Computational Fluid Dynamics Applications on the Cray X1 Architecture: Experiences, Algorithms, and Performance Analysis

Andrew A. Johnson, Ph.D.
Army HPC Research Center
Network Computing Services, Inc.
Minneapolis, Minnesota
ajohn@ahpcrc.org

Outline

- AHPCRC Background
- CFD Code Description
- Cray X1 Overview
- Porting and Code Modifications
- Processor Performance
- Scalability and Network Performance
- Final Thoughts

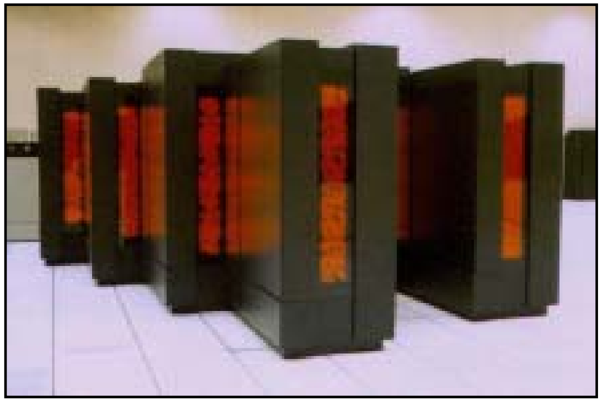
AHPCRC Background

- Program is in its 13th year
- Funded by the U.S. Army through the Army Research Laboratory
- Funding for hardware acquisition is through the Department of Defense High Performance Computing Modernization Program

- Government-University-Industry partnership for research and development of HPC applications and systems
  - University of Minnesota
  - Clark Atlanta University, Jackson State University, Howard University, Florida A&M University, University of North Dakota
  - Network Computing Services, Inc.

AHPCRC Background (continued)

- The Cray X1 is the center's "third generation" parallel computing system
  - All systems are heavily used by AHPCRC, Army, and DOD researchers



- Thinking Machines CM-5 (1991 – 1998)
  - Serial Number 1
  - 896 processors
- Cray T3E-1200 (1998 – Today)
  - 1088 processors
- Cray X1 (2002 – Today)

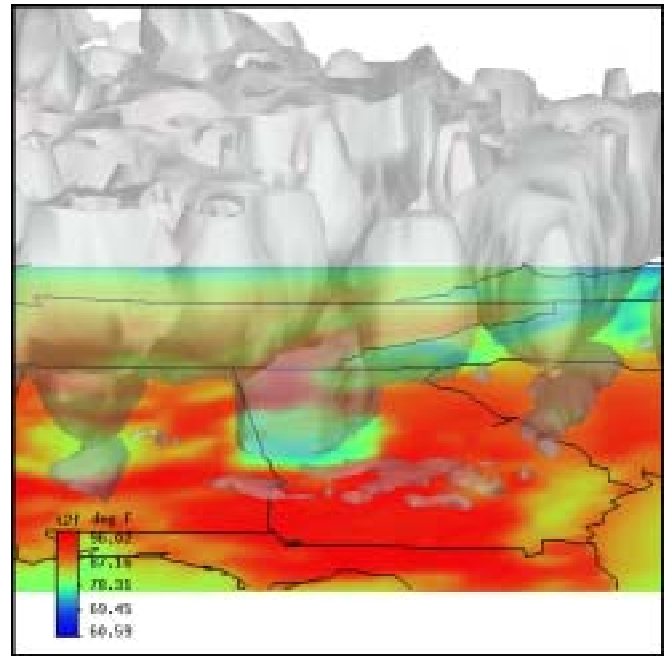
- Other Smaller Systems
  - CM-2, SGI Onyx, IBM SP

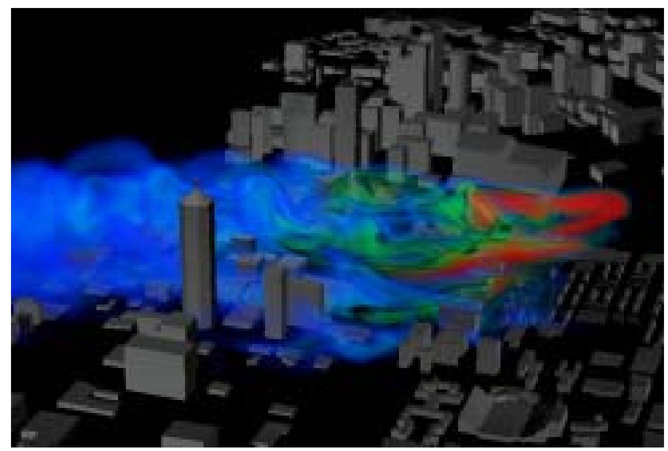
AHPCRC Background (continued)

- AHPCRC / NetworkCS, Inc. is the first non-classified site to receive Cray X1 systems
- Two Early Production (EP) systems
  - Installed on September 27, 2002
  - Air Cooled (AC) systems (16 CPU each)
  - Half clock and not “final” processor design
- Full Production system
  - Installed February 20, 2002
  - Liquid Cooled (LC) system
  - Half populated (32 CPU total)
- Planned upgrades
  - Full LC cabinet (64 CPU) May 2
  - Second full LC cabinet June…
  - Single system image (128 CPU) ~August
  - AC system upgrade June…

AHPCRC Background (continued)

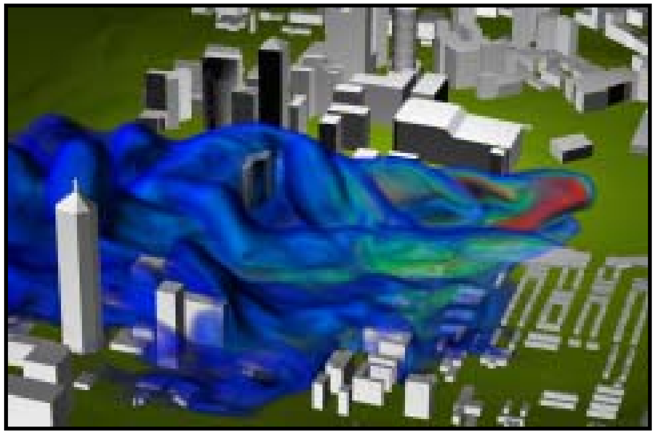
- Targeted Applications
  - Weather Modeling and Forecasting
    - MM5 & WRF
  - Projectile / Target Interactions
    - EPIC
    - Example: Behavior of ceramic-based armor
  - Computational Fluid Dynamics
    - Example: Contaminant dispersion in urban environments
  - Others
    - Computational Chemistry
    - Electromagnatics & Signature Modeling
    - Other Enabling Technologies





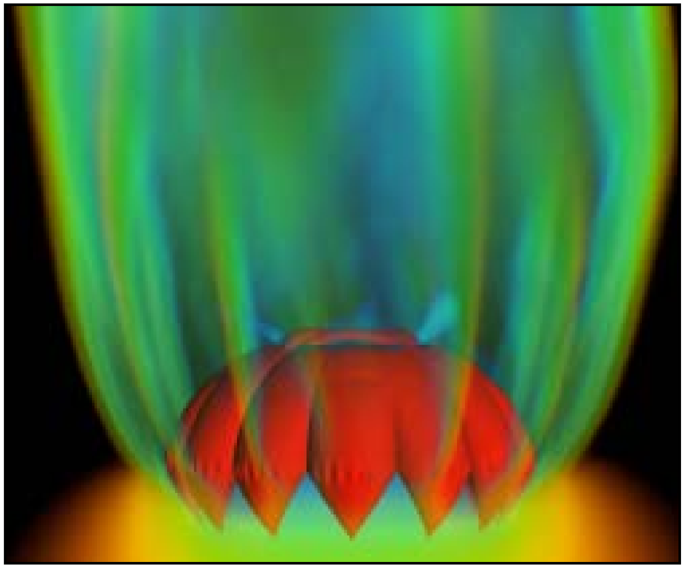
CFD Code Description

- Time accurate incompressible flow solvers built for unstructured meshes
- Developed at the AHPCRC
- In use for over 10 years
- Fully parallel and scalable based on MPI



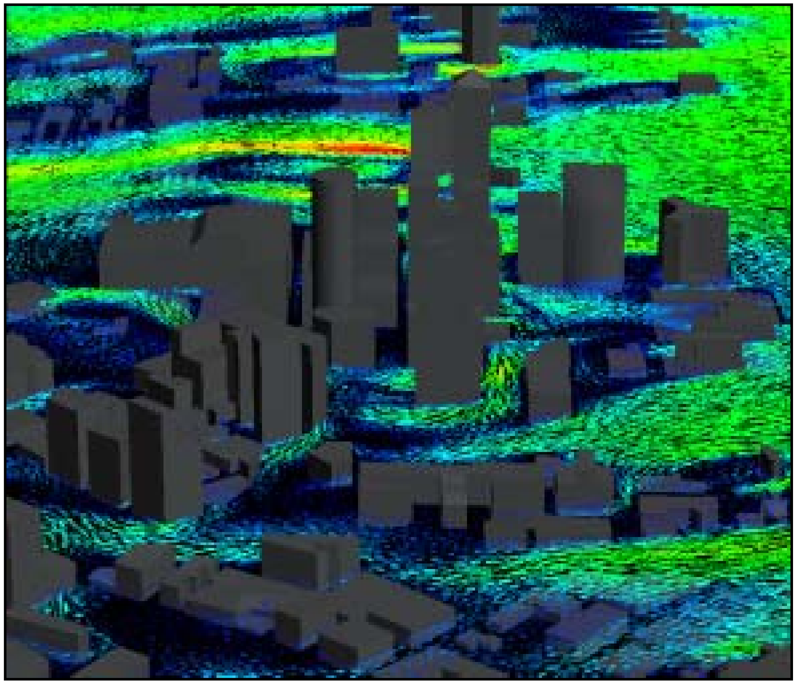
Contaminant dispersion in urban environments (Clark Atlanta University / AHPCRC)

- Specific code being tested is called "**BenchC**"
  - Representative of most CFD codes at the AHPCRC
  - Written entirely in C, roughly 6,700 lines
  - Built-in performance monitoring



Parachute aerodynamics.

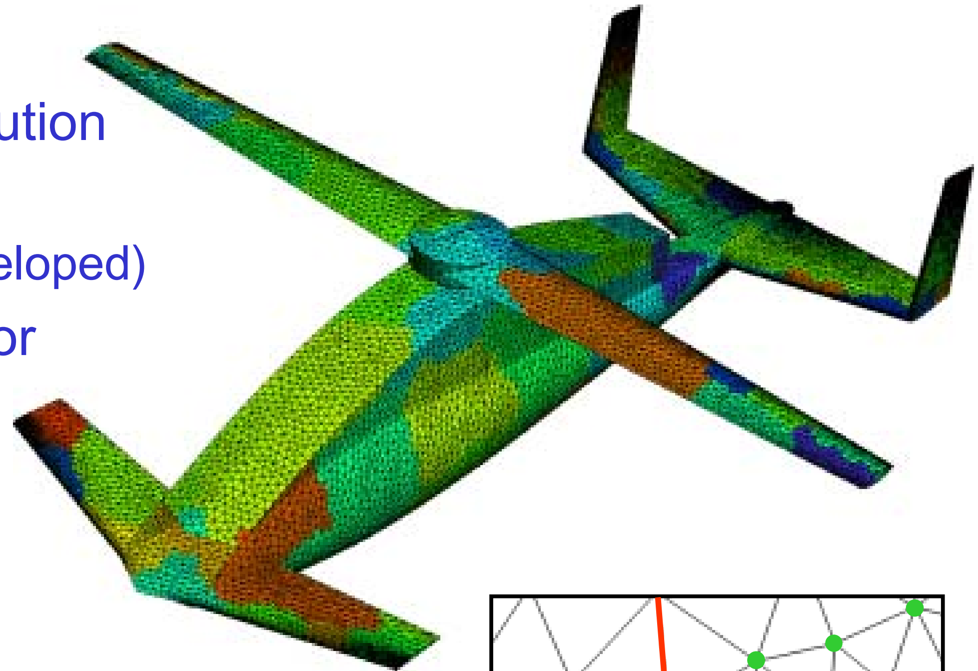
CFD Code Description (continued)

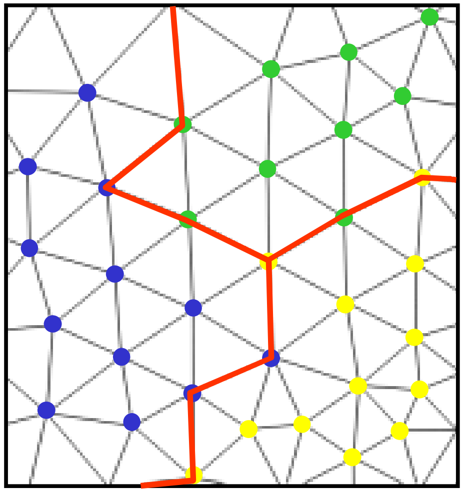


- Incompressible Flow
  - Navier-Stokes equations
    - Velocity and Pressure
  - LES turbulence model
- Finite Element Based
  - Fully unstructured meshes
    - Tetrahedral (4-nodded) elements
  - Fully stabilized (SUPG/PSPG)
  - Linear basis functions for all variables
- Fully Coupled Equation System
  - GMRES iterative solver
  - Matrix-free strategy
    - Direct formation of matrix-vector product when required
      - Vector based algorithm
    - Low memory requirements

CFD Code Description (continued)

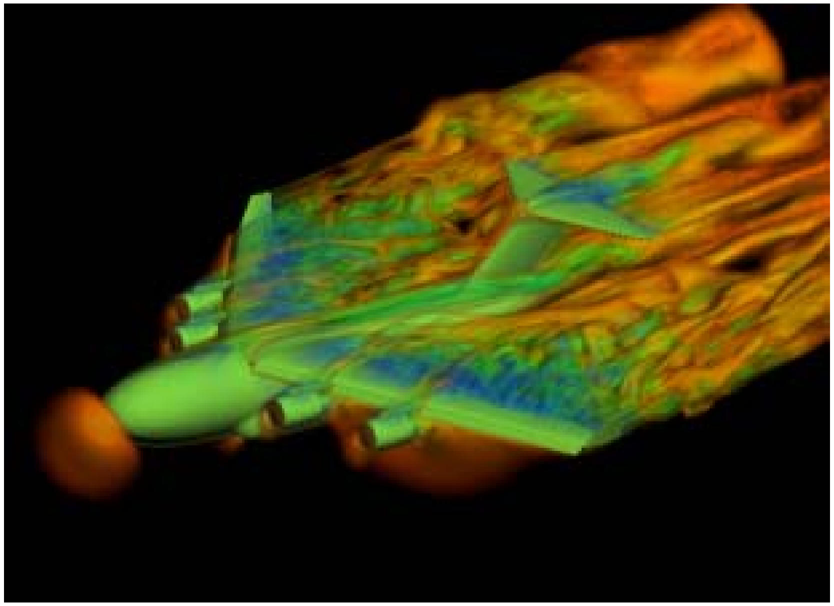
- Fully Parallel Based on MPI
  - Mesh partitioning and re-distribution
    - In-house RCB algorithm
    - ParMetis (AHPCRC/UMN Developed)
  - Fast and efficient inter-processor communication
    - Non-blocking routines
  - Fully scalable
    - Both computation and memory
- Portable to All Parallel Systems
  - Requires only C and MPI
  - Tested on X1,T3E,SGI,IBM,PC Clusters



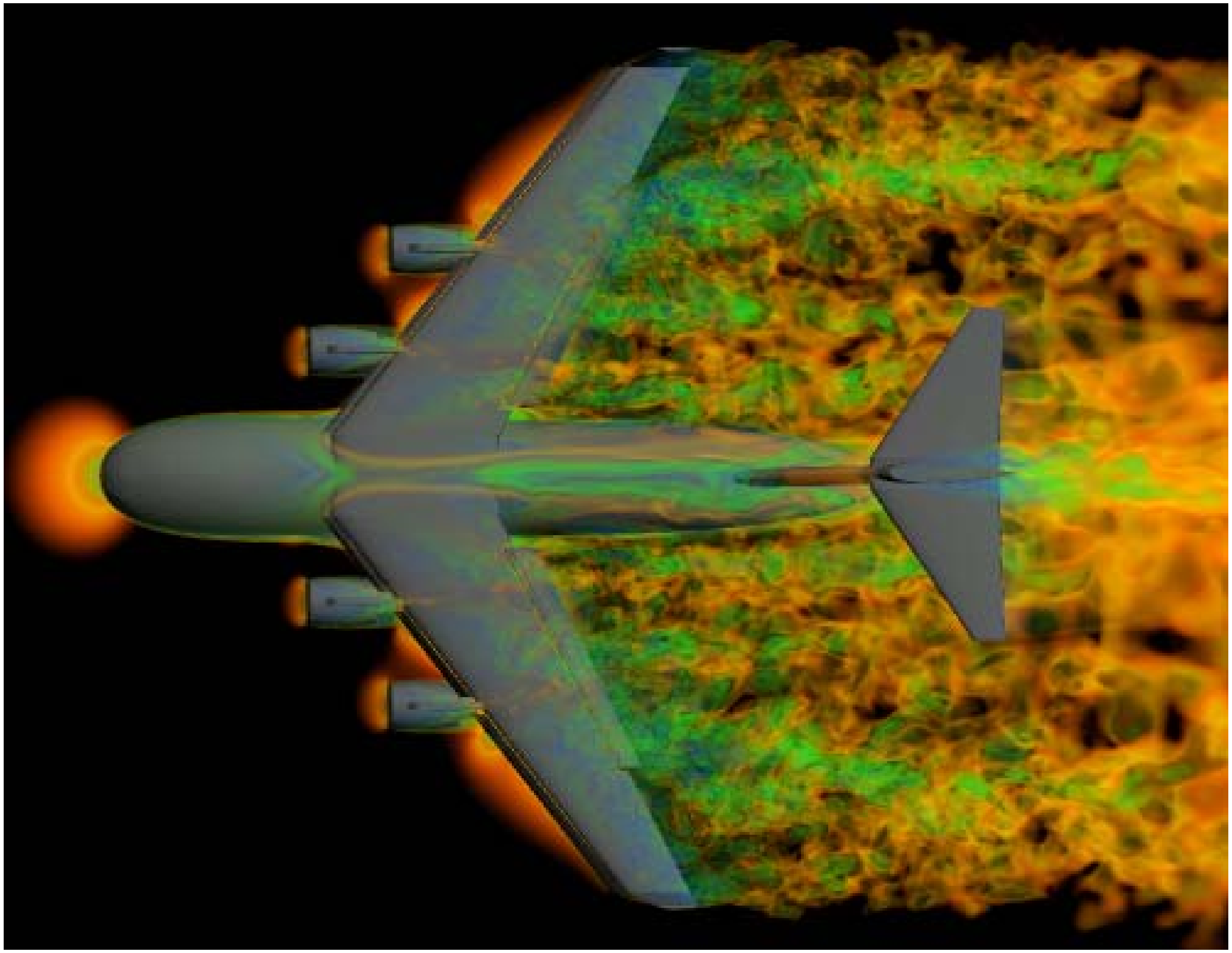


CFD Code Description (continued)

- Large Scale Applications
  - 1 to 35 Million tetrahedral element meshes are "typical"
    - 100 to 2000 time steps are common
  - 1 Billion tetrahedral element simulation computed using 1056 processors of a T3E
    - 850 Million equations
  - 243 Million tetrahedral elements solved using 28 processors of an X1
    - 41 million nodes
    - 160 Million equations
    - 2000 time steps computed
    - Largest problem we could fit
    - Performed as expected



CFD Code Description (continued)



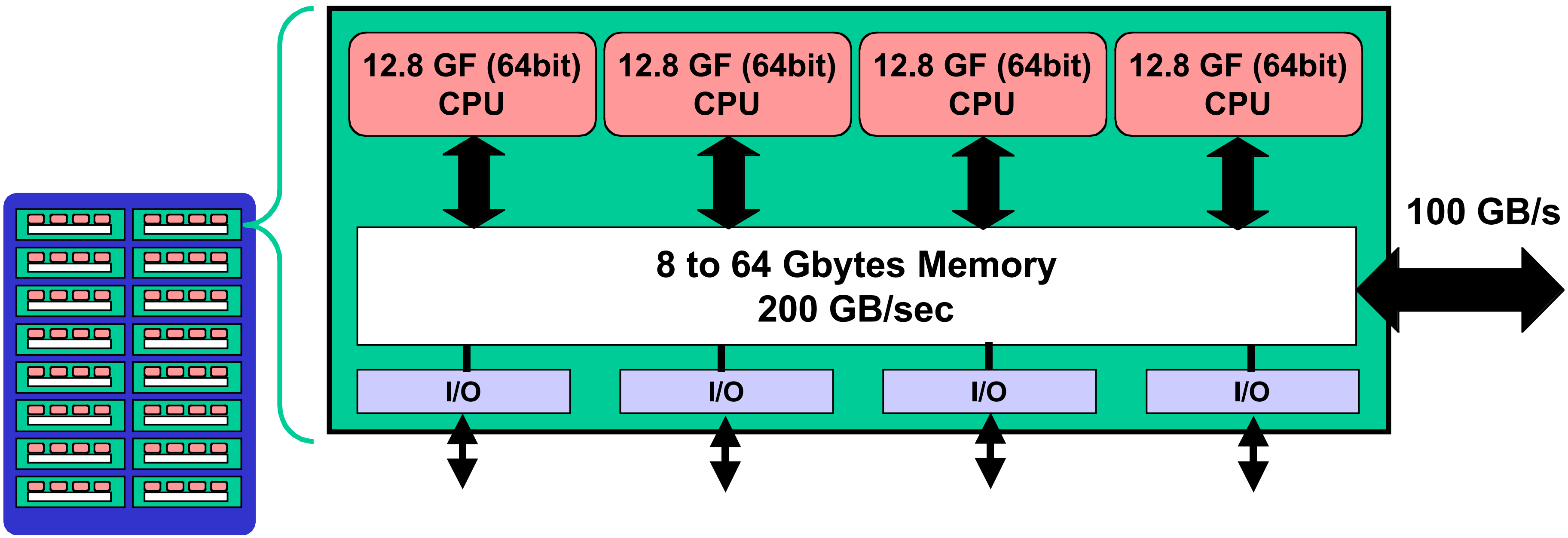
Airflow past a cargo aircraft in a take-off configuration. Mesh contains 243 million tetrahedral elements and 41 million nodes (160 million equations). Shown is a volume-rendered image of velocity magnitude.

Cray X1 Overview

- Brand new system
  - New architecture (hardware)
  - New operating system
  - New programming environment
    - Compilers
  - New programming models
    - MPI – Standard
    - Co-Array Fortran (CAF)
    - Unified Parallel C (UPC)
    - SHMEM
    - OpenMP / P-Threads (later)

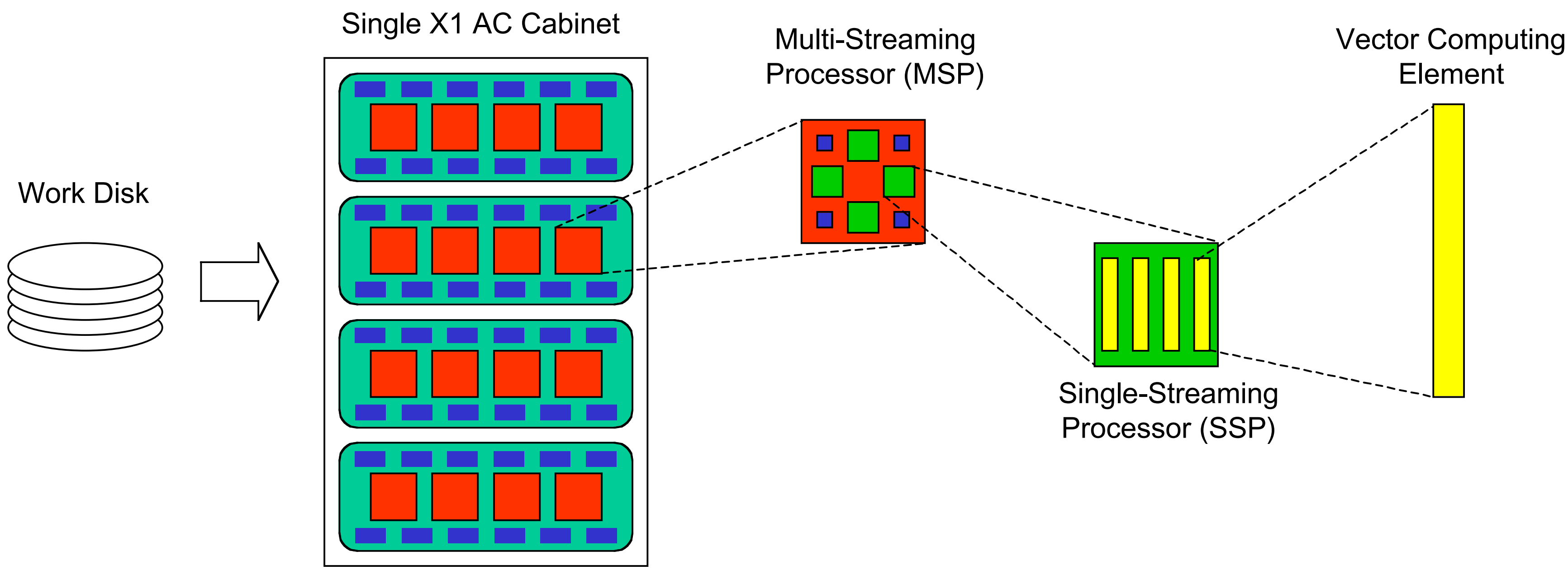
- Capability machine
- Excellent bandwidth and latency to memory
- Excellent vector performance
- Not the choice for scalar applications

Cray X1 Overview (continued)

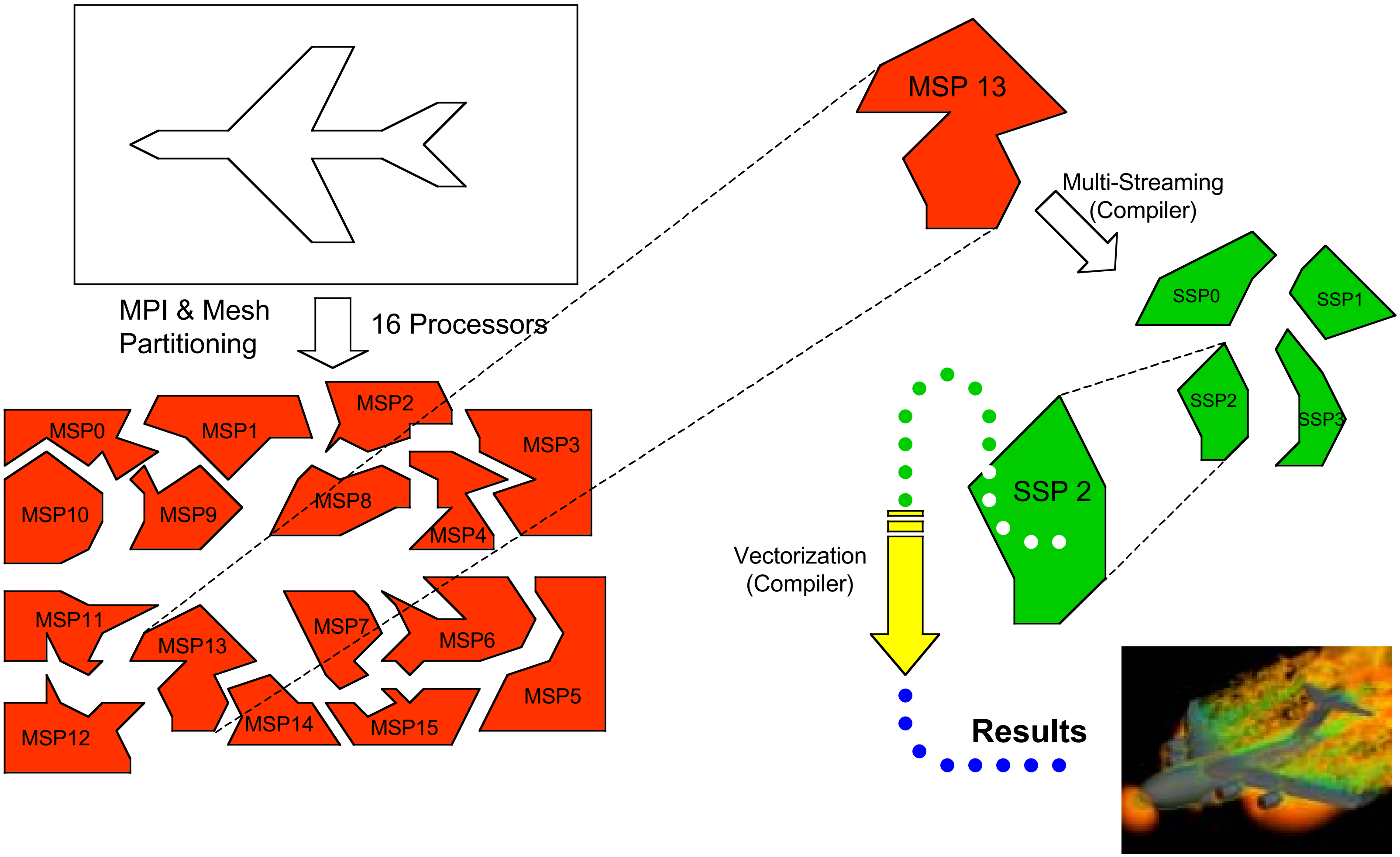


- 16 nodes in a liquid cooled chassis
- 4 nodes in an air cooled chassis

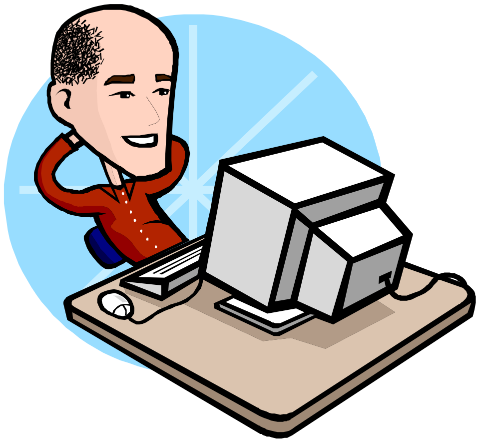
Cray X1 Overview (continued)



Cray X1 Overview (continued)



Porting and Code Modifications



- Original port took less than 1 hour
  - Didn't include full vectorization
- No issues related to binary data formats
  - Binary formats are more "standard"
- Compiler includes helpful features to identify vectorized and multi-streamed loops
  - Everything must vectorize and multi-stream
  - Multi-streaming is usually "mixed-in" with vectorization
    - Long enough loops (256+)
    - Shorter and/or nested loops may separate multi-streaming from vectorization
- Vectorization issues relating to unstructured (random) memory access

Porting and Code Mods. (continued)

Main computational kernel outline ("Block")

```
do i=1,number_of_elements
     node1 = ien(1,i)
     x1 = x(1,node1)
     y1 = x(2,node1)
     z1 = x(3,node1)
     …Several more lines like these…

     …Lots of calculations using these "localized" variables…



     d(1,node1) = d(1,node1) + result1
     d(2,node1) = d(2,node1) + result2
     d(3,node1) = d(3,node1) + result3
     …Several more lines like these…
enddo
```

Total number of mesh elements on this processor. Usually in the 100 thousand or more range.

"Gather" values from global memory.

In the range of 1100 flops per iteration.

"Scatter" values back into global memory. (Traditional Finite Element Assembly Operation)

Porting and Code Mods. (continued)

- Did not vectorize by default due to the “scatter” procedures at the end
  - If vectorized, results write to same memory location if the global memory index are the same (i.e. ‘node1’ value in the previous example)
  - For unstructured mesh calculations, this is likely to occur
  - Errors are observed, if this loop is forced to vectorize
- Achieved full vectorization by re-arranging the element order, and grouping elements (iterations) into smaller vectorizable pieces
  - Guarantees that there will be no memory access conflicts
    - No repeated mesh-node references in each element group

Porting and Code Mods. (continued)

New main computational kernel outline

```
do ig=1,number_of_groups
  i1 = gg_begin(ig)
  i2 = gg_end(ig)

!DIR$ CONCURRENT
  do i=i1,i2
    node1 = ien(1,i)
    x1 = x(1,node1)
    y1 = x(2,node1)
    z1 = x(3,node1)
    …Several more lines like these…

    …Lots of calculations using these "localized" variables…

    d(1,node1) = d(1,node1) + result1
    d(2,node1) = d(2,node1) + result2
    d(3,node1) = d(3,node1) + result3
    …Several more lines like these…
  enddo
enddo
```

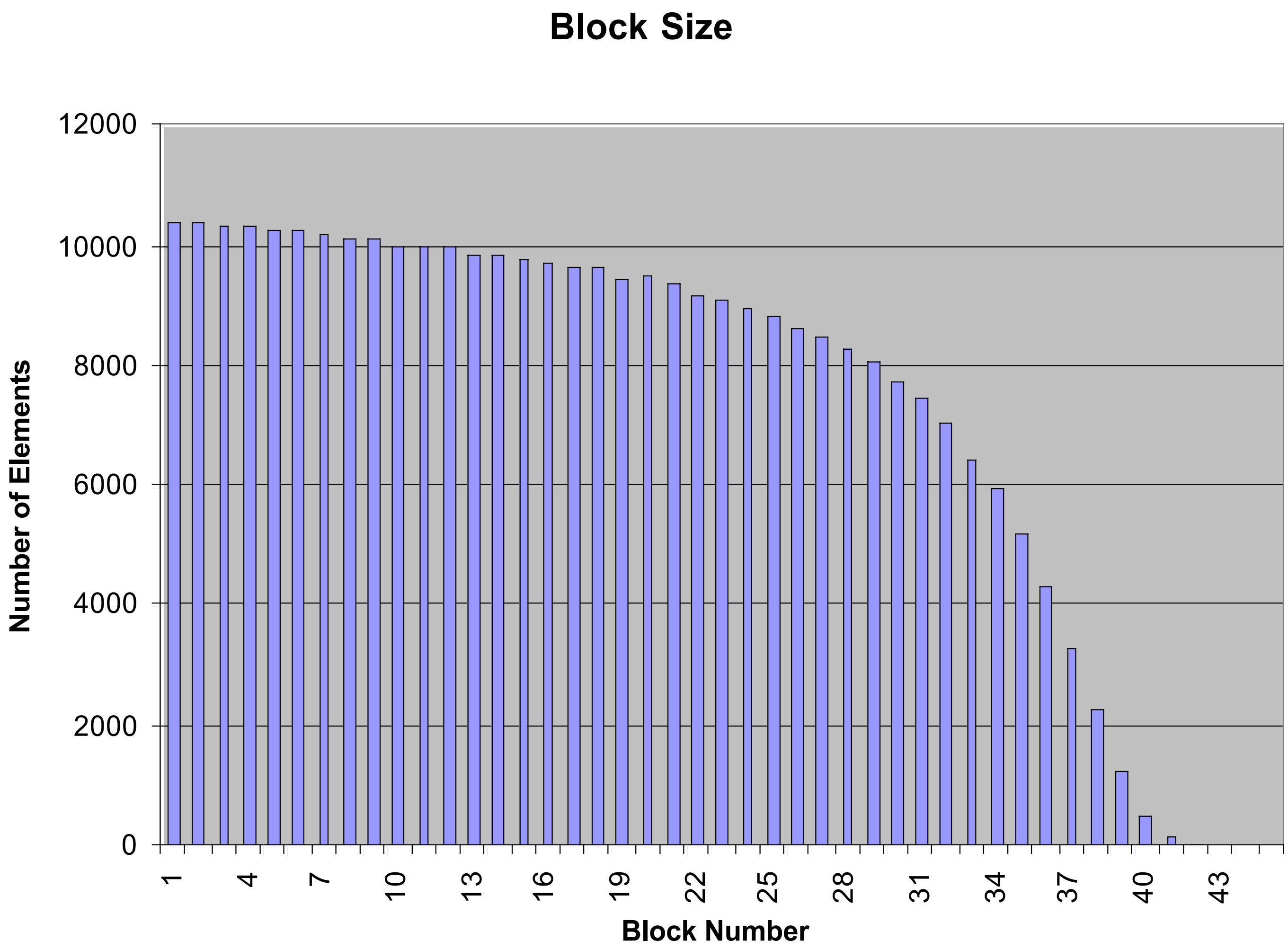
Outer group loop not vectorized.

Groups are pre-computed during the set-up stage.

Inner group loop can now be fully vectorized and multi-streamed.
Iteration count is usually in the 10k or larger range.
A few small group sizes exist at the end.

Can guarantee that these values will always be different for all iterations of this loop.

Porting and Code Mods. (continued)



Processor Performance

- Chose three representative CFD test cases of various sizes

- "Small" Data Set
  - 0.44 Million tetrahedral elements
  - 40 Time steps
- "Medium" Data Set
  - 2.0 Million tetrahedral elements
  - 10 Time steps
- "Large" Data Set
  - 4.3 Million tetrahedral elements
  - 5 Time steps

Processor Performance (continued)

- **Block**
  - Main computational kernel (70% of computational time)
  - Formation of two large vectors
    - A few main loops with lots of work inside (each 10k iterations or larger in size)
    - Each iteration includes gathers and scatters to and from memory
- **Block MegaFLOPS (MF)**
  - Based on our own counting of operations within the “Block” source code
  - No “hardware performance monitoring” information exists at this time
- **GMRES**
  - Lots of work with vectors (small amount of operations per loop)
    - Vector-vector multiplication, vector-scalar multiplication, dot products
    - Each vector is 400k or larger in size
  - May include a small amount of “serial” work
- **Total**
  - Includes everything except set-up time
- **% Communication**
  - Percent of the “Total” time spent performing the inter-processor data transfers
    - Also includes broadcasts and reductions
    - May include some non-vectorized data re-arrangements on the X1

Processor Performance (continued)

Systems Tested

- Cray T3E-1200
  - 600 MHz Alpha
- Cray X1 Production System
  - 800 MHz / PrgEnv 4.3
- IBM p690 SP
  - 1.3 GHz Power4
- SGI Origin 3000
  - MIPS R14000 500 MHz

- 4, 8, 12 Processors (CPU)

Processor Performance (continued)

Small Data Set

		T3E-1200	Production X1	
4CPU	Block	3,537.0	66.5	***53.2 x***
	Block MF	*304.6*	*16,195.0*	***31.6%***
	GMRES	360.5	16.4	***22.0 x***
	Total	4,203.6	103.0	***40.8 x***
	% Comm	1.0	7.1	
8CPU	Block	1,749.0	31.3	***55.9 x***
	Block MF	*616.0*	*34,423.0*	***33.6%***
	GMRES	188.8	9.9	***19.1 x***
	Total	2,099.4	55.1	***38.1 x***
	% Comm	1.2	14.0	
12CPU	Block	1,133.7	22.1	***51.3 x***
	Block MF	*950.0*	*48,728.4*	***31.7%***
	GMRES	128.1	9.3	***13.8 x***
	Total	1,376.6	44.3	***31.1 x***
	% Comm	1.7	18.9	

Large Data Set

		T3E-1200	Production X1	
4CPU	Block	4,327.3	82.5	***52.5 x***
	Block MF	*304.9*	*15,991.6*	***31.2%***
	GMRES	438.3	17.4	***25.2 x***
	Total	5,120.0	117.2	***43.7 x***
	% Comm	0.7	2.8	
8CPU	Block	2,175.5	41.5	***52.4 x***
	Block MF	*606.5*	*31,791.8*	***31.0%***
	GMRES	232.9	9.6	***24.3 x***
	Total	2,587.7	61.1	***42.4 x***
	% Comm	0.8	4.2	
12CPU	Block	1,466.4	27.5	***53.3 x***
	Block MF	*899.8*	*47,923.6*	***31.2%***
	GMRES	151.9	7.0	***21.7 x***
	Total	1,741.8	42.2	***41.3 x***
	% Comm	0.9	5.8	

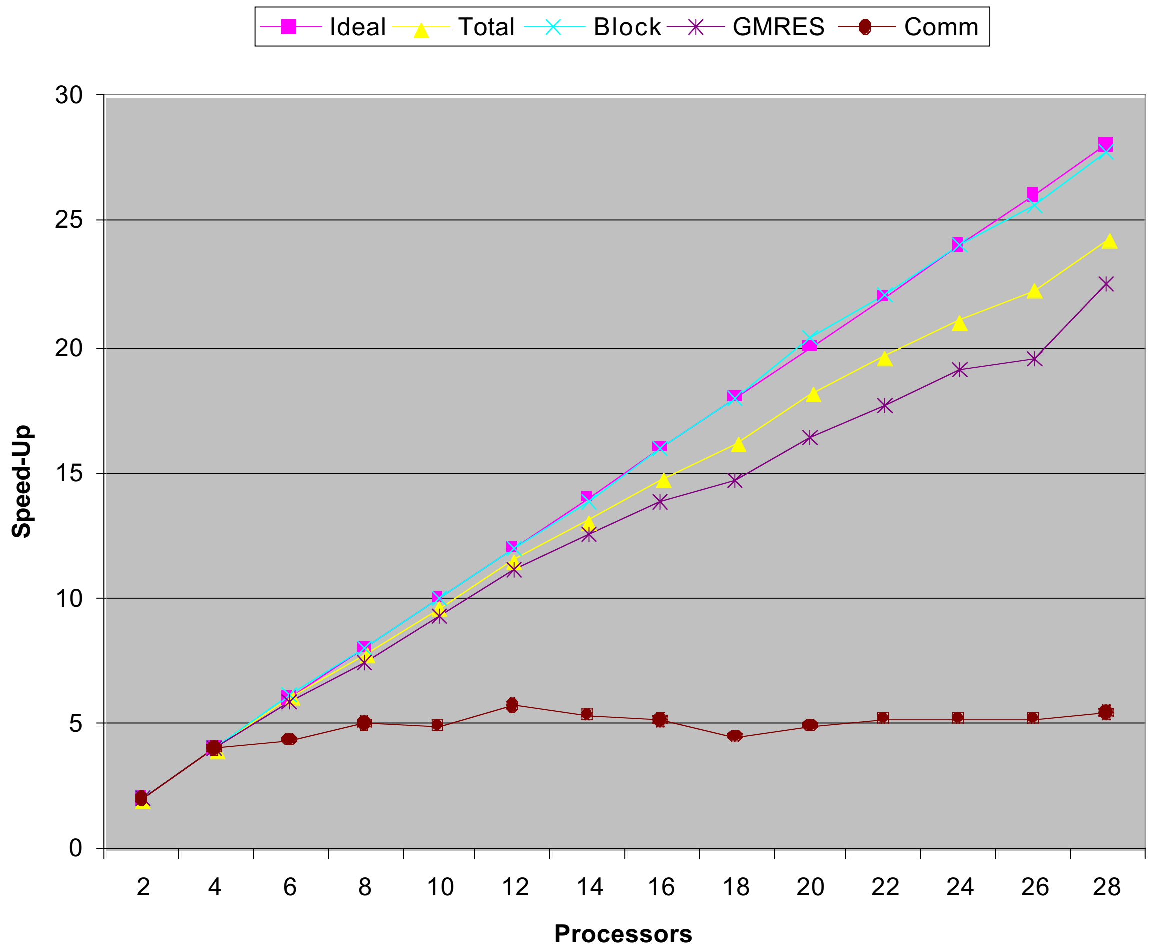
Processor Performance (continued)

Medium Data Set (2 Million Elements)

	T3E-1200	SGI Origin	IBM p690 SP	Production X1			
4pe Block	3,997.7	2,049.7	1,164.2	76.0	***52.6 x***	***27.0 x***	***15.3 x***
Block MF	304.1	593.1	1,044.3	15,995.0	***31.2%***		
GMRES	388.1	209.7	115.1	14.7	***26.4 x***	***14.3 x***	***7.8 x***
Total	4,715.4	2,415.6	1,375.8	107.2	***44.0 x***	***22.5 x***	***12.8 x***
% Comm	0.8	1.4	1.2	3.5			
8pe Block	1,994.0	786.0	503.2	38.2	***52.2 x***	***20.6 x***	***13.2 x***
Block MF	609.7	1,546.9	2,416.1	31,830.3	***31.1%***		
GMRES	198.3	126.9	61.6	8.1	***24.5 x***	***15.7 x***	***7.6 x***
Total	2,361.5	981.1	614.4	57.0	***41.4 x***	***17.2 x***	***10.8 x***
% Comm	0.9	1.9	2.1	6.8			
12pe Block	1,335.9	441.4	374.7	25.1	***53.2 x***	***17.6 x***	***14.9 x***
Block MF	910.1	2,754.4	3,244.5	48,411.6	***31.5%***		
GMRES	132.3	66.9	50.4	5.6	***23.6 x***	***11.9 x***	***9.0 x***
Total	1,589.9	550.5	466.8	39.0	***40.8 x***	***14.1 x***	***12.0 x***
% Comm	1.1	2.5	3.0	9.5			

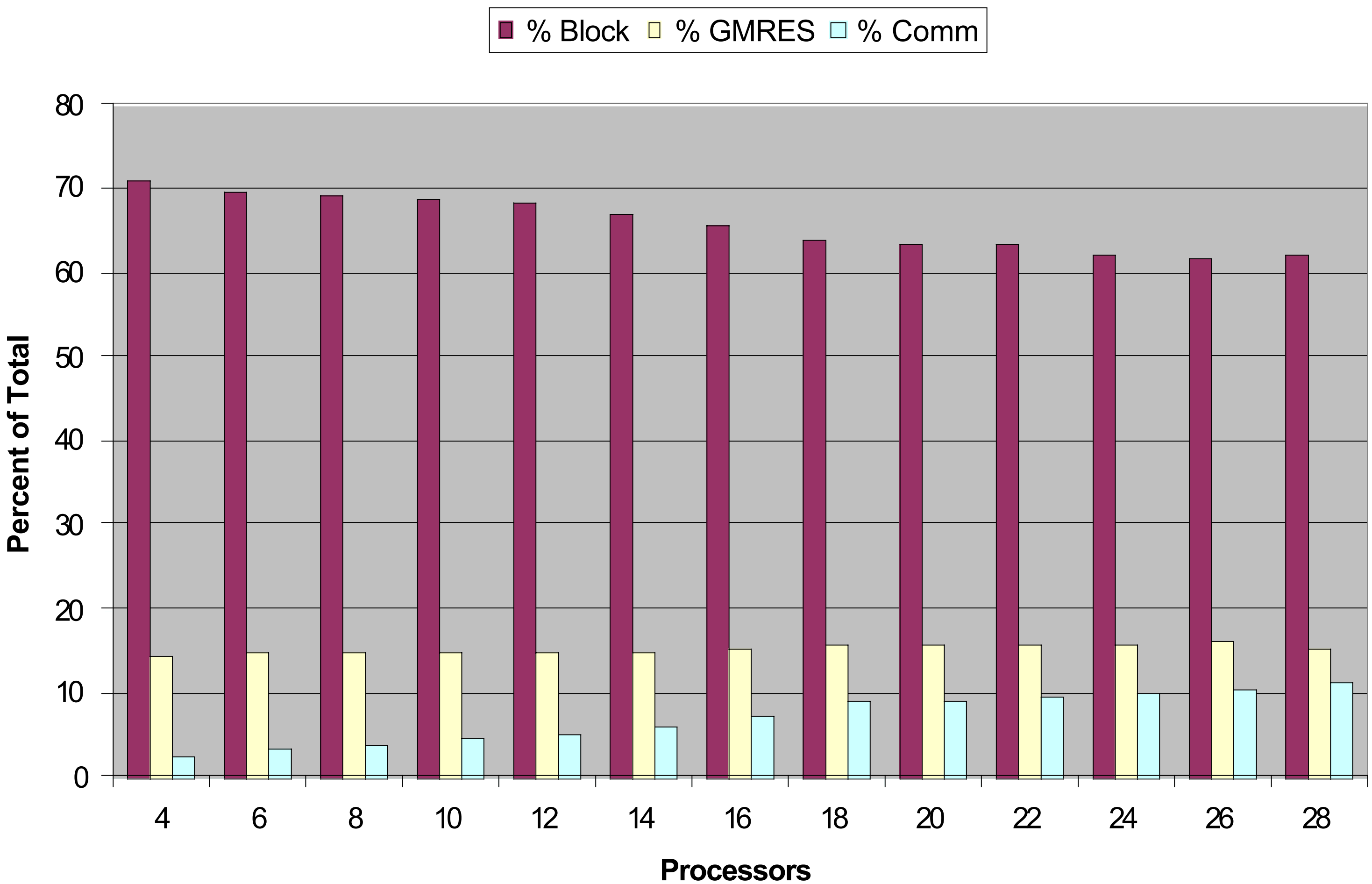
Scalability and Network Performance

- Scalability up to 28 processors (MSPs)



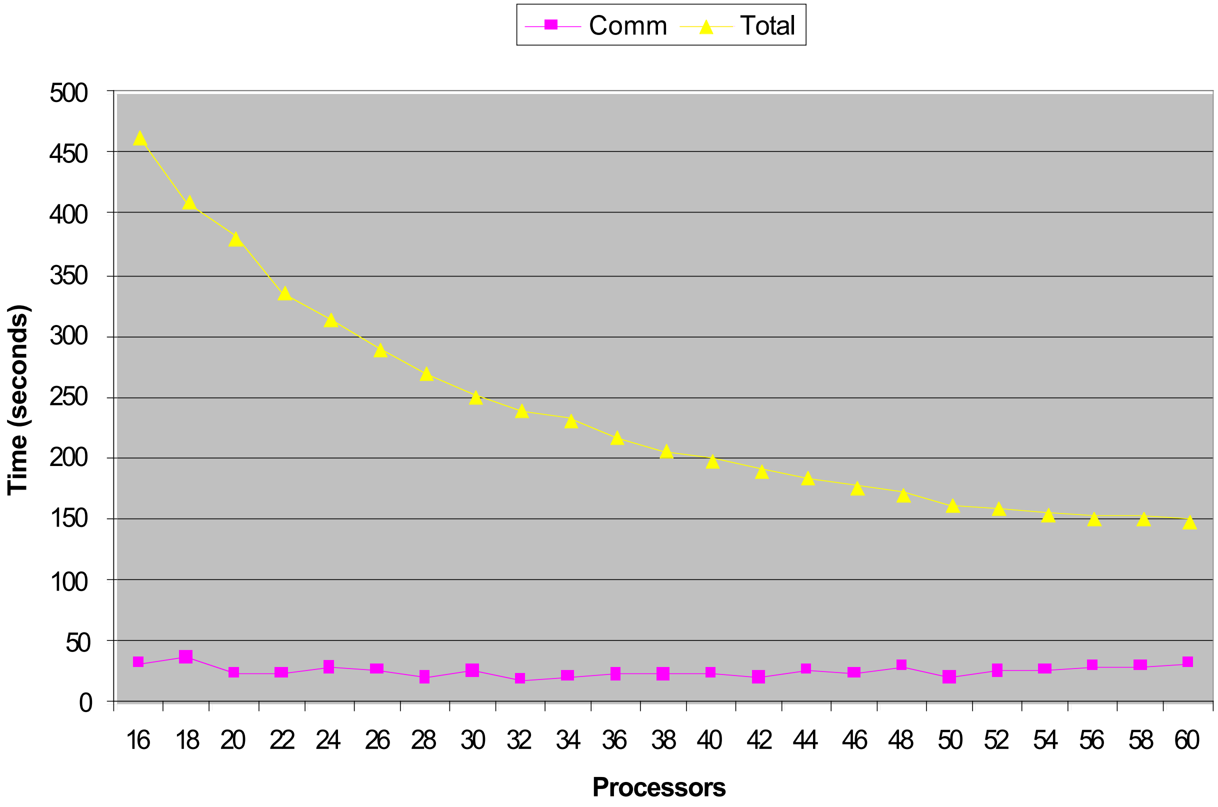
Scalability (continued)

- Where **BenchC** spends its time



Scalability (continued)

- Scalability up to 60 processors
  - Data set may be too small for this size of a job
    - Network performance begins to dominate



Scalability (continued)

- Communication times (percent of time spent; scalability) needs to improve
- Difficult to make an accurate measure of communication time
  - More detailed communication measurements show a significant amount of synchronization time is being measured
  - Due to element grouping strategy, each MSP performs at slightly different rates (up to 9%) so synchronization time is significant
- CAF has been implemented for a Fortran CFD code
  - 3 to 4 percent of measured communication time is actually spent communicating
  - Detailed measurements of actual communication time show roughly 2 to 3 Gbytes per second
- Plan to test UPC with the BenchC code
  - Should show better overall scalability

Final Thoughts

- Long vectors with lots of work are the best
  - Long vectors with a few operations each iteration do not see the same performance increases
- Non-vectorized work is very slow and will limit overall performance
  - Everything must vectorize (and multi-stream)
- Performance increases observed (so far) with new releases of UNICOS/MP and/or Programming Environment
  - Probably won't continue much longer
- Further vectorization and optimization could increase overall GFLOP rates even further
  - Better memory (and memory flow) management

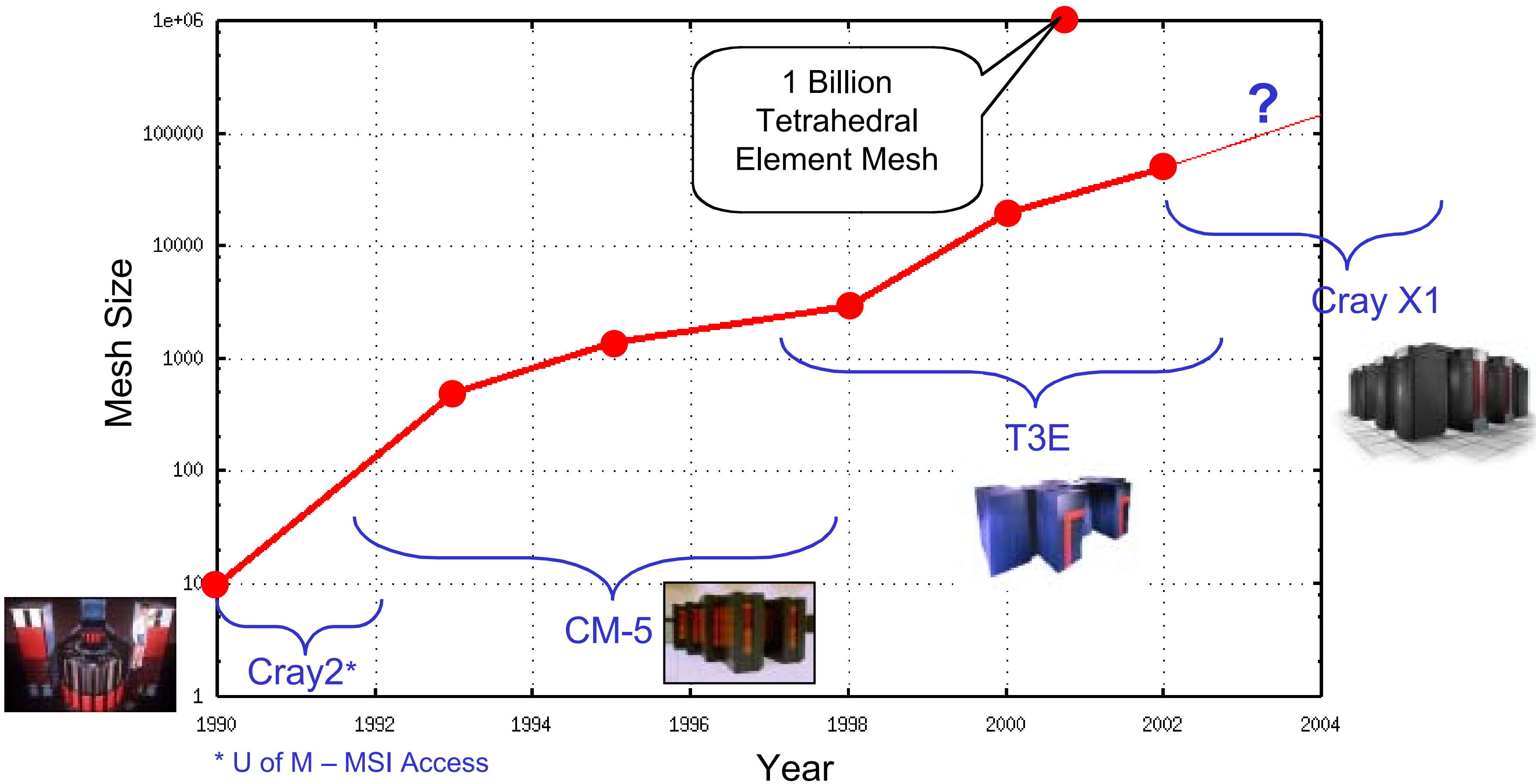
Final Thoughts (continued)

- Cray X1 has proven to be a very stable and productive system
  - A few initial OS (PrgEnv) problems have been resolved quickley
- Have performed several large and detailed long-running CFD simulations
- Performance of these CFD codes has been very impressive
  - Other similar CFD codes (Fortran-based) have also been ported and are running well on the system (same performance as BenchC)
- Good performance is seen for other applications
  - See AHPCRC's MM5 talk on Thursday

- The AHPCRC is ready to go ahead with the X1 as a full production system, available to all AHPCRC, Army, and DOD researchers

Final Thoughts (continued)

- Application sizes
  - Proportional to complexity and accuracy



* U of M – MSI Access