WHAT STARTS HERE CHANGES THE WORLD
THE UNIVERSITY OF TEXAS AT AUSTIN

DDDAS Project Update

D. Fuentes

Institute for Computational Engineering and Sciences
The University of Texas at Austin

Austin, Texas
February 1, 2007

Outline

- **Hybrid OpenMP/MPI Paradigm**

- **Code Demonstration**

  Data Transfer , Filtering, Visualization

- **Computations**

  Optimization window

OpenMP/MPI Paradigm

- **Computer Architecture at TACC is cluster of SMP nodes**

  Lonestar: 4 processors per compute nodes

- **Profiling shows linear solve time $\approx$ constant as increase processors for a given problem size**

- **Large number of processors communication between MPI tasks dominates solve time**

- **OpenMP Offers a way to avoid communication overhead**

  no communication between OpenMP threads, but Fork/Join overhead very expensive

OpenMP/MPI Paradigm

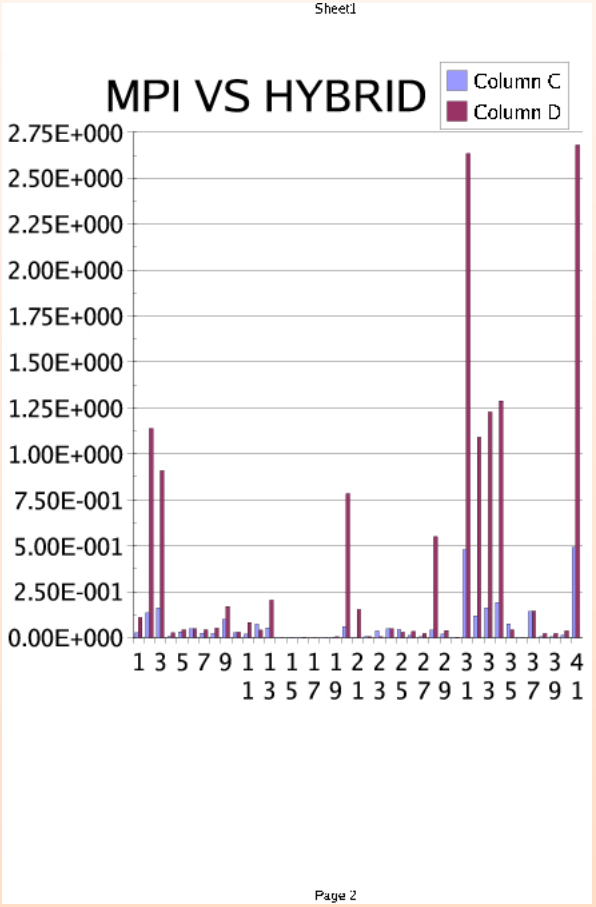
Sheet1

petsc to hp3d	1	2.59E-002	1.09E-001
assemble fnc	2	1.35E 001	1.14E 000
assemble jac	3	1.62E-001	9.10E-001
eval objective	4	6.17E-003	2.39E-002
elemfnc setup	5	3.17E-002	4.25E-002
elemfncsumfact	6	4.89E-002	4.74E-002
elemfnc bndry	7	2.37E 002	4.21E 002
elemjac setup	8	2.32E-002	5.19E-002
elemjacsumfact	9	9.98E-002	1.69E-001
elemjac bndry	10	2.52E-002	2.89E-002
elem tmp	11	1.87E 002	8.23E 002
VecMDot	12	7.30E-002	3.98E-002
VecNorm	13	5.33E-002	2.03E-001
VecScale	14	3.61E-004	1.50E-001
VecCopy	15	9.66E-005	2.80E-004
VecSet	16	8.41E 004	1.89E 003
VecAXPY	17	5.36E-005	1.13E-004
VecWAXPY	18	7.22E-005	1.61E-004
VecMAXPY	19	2.49E-003	4.81E-003
VecAssemblyBegin	20	6.11E-002	7.83E-001
VecAssemblyEnd	21	3.17E-004	1.53E-001
VecScatterBegin	22	6.55E-003	3.43E-003
VecScatterEnd	23	3.63E-002	1.04E-003
VecNormalize	24	4.85E-002	4.83E-002
MatMult	25	4.34E 002	2.94E 002
MatSolve	26	1.08E-002	3.29E-002
MatLUFactorNum	27	4.66E-003	2.24E-002
MatAssemblyBegin	28	4.12E-002	5.49E-001
MatAssemblyEnd	29	1.94E-002	3.83E-002
MatZeroEntries	30	6.16E 004	2.02E 003
SNESSolve	31	4.78E-001	2.63E-000
SNESLineSearch	32	1.19E-001	1.09E-000
SNESFunctionEval	33	1.62E-001	1.73E-000
SNESJacobianEval	34	1.91E 001	1.29E 000
KSPGMRESOrthog	35	7.44E-002	4.31E-002
KSPSetup	36	2.74E-005	4.08E-005
KSPSolve	37	1.41E-001	1.41E-001
PCSetUp	38	4.83E-003	2.27E-002
PCSetUpOnBlocks	39	4.76E 003	2.26E 002
PCApply	40	1.37E-002	3.66E-002
TaoAppObjective	41	1.90E-001	2.68E-000

Time for fork/join .1sec

Page 1

Sheet1



Page 2

Code Demonstration

- **Data Transfer scripts written in Python**

- **Bottleneck was writing to disk**

  Solution: collect onto one processor, collect into one buffer then write visualization files to disk

- **Computations**

  Optimization window

Computations

- **Optimization window**

  TWO groups of processors: 1st group is constantly calibrating and sending the calibrated parameters to 2nd group of processors. 2nd group constanly solving optimal control problem given calibrated data.

- **Time constraint resrictions imposed by real time calculation allow only a handful of objective function/gradient evaluations**

  Quasi Newton Methods taking too many iterations

  Steepest descent guarantees objective function decrease