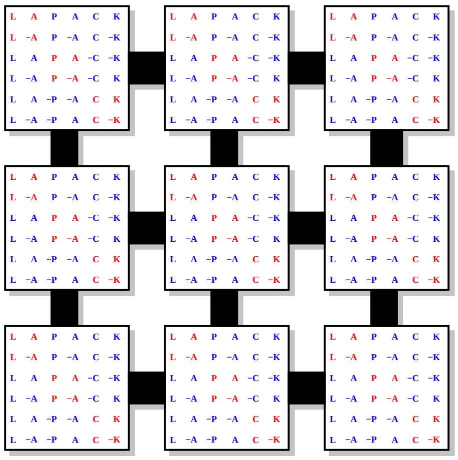
SUPERCOMPUTING '96

ScaLAPACK: A Portable Linear Algebra Library For Distributed Memory Computers – Design Issues And Performance

L. S. Blackford, J. Choi, A. Cleary, J. Demmel,
I. Dhillon, J. Dongarra, S. Hammarling, G. Henry,
A. Petitet, K. Stanley, D. Walker, and R. C. Whaley



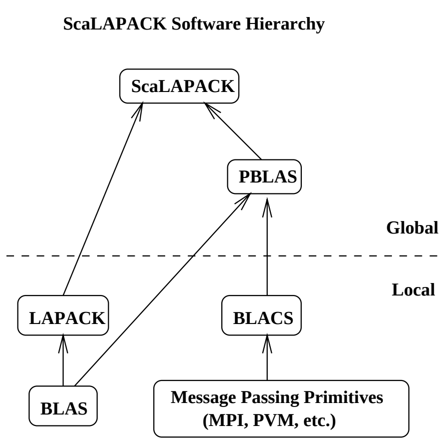
http://www.netlib.org/scalapack/index.html

DARPA / DOE / NSF / CRPC

OUTLINE

- **Design of ScaLAPACK**
- **Parallel BLAS (PBLAS)**
- **Contents of ScaLAPACK**
- **HPF Interface to ScaLAPACK**
- **Issues of Heterogeneous Computing**
- **Performance**
- **Future Directions**
- **Conclusions**

ScaLAPACK Software Hierarchy



Motivation and Goals

Goal: Port LAPACK to distributed-memory environments.

- **Efficiency**
  - Optimized compute and communication engines
  - Block-partitioned algorithms (Level 3 BLAS) utilize memory hierarchy and yield good node performance
- **Scalability**
  - as the problem size and number of processors grow
- **Reliability**
  - Whenever possible, use LAPACK algorithms and error bounds.
- **Portability**
  - isolate machine dependencies to BLAS and the BLACS
- **Flexibility**
  - Modularity: Build rich set of linear algebra tools: BLAS, BLACS, PBLAS
- **Ease-of-Use**
  - Calling interface similar to LAPACK

Many of these goals have been achieved through the promotion of *standards* for computation (BLAS, PBLAS) and communication (PVM, MPI).

OBJECT-BASED DESIGN IN FORTRAN77

> Each global data object is assigned an **array descriptor**.

The **array descriptor**

- contains the information required to establish the mapping between a global array entry and its corresponding process and memory location.
- is differentiated by the DTYPE_ (first entry) in the descriptor.
- provides a flexible framework to easily specify additional data distributions or matrix types.

Array descriptors are currently supported for:

- **dense** matrices
- **band** and **tridiagonal** matrices
- **out-of-core** matrices

User must distribute all global arrays prior to the invocation of a ScaLAPACK routine.

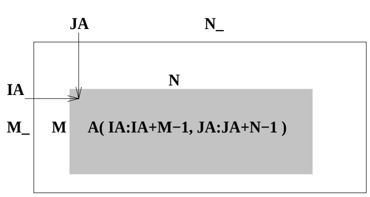
PBLAS – INTRODUCTION

Parallel Basic Linear Algebra Subprograms for distributed-memory MIMD computers.

- **Similar functionality** and **naming scheme** as the BLAS
- **Simplification of the parallelization** of dense linear algebra codes: especially when BLAS-based,
- **Clarity:** code is shorter and easier to read,
- **Modularity:** gives programmer larger building blocks,
- **Portability:** machine dependencies are confined to the PBLAS (BLAS and BLACS).

PBLAS – ARGUMENT CONVENTIONS

- **Global view of the matrix operands**, allowing global addressing of distributed matrices (hiding complex local indexing),



- **Code reusability**, interface very close to sequential BLAS:

```
CALL  DGEXXX( M, N, A( IA, JA ), LDA )
                    ↓
CALL PDGEXXX( M, N, A, IA, JA, DESCA )

 CALL  DGEMM( 'No Transpose', 'No Transpose',
$    M-J-JB+1, N-J-JB+1, JB, -ONE, A(J+JB,J),
$    LDA, A(J,J+JB), LDA, ONE, A(J+JB,J+JB),
$    LDA )
                    ↓
 CALL PDGEMM( 'No Transpose', 'No Transpose',
$    M-J-JB+1, N-J-JB+1, JB, -ONE, A, J+JB,
$    J, DESCA, A, J, J+JB, DESCA, ONE, A,
$    J+JB, J+JB, DESCA )
```


TABLE OF Routines in ScaLAPACK

Problem type	SDrv	EDrv	Fact	Solve	Inv	Cond Est	It Ref
$Ax = b$							
Triangular				X	X	X	X
SPD	X	X	X	X	X	X	X
SPD banded	X		X	X			
SPD tridiagonal	X		X	X			
General	X	X	X	X	X	X	X
General banded	X		X	X			
General tridiagonal	X		X	X			
Least squares (full rank)	X		X	X			
GQR			X				
GRQ			X				
$Ax = \lambda x$ or $Ax = \lambda Bx$							
Symmetric (2 types)	X	X	X				
General (2 types)			X				
Generalized B SPD		X	X				

- **Orthogonal/unitary transformation routines.**
- **Comprehensive Installation Guide and Test Suite for all ScaLAPACK routines.**
- **ScaLAPACK Users' Guide.**

Prototype Codes

- HPF interface to ScaLAPACK
- Matrix Sign Function for Eigenproblems
- Out-of-core solvers
- SVD

DOCUMENTATION, TEST SUITES, EXAMPLES, ...

- **Documentation**
  - **ScaLAPACK Users' Guide**
  - **Installation Guide for ScaLAPACK**
- **Test Suites for ScaLAPACK, PBLAS, BLACS**
- **Example Programs**
  - `http://www.netlib.org/scalapack/examples/`
- **Pre-built ScaLAPACK libraries on netlib**
- **Commercial Use**
  - NAG Numerical PVM (and MPI) Library
  - IBM Parallel ESSL
  - SGI Cray Scientific Software Library
  - Fujitsu
  - Visual Numerics (IMSL)

HPF INTERFACE TO SCALAPACK

Given these declarations:

```
      INTEGER :: IPIV(1000), INFO
      DOUBLE PRECISION, DIMENSION(1000,1000) :: A, B, C
!HPF$ DISTRIBUTE (CYCLIC(64), CYCLIC(64)) :: A, B, C
!HPF$ ALIGN IPIV(:) WITH A(:,*)
```

calls could be made as simple as:

```
      CALL HPF_GESV(A, B)
      CALL HPF_POSV(A, B)
      CALL HPF_GELS(A, B)
      CALL HPF_GEMM(A, B, C)
```

or slightly more complex as:

```
      CALL HPF_GESV(A(65:100,65:100),B(65:100,129:200), &
     &              IPIV(65:100),INFO)
      CALL HPF_POSV(A, B, 'L', INFO)
      CALL HPF_GELS(A, B, 'T', INFO)
      CALL HPF_GEMM(A, B, C, 'N', 'T', 1.0D0, 1.0D0)
```


HETEROGENEOUS COMPUTING

Software intended to be used in cluster computing contexts.

Difficulties arise with the following issues:

- Communication of floating point values between processors
- Machine precision and other machine parameters
- Different versions of compilers
- Checking global floating-point arguments
- Iterative convergence across cluster
- Deadlock

Defensive programming required to address these issues.

Details of ScaLAPACK experiences can be found in:

`http://www.netlib.org/lapack/lawns/lawn112.ps`

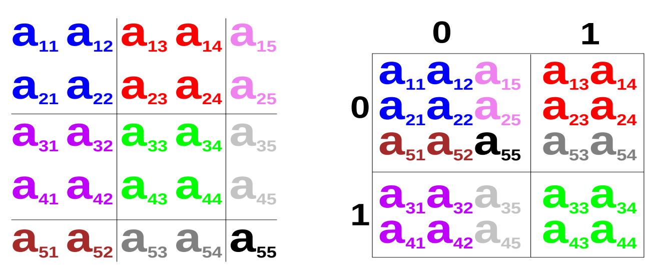
SCALAPACK – STORAGE EXAMPLE

- A dense `M_`-by-`N_` **matrix is block-partitioned** and these `MB_`-by-`NB_` blocks are distributed according to the

  2-dimensional block-cyclic scheme
  ⇒ **load balanced computations, scalability**

- Locally, the scattered **columns are stored contiguously** (FORTRAN "Column-major")

  ⇒ **re-use of the BLAS** (leading dimension `LLD_`).

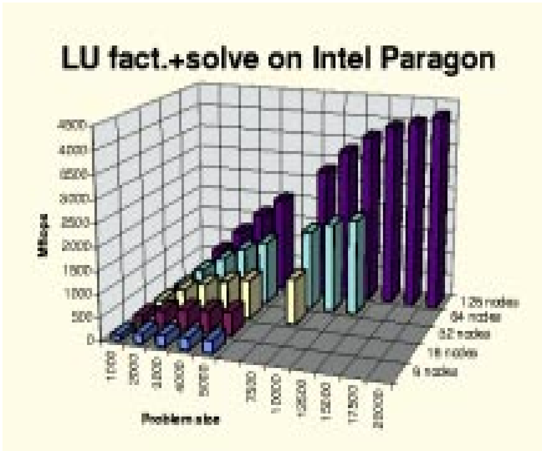


5 x 5 matrix partitioned in 2 x 2 blocks — 2 x 2 process grid point of view

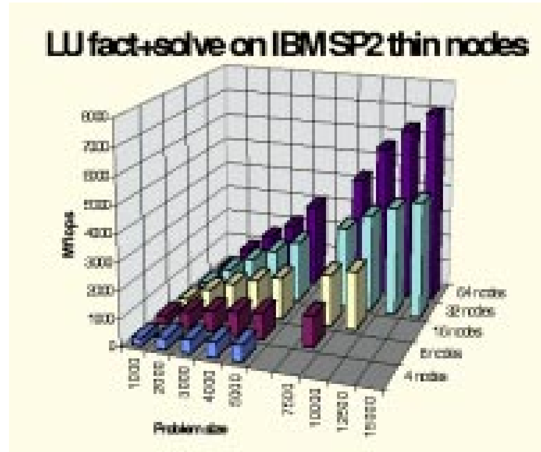
Descriptor `DESC_`**:** 9-Integer array describing the matrix layout, containing `DTYPE_`, `CTXT_`, `M_`, `N_`, `MB_`, `NB_`, `RSRC_`, `CSRC_`, and `LLD_`, where (`RSRC_`, `CSRC_`) are the coordinates of the process owning the first matrix entry in the grid specified by `CTXT_`.

Ex: `M_=N_=5`, `MB_=NB_=2`, `RSRC_=CSRC_= 0`, `LLD_` $\geq$ 3 (in process row 0), and 2 (in process row 1).

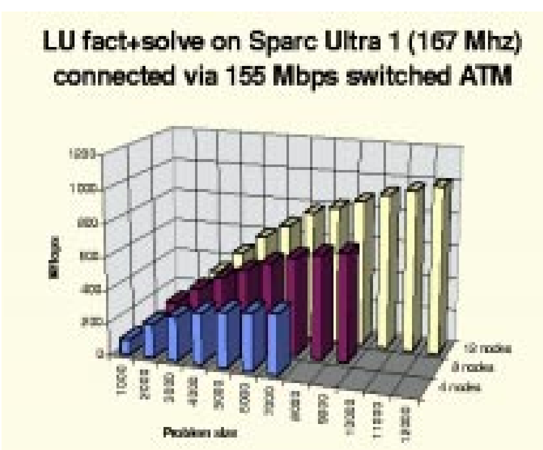
PERFORMANCE RESULTS



PERFORMANCE RESULTS

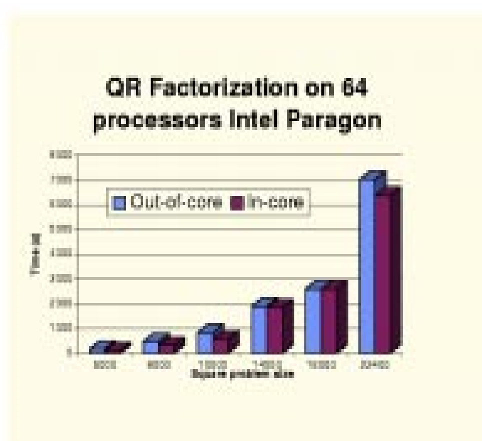


PERFORMANCE RESULTS



OUT-OF-CORE PERFORMANCE RESULTS

- Prototype code for Out-of-Core extension to ScaLAPACK Library
- Linear solvers based on "Left-looking" variants of LU, QR, and Cholesky factorization
- Portable I/O interface for reading/writing ScaLAPACK matrices



SCALAPACK – ONGOING WORK

- **Increased flexibility and usability**
  - **Algorithmic redistribution** in PBLAS
  - Removal of **alignment restrictions**
  - **Algorithmic blocking**
- **Increased functionality**
  - Divide and Conquer
  - Rank-deficient Linear Least Squares
- **HPF interface** to all ScaLAPACK routines
- **Sparse Direct Solvers**
- **Iterative Solvers**
- ...

Conclusions

ScaLAPACK – Scalable Linear Algebra Package

- **Goals attained through the promotion and development of standards (BLAS, PBLAS)**
- **Object-based design provides a framework for additional data distributions and matrix types**
- **Portable across a wide range of architectures**
  - distributed-memory MIMD computers
  - networks of workstations supporting MPI or PVM
- **Available from commercial vendors**

SCALAPACK – CONTRIBUTORS

University of Tennessee, Knoxville
Jack Dongarra, Principal Investigator

- Susan Blackford
- Andrew Cleary
- Brett Ellis
- Sven Hammarling
- Greg Henry
- Antoine Petitet
- R. Clint Whaley

University of California, Berkeley
Jim Demmel, Co-Principal Investigator

- Soumen Chakrabarti
- Inderjit Dhillon
- Ray Fellers
- Melody Ivory
- Ren-Cang Li
- Xiaoye Li
- Huan Ren
- Howard Robinson
- Ken Stanley
- Andrey Zege

ScaLAPACK Project

The ScaLAPACK project is a collaborative effort between the following institutions and principal investigators:

- Oak Ridge National Laboratory
  Ed D'Azevedo, Co-Principal Investigator
- Rice University
  Dan Sorensen, Co-Principal Investigator
- University of California, Berkeley
  Jim Demmel, Co-Principal Investigator
- University of California, Los Angeles
  Tony Chan, Co-Principal Investigator
- University of Illinois
  Michael Heath, Co-Principal Investigator
  Padma Raghavan, Co-Principal Investigator
- University of Tennessee, Knoxville
  Jack Dongarra, Principal Investigator

The ScaLAPACK project is comprised of 4 components:

- dense matrix software (**ScaLAPACK**)
- large sparse eigenvalue software (**PARPACK**)
- sparse direct systems software (**CAPSS**)
- preconditioners for large sparse iterative solvers (**PARPRE**)

Funding for this effort comes in part from **DARPA**, **DOE**, **NSF**, and **CRPC**.

`http://www.netlib.org/scalapack/index.html`

SCALAPACK – REFERENCES

- ScaLAPACK software and documentation can be obtained via:
  - WWW: `http://www.netlib.org/scalapack`,
  - WWW: `http://www.netlib.org/lapack/lawns`,
  - (anonymous) ftp `ftp.netlib.org`:
    - cd scalapack; get index
    - cd lapack/lawns; get index
  - email `netlib@www.netlib.org` with the message:

    `send index from scalapack`

- Comments and questions can be addressed to

  `scalapack@cs.utk.edu`

- LAPACK Working Notes:
  #43, #55, #57, #58, #61, #65, #73, #80, #86, #91, #92, **#93**, **#94**, **#95**, #96, **#100**, **#112**.
- J. Dongarra and D. Walker, *Software Libraries for Linear Algebra Computations on High Performance Computers*, SIAM Review, Vol. 37, (2), pp. 151 – 180, 1995.
- J. Choi, J. Dongarra, L. S. Ostrouchov, A. Petitet, D. Walker, and R. C. Whaley, *The Design and Implementation of the ScaLAPACK LU, QR, and Cholesky Factorization Routines*, Scientific Programming, Vol. 5, pp. 173–184, 1996.