Cost-Effective Parallel Computational Electromagnetic Modeling

Daniel S. Katz, Tom Cwik
{Daniel.S.Katz, cwik}@jpl.nasa.gov
JPL

Beowulf System at JPL (Hyglac)

- 16 Pentium Pro PCs, each with 2.5 Gbyte disk, 128 Mbyte memory, Fast Ethernet card.
- Connected using 100Base-T network, through a 16-way crossbar switch.
- Theoretical peak: 3.2 GFLOP/s
- Sustained: 1.26 GFLOP/s

Beowulf System at Caltech (Naegling)

- ~120 Pentium Pro PCs, each with 3 Gbyte disk, 128 Mbyte memory, Fast Ethernet card.
- Connected using 100Base-T network, through two 80-way switches, connected by a 4 Gbit/s link.
- Theoretical peak: ~24 GFLOP/s
- Sustained: 10.9 GFLOP/s

Hyglac Cost

- Hardware cost: $54,200 (as built, 9/96)
  $22,000 (estimate, 4/98)
  - » 16 (CPU, disk, memory, cables)
  - » 1 (16-way switch, monitor, keyboard, mouse)
- Software cost: $600 (+ maintainance)
  - » Absoft Fortran compilers (should be $900)
  - » NAG F90 compiler ($600)
  - » public domain OS, compilers, tools, libraries

Naegling Cost

- Hardware cost: $190,000 (as built, 9/97)
  $154,000 (estimate, 4/98)
  - » 120 (CPU, disk, memory, cables)
  - » 1 (switch, front-end CPU, monitor, keyboard, mouse)
- Software cost: $0 (+ maintainance)
  - » Absoft Fortran compilers (should be $900)
  - » public domain OS, compilers, tools, libraries

Performance Comparisons

	Hyglac	Naegling	T3D	T3E600
CPU Speed (MHz)	200	200	150	300
Peak Rate (MFLOP/s)	200	200	300	600
Memory (Mbyte)	128	128	64	128
Communication Latency (µs)	150	322	35	18
Communication Throughput (Mbit/s)	66	78	225	1200

(Communication results are for MPI code)

Message-Passing Methodology

- Issue (non-blocking) receive calls:

```
CALL MPI_IRECV(...)
```

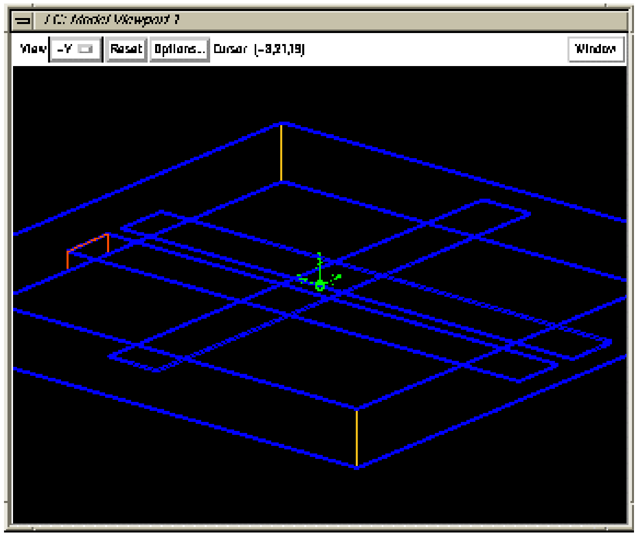
- Issue (synchronous) send calls:

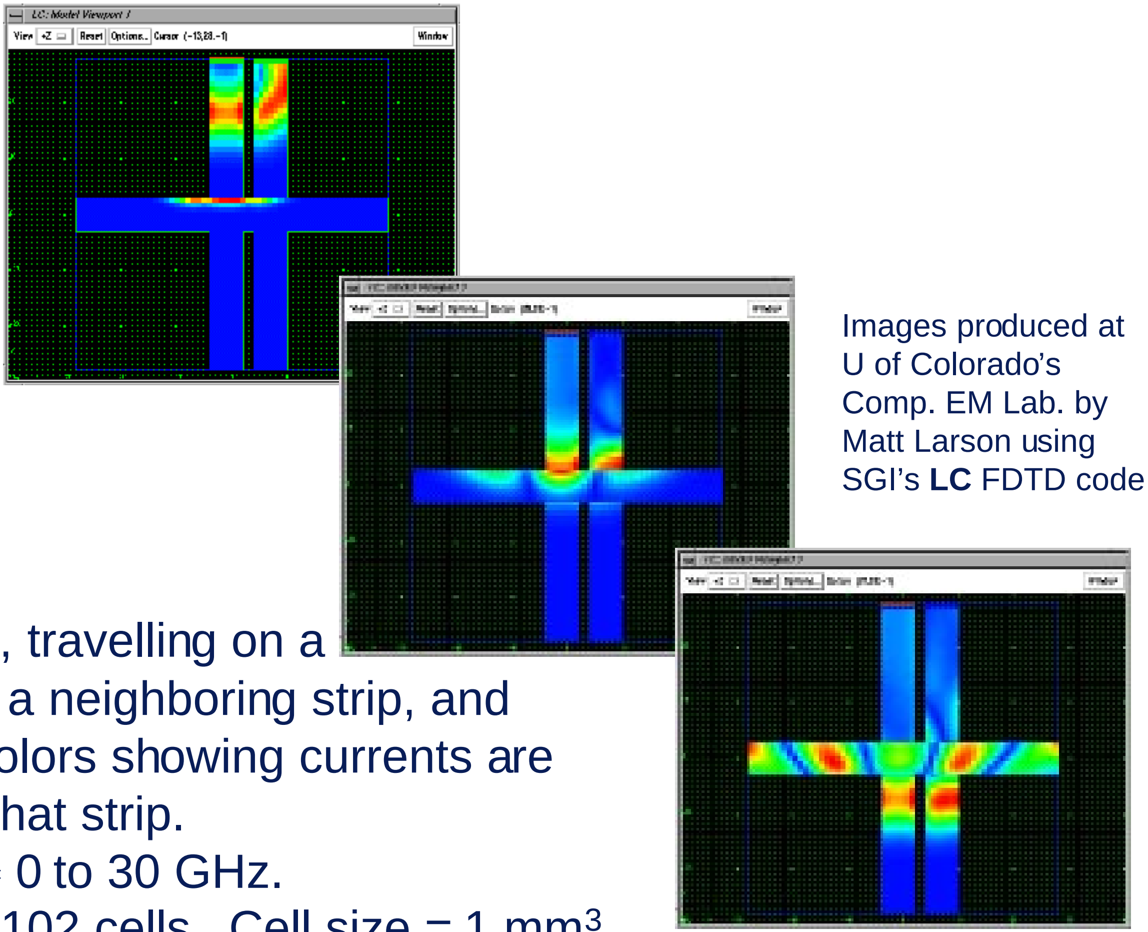
```
CALL MPI_SSEND(...)
```

- Issue (blocking) wait calls (wait for receives to complete):

```
CALL MPI_WAIT(...)
```

Finite-Difference Time-Domain Application





Images produced at
U of Colorado's
Comp. EM Lab. by
Matt Larson using
SGI's **LC** FDTD code

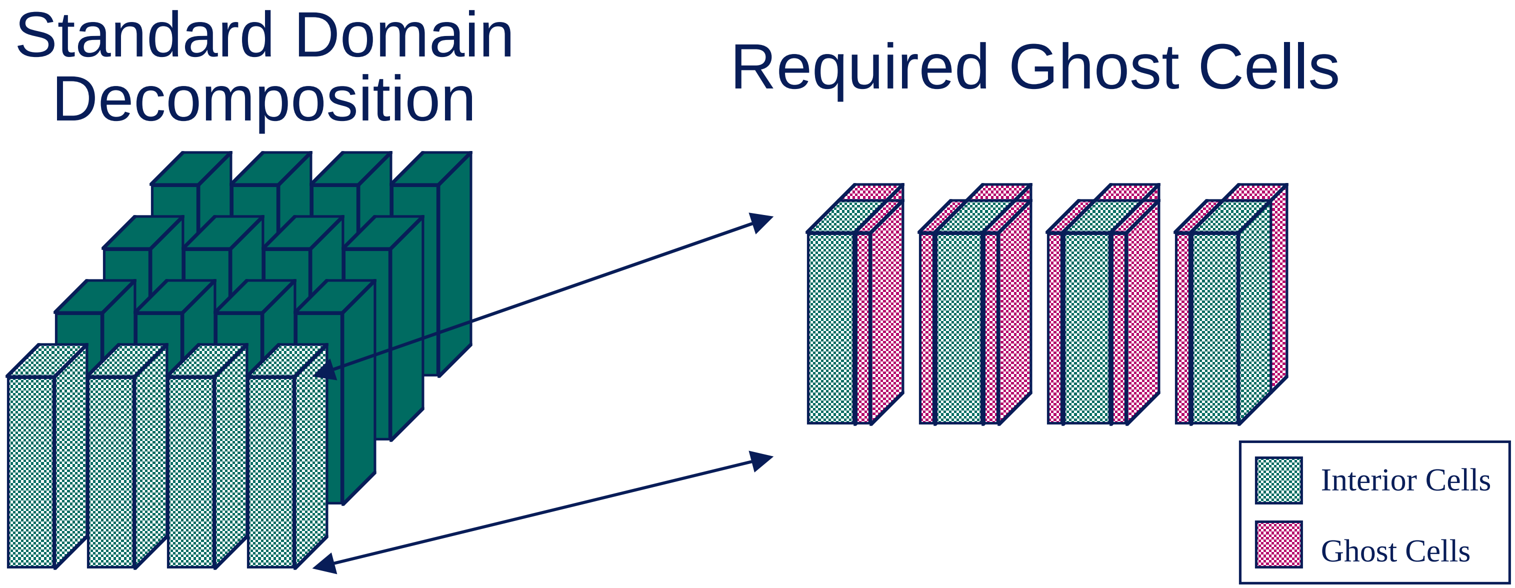
Time steps of a gaussian pulse, travelling on a microstrip, showing coupling to a neighboring strip, and crosstalk to a crossing strip. Colors showing currents are relative to the peak current on that strip.

Pulse: rise time = 70 ps, freq. ≈ 0 to 30 GHz.

Grid dimensions = 282 × 362 × 102 cells. Cell size = 1 mm^3.

FDTD Algorithm

- Classic time marching PDE solver
- Parallelized using 2-dimensional domain decomposition method with ghost cells.


Standard Domain Decomposition

Required Ghost Cells

FDTD Algorithm Details

- Uses Yee’s staggered grid
- Time Stepping Loop:
    - Update Electric Fields (three 5-point stencils, on x-y, x-z, y-z planes)
    - Update Magnetic Fields (three 5-point stencils, on x-y, x-z, y-z planes)
    - Communicate Magnetic Fields to ghost cells of neighboring processors (in x and y)

FDTD Results

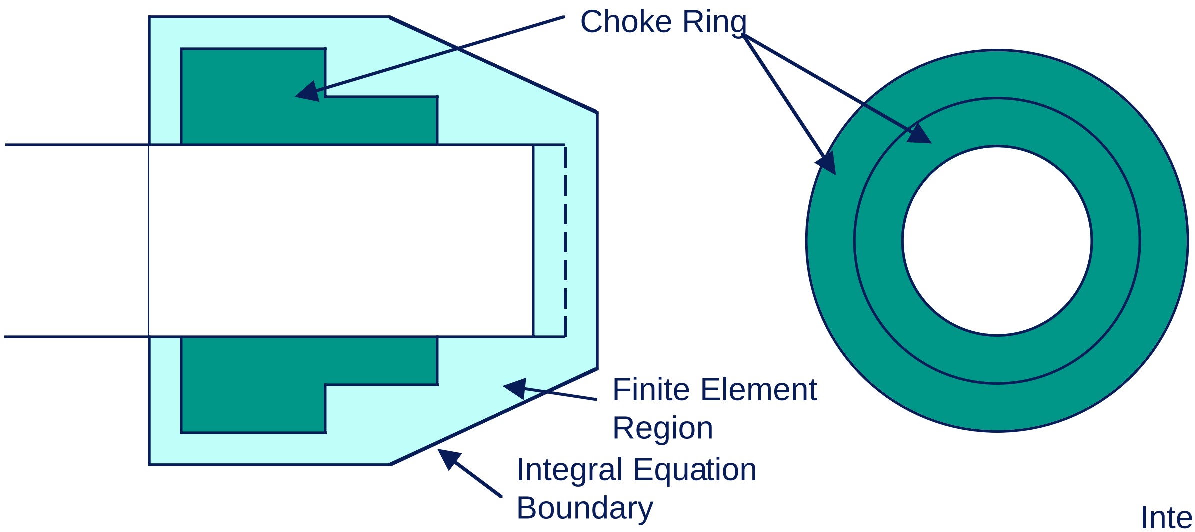
Number of Processors	Naegling	T3D	T3E-600
1	2.44 - 0.0	2.71 - 0.0	0.851 - 0.0
4	2.46 - 0.097	2.79 - 0.026	0.859 - 0.019
16	2.46 - 0.21	2.79 - 0.024	0.859 - 0.051
64	2.46 - 0.32	2.74 - 0.076	0.859 - 0.052

Time (wall clock seconds / time step),
scaled problem size (69 × 69 × 76 cells / processor),
times are: computation - communication

FDTD Conclusions

- Naegling and Hyglac produce similar results for 1 to 16 processors
- Scaling from 16 to 64 processors is quite reasonable
- On all numbers of processors, Beowulf-class computers perform similarly to T3D, and worse than T3E, as expected.

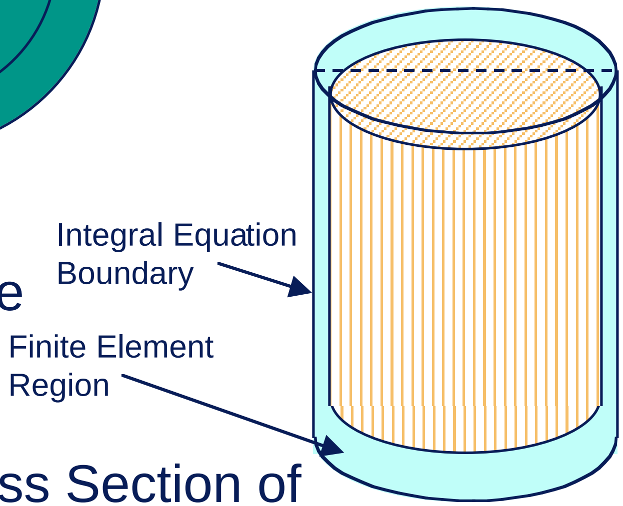
PHOEBUS


Choke Ring

Finite Element Region

Integral Equation Boundary

Radiation Pattern from JPL Circular Waveguide
(from C. Zuffada, *et. al.*, IEEE AP-S paper 1/97)


Integral Equation Boundary

Finite Element Region

Radar Cross Section of a dielectric cylinder

Typical Applications:

PHOEBUS Coupled Equations

$$\begin{bmatrix} K & C & 0 \\ C^{\dagger} & 0 & Z_0 \\ 0 & Z_M & Z_J \end{bmatrix} \begin{bmatrix} H \\ M \\ J \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ V_{inc} \end{bmatrix}$$

- This matrix problem is filled and solved by PHOEBUS
  - » The K submatrix is a sparse finite element matrix
  - » The Z submatrices are integral equation matrices.
  - » The C submatrices are coupling matrices between the FE and IE matrices.

PHOEBUS Solution Process

$$\begin{bmatrix} K & C & 0 \\ C^{\dagger} & 0 & Z_0 \\ 0 & Z_M & Z_J \end{bmatrix} \begin{bmatrix} H \\ M \\ J \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ V \end{bmatrix}$$

$$H = -K^{-1}CM$$

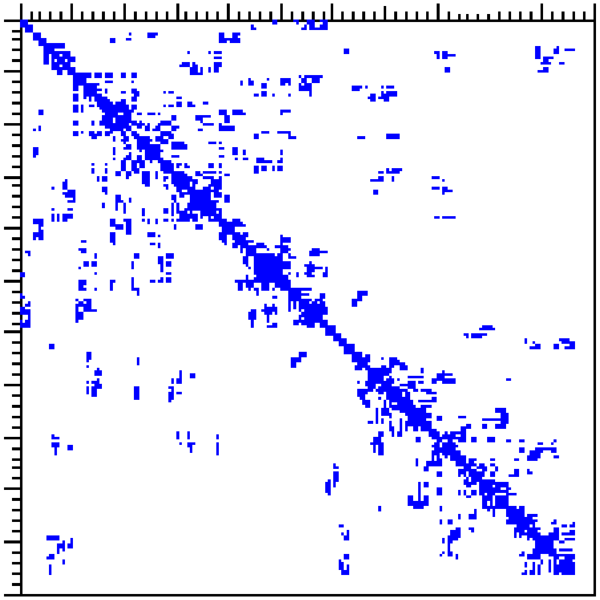
$$\begin{bmatrix} -C^{\dagger}K^{-1}C & Z_0 \\ Z_M & Z_J \end{bmatrix} \begin{bmatrix} M \\ J \end{bmatrix} = \begin{bmatrix} 0 \\ V \end{bmatrix}$$

- Find $-C^{\dagger}K^{-1}C$ using QMR on each row of C, building x rows of $K^{-1}C$, and multiplying with $-C^{\dagger}$.
- Solve reduced system as a dense matrix.

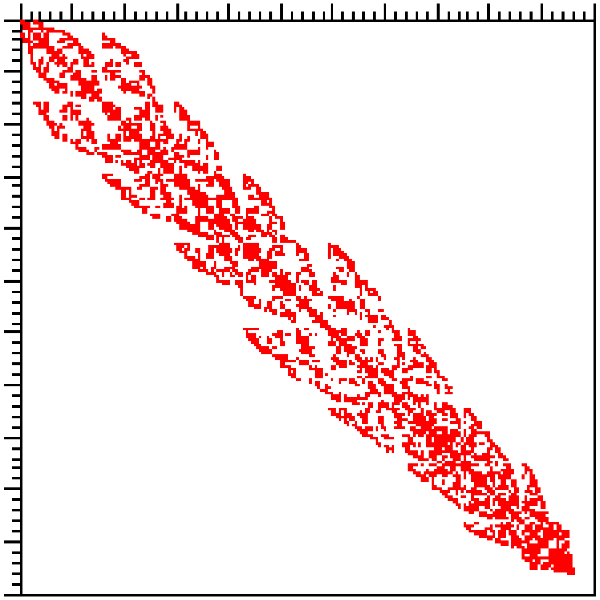
PHOEBUS Algorithm

- Assemble complete matrix
- Reorder to minimize and equalize row bandwidth of *K*
- Partition matrices in slabs
- Distribute slabs among processors
- Solve sparse matrix equation (step 1)
- Solve dense matrix equation (step 2)
- Calculate observables

PHOEBUS Matrix Reordering



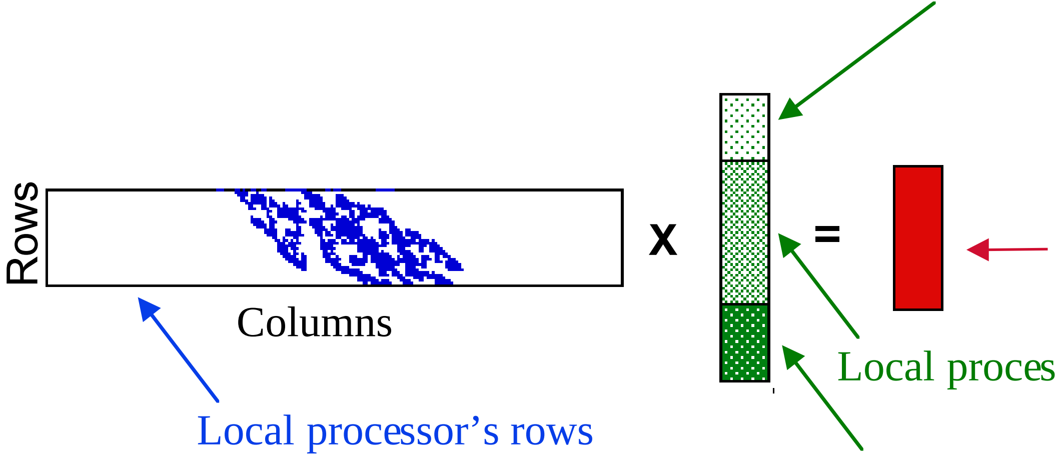
Original System



System after Reordering
for Minimum Bandwidth

Non-zero structure of matrices,
using SPARSPAK's GENRCM Reordering Routine

PHOEBUS
Matrix-Vector Multiply



Communication from processor to left

Local processor's rows

Local processor's rows

Communication from processor to right

Local processor's rows

PHOEBUS Solver Timing

Model: dielectric cylinder with 43,791 edges, radius = 1 cm, height = 10 cm, permittivity = 4.0, at 5.0 GHz

Number of Processors	T3D (shmem)	T3D (MPI)	Naegling (MPI)
Matrix-Vector Multiply Computation	1290	1290	1502
Matrix-Vector Multiply Communication	114	272	1720
Other Work	407	415	1211
Total	1800	1980	4433

Time of Convergence (CPU seconds), solving using 16 processors, pseudo-block QMR algorithm for 116 right hand sides.

PHOEBUS Solver Timing

Model: dielectric cylinder with 100,694 edges, radius = 1 cm, height = 10 cm, permittivity = 4.0, at 5.0 GHz

Number of Processors	T3D (shmem)	T3D (MPI)	Naegling (MPI)
Matrix-Vector Multiply Computation	868	919	1034
Matrix-Vector Multiply Communication	157	254	2059
Other Work	323	323	923
Total	1348	1496	4016

Time of Convergence (CPU seconds), solving using 64 processors, pseudo-block QMR algorithm for 116 right hand sides.

PHOEBUS Conclusions

- Beowulf is 2.4 times slower than T3D on 16 nodes, 3.0 times slower on 64 nodes
- Slowdown will continue to increase for larger numbers of nodes
- T3D is about 3 times slower than T3E
- Cost ratio between Beowulf and other machines determines balance points

General Conclusions

- Beowulf is a good machine for FDTD
- Beowulf may be ok for iterative solutions of sparse matrices, such as those from Finite Element codes, depending on machine size
- Key factor: amount of communication