

High Performance Computing and Beowulf Clusters

R.J. Allan, S.J. Andrews and M.F. Guest

Daresbury Laboratory, Daresbury, Warrington WA4 4AD, UK

Introduction

Networks of personal computers (so called Beowulf systems) composed of fast PCs configured with large quantities of RAM and hard disk, and running the Linux operating system are becoming more and more attractive as cheap and efficient platforms for distributed applications. The main drawback of a standard Beowulf architecture is the poor performance of the conventional inter-process communication mechanisms based on RPC, sockets, TCP/IP, Ethernet. Such standard mechanisms are thought to perform poorly both in terms of throughput and of message latency. Nevertheless, there is increasing interest in the use of commodity "off-the-shelf" components as building blocks for high-performance computing. This is evident in many areas. An example within the UK academic community is provided by the recent round of EPSRC's Strategic Equipment Initiative (SEI). This includes equipment requests for more commodity-based clusters than proprietary SMP-based solutions (e.g. Origin 2000), in stark contrast to previous rounds of such Initiatives.

Two main modes of operation are often envisaged for "Beowulf-class" systems; the first for high performance parallel applications and the second for high throughput serial codes, generally in a specialised area. All other things being equal, therefore, the choice of an appropriate architecture is dependent upon the use to which the machine is to be put. However the decision, as for so much in life, is not quite that simple.

With a high rate of change in the field there is a bewildering choice of processor, motherboard, interconnect, resource management tools etc. and it is vital for potential owners to be well informed on the pros and cons of individual components and their combinations both in general and for the envisaged application.

By far the most popular cpu choices are the Pentium and Alpha although the AMD Athlon is often considered and IA-64 based architectures are in the pipeline. The logic behind using commodity processors is not hard to see given their ubiquitous presence in a variety of vendor platforms and their creditable performance against many proprietary RISC processors in floating point applications (see www.cse.clrc.ac.uk/Activity/DisCo). When

considering an operating system the overwhelming preference is for some flavour of Unix, either proprietary or, increasingly, Linux. Again there is a critical mass forming here leading to an installed base of tools and devices which creates a driving force towards an ad hoc standard.

Interconnects can be classified as: cheap but slow; or fast but expensive. Fast (sic) Ethernet is almost universally used for the former but is currently limited in both latency and bandwidth (see HPCProfile issue 24). This can be quite sufficient if there is little requirement for inter-processor communication but is woefully slow for high performance, fine-grained parallel applications. To address this there are a number of commercial switches available that make use of dedicated NICs to off load the communication i/o from the cpu and the performance of such interconnects can approach that found in proprietary supercomputers (again see previous article). The advent of Gigabit Ethernet enables the creation of high performance networks to be built from commodity components, although processor and bus limitations currently prevent realisation of its full potential. A useful starting point for all things Beowulf is www.beowulf.org

To investigate the potential of this class of system, a number of these Beowulf-class prototype systems have been assembled and evaluated at Daresbury. The programme of evaluation is focused on both system software and hardware, and on assessing the likely delivered performance across a broad spectrum of end-applications. The programme aims to inform the community over the wide variety of available options, from choice of CPU (Alphas, single-, dual-, quad-Pentiums, caches, Xeon or old memory subsystems) to choice of interconnect (Ethernet, Myrinet, QsNet etc.). We provide below an update on progress, with a particular focus on application performance. As part of this assessment, we also take the opportunity to present initial application results on the IBM SP/WH2-375 recently installed at Daresbury.

32-processor Pentium Beowulf II with Fast Ethernet (Beowulf1)

This commodity based machine, funded by EPSRC, has been used to investigate the performance of the Pentium platform with

channel-bonded Ethernet, Myrinet and, currently, 100BaseT Fast Ethernet.

Master Node

- 440LX based motherboard with on-board Adaptec Ultra Wide SCSI
- Intel Pentium II 266 MHz CPU
- 256 MB SDRAM
- 4 MB AGP Graphics card
- 2 x PCI Fast Ethernet NIC
- 4 GB (system) and 9GB (user) Ultra Wide SCSI disk
- CD-ROM
- 17inch monitor, keyboard and mouse

Compute Nodes (x32)

- 440BX based motherboard
- Intel Pentium III 450 MHz CPU
- 256 MB ECC PC100 SDRAM
- Minimal AGP Graphics card
- 2 x PCI Fast Ethernet NIC
- 10 GB U-DMA (ATA-33) disk

Networking

- 2 x Extreme Summit48 Fast Ethernet switches

The master node and the compute nodes are attached via 1 NIC to one of the Summit48 switches. This provides a link for NFS and other IP traffic between the compute nodes and the master which is isolated from the site network. The master node is connected via its second NIC to the site network. All the compute nodes are connected via their second NIC to the second Summit48 switch. This provides a single network link dedicated to parallel communications between the nodes.

All 32 compute nodes are connected to a single monitor and keyboard via a set of 8 port KVM switches. This provides us with the ability to directly logon to the compute nodes if something goes wrong whilst avoiding unnecessary cost and using up space.

All the compute nodes are configured the same way, with a 512 MB swap partition, a 512 MB system partition with a minimal RedHat 6.0 installation (takes around 120 MB), and approximately 9 GB available as scratch space. User's home directories are NFS mounted from the master node.

The present scheduling system is provided by PBS (see www.nas.nasa.gov/Software/PBS) which is designed to operate on networked, multi-platform UNIX environments, including heterogeneous clusters of workstations,

supercomputers and massively parallel systems.

32-Processor Alpha Beowulf III with QsNet (Loki)

Our internally-funded Alpha machine, Loki, currently contains 17 dual processor UP2000 boards from API (www.alpha-processor.com/products/up2000-board.shtml) including the master node. The principal interconnect solution is QsNet, in which the network interface is based on QSW's Elan III ASIC (see www.quadrics.com/web/developer/tech/index.html). A secondary Fast Ethernet is used for cluster management and a further Ethernet is available for communication traffic enabling comparison of the two interconnects on the same hardware platform.

The Alpha/ QsNet combination is already well established under the Compaq Tru64 OS and we have been working closely with QSW to evaluate the port of this to Linux using RedHat 6.1.

Using API's 64-bit platform and the Linux open-source operating system coupled with QsNet, a high-bandwidth, ultra low-latency interconnect for commodity SMP nodes, the systems offer one of the highest possible interconnect performances available. The solution, based on QSW's third-generation "Elan" ASIC and API's UP2000 Ultimate Performance Series motherboards, consists of a network interface and a high-performance multi-rail data network.

Master node

- Alpha 21264A, 1 GB; 16 GB UW-SCSI user disk

Compute processors (32)

- Alpha EV67 (667 MHz) 21264A, 512 MB ECC SDRAM (1 GB per node), 9 GB SCSI disk, connected to the master via a Fast Ethernet switch.
- 2nd Fast Ethernet switch dedicated to the parallel communications traffic.
- High Performance Interconnect via QSW Elan III NICs
- Linux MPI + Quadrics 16-way switch (QsNet)

16-processor Beowulf with AMD Athlon K7 (Wulfgar)

A third Beowulf IV machine at the Rutherford-Appleton Laboratory, Wulfgar, has 16 AMD K7 processors running at 800 MHz and is

connected by Myrinet or Fast Ethernet. This was used for comparison in the table below using a similar ping-pong test.

Processor and Interconnect Performance

As one might expect from a year-old processor, the 450MHz Pentium III with a SPECfp95 rating of only 13.3, is considerably inferior in performance to today's close-to-gigahertz IA-32 chips. Nonetheless this emphasises an important feature of the commodity Beowulf approach, namely that instead of using the latest, fastest chip, second, or even third-line processors can be harnessed inexpensively. In contrast the 667 MHz Alpha EV67 has a SPECfp95 of 54.1, still demonstrating a performance twice that of the current fastest Pentium. More information on serial performance is given at the annual Machine Evaluation Workshop and on the DisCo Web pages www.cse.clrc.ac.uk/Activity/DisCo.

A comparison of the interconnects on the two machines shows that MPI communications on the Fast Ethernet are effectively independent of the processor speed using the ping-pong benchmarks available from Pallas (www.pallas.com/pages/pmbd.htm). Some improvement is seen using the LAM implementation rather than MPICH but for high performance parallel applications the

proprietary QsNet solution has a huge advantage.

Real codes, too, have been benchmarked on these machines and results of these experiences are summarised in the following articles. For many scalar operations, or task farming, the choice is fairly clear-cut – chose a fast processor connected by Fast Ethernet to get the best price/ performance ratio. Parallel applications have more varied requirements and, where time-to-solution is less important or there is little inter-processor communication, Ethernet may still be the way to go, but for more demanding communications we start to see the onset of scalability issues. The influence of the interconnect becomes more apparent as the granularity of the code becomes finer.

It is becoming accepted that, for many applications requiring mid-range compute power, there is now an alternative to expensive SMP boxes. Several vendors are therefore beginning to roll out total Beowulf solutions, but the array of choices described above still remains. Many of these systems will be available for evaluation and comparison at the Machine Evaluation Workshop in November. Both the DisCo team and the HPCI Centre already have a great deal of experience in cluster computing and further information may be obtained from the authors via s.j.andrews@dl.ac.uk.

	Bandwidth (Mb/s, long message)	Latency (μ s, zero-length message)
Beowulf1: Ethernet, MPICH	10	135
Beowulf1: Ethernet, LAM	9	80
Loki: Ethernet, MPICH	11	140
Loki: QsNet, MPICH	200	5
Wulfgar: Ethernet, LAM	12	73
Wulfgar; Myrinet, MPICH	40	11

Table 1: performance of standard MPI communications on Beowulf systems.

GAMESS-UK

GAMESS-UK [1] represents a typical established electronic structure code for computational chemistry. It comprises some 700k lines of Fortran permitting a wide range of computational methodology to be applied to molecular systems. Other such codes include the NWChem [2], the GAUSSIAN series [4], CADPAC [5], GAMESS-US [6], MOLPRO [7], Turbomole [8], HONDO [9] etc. Present functionality lies at varying levels of approximation, including SCF, CASSCF, MCSCF, GVB, MP2/3/4, CCSD(T), MRDCI, Direct-CI, Full-CI, and analytic second

derivatives. The package is available on vector hardware (e.g., Cray C90 and J90, and Fujitsu VPP/300), on workstations from all the leading vendors, and on parallel hardware from Cray/SGI (Cray T3D and T3E, and SGI Origin series) and from IBM (the SP series). There are currently over 110 academic and industrial user sites.

Initial parallel SCF implementations of GAMESS-UK included those for the Intel iPSC-2 and iPSC/860 hypercubes, with more recent developments undertaken within the Esprit-funded EUROPORT 2 project, IMMPP (Interactive Modelling through Parallelism).

The principle aim of EUROPORT was to provide exemplars of commercial codes demonstrating the potential of parallel processing to industry; the primary goal was to establish a broad spectrum of standards-based medium-scale parallel application software rather than targeting the highest levels of scalability and performance.

In contrast to NWChem, both SCF and DFT modules are parallelised in a replicated data fashion, with each node maintaining a copy of all data structures present in the serial version. While this structure limits the treatment of molecular systems beyond a certain size, experience suggests that it is possible on machines with 256 MByte memory per node to handle systems of up to 2,000 basis functions. The main source of parallelism in the SCF module is the computation of the one- and two-electron integrals and their summation into the Fock matrix, with the more costly two-electron quantities allocated dynamically using a shared global counter. The result of parallelism implemented at this level is a code scalable to a modest number of processors (around 32), at which point the cost of other components of the SCF procedure starts to become relatively significant. The first of these addressed was

the diagonalisation, which is now based on the PeIGS eigen-solver developed for NWChem.

Once the capability of the Global Array Toolkit [3] is added, some distribution of the linear algebra becomes trivial. As an example, the SCF convergence acceleration algorithm (DIIS - direct inversion in the iterative subspace) is distributed using GA storage for all matrices, and parallel matrix multiply and dot-product functions. This not only reduces the time to perform the step, but the use of distributed memory storage (instead of disk) reduces the need for I/O during the SCF process.

DFT, MP2 and SCF 2nd Derivative Parallel Implementations

The performance of the DFT, MP2 and 2nd Derivative modules on the Cray T3E/1200E, IBM SP/WH2-375 and Pentium and Alpha Beowulf systems are shown in Table 2. The DFT calculations on morphine used a 6-31G** basis of 410 functions, those on cyclosporin a 6-31G basis of 1000 functions, both using the B3LYP hybrid functional. Note that the DFT calculations did not exploit CD fitting, but evaluated the coulomb matrix explicitly. Speedups of 95 and 108 are obtained on 128 Cray T3E nodes for the morphine and cyclosporin DFT calculation, respectively.

Machine	CPUs	Morphine DFT/B3LYP 6-31G**	Cyclosporin DFT/B3LYP 6-31G	(C ₆ H ₄ (CF ₃) ₂) ₂ 2 nd Derivatives 6-31G	Mn(CO) ₅ H MP2 gradient TZVP
Cray T3E/1200E	16	990	6927	2687	6713
	32	515	3612	1439	3530
	64	278	2003	803	1923
	128	160	1039	499	1158
	256		721		792
IBM SP/WH2-375	16	394	3135	1842	3531
	32	236	1779	1073	2158
Pentium Beowulf II	16	1183	9156	2977	11499
	32	719	5182	1809	8113
Alpha Beowulf III	16	312	2203	1450	2500
	32	196	1348	900	1550

Table 2: Total Elapsed times (seconds) using the GAMESS-UK DFT, SCF 2nd derivatives and MP2 gradient modules in calculations on Morphine, Cyclosporin, di(tri-fluoromethyl)-biphenyl and Mn(CO)₅H on the Cray T3E/1200, IBM SP/WH2-375 and Beowulf Systems.

Considering the total times to solution on 32 CPUs, we see that the Pentium Beowulf II is delivering 72% (morphine) and 62% (cyclosporin) of the Cray T3E/1200E in the DFT B3LYP calculations. As expected, the more powerful CPUs of the IBM SP and Alpha Cluster lead to much high percentage delivery. Thus the 32-CPU IBM/SP-WH2-375 delivers 218% (morphine) and 203% (cyclosporin) of

the Cray T3E, while the Linux Alpha Cluster outperforms the IBM SP, with corresponding figures of 263% (morphine) and 268% (cyclosporin). In both benchmarks we find the elapsed times on the 32-CPU Alpha Beowulf fall between the 64- and 128-node T3E timings, lying somewhat closer to the 128-node T3E time in each case.

Substantial modifications were required to enable the MP2 gradient and SCF 2nd derivatives to be computed in parallel. In both cases the conventional integral transformation step has been omitted, with the SCF step performed in direct fashion and the MO integrals, generated by re-computation of the AO integrals, and stored in the global memory of the parallel machine. This storage, and subsequent access, is managed by the GA tools. The basic principle by which the subsequent steps are parallelized involves each node computing a contribution to the current term from MO integrals resident on that node. For some steps, however, more substantial changes to the algorithms are required. For the MP2 gradient, the construction of the Lagrangian (the right-hand side of the coupled Hartree-Fock (CPHF) equations) requires MO integrals with three virtual orbital indices. Given the size of this class of integrals, they are not stored, the required terms of the Lagrangian being constructed directly from AO integrals. A second departure from the serial algorithm concerns the MP2 2-particle density matrix. This quantity, which is required in the AO basis, is of a similar size to the 2-electron integrals and is stored on disk in the conventional algorithm, but generated as required during the 2-electron derivative integral generation from intermediates stored in the GAs. In the SCF 2nd derivative module the coupled Hartree-Fock (CPHF) step and construction of perturbed fock matrices are again parallelised according to the distribution of the MO integrals. The most costly step in the serial 2nd derivative algorithm is the computation of the 2nd derivative two-electron integrals. This step is trivially parallelised through a similar approach to that adopted in the direct SCF scheme - using dynamic load-balancing based on a shared global counter. In contrast to the serial code, the construction of the perturbed Fock matrices dominates the parallel computation. It seems almost certain that these matrices would be more efficiently computed in the AO basis, rather than from the MO integrals as in the current implementation, thus enabling more effective use of sparsity when dealing with systems comprising more than 25 atoms.

Performance data for both modules are shown in Table 2. The MP2 geometry optimisation of the Mn(CO)₅H molecule (with 217 basis functions) shows a speedup of 93 achieved using 128 T3E/1200 processors to perform the complete optimisation (involving 5 energy and 5 gradient calculations). A corresponding speedup of 86 is found when calculating the frequencies of 2,2'-di(tri-fluoromethyl)-biphenyl

using a 6-31G basis of 196 functions. The greater dependence on the global arrays (GAs) in both MP2 gradient and analytic 2nd derivative applications produces the expected impact in performance of the Beowulf clusters. Considering the total times to solution on 32 CPUs, we see that the Pentium Beowulf II is delivering a much reduced percentage of the Cray T3E (44%) in the MP2 gradient calculation, with only a modest reduction in elapsed time between 16 (11,499 seconds) and 32 CPUs (8,113 seconds). A similar degradation in performance was originally noted on the Alpha Beowulf, caused not by limited communications but by problems in the effective utilisation of shared memory on the dual CPUs of the UP2000. Revisions in release 3.1 of the GAs have largely addressed this, with the 32-CPU Alpha now delivering 228% of 32-node Cray performance and outperforming the IBM/SP-WH2-375 (1550 vs. 2158 seconds).

Somewhat surprisingly the Pentium Beowulf II performs far more effectively in the SCF 2nd derivative benchmark, achieving a much higher percentage of Cray performance (81% of the 32-node Cray T3E/1200E). An initial performance analysis reveals load balancing problems in the Fock matrix construction which may explain this effect. Neither IBM/SP nor the Alpha Cluster perform that effectively on this benchmark; while the revised GAs have improved the Alpha performance, the 32-CPU Linux Cluster delivers only 160% of T3E performance, one of the lowest such figures recorded in these benchmarks.

Developments to the Fitted Coulomb Module

More recent work has focused on optimising and extending the fitted Coulomb module of the CCP1 density functional theory (DFT) code within GAMESS-UK for use on MPP machines. This will be reported in a separate article.

DL_POLY

DL_POLY [10] is the parallel molecular dynamics simulation package developed at Daresbury Laboratory by W. Smith and T.R. Forester for CCP5 (the Collaborative Computational Project for the Computer Simulation of Condensed Phases). The parallel implementation was initially based on a replicated data (RD) strategy, and was designed at the outset for computers with up to 64 processors (e.g. the Intel iPSC/860) and systems of up to 30,000 atoms, although it has since found use on much larger machines. Implicit in the RD approach is a dependence

on global summations being fast, which is not always the case. The performance scaling varies according to the kind of simulation being undertaken - systems possessing complex molecular topologies and constraint bonds typically scale less well than ones requiring simple atomic descriptions, as they require a higher communication overhead. If constraint bonds are present, as they usually are in bio-molecular or polymer systems, then significant deviations from ideal behaviour are to be expected. The four benchmarks described below are those described on the CCP5 Web site [11], with the same numbering scheme adopted here:

Benchmark 4: A straightforward simulation of sodium chloride at 500k, using the standard Ewald summation method to handle the electrostatic forces. A multiple time-step algorithm is used to increase performance, which requires recalculating the reciprocal space forces only twice in every five time steps. The electrostatic cutoff is set at 24 Å in real space, with a primary cutoff of 12 Å for the multiple time-step algorithm. The Van der Waals terms are calculated with a cutoff of 12 Å. The simulation is for 200 steps with a time step of 1 fs in the Berendsen NVT ensemble. The system size is 27,000 ions.

Benchmark 5: This simulation is of 8,640 atoms of an alkali disilicate glass at 1000k. The electrostatics are again handled by the Ewald sum, with the interaction potential including a three-body valence angle term, which requires a link-cell scheme to locate atom triplets. The electrostatic cutoff is 12 Å and the Van der Waals cutoff is 7.6 Å; 3-body forces are cut off at 3.45 Å. The simulation is for 300 steps in the Hoover NVT ensemble, with a time step of 1 fs.

Benchmark 3: This simulation is of the enzyme transferrin in a solution comprised of 8102 TIP3P water molecules. A total of 27,593 atoms are in the system. The electrostatic

forces are handled by a combination of neutral groups with the Coulombic potential. All force cutoffs are set at 8 Å. The simulation is for 250 steps with a time step of 0.1 fs, in the NVE ensemble. The water molecules are treated as rigid bodies and the transferrin is maintained by bond constraints using SHAKE. Valence angles and dihedral potentials are present in the transferrin model.

Benchmark 7: This system is comprised of 13,390 atoms, including 4012 TIP3P water molecules solvating the gramicidin-A protein molecule at 300k. Both the protein and water molecules are defined with rigid bonds and maintained by the SHAKE algorithm. The water is held completely rigid, while the protein has angular and dihedral potential terms. Electrostatic interactions are handled by the neutral group method with a Coulombic potential truncated at 12 Å. The Van der Waals interactions are truncated at 8 Å. The simulation is for 500 time steps in the NVE ensemble with a 1 fs time step.

Performance scaling on the Cray T3E for Benchmarks 4 and 5 is almost linear over the entire range of processor numbers. This reflects the high parallel efficiency of the Ewald sum implementation and good communication subsystem of the T3E. Significantly inferior scaling is found in the two macromolecular benchmarks, 3 and 7. This may be attributed to the difficulty in apportioning the neutral group calculations across processors, and the use of SHAKE for the bond constraints. Somewhat better scaling is found in (iv) which uses a larger cutoff in the electrostatic calculations and hence a lower communication/computation ratio. ???

The performance of the four DL_POLY benchmarks on the Cray T3E-1200E, IBM SP/WH2-375 and Pentium, AMD/Myrinet and Alpha Beowulf systems are shown in Table 3.

Benchmark	Nodes	Cray T3E / 1200E	IBM SP/WH2- 375	Pentium Beowulf II (LAM/MPI)	Alpha Beowulf III (QsNet)	AMD Beowulf IV (Myrinet)
4 NaCl	4	3176	860	2621	689	1760
	8	1588	446	1355	326	778
	16	817	239	670	158	351
	32	376	124	352	84	
	64	179		-		
	128	94				
5 NaK	4	1927	685	1527	503	869
	8	918	353	768	214	401
	16	479	184	412	106	188

disilicate glass	32	225	106	238	64	
	64	121		-		
	128	75				
3	4		338	652	166	
	8		202	483	109	196
	16	191	157	410	78	163
	32	132	130	391	92	
	64	115				
	128	104		-		
7	4	2400	1613	2992	641	1347
	8	1243	857	1671	351	772
	16	688	483	1004	193	442
	32	382	288	681	147	
	64	232				
	128	166		-		

Table 3: Time in wall clock seconds for the four DL_POLY benchmark calculations on the Cray T3E/1200, IBM SP/WH2-375 and Pentium, AMD Athlon and Alpha Beowulf Systems.

Modifications made to DL_POLY to improve performance included replacing the MPI_ALLREDUCE routines from both LAM and MPICH libraries with a hypercube-based version written by the Daresbury HPCI Centre.

Considering the Ewald-based benchmarks, we again find excellent scalability on the Cray T3E/1200E, with speedups of 135 (super-linear) and 103 obtained on 128 nodes Cray for Benchmark 4 and 5 respectively. This excellent scaling on the T3E is put into perspective when considering the total times to solution on 32 CPUs. The weakness of the Cray EV56 CPU is clearly apparent, for the Pentium Beowulf II is outperforming the Cray T3E/1200E in the NaCl simulation (352 vs. 376 seconds), and is only marginally slower than the T3E (238 vs. 225 seconds) in the NaK silicate simulation. These percentage delivery figures of 107% and 95% on the Pentium increase substantially on the more powerful CPUs of the IBM SP and AMD Athlon and Alpha Clusters. Thus the 32-CPU IBM/SP-WH2-375 delivers 303% (Benchmark 4) and 213% (Benchmark 5) of the Cray T3E, while the Linux Alpha Cluster again outperforms the IBM SP, with corresponding figures of 447% (Benchmark 4) and 352% (Benchmark 5). Corresponding 16-node figures for the AMD Athlon Beowulf IV are 233% (Benchmark 4) and 255% (Benchmark 5). The potential of the Beowulf systems in these simulations is striking; the 32-CPU Linux Alpha Cluster is outperforming 128 nodes of the Cray T3E in both benchmarks.

A quite different picture of performance is revealed when considering the two macromolecular simulations, Benchmarks 3 and 7. Now the scalability on the T3E/1200E is far more limited, with speedups of just 27

(Benchmark 3) and 61 (Benchmark 7) on 128 nodes of the Cray. This lack of scalability has a predictable effect on the performance of the Beowulf clusters, which now deliver significantly lower percentage delivery figures compared to those found on the Ewald-based simulations. Considering the Pentium Beowulf, we find 32-node delivery figures of just 34% and 56% for Benchmarks 3 and 7 respectively. While these figures increase significantly on the more powerful CPUs, they are far from impressive; the 32-CPU Linux Alpha Cluster delivers 143% (Benchmark 3) and 260% (Benchmark 7) of the Cray T3E, while the IBM SP/WH2-375 is little faster than the Cray on Benchmark 3. A significant performance limitation on the Alpha Cluster arose from the way DL_POLY handled both co-ordinate and forces arrays. The x-, y- and z-coordinates and corresponding forces were stored as separate linear arrays, $\underline{x}(m\text{xatms})$, $\underline{y}(m\text{xatms})$ etc., coding that led to exceedingly poor cache re-use on the UP2000 processor. Re-writing the code to use, in hopefully obvious notation, $\underline{xyz}(3,m\text{xatms})$ and $\underline{fxyz}(3,m\text{xatms})$ improved overall performance on the Alpha Beowulf III by a factor of 2.5 (although it had little effect on, for example, the IBM/SP-WH2 with its larger 8 MByte cache). Having made these changes, the 32-CPU Linux Alpha Cluster again outperforms 128 nodes of the Cray T3E in both benchmarks.

The performance of the AMD Athlon Beowulf IV is worth noting; the 16-CPU cluster performs largely on a par with 16 CPUs of the IBM SP/WH2-375, and delivers about 50% of the corresponding partition of the Linux Alpha Cluster. An additional feature exemplified by these benchmarks is the impact of the underlying MPI libraries on performance. While little effect was found in the Ewald-based

simulations, a much greater impact was apparent on Benchmarks 3 and 7. Thus the reduced latency associated with LAM MPI as against MPICH reduced the 32-node Benchmark 3 timing on the Pentium Beowulf II from 583 (MPICH) to 391 seconds (LAM).

Bibliography

[1] GAMESS-UK is a package of *ab initio* programs written by M.F. Guest, J.H. van Lenthe, J. Kendrick, K. Schoeffel and P. Sherwood, with contributions from R.D. Amos, R.J. Buenker, M. Dupuis, N.C. Handy, I.H. Hillier, P.J. Knowles, V. Bonacic-Koutecky, W. von Niessen, R.J. Harrison, A.P. Rendell, V.R. Saunders, and A.J. Stone. The package is derived from the original GAMESS code due to M. Dupuis, D. Spangler and J. Wendoloski, NRCC Software Catalog, Vol. 1, Program No. QG01 (GAMESS), 1980.

[2] See the home Web page of the Environmental Molecular Sciences Laboratory, Pacific Northwest National Laboratory, Battelle Memorial Institute, and the U.S. Department of Energy is available at <http://www.pnl.gov:2080/>

[3] J. Nieplocha, R.J. Harrison and R.J. Littlefield, Global arrays; A portable shared memory programming model for distributed memory computers, in: Supercomputing '94, IEEE Computer Society Press, Washington, D.C. (1994).

[4] M. Frisch et al., Gaussian, Inc., 4415 Fifth Avenue, Pittsburgh, PA 15213, USA (1992).

[5] R.D. Amos, I.L. Alberts, J.S. Andrews, S.M. Colwell, N.C. Handy, D. Jayatilaka, P.J. Knowles, R. Kobayashi, N. Koga, K.E. Laidig, P.E. Maslen, C.W. Murray, J.E. Rice, J. Sanz, E.D. Simandiras, A.J. Stone and M.-D. Su, CADPAC, Issue 6, University of Cambridge, (1995).

[6] M.W. Schmidt, et al., QCPE Bulletin 7:115 (1987).

[7] MOLPRO is a package of *ab initio* programs written by H.-J. Werner and P.J. Knowles, with contributions from R.D. Amos, A. Berning, D.L. Cooper, M.J.O. Deegan, A.J. Dobbyn, F. Eckert, C. Hampel, T. Leininger, R. Lindh, A.W. Lloyd, W. Meyer, M.E. Mura, A. Nicklass, P. Palmieri, K. Peterson, R. Pitzer, P. Pulay, G. Rauhut, M. Schütz, H. Stoll, A.J. Stone and T. Thorsteinsson.

[8] R. Ahlrichs, M. Br, M. Hser, H. Horn and C. Klöml, Electronic structure calculations on workstation computers: The program system TURBOMOLE, *Chem. Phys. Letters* 162:165 (1989); R. Ahlrichs and M.v. Arnim, TURBOMOLE, parallel implementation of SCF, density functional, and chemical shift modules, in: *Methods and Techniques in Computational Chemistry: METECC-95*, E. Clementi and G. Corongiu, eds.

[9] M. Dupuis, J.D. Watts, H.O. Villar and G.J.B. Hurst, The general atomic and molecular

electronic structure system HONDO: version 7.0., *Comput. Phys. Commun.* 52:415 (1989); M. Dupuis, A. Farazdel, S.P. Karma and S.A. Maluendes, HONDO: A general atomic and molecular electronic structure system, in: *MOTECC*, E. Clementi ed., ESCOM, Leiden (1990).

[10] see, http://www.dl.ac.uk/TCSC/Software/DL_POLY/main.htm

[11] see, http://www.dl.ac.uk/TCS/Software/DL_POLY/dl_poly.t3e.htm

[12] B.R. Brooks, R.E. Bruccoleri, B.D. Olafson, D.J. States, S. Swaminathan and M. Karplus, *J. Comp. Chem.*, 4(2), 1983, 187-217.