Sparse Direct Solver on Today′s Architectures

Sherry Li
Lawrence Berkeley National Laboratory

DOE/DOD Workshop on Emerging High Performance Architectures and Applications

November 29-30, 2007

Content

- Algorithm flowchart
- Performance on representative machines
  - **IBM power5 (bassi @ NERSC)**
  - **Multicores: Intel Clovertown, Sun Niagra2 (Clusters @ UC Berkeley)**
- Acknowledgement
  - **Rich Vuduc, Georgia Tech**
  - **Sam Williams, UC Berkeley**

Gaussian Elimination (GE)

- Solving a system of linear equations Ax = b
- First step: (make sure α not too small . . . may need pivoting)

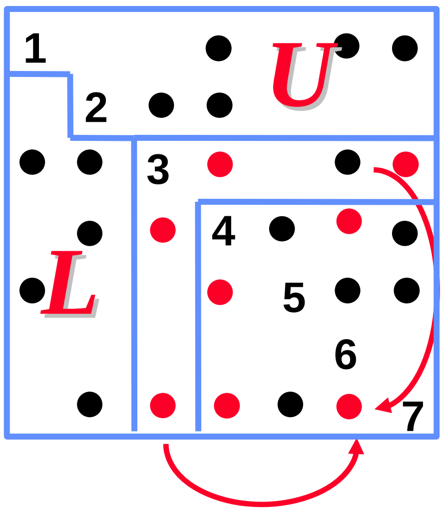
$$A = \begin{bmatrix} \alpha & w^T \\ v & B \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ v/\alpha & I \end{bmatrix} \cdot \begin{bmatrix} \alpha & w^T \\ 0 & C \end{bmatrix}$$

$$C = B - \frac{v \cdot w^T}{\alpha}$$

- Repeats GE on C
- Results in {L\U} decomposition (A = LU)
  - **L lower triangular with unit diagonal, U upper triangular**
- Then, x is obtained by solving two triangular systems with L and U

Sparse GE

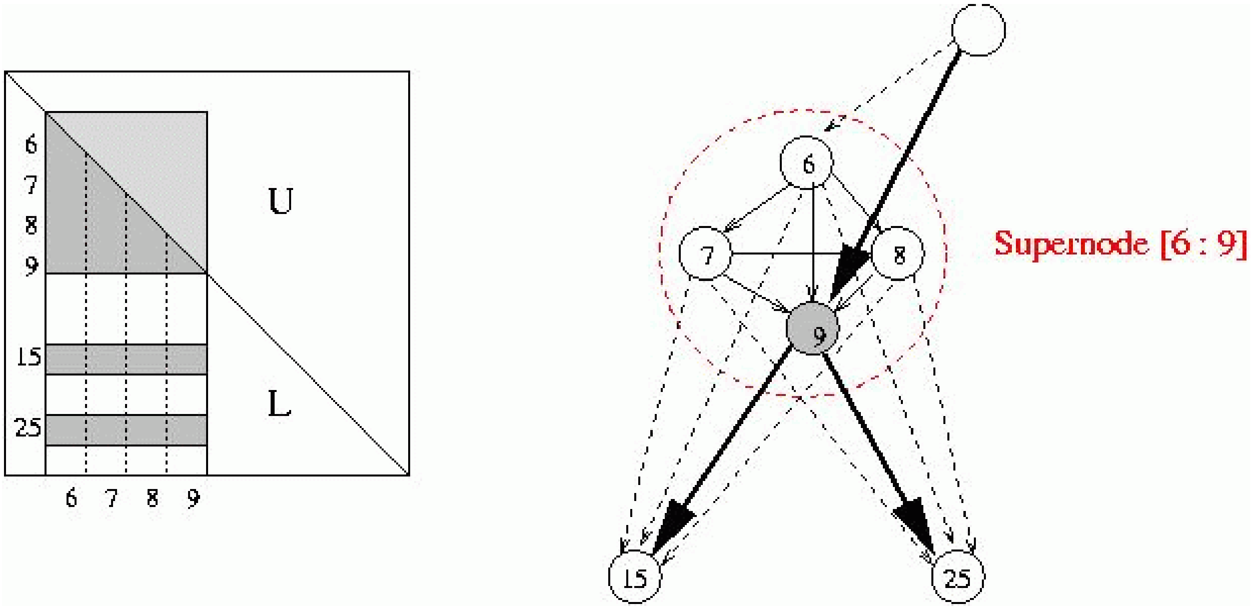
Scalar version : 3 nested loop




```
for i = 1 to n-1
    for j = i+1 to n
        A(j,i) = A(j,i) / A(i,i)
    for k = i+1 to n  s.t.  A(i,k) != 0
        for j = i+1 to n  s.t.  A(j,i) != 0
            A(j,k) = A(j,k) - A(j,i) * A(i,k)
```

Typical fill ratio: 10x for 2D problems, 30-50x for 3D problems

Supernode: dense blocks in {L\U}



- Good for high performance
  - **Enable use of Level 3 BLAS**
  - **Reduce inefficient indirect addressing (scatter/gather)**
  - **Reduce time of the graph algorithms by traversing a coarser graph**

SuperLU usage

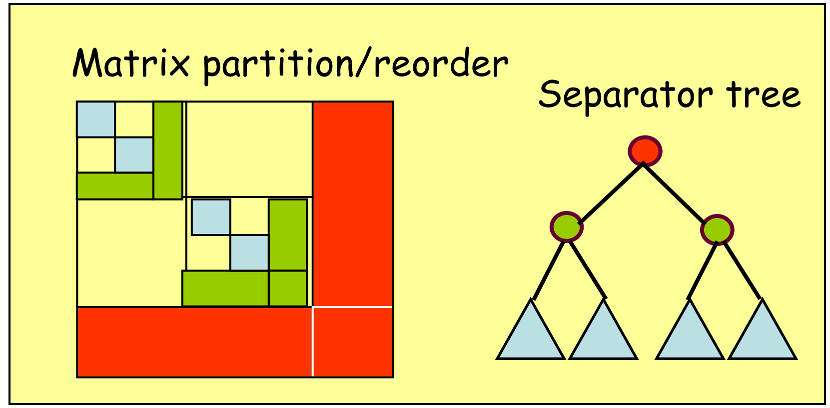
- Over 6000 downloads each year (FY2005, FY2006)
- Research
  - **In other DOE ACTS Tools: Hypre, PETSc, Overture, Trilinos**
  - **M3D-C1, NIMROD (fusion SciDAC)**
  - **Omega3P (accelerator SciDAC)**
  - **. . .**
- Industrial
  - **Cray Scientific Libraries**
  - **FEMLAB**
  - **HP Mathematical Library**
  - **IMSL Numerical Library**
  - **NAG**
  - **Sun Performance Library**
  - **Python (NumPy, SciPy extensions)**

SuperLU_DIST major steps
(parallelization perspectives)

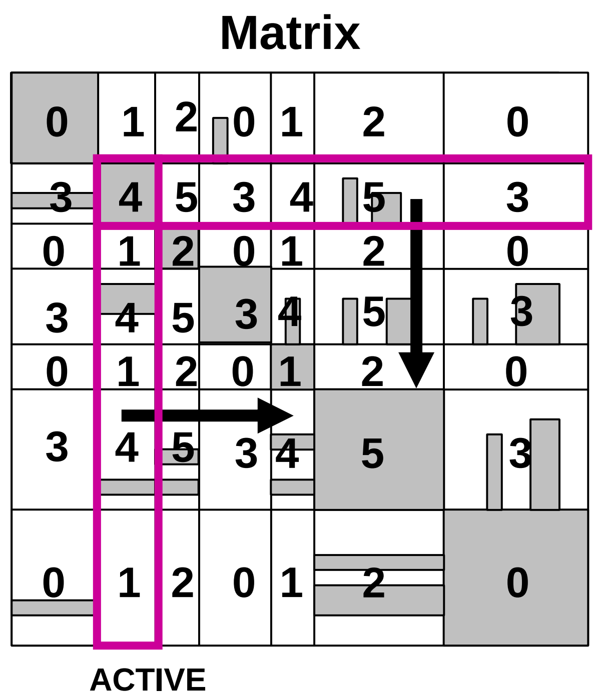
- Static numerical pivoting: improve diagonal dominance
  - **Currently use MC64 (HSL, serial)**
  - **Being parallelized [Riedy]**
- Ordering to preserve sparsity
  - **Can use parallel graph partitioning (e.g., ParMetis, Scotch)**
- Symbolic factorization: determine pattern of {L\U}
  - **Parallelized recently [Grigori]**
- Numerics: factorization, triangular solutions, iterative refinement (usually dominate total time)
  - **Parallelized a while ago; need to improve load balance, latency-hiding, . . .**

Data distribution

Ordering & Symbfact



Numerical phases



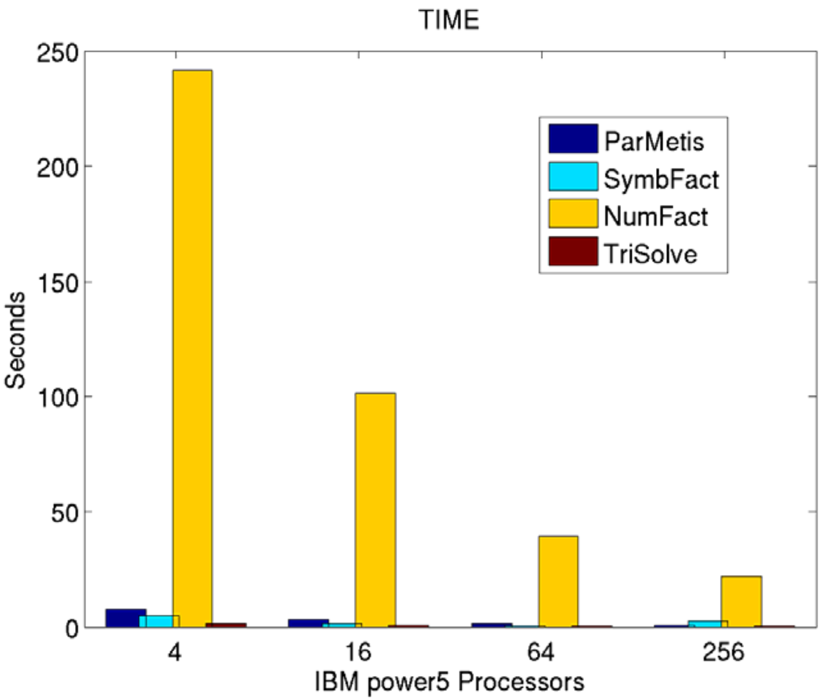
Processors

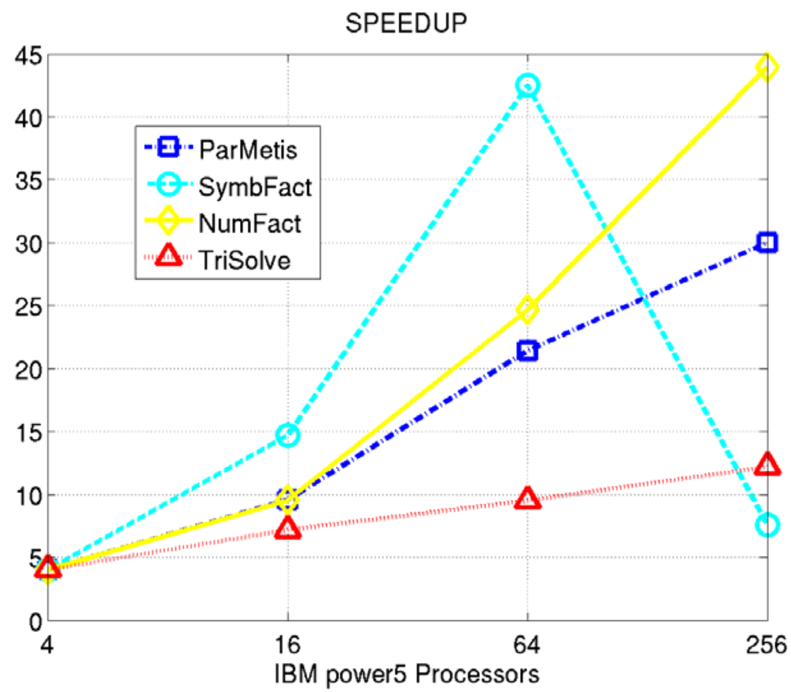
0	1	2
3	4	5

Time & speedup on IBM power5

Matrix181 (M3D-C1, Fusion), N=589,698, fill-ratio=9.3

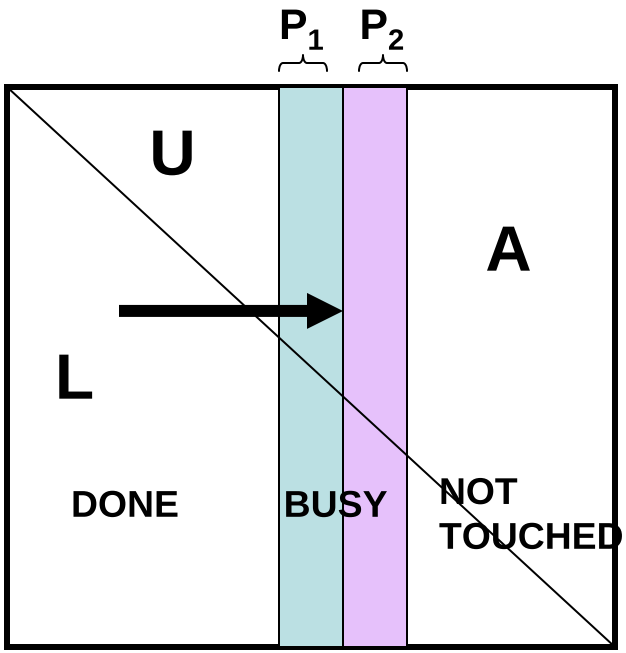
- **NumFact dominates total time, scales better**
- **How to make TriSolve scale better? . . . O(1) ops/message**

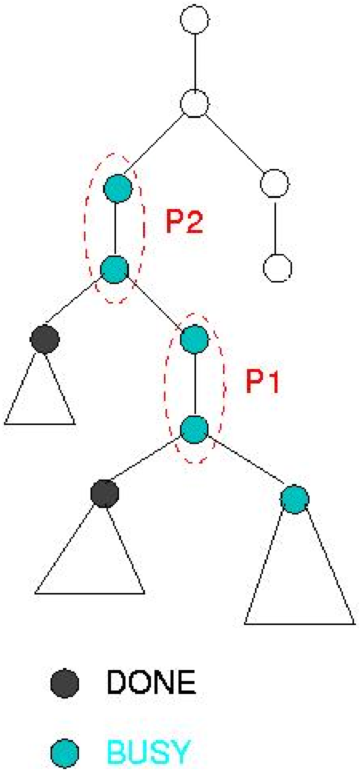




Multicore: SuperLU_MT [Li/Demmel/Gilbert]

- Pthread or OpenMP
- Left looking : many more reads than writes
- Use shared task queue to schedule ready columns in the elimination tree (bottom up)

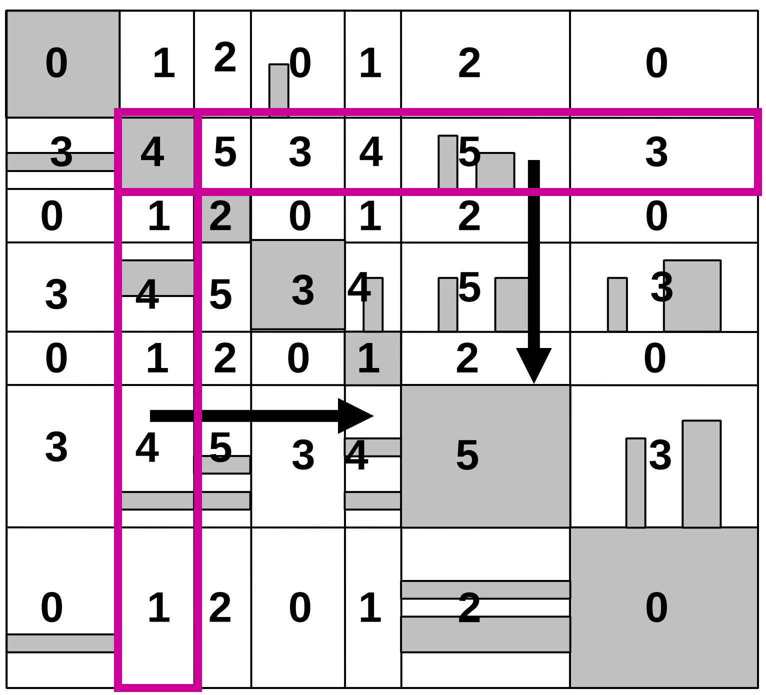




Multicore: SuperLU_DIST [Li/Demmel]

- MPI
- Right looking : many more writes than reads
- 2D block cyclic layout

Matrix



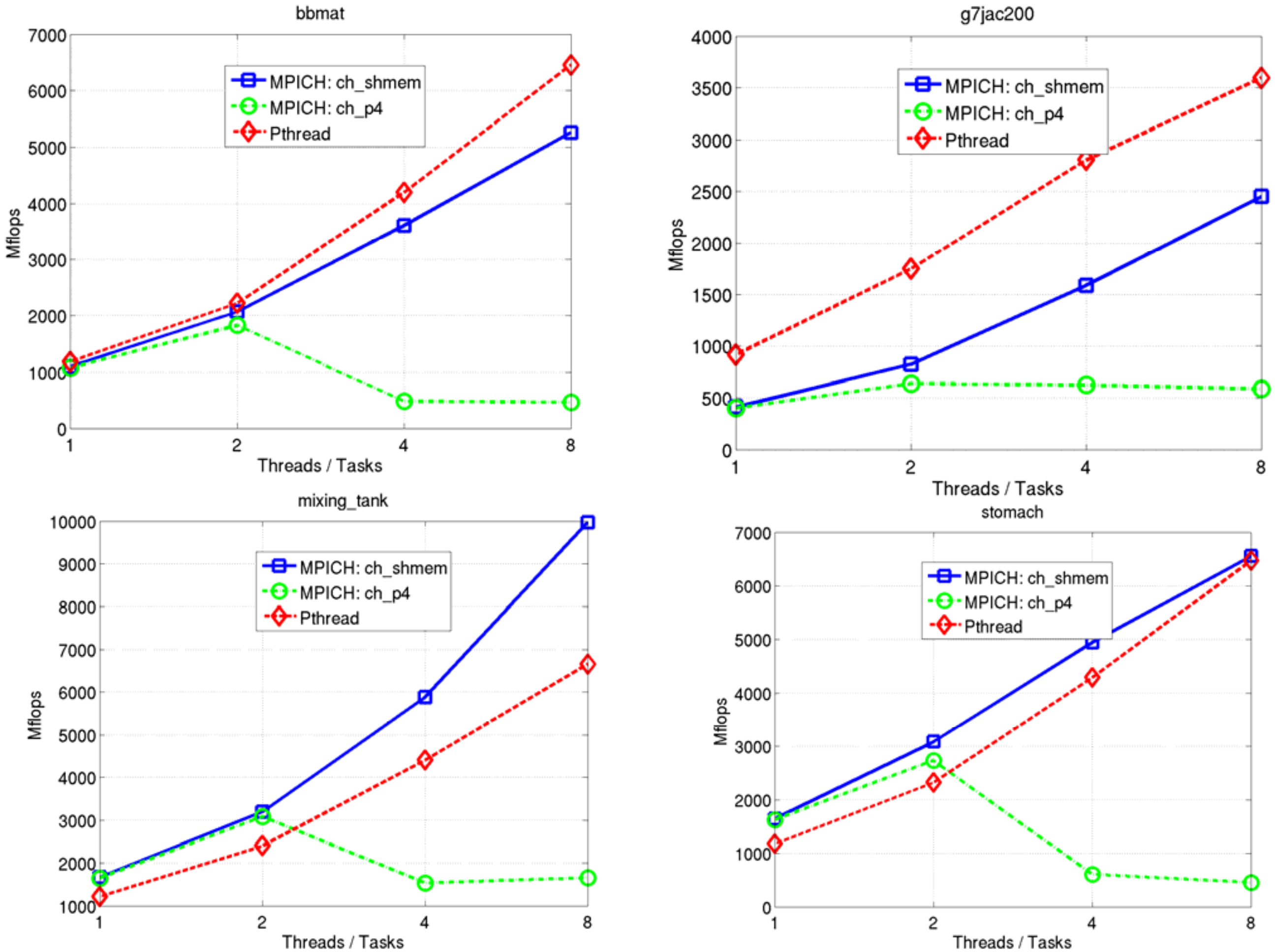
ACTIVE

Process mesh

0	1	2
3	4	5

Intel Clovertown: 2.33 GHz Xeon

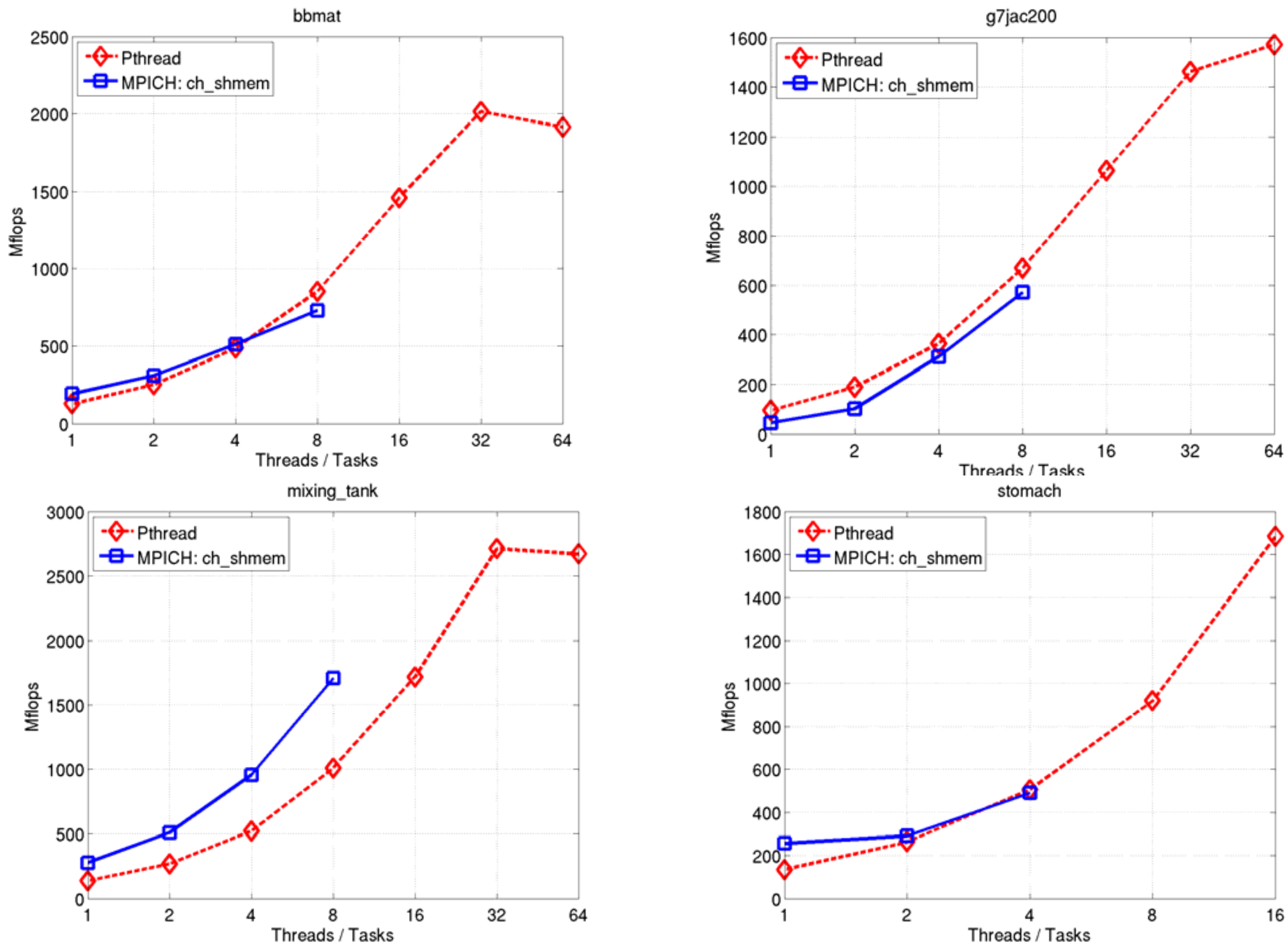
2 sockets X 4 cores,   L2 cache: 4 MB/2 cores



- Important to configure MPICH in shared memory mode
- Achieves only 13% peak

Sun Niagara2: 1.4 GHz UltraSparc T2

8 cores, 8 hw-threads/core, L2 cache shared: 4 MB



- Achieves 24% peak

Observations

- Explicit thread programming is beneficial
- MPI better be configured in both modes
- MPI + Pthread requires significant code rewriting

- How to do auto-tuning?