Design and evaluation of a TOP100 Linux Super Cluster system
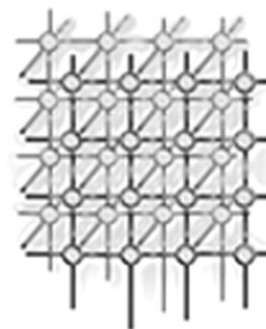
Niklas Edmundsson, Erik Elmroth[*,†], Bo Kågström,
Markus Mårtensson, Mats Nylén, Åke Sandgren and
Mattias Wadenstein

*High Performance Computing Center North (HPC2N), Umeå University,
SE-901 87 Umeå, Sweden*

SUMMARY

The High Performance Computing Center North (HPC2N) Super Cluster is a truly self-made high-performance Linux cluster with 240 AMD processors in 120 dual nodes, interconnected with a high-bandwidth, low-latency SCI network. This contribution describes the hardware selected for the system, the work needed to build it, important software issues and an extensive performance analysis. The performance is evaluated using a number of state-of-the-art benchmarks and software, including STREAM, Pallas MPI, the Atlas DGEMM, High-Performance Linpack and NAS Parallel benchmarks. Using these benchmarks we first determine the raw memory bandwidth and network characteristics; the practical peak performance of a single CPU, a single dual-node and the complete 240-processor system; and investigate the parallel performance for non-optimized dusty-deck Fortran applications. In summary, this \$500 000 system is extremely cost-effective and shows the performance one would expect of a large-scale supercomputing system with distributed memory architecture. According to the TOP500 list of June 2002, this cluster was the 94th fastest computer in the world. It is now fully operational and stable as the main computing facility at HPC2N. The system's utilization figures exceed 90%, i.e. all 240 processors are on average utilized over 90% of the time, 24 hours a day, seven days a week. Copyright © 2004 John Wiley & Sons, Ltd.

1.   INTRODUCTION

The number of Linux clusters in the high-performance computing area has increased rapidly over the past few years. From being cheap self-made platforms for parallel computing with a small number of standard PCs interconnected with a standard Ethernet network, recent clusters also include systems

---

[*]Correspondence to: Erik Elmroth, High Performance Computing Center North (HPC2N), Umeå University, SE-901 87 Umeå, Sweden.
[†]E-mail: elmroth@cs.umu.se

with true supercomputer performance. Today, clusters can also be bought as complete products from, e.g., traditional supercomputer vendors or specialized cluster vendors.

Part of the popularity comes from the fact that the systems can be self-made with reasonable effort, most often using standard desktop PCs and Ethernet networks. The self-made systems are often tailored for the specific use and completed with a small budget. A slightly larger budget and recent technology developments also open the possibility for self-made high-end supercomputers.

This contribution presents the development of a self-made, large-scale, highly integrated, rack-mounted Linux cluster with a high-end interconnection network. The system is truly self-made, as each individual PC was constructed from off-the-shelf PC components. The cluster was built during Spring 2002 and according to the TOP500 list of June 2002 [1], it was the 94th fastest computer in the world and the fastest computer in Sweden.

The system consists of 240 rack-mounted AMD processors interconnected with a high-performance SCI network. The cost for the system is around U.S. \$500 000 (5 000 000 Swedish Krona) excluding our own work, making the system extraordinarily cost-effective and possibly the cheapest system ever to be among the 100 fastest computers in the world.

This presentation includes a description of the system, the work needed to build it and benchmark results for the HP Linpack and NAS Parallel benchmarks.

The system was built by the High Performance Computing Center North (HPC2N). HPC2N is a Swedish national centre for scientific and parallel computing, providing support for research in application areas characterized by the need for various high-performance computing resources. The HPC2N partners are: Luleå University of Technology; Mid Sweden University; The Swedish Institute of Space Physics (IRF) in Kiruna; Swedish University of Agricultural Sciences (SLU) in Umeå; and Umeå University. This collaboration and coordination between universities and research institutes forms a competence network for high-performance computing, scientific visualization and virtual reality in Northern Sweden.

In the end of 2001, HPC2N received a donation from the Kempe Foundations for building a large-scale cluster. The system, named Seth, was built by HPC2N after careful testing and selection of appropriate components. In the following, we review some more details about the processing nodes, the network, the work required to build the system and a careful performance evaluation of the system using state-of-the-art benchmarks.

2. SYSTEM DESCRIPTION

The system consists of 240 AMD Athlon MP2000+ processors [2], organized in 120 rack-mounted dual nodes. Each node has a theoretical peak performance of 6.7 Gflops s^{-1} giving a total system peak performance of 800 Gflops s^{-1}. For high parallel performance the system is equipped with the third generation, three-dimensional, high-bandwidth, low-latency SCI interconnect from Dolphin ICS [3]. The network connects the nodes in a three-dimensional torus.

There are several reasons for selecting these components, most of which are not of general interest at the time of writing this manuscript, so therefore we only give some brief comments. For example, the main reason for choosing AMD processors instead of Intel was that, at the time of purchase, the Pentium 4 processors and motherboards were not yet available for rack-mounted configurations and the AMD processors outperformed the Pentium 3 on our and most other tests relevant for this system.

The main reason for selecting the SCI network instead of Myrinet, which also was considered, was the significantly lower cost. From our investigations we also expected slightly higher performance of the SCI network, and we had positive experiences with the SCI network from a previously built small cluster at HPC2N [4]. Notably, as our installation was the first larger system using the third generation of the SCI network from Dolphin ICS, we did not have the opportunity to perform large-scale comparisons between the networks before making our decision.

2.1. Processor nodes

The nodes are built with TYAN Tiger MPX motherboards using the AMD-760 MPX chipset. Each node is equipped with two processors, 1–3 Gbytes of DDR memory, a disk, serial console connection, fast Ethernet and an SCI network card. The system is mounted in 12 racks, each with 10 nodes.

The AMD MP2000+ is a 1.667 GHz processor with support for two-processor shared-memory configurations. It includes a superscalar, fully-pipelined floating point engine capable of performing two arithmetic floating point operations per clock cycle. It also includes hardware support for data prefetch and speculative translation look-aside buffers (TLBs).

The memory hierarchy includes in total 384 KB on-chip cache, organized as 64 KB instruction cache and 64 KB data cache for a total of 128 KB level 1 (L1) cache and 256 KB of integrated, on-chip level 2 (L2) cache. An optimized cache coherency protocol manages data and memory traffic, as well as innovative 'snoop' buses, which offer high-speed communication between the CPUs in a multiprocessing system.

A 266 MHz system bus gives a peak data rate of 2.1 Gbytes s^{-1}. A high-speed 66 MHz/64-bit PCI bus provides high bandwidth towards the SCI network.

For further technical specifications, see the HPC2N Web page [5].

2.2. High-performance network

The high-performance network is a WulfKit3 SCI-network, a three-dimensional torus switching topology, offering high scalability and parallel performance. It connects the PCI bus to three high-performance SCI rings providing the connections of the three-dimensional torus. Our torus is organized as a $6 \times 4 \times 5$ grid. The network offers 667 Mbytes s^{-1} hardware bandwidth and sub-microsecond hardware latency.

WulfKit3 uses the ScaMPI message-passing library developed by Scali AS of Oslo, Norway. ScaMPI is a native high-performance MPI implementation that features multi-thread safe, command line replication to all the involved nodes in the cluster and simultaneous inter-node and intra-node MPI communication. ScaMPI can also interface with low-level hardware and operating system functions for improved performance and reliability.

The software also includes modules for monitoring and management functions for the cluster and the interconnecting network, as well as secure remote management and application launching.

3. SYSTEM SOFTWARE

The system runs the Debian 3.0 Linux distribution [6] with kernel version 2.4.18. The drivers and operating software for the high-performance network is provided by Scali AS as part of their SSP

Table I. Installed compiler collections.

Collection	Supported languages	Version installed
Intel	C, C++ and Fortran 95	6.0
Portland Group	C, C++, Fortran 90, Fortran 77 and HPF	4.0
GNU	C, C++ and Fortran 77	2.95 and 3.1

(Scali Software Platform) software [7]. In addition to the drivers and libraries for the SCI adapters, SSP also provides tools for diagnostics and a tuned MPI implementation, ScaMPI.

The batch-queue on the system is handled with a combination of OpenPBS [8] and the Maui scheduler [9]. The use of Maui as the scheduler makes the efficient use of the system possible under a variety of work-loads, as well as facilitating the enforcement of user and system policies.

The installed compiler collections are displayed in Table I. In order to evaluate the compilers we compared their performance using the serial versions of the class W NAS Parallel benchmarks. During these tests we have found that the best performance is obtained with codes compiled with the Intel compilers [10]. The Portland Group Compiler Collection [11] offers the most complete set of features and language support. Included in the tests and evaluation is also the GNU Compiler Collection [12].

4. SYSTEM ASSEMBLY

As the system was built from basic components, substantial work was required to build the 120 dual CPU nodes. This work was performed by HPC2N staff and some students and organized in an assembly line with a number of stations.

First, all the unpacking was done at a designated area. The first assembly station was mainly aimed at mounting the power supplies and hard disks, and to perform some cable and sheet-metal work. At the second station, all the static-sensitive assembly was performed, including mounting the processors, memory and cooling on the motherboards, as well as mounting the boards into the chassis. At the next station, function tests were performed for each individual node. After passing the tests, the SCI-cards were installed and the chassis were completed at the fourth station.

In parallel with the work in this assembly line, the twelve racks were built by another group of people. At the end of the assembly line, the rack-rails were attached on the nodes and the nodes were installed in the racks.

With all 120 nodes in place, the work continued with electrical installations, including one power distribution unit (PDU) per frame. Moreover, all the cables for the fast Ethernet and serial consoles were also connected. Finally, the SCI-network connecting the nodes in a three-dimensional torus was set up in a $6 \times 4 \times 5$ configuration.

The work that started with the unpacking of 800–1000 boxes was completed after mounting approximately 10 000 screws and over 700 network and power cables with a total length of 1000–1500 meters (not including the cables inside each node).

5. BENCHMARKING THE CLUSTER

We evaluate the performance of the cluster through a series of benchmarks. First we use the STREAM and Pallas MPI benchmarks for evaluating the raw memory bandwidth and network performance, respectively.

The 'practical peak performance' of individual processors and individual dual-nodes are evaluated through performance measures of the Atlas DGEMM subroutine, for dense matrix–matrix multiplication. This roughly shows the maximum performance that can be obtained on individual processors and nodes for highly optimized codes that make close-to-perfect use of the registers and cache memories in the memory hierarchy.

The High-Performance Linpack (HP Linpack) benchmark measures the performance by solving a large-scale dense linear system of equations using the complete computer system. As DGEMM is the most important underlying subroutine, HP Linpack is also characterized by favourable memory reference patterns and highly optimized software for memory hierarchies. However, HP Linpack also requires a substantial amount of inter-processor communication and is especially sensitive to high communication latency. Moreover, the subproblems handled by DGEMM involve highly rectangular matrices, which are normally considered less favourable than square matrices, from a performance point of view.

Finally, we use the NAS Parallel benchmarks to evaluate the parallel performance on highly non-optimized Fortran codes. We present three of the NAS benchmarks that show typical characteristics for the system, including the ability to efficiently utilize two processors per node without optimizing for the memory hierarchy and scalability characteristics for real application-type software. Moreover, we compare these results to the NAS benchmark performance of two Cray T3E systems, both in terms of real performance and in terms of performance up-scaled by the frequency differences of the older Cray systems and the new Linux cluster.

5.1. STREAM

STREAM is a synthetic benchmark for measuring the sustainable memory bandwidth in Mbytes s^{-1} [13]. The performance is measured for the four vector operations described in Table II. Our results are presented in Figure 1 for one and two processors in one node.

One processor can reach a sustained memory bandwidth of between 651 and 769 Mbytes s^{-1} for these four operations. When adding a second CPU into the operations, the sustained bandwidth increases by 17–45%, with a maximum of 964 Mbytes s^{-1}. For applications where performance is bounded by the memory bandwidth on one processor, the gain by using a second processor in a dual node is expected to be in this range. However, by making efficient use of the memory hierarchy, the gain can be significantly higher, which is investigated using the Atlas DGEMM in Section 5.3.
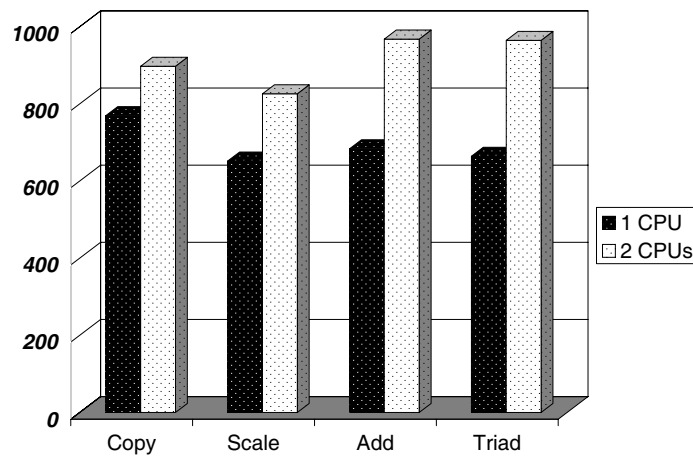
5.2. The Pallas MPI benchmark

The Pallas MPI benchmark is a benchmark suite for evaluating MPI implementations and parallel platforms using MPI [14]. The suite comprises a set of tests of communication operations specified by MPI. It is available in two versions for MPI-1 and MPI-2, respectively. In this document we consider the MPI-1 version, release 2.2.

Table II. Operations performed in the STREAM benchmark.

Name	Operation	Bytes per iteration	Flops per iteration
Copy	$a(i) = b(i)$	16	0
Scale	$a(i) = q*b(i)$	16	1
Sum	$a(i) = b(i) + c(i)$	24	1
Triad	$a(i) = b(i) + q*c(i)$	24	2

Figure 1. Performance in Mbytes s^{-1} for the STREAM benchmark for one and two processors in a dual node.

The benchmark includes a set of tests for basic communication operations. The tests are available in 'standard' versions and in 'multi' versions, where the 'multi' versions perform the benchmark simultaneously for multiple processor groups.

Figure 2 shows the performance on the Pallas ping-pong benchmark. The curve marked 'Standard' shows the memory bandwidth utilized when one processor sends a message to a processor that immediately returns the message to the first processor. Results are presented for increasing message sizes up to 4 Mbytes (2^{22} bytes in the figure). The maximum bandwidth obtained was 230 Mbytes s^{-1} and is the same for messages larger than those plotted in the figure. The minimum latency measured is 3.7 μs.

The results marked 'Multi(16): max' and 'Multi(16): min' show the corresponding highest and lowest results obtained when simultaneously performing the ping-pong benchmark between eight pairs of processors. The immediate observation is that the fact that several processors perform the operation simultaneously does not have a large effect on the performance. The best results are in the level of the
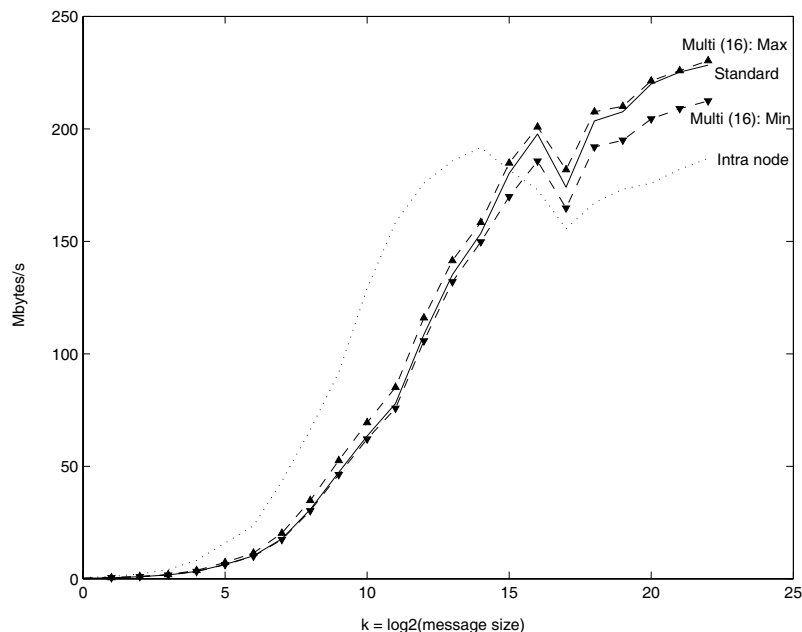
Figure 2. Performance in Mbytes s^{-1} for the MPI ping-pong benchmark.

'standard' results, or actually slightly larger due to small variations, and the lowest results are roughly 10% lower.

The fourth curve marked 'Intra node' shows the results for the MPI ping-pong benchmark for two processors within the same node, i.e. the performance is not limited by the interconnection network. For small messages, these results are around a factor of two higher than the 'Standard' results, but for larger message sizes the performance drops below the performance achieved between nodes in the cluster.

Notably, a minimum in transfer bandwidth is observed at message sizes of 128 KB for all tests. This is most likely caused by cache effects when copying between buffers in the MPI library.

Finally, we measured the time for performing a two processor MPI barrier operation as 5.3 μs.

We have also performed the ping-pong benchmark with the SCI-MPICH MPI library [15]. Compared to the ScaMPI results, SCI-MPICH shows within 10% lower performance for message sizes up to 32 bytes (due to slightly higher latency) and within 5% for messages 64 bytes to 128 KB. Notably, SCI-MPICH shows the same performance drop as ScaMPI for 128 KB messages. After this performance drop, SCI-MPICH does not recover as well as ScaMPI, and reaches around 180 Mbytes s^{-1} for larger messages.
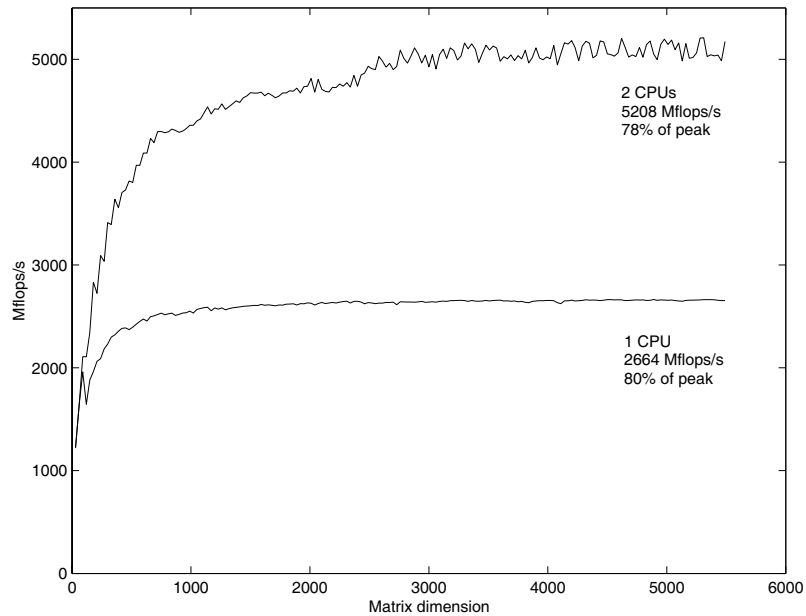
Figure 3. Performance in Mflops s^{-1} for Atlas DGEMM on one and two processors.

5.3. Atlas DGEMM

The performance obtained with a highly optimized level-three BLAS DGEMM routine [16] for matrix–matrix multiplication is often referred to as the 'practical peak performance' of the system. DGEMM performs $C = \beta C + \alpha \, \mathrm{op}(A) \cdot \mathrm{op}(B)$, where α and β are scalars, and C, A and B are matrices of size $m \times n$, $m \times k$ and $k \times n$, respectively. Here, $\mathrm{op}(A)$ is A or its transpose A^{T}. The fact that this operation involves $\mathcal{O}(n^3)$ floating point operations on $\mathcal{O}(n^2)$ data makes it well-suited for achieving high rates of floating point operations per second.

Highly optimized DGEMM routines are based on blocking strategies that allow for the reuse of data at different levels of the memory hierarchy (see, e.g., [17,18]). Such blockings typically make close to perfect use of the memory hierarchy and thereby reduce the effect of, e.g., low memory bandwidth.

For evaluation of the practical peak performance, in Figure 3 we present the results for the Atlas DGEMM [19] for square matrices of sizes from 30 to 5490, with steps of 30. Results are presented for the non-transposed case (i.e. $\mathrm{op}(A) = A$ and $\mathrm{op}(B) = B$) both for one processor and for two processors in one node, using the threaded version of Atlas DGEMM. It should be noted that slightly higher performance often may be achieved for the case where A and B^{T} are used, since that leads to operations on elements stored with stride 1 in both A and B.

The single processor results reach at most 2.664 Gflops s^{-1}, which corresponds to 80% of the theoretical peak performance. Performance close to this was already obtained for moderate matrix sizes, e.g. 95% of this performance was obtained for all matrix sizes 750 and larger.

With a parallel performance of 5.208 Gflops s^{-1} on a dual node the parallel speedup is close to perfect (1.96) and corresponds to a parallel efficiency of 98%.

The DGEMM results clearly demonstrate that with highly optimized software it is possible to reach close to peak performance on both one and two processors on the dual nodes of the cluster. That is, it has been shown that the lack of memory bandwidth demonstrated in the STREAM benchmark when utilizing two processors per node is not the bottleneck for DGEMM, with an efficient utilization of the different levels of the memory hierarchy.

5.4. HP Linpack

The HP Linpack benchmark measures the performance obtained when solving a large-scale dense system of linear equations, $Ax = b$, where A is of size $N \times N$. The tests have been performed using the HPL software package available from Netlib [20] with Atlas as the underlying BLAS library. For this cluster, a result of 480.7 Gflops s^{-1} was obtained for solving a system with $N = 116\,100$. The problem size (known as $N_{1/2}$) required to reach half of this performance was 24 570. The maximum result corresponds to 60.1% of the peak performance of the complete system and is 110 times higher than the HP Linpack result obtained on one node. It also corresponds to 75.2% of the accumulated processor DGEMM performance and 76.9% of the accumulated node DGEMM performance.

Seth in a TOP500 perspective

The TOP500 list of the world's fastest supercomputers is published twice every year. The list is ranked by the performance on the HP Linpack benchmark. With the performance of 480.7 Gflops s^{-1}, the HPC2N Super Cluster was ranked 94th on the list of June 2002 [1]. In total, there are 42 Intel-based and seven AMD-based clusters on this list. Out of all the systems on the list, 14 systems are labelled self-made. The label self-made does, however, not require that the individual nodes are self-made. Most of these systems are built from standard PCs or workstations. The HPC2N Super Cluster is the third fastest of the AMD-based clusters.

Three systems on the list are built with the WulfKit interconnect. The HPC2N Super Cluster is the fastest of these and is also the only one with the three-dimensional torus topology.

5.5. NAS Parallel benchmark

The NAS Parallel benchmark is a set of eight programs derived from computational fluid dynamics (CFD) applications. Five of the programs are application kernel routines and three are synthetic benchmarks, with behaviour similar to real applications. The benchmark comes in two sets, of which NPB2 is that relevant for large-scale clusters. The test problems are available in three classes A, B and C, of which A contains the smallest problems and C contains the largest.

Table III. True NAS Parallel benchmark class C results for Cray T3E systems and the results up-scaled by the CPU frequency difference compared to AMD MP2000+.

	CPUs	Results	'Frequency up-scaled results'
BT			
Cray T3E-1200	16	68.0	189.0
Cray T3E-900	64	55.7	206.6
CG			
Cray T3E-1200	16	9.3	25.9
Cray T3E-900	64	9.3	34.5
Cray T3E-900	128	11.8	43.8
LU			
Cray T3E-1200	16	73.1	203.2
Cray T3E-900	64	60.4	224.1
Cray T3E-900	128	56.0	207.8

The basic codes for the benchmark are available from the benchmark's Web page [21]. The benchmark admits the optimization of the codes, but we have chosen to run the original benchmark codes as is. These codes are not optimized, e.g. for memory hierarchies, which makes this benchmark a good test for the system's performance on non-optimized 'dusty-deck' Fortran codes.

In the following, we present results for three of these benchmarks, BT, CG and LU, for all three classes of test problems, i.e. A, B and C. These three benchmarks have different characteristics and are selected in order to highlight the different aspects of the performance of the cluster.

In order to evaluate the results, we make comparisons with the corresponding results for Cray T3E systems for class C problems. The Cray systems are interesting for these comparisons, both for being a relatively common and well-recognized supercomputer architecture and for having a similar network topology as the cluster. In Table III, we present both real performance numbers available at [21] and scaled performance numbers, where the real numbers are up-scaled by the frequency-difference compared to the AMD MP2000+. This difference is a factor of 3.71 and 2.78 for Cray T3E-900 and Cray T3E-1200, respectively. Such a scaling of the benchmark results would correspond to an absolutely ideal performance scaling of the system. In practice, we would expect lower results due to an increasing significance of bottlenecks in the memory subsystem and the interconnection network. Hence, the up-scaled numbers are likely to favour the Cray systems.

In the following three subsections, the Cray results in Table III are used for comparisons with the NAS benchmark results for Seth.

For each benchmark, we present the results in Mflops s^{-1} per processor, for varying numbers of processors. We present results using both one and two CPUs per node, in order to be able to evaluate the performance and bottlenecks of the CPUs, the nodes and the network.

BT benchmark

The BT benchmark solves multiple systems of 5×5 block tridiagonal linear systems of equations. Figure 4 shows the results in Mflops s^{-1} per processor for all three classes of the benchmark. As seen from both the nearly horizontal lines of the performance per processors and the nearly linear speedup curves, this benchmark shows the excellent scalability properties of the cluster.

The results also show a fairly low efficiency when going from one to two CPUs per node, with a parallel efficiency around 70%. This follows from the fact that the code does not make good use of the memory hierarchy and the memory bandwidth does not match the speed of two processors without a reasonable amount of data reuse in the cache memories.

The figure also shows that the cluster performance is similar to a 'frequency up-scaled' Cray T3E when using one CPU per node, i.e. compared with the up-scaled results of 189.0 and 206.6 on 16 and 64 processors, respectively.

CG benchmark

Figure 5 shows the performance for solving non-structured sparse linear systems with a conjugate gradient method. Here, the efficiency when using two CPUs per node is close to perfect, i.e. the performance per processor on p processors in n nodes is about the same as the performance per processor when using $2p$ processors in n nodes.

The parallel speedup is also very good here, especially up to 32 processors for all three classes of the benchmark, though not quite as high as for the BT benchmark.

The up-scaled Cray T3E-900 results per processor are at the same level as the corresponding results for the cluster, when using both one and two CPUs per node. The up-scaled Cray T3E-1200 results are clearly lower than both class C results for the cluster on 16 processors.

For some cases, the performance per processor actually increases with increasing number of processors. This anomaly follows from the reuse of data in the level-2 cache memory that automatically occurs when the problem becomes small enough. We further comment on this issue in our analysis of the NAS LU benchmark below, as the effect of this phenomena is even larger in that case.

LU benchmark

The final NAS benchmark results presented in Figure 6 are for solving 5×5 lower and upper triangular systems from Navier–Stokes equations. These results highlight the effects of the memory subsystem when running non-optimized codes. For all three classes of the benchmark, the performance per processor increases when the problem size (array data) per processor becomes small enough (up to a certain point when the problem size becomes too small). This performance increase follows from cache reuse that automatically occurs when significant parts of each processor's data fit in the L2 cache memory.

As a consequence of these cache effects, the parallel speedup is extreme. The speedup is at most around 180 on 128 processors. The ability to make an efficient use of two processors per node is not perfect as for the CG benchmark, but is better than for the BT case.

Finally, the 'frequency up-scaled' Cray T3E results are below the cluster's single-CPU results for 16 and 64 processors and below the dual-CPU results for 128 processors.
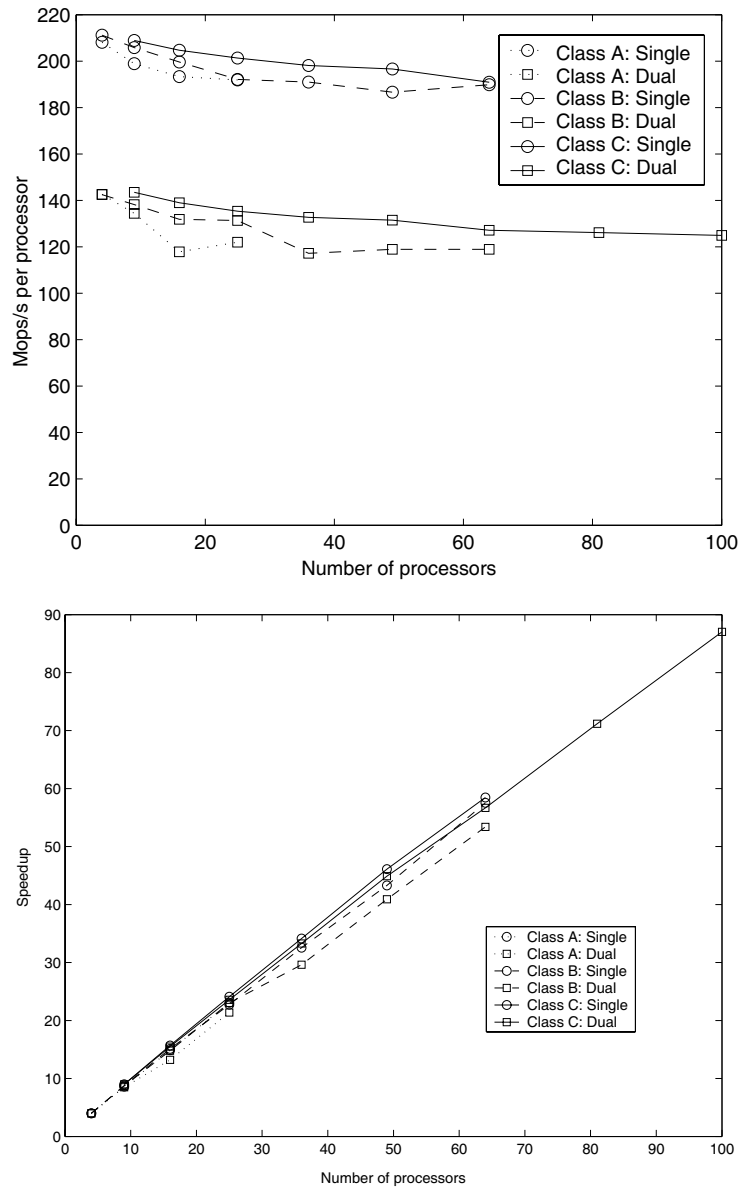
Figure 4. Performance in Mflops s^{-1} per processor and parallel speedup for the NAS BT benchmark.
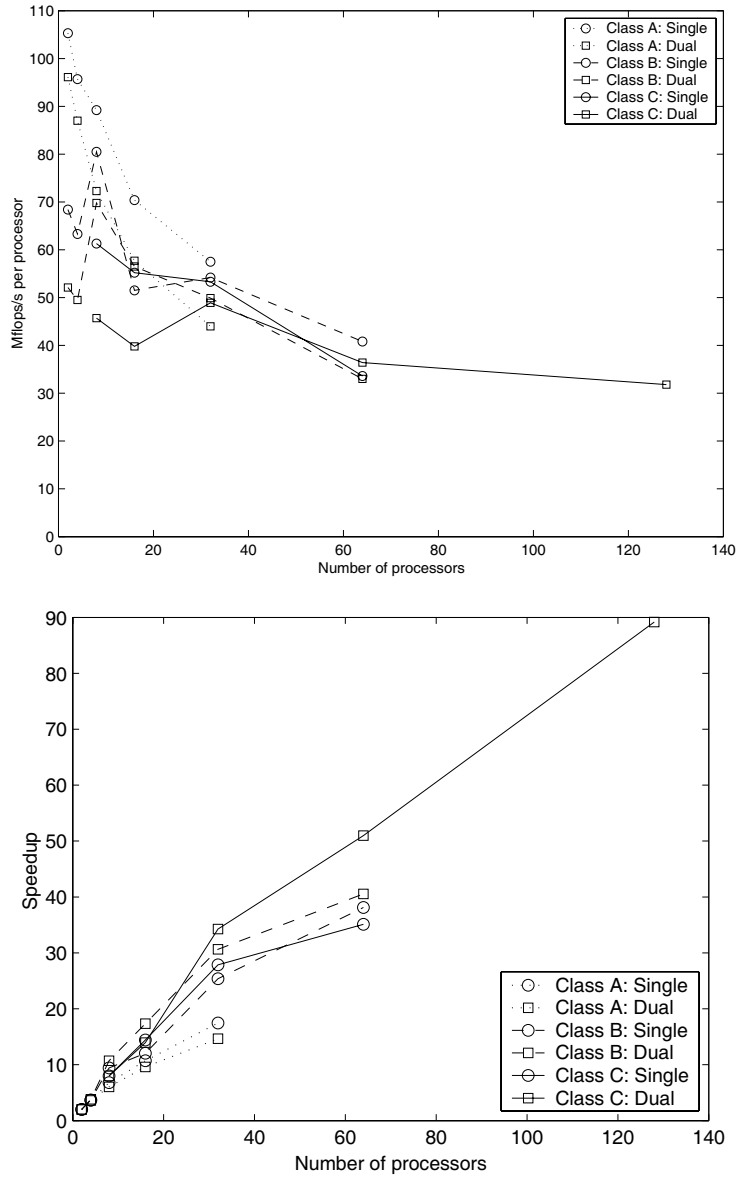
Figure 5. Performance in Mflops s^{-1} per processor and parallel speedup for the NAS CG benchmark.
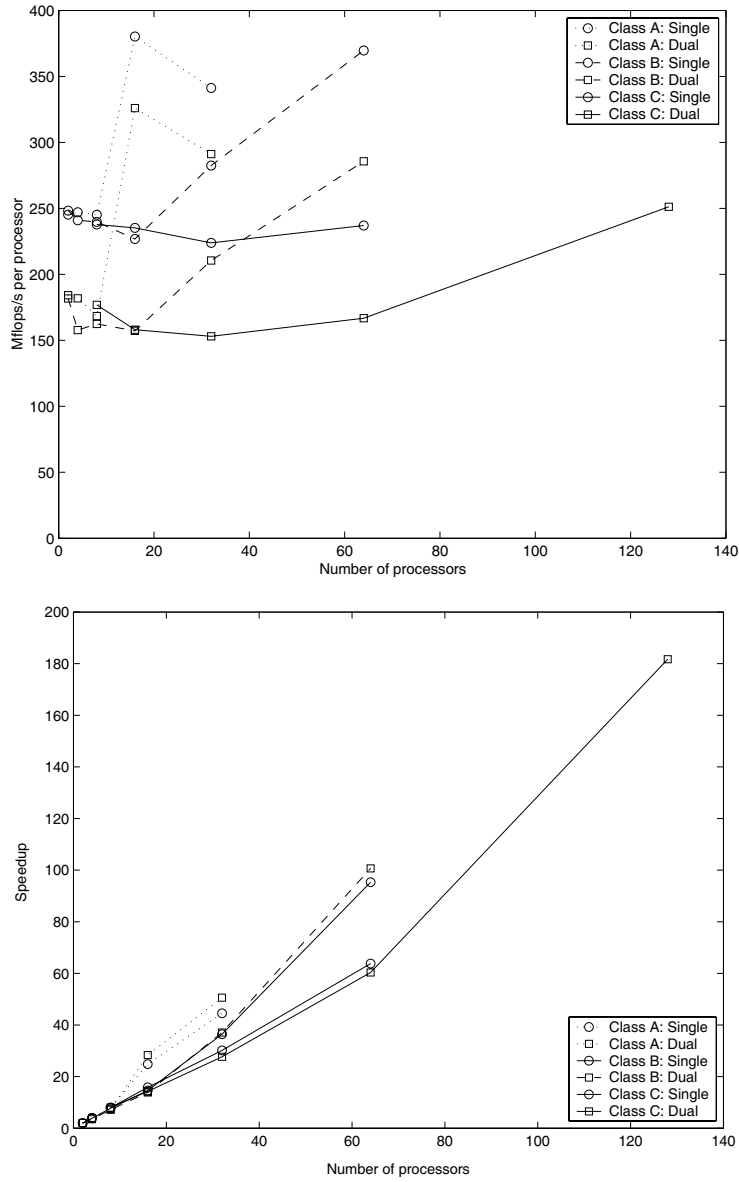
Figure 6. Performance in Mflops s^{-1} per processor and parallel speedup for the NAS LU benchmark.

6. CONCLUDING REMARKS

We have presented a truly self-made high-performance Linux Super Cluster with 240 processors in 120 dual nodes, interconnected with a high-bandwidth, low-latency SCI network. The performance of the system has been evaluated using a number of state-of-the-art benchmarks.

The peak performance of the system is 800 Gflops s^{-1}. For code optimized memory hierarchies, it has shown excellent performance on single CPUs, on both CPUs in dual nodes and on the complete system. Such results include a single-CPU DGEMM and a dual-CPU DGEMM performance of 80% and 78% of the respective peak performance. The HP Linpack result of 480.7 Gflops s^{-1} corresponds to 60.1% of the complete system's peak performance and placed the system in position 94 on the TOP500 list of June 2002.

For unoptimized code, using the original codes for the NAS Parallel benchmarks, the system also shows excellent scalability. The ability to use efficiently two processors per node for unoptimized code is sometimes perfect and sometimes moderate due to the fact that the processors need to share the available memory bandwidth. We have compared the NAS benchmark results with the corresponding 'frequency up-scaled' results for Cray T3E systems. In summary, the cluster shows a performance at least as good as the 'frequency up-scaled' Cray results when using one CPU per node. The same is true for some codes of the NAS benchmark when using two CPUs per node; however, for codes where the memory bandwidth is already a bottleneck when using one CPU, the gain from using two CPUs per node is lower.

We remark that our users' codes cover the whole range from dusty deck Fortran codes to optimized programs that make use of highly tuned software libraries for major computations. Their experiences give further support for our conclusions that unoptimized codes make fairly inefficient use of the second CPU in each node, while very high efficiency on both CPUs are obtained with codes optimized for the memory hierarchy. Moreover, the network bandwidth and latency are well in balance with the processor speed and thereby facilitates efficient use of the complete system, also for real applications.

In summary, the system is extremely cost-effective and shows the performance we would expect of a large-scale supercomputer system with distributed memory architecture. For the benchmarks presented, the interconnection network does not show to be a significant limiting factor. The system can use two CPUs per node with very high efficiency, given that the software is optimized for making efficient use of the different levels of the node's memory hierarchy. For unoptimized and memory bandwidth demanding codes, the efficiency for the second CPU per node is moderate, but normally enough to motivate the small extra cost for adding them to the system.

The system has been in production use by the HPC2N academic users since June 2002, and since August 2002 the utilization has been, on average, over 90%, i.e. all the processors have been utilized on average over 90% of the time, 24 hours per day, seven days per week.

REFERENCES

1. TOP500 list of June 2002. http://www.top500.org/list/2002/06/ [June 2002].
2. Advanced Micro Devices, Inc. http://www.amd.com [August 2003].
3. Dolphin Interconnect Solutions, Inc. http://www.dolphinics.com/ [August 2003].
4. Augustsson F. Beowulf cluster, an affordable concept. *Master's Thesis UMNAD-453.03*, Department of Computing Science, Umeå University, SE-901 87 Umeå, Sweden, 2003.
5. HPC2N Super Cluster Specifications. http://www.hpc2n.umu.se/resources/super-cluster.html [August 2003].
6. Debian. http://www.debian.org [August 2003].
7. Scali. http://www.scali.com [August 2003].
8. Portable Batch System. http://www.openpbs.org [August 2003].
9. Supercluster.org. Center for HPC Cluster Resource Management. http://www.supercluster.org [August 2003].
10. Intel Compilers. http://www.intel.com/software/products/compilers/ [August 2003].
11. The Portland Group Compiler Technology. http://www.pgroup.com [August 2003].
12. GNU Compiler Collection. http://www.gnu.org/software/gcc/ [August 2003].
13. McCalpin JD. Sustainable memory bandwidth in current high performance computers. *Technical Report*, Silicon Graphics, Inc., October 1995.
14. Pallas. Pallas MPI Benchmarks—BMP, Part MPI-1. *Technical Report*, Pallas Gmbh, Herülheimer Str. 10, D-50321 Brühl, Germany, 2000.
15. SCI-MPICH. http://www.lfbs.rwth-aachen.de/~joachim/SCI-MPICH [August 2003].
16. Dongarra J, Du Croz J, Duff I, Hammarling S. A set of level 3 basic linear algebra subprograms. *ACM Transactions on Mathematical Software* 1990; **16**(1):1–17.
17. Kågström B, Ling P, Van Loan C. GEMM-based level 3 BLAS: High-performance model implementations and performance evaluation benchmark. *ACM Transactions on Mathematical Software* 1998; **24**(3):268–302.
18. Kågström B, Ling P, Van Loan C. GEMM-based level 3 BLAS: Portability and optimization issues. *ACM Transactions on Mathematical Software* 1998; **24**(3):303–316.
19. Whaley RC, Dongarra JJ. Automatically tuned linear algebra software. *Technical Report TN 37996-1301*, Computer Science Department, University of Tennessee, 1997.
20. Petitet A, Whaley RC, Dongarra J, Cleary A. HPL—A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers, Version 1.0. http://www.netlib.org/benchmark/hpl/ [August 2003].
21. NAS Parallel Benchmark. http://www.nas.nasa.gov/Software/NPB [August 2003].