The Space Simulator: Modeling the Universe from Supernovae to Cosmology

Michael S. Warren
msw@lanl.gov

Chris L. Fryer
fryer@lanl.gov

M. Patrick Goda
pgoda@lanl.gov

Theoretical Astrophysics
Mail Stop B227
Los Alamos National Laboratory
Los Alamos, New Mexico 87545

ABSTRACT

The Space Simulator is a 294-processor Beowulf cluster with theoretical peak performance just below 1.5 Teraflop/s. It is based on the Shuttle XPC SS51G mini chassis. Each node consists of a 2.53 GHz Pentium 4 processor, 1 Gb of 333 MHz DDR SDRAM, an 80 Gbyte Maxtor hard drive, and a 3Com 3C996B-T gigabit ethernet card. The network is made up of a Foundry FastIron 1500 and 800 Gigabit Ethernet switch. Each individual node cost less than $1000, and the entire system cost under $500,000. The cluster achieved Linpack performance of 665.1 Gflop/s on 288 processors in October 2002, making it the 85th fastest computer in the world according to the 20th `TOP500` list. Performance has since improved to 757.1 Linpack Gflop/s, ranking at #88 on the 21st TOP500 list. This is the first machine in the TOP500 to surpass Linpack price/performance of 1 dollar per Mflop/s.

Keywords

Beowulf, cluster, price/performance, astrophysics, N-body

1. INTRODUCTION

The Space Simulator [13, 14] is our third generation Beowulf cluster. The first was Loki [16], which was constructed in 1996 from 16 200 MHz Pentium Pro processors for $50k. Loki was among the earliest generation of Beowulf clusters [3], and was the first to be recognized with the Gordon Bell price/performance award [22]. Loki was followed by the Avalon cluster [17], which used 144 alpha processors and cost about $300k. Avalon also won a Gordon Bell prize [15] and was ranked as the 113th fastest computer in the world in June 1998 [4]. The Space Simulator follows the same basic architecture as our previous machines, but is the first to use Gigabit Ethernet as the network fabric, and requires significantly less space than a cluster using typical ATX cases.

SC'03 November 15-21, 2003, Phoenix, Arizona, USA

2. BUILDING THE CLUSTER

The funds to purchase the Space Simulator became unexpectedly available in mid-July of 2002. Fiscal constraints required the system to be delivered by September 31. This left little time for benchmarking systems and testing components. Our goal was to purchase a computer which would obtain the highest performance possible on the astrophysics codes we wanted to run, within the budget we were alloted. It had to be delivered within one month. The machine also had to be reliable and maintainable enough that our very limited system administration resources would be capable of keeping the machine operational. We estimated the amount of cooling capacity available would limit the cluster to about 35 kW of power dissipation. The Space Simulator architecture (see Table 1) was defined by mid-August of 2002, based on the Shuttle XPC chassis.

We obtained quotes for clusters from several manufacturers. While one can argue the specifics of the value of extras typically included in such clusters (high-performance interconnects, dual-CPU motherboards, system management features), we estimated that a "commercial" solution would provide about 50% of the performance of what we could build ourselves, for the same amount of money. For many customers, this would be an acceptable markup. Certainly, it is a vast improvement over the factor of 10 difference that was typical between a self-built Beowulf cluster and the available commercial solutions 5 years ago.

Our limited preliminary benchmarking demonstrated our codes were faster on Intel processors than on similarly priced AMD processors and that higher performance RDRAM did not justify its extra cost over DDR SDRAM. The Green Destiny architecture [23] was ruled out on price-performance grounds, since our space, power and cooling were not sufficiently constrained. The XPC system was selected due to its small size, the elimination of the failure-prone CPU fan, and its ability to support the relatively new 533 MHz front side bus for the Intel Pentium 4 architecture. The main disadvantages of the system were that it provided only a single 32-bit 33 MHz PCI expansion slot, and 10% of its memory bandwidth was shared with the on-board video controller. It is interesting to note that the volume occupied by a Shuttle XPC chassis (30x20x18.5 cm or 0.011 cubic meters) is nearly the same as a 1U rackmount chassis (19x23.6x1.75 inches or

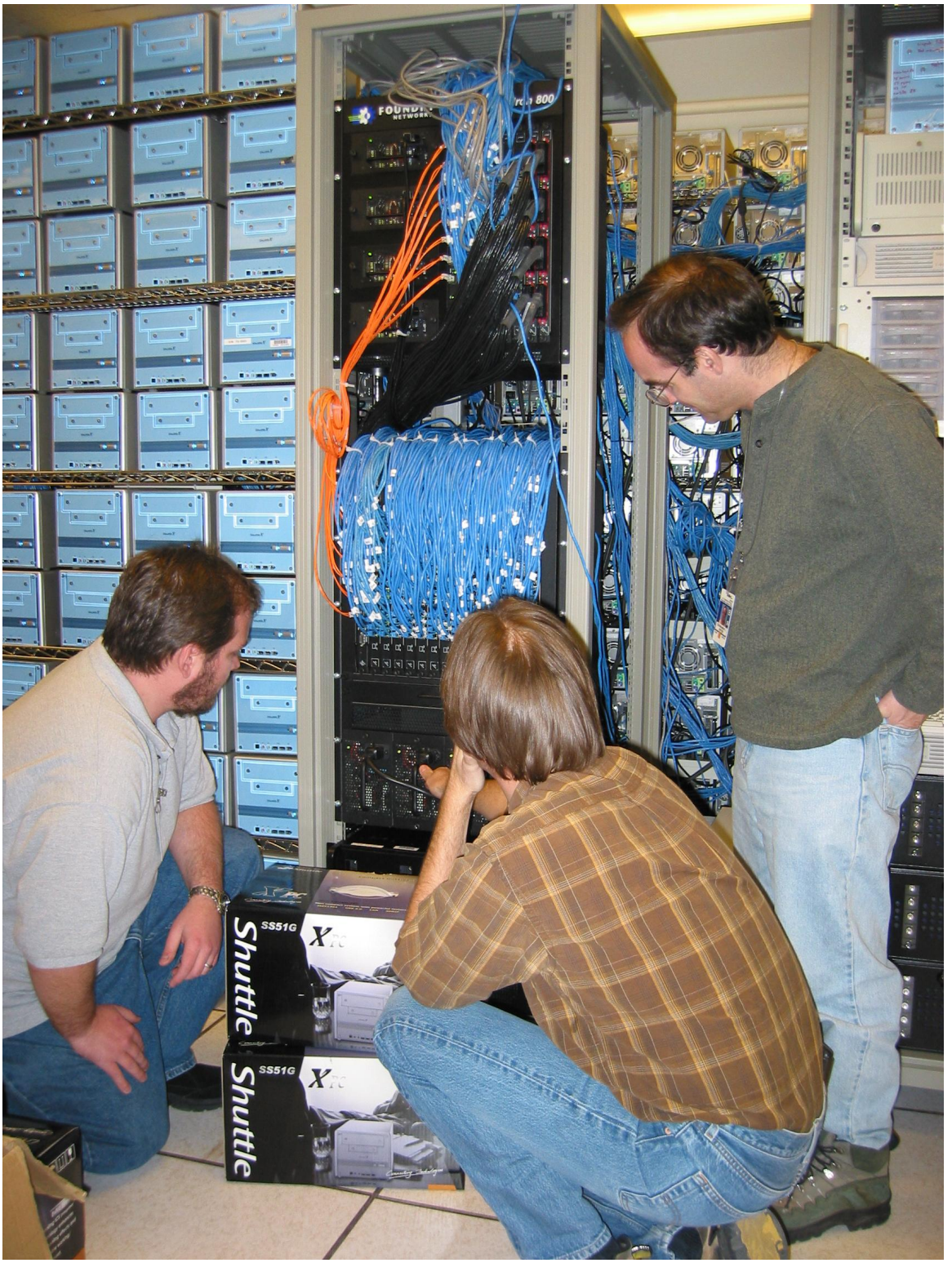

Figure 1: A front view of the Space Simulator which shows the rack containing the two Foundry Gigabit Ethernet switches. The fiber trunk between the two switches are the orange cables in the picture. The 224 ethernet cables attached to the lower switch obscures the upper half of the Foundry 1500 switch, while the Foundry 800 switch is mounted on the top portion of the rack.

Qty.	Price	Ext.	Description
294	280	82,320	Shuttle SS51G mini system (bare)
294	254	74,676	Intel P4/2.53GHz, 533MHz FSB, 512k cache
588	118	69,384	512Mb DDR333 SDRAM (1024Mb per node)
294	95	27,930	3com 3c996B-T Gigabit Ethernet PCI card
294	83	24,402	Maxtor 4K080H4 80Gb 5400rpm Hard Disk
294	35	10,290	Assembly Labor/Extended Warranty
		4,000	Cat6 Ethernet cables
		3,300	Wire shelving/switch rack
		1,378	Power strips
1		186,175	Foundry FastIron 1500+800, 304 Gigabit ports
Total		$483,855	$1646 per node 5.06 Gflop/s peak per node

Table 1: Space Simulator architecture and price (September, 2002). The average cost per node was $1646, with $728 (44%) of that figure representing the Network Interface Cards and Ethernet switches.

0.013 cubic meters). We have found the ability to easily remove a node from a shelf to be a major advantage over the complexities of a typical 1U rackmount on rails solution. Overall, the architecture which was chosen has proven to be an excellent solution, and the limited amount of benchmarking and testing prior to ordering the system has not resulted in problems.

2.1 Reliability

It is hard to quantify whether a do-it-yourself cluster solution requires more effort to make operational after delivery than a commercial solution. It is certainly helpful to have as much burn-in and testing from the vendor as possible, but the final verdict will always be whether your code runs or not. With large clusters, almost any problem you can think of will occur. We had some nodes which would not boot because PXE was not manually enabled on the ethernet card before shipment. Drive cables and PCI cards can come loose during shipment. The BIOS can be set inconsistently or incorrectly.

In our experience, fans are the component most likely to fail in a cluster. We expected increased reliability from the SS cluster, since the CPU fan is eliminated by the heat pipe used in the Shuttle chassis. This has been borne out by the first 9 months of failure statistics.

During the installation of the cluster and the initial large Linpack benchmark runs we identified the following defective hardware:

```
3 power supplies  6 disk drives    4 motherboards
6 sticks of DRAM  1 ethernet card
```

During the nine month period since the initial failures, the following hardware has failed:

```
2 power supplies 16 disk drives    1 motherboard
3 sticks of DRAM  1 fan loose
```

It is possible that the faulty DRAM and motherboard/CPU discovered after the initial installation was defective to begin with, but it took longer to identify since the errors were very infrequent. Additionally, there have been less than 10 "soft" node errors, which resolved themselves, or did not occur again after the node was rebooted. One perhaps interesting anecdote is that on every occasion where a node has failed with a Linux kernel panic, the cause was traced to bad hardware. The most common failure has been with disk drives. We have started monitoring each drive with the Linux SMART tools, and believe that a majority of the drive failures can be predicted.

The entire cluster has gone down on three occasions, once when the 120 kVa power distribution unit for the machine room failed and had to be replaced, which resulted in three days of down-time, and twice during power outages. On several other occasions, 15-amp breakers on individual power strips tripped, necessitating a rebalancing of the power distribution using a slightly more conservative maximum power consumption figure.

Additionally, there have been 4 soft failures of ports on the ethernet switch which were resolved by a power cycle of the switch. Since a recent upgrade to the switch firmware, no soft failures of the switch have occurred.

3. BENCHMARKS

In this section, we attempt to characterize the performance of the Space Simulator, proceeding from benchmarks of particular aspects of the architecture to more general measurements. In particular, we try to provide a clear baseline for the measurement of gigabit-ethernet connected clusters, and provide a basis for future comparison with clusters connected via less mainstream technology. These results are also intended to provide a general context in which to view the price/performance results quoted in the application section.

3.1 Network Performance

Gigabit network performance varies dramatically depending on the particular NIC used in the Shuttle systems. Some cards which had very good performance on a 64-bit or 66 MHz PCI bus performed poorly with the Shuttle 32-bit 33 MHz bus. We selected the 3Com 3c996B-T cards based on testing a variety of cards on a prototype system. The performance results are listed below using Linux 2.4.20. The 1.0 version of the `tg3` driver that shipped with Linux 2.4.20 turned out to be significantly slower than earlier versions, so the results below use the earlier 0.99 version.

In order to determine the behavior of the Foundry switch backplane, we wrote a small MPI program which simultaneously sends messages between pairs of processors along various hypercube edges. Within a 16-port switch module, the messages are non-blocking. The capacity from one mod-

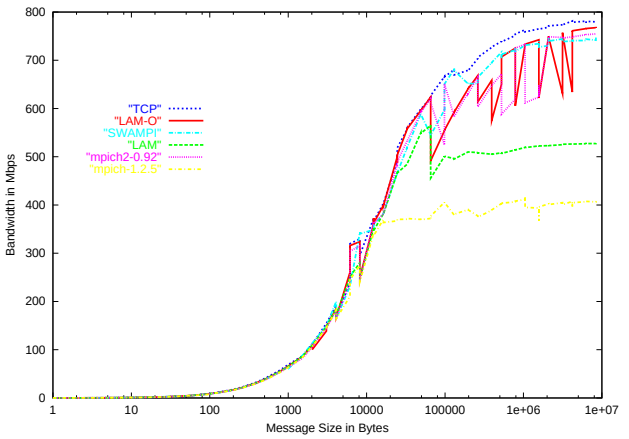

Figure 2: We show bandwidth vs message size for a variety of message passing libraries as measured via `NetPIPE`. Several features are evident, showing `mpich-1.2.5` has lower performance for large messages than the rest of the libraries. The 0.92 beta release of `mpich2` has apparently solved that problem. Also, using `LAM 6.5.9` with the `-O` flag (designating a homogeneous environment) significantly improves performance. The highest bandwidth is obtained via plain `TCP`, achieving 779 Mbits/sec. The latencies for small messages range from 79 microseconds for `TCP`, to 83 microseconds for `LAM`, and 87 microseconds for `mpich-1.2.5` and `mpich2-0.92`.

ule to another is only 8 gigabits. We verified that with 16 processors on one module sending to 16 modules on another module, the total throughput was about 6000 Mbits. Further, since our overall switch is a trunked combination of a Fastiron 1500 and a Fastiron 800, messages between the two switches are limited by the 8 Mbit trunk. This limits the scaling of codes running on more than about 256 processors.

3.2 Memory Bandwidth

The XPC node memory bandwidth is less than optimal for its 333 MHz frequency due to the fact that the on-board video system uses the system DRAM for it's frame buffer. It is possible to disable the on-board VGA controller and increase memory copy bandwidth by 10%, but you must then insert an AGP video card into the system in order for it to boot.

It has been our experience that the factor limiting node performance for a large fraction of scientific applications is the local node memory bandwidth. To quantify the effects of memory bandwidth, we performed several experiments which were made possible by the control enabled by the BIOS setup of the XPC nodes. It is possible to independently control the frequency of the processor and memory bus. One can then overclock the entire system, or underclock the CPU or memory, and measure the resulting performance. Using these figures, it is possible to roughly predict how much the system performance depends on memory bandwidth or processor performance independently.

We measured the performance of a variety of standard benchmarks. We measured memory bandwidth (STREAM), overall application performance (SPEC CPU2000 and NPB) and Linpack (HPL). Each was measured using a normal system (DDR333 memory, 2.53Gz P4 CPU) a slow memory system (DDR200 slowed memory by a factor 0.6) a slow CPU system 1.9GHz (slowed processor by a factor of 0.75) and overclocking the system to a 140MHz front-side bus (sped up CPU and memory by a factor of 1.0526). Results are shown in Table 3.2.

From these results, it is apparent that changing the memory bandwidth has a large effect on overall performance. Especially for the NAS benchmarks SP, MG and CG, scaling the memory frequency by 0.6 results in a performance reduction near 0.6. This indicates that these benchmarks are memory bandwidth limited, and increasing the CPU speed without more memory bandwdith would result in little improvement.

3.3 Linpack

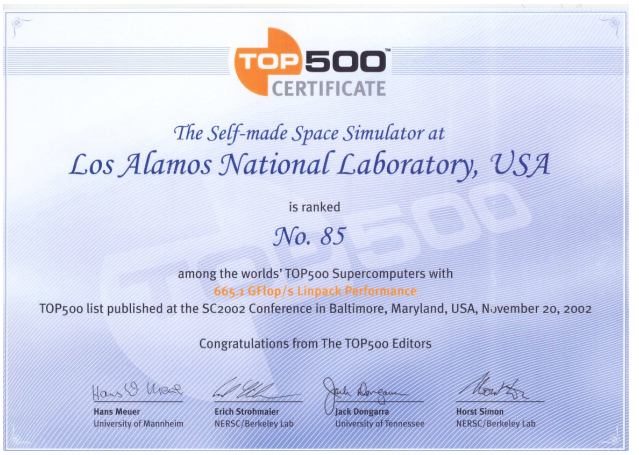

Figure 3: The Space Simulator ranked at #85 on the 20th `TOP500` list of the fastest computers in the world, as determined by the Linpack benchmark. Performance of 665.1 Gflop/s was obtained on 288 processors in October 2002. In April 2003, we obtained a higher figure of 757.1 Gflop/s through the use of a slightly faster version of ATLAS and using LAM instead of `mpich`. This ranks at #88 on the 21st `TOP500` list, and that performance would have ranked the Space Simulator at #69 on the 20th `TOP500` list. We believe our results are the first example of a machine in the `TOP500` with price/performance of better than 1 dollar per Mflop/s (we obtain 63.9 cents per Mflop/s, or $639 per Gflop/s).

In contrast to commercial machines which use a variety of proprietary libraries and compilers to obtain their peak performance, our Linpack benchmark result of 665.1 Gflop/s was obtained using freely available software and commodity off-the-shelf hardware. The OS (RedHat 7.3), kernel (Linux 2.4.20), message passing library (MPICH), compiler (gcc 3.1.1), BLAS library (ATLAS) and the High Performance Linpack (HPL) software are all freely available. This Linpack result ranks us as the 85th fastest computer in the world on the November 2002 `TOP500` list [5].

In April 2003, the Linpack benchmarks were run again on 288 processors. Instead of `MPICH 1.2.4`, we used `LAM 6.5.9` as the MPI implementation. In addition, the somewhat newer `ATLAS 3.5.0` distribution was used for the level

	Normal	Slow mem	Slow CPU	Overclock
copy	1203.5	761.8(0.63)	1143.4(0.95)	1268.5(1.054)
add	1237.2	749.8(0.61)	1165.3(0.94)	1302.8(1.053)
scale	1201.8	756.1(0.63)	1142.8(0.95)	1267.0(1.054)
triad	1238.2	748.9(0.61)	1160.7(0.94)	1304.1(1.053)
BT	321.2	204.1(0.635)	293.9(0.915)	342.3(1.066)
SP	216.5	131.7(0.608)	200.1(0.924)	229.6(1.061)
LU	404.3	262.2(0.649)	366.2(0.906)	427.4(1.057)
MG	385.1	231.4(0.601)	360.8(0.937)	400.1(1.039)
CG	313.1	189.4(0.605)	273.9(0.875)	330.2(1.055)
FT	351.0	248.7(0.708)	302.9(0.863)	385.1(1.097)
IS	27.2	21.2(0.779)	22.5(0.827)	28.9(1.063)
CINT2000	790	655(0.83)	640(0.81)	830(1.051)
CFP2000	742	527(0.71)	646(0.87)	782(1.054)
Linpack	3.302	2.865(0.868)	2.602(0.788)	3.476(1.053)

Table 2: The effects of changing processor speed and memory bandwidth are shown for a number of benchmarks. The "normal" system uses DDR333 memory. For "slow mem" the memory clock is reduced from 2x166 Mhz to 2x100 Mhz, resulting in the equivalent of DDR200 performance, which is a factor of 0.6 less. For "slow CPU" the processor is clocked down from 2.53 GHz to 1.9 Ghz (a factor of 0.75). For "overclock", the FSB is increased from 133 MHz to 140 Mhz. For STREAM copy, add, scale and triad, results are in Mbytes/sec. For NPB, results are Mop/sec. For Linpack, results are in Gflop/sec. Values in parenthesis are the ratio to the value of the normal system. The results demonstrate that the performance of most benchmarks is sensitive to memory bandwidth, and less so to CPU frequency.

Benchmark	SS	ASCI Q
BT	17032	22540
SP	7822	17775
LU	27942	40916
CG	3291	4129
FT	9860	7275
IS	232	286

Table 3: 64-processor performance (Mops) for Class C NPB 2.4 benchmarks. Data from the Space Simulator with the Intel Compiler, version 7.1 and the ASCI Q system are presented

Benchmark	SS	ASCI Q
BT	63044	80418
SP	29348	55327
LU	81472	135650
CG	4913	10149
FT	21995	30100

Table 4: 256-processor performance (Mops) for Class D NPB 2.4 benchmarks. Data from the Space Simulator with the Intel Compiler, version 7.1 and the ASCI Q system are presented

3 BLAS, and the Intel 7.1 compiler suite was used to compile HPL (while gcc was used for ATLAS). The significant improvement seen, from 665.1 Gflop/s to 757.1 Gflop/s, we believe was mostly due to improved network performance via the switch to LAM. It is notable that the Space Simulator obtains performance almost as high as a 256 processor Itanium2 cluster connected with Myrinet.

3.4 NAS parallel benchmarks

The results shown in Figure 4 and 5 use the NAS Parallel benchmarks version 2 [1]. These benchmarks, based on Fortran 77 and the MPI standard, are intended to approximate the performance a typical user can expect for a portable parallel program on a distributed memory computer. All results use the Intel version 7.1 compilers with `FFLAGS = -O3 -tpp7 -xW -ipo -fno-alias` and `LAM 6.5.9` as the message passing library. Tables 3 and 4 compare the Space Simulator performance with that of the ASCI Q system at Los Alamos for identical problem sizes and numbers of processors.

3.5 SPEC CPU2000

We have measured the SPEC CPU performance [12] of our nodes with the Intel version 7.1 compilers. SPECfp2000 is 742. SPECint2000 is 790. These results have not been submitted to SPEC, and so are unofficial, but did follow all of the run rules for the SPEC benchmarks. In terms of node price/performance, neglecting the cost of the network and racks, each node cost $888, giving $1.20 per unit of SPECfp. Currently the fastest SPECfp result reported is 2119 (an HP Integrity Server rx2600 using a 1500 MHz Itanium 2 processor). In order to beat the SPECfp price/performance of a Shuttle XPC node, the HP system would have to cost less than $2500.

The equivalent Shuttle XPC systems have since declined in price. As of July 2003, the per node cost has decreased over $200, so SPECfp price/performance with a new system would be better than $1.00 per unit of SPECfp.

3.6 Gravity Kernel Performance

Execution time for our parallel N-body application is dominated by the force calculation in the inner loop. We have collected performance figures on a variety of processors in Table 5. The SS results correspond to the 2530-MHz Intel P4.

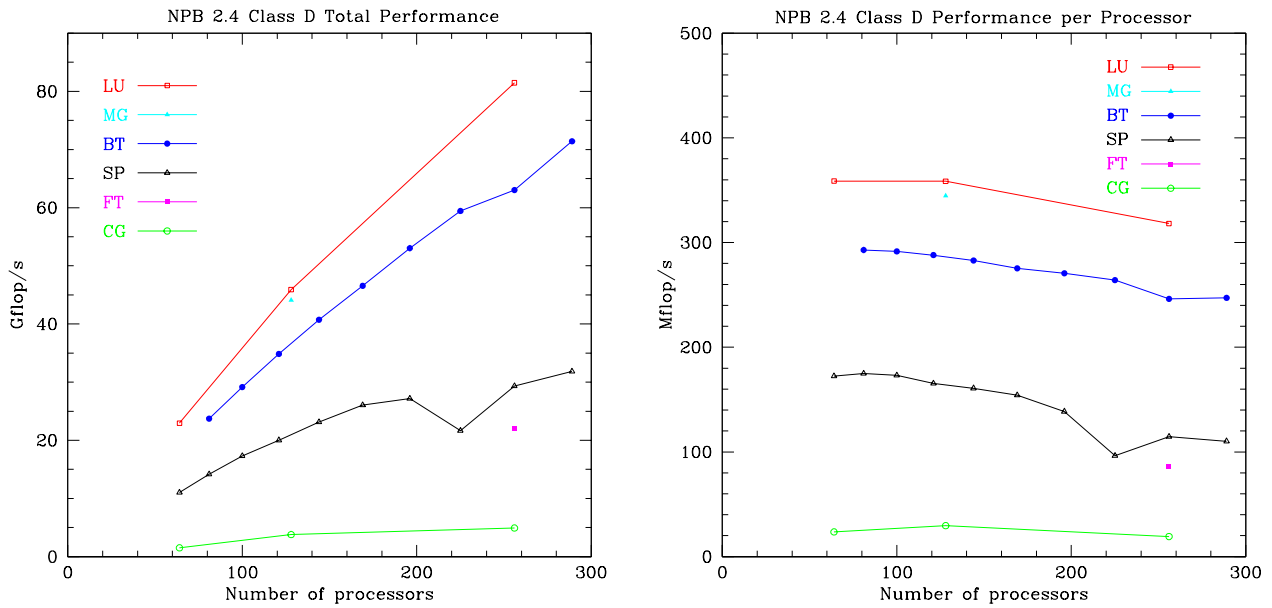

Figure 4: Scaling of the NAS Class D benchmarks on the Space Simulator. Perfect scaling would be a straight horizontal line for the plot on the right.

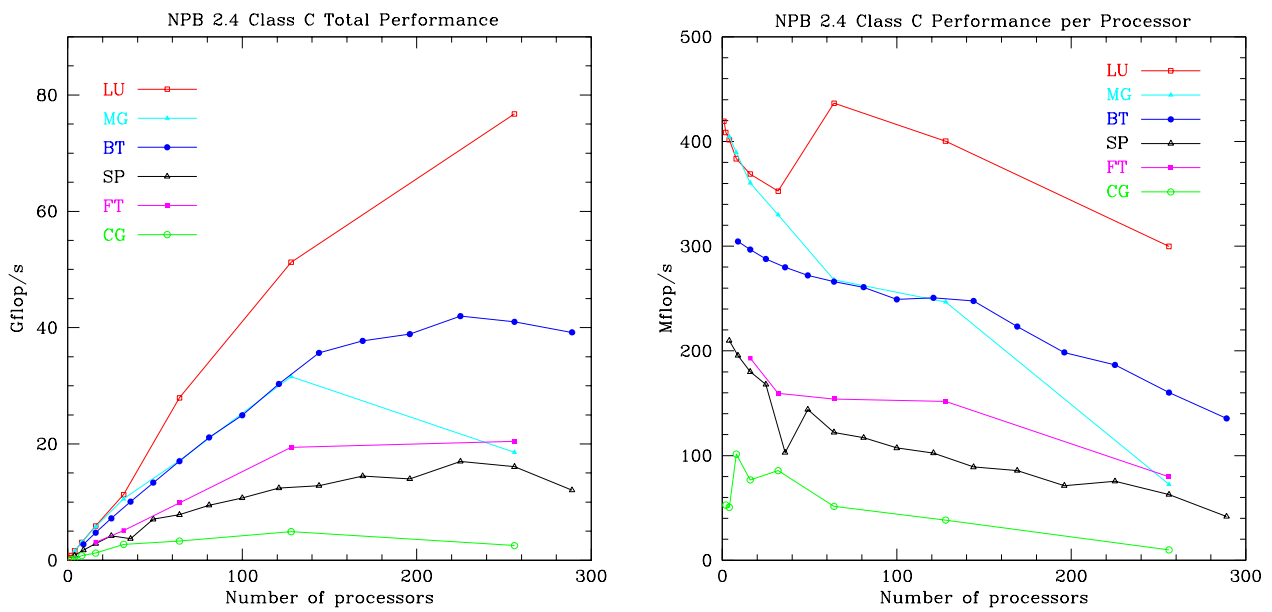

Figure 5: Scaling of the NAS Class C benchmarks on the Space Simulator. The computational problems are smaller than the Class D results shown in the previous plot, so the scaling is not as good for large numbers of processors. The feature in the plot for the LU benchmark (where the performance per processor becomes more higher on 64 processors than on a single processor) is likely due to the problem being divided into enough pieces that it fits into L2 cache on the processor.

Processor	libm	Karp
533-MHz Alpha EV56	76.2	242.2
667-MHz Transmeta TM5600	128.7	297.5
933-MHz Transmeta TM5800	189.5	373.2
375-MHz IBM Power3	298.5	514.4
1133-MHz Intel P3	292.2	594.9
1200-MHz AMD Athlon MP	350.7	614.0
2200-MHz Intel P4	668.0	655.5
2530-MHz Intel P4	779.3	792.6
1800-MHz AMD Athlon XP	609.9	951.9
1250-MHz Alpha 21264C	935.2	1141.0
2530-MHz Intel P4 (icc)	1170.0	1357.0

Table 5: Mflop/s obtained on our gravitational micro-kernel benchmark. The first column uses the math library *sqrt*, the second column uses an optimization by Karp, which decomposes the reciprocal square root into a table lookup, Chebychev interpolation and Newton-Raphson iteration, which uses only adds and multiplies. Note the significant improvement obtained through the use of the Intel version 6.0 compiler, which enables the P4 SSE and SSE2 capabilities.

4. APPLICATIONS

4.1 N-body methods

N-body methods are widely used in a variety of computational physics algorithms where long-range interactions are important. Several methods have been introduced which allow N-body simulations to be performed on arbitrary collections of bodies in time much less than $O(N^2)$, without imposition of a lattice [2, 8]. They have in common the use of a truncated expansion to approximate the contribution of many bodies with a single interaction. The resulting complexity is usually determined to be $O(N)$ or $O(N \log N)$, which allows computations using orders of magnitude more particles. These methods represent a system of N bodies in a hierarchical manner by the use of a spatial tree data structure. Aggregations of bodies at various levels of detail form the internal nodes of the tree (cells). These methods obtain greatly increased efficiency by approximating the forces on particles. Properly used, these methods do not contribute significantly to the total solution error. This is because the force errors are exceeded by or are comparable to the time integration error and discretization error.

Using a generic design, we have implemented a variety of modules to solve problems in galactic dynamics [18] and cosmology [24] as well as fluid-dynamical problems using smoothed particle hydrodynamics [21], a vortex particle method [9] and boundary integral methods.

4.2 The Hashed Oct-Tree Library

Our parallel N-body code has been evolving for over a decade on many platforms. We began with an Intel ipsc/860, Ncube machines, and the Caltech/JPL Mark III [11, 18]. This original version of the code was abandoned after it won a Gordon Bell Performance Prize in 1992 [19], due to various flaws inherent in the code, which had been ported from a serial version. A new version of the code was initially described in [20].

The basic algorithm may be divided into several stages. Our discussion here is necessarily brief. First, particles are domain decomposed into spatial groups. Second, a distributed tree is constructed. In the main stage of the algorithm, this tree is traversed independently in each processor, with requests for non-local data being generated as needed. In our implementation, we assign a `Key` to each particle, which is based on Morton ordering. This maps the points in 3-dimensional space to a 1-dimensional list, while maintaining as much spatial locality as possible. The domain decomposition is obtained by splitting this list into N_p (number of processors) pieces (see Figure 6). The implementation of the domain decomposition is practically identical to a parallel sorting algorithm, with the modification that the amount of data that ends up in each processor is weighted by the work associated with each item.

The Morton ordered key labeling scheme implicitly defines the topology of the tree, and makes it possible to easily compute the key of a parent, daughter, or boundary cell for a given key. A hash table is used in order to translate the key into a pointer to the location where the cell data are stored. This level of indirection through a hash table can also be used to catch accesses to non-local data, and allows us to request and receive data from other processors using the global key name space. An efficient mechanism for latency hiding in the tree traversal phase of the algorithm is critical. To avoid stalls during non-local data access, we effectively do explicit "context switching" using a software queue to keep track of which computations have been put aside waiting for messages to arrive. In order to manage the complexities of the required asynchronous message traffic, we have developed a paradigm called "asynchronous batched messages (ABM)" built from primitive send/recv functions whose interface is modeled after that of active messages.

In Table 6 we show the performance of the Space Simulator on a standard simulation problem which we have run on most of the major supercomputer architectures of the past decade. The problem is a spherical distribution of particles which represents the initial evolution of a cosmological N-body simulation. Overall, the performance of the full Space Simulator cluster is similar to that of 256 processors on ASCI Q, or a 1024 processor SP-3.

4.3 Cosmology Simulations

Obtaining a quantitative understanding of galaxy formation and clustering is the most important open theoretical problem in the study of the large-scale structure of the Universe. Observations strongly support the theoretical paradigm that structure evolves through gravitational collapse of primarily dark matter. The revolutionary transformation of cosmology from a qualitative to a quantitative science has occurred over just the last fifteen years. Driven by a powerful and diverse suite of observations, the parameters describing the large-scale Universe are now known to extraordinary precision.

Structure in the Universe forms almost entirely due to the gravitational collapse of primordial density fluctuations. On the very largest scales, such as those characteristic of the microwave background, linear theory is applicable; with the addition of a small amount of thermodynamics and linearized gas dynamics, a quantitative understanding has been achieved. At smaller scales, the essential nonlinearity of gravitational collapse makes such an understanding much

Year	Site	Machine	Procs	Gflop/s	Mflops/proc
2003	LANL	ASCI QB	3600	2793	775.8
2003	LANL	Space Simulator	288	179.7	623.9
2002	NERSC	IBM SP-3(375/W)	256	57.70	225.0
2002	LANL	Green Destiny	212	38.9	183.5
2000	LANL	SGI Origin 2000	64	13.10	205.0
1998	LANL	Avalon	128	16.16	126.0
1996	LANL	Loki	16	1.28	80.0
1996	SC '96	Loki+Hyglac	32	2.19	68.4
1996	Sandia	ASCI Red	6800	464.9	68.4
1995	JPL	Cray T3D	256	7.94	31.0
1995	LANL	TMC CM-5	512	14.06	27.5
1993	Caltech	Intel Delta	512	10.02	19.6

Table 6: Historical Performance of the Treecode.

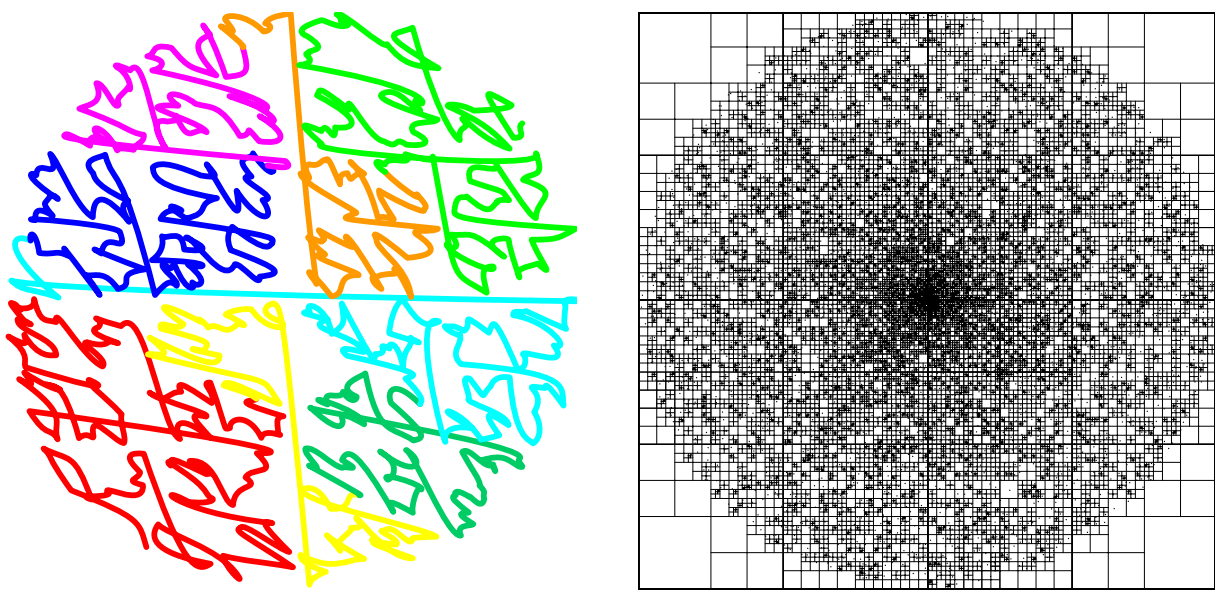

Figure 6: On left is the self-similar curve used for load-balancing, while the right figure represents a tree data structure in 2d for a group of centrally condensed particles.

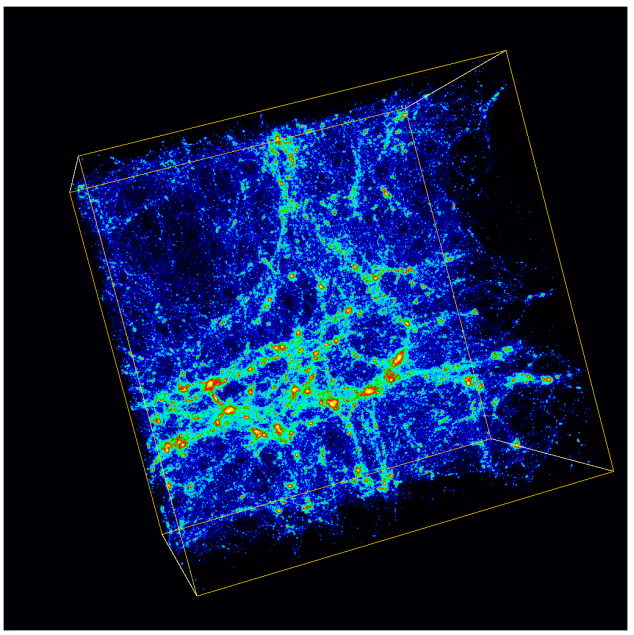

Figure 7: The figure represents a portion of the Universe about 125 Megaparsecs on a side at a redshift of 0.3, or an age of 3.5 billion years prior to the present epoch. The overall simulation of about 700 timesteps used 134 million particles, and was completed in a single run of just over 24 hours on 250 processors of the Space Simulator. 1.5 Terabytes of data from this simulation was saved.

harder to attain. In this regime, the distribution of matter can be studied only via large scale N-body simulations.

Our recent N-body simulations have achieved unprecedented spatial and mass resolution. Simulations at this resolution allow us to examine the sub-structure of dark matter halos and approach the problem of galaxy formation in a very direct way, possibly resolving many of the problems posed by bias, both in galaxy position and velocity fields. We are currently performing several 134 million particle cosmological N-body simulations per week on the Space Simulator, and have mostly completed a run with over 1 billion particles. Even larger simulations are possible using the out-of-core version of our code [10].

We quote performance results from a typical run, which took place over a continuous 24 hour period on 250 processors. The code saved 1.5 Tbytes of data, and performed 10^{16} floating point operations, for an average I/O rate of 417 Mbytes/sec and 112 Gflop/s. I/O was done in parallel to and from the local disk on each processor, so the peak I/O rate was near 7 Gbytes/sec.

4.4 Core-Collapse Supernovae

Core-collapse supernovae play a vital role in nearly every aspect of astronomy both as major sources of the emission of gamma-rays, neutrinos and gravitational waves to the chemical enrichment of the universe and the formation of compact objects (black holes, neutron stars, quark stars). These explosions are driven by neutrinos emitted from the collapsed core of a massive star. Studying core-collapse supernovae requires the coupling of gravitational and pressure forces of the core as it collapses down to nuclear densities with the radiation effects from neutrinos — a true radiation/hydrodynamics problem. The combination of radiation transport and the complex description of pressure forces for matter at nuclear densities pose difficulties both in optimization and message passing. In addition, these simulations must be run for 0.1-0.2 million timesteps. Because of these difficulties, nearly all previous simulations of these events have been limited to 2-dimensions.

Fryer, Warren and collaborators have performed the first ever full-physics three-dimensional simulations of supernova core-collapse [6] as part of the DOE SCIDAC Supernova Science Center `http://www.supersci.org`. By implementing the smooth particle hydrodynamics formalism onto the tree structure described above for N-body studies, we have been able to include both the essential physics and a flux-limited diffusion algorithm to model the neutrino transport. The largest simulation using 5 million particles was finished recently, and required roughly one month of time to model 100,000 timesteps on a 256 processors of the ASCI Q system. Taking advantage of the Lagrangian nature of smooth particle hydrodynamics, we have begun to study global asphericities in core-collapse: rotation [7] (Fig. 8) and asymmetric collapse. These 3-dimensional simulations allow us to address a number of outstanding questions in core-collapse astrophysics such as the origin of neutron star kicks and the gravitational wave signal for stellar collapse.

We are currently performing several follow-up simulations on the Space Simulator. For our 1 million particle simulations on 128 processors, per processor performance (using gcc/g77) is about 1/2 that of the ASCI Q system on an equivalent number of processors. We are running several simulations using 32 processors out to 150,000 timesteps. These simulations take roughly 4 months. Performance tuning remains to be done, especially investigating the use of the Intel 7.0 compilers.

5. CONCLUSIONS

Beowulf clusters have proven to be an effective computing resource in many appplication areas. We have demonstrated that commodity PC technology coupled with the latest generation of high-volume ethernet technology is capable of supercomputer-class performance. It is interesting to note there have been exactly six years between the completion of the Loki and Space Simulator clusters, which results in four "Moore's Law" doublings. Comparing the Loki architecture and price in Table 7 to the Space Simulator, we can see that Moore's Law scaling has actually been greatly exceeded in some aspects of the architecture. For instance, in 1996, Loki's disks cost $111 per Gigabyte. For the SS, they are close to $1 a Gigabyte, which is a factor of seven beyond the factor of 16 dictated by Moore's Law over six years. For memory, in the Loki days it was $7.35 per Megabyte, and it's now 23 cents per Megabyte, 2x lower than Moore's Law would have predicted.

These factors of improvement over Moore's Law are realized in the NPB price/performance results. Loki 16-processor performance on the NPB class B benchmarks was 355, 255, 428 and 296 Mflops for BT, SP, LU and MG respectively. For the SS, the corresponding 16-processor class B figures are 4480, 2560, 6640 and 4592, resulting in improvement ratios of 12.6, 10.0, 15.5 and 15.5. Since each SS processor cost only half as much as the Loki nodes, we see that the NPB

Qty.	Price	Ext.	Description
16	595	9520	Intel Pentium Pro 200 Mhz CPU/256k cache
16	15	240	Heat Sink and Fan
16	295	4720	Intel VS440FX (Venus) motherboard
64	235	15040	8x36 60ns parity FPM SIMMS (128 Mb per node)
16	359	5744	Quantum Fireball 3240 Mbyte IDE Hard Drive
16	85	1360	D-Link DFE-500TX 100 Mb Fast Ethernet PCI Card
16	129	2064	SMC EtherPower 10/100 Fast Ethernet PCI Card
16	59	944	S3 Trio-64 1Mb PCI Video Card
16	119	1904	ATX Case
2	4794	9588	3Com SuperStack II Switch 3000, 8-port Fast Ethernet
		255	Ethernet cables
Total		$51,379	$3211 per node 200 Mflop/s peak per node

Table 7: Loki architecture and price (September, 1996).

price/performance exceeds Moore's Law scaling by 25% for BT, and close to a factor of two for LU and MG.

For the N-body code, the overall price/performance improvement that clusters have obtained over the past six years has not differed much from Moore's Law extrapolations. Loki obtained performance of 1.28 Gflop/s for for the N-body code, while the whole SS obtains 180 Gflops, an improvement of a factor of 140. The price ratio between the machines is 9.4, which when multiplied by 16 for four 18-month Moore's Law doublings, results in a ratio of 150. By hand coding our inner loop with SSE instructions, we hope to be able to reach 2x higher performance with our N-body code, however.

Overall, the Space Simulator provides a reliable computing resource with unbeatable price/performance for our applications. It is fortunate that the niche of small, quiet computers that Shuttle targeted with the Shuttle XPC series was quite well suited to the architecture of the latest generation of Beowulf clusters.

6. ACKNOWLEDGMENTS

We thank Aric Hagberg for suggesting the name of the cluster, and Richard Fryer of BEOMAX for suggesting the Shuttle XPC chassis. This work was partially supported by the NASA Applied Information Science Research Program (AISRP). The supernova simulations were partially supported by the Scientific Discovery through Advanced Computing (SciDAC) program of the DOE, grant number DE-FC02-01ER41176. This work was performed under the auspices of the U.S. Dept. of Energy, and supported by its contract #W-7405-ENG-36 to Los Alamos National Laboratory.

7. REFERENCES

[1] D. Bailey, T. Harris, W. Saphir, R. van der Wijngaart, A. Woo, and M. Yarrow. The NAS parallel benchmarks 2.0. Technical Report NAS-95-020, NASA Ames Research Center, 1995.

[2] J. Barnes and P. Hut. A hierarchical O(NlogN) force-calculation algorithm. *Nature*, 324:446, 1986.

[3] D. J. Becker, T. Sterling, D. Savarese, J. E. Dorband, U. A. Ranawake, and C. V. Packer. BEOWULF: A parallel workstation for scientific computation. In *Proceedings of the 1995 International Conference on Parallel Processing (ICPP)*, pages 11–14, 1995.

[4] J. J. Dongarra, H. W. Meuer, and S. E. TOP500 supercomputer sites. *Supercomputer*, 13(1):89–120, 1997.

[5] J. J. Dongarra, H. W. Meuer, H. Simon, and E. Strohmaier. Top 500 supercomputer sites. `http://www.top500.org/`.

[6] C. L. Fryer and M. S. Warren. Modeling core-collapse supernovae in three dimensions. *Ap. J. (Letters)*, 574:L65, 2002.

[7] C. L. Fryer and M. S. Warren. The collapse of rotating massive stars in 3-dimensions. *Ap. J.*, 2003. (submitted).

[8] L. Greengard and V. Rokhlin. A fast algorithm for particle simulations. *J. Comp. Phys.*, 73:325–348, 1987.

[9] P. Ploumans, G. S. Winckelmans, J. K. Salmon, A. Leonard, and M. S. Warren. Vortex methods for high-resolution simulation of three-dimensional bluff body flows; application to the sphere at Re=300, 500 and 1000. *J. Comp. Phys.*, 178:427–463, 2002.

[10] J. Salmon and M. S. Warren. Parallel out-of-core methods for N-body simulation. In *8th SIAM Conf. on Parallel Processing for Scientific Computing*, Philadelphia, 1997. SIAM.

[11] J. K. Salmon, P. J. Quinn, and M. S. Warren. Using parallel computers for very large N-body simulations: Shell formation using 180k particles. In A. Toomre and R. Wielen, editors, *Proceedings of 1989 Heidelberg Conference on Dynamics and Interactions of Galaxies*. Springer-Verlag, New York, 1990.

[12] Spec cpu200 benchmarks. `http://www.specbench.org/cpu2000/`.

[13] M. S. Warren, C. L. Fryer, and M. P. Goda. The Space Simulator. `http://space-simulator.lanl.gov/`, 2002.

[14] M. S. Warren, C. L. Fryer, and M. P. Goda. The Space Simulator. In *Proceedings of CWCE '03*, San Jose, 2003.

[15] M. S. Warren, T. C. Germann, P. S. Lomdahl, D. M. Beazley, and J. K. Salmon. Avalon: An Alpha/Linux cluster achieves 10 Gflops for $150k. In *Supercomputing '98*, Los Alamitos, 1998. IEEE Comp. Soc.

[16] M. S. Warren and M. P. Goda. Loki – commodity

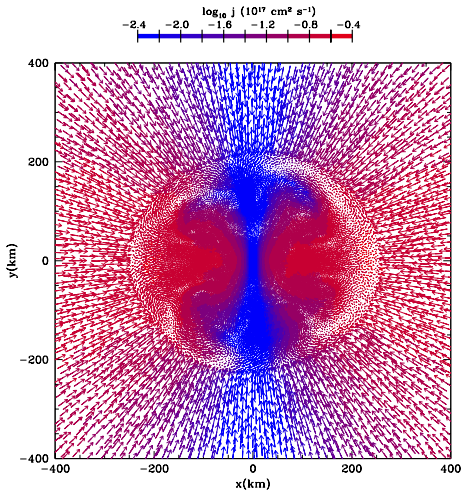

Figure 8: The image shows the angular momentum distribution a 0.5° slice across the core of a rotating supernova 40 ms after the core bounces. The colors denote the specific angular momentum of the material and the vectors show velocity direction and magnitude (vector length). Note that the bulk of the angular momentum lies along the equator (the angular momentum in the a 15° cone along the poles is 2 orders of magnitude less than that in the equator). The specific angular momentum in the equator over $10^{16}\mathrm{cm}^2\mathrm{s}^{-1}$ corresponding to a rotation velocity of nearly $5000\mathrm{kms}^{-1}$ and a rotational period of 250 ms. This angular momentum causes the star to deviate from spherical symmetry, leading to the emission of gravitational waves.

parallel processing. `http://loki-www.lanl.gov/`, 1996.

[17] M. S. Warren, A. Hagberg, D. Moulton, and D. Neal. The avalon beowulf cluster. `http://cnls.lanl.gov/avalon`, 1998.

[18] M. S. Warren, P. J. Quinn, J. K. Salmon, and W. H. Zurek. Dark halos formed via dissipationless collapse: I. Shapes and alignment of angular momentum. *Ap. J.*, 399:405–425, 1992.

[19] M. S. Warren and J. K. Salmon. Astrophysical N-body simulations using hierarchical tree data structures. In *Supercomputing '92*, pages 570–576, Los Alamitos, 1992. IEEE Comp. Soc.

[20] M. S. Warren and J. K. Salmon. A parallel hashed oct-tree N-body algorithm. In *Supercomputing '93*, pages 12–21, Los Alamitos, 1993. IEEE Comp. Soc.

[21] M. S. Warren and J. K. Salmon. A portable parallel particle program. *Computer Physics Communications*, 87:266–290, 1995.

[22] M. S. Warren, J. K. Salmon, D. J. Becker, M. P. Goda, T. Sterling, and G. S. Winckelmans. Pentium Pro inside: I. A treecode at 430 Gigaflops on ASCI Red, II. Price/performance of $50/Mflop on Loki and Hyglac. In *Supercomputing '97*, Los Alamitos, 1997. IEEE Comp. Soc.

[23] M. S. Warren, E. H. Weigle, and W. Feng. High-density computing: A 240-processor Beowulf in one cubic meter. In *SC '02*, Los Alamitos, 2002. IEEE Comp. Soc.

[24] W. H. Zurek, P. J. Quinn, J. K. Salmon, and M. S. Warren. Large scale structure after COBE: Peculiar velocities and correlations of cold dark matter halos. *Ap. J.*, 431:559–568, 1994.