The Integrative Role of COW's and Supercomputers in Research and Education Activities

Don Morton, Ganesh Prabu, Daniel Sandholdt, Lee Slater
Department of Computer Science
The University of Montana

Introduction

- Thesis – workstation clusters and supercomputers can be used together in environments that benefit everybody
- COW’s (e.g. Beowulf) – training and development activities in HPC
- Supercomputers (e.g. Cray T3E) – large-scale production runs

Acknowledgements

- Arctic Region Supercomputing Center
- SGI/CRI
- National Science Foundation
- Pallas

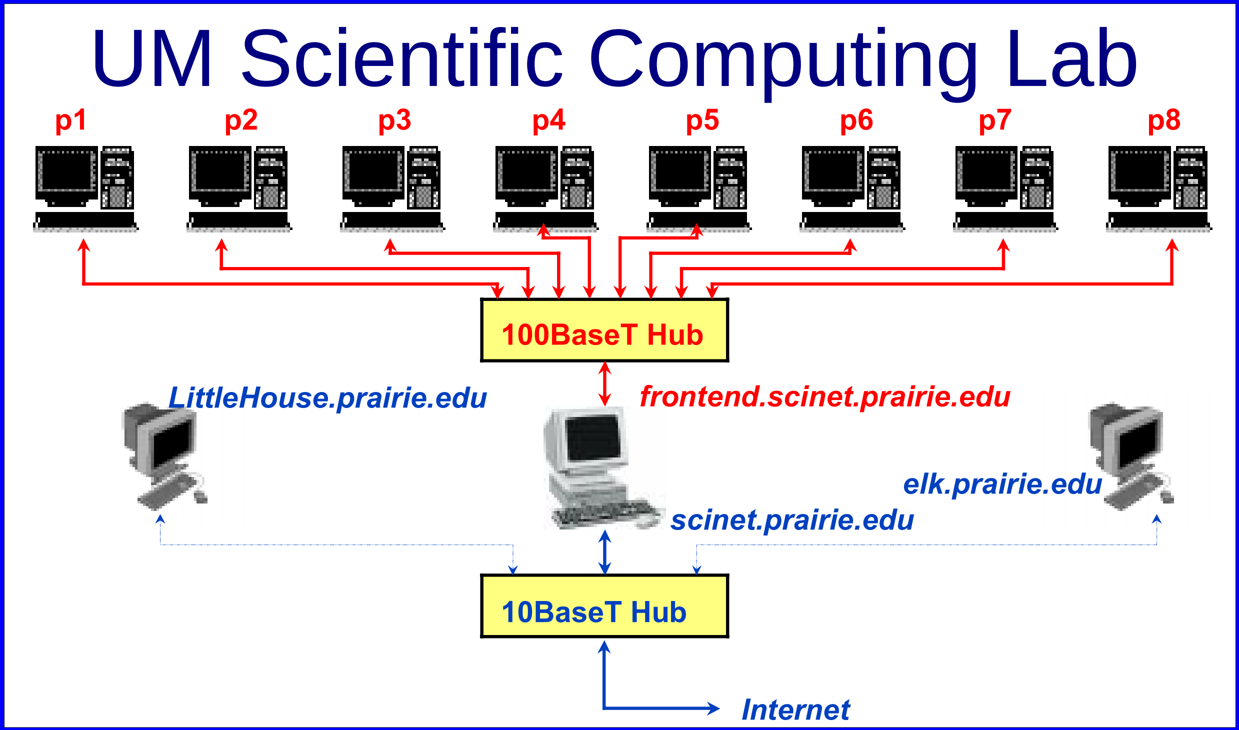
Outline

- Background
- Current Computing Environments
- Case Study – Parallel Programming Course
- Research and Development Activities
- COW/Supercomputer Integration Issues
- Conclusions

Background

- 1991 – 80486, Linux
- 1993–94 – PVM, RS6000, T3D
- 1994–97 – Cameron University, ARSC
- 1997–Present – U. Montana, ARSC

Current Computing Environments



Case Study – Parallel Programming Course

- Graduate (masters) course
- Goals
  - Hands-on experience using common, portable, programming tools
  - Explore concept of training on COW’s, then moving to supercomputers

Parallel Programming Course Outline

- Discuss basic concepts of parallel programming
- Implement solution to *n*–body problem with PVM, then MPI, then HPF
- Introduce performance analysis tools
- Lab session based on Linux/T3E portability issues
- Special projects

Lab Session – Linux/T3E

- Port Linux PVM *n*–body code to T3E PVM
- Port Linux MPI *n*–body code to T3E MPI
- Vampir analysis of MPI *n*–body code
- Performance modeling and analysis of MPI Jacobi program on T3E
- Analysis and improvement of an MPI code

Linux PVM to T3E PVM

- Network PVM and Cray MPP PVM have significant differences
  - Heterogeneous vs. Homogeneous SPMD
  - Dynamic vs. static task allocation
  - Cray-specific PVM calls
  - Need to be aware of different size datatypes
- Portable codes must be written in SPMD, with conditional compilation

Conditional Compilation for Portable PVM

```
#ifdef _CRAYMPP
// In Cray MPP, the "global" group is indicated by null pointer
#define GROUPNAME (char *) 0
#else
#define GROUPNAME "alltasks"
#endif

......

#ifdef _CRAYMPP
// Cray MPP does not support joining a "global" group, so we simply
// use the Cray-specific routine for getting the PE number
mype = pvm_get_PE(mytid);
#else
mype = pvm_joingroup(GROUPNAME);
#endif
........

#ifndef _CRAYMPP
// This is not executed for Cray MPP PVM - pvm_spawn() is not
// implemented - all tasks startup SPMD at beginning
if(mype == 0)  // I'm the master, spawn the others
  info = pvm_spawn(argv[0], (char**) 0, PvmTaskDefault, (char*) 0,
       ntasks-1, &tid_list[1]);
#endif
```

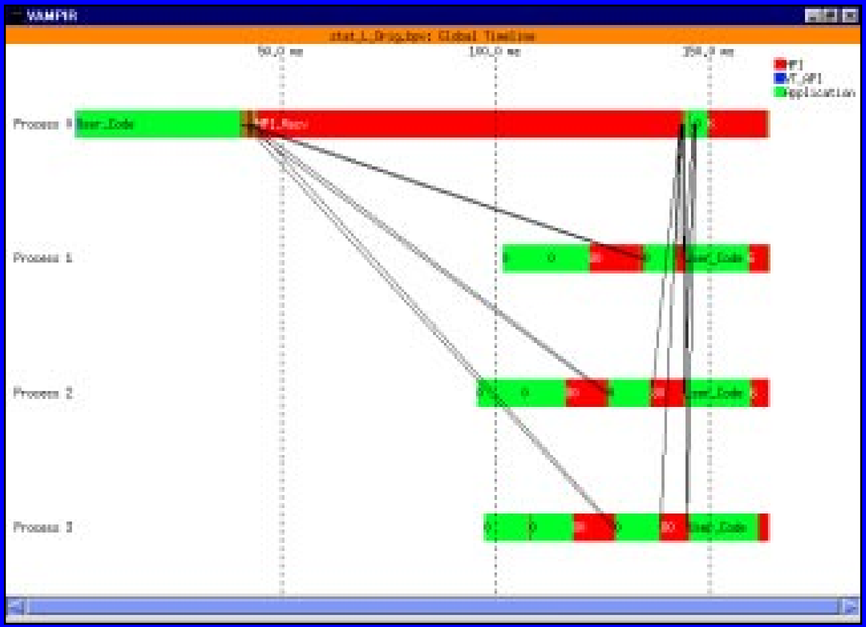
Comments on Porting PVM and MPI Codes

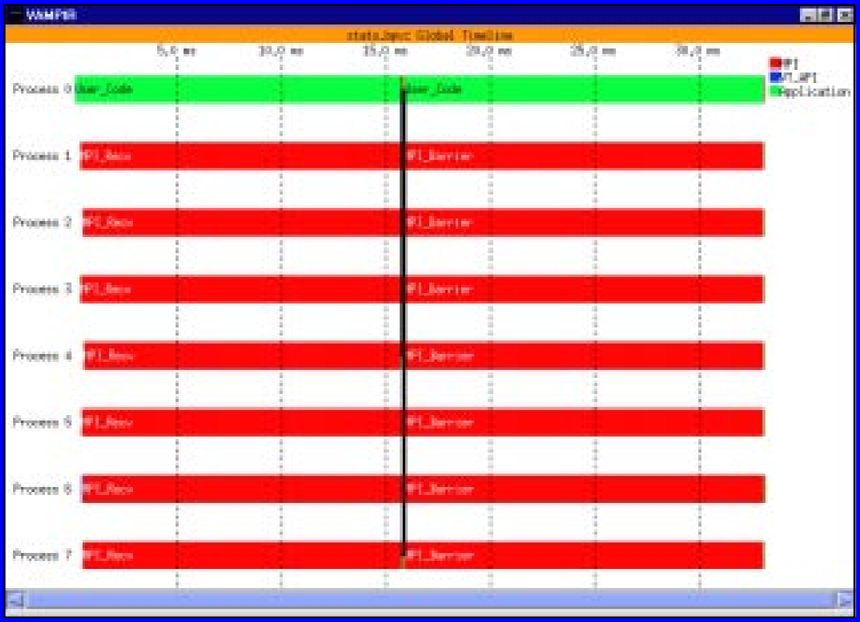
- PVM difficult to port, until network vs. Cray MPP differences are understood
- MPI ports easily
- Cray MPP is less forgiving of programmer errors than other systems
- In general, experienced students found transition from Linux to T3E straightforward

Performance Analysis

- Use of *Vampir* as a common tool
  - Vampirtrace – library of routines for generating tracefiles
  - Vampir – viewer for looking at tracefiles

Vampir

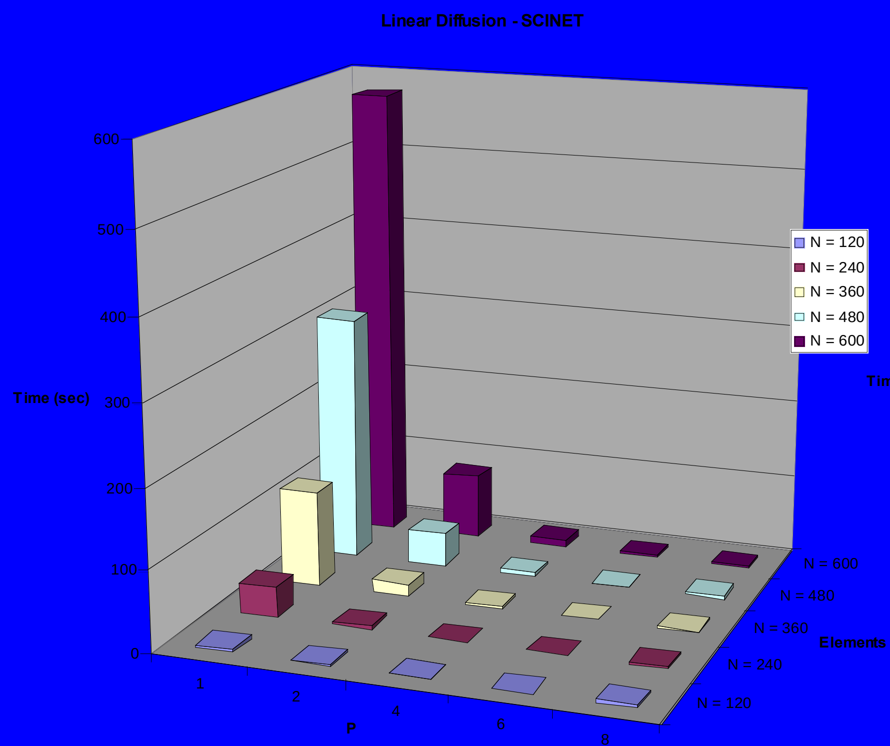


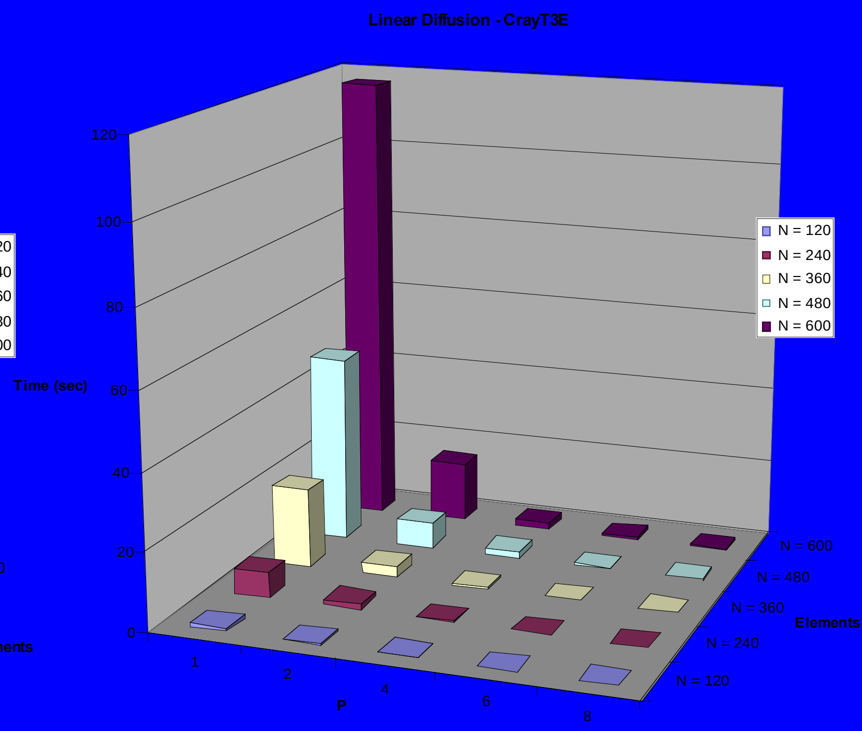


Special Projects

- Conversion of C++ MPI Jacobi program to Fortran
- Conversion of C++ MPI Jacobi program to C++ PVM
- Porting of Linux C++ parallel finite element code to T3E

Porting of Linux C++ Parallel Finite Element Code to T3E

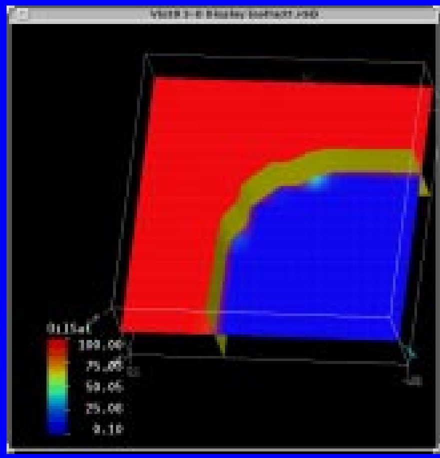


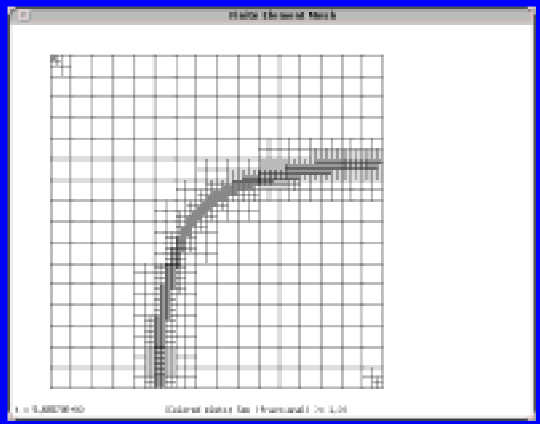


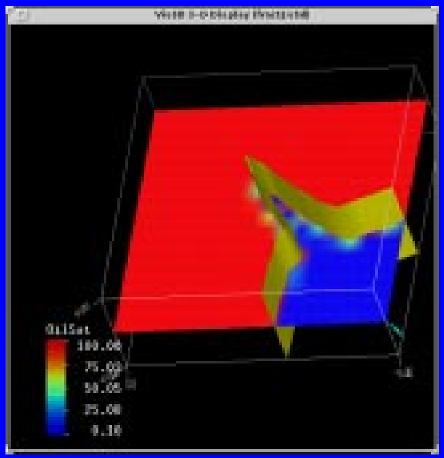
Research and Development Activities

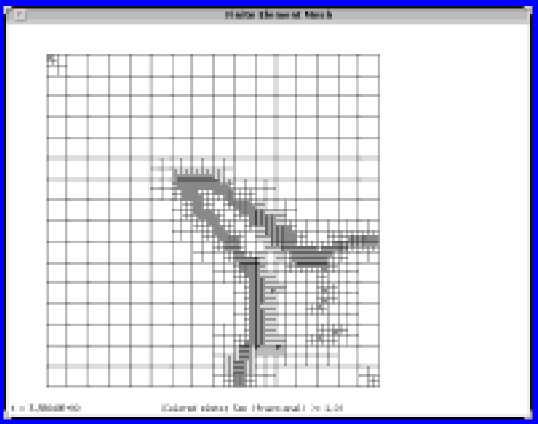
- Parallel, adaptive, finite element methods
- Parallelisation of hydrologic model for arctic ecosystems
- Coupling of parallel thermal and hydrologic models

Parallel, Adaptive Finite Element Methods





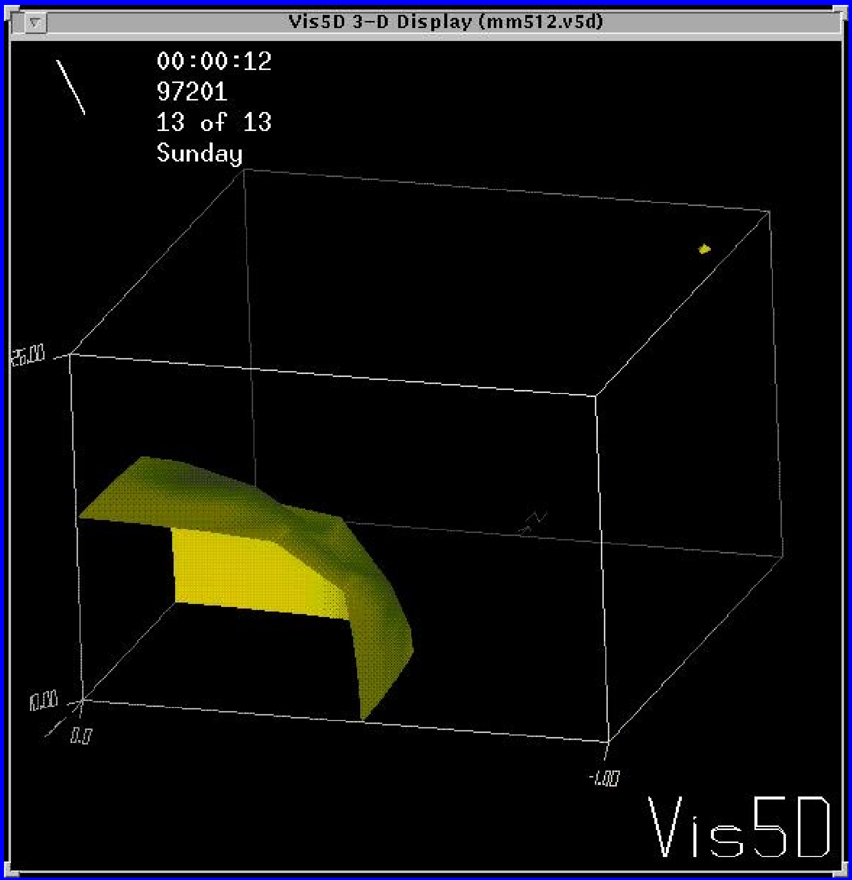


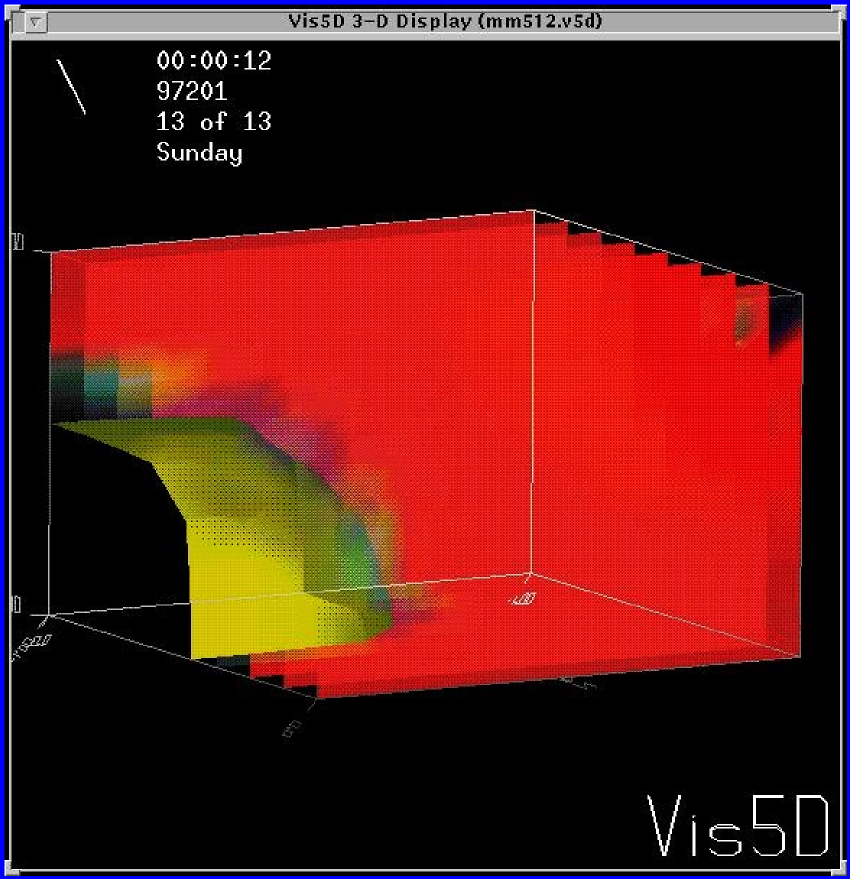


Homogeneous absolute permeabilities.

Heterogeneous absolute permeabilities.

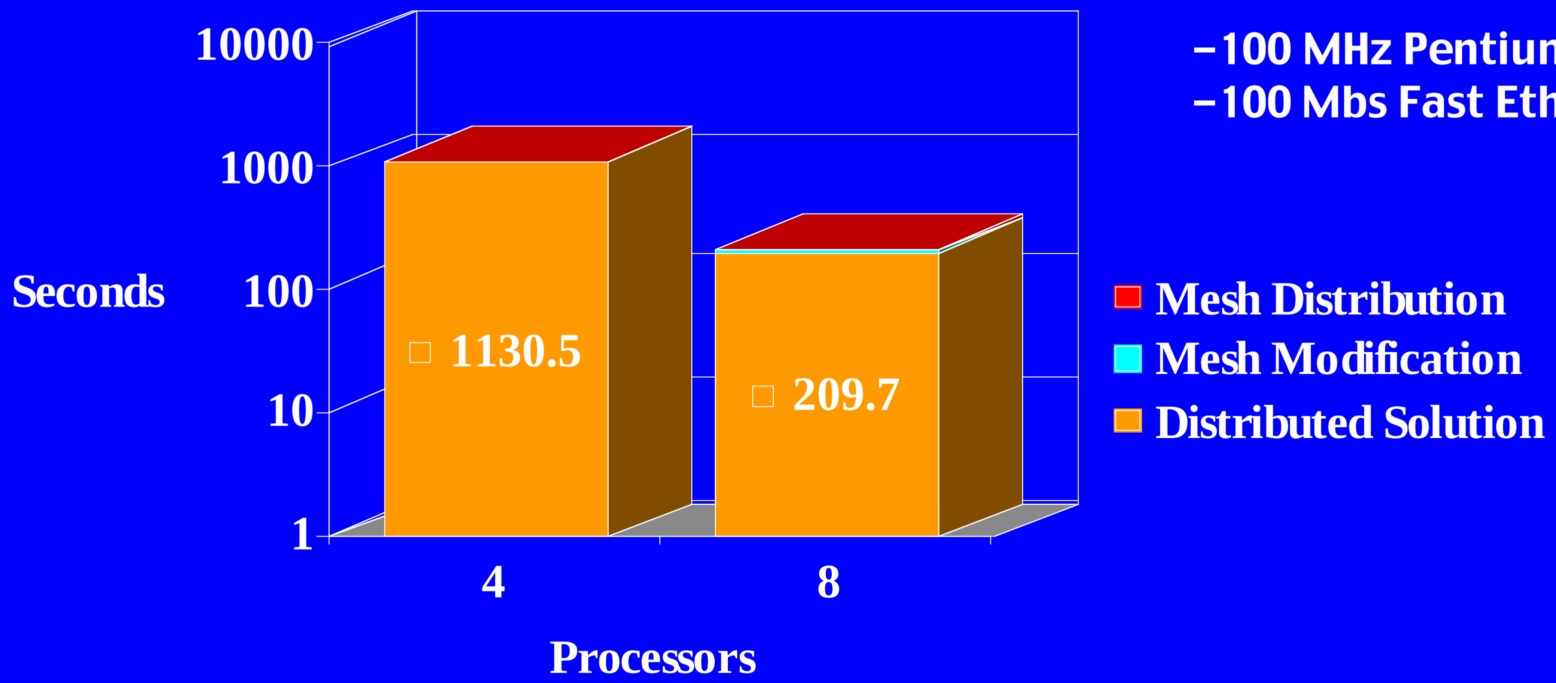
3D Isosurface (Oil/Water Interface)





Timings - Linux Cluster

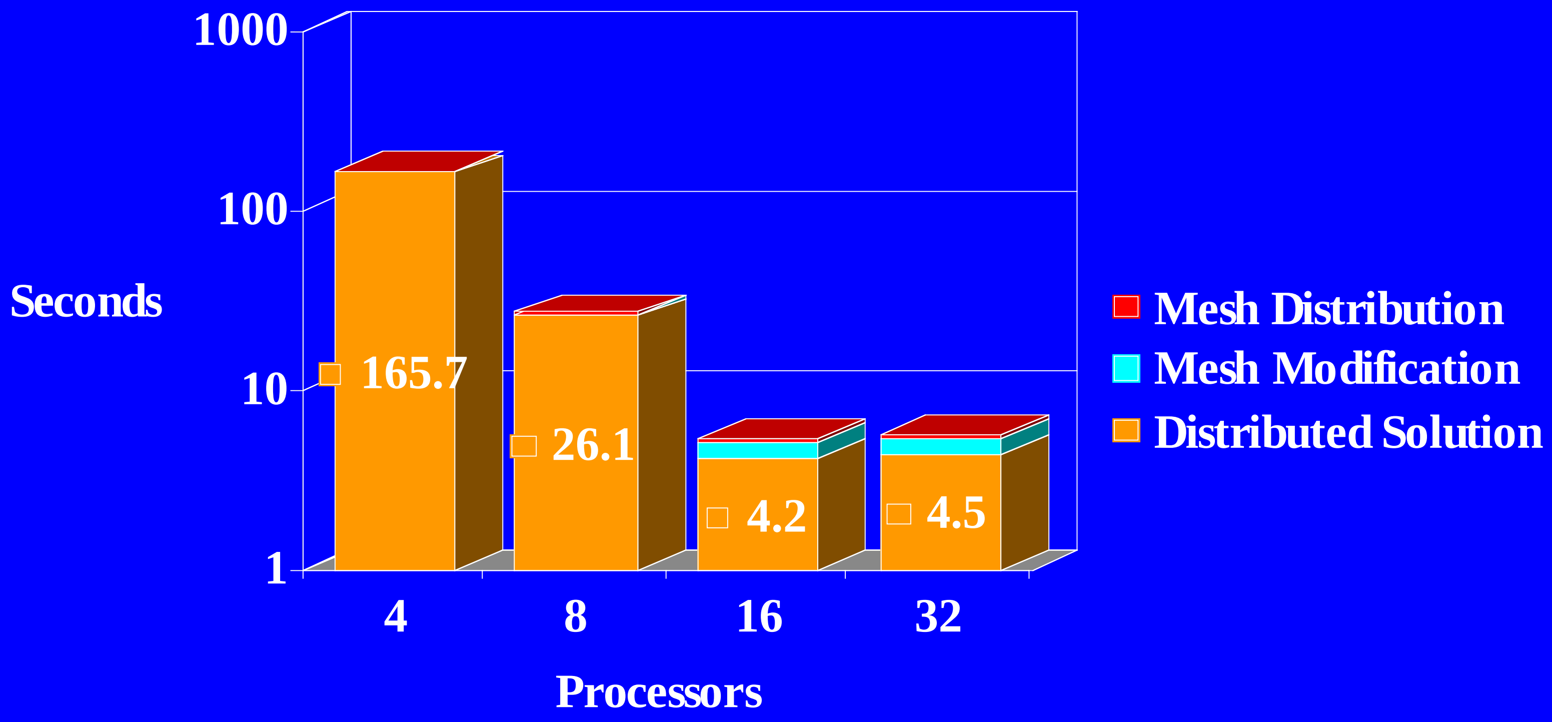
Wall time (seconds) required for single timestep with 4548 unknowns.



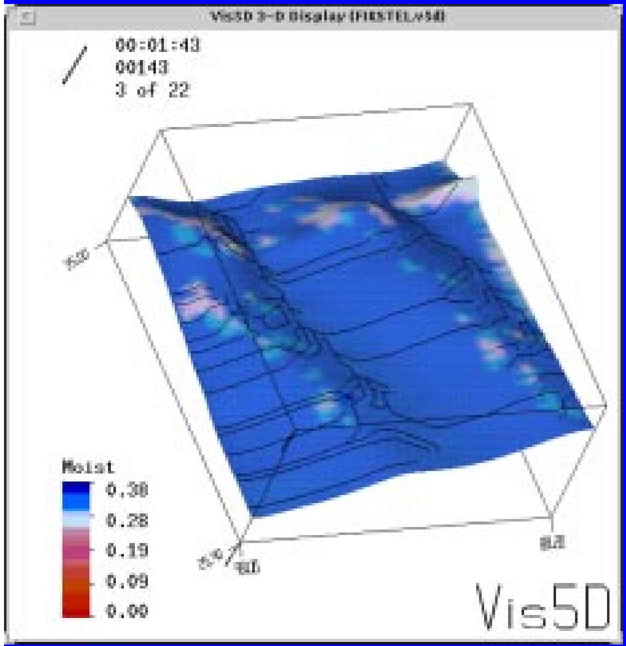
-100 MHz Pentiums
-100 Mbs Fast Ethernet

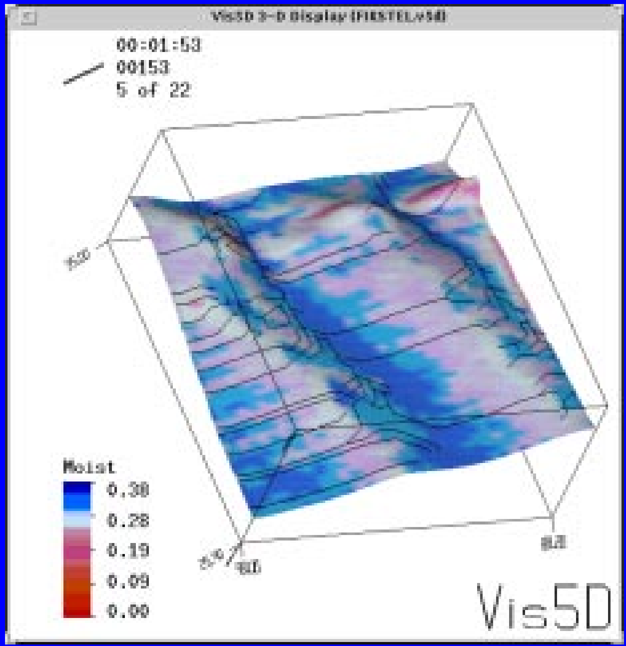
Timings - Cray T3E

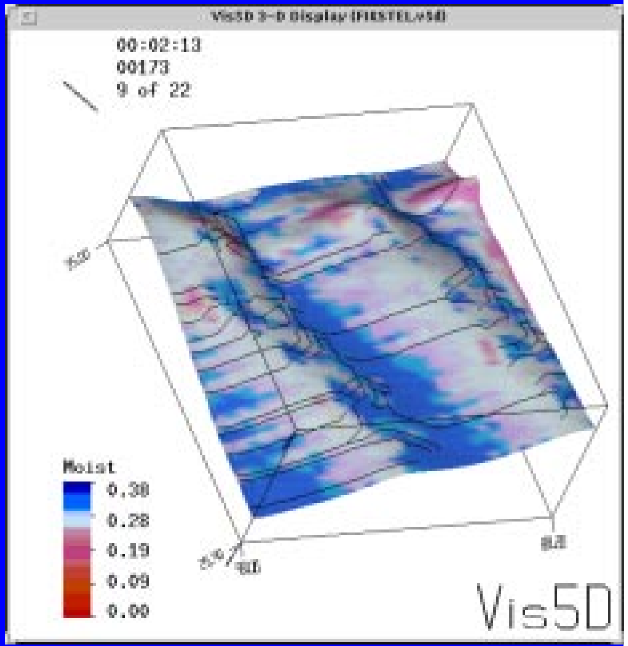
Wall time (seconds) required for single timestep with 4548 unknowns.



Parallelisation of Hydrologic Model

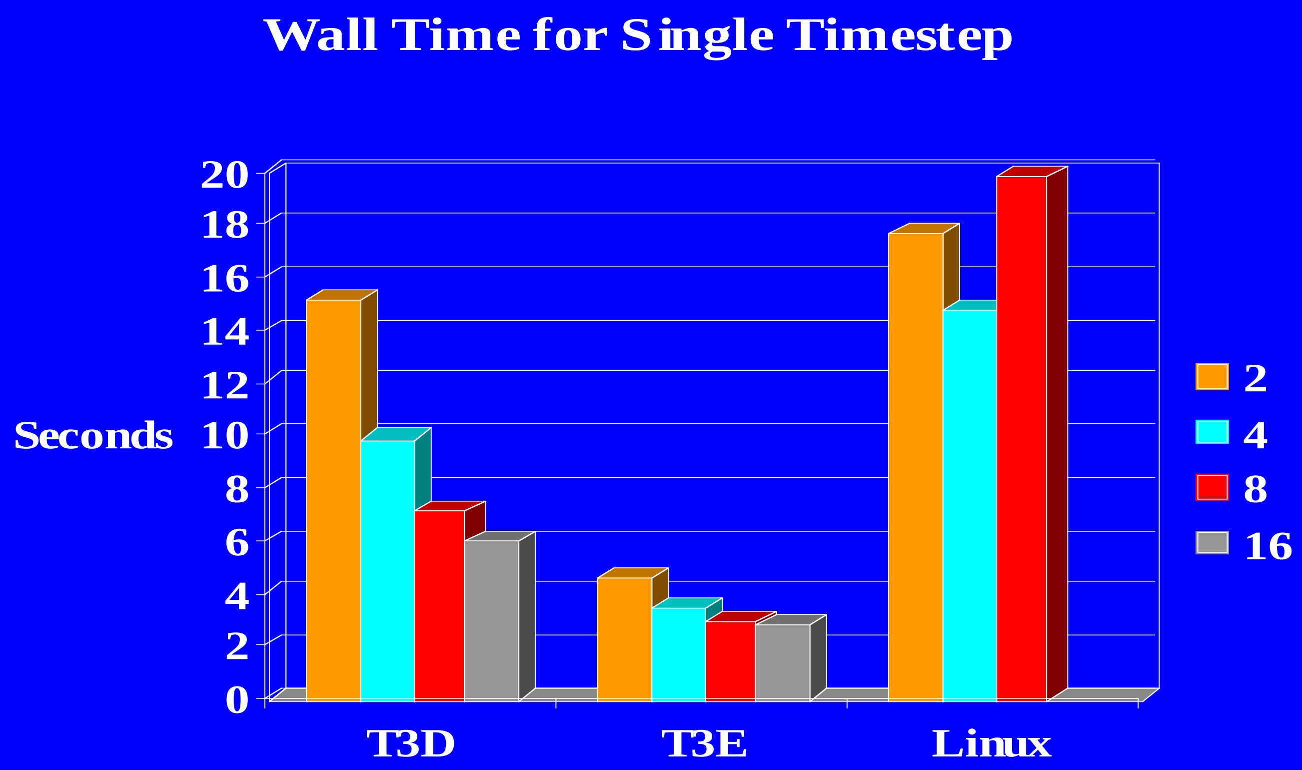






Time Measurements

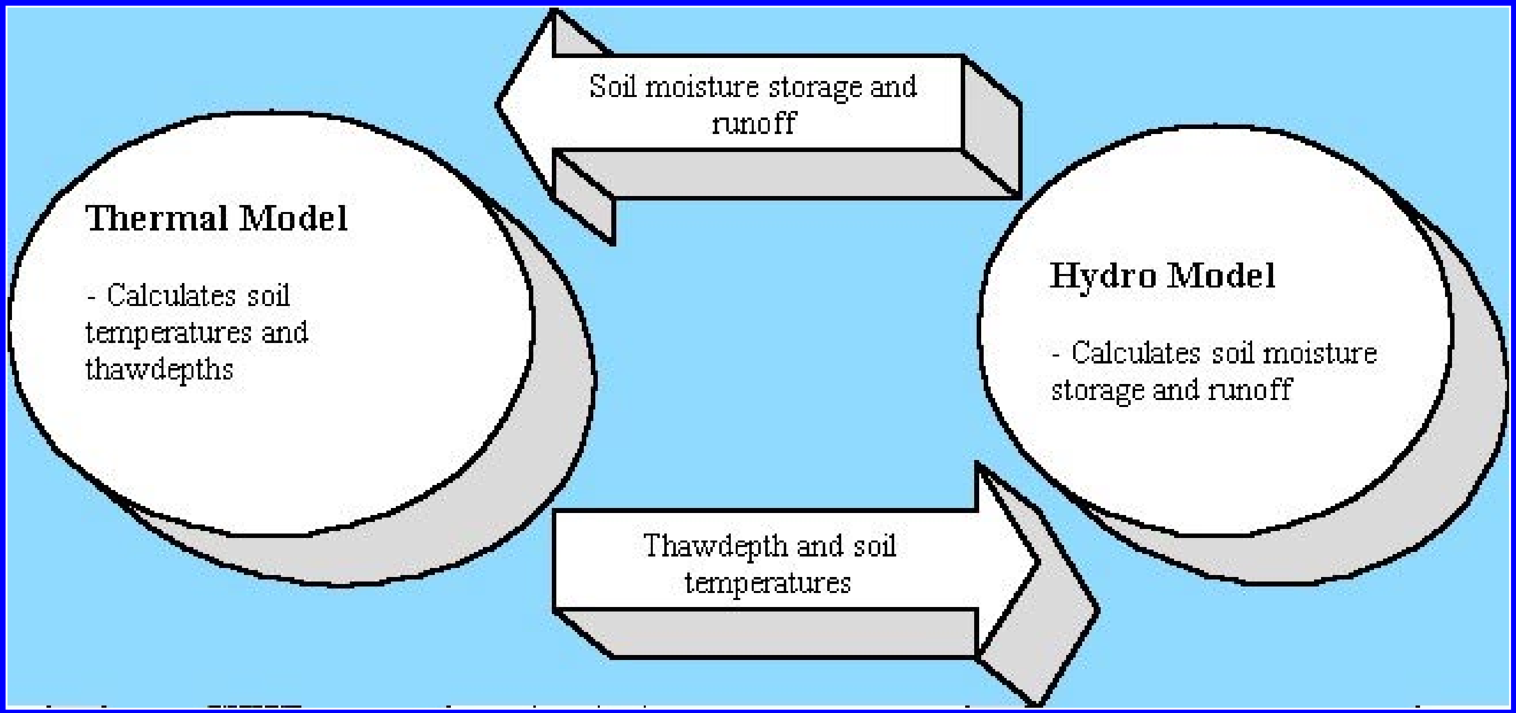
- 6448 elements
- Use of MPI+METIS+Shmem on Cray, MPI+METIS on Linux



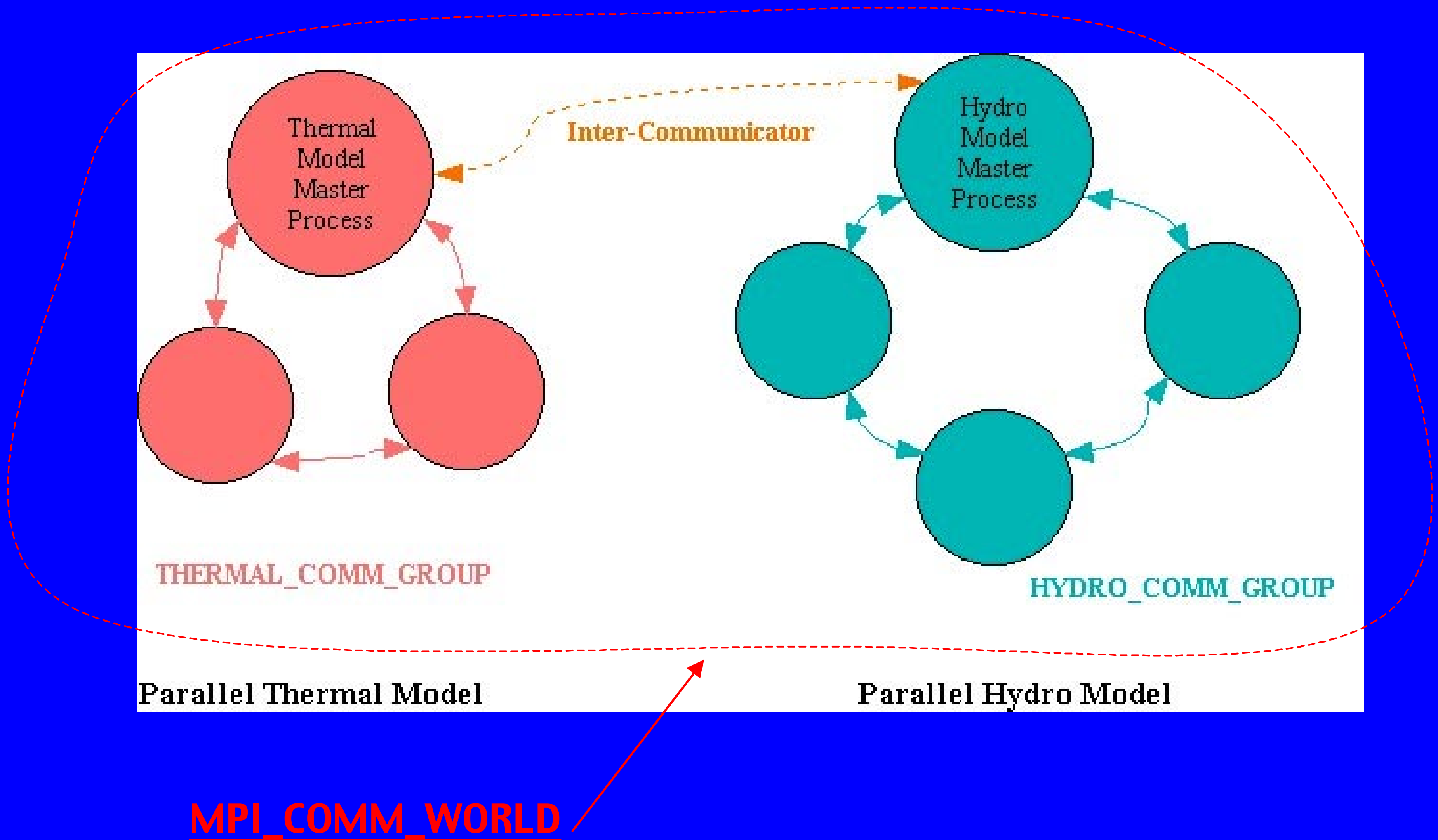
Coupling of Thermal and Hydro Models

- Background – previously existing hydro and thermal models
- Benefits of coupling – increased detail, capture feedback loops inherent in arctic ecosystems

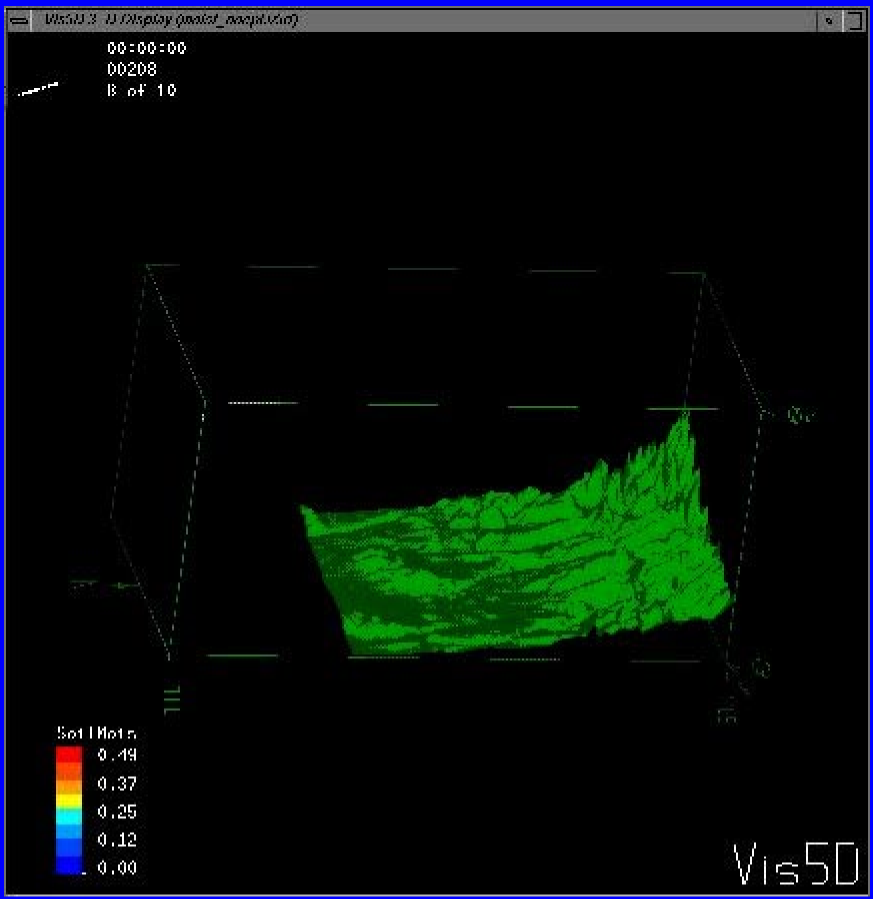
Coupled Models

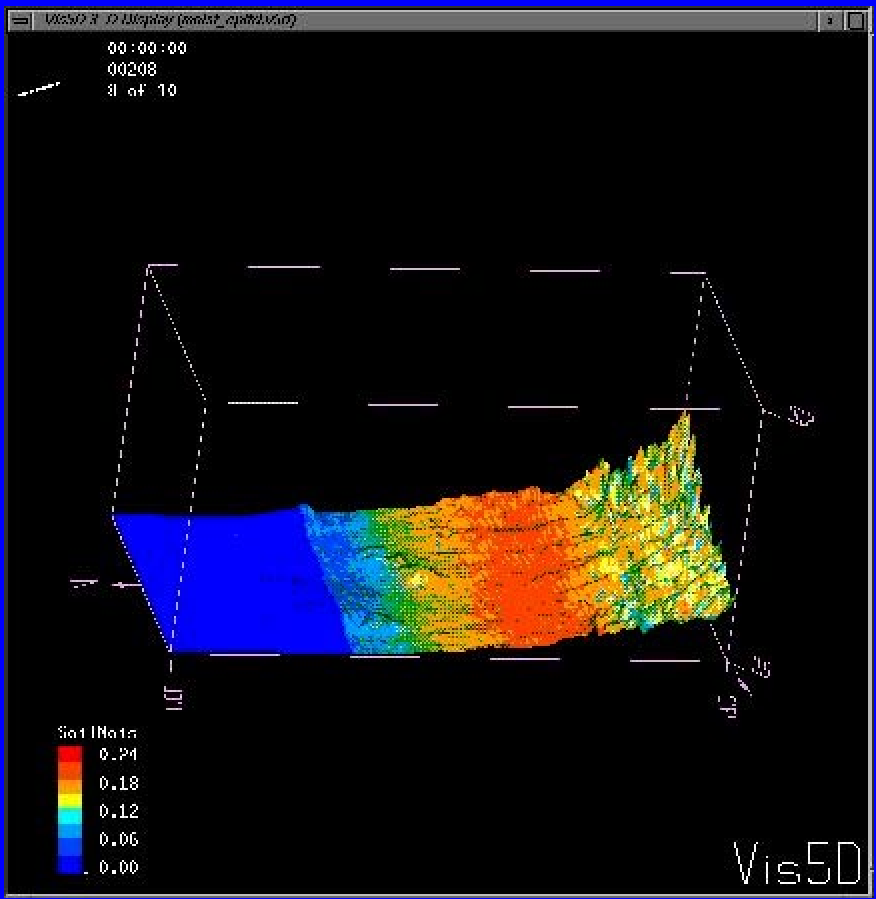


MPI Inter-communicators



Non-coupled vs. Coupled Simulation





COW/Supercomputer Integration Issues

- Code written on COW's should run on the T3E, and vice versa
- Integration should focus on creating similar programming environments
  - Users should be able to run programs identically on COW's and supercomputers
  - Scripts (mostly on COW side) can aid in this

COW/Supercomputer Integration Issues (continued)

- Portable analysis tools (e.g. Vampir, pgprof)
- Affordable, portable, integrated debuggers (Totalview?)

Conclusions

- COW's and supercomputers have complementary roles in HPC
- Local COW's are ideal training and development platform
- Supercomputers always needed
- Increased usage of COW's for training and development should result in more HPC experts, and greater demand for supercomputers