be modified in the subroutine.

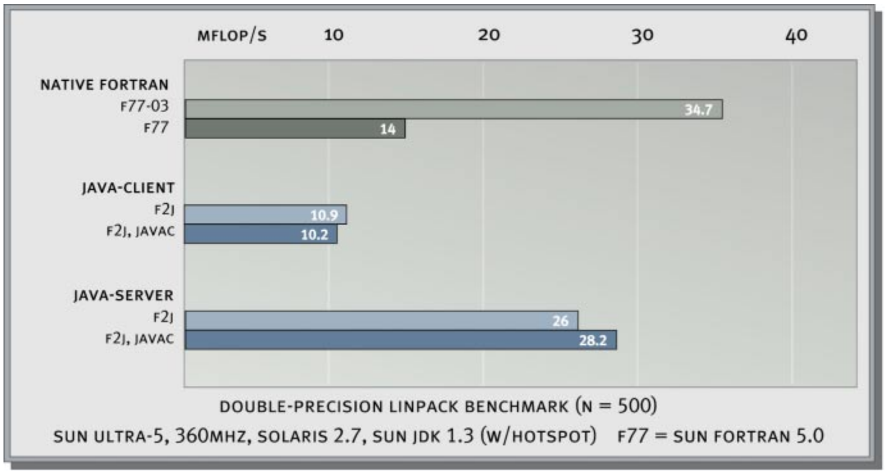


FIGURE 1
Performance results comparing f2j with native Fortran

The scalar "optimization'' phase determines which parameters of each subroutine absolutely need to be wrapped. The rest are passed as Java primitive data types (int, double, etc) in order to improve access times and save memory. The determination is made as follows:

A variable must be wrapped if:

- The variable is an argument to this function and it is on the left side of an assignment statement in this program unit
- The variable is an argument to this function and it is an argument to a READ statement
- The variable is passed to a function/subroutine that modifies it

The third rule implies that every function/subroutine that the current program unit depends on must be checked before this unit can be completely checked. F2j resolves the dependencies before continuing to check the current unit.

TYPE ASSIGNMENT

This stage is not "type checking'' in the semantic analysis sense. In this stage, f2j performs a traversal of the AST and assigns type information to each node, propagating information up the tree. For example, f2j looks at both sides of an addition operation and assigns the widest type to the addition node and so on up the tree. This information helps the code generator emit the appropriate type-specific opcodes and type casts when necessary.

CODE GENERATION

Code generation is by far the largest and most complicated stage in the translator. In this stage, f2j traverses the AST, generating code as it walks down the tree. The code generator depends on the information determined in all the prior steps to generate correct code. Currently, Java source code and JVM bytecode are generated during the same pass because using separate passes would have resulted in substantial duplicated code and would have made maintenance more difficult.

Each Fortran program unit is generated as a separate Java class containing a single static method. For example, the Fortran subroutine DGEMM would be translated to a Java class named *Dgemm* containing only a single method named *dgemm*. All arrays are laid-out in column-major fashion, with 2D Fortran arrays being translated as linearized 1D Java arrays.

Fortran GOTO statements are easily translated to JVM bytecode since there exists a *goto* opcode. However, Java source code does not provide a GOTO statement, thus we must perform some post-processing on the class files that were generated from Java source in order to correctly generate the GOTO statements.

JLAPACK benefits Java application developers who require a wide range of linear algebra routines in a pure Java implementation. The availability of a Java implementation of LAPACK facilitates the creation of web-based numerical and engineering applications.

RECENT PUBLICATIONS

Seymour, S., Dongarra, J. "Automatic Translation of Fortran to JVM Bytecode," *Proceedings of the ACM 2001 Java Grande/ISCOPE Conference*, 126-133, June 2001.

RELATED URLS

Bison - http://www.gnu.org/software/bison/bison.html
BLAS - http://www.netlib.org/blas/
LAPACK - http://www.netlib.org/lapack/

LINPACK Benchmark/HPL

HIGH PERFORMANCE LINPACK BENCHMARK

DEVELOPMENT TEAM
Jack Dongarra

E-MAIL
linpack-benchmark@cs.utk.edu

WEB SITE
http://icl.cs.utk.edu/linpack-benchmark/

COLLABORATORS
Antoine Petitet
SUN MICROSYSTEMS FRANCE

AGENCY FUNDING
Department of Energy FOR HPL

The LINPACK Benchmark is in some sense an accident. It was originally designed to assist users of the LINPACK package by providing information on the execution times required to solve a system of linear equations. LINPACK is a collection of Fortran routines designed to solve various systems of linear equations. The package itself is based on another package called the Basic Linear Algebra Subprograms, today referred to as the Level 1 BLAS. The first “LINPACK Benchmark report” appeared as an appendix in the LINPACK Users’ Guide in 1979 (See figure 1).

The appendix comprised data for one commonly used path in LINPACK for a matrix problem of size 100 on a collection of widely used computers (23 in all), so users could estimate the time required to solve their matrix problems. Over the years, additional data was added, more as a hobby than any thing else, and today the collection includes thousands of different computer systems all measured using the same piece of software. In addition to adding more computer systems’ performance to the list, the scope of the benchmark has also changed. The benchmark today reports four basic sets of performance numbers; the execution rate for solving a system of equations of order 100 using the Fortran LINPACK Software, the execution rate for solving a system of equations of order 1000 using the best method (which does not have to be Fortran), the asymptotic execution rate for solving a system of equations (High Performance LINPACK (HPL)), and the theoretical execution rate for the computer system.

In order to have an entry included in the LINPACK Benchmark report, the results must be computed using full precision. By full precision, we generally mean 64-bit floating point arithmetic or higher. Note that this is not an issue of single or double precision since some systems have 64-bit floating point arithmetic as single precision. It is a function of the arithmetic used. More precisely, the solution to all three benchmarks must satisfy the following mathematical formula:

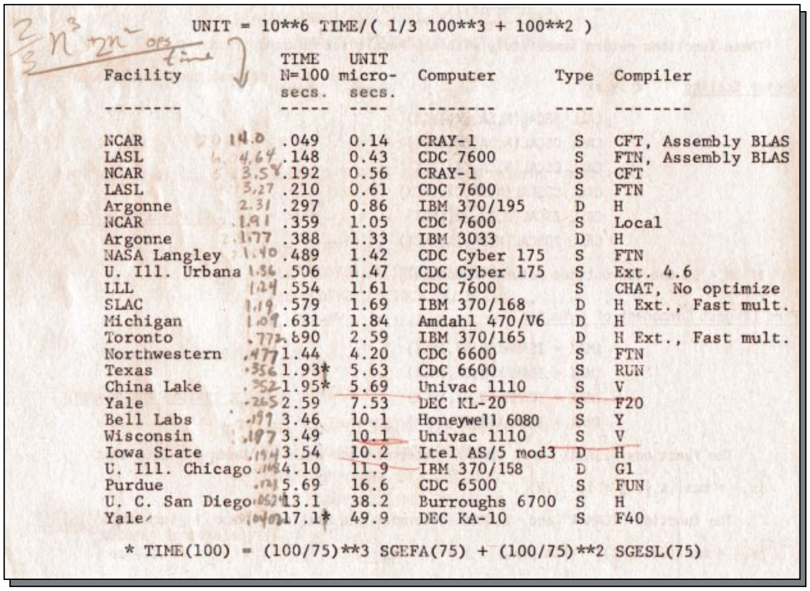
$$\frac{\| Ax-b \|}{\| A \| \| x \| n\varepsilon} \leq 0(1)$$

where ε = the machine precision (On IEEE machines this is 2^{-53}) and n is the size of the problem.

HPL is a software package that solves a (random) dense linear system in double precision (64 bits) arithmetic on distributed-memory computers. It can thus be regarded as a portable, as well as freely available, implementation of the HPL Benchmark.

The algorithm used by HPL can be summarized by the following keywords: Two dimensional block-cyclic data distribution; Right-looking variant of the LU factorization with row partial pivoting featuring multiple look-ahead depths; Recursive panel factorization with pivot search and column broadcast combined; Various virtual panel broadcast topologies; bandwidth reducing swap-broadcast algorithm; backward substitution with look-ahead of depth 1.

The HPL package provides a testing and timing program to quantify the accuracy of the obtained solution as well as the time it took to compute it. The best performance achievable by this software on a system depends on a large variety of factors. Nonetheless, with some restrictive assumptions on the interconnection network, the algorithm described here and its attached implementation are scalable in the sense that their parallel efficiency is maintained constant with respect to the per processor memory usage.



$\frac{2}{3}n^3 + 2n^2$ ops / time

UNIT = 10**6 TIME/(1/3 100**3 + 100**2)

Facility		TIME N=100 secs.	UNIT micro- secs.	Computer	Type	Compiler
NCAR	14.0	.049	0.14	CRAY-1	S	CFT, Assembly BLAS
LASL	4.64	.148	0.43	CDC 7600	S	FTN, Assembly BLAS
NCAR	3.58	.192	0.56	CRAY-1	S	CFT
LASL	3.27	.210	0.61	CDC 7600	S	FTN
Argonne	2.31	.297	0.86	IBM 370/195	D	H
NCAR	1.91	.359	1.05	CDC 7600	S	Local
Argonne	1.77	.388	1.33	IBM 3033	D	H
NASA Langley	1.40	.489	1.42	CDC Cyber 175	S	FTN
U. Ill. Urbana	1.56	.506	1.47	CDC Cyber 175	S	Ext. 4.6
LLL	1.24	.554	1.61	CDC 7600	S	CHAT, No optimize
SLAC	1.19	.579	1.69	IBM 370/168	D	H Ext., Fast mult.
Michigan	1.09	.631	1.84	Amdahl 470/V6	D	H
Toronto	.772	.890	2.59	IBM 370/165	D	H Ext., Fast mult.
Northwestern	.477	1.44	4.20	CDC 6600	S	FTN
Texas	.356	1.93*	5.63	CDC 6600	S	RUN
China Lake	.352	1.95*	5.69	Univac 1110	S	V
Yale	.265	2.59	7.53	DEC KL-20	S	F20
Bell Labs	.199	3.46	10.1	Honeywell 6080	S	Y
Wisconsin	.197	3.49	10.1	Univac 1110	S	V
Iowa State	.194	3.54	10.2	Itel AS/5 mod3	D	H
U. Ill. Chicago	.166	4.10	11.9	IBM 370/158	D	G1
Purdue	.121	5.69	16.6	CDC 6500	S	FUN
U. C. San Diego	.0524	13.1	38.2	Burroughs 6700	S	H
Yale	.0402	17.1*	49.9	DEC KA-10	S	F40

* TIME(100) = (100/75)**3 SGEFA(75) + (100/75)**2 SGESL(75)

FIGURE 1
A comparison of execution times of various computer-compiler combinations

From Jack Dongarra's copy of the *LINPACK User's Guide* circa 1979

The HPL software package requires the on-system availability of an implementation of the Message Passing Interface (MPI) (1.1 compliant). An implementation of either the BLAS or the Vector Signal Image Processing Library (VSIPL) is also needed. Machine-specific as well as generic implementations of MPI, the BLAS, and VSIPL are available for a large variety of systems.

RECENT PUBLICATIONS

Dongarra, J. "Performance of Various Computers Using Standard Linear Equations Software (LINPACK Benchmark Report)," University of Tennessee Computer Science *Technical Report*, CS-89-85, 2001.

FAQ

http://www.netlib.org/utk/people/JackDongarra/faq-linpack.html

RELATED URLS

Basic Linear Algebra Subprograms (BLAS) - http://www.netlib.org/blas/

High Performance LINPACK (HPL) - http://icl.cs.utk.edu/hpl/

Message Passing Interface (MPI) - http://www.unix.mcs.anl.gov/mpi/

Vector Signal Image Processing Library (VSIPL) - http://www.vsip.org/

NA-Net/NA-Digest

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra

Jeremy Millar

Keith Moore

Mike Walters U

U = UNDERGRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/nadigest/

E-MAIL

nadigest@cs.utk.edu

COLLABORATORS

Lucent Technologies

Stanford University

The MathWorks

NA-Net is a system developed to serve the numerical analysis community and researchers with similar interests. This system provides three services to its members: 1) a mailing list, 2) a white pages database, and 3) a weekly news digest, NA-Digest, which contains news and correspondence relevant to the numerical analysis community, e.g. conference announcements, book announcements, software releases, etc. and is distributed to NA-Net members. The NA-Net mailing list also provides a forum for discussing issues relevant to the numerical analysis community. The NA-Net white pages database provides a directory of contact information for members of the numerical analysis community. This information includes current phone numbers, email addresses, and postal addresses. Figure 1 and Table 1 provide the latest demographic information for NA-Net membership.

TABLE 1. NA-NET MEMBERSHIP DEMOGRAPHIC INFORMATION

DOMAIN	COUNTRY CODE	MEMBERSHIP	DOMAIN	COUNTRY CODE	MEMBERSHIP
am	Armenia	1	cz	Czech Republic	36
ar	Argentina	27	de	Germany	558
at	Austria	51	dk	Denmark	52
au	Australia	155	dz	Algeria	2
be	Belgium	99	ec	Ecuador	1
bg	Bulgaria	25	edu	US Educational	1897
bh	Bahrain	1	ee	Estonia	3
bo	Bolivia	3	eg	Egypt	1
br	Brazil	82	es	Spain	162
bw	Botswana	1	et	Ethiopia	1
by	Belarus	7	fi	Finland	36
ca	Canada	228	fr	France	410
ch	Switzerland	75	gb	Great Britain	1
cl	Chile	16	ge	Georgia	1
cm	Cameroon	1	gov	US Government	306
cn	China	67	gr	Greece	76
co	Colombia	5	hk	Hong Kong	31
com	US Commercial	1377	hr	Croatia	15
cr	Costa Rica	3	hu	Hungary	20
cu	Cuba	5	id	Indonesia	15
cy	Cyprus	3	ie	Ireland	22

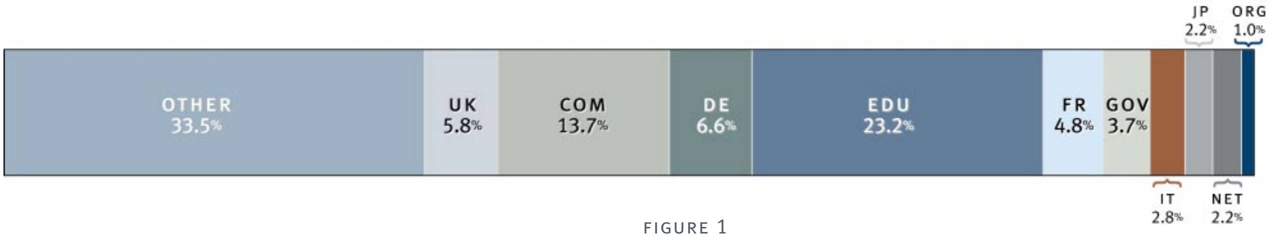


FIGURE 1
Percentage breakdown of the NA-Net Membership
Data as of October 15, 2001

TABLE 1 CONT'D . NA-NET MEMBERSHIP DEMOGRAPHIC INFORMATION

DOMAIN	COUNTRY CODE	MEMBERSHIP	DOMAIN	COUNTRY CODE	MEMBERSHIP	DOMAIN	COUNTRY CODE	MEMBERSHIP
il	Israel	77	my	Malaysia	7	si	Slovenia	10
in	India	114	na	Namibia	1	sk	Slovak Republic	9
int	International	1	net	Network	230	sn	Senegal	1
ir	Iran	33	nl	Netherlands	153	su	USSR (former)	29
is	Iceland	3	no	Norway	77	sz	Swaziland	1
it	Italy	251	nz	New Zealand	43	tc	Turks/Caicos Islands	1
jo	Jordan	3	om	Oman	1	th	Thailand	9
jp	Japan	186	org	Non-Profit	103	tn	Tunisia	2
kr	South Korea	77	pa	Panama	2	tr	Turkey	41
kw	Kuwait	2	pe	Peru	5	tw	Taiwan	46
kz	Kazakhstan	1	ph	Philippines	6	ua	Ukraine	16
lb	Lebanon	41	pk	Pakistan	6	uk	United Kingdom	483
lt	Lithuania	4	pl	Poland	65	us	United States	6
lv	Latvia	1	pt	Portugal	53	uy	Uruguay	2
ma	Morocco	6	py	Paraguay	1	ve	Venezuela	13
md	Maldova	4	ro	Romania	44	vn	Vietnam	6
mil	US Military	55	ru	Russia	132	yu	Yugoslavia	13
mk	Macedonia	1	sa	Saudi Arabia	15	za	South Africa	33
mo	Macau	3	se	Sweden	144		**TOTAL**	**8600**
mx	Mexico	27	sg	Singapore	23			

RELATED URLS

NA-Net - http://www.netlib.org/na-net/na_home.html

NetBuild

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra

Keith Moore

WEB SITE

http://icl.cs.utk.edu/netbuild/

E-MAIL

netbuild@cs.utk.edu

AGENCY FUNDING

National Science Foundation

The purpose of the NetBuild project is to make it easier for authors and users of scientific computation software to utilize standardized mathematical software libraries. Specifically, NetBuild attempts to eliminate the need for authors and users to locate, download, configure, compile, and install each of the mathematical software libraries that are required by a program. NetBuild will attempt to find pre-compiled versions of those software libraries that are suitable for the target platform, automatically download those components, and incorporate them into the program. When no pre-compiled libraries are available, NetBuild will have the capability to download the source code and to compile the libraries from source. Though similar facilities exist on some platforms, notably Linux, FreeBSD, and NetBSD, NetBuild attempts to be usable on a wide range of platforms. It is also specifically intended for use with high-performance mathematical software, which, for good efficiency, often requires fine-grained tuning to specific characteristics of target platforms.

As currently envisioned, NetBuild consists of three principal tools. The *netbuild* tool assists users of computational software with incorporating mathematical software libraries into their codes at link time. The *netloader* library performs a similar function to the netbuild tool, but is designed to select and download appropriate libraries at run-time. This will allow programs to be extensible at run time without those programs having to have specific knowledge of the target platform. It also allows such programs to automatically utilize the very latest version of each library, even it if has been updated since compile time. The *netcompile* tool enables builders of mathematical software libraries to supply those libraries in a form that can be used by netbuild and/or netloader thus supplying appropriate metadata and digitally signing the libraries for integrity protection. The netcompile tool is similar to netbuild in that it also runs the system's existing compilers and other tools within a special environment. However the purpose of netcompile is to help automate the process of creating libraries for use by netbuild and netloader. Netcompile automatically records the characteristics of the environment in which the library was compiled along with other information supplied by the library developer. This information will then be added to NetBuild's networked database and to NetBuild's clients for use in target matching. Netcompile can also help automate the process of creating several variants of a library for slightly different target environments.

In order to be as compatible as possible with existing Makefiles and compile scripts, NetBuild provides a transparent interface. The outer make process or compile script is invoked via the netbuild command, so instead of typing *make* one would simply type *netbuild make*. NetBuild then arranges for subprocesses to be run in a modified environment that intercepts calls to compilers and linkers. Whenever a compiler or linker is called, NetBuild will parse that tool's command line, determine which (if any) libraries that are requested are non-resident, and download and install the necessary libraries. NetBuild then invokes the real compiler or linker with suitably-modified arguments to cause the newly downloaded libraries to be linked along with any libraries that were already resident. To the NetBuild user it appears that all of the necessary libraries are already present. In most cases, no changes to Makefiles or compile scripts are necessary.

NetBuild attempts to minimize the security risk to the user that is inherent in use of downloaded software. Downloaded libraries are checked for integrity and must be cryptographically signed by a party that is trusted by the user. Each user can specify his or her own policy for whom to trust. For this reason, a library installed by a particular user is only used by that user; multiple users on the same system do not automatically share libraries that have been installed by NetBuild.

NetBuild is intended to work in conjunction with the Netlib repositories to make a wide range of high-quality mathematical software automatically available to users. NetBuild can also utilize libraries found in other repositories. In doing so, NetBuild can free up resources that would otherwise be used to maintain those libraries separately on each user's computer, allowing libraries to be automatically updated whenever updates are needed.

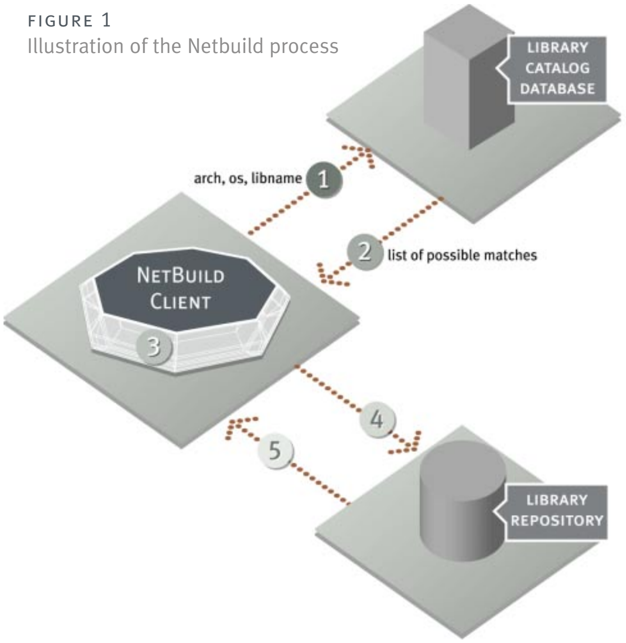
Software packages such as ATLAS and PAPI exploit specific characteristics of the target platform in order to provide increased efficiency or functionality. In order to choose the most appropriate versions of such libraries, NetBuild is capable of detecting fine-grained attributes of the target platform and matching those against the attributes of several different versions of a library. This prevents linking of library versions that will not run on the target platform and also allows the user to choose the fastest version of a library for the target platform when multiple versions are available.

Among the attributes which can be examined are:

- Instruction set architecture (e.g., Alpha, IA32, Sparc, etc.)
- Instruction set version
- Instruction set extensions (e.g., SSE, 3DNow!)
- Sizes of various caches
- CPU chip vendor and chip version
- Operating system
- Operating system version
- Operating system extensions
- Source language
- Compiler and version
- Compiler ABI / calling sequence
- Object file format

In order to select the best match of a library for a target platform, NetBuild queries a network-accessible database to learn which libraries are available for that general class of platforms (where class is CPU architecture and OS). That database then returns a list of possible matches. NetBuild evaluates each of those alternatives to eliminate unsuitable matches and to order those that are suitable according to their predicted performance on that target. This two-step strategy allows NetBuild's matching algorithm to be adjusted on a per-user or per-platform basis, allowing NetBuild to be used with new platforms not anticipated by the library builders.

FIGURE 1
Illustration of the Netbuild process



1 The netbuild tool requests a list of candidate libraries which match the target architecture, operating system, and library name.

2 The catalog database returns a list of candidate libraries which match the criteria supplied by the netbuild tool.

3 The netbuild tool evaluates each candidate for suitability, and produces a list of the suitable candidates which is ordered according to preference.

4 The netbuild tool requests the selected library from a library server.

5 The netbuild tool downloads the selected library, verifies its signature, and installs it where it can be linked with the user's code.

RECENT PUBLICATIONS

Moore, K., Dongarra, J. "NetBuild: Transparent Cross-Platform Access to Computational Software Libraries", submitted to *Concurrency: Practice and Experience*, July 2001.

http://icl.cs.utk.edu/publications/pub-papers/2001/netbuild-C-PE.pdf

RELATED URLS

ATLAS – http://icl.cs.uk.edu/atlas/

FreeBSD - http://www.freebsd.org/

NetBSD - http://www.netbsd.org/

PAPI – http://icl.cs.utk.edu/papi/

Netlib

Netlib is an online repository of freely available software, documents, and databases. Created in 1985 to provide a central location for dissemination of high-quality scientific computing codes, Netlib is now replicated worldwide. Primary servers are located in Tennessee (UTK/ICL), New Jersey (Bell Labs), the United Kingdom, and Norway, with 16 mirror sites scattered across the globe. In addition to distributing content, the Netlib system also distributes the administrative load of collection maintenance. Each server can act as a master server for some subset of the Netlib collection. In addition, each subset is replicated and synchronized across all of the Netlib servers, providing a globally consistent view of the collection.

Netlib is an evolving collection, and submissions from the scientific computing community are welcome. Each submission undergoes review by the Netlib editorial board, ensuring quality and relevance to the collection. Currently, Netlib contains over 15,000 files totaling approximately 2GB.

The Netlib collection is available via a number of protocols including email, gopher, FTP, HTTP, rsync, and Xnetlib. Users can retrieve individual routines or entire libraries. Additionally, Netlib maintains lists of subroutine dependencies. This enables a user to download an individual routine plus any additional required routines.

Netlib has been and continues to be an important fixture of the scientific computing community. Figure 1 illustrates the total number of requests to the Tennessee server since its inception in 1985.

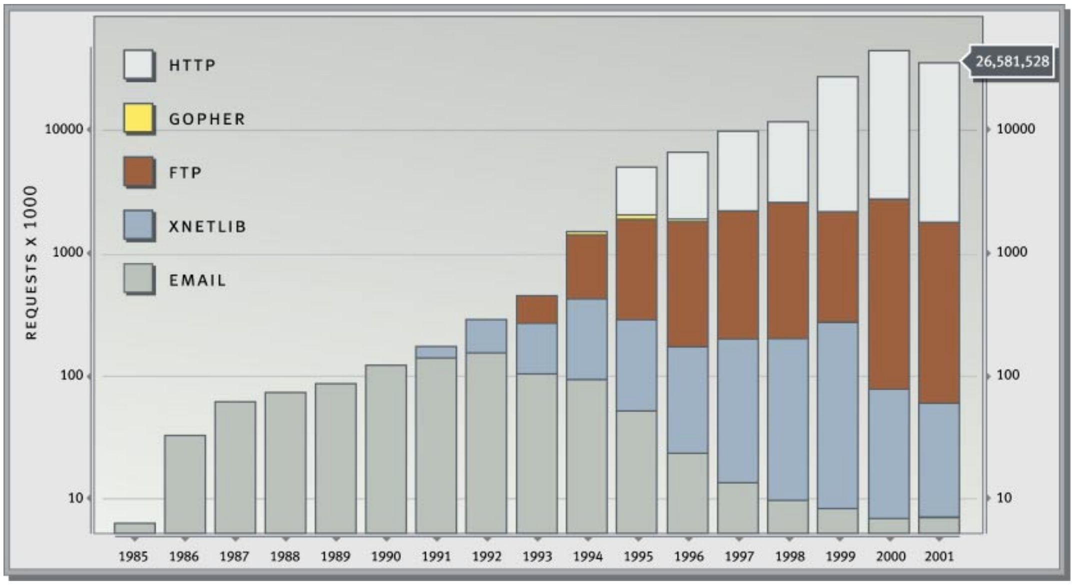


FIGURE 1
Logarithmic chart depicting requests made to Netlib repositories at UT and ORNL Data as of October 15, 2001

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra

Jeremy Millar

Farial Shahnaz U

Mike Walters U

U = UNDERGRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/netlib/

E-MAIL

netlib@cs.utk.edu

COLLABORATORS

Lucent Technologies

AGENCY FUNDING

National Science Foundation

MIRROR SITES

netlib.uow.edu.au	AUSTRALIA
AARNet	AUSTRALIA/NEW ZEALAND
unicamp.br	BRASIL
netlib.amss.ac.cn	CHINA
Daresbury Lab	ENGLAND
Codiciel	FRANCE
ENSEEIHT	FRANCE
ZIB	GERMANY
Athens	GREECE
Univ Thrace	GREECE
CASPUR	ITALY
C.D.S.	ITALY
CILEA	ITALY
Phase AIST	JAPAN
Ewha w. University	KOREA
ChgNet	RUSSIA
NCHC	TAIWAN
freesoftware.com	USA WEST

RECENT PUBLICATIONS

See http://www.netlib.org/srwn/index.html

FAQ

http://www.netlib.org/misc/faq.html

RELATED URLS

Matrix Market - http://math.nist.gov/MatrixMarket/

NA-Net – http://www.netlib.org/na-net/

National High-Performance Software Exchange (NHSE) - http://www.nhse.org/

MAIN NETLIB SERVERS

Bell Labs (Lucent Technologies) -
http://netlib.bell-labs.com/netlib/master/readme.html

Norway (Bergen) -
http://www.netlib.no/netlib/master/readme.html

United Kingdom (Kent) -
http://www.mirror.ac.uk/sites/
netlib.bell-labs.com/netlib/master/readme.html

NHSE

NATIONAL HIGH-PERFORMANCE SOFTWARE EXCHANGE

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra

Don Fike

Kevin London

Jeremy Millar

Shirley Moore

Mike Walters U

Scott Wells

U = UNDERGRADUATE STUDENT

WEB SITE

http://icl.cs.utk.edu/nhse/

E-MAIL

nhse@cs.utk.edu

COLLABORATORS

Argonne National Laboratory

California Institute of Technology

Rice University

Syracuse University

AGENCY FUNDING

National Aeronautics and Space Administration

The National High-performance Software Exchange (NHSE) project started in 1994 as an effort to promote sharing and reuse of parallel computing software and tools among and between the federal high performance computing (HPC) agencies. Using the highly successful Netlib mathematical software repository as a model, the NHSE sought to establish discipline-oriented software repositories that could be contributed to and maintained by experts in their respective fields. Because of the need to share software between organizations and across disciplines, repository interoperability was an important goal. Although much of the software was to be freely available, the NHSE developed recommendations for handling export controlled and otherwise restricted software.

To enable the establishment of discipline-oriented software repositories, the NHSE project developed the Repository in a Box (RIB) toolkit. RIB makes use of a Web server as well as an underlying database and facilitates creation and maintenance of interoperable, Web-based metadata repositories. To achieve interoperability, RIB is based on standards, including the IEEE Basic Interoperability Data Model (BIDM) for exchanging software metadata and the WWW Consortium's eXtensible Markup Language (XML).

In addition, the NHSE has led and/or participated in the development of two IEEE standard extensions to the BIDM. The first, called the Asset Certification Framework (ACF), was created to promote systematic review of HPC software. The second extension, called the Intellectual Property Rights Framework (IPRF), allows organizations to describe software access restrictions and property rights such as copyrights, export restrictions, and licensing requirements. The NHSE has also developed its own non-standard extensions to the BIDM for describing software deployment (i.e., where software is installed and how to use it) and benchmark and performance results. The extensible data modeling approach based on the BIDM allows all relevant information about HPC software to be maintained and accessed through a common interface.

A number of organizations within the HPC community, including the National Aeronautics and Space Administration (NASA) the National Science Foundation (NSF), the Department of Defense (DoD), and the Department of Energy (DOE) have used RIB to establish software repositories in such areas as programming tools, computational chemistry, signal-image processing, and challenge applications. The NHSE itself maintains repositories in the areas of high performance math software, parallel tools, and benchmark programs, and provides a high level view of all the repositories with which it interoperates. Many of these repositories are located here at ICL and are maintained by discipline experts within our group.

Also in the area of software review, the NHSE continues to produce the electronic journal called the *NHSE Review*. The *NHSE Review* publishes articles by experts on comparative evaluations of different types of HPC software, including batch queuing systems, Message Passing Interface (MPI) implementations, debugging and performance analysis tools, as well as iterative methods for solving linear systems. The *Review* is published at ICL and can be found on the NHSE Web site, which is also maintained by ICL.

RELATED URLS

Netlib - http://icl.cs.utk.edu/netlib/

Repository in a Box (RIB) - http://icl.cs.utk.edu/rib/

NetSolve

R&D 100
1999
WINNER

DEVELOPMENT TEAM
ALPHABETICAL

Sudesh Agrawal G
Dorian Arnold
Susan Blackford
Jack Dongarra
Brian Drum U
Victor Eijkhout
Michelle Miller
Keith Moore
Kiran Sagi G
Zhiao Shi G
Sathish Vadhiyar G
Dong Woo Lee V
Asim YarKhan

U = UNDERGRADUATE STUDENT
G = GRADUATE STUDENT
V = VISITOR

WEB SITE

http://icl.cs.utk.edu/netsolve/

E-MAIL

netsolve@cs.utk.edu

COLLABORATORS

University of California, San Diego
University of Wisconsin

AGENCY FUNDING

Department of Defense
Department of Energy
National Science Foundation

NetSolve is a project that investigates the use of distributed computational resources connected by computer networks to efficiently solve complex scientific problems. It is a remote procedure call (RPC)-based client/agent/server system that allows users to discover, access, and utilize remote software modules and the hardware needed to run these modules. NetSolve facilitates heterogeneous computing, or the ability to combine different machine architectures and/or operating systems to solve a problem.

NetSolve is grid middleware that binds grid systems and problem-solving environments, while allowing the flexibility for a user to write their own front-end or embed a call to NetSolve using a C or Fortran language client API. Fundamental characteristics include:

- Ease-of-use for both the user and administrator
- Efficient utilization of resources
- Ease-of-integration of new software modules

Although many research groups and organizations are investigating distributed and grid computing concepts, NetSolve's niche is providing access to complex libraries of high-performance software that run on clusters of commodity processors or supercomputers. Such access reduces the effort scientists normally exert locating and installing these software resources. At the same time, the system can aggregate hardware resources to solve larger problems.

A NetSolve client user accesses the system through the use of simple and intuitive application programming interfaces (APIs). An API has been implemented in a variety of languages and environments, including C, Fortran, Matlab, and Mathematica. These interfaces allow client users to request NetSolve to operate on user-provided data, using software previously provisioned on a NetSolve server. The NetSolve client call automatically contacts the NetSolve information service and resource scheduler, the NetSolve agent, to find a machine resource on which to run the software library call. The agent uses both static and dynamic information collected from NetSolve computational servers to determine which server can service the client user's request most efficiently. The recommended server is returned to the NetSolve client software, which automatically negotiates with the specified computational server to start running the job with the input data. Once the job completes, output data is returned to the user's NetSolve client call.

From a NetSolve administrator's perspective, the system has two key components: an agent and a server. The agent represents the gateway to the NetSolve system. It maintains a database of NetSolve servers along with their capabilities (hardware performance and allocated software) and dynamic usage statistics for use in scheduling decisions (mapping client requests to software servers). The NetSolve agent attempts to find the server that will service the request in the least time, balance the load amongst its servers, and keep track of failed servers. Requests are directed away from failed servers. The agent also adds fault-tolerant heuristics that attempt to use every likely server until it finds one that successfully services the request.

The NetSolve server, the computational backbone of the system, is a daemon process that awaits client requests. The server can run on single workstations, clusters of workstations, symmetric multiprocessors (SMPs), or massively parallel processors (MPPs). One key component of the server is the ability to wrap software library routines into NetSolve software services by using an Interface Definition Language (IDL) facility called the NetSolve Problem Description File (PDF). A PDF specifies the software interface to library routines. A code generator converts the PDF into source code and then executable code during compilation of NetSolve. Then a computational server calls the generated code to invoke the library code to service a user request for these routines.

FIGURE 1
Illustration depicting some of the applications and research fields that utilize NetSolve

There are many advantages to using NetSolve. NetSolve can provide access to otherwise unavailable software and, in cases where the software is readily available, it can make the power of supercomputers accessible from low-end machines such as laptop computers. NetSolve is designed to suit the needs of the domain scientist who is not necessarily a mathematician or computer scientist. For such a user, the system provides simple, intuitive, and uniform interfaces to a wide variety of numerical functions. NetSolve is also designed to increase the accessibility of larger software systems like simulators and modeling software. NetSolve aids the portability problem by allowing users to access non-portable code from any kind of machine via a NetSolve client call. NetSolve can also be used to extend the capabilities of problem solving environments (PSE), such as Matlab, by increasing the number and types of implemented algorithms available. The system also provides these environments with the ability to distribute NetSolve's computational tasks among multiple processors – a feat, for example, that is not possible with Matlab alone.

NetSolve has been employed by users in a variety of scientific domains ranging from image processing to nuclear engineering, microbiology, sub-surface fluid modeling, and bioelectric field modeling. Since the software system is freely available for download and use from the project Web site, there are many undocumented cases of NetSolve use. Our experience, in addition to requests for technical support, tells us that the majority of these cases are of users who have embedded calls to NetSolve within their computational science applications to access lower-level services (i.e., numerical linear algebra routines like linear system solvers, eigensolvers, differential equations, etc.). The NetSolve project has also been part of larger collaborations with research groups from universities, government laboratories, and private research organizations. The aims of these collaborations have primarily been to integrate NetSolve with large scientific applications and simulations to improve the performance of these applications through resource aggregation, as well as to make such software more readily accessible to user communities.

RECENT PUBLICATIONS

Miller, M., Moulding, C., Dongarra, J., Johnson, C. "Grid-Enabling Problem Solving Environments: A Case Study of SCIRUN and NetSolve," *Proceedings of the High Performance Computing 2001 Grand Challenges in Computer Simulation [HPC], Special Track on "High Performance Simulation Environments"*, Seattle, Washington, April 22-26, 2001.

RELATED URLS

IPARS - http://www.ticam.utexas.edu/CSM/ACTI/ipars.html
MCell - http://www.mcell.cnl.salk.edu/
ORNL - http://www.ornl.gov/
Salk Institute for Biological Studies- http://www.salk.edu/
SCIRun – http://www.sci.utah.edu/

PAPI

PERFORMANCE APPLICATION PROGRAMMING INTERFACE

R&D 100
2001
WINNER

DEVELOPMENT TEAM
ALPHABETICAL

Leon Dong G
Jack Dongarra
Kevin London
Shirley Moore
Phil Mucci
Keith Seymour
Thomas Spencer U
Dan Terpstra
Luke Zhou G

U = UNDERGRADUATE STUDENT
G = GRADUATE STUDENT

WEB SITE
http://icl.cs.utk.edu/papi/

E-MAIL
papi@cs.utk.edu

AGENCY FUNDING
Department of Defense
Department of Energy
National Science Foundation

INDUSTRIAL SUPPORT
IBM Research
Intel

The purpose of the PAPI project is to define a standardized, easy to use interface that provides access to the hardware performance counters on most major processor platforms, thereby providing application developers the information they need to tune their software on different platforms. The goal is to make it easy for users to gain access to the counters to aid in performance analysis, modeling, and tuning.

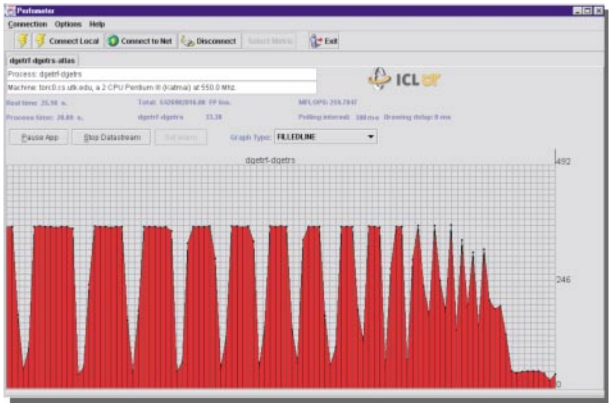
The approach of the PAPI project has been to work with the high performance computing (HPC) community, including users and vendors, to choose a common set of hardware events and to define a cross platform library interface to the underlying counter hardware. The common events are those considered to be most relevant and useful in tuning application performance. As large a subset as possible has been mapped to the corresponding machine-specific events on the major HPC platforms. The intent is that the same metric would count similar, and possibly comparable, events on different platforms. The intent is to standardize the names for the metrics, not the exact semantics, which necessarily depend on the processor under study. In addition to providing access to hardware counters, PAPI provides timing routines and routines for obtaining information about the hardware and the executable. The timing routines are implemented using the most accurate timers available on the platform and have the advantage of allowing the same timer calls to be used across platforms.

Using traditional tools to obtain performance data for large-scale applications can be cumbersome and inefficient. The data collected may consist only of summary statistics and may provide little insight into the dynamic runtime behavior of the application. Many tools require that the data to be collected be specified prior to runtime and do not allow for runtime selection or adjustment of which events to measure or the level of detail or granularity at which measurements should be made. To address these issues, the tool infrastructure being developed to complement the PAPI library facilitates runtime tracing and analysis of hardware counter data, as well as runtime control over performance instrumentation. A dynamic instrumentation capability currently under development will provide mechanisms for calls to PAPI library and utility routines to be dynamically inserted into and removed from running applications, thus allowing runtime control over the selection of hardware events and of counting modes and granularity. In addition, work is underway to provide routines for portable access to memory utilization information. We plan to provide routines to obtain memory information such as the following:

- memory available on a node
- total memory available/used
- 'high-water-mark' memory used by process and by thread
- disk swapping by process
- process/memory locality
- location of memory used by an object (e.g., array or struct).

Although the PAPI library can be used directly by application developers, most application scientists will prefer to access hardware performance data via end-user performance analysis tools. The PAPI project team has developed a tool for the graphical display of PAPI performance data in a manner useful to the application developer. The tool front end is written in Java and can be run on a separate machine from the program being monitored. All that is required for real-time monitoring and display of application performance is a socket connection between the machines. The tool, called the perfometer, provides a runtime trace of a chosen PAPI metric, as shown in Figure 1 for floating operations per second (PAPI__FLOPS). This particular example illustrates how calls to PAPI routines at

FIGURE 1
PAPI Perfometer displaying MFLOPS



Process: dgetrf-dgetrs
Machine: torc0.cs.utk.edu, a 2 CPU Pentium III (Katmai) at 550.0 Mhz.
Real time: 25.58 s.
Process time: 26.88 s.
dgetrf-dgetrs 33.38
Graph Type: FILLEDLINE

the beginnings and ends of procedures can provide information about the relative floating point performance of those procedures. The same display can be generated for any of the PAPI metrics.

Performance tool developers are beginning to exploit the capability of the PAPI interface. For example, the Paradyn project provides a performance measurement tool that scales to long-running programs on large parallel and distributed systems. The Paradyn developers plan to use PAPI in future versions of Paradyn, both because it offers a single API to access timer data across different computing platforms and because PAPI unifies the semantics of the counter data access, providing Paradyn with a simpler interface.

Another tool to benefit from PAPI is VAMPIR, a performance analysis tool for MPI parallel programs developed by Pallas in Germany. The next version of VAMPIR will support OpenMP in addition to MPI and will use PAPI to access hardware performance counters. With PAPI, VAMPIR will be able to use hardware counter data to guide detailed event based analysis which will allow users to quickly locate performance bottlenecks in their applications. Pallas and Intel/KAI are developing a new performance analysis tool set for combined MPI and OpenMP program analysis that uses PAPI to access the hardware performance counters. PAPI's standard performance metrics, which include metrics for shared memory processors (SMPs), will provide accurate and relevant performance data for the clustered SMP environments targeted by the new tool set. Other tools that have incorporated support for PAPI include SvPablo from the University of Illinois and IBM and TAU from the University of Oregon.

Efforts continue to expand both the feature set of PAPI and the number of platforms on which it can be used. In the past year, software multiplexing of the often limited hardware counters was incorporated into the PAPI interface. This feature makes it possible to monitor a large number of events in long running programs where sampling granularity is not an issue. PAPI also increased the breadth of supported platforms by adding the Compaq Alpha and Intel Itanium to its hardware repertoire, and by adding Windows NT/2000/XP to its list of supported operating systems.

PAPI has developed a substantial user base. Several DOE labs are using PAPI and it is installed at most of the Department of Defense (DoD) Major Shared Resource Centers (MSRCs) and PAPI is used outside the US as well, with reports of use coming from Australia, Austria, Brazil, Canada, France, Germany, Greece, Ireland, Italy, Japan, Spain, Sweden, Switzerland and Vietnam.

RECENT PUBLICATIONS

London, K., Moore, S., Mucci, P., Seymour, K., Luczak, R. "The PAPI Cross-Platform Interface to Hardware Performance Counters", *Department of Defense Users' Group Conference Proceedings* (to appear), Biloxi, Mississippi, June 18-21, 2001.

Dongarra, J., London, K., Moore, S., Mucci, P., Terpstra, D. "Using PAPI for Hardware Performance Monitoring on Linux Systems," *Conference on Linux Clusters: The HPC Revolution*, Urbana, Illinois, June 25-27, 2001.

FAQ

http://icl.cs.utk.edu/projects/papi/faq.html

RELATED URLS

Autopilot and SvPablo – http://vibes.cs.uiuc.edu/

DoD Major Shared Resource Centers -
http://www.hpcmo.hpc.mil/Htdocs/MSRC/index.html

KAI - http://www.kai.com/parallel/kappro/

Lawrence Livermore National Laboratory - http://www.llnl.gov/

Los Alamos National Laboratory – http://www.lanl.gov/

Pacific Sierra Research –
http://www.psrw.com/deep_papi_top.html

Parallel Tools Consortium – http://www.ptools.org/

Paradyn - http://www.cs.wisc.edu/paradyn/

Sandia Livermore National Laboratory - http://www.sandia.gov/

TAU – http://www.cs.uregon.edu/research/paracomp/tau/

VAMPIR - http://www.pallas.com/pages/vampir.htm

RIB

REPOSITORY IN A BOX

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra
Don Fike
Jeremy Millar
Shirley Moore
Terry Moore
Scott Wells

WEB SITE

http://icl.cs.utk.edu/rib/

E-MAIL

rib@cs.utk.edu

AGENCY FUNDING

Department of Defense
National Aeronautics and Space Administration

Repository In a Box (RIB) is a toolkit for creating and maintaining web-based, interoperable metadata repositories. In this context, a repository is a collection of metadata and a searchable catalog for browsing such metadata. Repositories created with RIB can seamlessly share their data (interoperate) with one another, which is the key functionality of the toolkit. Furthermore, RIB allows the creation of "virtual repositories" – repositories that contain no metadata but contain catalogs of metadata gathered through various interoperations.

The RIB software was developed by the National High-performance Software Exchange (NHSE) technical team at ICL. RIB was originally conceived as a tool to facilitate the exchange and reuse of high performance applications within the high performance computing (HPC) community, and this remains its primary use. However, RIB has evolved in such a way that it now supports the creation and exchange of nearly any digital object.

Each repository created with RIB contains a data model to which the cataloged metadata conforms. The data models supported by RIB are entity-relationship models and are extremely flexible. RIB uses a default data model that has been standardized by the Institute of Electrical and Electronics Engineers (IEEE). This standard, IEEE Std. 1402 - Basic Interoperability Data Model (BIDM), defines the minimal set of information necessary for the exchange of digital library objects between libraries. The primary object defined by the BIDM is an *Asset*. This object class is often used to describe a particular piece of software, e.g., a library (ScaLAPACK) or application (Globus). Other object classes defined by the BIDM include *Organization*, *Element* (files, etc.), and *Library*. Each of these classes defines several descriptive attributes. For example, the *Asset* class defines descriptive attributes, such as abstract, cost, and version. Figure 1 shows an illustration of RIB's interoperability function.

Since publication of the BIDM standard, IEEE and NHSE have defined several extensions to the basic BIDM data model. IEEE extensions include the 1420.a - Asset Certification Framework (ACF) and 1420.b - Intellectual Property Rights Framework (IPRF). These are official extensions to the standard. NHSE extensions, while not standard, are also widely deployed throughout the RIB community. These include minor additions to the *Asset* class, and the addition of *Machine* and *Deployment* classes. The addition of these latter two classes allows information regarding software deployments to be cataloged and published.

RIB supports each of these data models as well as the ability to support many others. Additionally, RIB provides a data model editor so that a repository administrator can extend existing models or create new ones. As a metadata management and interoperation tool, RIB is capable of building catalogs of any type of object that can be described by an entity-relationship data model.

RIB also uses the World Wide Web (W3C) Consortium's eXtensible Markup Language (XML) standard to achieve maximum interoperability. Coupled with the RIB application programmer interface (RIB API), XML allows metadata to be shared seamlessly between repositories and allows RIB to export metadata in a readily understandable manner.

RIB generates catalogs based on a repository's data model and the metadata contained within its database. The catalogs are structured hierarchically, based on a key attribute called a domain, which is typically a broad subject area. Each catalog page is dynamically generated by RIB so that updates to

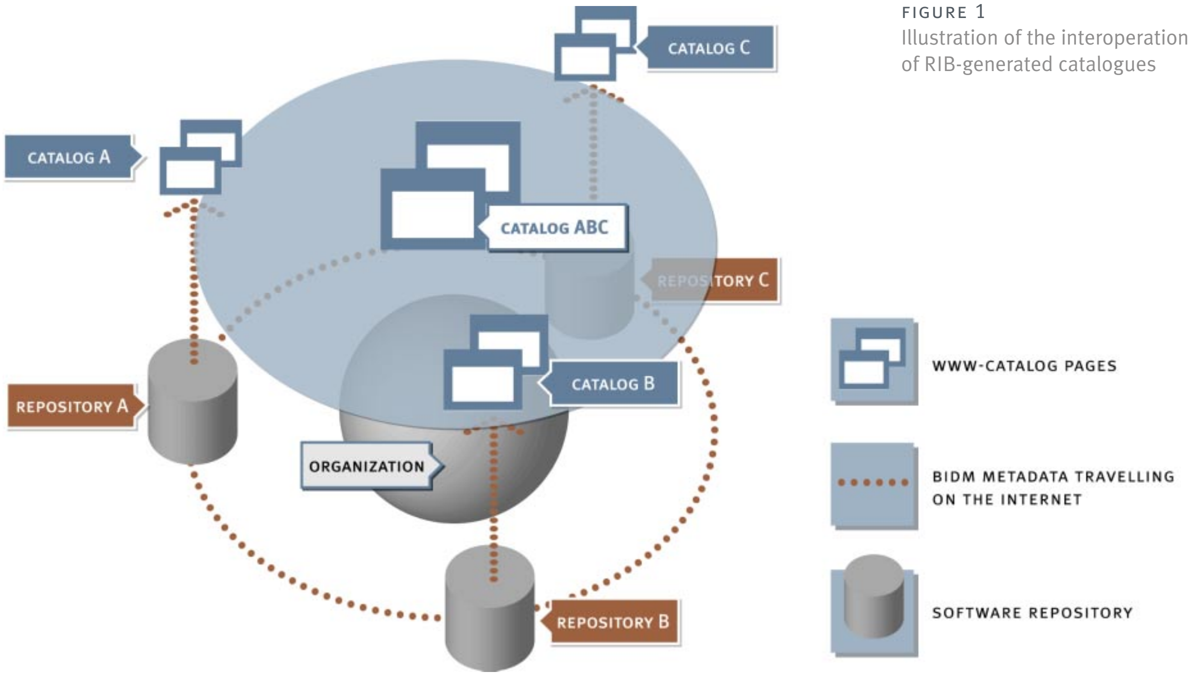


FIGURE 1
Illustration of the interoperation of RIB-generated catalogues

the repository propagate immediately. This is in contrast to systems such as Netlib, for example, where updates become visible the next day.

In addition, RIB provides the ability to form "joins" for repositories with a properly structured data model. In order to use this functionality, the data model must define at least one "intersection class." Once this is done, RIB can dynamically generate tables depicting the join. This feature is often used to generate software deployment tables with software along one axis, machines along the other, and deployment status as the table data.

RIB is currently in release 2.1.1 and runs on several widely available platforms including Linux and Windows. RIB has been deployed by various government organizations, including DoD, NASA, DOE, and NSF. Commercial/industrial customers such as Raytheon and Delphi Automotive Systems also make use of RIB. Additionally, the NHSE maintains several RIB repositories.

RECENT PUBLICATIONS

Dongarra, J., McMahan, P., Millar, J. "RIBAPI - Repository in a Box Application Programmer's Interface," University of Tennessee Computer Science Department *Technical Report*, UT-CS-00-438. January 2000.

RELATED URLS

eXtensible Markup Language (XML) - http://www.w3.org/XML/
National HPCC Software Exchange (NHSE) - http://www.nhse.org/
Netlib - http://www.netlib.org/

FAQ

http://icl.cs.utk.edu/projects/papi/faq.html

SInRG

SCALABLE INTRACAMPUS RESEARCH GRID

DEVELOPMENT TEAM
ALPHABETICAL

Micah Beck

Jack Dongarra

Brett Ellis

Terry Moore

WEB SITE

http://www.cs.utk.edu/sinrg/

E-MAIL

sinrg@cs.utk.edu

COLLABORATORS
UNIVERSITY OF TENNESSEE

Don Bouldin
ELECTRICAL & COMPUTER ENGINEERING

Peter Cummings
CHEMICAL ENGINEERING

Jens Gregor
COMPUTER SCIENCE

Louis Gross
ECOLOGY & EVOLUTIONARY BIOLOGY

Michael Langston
COMPUTER SCIENCE

James Plank
COMPUTER SCIENCE

Gary Smith
MEDICAL ARTS

Michael Thomason
COMPUTER SCIENCE

Robert Ward
COMPUTER SCIENCE

Rich Wolski
COMPUTER SCIENCE

AGENCY FUNDING

National Science Foundation

INDUSTRIAL SUPPORT

Dell Computer Corporation

Microsoft Research

Sun Microsystems

The Innovative Computing Laboratory is leading a large collaboration of faculty members from Computer Science and other UT departments in the creation of an experimental technology Grid on the Knoxville campus. The purpose of the Scalable Intracampus Research Grid (SInRG) is to support leading-edge research on technologies and applications for *grid* computing, which is the new paradigm for high performance distributed computing and information systems. A *computational power grid* like SInRG uses special system software, sometimes known as *network middleware*, to integrate high performance networks, computers, and storage systems into a unified system that can provide advanced computing and information services (e.g., data staging, remote instrument control, & resource aggregation) in a pervasive and dependable way for an entire community.

A national technology Grid is now growing out of the convergent efforts of NSF's Partnerships for Advanced Computational Infrastructure (PACI) and several other government agencies, including NASA, DOD, and DOE; a similar collective effort is currently underway in the European research community. As SInRG is deployed over the next few years, it will mirror within the boundaries of the Knoxville campus both the underlying technologies and the interdisciplinary research collaborations that are characteristic of the national and international technology Grid. SInRG's primary purpose is to provide a technological and organizational microcosm of this effort so that key research challenges of grid-based computing can be addressed using the advantages of local communication and control.

SInRG is supported by a five-year grant from the Research Infrastructure Program of the Computer and Information Science and Engineering directorate of the National Science Foundation (NSF). It is currently midway through its second year of development.

The vast majority of SInRG's funding will go to purchase special Grid Service Clusters (GSCs), which are hardware ensembles specifically designed and configured to fit SInRG's multifaceted research agenda. As shown in figure 1, which presents the deployment SInRG today (Fall 2001), each GSC consists of a compute engine (e.g., a large commodity cluster), a mass storage device, and a fast data switch that integrates them all and connects them to the campus' high performance network. While each GSC has been designed from the ground up to be a node on a grid, each is also assigned to one of the collaborating teams and is customized to meet their special needs. For example, the two newest GSCs deployed this year have unique features that were determined by the computational demands of the research:

- A collaboration between a team lead by Lou Gross (Computational Ecology) and Mike Berry (Computer Science) is exploring such phenomena as the optimal spatial patterns of utilization of antibiotics to control the spread of resistant bacteria, and the sensitivity of the Spatially Explicit Species Index (SESI) Models for Everglades restoration to changes in rainfall patterns and associated hydrologic controls. The computations to be parallelized in these applications are too tightly coupled to achieve good performance on today's commodity clusters, so the GSC for this SInRG application group (GSC #5) uses a 14 processor Symmetrical Multiprocessor (SMP) from Sun Microsystems as its compute engine.
- The image processing research being pursued by the SInRG application team led by Don Bouldin (Electrical and Computer Engineering) and Mike Langston (Computer Science) involves the use of cluster nodes containing special Field Programmable Gate Array (FPGA) chips that can be configured on the fly to implement application-specific computations. Their work has already shown that for some key applications these FPGAs can achieve order of

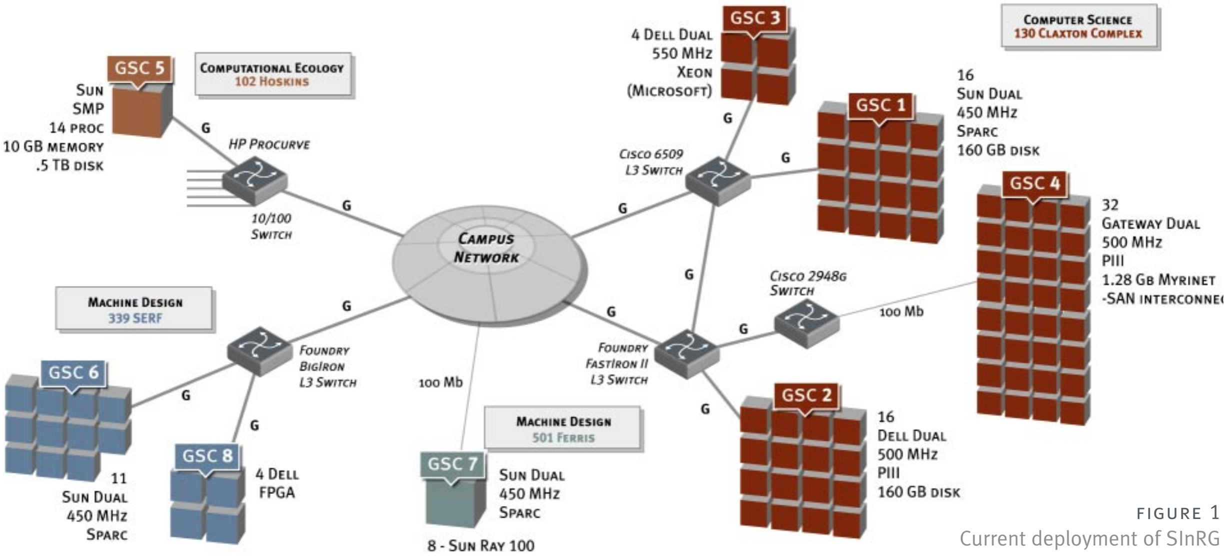


FIGURE 1
Current deployment of SInRG

magnitude improvements in price/performance over conventional processors. Because these special FPGAs are in a GSC (#6) that is part of SInRG, they will also be seamlessly available to other SInRG application teams who might benefit from their use, such as the Medical Imaging group.

These research applications reflect another aspect of the national effort via the fact that the effort is based on interdisciplinary collaboration between computer scientists and researchers from other domains with extremely challenging computational problems to solve. The Grid community recognizes that, in order to make rapid progress, the requirements of advanced applications must drive the development of Grid technology. The fact that SInRG has such well-established research collaborations to build on and that all the collaborators are on the same campus is a major advantage for the project.

Alongside these application groups, SInRG has a basic research group in Computer Science focused on research for grid middleware. The CS researchers, who make up the middleware group (Jack Dongarra, Jim Plank, Rich Wolski, and Micah Beck), bring complementary research interests and component software to the task at hand. This includes software for remote scientific computing (NetSolve), distributed scheduling (AppLeS), resource monitoring and performance prediction (Network Weather Service), and flexible management of distributed storage (IBP). The latest release of NetSolve, which is currently being deployed on SInRG, integrates support for both IBP and NWS into the system.

To create SInRG's system software this group is not only building on these different components, they are also leveraging the work of the PACI's and other parts of the national Grid community. This work is therefore part of a much larger story about the evolution of modern information power grids.

Like the national Grid, SInRG will be a geographically distributed system. By the end of the five years, there will be at least seven Grid Service Clusters spread among six different locations around the campus, including one across the Tennessee River at the UT Medical Center. Each will be managed with some degree of autonomy by these groups, with oversight and coordination from the CS co-PIs who collaborate with them. The campus network will provide the underlying fabric that makes it possible to use all this distributed hardware as a single collective resource.

Though the SInRG project is primarily supported by NSF, it also represents an excellent example of mutually beneficial collaboration between academia and private industry. Eager to support leading edge research on grid-based computing and to test out their technology in the environment SInRG provides, Microsoft and Sun Microsystems have made significant contributions of funding and/or technology to the project. Project leaders expect this kind of collaboration with industry to continue throughout the life of the project.

RECENT PUBLICATIONS

See http://www.cs.utk.edu/sinrg/docs/

RELATED URLS

AppLeS - http://apples.ucsd.edu/

IBP - http://icl.cs.utk.edu/ibp/

NetSolve - http://icl.cs.utk.edu/netsolve/

Network Weather Service - http://nws.cs.utk.edu/

Sparse Matrix Algortihms & Software

DEVELOPMENT TEAM ALPHABETICAL

Zizhong Chen G

Jun Ding G

Jack Dongarra

Victor Eijkhout

Piotr Luszczek G

Ken Roche G

G = GRADUATE STUDENT

E-MAIL

sparse@cs.utk.edu

WEB SITE

http://icl.cs.utk.edu/sparse/

Currently, several projects are underway at ICL related to the iterative solution of sparse linear systems arising from discretised partial differential equations.

sparse benchmark

We have developed a benchmark suite of representative iterative solvers. This suite includes several iterative methods, storage schemes, and preconditioners. Together these can give a representative picture of a machine's performance on iterative solvers. The intended use of the benchmark is as follows: users pick those kernels that are representative of the ones in their code and assemble the performance numbers to give an estimate of the per-iterative performance of their code on the benchmarked machine. Together with tests of the expected number of iterations (this can often be done by extrapolation from small problems) the user can gain an accurate idea of expected whole-code performance.

The benchmark comes with a test matrix generator and shell scripts for post- processing of results, including graphic display and automatic reporting. We are creating a results database on Netlib.

adaptive software

We are investigating various approaches to making software adapt itself to the existing problem and the available problem solving environment. In particular, we will focus on tuning method parameters to adapt to the problem, and tuning the implementation of the method chosen to adapt to the parallel architecture available.

problem-based method tuning

Iterative methods are sensitive to the numerics of a linear system in a far more pronounced way than direct methods. We are investigating two ways of tuning method parameters to adapt to the existing problem. First, we have developed a method for determining load balancing based on the matrix structure. The rationale for this is that matrix structure often reflects physical structure, and often follows changes in differential equation coefficients. Thus, for domain-decomposition based preconditioners we propose basing the subdomains on reconstructions of the physical domains. Figure 1 illustrates a reconstructed block structure.

Using such *a~posteriori* block structures instead of a more naïve split, based for instance on balancing numbers of rows or nonzeros, can lead to substantial improvements in performance. Figure 2 illustrates the convergence curve for an iterative method with an additive Schwarz preconditioner with partitionings based once on even numbers of rows and once based on reconstructed block structure. In preliminary tests we have seen reductions of the number of iterations by up to a third of the number for a naive partitioning. The partitioning approach just described uses only the graph structure of the matrix. Our second approach is to investigate tuning of the method and preconditioner based on the numerics of the matrix. For this we use 'measurement' of the matrix by making a limited length test run with GMRES.

performance optimization of iterative methods

In the parallel execution of iterative methods, most operations are fully parallel, or involve only nearest-neighbor communications that can be overlapped with computation. The inner product and norms form an exception, however. They induce a global synchronization reduction/broadcast operation, which cannot be overlapped with useful computation, at least in a naive formulation of the methods. We have made an inventory of existing approaches for alleviating this load and have proposed a few new methods. This suite of equivalent formulations of the iterative methods has been

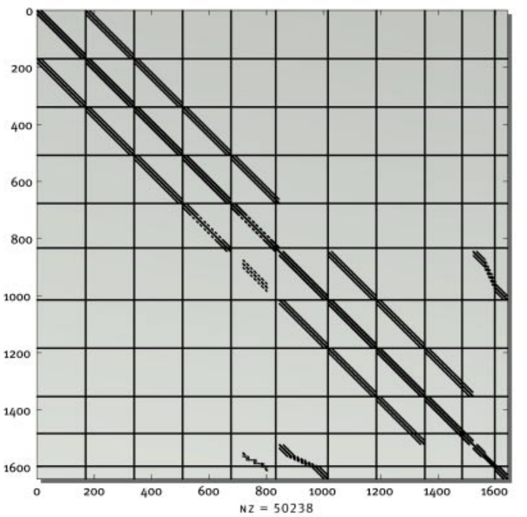


FIGURE 1
A Posteriori discovered block structure of a sparse matrix

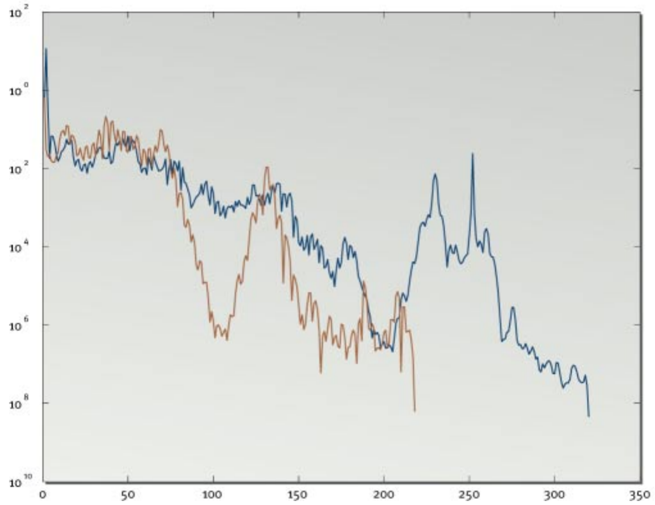


FIGURE 2
Convergence history with partitioning based on even division into block rows (blue) and block structure based division (red)

implemented in the PETSc toolkit and is undergoing testing on several parallel machines. Since different machines may exhibit a preference for differing methods, we are also developing a dynamically optimizing framework where a switch between methods can be made based on runtime measurements.

PERFORMANCE OPTIMIZATION OF MULTI-GRID SMOOTHERS

The main determinant of multi-grid performance is the smoother, accounting for up to 85% of the total work. Smoothers, typically SSOR iteration, are also one of the few sparse kernels of any kind that have appreciable reuse of data. Previous approaches have been aimed at explicitly fitting the data to the processor cache. We are using recursive division of the data, which will automatically accommodate caches on all levels. For the optimization of the in-cache operation, we use the techniques that have been developed previously in ATLAS.

SUPERNOVA INITIATIVE

We are part of a collaboration led by Anthony Mezzacappa of the Oak Ridge National Lab for the simulation of core-collapse supernovae. This problem leads to sparse systems with a component of dense calculations: on each point of the space discretisation there is full coupling between all angles and energy groups of interacting neutrinos. Solving the transport and interaction equations by preconditioning in an ADI-like manner has proved successful, but this still requires full solution of the dense diagonal blocks. We will investigate various approaches to diminish this cost, for instance by compressing the blocks using wavelet transforms.

In another approach to this problem, people have used sparse approximate inverses based on least-squares approximation. We intend to extend this approach, using a multi-color variant of a sparse inverse technique that mimics factorization techniques.

ITERATIVE METHODS IN NETSOLVE, GRADS, AND HARNESS

We have written simple interfaces between NetSolve and a number of iterative and direct solver libraries. This makes it possible for users with limited access to high-performance platforms, for example running MATLAB on a laptop, to use sophisticated parallel libraries such as PETSc.

For the Grads project we have written an iterative method with performance model. This will be extended to a multi-level preconditioner that reflects the cluster-like nature of the grid. We are also writing a demo of a fault-tolerant, time-stepping method, integrating FT-MPI, IBP, and PETSc in a HARNESS framework. The application will be the time solution of a heat equation, where we extend the PETSc time-stepper with checkpointing saved through IBP and a roll-back mechanism based on FT-MPI.

RELATED URLS

ATLAS - http://icl.cs.utk.edu/atlas/

MATLAB - http://www.mathworks.com/products/matlab/

Netlib – http://www.netlib.org/

PETSc - http://www-fp.mcs.anl.gov/petsc/

RECENT PUBLICATIONS

Dongarra, J., Eijkhout, V., van der Vorst, H., "An Iterative Solver Benchmark," *Scientific Programming*, Vol. 9, Number 2, 2001.

TOP500/TOP100

SUPERCOMPUTERS CLUSTERS

DEVELOPMENT TEAM
Jack Dongarra

WEB SITE
http://icl.cs.utk.edu/top500/
http://clusters.top500.org/

E-MAIL
top500@cs.utk.edu

COLLABORATORS
Hans Meuer
UNIVERSITY OF MANNHEIM
Horst Simon
LAWRENCE BERKELY
NATIONAL LABORATORY
Erich Strohmaier
LAWRENCE BERKELY
NATIONAL LABORATORY

Back in 1993, the list of the top 500 supercomputer sites worldwide was made available for the first time. Every year since, the TOP500 list has been published bi-annually. The best LINPACK Benchmark performance is used as a performance measure in ranking the computers. The list allows a detailed and well-founded analysis of the state of high performance computing (HPC). Previously, data such as the number and geographical distribution of supercomputer installations were difficult to obtain and only a few analysts undertook such an effort by tracking press releases of dozens of vendors. Data for the TOP500 are submitted by manufacturers of high performance computing systems as well as from users and managers of sites owning such systems. These submissions are reviewed by the TOP500 maintainers and by independent reviewers to ensure consistent, high quality data. With the TOP500 report now easily available, it is possible to present an analysis of the state of HPC.

While many aspects of the HPC market change dynamically over time, the evolution of performance seems to follow some empirical laws such as Moore's law. The TOP500 provides an ideal data basis to verify such an observation. By examining the computing power of the individual machines present in the TOP500 and the evolution of the total installed performance, we plot the performance of the systems at positions one, ten, 100 and 500 in the list as well as the total accumulated performance. In figure 1, the curve of position 500 shows, on average, an increase of a factor of two within one year. All other curves show a growth rate of 1.8 (+/- 0.07) per year.

Based on the current TOP500 data, which cover the last eight years, and the assumption that the current performance development continues for the near future, we can now extrapolate the observed performance and compare these values with the goals of the DOE ASCI program in the USA and the Earth Simulator project in Japan. In figure 2, we extrapolate the observed performance values using linear regression on the logarithmic scale. In other words, we fit exponential growth to all levels of performance in the TOP500. These simple interpretations of the data show surprisingly consistent results. Based on the extrapolation from these interpretations, we can expect to have the first ~100 TFlop/s systems by the year 2005, which is about one to two years later than the ASCI path projected plans. Conversely, by the year 2005 we would also expect that no system smaller than ~1 TFlop/s will be able to make the TOP500.

Looking even further into the future, we could speculate that, based on the current doubling of performance every year, the first Petaflop system would be available around 2010. Currently, due to the rapid changes in the technologies used in HPC systems, there is no reasonable projection possible for the architecture of a Petaflop system at the end of the next decade. Even as the HPC market has changed quite substantially since the introduction of the Cray 1 three decades ago, there is no end in sight for such rapid cycles of redefinition.

In an effort to continue the analysis of trends in the HPC area, we have initiated a study of the top 100 clusters and we are planning to rank the top 100 cluster sites. This initially will take the same form as the TOP500 list. There is also a plan to change the performance metric after some experience is gained with the existing set of systems. The new metric will include floating point performance, communication speed, as well as I/O performance.

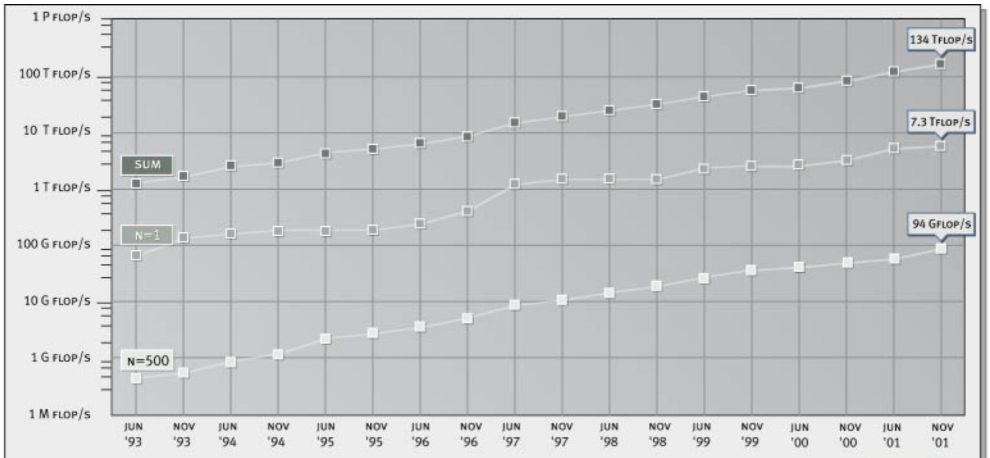


FIGURE 1
Chart depicting the overall growth of accumulated and individual performance as seen in the TOP500

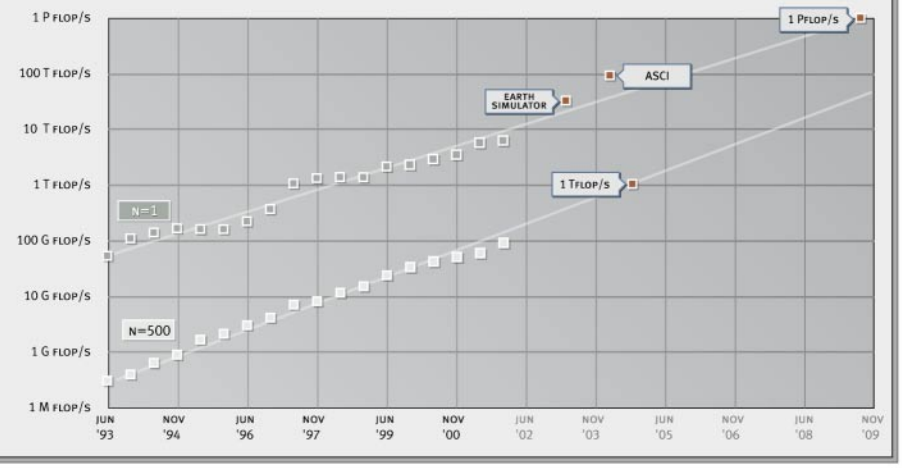


FIGURE 2
Chart depicting the extrapolation of recent growth rates of performance seen in the TOP500

TABLE 1. TOP 5 EXCERPT FROM NOVEMBER 2001 TOP500 SUPERCOMPUTERS

RANK	MANUFACTURER	COMPUTER	RMAX (GFLOPS)	INSTALLATION SITE	PROCESSORS
1	IBM	ASCI White, SP Power3 375 MHz	7226	Lawrence Livermore National Laboratory	8192
2	Compaq	AlphaServer SC ES45/1 GHz	4059	Pittsburgh Supercomputing Center	3024
3	IBM	Power3 375 MHz 16 way	2526	NERSC/LBNL	2528
4	Intel	ASCI Red	2379	Sandia National Labs	9632
5	IBM	ASCI Blue-Pacific SST, IBM SP 604e	2144	Lawrence Livermore National Laboratory	5808

TABLE 2. TOP 5 EXCERPT FROM CURRENT TOP100 CLUSTERS

RANK	INSTALLATION SITE	INTEGRATOR	PEAK PERFORMANCE	PROCESSORS	INTERCONNECT
1	Locus Discovery	Western Scientific	1416	1416	Fast Ethernet
2	Inpharmatica Ltd.	Inpharmatica Ltd.	1061	1220	Fast Ethernet
3	Shell Technology Exploration and Production	IBM	1030	1038	Gigabit Ethernet
4	NCSA	IBM	1032	1032	Myrinet 2000
5	Brookhaven National Laboratory	VA Linux and IBM	991	1276	Fast Ethernet

RELATED URLS

DOE ASCI - http://www.asci.doe.gov/ovrview/overview.htm

Earth Simulator Project - http://www.gaia.jaeri.go.jp/

FAQ

http://www.netlib.org/utk/people/JackDongarra/faq-linpack.html

TORC

TENNESSEE OAK RIDGE CLUSTER

DEVELOPMENT TEAM
ALPHABETICAL

Jack Dongarra

Brett Ellis

Paul Peltz

WEB SITE

http://icl.cs.utk.edu/torc/

E-MAIL

torc@cs.utk.edu

COLLABORATORS

Grid Application Development System (GrADS)

Hewlett-Packard

Intel

Microsoft Research

Myricom

Oak Ridge National Laboratory

Scali

University of Tennessee

AGENCY FUNDING

National Science Foundation

The Tennessee Oak Ridge Cluster (TORC) is a two-part cluster environment consisting of high-speed, low latency interconnects. It is primarily comprised of commodity hardware and software, and its purpose is to provide a local, easily accessible computer resource for parallel computing, grid-based metacomputing, different network technologies, and research/development of software. The two-part cluster consists of a production portion and an experimental portion. TORC resides at both the University of Tennessee (UT) and Oak Ridge National Lab (ORNL), but only the UT portion is described below.

Our close working relationship with companies like Myricom, Scali, Giganet, Microsoft, Intel, and HP, make the cluster possible. The commodity part of the name is very important in its design. We want proof that with small amounts of funding, when compared to the price of a modern supercomputer, we could attain decent performance and stability measured by cost per flop.

The production cluster consists of homogeneous, highly stable machines for long runs as well as modeling and is accessible by ICL staff at all times. The production cluster consists of eight machines and is designed solely for computational purposes. It has the following configuration:

- Dual PIII 550 MHz Processors
- 512MB memory
- 9GB UltraWide SCSI
- RedHat Linux 6.2

The production cluster employs three interconnect types for a rich variety of networking experiences: Myrinet, Scali, and 100 Mbit Ethernet. There is also a head node called Torc0 that maintains a consistent state of software across the cluster. Common research software installed include ATLAS, BLACS, Globus, LAPACK, Legion, Matlab, MPICH, NetSolve, NWS, Petsc, PVM, ScaLAPACK, TotalView, and various compilers. The Repository in a Box (RIB) software toolkit is used to maintain an easily accessible software catalog for our users, and the production cluster is available to all members of the UT Computer Science (CS) Department as well as researchers from other universities and institutions.

The experimental portion of the cluster exists as a configurable group of machines that can be booted into multiple operating systems. This cluster includes Pentium IIs, IIIs, 4s, AMD Athlons, and an Itanium in both single and SMP processor arrangements. It consists of 12 additional machines that can be (by request) subdivided in any way to run any OS that runs on the I86 architecture. Currently, the default OS on most machines is Linux. Table 1 at right shows the various configurations of the machines that comprise the experimental cluster. The interconnects available consist of all of the ones included in the production clusters as well as several other kinds of Gigabit ether technology.

The experimental cluster is used to perform many of the same tasks as the production cluster. It also serves as a roll out area for new OSs. On the experimental cluster, we test both Windows and Unix operating systems for usability, stability, and development. It is very important that software produced at ICL, and within the CS department as a whole, has the widest possible test bed to ensure functionality. The experimental cluster provides this on a small scale. If unreserved, the experimental cluster is available to the same users of the production cluster. There are times, depending on need, that much of the experimental cluster is reserved for private use.

An additional benefit of the experimental cluster is the ability to test new machines. The architectures have included Compaq Alpha, Alpha Clones, and Sun Sparcs. Such testing has been extremely valuable from the viewpoint of both the software developer and systems administrator.

TORC is currently used in a variety of projects within ICL and the CS department. Ongoing research projects currently utilizing TORC include ATLAS, NetSolve, ScaLAPACK, GrADS, SInRG, and RIB. We have also used it to test LSF, Globus, Legion, and PBS for several funding institutions. The most obvious use for the cluster has been to test clustering technologies. It has helped us, and others, to understand the intricacies and problems inherent in creating a cluster of machines that is required to be very flexible and stable for users.

TABLE 1. TORC EXPERIMENTAL CLUSTER CONFIGURATION

QTY.	HARDWARE	RAM	STORAGE	OS
3	Pentium III 600MHz	256MB	9 GB EIDE Drive	Windows NT/Redhat Linux 6.2
2	Pentium II 450MHz	256MB	8GB SCSI Drive	Windows NT/Redhat Linux 6.2
3	Athlon 1.2 GHz	256MB	40 GB EIDE Drive	Redhat Linux 7.1
1	Pentium 4 1.7GHz	384MB	20 GB EIDE Drive	Windows 2000/RedHat Linux 7.1
1	Dual PIII 550MHz	512MB	9GB Ultrawide SCSI	Redhat Linux7.2
1	Pentium 5 1.5GHz	512MB	20 GB EIDE Drive	Windows 2000/RedHat Linux 7.1
1	PC164 Alpha Clone 533MHz EV56	256MB	2- 2GBEIDE Disk Drives	Redhat Linux 6.2

TABLE 2. ADDITIONAL CS/ICL CLUSTER CONFIGURATIONS

NAME	# OF MACHINES	PROCESSOR	MEMORY	OS	INTERCONNECT
GSC#1	16	Dual 450MHz UltraSparc II	512MB	Solaris2.7	1Gb Fiber
GSC#2	16	Dual PIII 500MHz	512MB	Debian Linux	1Gb Fiber
GSC#3	4	Quad PIII Xeon 550MHz	2GB	Windows 2000 Advanced Server	1Gb Fiber
GSC#4	32	Dual PIII 500MHz	512MB	Redhat Linux 7.1	Myrinet
Microsoft	16	Duall PIII 933MHz	512MB	Dual Redhat Linux 7.1/Windows 2000	1Gb Copper

RELATED URLS

Beowulf - http://www.beowulf.org/

Berkeley NOW Project - http://now.cs.berkeley.edu/

High Performance Virtual Machines Project - http://www-csag.ucsd.edu/projects/clusters.html

High Speed Networks and Parallel Applications Project - http://lhpca.univ-lyon1.fr/

Jazznet - http://math.nist.gov/jazznet/

UNIVERSITY OF TENNESSEE
CENTERS OF EXCELLENCE

The University of Tennessee is focused on becoming one of America's top 25 public research universities. In 2000 the university has embarked upon a five-year, $335 million Tennessee Plan for Academic Excellence, including nine new research centers of excellence, scholarships to keep the best and brightest students in Tennessee, and an infusion of dollars to improve academic programs and establish additional small centers of excellence.

The university's nine new research centers of excellence-five in Knoxville and four at the Health Science Center in Memphis-promise 1,000 new jobs and as many as 20 new companies for Tennessee. The centers represent a $280 million investment, including $56 million from the university and the state and the balance primarily from grants. Below is a list of each of these centers including their directors.

CENTERS BASED IN KNOXVILLE:

- Advanced Materials Center, Dr. Ward Plummer
- Center for Information Technology Research, Dr. Jack Dongarra
- Environmental Biotechnology Center, Dr. Gary Sayler
- Food Safety Center, Dr. Stephen P. Oliver and Dr. Ann Draughon
- Structural Biology Center, Dr. Engin Serpersu

CENTERS BASED IN MEMPHIS:

- Connective Tissues Diseases Center, Dr. Andrew H. Kang
- Genomics and Bioinformatics Center, Dr. Dan Goldowitz
- Neurobiology and Imaging of Brain Disease Center, Dr. S. T. Kitai
- Vascular Biology Center, Dr. Lisa Jennings

UT CENTER OF EXCELLENCE

CITR

CENTER FOR INFORMATION TECHNOLOGY RESEARCH

BACKGROUND

As one of the nine Centers of Excellence at the University of Tennessee, the Center for Information Technology Research (CITR) was established in the spring of 2001 in order to drive the growth and development of leading edge Information Technology Research (ITR) at the University. Information Technology Research (ITR) is a broad, cross-disciplinary area that investigates ways in which fundamental innovations in Information Technology affect and are affected by the research process.

The mission of CITR is to build up a thriving, well-funded community in basic and applied ITR at the University of Tennessee in order to help the University capitalize on the rich supply of research opportunities that now exist in this area. To carry out this mission CITR is implementing a two pronged strategy: First, CITR will invest in a diverse group of ITR laboratories, each one led by an established researcher or an emerging leader in some significant area of ITR. The model for this effort will be the ICL and will be led by ICL Director Dr. Dongarra who also serves as the director of CITR. Second, CITR will develop a complimentary set of university-wide programs that can serve to foster innovative research ideas in the University community, seed the creation of new CITR laboratories, and help the University exploit the broadest possible spectrum of ITR opportunities.

The initial program offers Challenge Grants to IT researchers who will be applying for agency funding in the near future. We view CG funding as an investment in the overall efforts of a CITR PI for a given year. Although each CG is correlated with a particular proposal, it can be used at the discretion of the PI. It is intended to support the PIs in various ways that may be important to the development of new ITR efforts and proposals: hiring a graduate student, relevant travel, purchase of special equipment, development of an early prototype, help in the proposal process, and so on.

CITR LABORATORIES

As mentioned, CITR will make strategic investments in several ITR Laboratories. Like ICL, each new CITR Lab will be developed not only for its potential in generating funded research, but also with an eye toward diversifying the range of research opportunities and funding sources that UT can address. Besides ICL, two additional CITR Laboratories have been named, Logistical Computing and Internetworking (LoCI) Lab run by Professors Micah Beck and Jim Plank from the Computer Science Department and The Institute for Environmental Modeling (TIEM) run by Professor Lou Gross from the Departments of Ecology and Evolutionary Biology and Mathematics.

hp workstation i2000

HARDWARE RESOURCES

Research in the many areas of high performance computing requires access to various hardware resources. As an academic research group, we take pride in our hardware assets, which consist of 42 desktop computers ranging from Dell Windows/Linux machines to Sun Sparc workstations. We have 25 machines currently configured as servers. In addition, we also have a host of parallel computing machines, which include high-end architectures such as two IBM Power 3s, two commodity based clusters consisting of 24 total Dual Intel based commodity machines, and a non-production cluster consisting of 12 machines of various configurations. We also have an SGI Octane and two Compaq Alphas. The most recent addition to our lab is a group of four Intel Itanium Processor based machines; two dual processor Itanium 800 MHz and two single processor 733 MHz machines. Our close working relationship with hardware vendors has been very beneficial due to the fact that over 49 of our desktops and servers were either donated or are on loan.

As part of the UT Computer Science Department, we also retain access to many other resources including several server class machines as well as four high performance computing (HPC) clusters. These clusters include 32 Dual PIII 500Mhz machines, 16 Dual PIII 550Mhz machines, 17 Sun Enterprise 220R 450MHz, four Quad PIII Xeon 500Mhz boxes, and a 14 processor Enterprise 4500 SMP. The clusters are arranged in the classic Beowulf cluster configuration in which machines are connected by low latency, high-speed network switches.

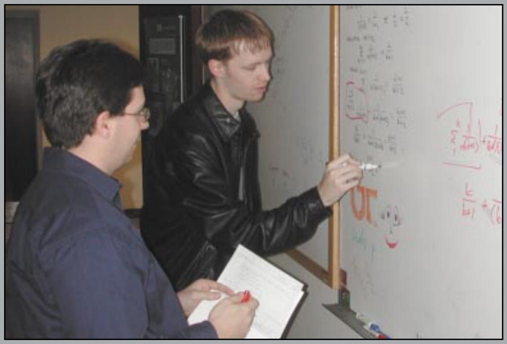
Because many of our staff frequently travel to conferences, workshops, and other events, it is important that they have access to mobile computing resources. To this end, we have acquired 38 laptops, which give us the ability to regularly provide demonstrations, tutorials, and presentations to our partners and funding agencies around the country.

In addition to the local resources at our disposal, we are also fortunate to have access to remote resources. Due to the many organizations and institutions with which we collaborate, a wide range of hardware architectures are made available to our research staff. Remaining in the forefront of computational research requires our staff to have access to the latest computing technology.

Our ability to harness the computing power of multiple architectures allows us to perform comprehensive software development and testing. In addition, we have the heterogeneous resources necessary to parallelize many applications that previously ran only sequentially.

NPACI and SInRG

Automatic optimization of parallel linear algebra software



PEOPLE

As is the case with most organizations, our employees drive our success. Equally important are the working relationships we have established with individuals and organizations within the high performance computing (HPC) community. Our staff, our partners and collaborators, and the many commercial vendors with which we work have helped us create a strong foundation for fostering creative, original research.

We have been very successful at attracting experts and top researchers that comprise our staff. With a large number of staff, we are able to apply adequate people resources to the projects on which we work. Currently, we employ 14 students, and 26 full or part-time staff, of which many worked for us as students themselves. Because ICL is known internationally as a leading HPC research group, we have been successful in attracting research scientists from around the world. Proudly, our staff includes representatives from many countries. Our ability to attract such experts from around the world is only one reason ICL remains an HPC research leader. Table 1 provides information about our current staff.

Equally important to our group are our students. As part of the computer science (CS) department of a large university, ICL has access to both graduate and undergraduate students. With a CS program consisting of nearly 200 students, additional help with our projects is just a job posting away. These students represent a resource that is not readily available to many research groups, and we have been very proactive in securing assistantships for those students who are motivated, hard working, and willing to learn. Table 2 provides a list of our current students.

In addition to our employees, we routinely host numerous visitors from around the globe. While many of our visitors stay briefly to give seminars or presentations, many remain with us for as long as a year collaborating, teaching, and learning. Though many of our visitors are professors from various international universities, we also host researchers and administrators from many research institutions. In addition, it is not uncommon to have students (undergraduate as well as graduate) from various universities study with us for months on end, learning about our approaches to computing problems. In fact, many Ph.D. students from universities as far away as Japan have passed through our doors in an effort to broaden their understanding of linear algebra techniques and how we apply them to our research. The experience shared between our visitors and ourselves has been extremely beneficial to us, and we will continue providing opportunities for visits from our international research colleagues. See Table 3 for the many guests who have stopped by in the last year to exchange ideas and share their expertise with us. We have worked hard to create and maintain many collaborative relationships and are always eager to open doors to new opportunities for sharing research endeavors.

As proud as we are of the many talented individuals that have worked for us over the years, we are equally proud that many of our former students and staff have moved on to do further interesting and useful research at places around the world. Many of our alumni have gone on to apply what they have learned at ICL with companies such as Hewlett-Packard, Hitachi, IBM, Inktomi, Intel, Microsoft, NEC, SGI, Sun Microsystems, and many others. Table 5 provides a list of past students, staff, and visitors to our group.

TABLE 1. CURRENT ICL STAFF

NAME	POSITION	DEGREE	PROJECTS
Bukovsky, Antonin	Research Associate	BS - University of Tennessee, 2000	FT-MPI, HARNESS
Cronk, David	Post-Doctoral Research Associate	PhD - College of William & Mary, 1998	Parallel I/O
Dongarra, Jack	ICL Director	PhD - University of New Mexico, 1980	
Eijkhout, Victor	Research Assistant Professor	PhD - Catholic University, Nijmegen, Netherlands, 1990	NetSolve, Sparse Matrix Algorithms & Software
Ellis, Brett	Senior Computer Systems Specialist	BS - University of Tennessee, 1996	GrADS, SInRG, TORC
Fagg, Graham	Research Assistant Professor	PhD - University of Reading, UK, 1998	FT-MPI, HARNESS, MPI_Connect, Parallel I/O
Finchum, Teresa	Administrative Services Assistant		
Fike, Don	Research Associate	BS - Illinois St. University, 2001	NHSE, RIB
Garner, Nathan	Research Consultant	MS - University of Tennessee, 1999	NetSolve
Jones, Jan	Publications Coordinator	MS - University of Tennessee, 1990	
Lee, Tracy	Senior Budget Assistant	BA - University of Tennessee, 1995	
London, Kevin	Research Associate		MPI_Connect, PAPI
Millar, Jeremy	Research Associate	BS - University of Tenneseee, 2000	HARNESS, I2-DSI, FT-MPI, NA-Net, NA-Digest, Netlib, NHSE, RIB
Miller, Michelle	Research Associate	MS - University of Utah, 1998	NetSolve
Mitchell, Cindy	Principal Secretary		
Moore, Shirley	Associate Director for Research	PhD - Purdue University, 1990	NHSE, PAPI, RIB
Moore, Keith	Research Associate	MS - University of Tennessee, 1996	HARNESS, NA-Net, NA-Digest, Netbuild, NetSolve
Moore, Terry	Assoc. Director for Project Development	PhD - University of North Carolina, Chapel Hill,1993	I2-DSI, IBP, NetSolve, RIB, SInRG
Mucci, Phil	Research Consultant	MS - University of Tennesseee, 1998	PAPI
Peltz, Paul	Computer Operations/Systems Programmer	BA expected 2002	TORC
Rafferty, Tracy	Manager		
Rogers, David	Graphic Artist	BFA - University of Tennessee, 2000	
Seymour, Keith	Research Associate	MS - University of Tennessee, 1997	f2j, PAPI
Terpstra, Dan	Post-Doctoral Research Associate	PhD - Florida State University, 1981	PAPI
Wells, Scott	Assistant Director for Communications	MS - University of Tennessee, 1996	NHSE, RIB
Whaley, R. Clint	Senior Research Associate	MS - University of Tennessee, 1994	ATLAS, BLAST, ScaLAPACK
YarKhan, Asim	Senior Research Associate	MS - Penn State, 1994	GrADS

TABLE 2. CURRENT STUDENTS

NAME	POSITION	PROJECTS
Agrawal, Sudesh	Graduate Research Assistant	NetSolve
Chen, Zizhong	Graduate Research Assistant	Sparse Matrix Algorithms & Software
Ding, Jun	Graduate Research Assistant	Sparse Matrix Algorithms & Software
Downey, Andrew	Graduate Research Assistant	f2j, Parkbench
Drum, Brian	Undergraduate Student	NetSolve
Luszczeck, Piotr	Graduate Research Assistant	Sparse Matrix Algorithms & Software
Roche, Ken	Graduate Research Assistant	Sparse Matrix Algorithms & Software, GrADS
Sagi, Kiran	Graduate Research Assistant	NetSolve
Shahnaz, Farial	Undergraduate Student	FT-MPI, Netlib
Shi, Zhiao	Graduate Research Assistant	NetSolve
Thomas, Joe	Graduate Research Assistant	Systems Support
Vadhiyar, Sathish	Graduate Research Assistant	FT-MPI, GrADS, NetSolve
Walters, Michael	Undergraduate Student	Netlib, NHSE
Zhou, Luke	Graduate Research Assistant	PAPI

TABLE 3. RECENT VISITORS

NAME	DATES VISITED	FROM
Canovas, Domingo Gimenez	September-October 2001	Universidad de Murcia - Murcia, Spain
Cuenca, Javier	September-October 2001	Universidad de Murcia - Murcia, Spain
Flemming, Rey	April 2001	United Devices - Texas, USA
Gabriel, Edgar	July – August 2001	Rechenzentrum Universitat - Stuttgart, Germany
Gelas, Jean Patrick	July 2001	RESAM Laboratory Univ. Claude Bernard - Lyon, France
Gentzech, Wolfgang	May 2001	Sun Microsystems - California, USA
Golub, Gene	November 2001	Stanford University
Giuseppe, Bruno	July 2001	Banca d'Italia - Rome, Italy
Hammarling, Sven	November 2000	The Numerical Algorithms Group Ltd. - Oxford, England
Hiroyasu, Tomo	July 2001	Doshisha University - Kyoto, Japan
Jeannot , Emmanuel	February 2001	ENS – Ecole Normale de Superieure - Lyon, France
Johansson, Patrik	February 2001	Lokomo Systems AB - Danderyd, Sweden
Lee, Dong Woo	June 2000 - February 2001	Kwangju Institute of Science and Technology - Kwangju, South Korea
Nilsson, Jonas	September 2001	Uppsala University - Uppsala, Sweden
Quinson, Martin	February 2001	ENS – Ecole Normale de Superieure - Lyon, France
Soendergaard, Peter	September 2001	Danish Technical University - Lyngby, Denmark
Wasniewski, Jerzy	November 2000	Danish Technical University - Lyngby, Denmark

Table 4. Graduated Students of Jack Dongarra

Name	Position	Name	Position
Barrett, Richard	MS Computer Science (University of Tennessee), 1994	McMahan, Paul	MS Computer Science (University of Tennessee), 1999
Blackford, Susan	MS Computer Science (University of Tennessee), 1990	Moulton, Steve	MS Computer Science (University of Tennessee), 1992
Casanova, Henri	PhD Computer Science (University of Tennessee), 1998	Mucci, Phil	MS Computer Science (University of Tennessee), 1998
Duitz, Mitchell	MS Computer Science (University of Tennessee), 1991	Nypaver, Delphy	MS Computer Science (University of Tennessee), 1997
Garner, Nathan	MS Computer Science (University of Tennessee), 1999	Payne, James	MS Computer Science (University of Tennessee), 1994
Green , Stan	MS Computer Science (University of Tennessee), 1994	Petitet, Antoine	PhD Computer Science (University of Tennessee), 1996
Ho, George	MS Computer Science (University of Tennessee), 1999	Phillips, Rick	MS Computer Science (University of Tennessee), 1998
Kalhan, Ajay	MS Computer Science (University of Tennessee), 1996	Raman, Ganapathy	MS Computer Science (University of Tennessee), 2000
Kim, Youngbae	PhD Computer Science (University of Tennessee), 1996	Sidani, Majed	PhD Mathematics (University of Tennessee), 1992
LaRose, Brian	MS Computer Science (University of Tennessee), 1993	Whaley, Clint	MS Computer Science (University of Tennessee), 1993
Liebrock, Lorie	PhD Computer Science (Rice University), 1994	Xu, Tinghua	MS Computer Science (University of Tennessee), 2000
Manchek, Robert	MS Computer Science (University of Tennessee), 1994		

Table 5. Former Staff, Students, and Resident Visitors of ICL

Name	Position with ICL and Dates	Name	Position with ICL and Dates
Aebischer, Carolyn	Graduate Student, 1990-1993	Choi, Jaeyoung	Post Doc, 1994-1996
Anderson, Ed	Research Associate, 1989-1991	Clarkson, Eric	Artist, 1998-1999
Arnold, Dorian	Research Associate, 1999-2001	Cleary, Andy	Post Doc, 1995-1997
Bai, Zhaojun	Post Doc, 1990-1992	Cox, Jason	Graduate Student, 1993-1997
Barrett, Richard	Graduate Research Assistant, 1992-1994	Deane, Cricket	Research Associate, 1998-1999
Bassi, Alex	Research Associate, 2000-2001	Desprez, Frederic	Post Doc, 1994-1995
Beck, Micah	Research Associate Professor, 1998-2001	Do, Martin	Graduate Research Assistant, 1993-1994
Beguelin, Adam	Post Doc, 1991-1994	Doolin, David	Research Associate, 1997
Benzoni, Annamaria	Visiting Research Associate 1991	Dong, Leon	Graduate Research Assistant, 200-2001
Betts, Scott	Undergraduate Student, 1997-1998	Drake, Mary	Office Supervisor, 1989-1992
Blackford, Susan	GRA, Research Associate, 1989-2001	Eyler-Walker, Zach	Undergraduate Student, 1998
Bond, Leon	Principal Secretary, 1999-2000	Fischer, Markus	Visiting Student, 1997-1998
Brown, Randy	Undergraduate Student, 1997-1999	Fuentes, Erika	Graduate Research Assistant, 2000-2001
Browne, Cynthia	Undergraduate Student Assistant, 2001	Gangwer, Lynn	Principal Secretary, 2000-2001
Browne, Murray	Research Associate, 1998-1999	Gangwer, Tracy	
Bunch, Greg		Gettler, Jonathan	Graduate Research Assistant, 1996
Casanova, Henri	GRA, Post Doc, 1995-1998	Greaser, Eric	Graduate Research Assistant, 1993
Chambers, Sharon	Undergraduate Student, 1998-2000	Green, Stan	GRA, Senior Research Associate, 1992-1996

TABLE 5. FORMER STAFF, STUDENTS, AND RESIDENT VISITORS OF ICL CONTINUED

NAME	POSITION WITH ICL AND DATES	NAME	POSITION WITH ICL AND DATES
Hagewood, Hunter	Graduate Research Assistant, 2000-2001	Moulton, Steven	Graduate Research Assistant, 1991-1993
Halloy, Christian	Associate Director, 1996-1997	Newton, Peter	Post Doc, 1994-1995
Hammarling, Sven	Visiting Professor, 1996-1997	Papadopoulos, Caroline	Graduate Student, 1997-1998
Hasegawa, Hidehiko	Visiting Research Associate 1995–1996	Petitet, Antoine	GRA, Post Doc, Research Scientist, 1993-2001
Hasegawa, Satomi	Visiting Research Associate 1995-1996	Pozo, Roldan	Post Doc, 1992-1994
Hastings, Chris	Research Associate, 1996	Race, Tammy	Graduate Research Assistant, 1999-2001
Henderson, David	Undergraduate Student, 1999-2001	Raman, Ganapathy	Graduate Research Assistant, 1998-2000
Henry, Greg	Post Doc, 1996-1996	Robert, Yves	Visiting Professor, 1996-1997
Hill, Sid	Undergraduate Student, 1996-1998	Rothrock, Tom	Undergraduate Student, 1998
Ho, George	GRA, Research Consultant, 1998-2000	Rowan, Tom	Collaborating Scientist, 1993-1997
Horner, Jeff	Undergrad., Research Associate, 1995-1999	Sams, Evelyn	Principal Secretary, 1998-1999
Huang, Yan	Graduate Research Assistant, 2000-2001	Sidani, Majed	GRA, Post Doc, 1990-1992
Jacobs, Paul	Undergraduate Student, 1992-1995	Singhal, Shilpa	Undergraduate Student, 1996-1998
Ji, Weizhong	Graduate Research Assistant, 1999-2000	Spencer, Thomas	Undergraduate Student, 2000-2001
Jiang, Weicheng	Post Doc, 1992-1995	Strohmaier, Erich	Post-Doctoral Research Associate, 1995-2001
Jin, Song	Graduate Student, 1998	Swany, Martin	Research Associate, 1996-1999
Kannan, Balajee	Graduate Research Assistant, 2000-2001	Talley, Judi	Sr. Computer Systems Specialist, 1993-1999
Kim, MyungHo	Visiting Scholar, 1998	Teranishi, Keita	Undergraduate Student, GRA, 1998
Kim, Youngbae	Graduate Research Assistant, 1992-1996	Thurman, John	Graduate Student, 1998-1999
Kolatis, Michael	GRA, Research Associate, 1993-1996	Tisseur, Françoise	Post Doc, 1997
Letsche, Todd	Graduate Student, 1993-1994	Tourancheau, Bernard	Post Doc, 1993-1994
Lewis, Sharon	GRA, Manager, 1992-1995	van de Geijn, Robert	Post Doc, 1990-1991
Li, Xiang	Graduate Research Assistant, 2001	Venckus, Scott	Graduate Research Assistant, 1993-1995
Liu, Chaoyang	Graduate Research Assistant, 2000	Wade, Reed	Research Associate, 1990-1996
Longley, Matt	Undergraduate Student, 1999	Wo, Susan	Graduate Research Assistant, 2000-2001
Luczak, Richard	ASC MSRC Programming Tools onsite Lead, 2000-2001	Xu, Tinghua	Graduate Research Assistant, 1998-2000
Manchek, Robert	Research Associate 1990-1996	Yang, Tao	Graduate Research Assistant, 1999
McMahan, Paul	GRA, Program Director, 1994-2000	Zheng, Yong	Graduate Research Assistant 2001

PARTNERSHIPS

Over the years, ICL has enjoyed many mutually beneficial working relationships with institutions all over the globe. The high performance computing (HPC) community consists of academic institutions, research centers, branches of the federal government, and various other public and private organizations. As a research group and a member of this community, ICL shares many of the same interests with numerous other institutions. Much of our growth has been in large part due to these relationships. Our ability to collaborate with such institutions has strengthened our research efforts by allowing us to share resources, both material and intellectual. Table 1 highlights many of our domestic partners and collaborators. In addition to our many US government and academic partners, we have also enjoyed a strong working relationship with many commercial software vendors as well as many international HPC research centers and organizations. Included in the list of software vendor collaborators are Etnus, Inc., developer of the TotalView debugger; Kuck and Associates, Inc., developer of the KAP/Pro toolset; and Pallas, developer of the Vampir performance visualization and analysis tool.

Figure 1 shows the geographical location of many of the domestic and international partners and collaborators in research with whom we continue to work.

TABLE 1. DOMESTIC PARTNERS AND COLLABORATORS

Argonne National Laboratory

California Institute of Technology Center for Advanced Computing Research (CACR)

Defense Advanced Research Projects Agency (DARPA)

Department of Defense (DoD)

Department of Energy (DOE)

Emory University

Information Sciences Institute (ISI)

Intel

International Business Machines (IBM)

Internet2

Joint Institute for Computational Science (JICS)

Lawrence Berkely National Laboratory

Lawrence Livermore National Laboratory (LLNL)

Los Alamos National Laboratory (LANL)

Microsoft Research

National Aeronautics and Space Administration (NASA)

National Computational Science Alliance (NCSA)

National HPCC Software Exchange (NHSE)

National Institute of Standards and Technology (NIST)

National Science Foundation (NSF)

National Partnership for Advanced Computational Infrastructure (NPACI)

Oak Ridge National Laboratory (ORNL)

Rice University

San Diego Supercomputing Center (SDSC)

Silicon Graphics Incorporated (SGI)

Sun Microsystems

University of California, Berkeley

University of California, San Diego

University of Kentucky

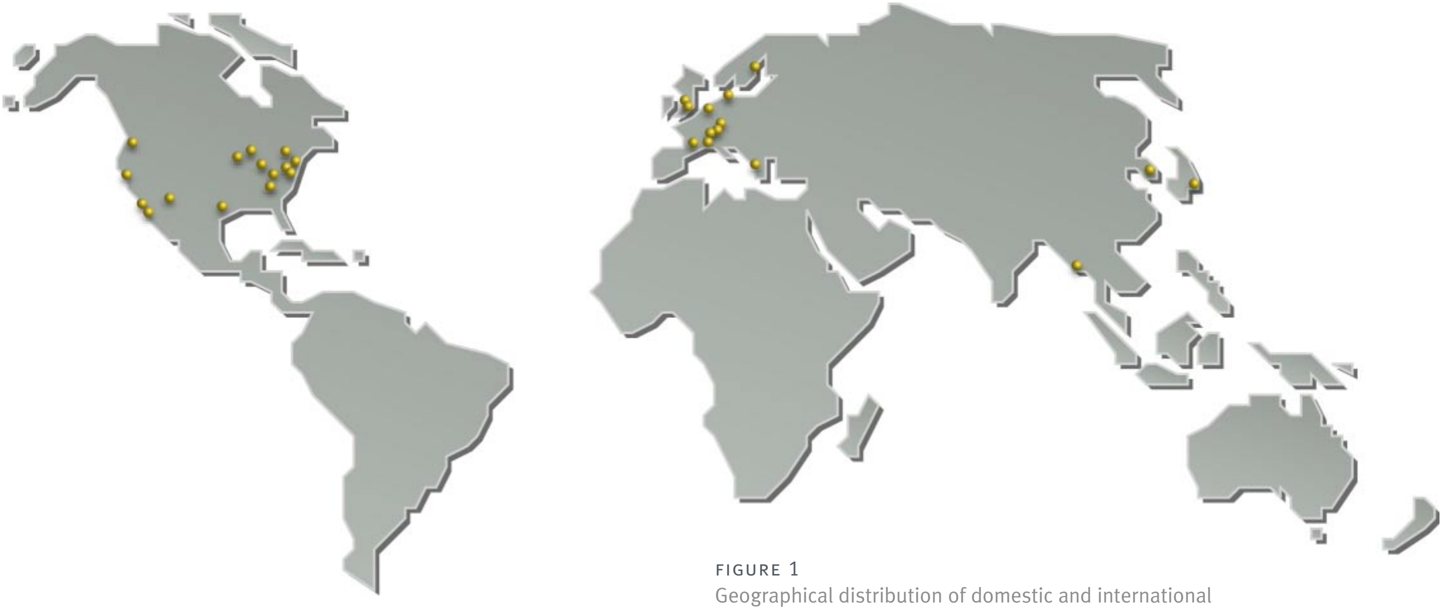


FIGURE 1
Geographical distribution of domestic and international partners and collaborators

TABLE 2. INTERNATIONAL PARTNERS AND COLLABORATORS

Danish Computing Centre for Research and Education - Lyngby, Denmark - http://lawra.uni-c.dk/

Department of Mathematical and Computing Sciences Tokyo Institute of Technology - Tokyo, Japan - http://matsu-www.is.titech.ac.jp/

Department of Mathematics, University of Manchester - Manchester, England - http://www.maths.man.ac.uk/

Electrotechnical Laboratory, Computer Systems Division - Tsukuba, Japan - http://phase.etl.go.jp/

European Centre for Research and Advanced Training in Scientific Computing (CERFACS) - Toulouse, France - http://www.cerfacs.fr/

Fakultät für Mathematik und Informatik, Universität Mannheim - Mannheim, Germany - http://www.uni-mannheim.de/

Institut für Wissenschaftliches Rechnen, ETH Zentrum - Zürich, Switzerland - http://www.inf.ethz.ch/

Istituto per le Applicazioni del Calcolo "Mauro Picone" del C.N.R. - Rome, Italy - http://www.iac.rm.cnr.it/

Kasetsart University - Bangkok, Thailand - http://smile.cpe.ku.ac.th/

Laboratoire de l'Informatique du Parallelisme, École Normale Superieure de Lyon - Lyon, France - http://www.ens-lyon.fr/

Laboratoire Réseaux Haut Débits et Support d'Applications Multimedia (RESAM) Jeune Equipe de l'Université Claude Bernard de Lyon- Lyon, France - http://lhpca.univ-lyon1.fr/

Mathematical Institute, Utrecht University - Netherlands - http://www.math.uu.nl/

The Numerical Algorithms Group Ltd. - Oxford, England - http://www.nag.co.uk/

Rutherford Appleton Laboratory - Oxford, England - http://www.rl.ac.uk/

Scole Polytechnique Federale de Lausanne - Lausanne, Switzerland - http://capawww.epfl.ch/

Soongsil University - Seoul, South Korea - http://www.soongsil.ac.kr/english/

University of Umeå - Umeå, Sweden - http://www.umu.se/umu/index_eng.html

PUBLICATIONS 2000-2001

2001 PUBLICATIONS

Alexandrov, V., Dongarra, J., Juliano, B., Renner, R., Tan, K. (Eds.) *Proceedings of Computational Science - ICCS 2001 San Francisco, CA. Lecture Notes in Computer Science*, Vol. 2073 and 2074 (Berlin: Springer Verlag), 2001.

Arnold, D., Agrawal, S., Blackford, S., Dongarra, J., Miller, M., Vahdiyar, S., Sagi, K., Shi, Z. “Users’ Guide to NetSolve v1.4,” University of Tennessee Computer Science *Technical Report*, UT-CS-01-467, 2001.

“Basic Linear Algebra Subprograms Technical (BLAST) Forum Standard”, to appear in *International Journal of High Performance Computing Applications*, Vol. 16 (May and November 2002).

Arnold, D., Dongarra, J. “Developing an Architecture to Support the Implementation and Development of Scientific Computing Applications,” Boisvert, R., Tang, P. (Eds). *The Architecture of Scientific Software*, (Norwell, MA: Kluwer Academic Publishers), 2001: 39-56.

Arnold, D., Vadhiyar, S., Dongarra, J. “On the Convergence of Computational and Data Grids,” *Parallel Processing Letters*, Vol. 11, Numbers 2 and 3 (October 2001): 1479-1496.

Barker, V., Blackford, S., Dongarra, S., Du Croz, J., Hammarling, S., Marinova, M., Wasniewski, J., Yalamov, P. *LAPACK95 Users’ Guide* (Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM) Publications), 2001.

Bassi, A., Beck, M., Plank, J., Wolski, R. “Internet Backplane Protocol: API 1.0,” University of Tennessee Computer Science *Technical Report*, UT-CS-01-455, 2001.

Bassi, A., Li, X. “Internet Backplane Protocol - Test Language v. 1.0,” University of Tennessee Computer Science *Technical Report*, UT-CS-01-464, 2001.

Beck, M., Arnold, D., Bassi, A., Berman, F., Casanova, H., Dongarra, J., Moore, T., Obertelli, G., Plank, J., Swany, M., Vadhiyar, S., Wolski, R. “Logistical Computing and Internetworking: Middleware for the Use of Storage in Communication,” *Third Annual International Workshop on Active Middleware Services (AMS)*, San Francisco, CA, August, 2001.

Beck, M., Moore, T., Abrahamsson, L., Achouiantz, C., Johansson, P. “Enabling Full Service Surrogates Using the Portable Channel Representation,” *Tenth International World Wide Web Conference Proceedings* (to appear), Hong Kong, May 1-5, 2001.

Beck, M., Moore, T., Plank, J. “Exposed vs. Encapsulated Approaches to Grid Service Architecture,” *2nd International Workshop on Grid Computing*, Denver, CO, Nov. 12, 2001.

Berman, F., Chien, A., Cooper, K., Dongarra, J., Foster, I., Gannon, D., Johnsson, L., Kennedy, K., Kesselman, C., Mellor-Crummey, J., Reed, D., Torczon, L., Wolski, R. “The GrADS Project: Software Support for High-Level Grid Application Development,” *International Journal of High Performance Applications and Supercomputing*, Vol. 15, Number 4 (Winter 2001): 327-344.

Blackford, S., Demmel, J., Dongarra, J., Duff, I., Hammarling, S., Henry, G., Heroux, M., Kaufman, L., Lumsdaine, A., Petitet, A., Pozo, R., Remington, K., Whaley, C. “Basic Linear Algebra Subprograms (BLAS),” (an update), submitted to *ACM TOMS*, February 2001.

Casanova, H., Matsuoka, S., Dongarra, J. “Network-Enabled Server Systems: Deploying Scientific Simulations on the Grid, “ *2001 High Performance Computing Symposium (HPC’01)*, part of the *Advance Simulation Technologies Conference*, Seattle, Washington, April 22-26, 2001.

Cronk, D., Fagg, G., Moore, S. “Parallel I/O for EQM Applications,” *Department of Defense Users’ Group Conference Proceedings* (to appear), Biloxi, Mississippi, June 18-21, 2001.

Dongarra, J. “Performance of Various Computers Using Standard Linear Equations Software (LINPACK Benchmark Report),” University of Tennessee Computer Science *Technical Report*, CS-89-85, 2001.

Dongarra, J., Eijkhout, V., van der Vorst, H., “An Iterative Solver Benchmark,” *Scientific Programming*, Vol. 9, Number 2, 2001.

Dongarra, J., London, K., Moore, S., Mucci, P., Terpstra, D. “Using PAPI for Hardware Performance Monitoring on Linux Systems,” *Conference on Linux Clusters: The HPC Revolution*, Urbana, Illinois, June 25-27, 2001. (to appear)

Dongarra, J., Moore, S., Trefethen, A., "Numerical Libraries and Tools for Scalable Parallel Cluster Computing," *International Journal of High Performance Applications and Supercomputing*, Vol. 15, Number 2 (Summer 2001): 175-180.

Dongarra, J., Meuer, H., Simon, H., Strohmaier, E. "High Performance Computing Trends," *Hellenic European Research on Mathematics and Information Science*, Vol. 2, 2001: 155-163.

Dongarra, J., Walker, D. "The Quest for Petascale Computing," *Computing in Science and Engineering*, May/June 2001: 22-29.

Eijkhout, V. "Automatic Determination of Matrix-Blocks," LAPACK Working Note 151, University of Tennessee Computer Science *Technical Report*, UT-CS-01-458, 2001.

Fagg, G., Bukovsky, A., Dongarra, J. "Fault Tolerant MPI for the HARNESS Meta-Computing System," Alexandrov, V., Dongarra, J., Juliano, B., Renner, R., Tan, K. (Eds.) In *Proceedings of International Conference of Computational Science - ICCS 2001*, San Francisco, CA. *Lecture Notes in Computer Science*, Vol. 2073 (Berlin: Springer Verlag), 2001: 355-366.

Fagg, G., Bukovsky, A., Dongarra, J. "HARNESS and Fault Tolerant MPI," *Parallel Computing*, Vol. 27, Number 11 (October 2001): 1479-1496.

Fagg, G., Gabriel, E., Resch, M, Dongarra, J. "Parallel IO Support for Meta-Computing Applications: MPI_Connect IO Applied to PACX-MPI," *EuroPVM/MPI Conference* (to appear), Greece, September 23-26, 2001.

London, K., Dongarra, J., Moore, S., Mucci, P., Seymour, K., Spencer, T. "End-user Tools for Application Performance Analysis, Using Hardware Counters," *International Conference on Parallel and Distributed Computing Systems* (to appear), Dallas, TX, August 8-10, 2001.

London, K., Moore, S., Mucci, P., Seymour, K., Luczak, R. "The PAPI Cross-Platform Interface to Hardware Performance Counters", *Department of Defense Users' Group Conference Proceedings* (to appear), Biloxi, Mississippi, June 18-21, 2001.

Miller, M., Moulding, C., Dongarra, J. "Grid-Enabling an Interactive Simulation/Visualization Environment," *2001 High Performance Computing Symposium (HPC'01)*, part of the *Advance Simulation Technologies Conference*, Seattle, Washington, April 22-26, 2001.

PUBLICATIONS ON-LINE

Many ICL Publications listed here can be downloaded from http://icl.cs.utk.edu/publications.html

Miller, M., Moulding, C., Dongarra, J., Johnson, C. "Grid-Enabling Problem Solving Environments: A Case Study of SCIRUN and NetSolve," *Proceedings of the High Performance Computing 2001 Grand Challenges in Computer Simulation [HPC], Special Track on "High Performance Simulation Environments"*, Seattle, Washington, Apri 22-26, 2001.

Moore, K., Dongarra, J. "NetBuild," University of Tennessee Computer Science *Technical Report*, UT-CS-O1-461, 2001.

Moore, S., Arnold, D., Cronk, D. "Metacomputing Support for the SARA3D Structural Acoustics Application," *Department of Defense Users' Group Conference*, Biloxi, Mississippi, June 18-21, 2001. (to appear)

Moore, S., Cronk, D., London, K., Dongarra, J. "Review of Performance Analysis Tools for MPI Parallel Programs," *EuroPVM/MPI Conference* (to appear), Greece, September 23-26, 2001.

Palma, J., Dongarra, J., Hernandez, V. (Eds.) *Vector and Parallel Processing - VECPAR 2000*, *Lecture Notes in Computer Science*, Vol. 1981 (Berlin: Springer Verlag), 2001.

Petitet, A., Blackford, S., Dongarra, J., Ellis, B., Fagg, G., Roche, K., Vadhiyar, S. "Numerical Libraries and The Grid: The Grads Experiments with ScaLAPACK," *International Journal of High Performance Applications and Supercomputing*, Vol. 15, Number 4 (Winter 2001): 359-374.

Plank, J., Bassi, A., Beck, M., Moore, T., Swany, M., and Wolski, R. "The Internet Backplane Protocol: Storage in the Network," *Internet Computing*, Vol. 5, Number 5, 2001.

Seymour, K., Dongarra, J. "Automatic Translation of Fortran to JVM Bytecode," *Joint ACM Java Grande - ISCOPE 2001 Conference*, Stanford University, California, June 2-4, 2001.

Vadhiyar, S., Fagg, G., Dongarra, J. "Toward an Accurate Model for Collective Communications," Alexandrov, V., Dongarra, J., Juliano, B., Renner, R., Tan, K. (Eds.) In *Proceedings of International Conference on Computational Science - ICCS 2001*, San Francisco, CA. *Lecture Notes in Computer Science*, Vol. 2073 (Berlin: Springer Verlag), 2001: 41-50.

van der Steen, A., Dongarra, J. "Overview of High Performance Computers," *Handbook of Massive Data Sets* (to appear), Kluwer Academic Publishers, 2001.

Whaley, C., Petitet, A., Dongarra, J. "Automated Empirical Optimization of Software and the ATLAS Project," *Parallel Computing*, Vol. 27, Numbers 1-2 (May/June 2001): 3-25.

2000 PUBLICATIONS

Arnold, D., Bachmann, D., Dongarra, J. “Request Sequencing: Optimizing Communication for the Grid,” In *Lecture Notes in Computer Science: Proceedings of 6th International Euro-Par Conference 2000, Parallel Processing*, (Germany: Springer Verlag), 2000, Vol. 1900: 1213-1222.

Arnold, D., Blackford, S., Dongarra, J., Eijkhout, V., Xu, T. “Seamless Access to Adaptive Solver Algorithms,” In *Proceedings of 16th IMACS World Congress 2000 on Scientific Computing, Applications Mathematics and Simulation*, Lausanne, Switzerland, August 22, 2000.

Arnold, D., Browne, S., Dongarra, J., Fagg, G., Moore, K. “Secure Remote Access to Numerical Software and Computational Hardware,” In *Proceedings of the DoD HPC Users Group Conference (HPCUG) 2000*, Albuquerque, NM, June 7, 2000.

Arnold, D., Dongarra, J. “Developing an Architecture to Support the Implementation and Development of Scientific Computing Applications,” In *Proceedings of Working Conference 8: Software Architecture for Scientific Computing Applications*, Ottawa, Canada, October 2-6, 2000.

Arnold, D., Dongarra, J. “The NetSolve Environment: Progressing Towards the Seamless Grid,” *2000 International Conference on Parallel Processing (ICPP-2000)*, Toronto, Canada, August 21-24, 2000.

Arnold, D., Lee, W., Dongarra, J., Wheeler, M. “Providing Infrastructure and Interface to High-Performance Applications in a Distributed Setting,” In *Proceedings of High Performance Computing 2000*, April, 2000.

Bai, Z., Demmel, J., Dongarra, J., Ruhe, A., van der Vorst, H., (Eds.) *Templates for the Solution of Large Algebraic Eigenvalue Problems: A Practical Guide*, Philadelphia, PA: Society for Industrial and Applied Mathematics (SIAM) Publications, 2000.

Beck, M., Moore, T., Plank, J., and Swany, M. “Logistical Networking: Sharing More Than the Wires,” Hariri, S., Lee, C., Raghavendra, C. (Eds.) In *Active Middleware Services*, (Norwell, MA: Kluwer Academic), 2000.

Berman, F., Chien, A., Cooper, K., Dongarra, J., Foster, I., Gannon, D., Johnsson, L., Kennedy, K., Kesselman, C., Mellor-Crummey, J., Reed, D., Torczon, L., Wolski, R. “The GrADS Project: Software Support for High-Level Grid Application Development,” *International Journal of High Performance Computing Applications*, Winter 2001 (Volume 15, Number 4).

Browne, S., Dongarra, J., Garner, N., Ho, G., Mucci, P. “A Portable Programming Interface for Performance Evaluation on Modern Processors,” *The International Journal of High Performance Computing Applications*, Vol. 14, Number 3 (Fall 2000): 189-204.

Browne, S., Dongarra, J., Garner, N., London, K., Mucci, P. “A Scalable Cross-Platform Infrastructure for Application Performance Tuning Using Hardware Counters, “ *Proceedings of SuperComputing 2000 (SC’00)* Dallas, TX, November 2000.

Carpenter, B., Moore, K., Fink, B. “Routing IPv6 over IPv4,” *The Internet Protocol Journal*, Vol. 3, Number 1 (March 2000): 2-10.

Casanova, H., Plank, J., Beck, M., Dongarra, J. “Deploying Fault-tolerance and Task Migration with NetSolve,” To appear in *The International Journal on Future Generation Computer Systems*.

Cronk, D., Ellis, B., Fagg, G. “Metacomputing: An Evaluation of Emerging Systems,” University of Tennessee Computer Science Department *Technical Report*, July 2000. UT-CS-00-445.

D’Azevedo, E., Dongarra, J. “The Design and Implementation of the Parallel Out of Core ScaLAPACK LU, QR, and Cholesky Factorization Routines,” *Concurrency: Practice and Experience*, Vol. 12, Number 15 (2000): 1481-1493.

Dongarra, J. “Performance of Various Computers Using Standard Linear Equations Software (LINPACK Benchmark Report),” University of Tennessee Computer Science Department *Technical Report*. CS-89-85, 2001.

Dongarra, J., Fagg, G., Hempel, R., Walker, D. “Message Passing Software Systems,” Webster, J. ed., to appear in *Encyclopedia of Electrical and Electronics Engineering* (New York: Wiley and Sons), 2000.

Dongarra, J., Eijkhout, V. “Numerical Linear Algebra Algorithms and Software,” *Journal of Computation and Applied Mathematics*, Vol. 123, (2000): 489-514.

Dongarra, J., Eijkhout, V., Luszczek, P. "Recursive approach in sparse matrix LU factorization," Bubak, M., Moscinski, J., Noga, M. (Eds.) In *Proceedings of 1st SGI Users Conference* (Cracow, Poland: Academic Computer Center, CYFRONET AGH), 2000: 409-418.

Dongarra, J., Kacsuk, P., Podhorszki, N. (Eds.) *Recent Advances in Parallel Virtual Machine and Message Passing Interface, Lecture Notes in Computer Science: Proceedings of 7th European PVM/MPI Users' Group Meeting 2000*, (Hungary: Springer Verlag), 2000, Vol. 1908.

Dongarra, J., Martin, J. "Benchmarks," Ralston, A., Reilly, E., Hemmendinger, D. (Eds.) In *Encyclopedia of Computer Science* 4th ed. (England: Nature Publishing Group), 2000: 137-138.

Dongarra, J., Meuer, H., Simon, H., Strohmaier, E. "High Performance Computing Today," Cummings, P., Westmoreland, P., Carnahan, B., (Eds.) In *FOMMS 2000: Foundations of Molecular Modeling and Simulation Conference*, Keystone, Colorado, July 2000 (New York: American Institute of Chemical Engineers), 2000, Vol. 325: 96-100.

Dongarra, J., Meuer, H., Strohmaier, E. "TOP500 Supercomputer Sites (15th edition)," University of Tennessee Computer Science Department *Technical Report*, June 2000. UT-CS-00-442.

Dongarra, J., Raghavan, P. "A Grid Computing Environment for Enabling Large Scale Quantum Mechanical Simulations," Buyya, R., Baker, M. (Eds.) In *Proceedings of GRID 2000: IEEE/ACM International Workshop on Grid Computing, Lecture Notes in Computer Science* , (Hungary: Springer Verlag), 2000, Vol. 1971: 102-110.

Dongarra, J., Raghavan, P. "A New Recursive Implementation of Sparse Cholesky Factorization," *Proceedings of 16th IMACS World Congress 2000 on Scientific Computing, Applications Mathematics and Simulation*, Lausanne, Switzerland, August 22, 2000.

Fagg, G., Dongarra, J. "FT-MPI: Fault Tolerant MPI, Supporting Dynamic Applications in a Dynamic World," Dongarra, J., Kacsuk, P., Podhorszki, N. (Eds.). In *Recent Advances in Parallel Virtual Machine and Message Passing Interface, Lecture Notes in Computer Science: Proceedings of 7th European PVM/MPI Users' Group Meeting 2000*, (Hungary: Springer Verlag), 2000, Vol. 1908: 346-353.

Fagg, G., Vadhiyar, S., "ACCT: Automatic Collective Communications Tuning," Dongarra, J., Kacsuk, P., Podhorszki, N. (Eds.). In *Recent Advances in Parallel Virtual Machine and Message Passing Interface, Lecture Notes in Computer Science: Proceedings of 7th European PVM/MPI Users' Group Meeting 2000*, (Hungary: Springer Verlag), 2000, Vol. 1908.

Henry, G., Watkins, D., Dongarra, J. "A Parallel Implementation of the Nonsymmetric QR Algorithm for Distributed Memory Architectures," to appear in *SIAM Journal on Scientific Computing*.

Kennedy, K., Broom, B., Cooper, K., Dongarra, J., Fowler, R., Gannon, D, Johnsson, L., Mellor-Crummey, J., Torczon, L. "Telescoping Languages: A Strategy for Automatic Generation of Scientific Problem-Solving Systems from Annotated Libraries," to appear in *Journal of Parallel and Distributed Computing*.

Millar, J., McMahan, P., Dongarra, J. "RIBAPI - Repository in a Box Application Programmer's Interface," University of Tennessee Computer Science *Technical Report*, January 2000. UT-CS-00-438.

Raman, G., Dongarra, J. "Design and Implementation of NetSolve using DCOM as the Remoting Layer," University of Tennessee Computer Science Department *Technical Report*, May 2000. UT-CS-00-440.

Vadhiyar, S., Fagg, G., Dongarra, J. "Automatically Tuned Collective Communications," *Proceedings of SuperComputing 2000 (SC'2000)* Dallas, TX, November 2000.

PUBLICATIONS ON-LINE

Many ICL Publications listed here can be downloaded from http://icl.cs.utk.edu/publications.html

CONTACT INFORMATION

ADDRESS

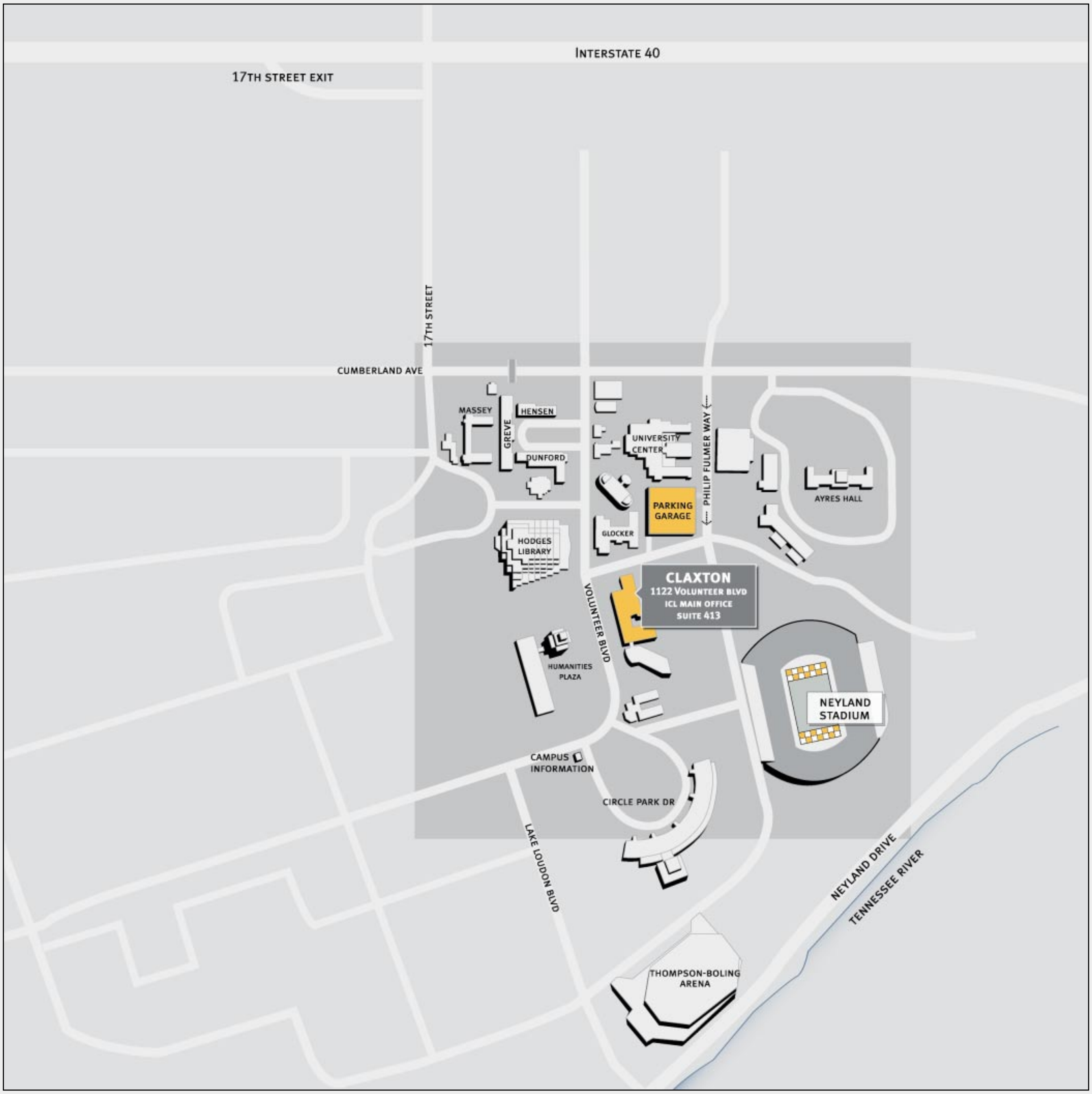
Innovative Computing Laboratory
Suite 203
1122 Volunteer Boulevard
Knoxvile, TN 37996-3450

WEB SITE http://icl.cs.utk.edu/

PHONE 865.974.8295

FAX 865.974.8296

MAP OF UT CAMPUS AREA OF ICL



http://icl.cs.utk.edu/