

CSE 260 – Introduction to Parallel Computation

Topic 8: Benchmarks & Applications

October 25, 2001

Parallel Benchmarks

- Microbenchmarks measure one aspect of computer.
 - e.g. MPI all-to-all bandwidth.
- Kernel benchmarks: “inner loop” from important codes.
 - Linpack, NPB (NAS Parallel Benchmark) kernels, ...
- PseudoApps:
 - NPB pseudo apps, SPLASH (for multiprocessors), ...
- Full applications:
 - SPEChpg (hpg = high performance group = SPEC+Perfect)
 - SPEChpg96 includes seismic, cfd, molecular dynamics, ...

John McCalpin observes Linpack has .015 correlation with application performance

NAS Parallel Benchmarks (NPB)

- NAS = Numerical Aeronautics Simulation
- Developed by NASA to help choose what to buy.
- Five kernels and three pseudo apps.
- Widely used in parallel computation community.
- NPB 1 were “pencil & paper”
 - Specified by simple untuned serial code.
 - Vendors write code - few limits, except get right answer
- NPB 2 are MPI implementations.
 - vendors can tune code, but must report how much.

LAPACK

- Evolved from Linpack and Eispack.
- Dense Linear Algebra:
 - Solve $Ax = b$ for x
 - A can be full, triangular, or symmetric forms
 - Least squares: choose x to minimize $\|Ax - b\|_2$
 - Eigenvalues: Find λ s.t. $\det(A - \lambda I) = 0$.
 - Singular Value Decomposition.
 - Decompositions: LU, QR, Cholesky, ...
- 600,000 lines of Fortran code

Basic Linear Algebra Subroutines (BLAS)

Called by LAPACK programs to do low-level stuff.

“Vanilla” implementations comes with LAPACK.

Vendors can supply well-tuned versions.

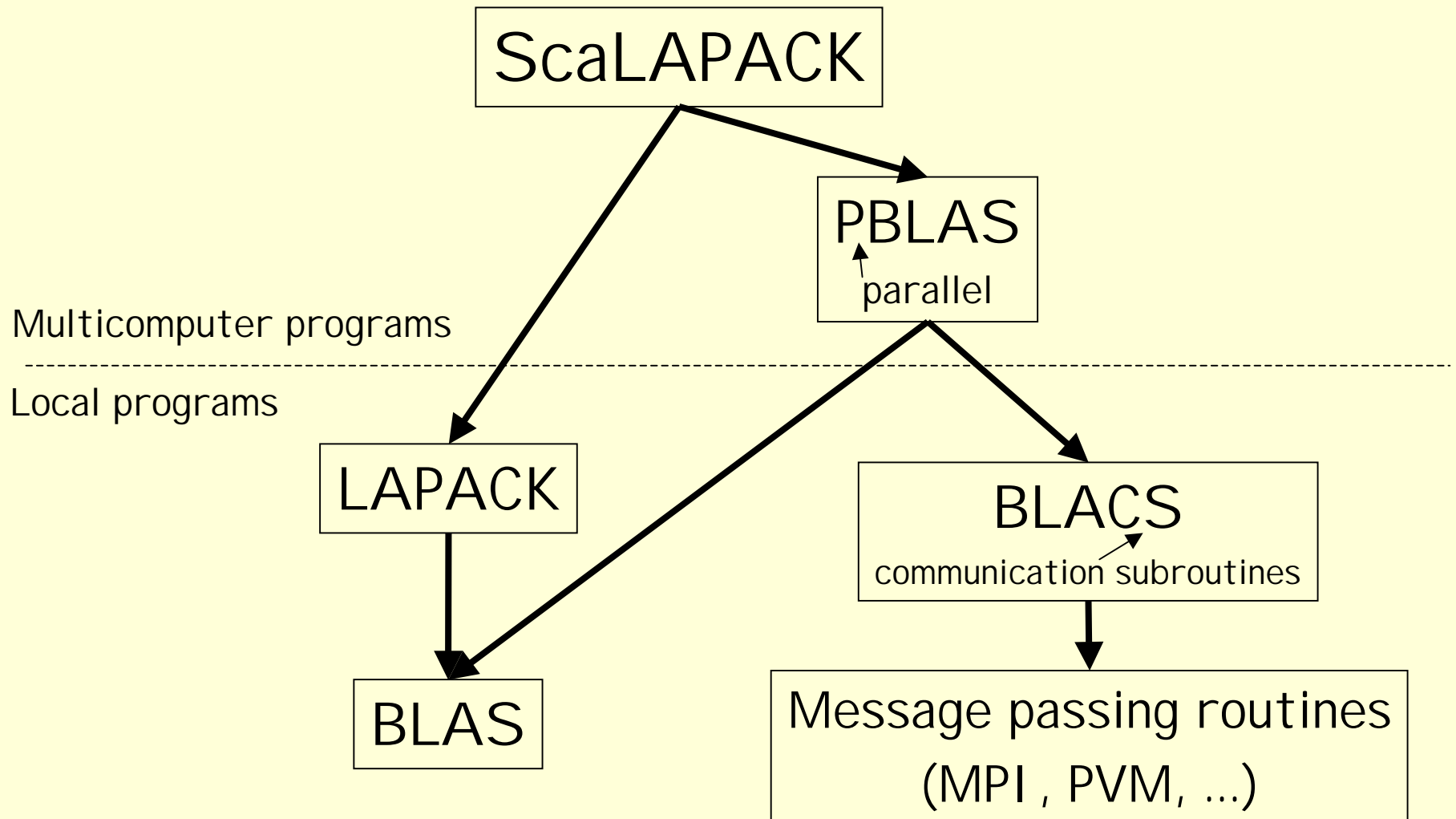
Levels:

- Level 1 for vector ops (DDOT, DAXPY, MAX...)
 - 1970's: Got 10x performance improvement on IBM 370.
- Level 2 for matrix-vector operations
 - 1980: For performance on Cray Vector and other machines.
- Level 3 for matrix-matrix
 - 1989: Needed for computers with memory hierarchies.

LAPACK names

- Routines have 4- to 6-letter names: TFFOO
 - T is precision:
 - Double, Single, Complex, Z (double complex)
 - FF is matrix structure:
 - GE = general (full rectangular)
 - TR = triangular, SY = symmetric, ...
 - OO is operation:
 - MM = matrix multiply, MV = matrix x vector
 - EV = eigenvalue, LS = least squares, ...
- Example: DGEMM is matrix-matrix product

ScaLAPACK: parallelized LAPACK



LU decomposition

$$\begin{bmatrix} 4 & 3 & 2 \\ 2 & 3 & 2 \\ 8 & 2 & 4 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 8 & 2 & 4 \\ 2 & 3 & 2 \\ 4 & 3 & 2 \end{bmatrix}$$

"Partial pivot":
Swap rows to maximize a_{11} .

$$= \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 1/4 & 1 & 0 \\ 1/2 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} 8 & 2 & 4 \\ 0 & 5/2 & 1 \\ 0 & 2 & 0 \end{bmatrix}$$

Subtract multiples of first row from other rows to get zeros in column 1.

$$= \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \times \begin{bmatrix} 1 & 0 & 0 \\ 1/4 & 1 & 0 \\ 1/2 & 4/5 & 1 \end{bmatrix} \times \begin{bmatrix} 8 & 2 & 4 \\ 0 & 5/2 & 1 \\ 0 & 0 & -4/5 \end{bmatrix}$$

(No swap needed here for a_{22}).

Now make rest of column 2 zeros.

Use P matrix to undo swaps

Use L matrix to undo row ops

$$\boxed{A = P \times L \times U}$$

The "P" is silent in (P) LU decomposition

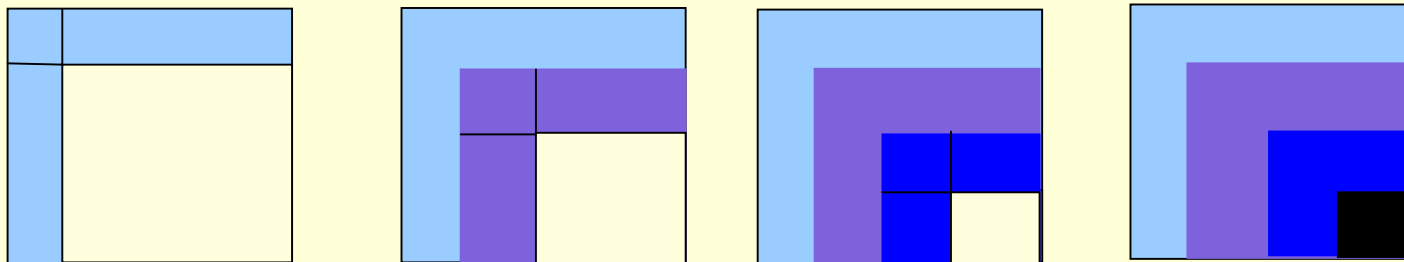
LU decomposition

- Solving $Bx = y$ is easy when $B = P, L,$ or $U.$

Example: for $Lx = y,$ first find $x_1,$ then x_2, \dots

- L and U can share storage originally occupied by A matrix.
- “Subtract multiples of one row from others” is $A' = A - (i\text{-th column of } L) \times (i\text{-th row of } U)$

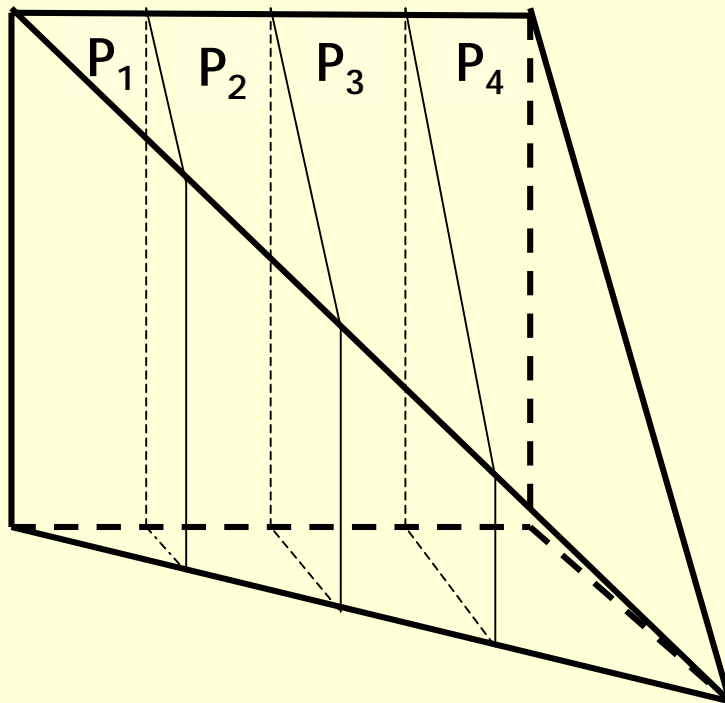
First step ... second ... third ...



Visualize as a pyramid

Parallelizing the LU pyramid

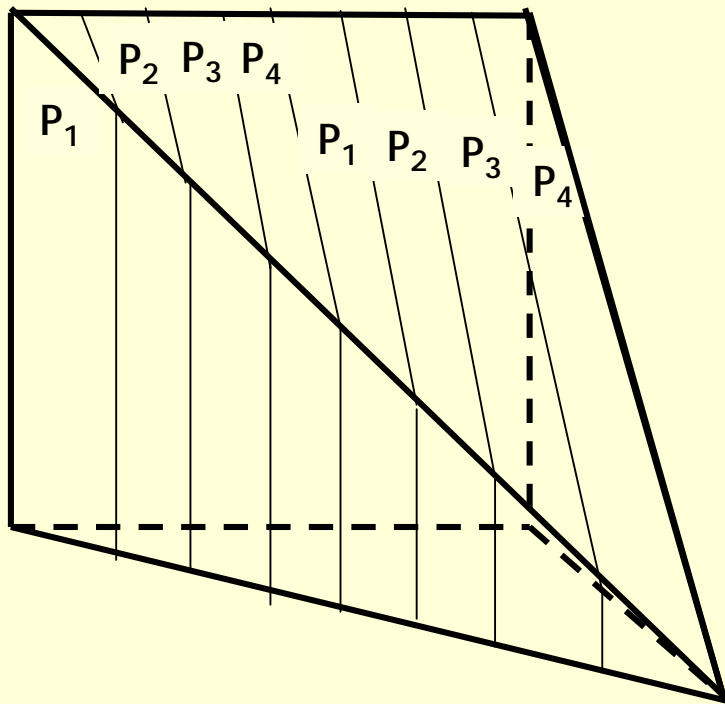
One-dimensional block decomposition



Problem: Load imbalance!
(e.g. P_1 is idle much of the time)

Parallelizing the LU pyramid

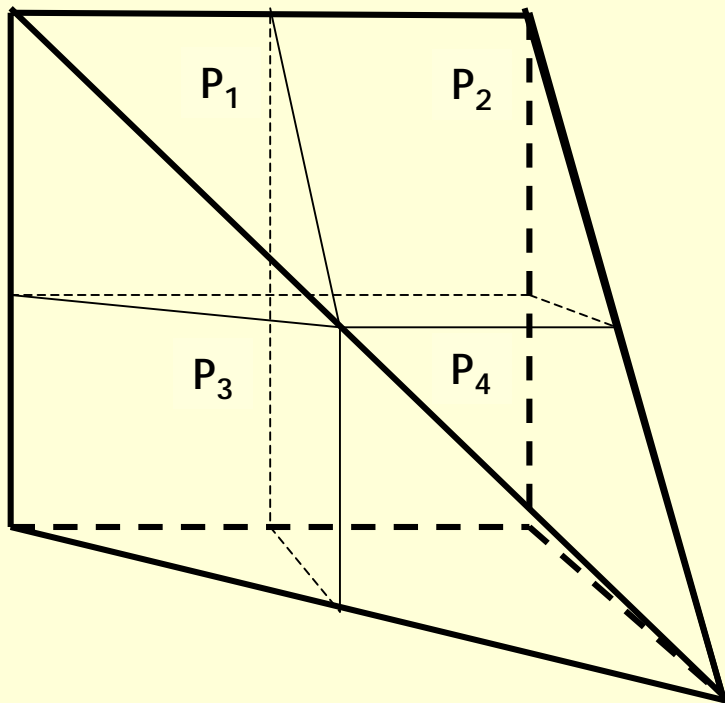
1-D block cyclic decomposition



- Load balance is improved by assigning multiple slices to each processor.
- But block cyclic needs more rounds of communication
- Compromise: 4 to 10 times as many slices as processors.

Parallelizing the LU pyramid

2-D block decomposition

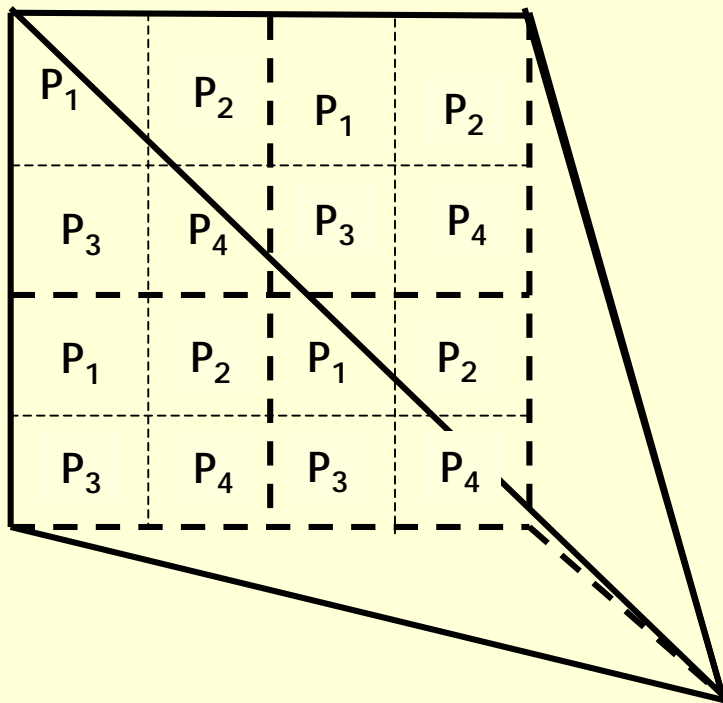


- With 1-D, each processor communicates to $P-1$ others.
- 2-D: processors communicate to $2(P^{1/2}-1)$ others.
- 2-D has fewer rounds of communication too.
- 2-D requires communication for pivoting (finding col min)

For small P (e.g. < 16), extra overhead and need to send both rows *and* columns makes 2-D undesirable

Parallelizing the LU pyramid

2-D block cyclic decomposition



Better load balancing.
Requires communication for
pivot step

Other common algorithms

- Finite Difference Methods
 - Used for PDE's with regular structure (like our project)
- Finite Element Methods (FEM's)
 - Often to solve PDE's (partial differential equations) with irregular structure.
 - Car crash simulations (LSDYNA)
 - Vibration analysis of buildings, bridges, airplanes
- Fast Fourier Transforms (FFT's)
 - Given position of vibrating object at various times, determines frequencies of vibration (or vice versa).

Algorithmic Improvements

Comparison of Finite Difference Solvers

Diffusion problem, run on NCube-2
study by Shadid & Tuminaro at Sandia (1990-ish)

Algorithm	Float Ops (billions)	Time (secs)	MFlop/sec
Jacobi	3820	2124	1800
Gauss- Seidel	1210	885	1365
Least Squares	259	185	1400
Multigrid	2	6.7	318

Applications run on parallel computers

- CFD = Computational Fluid Dynamics

- Aerodynamics of airplanes, ink jet blobs, ...
- Ocean and air circulation (weather & climate)
- Petroleum reservoir modeling
- Combustion chamber design
- Plasma physics in stars and reactors

Use Finite Difference or Finite Element Methods

- Structural Dynamics

- Car crash simulations
- Building, bridge, or airplane vibration analysis

Usually use FEM's

Applications (continued)

- Signal processing
 - Seismic analysis (e.g. to find underground oil)
 - Radar & sonar
 - Search for Extraterrestrial Intelligence (SETI)

Usually use FFT's

- Molecular dynamics

(simulate forces on molecules, see how they move)

 - Protein folding
 - Drug action
 - Materials analysis (e.g. crack propagation)

Need lots of random numbers

Applications (continued)

- Electromagnetic simulation

- Antenna design
- Determining if computer emits radio interference

Sometimes use huge dense matrix calculations

- Graphic and visualization

- Surface rendering
- Volume rendering (for translucent objects)

- Optimization

Maximize function subject to various constraints

- Scheduling (airlines, delivery routes, materials flow, ...)

Uses linear programming, combinatorial searches, ...

Applications (continued)

- N-body problems
 - Simulate N objects affected by gravity or other forces
 - Galaxy evolution,
 - “Fast Multipole” methods are good for this
- Genomics, proteomics
 - Determine if two proteins or DNA strings are similar
 - useful to trace evolution, determine function of genes,...
 - Determine likely structure of proteins
- Information retrieval
 - GIS data from satellites
 - Web searches

US Government Funding Agencies

- NSF: National Science Foundation
 - CISE (Computer and Information Science) directorate funds lots of parallel computing.
 - NPACI (SDSC at UCSD is lead site) and NCSA (UIUC is lead site) are large NSF centers.
- DOE: Department of Energy
 - Sponsors various national labs and the ASCI program
 - LBNL = Lawrence Berkeley National Lab (includes NERSC)
 - LLNL = Lawrence Livermore National Lab (Bay Area)
 - LANL = Los Alamos National Lab (New Mexico)
 - Sandia National Lab (New Mexico)
 - Argonne (U.Chicago), Oak Ridge (Tennessee), ...

More US Government Agencies

- DOD: Department of Defense
 - DARPA: Defense Advanced Research Projects Agency
 - Army, Navy and Air Force have funding orgs
(Not easy to break into funding circles)
- NIH: National Institute of Health
- NASA (includes NASA Ames lab in Bay Area)
- NSA: National Security Agency
- NOAA: National Ocean and Atmospheric Adm.
 - Climate & Weather - includes NCAR Lab (Boulder)

Assignment for Next Tuesday

- Read Culler et al, LogP: Towards a Realistic Model of Parallel Computation.

www.cs.berkeley.edu/~culler/papers/logp.ps

- Write down (to hand in at beginning of class) a question or comment you have on the paper. These will be used to stimulate discussion.