Exploring Extreme Scalability in Scientific Applications

Mike Ashworth, Ian Bush, Charles Moulinec and Ilian Todorov, *STFC Daresbury Laboratory, Daresbury Science and Innovation Campus, Warrington WA4 4AD, UK*

ABSTRACT: *Many areas of scientific research are underpinned by computational methods that require ever increasing levels of computer performance. In order to meet this demand high-performance systems are rapidly heading towards Petascale performance levels with planned systems typically consisting of O(100,000) processors. We are investigating whether current applications used in the UK are capable of scaling to these levels. We present performance results for five applications (SBLI, Code_Saturne, POLCOMS, DL_POLY_3 and CRYSTAL) from a range of scientific areas on Cray XT, IBM POWER5 and IBM BlueGene systems up to 16,384 processors. Most codes scale well with sufficiently large problem sizes, though we have identified a requirement for further research in efficient parallel I/O, parallel partitioning for unstructured mesh codes and diagonalisation-less methods for quantum chemistry*

KEYWORDS: high-performance applications, parallel computing, petascale, scalability

1. Introduction

The Science and Technology Facilities Council's (STFC) Computational Science & Engineering Department (CSED) led from Daresbury Laboratory is a UK focus for computational science and engineering, ensuring that UK researchers benefit from the best computational methods, and supporting them through research and collaboration, theory and software development, facilities and training. CSED has a long track-record of collaboration with UK academic consortia, especially through the Collaborative Computational Projects (CCPs) funded by the Engineering and Physical Sciences Research Council and through its involvement in computational science support for the UK national High-Performance Computing (HPC) services. STFC, as a member of UoE HPCx Ltd, works with Edinburgh Parallel Computing Centre (EPCC) to provide service support for both National HPC Services in the UK, HECToR and HPCx (see below). CSED leads the in-depth Terascaling support for HPCx.

The UK has a strategy for overlapping National Service provision with 6-year service contracts including technology upgrades at approximately 2-year intervals (for historical details see [1]). A recent strategy document [2] states

> *"… the UK should aim to achieve sustained Petascale performance as early as possible across a broad field of scientific applications, permitting the UK to remain internationally competitive in an increasingly broad set of high-end computing grand challenge problems."*

Current systems worldwide are heading rapidly towards Petascale performance. The number one system in the TOP500 list of supercomputer sites[1] is currently (May 2008) the 212,992-processor IBM BlueGene/L at Lawrence Livermore National Laboratory with a Linpack performance of 0.478 Pflop/s. Trends clearly indicate that Petascale performance will be reached within 12 months by systems of order 100,000 processors utilizing multi-core components. There are still some existing and proposed systems that use a smaller number of much more powerful (around 100 Gflop/s) vector processors [3], but these systems represent a small fraction of the global installed base.

[1] http://www.top500.org/

There are significant challenges at the Petascale. There are scientific challenges: what new science can you do with 1000 Tflop/s, and how can you exploit such systems to simulate larger problems, and to address new regimes of multi-scale and multi-disciplinary science? There are also technical challenges. How will existing codes scale to 100,000 processors? This must take into account not only scaling of time with number of processors, but also of time with problem size, and of memory with problem size. We need to scale simulations to larger problems to exploit Petascale systems, but this will fail if scaling the problem size causes codes to exceed available memory limits. This is becoming exaggerated as the cost of memory begins to exceed that of the processors leading to reduced memory/processor.

There are many other issues that will be very important at the Petascale, such as exploitation of the memory hierarchy, data management (including pre- and post-processing), visualisation and fault tolerance. In this paper we look at the scalability issue alone. We are interested at how current application codes of major importance to UK scientists will scale to the large numbers of processors we expect to see in Petascale systems. Will current algorithms scale? Are they amenable to optimisation? Or will we have to develop completely new algorithms at this level of parallelism?

At CSED we have embarked on a project to evaluate and analyse the performance of a small number of application codes on currently available systems with O(10,000) processors with a view to assessing their suitability for O(100,000) processors. We shall look at the possibilities for optimisation of current algorithms and for the development of new algorithms. This paper presents the initial results from this study.

Current codes all use the standard programming model of a serial language (Fortran) combined with the Message Passing Interface (MPI). We are also interested in whether scalability is enhanced by using alternative programming models (e.g. hybrid MPI/OpenMP or global address space languages). This topic is outside the scope of this paper, although work on porting one of the codes to Co-Array Fortran is reported elsewhere at this conference [4]. A detailed comparison of the performance of a range of application codes on the HECToR and HPCx systems is also presented elsewhere [5].

3. Systems

3.1 Cray XT series

The Cray XT massively parallel computers combine commodity and open source components with custom-designed components. The architecture is based on the Red Storm technology that was developed jointly by Cray Inc. and the U.S. Department of Energy's Sandia National Laboratories. The XT system is based around AMD Opteron 64-bit processors, with each processor being directly connected via the chip's HyperTransport to a dedicated Cray SeaStar chip (based on the IBM POWER architecture). Each SeaStar contains a 6-Port router and communications engine.

We have run on both the Cray XT3 system at the Swiss National Supercomputing Centre (CSCS[2]) and the newer Cray XT4 HECToR[3] system in the UK. The Swiss XT3 runs the older Catamount operating system and has 1664 2.6 GHz dual-core Opterons with 2 GB memory per node, whereas the HECToR system runs the Cray Linux Environment (formerly known as Compute Node Linux) and has 5664 2.8 GHz dual-core parts with 6 GB memory per node. The 6 GB per node is implemented as two memory banks of 4 GB + 2 GB and performance results reported elsewhere [5] have revealed a measurable performance degradation for memory intensive codes due to the asymmetric nature of this configuration. Both Catamount and CNL are light-weight single-task operating systems, minimizing OS jitter effects, providing contiguous physical allocations and no demand paging. This places some limitations on the API available to applications. Codes were run with the PGI Fortran compiler using "-O3 –fastsse" compiler options.

3.2 IBM POWER5

HPCx[4] has been the UK's leading national High Performance Computing service from its beginning in November 2002 until the start of the HECToR service in October 2007. The current Phase3 system, installed towards the end of 2006, consists of 160 IBM p5-575 nodes to give a total of 2560 1.5 GHz POWER5 processors. The p5-575 is a 16-way shared memory system with a three-level cache architecture. There are two POWER5 processors per chip each with its own Level 1 data and instruction caches and with a shared on-chip Level 2 cache. Each 8-way Multi-Chip Module in the node shares 128 MB of Level 3 cache and 16GB of main memory. Communication between nodes is provided by IBM's High performance Switch (HPS), formerly known as "Federation". Each node runs a single copy of IBM's AIX operating system.

3.3 IBM BlueGene/L

IBM's BlueGene systems offer a low-power, small-footprint high-performance solution. To achieve this, processor speeds are much lower than other high-end offerings and achieving high performance levels relies on scaling to very large numbers of processors. A node of BlueGene/L consists of a dual-core 700 MHz POWERPC 440 processor with 1 GB memory. The POWERPC design is augmented by an attached SIMD floating point unit, which is capable of handling two discrete sets of operands with a single floating point operation.

[2] http://www.cscs.ch/
[3] http://www.hector.ac.uk/
[4] http://www.hpcx.ac.uk/

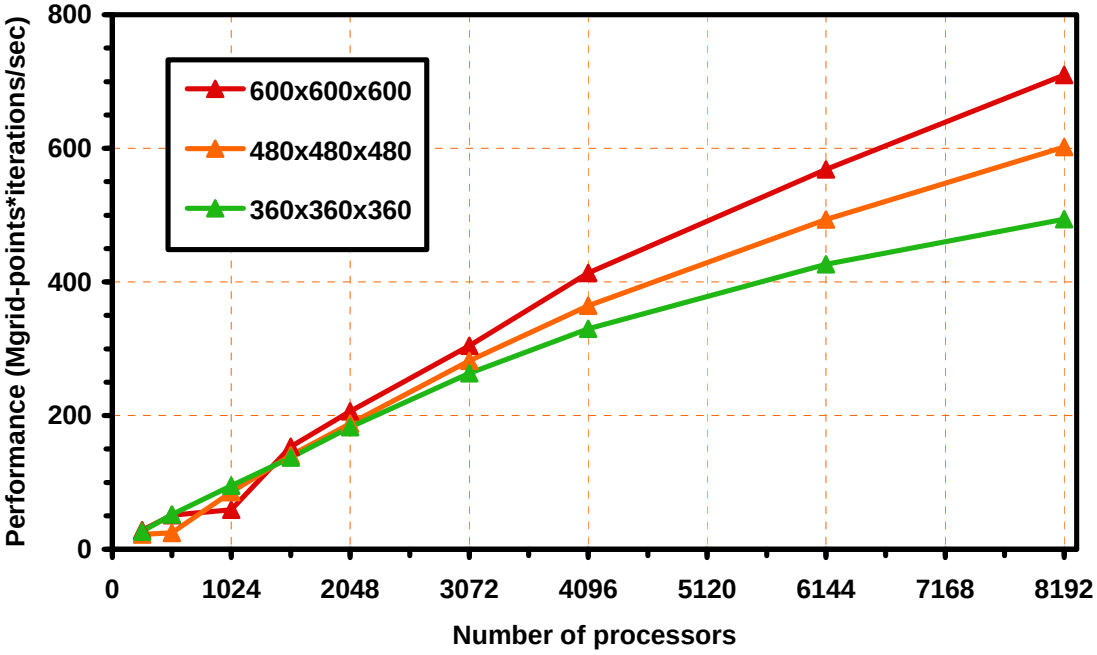

Figure 1: Performance of the SBLI code on the HECToR Cray XT4 for three different problem sizes.

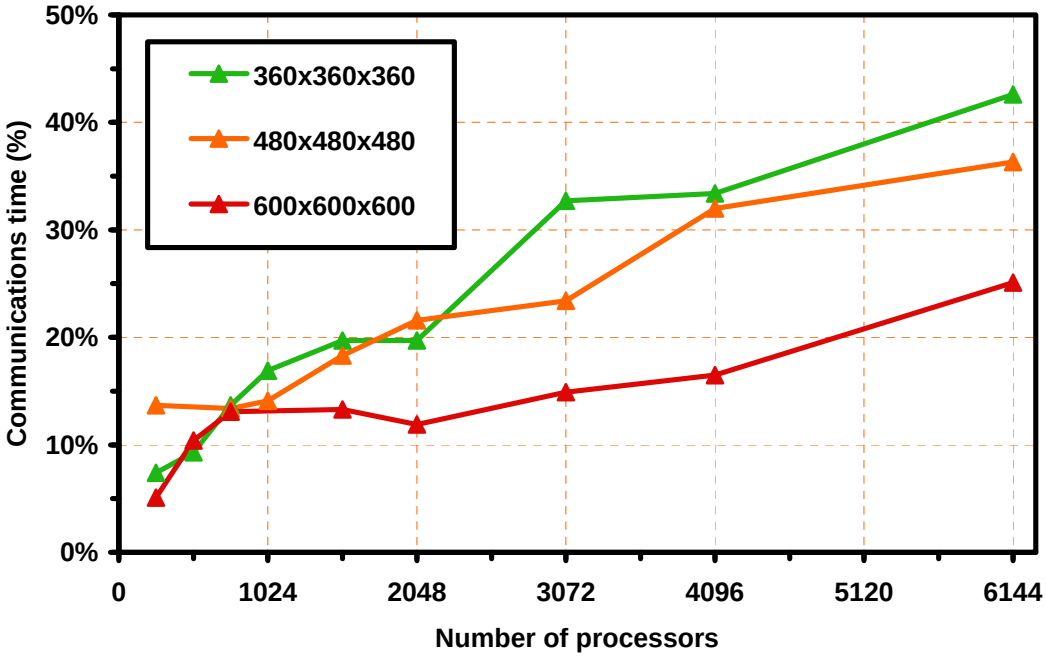

Figure 2: Percentage communications time reported by craypat on the HECToR Cray XT4

The result is that, like the POWER5, four floating point operations per clock are possible, whereas the dual-core Opterons are limited to two flops per clock (this is increased to four in the latest quad-core parts). Three separate networks connect all the compute chips together: a three-dimensional torus, a global collective network and the control system network. The MPI implementation chooses the best network for each MPI invocation; the network used for the majority of the MPI traffic will be the three-dimensional torus.

We carried out tests with DL_POLY_3 on the 16,384 processor IBM BlueGene/L at FZ Jülich as part of the Blue Gene Scaling Workshop, 5th-8th Dec 2006

4. Performance of Application Codes

4.1 Introduction

Figures show a performance metric which is proportional to the inverse of the execution time. By suitable choice of the constant this can be a scientifically meaningful quantity such as model days per day or iterations per second. Ideal scaling is linear allowing deviations from perfect scaling to be easily seen. The abscissa is the number of processors, where we use the term *processor* as a short form of Central Processing Unit, i.e. a processing unit that runs a single instruction stream, and that runs a single MPI task or a team of threads.

4.2 SBLI

The SBLI code [6] (also known as PDNS3D or the PCHAN benchmark) is a sophisticated DNS code that incorporates a number of advanced features: namely high-order central differencing; a shock-preserving advection scheme from the total variation diminishing (TVD) family; entropy splitting of the Euler terms and the stable boundary scheme. The code has been written using standard Fortran 90 code together with MPI in order to be efficient, scalable and portable across a wide range of high-performance platforms.

The benchmark is a simple turbulent channel flow run for 100 iterations using a grid size which can be varied from 360^3 through 480^3 to a largest size of 600^3. The most important communications structure is a three-dimensional halo-exchange between adjacent computational sub-domains. For a fixed problem size, as the number of processors increases and the data size per processor decreases, the surface area to volume ratio for each sub-domain grows and communications costs start to dominate. This is evident in Figure 1 which shows performance for the three problem sizes on the HECToR Cray XT4 out to 8192 processors, and confirmed in Figure 2 which shows the percentage communication cost as reported by CrayPat [7], Cray's profiling tool. Clearly, larger problem sizes scale better. Another effect of the diminishing problem size per processor is that key data structures arising from the finite difference stencil fit into cache sooner (at smaller processor numbers) for the smaller problem sizes. This is evident in Figure 1 at 512 and 1024 processors, but by 1536 processors we surmise that the cache behaviour is the same for all three grids.

4.3 Code_Saturne

The basic capabilities of Code_Saturne [8] enable the handling of either incompressible or expandable flows with or without heat transfer and turbulence. Dedicated modules are available for specific physics such as radiative heat transfer, combustion, magneto-hydrodynamics, compressible flows, and two-phase flows. We have run a 3-D Large-Eddy Simulation (LES) of a channel flow to evaluate the performance of Code_Saturne. The mesh is structured whereas the code is written so that it can handle fully unstructured grids. For that reason, an advanced partitioner is required. Code_Saturne presently uses Metis [9] or Scotch [10], both of them serial partitioners. This means that this part of the pre-processing has to be performed on the front-end of the supercomputer and is limited by the memory available to a single processor. Two sizes of mesh are considered, 78M and 120M cells. Figure 3 shows the performance scaled by the number of cells. The 78M case scales well up to 4096 processors and the 120M one shows an almost linear behaviour up to 8192.

Considering the performance as a function of the calculation effort (equation resolution), the larger the original mesh size, the better the load-balancing. We expect the code to show good scaling up to 100,000 cores for a very large problem of, say, 1 billion cells. The key issue with such large meshes is the quality of the partitioning that can be achieved by current partitioners (even the serial ones) for a partition of 100,000 or 1M tasks. The memory requirement for serial partitioners will be prohibitive with the size of meshes we expect to handle, so a switch to a parallel partitioner will be compulsory. It is not yet clear how much this is going to impact on the quality of the mesh partitioning. This is the subject of current research at CSED.

The results presented are for test runs not including Input/Output (I/O). If I/O is included poor scaling is achieved, mainly because of the serial nature of the current implementation. We are looking at implementing parallel I/O (e.g. MPI-IO) in order to improve the scaling. Depending on the desired output, it may be possible to limit the size of data being handled by I/O and post-processing. For instance, generating frames for a film can be carried out only with 1/8, 1/64 or even 1/512 of the grid cells, instead of all of them. This part of the post-processing could then be achieved by cell-collapsing. The checkpoint/restart procedure also currently requires dumping/loading all the data for every cell. We are investigating whether a similar procedure can be used here also.

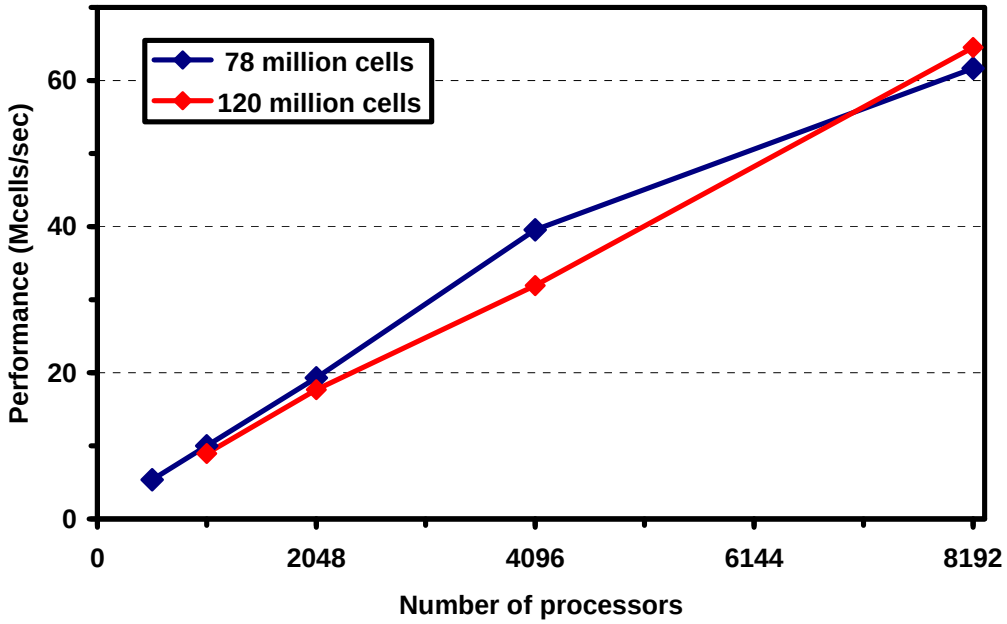

Figure 3. Performance in Mcells/sec of Code_Saturne on the HECToR Cray XT4 for two grid sizes

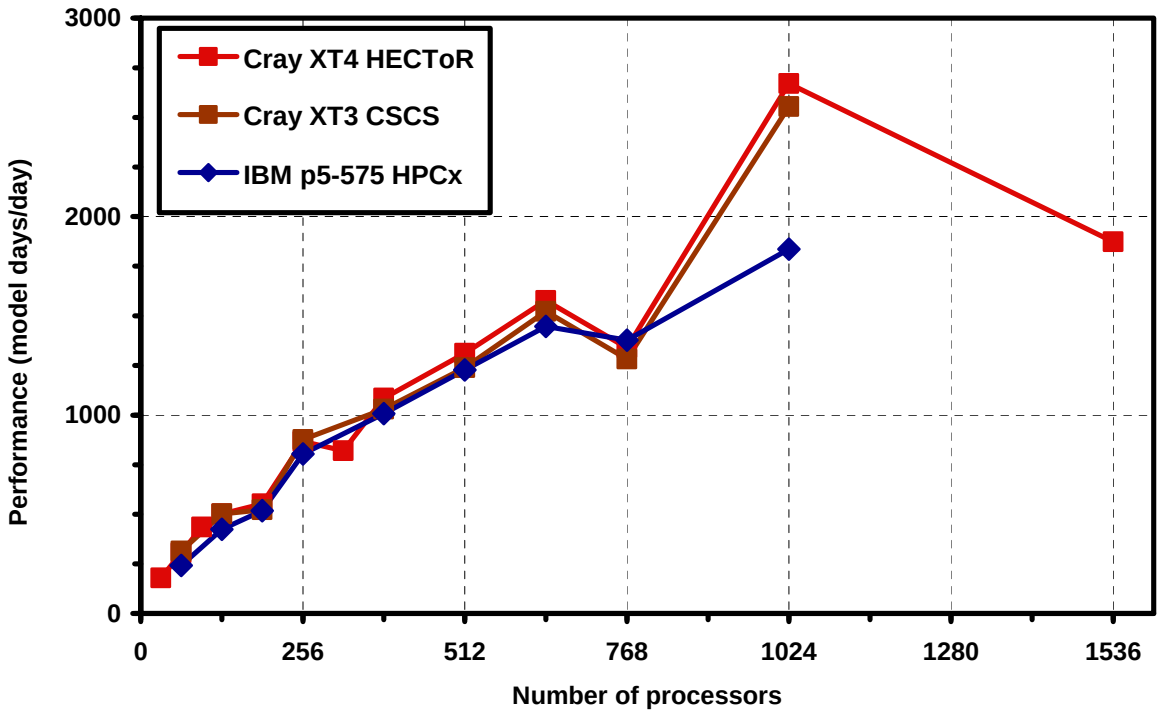

Figure 4. Performance in model days per day of the POLCOMS HRCS model (physics only) on the HECToR Cray XT4, the CSCS Cray XT3 and the HPCx IBM p5-575 systems

4.4 POLCOMS

The Proudman Oceanographic Laboratory Coastal Ocean Modeling System (POLCOMS) has been developed to tackle multi-disciplinary studies in coastal/shelf environments and optimized to make use of high-performance parallel computers [11]. In order to improve simulations of marine processes, we need accurate representation of eddies, fronts and other regions of steep gradients. The current generation of models includes simulations which cover large regions of the continental shelf at approximately 1km resolution.

The central core is a sophisticated 4-dimensional hydrodynamic model that provides realistic physical forcing to interact with, and transport, environmental parameters. The hydrodynamic model is a finite difference model based on a latitude-longitude Arakawa B-grid in the horizontal and S-coordinates in the vertical. Conservative monotonic advection routines using the Piecewise Parabolic Method are used to ensure strong frontal gradients. Vertical mixing is through turbulence closure (Mellor-Yamada level 2.5).

The benchmark case is the High-Resolution Continental Shelf (HRCS) model. This is a large model with a grid of size 1001 x 801 x 34 points representing a resolution of 1/40 degree x 1/60 degree covering the north-west European shelf seas. This is a research model used for investigations of the circulation of the north-west European continental shelf.

Performance in model days per day is shown in Figure 4 for the CSCS Cray XT3, the HECToR Cray XT4 and the HPCx IBM p5-575. The parallel implementation uses a recursive grid partitioning scheme to achieve a load balance for an arbitrary sea area in which the partitioning allocates each processor an approximately equal number of sea points. Performance is dependent on the details of the partitioning and this results in fluctuations in the scaling, which ultimately limit the scalability for a given problem size. We are currently investigating a residual dependence on the number of land points which is believed to be responsible for poor performance at some combinations of problem size and number of processors.

4.5 DL_POLY_3

DL_POLY_3 [12] is a general-purpose molecular dynamics (MD) simulation package designed to address a wide range of possible scientific applications such as ionic solids, solutions, metals, zeolites, surfaces and interfaces, complex systems (e.g. liquid crystals), minerals, bio-systems, and spectroscopy. DL_POLY_3 has been specifically engineered to harness the power of high-end parallel systems [13].

The overall scaling of DL_POLY_3 has not changed much for the last two years, however there has been an overall speed-up (up to 35% depending on functionality) and decrease in the memory usage (~15%). We find that excellent scaling is achieved when a processor holds data from around 1000 particles or greater.

Figure 5 shows the performance of DL_POLY_3 for a 14.6M particle $Gd_2Zr_2O_7$ system on the HECToR Cray XT4 and the Jülich IBM BlueGene/L. In Figure 6 we show for the Jülich IBM BlueGene/L the speed-up (scaled from 2048 processors) of various components which go to make up the total time for the MD step. Note that as this is speed-up the graph does not reflect the fact that not all components contribute equally to the total time. Most components show linear scaling. It is only the long range forces, shown as the k-space Ewald calculations, which limit scaling at very large processor counts. For such large systems it is feasible to use force-shifted Coulombic electrostatics which is not long-ranged. When the systems are even larger (100M atoms) we can use fast multipole electrostatics which is beneficial for the scaling (even after the pre-factors penalty).

However, as with Code_Saturne, these performance data neglect I/O which turns out to be the real bottleneck for the simulations. Every processor holds partial data of the evolution of the system at certain point in time that needs to be saved on disk incrementally for further analysis. This is the subject of current investigations reported elsewhere [14]. There are additional challenges for large-scale MD simulations. Scientific progress is limited by the lack of tools to handle the datasets; to analyse output datasets and in some cases to construct problems (e.g. trans-membrane proteins with signalling components, unnatural coarse-grained interfaces).

4.6 CRYSTAL

The CRYSTAL program [15] was jointly developed by the Theoretical Chemistry Group at the University of Torino and the Computational Materials Science Group in CSED. The program computes the electronic structure of periodic materials within Hartree Fock, density functional or various hybrid approximations. The Bloch functions of the periodic systems are expanded as linear combinations of atom centred Gaussian functions. Powerful screening techniques are used to exploit real space locality. The code may be used to perform consistent studies of the physical, electronic and magnetic structure of molecules, polymers, surfaces and crystalline solids.

Figure 7 shows the performance for a 1737 atom, 23268 basis function self consistent field (SCF) calculation in CRYSTAL on the HECToR Cray XT4 and the HPCx IBM p5-575 systems. Scaling for such a large system is relatively good with a parallel efficiency of 60% at 4096 processors, though there is a definite tailing off at large processor counts. This behaviour is due to the different scaling of the two major components of the code:

1) The building of the Kohn-Sham matrix
2) The dense linear algebra (which is dominated by diagonalisation)

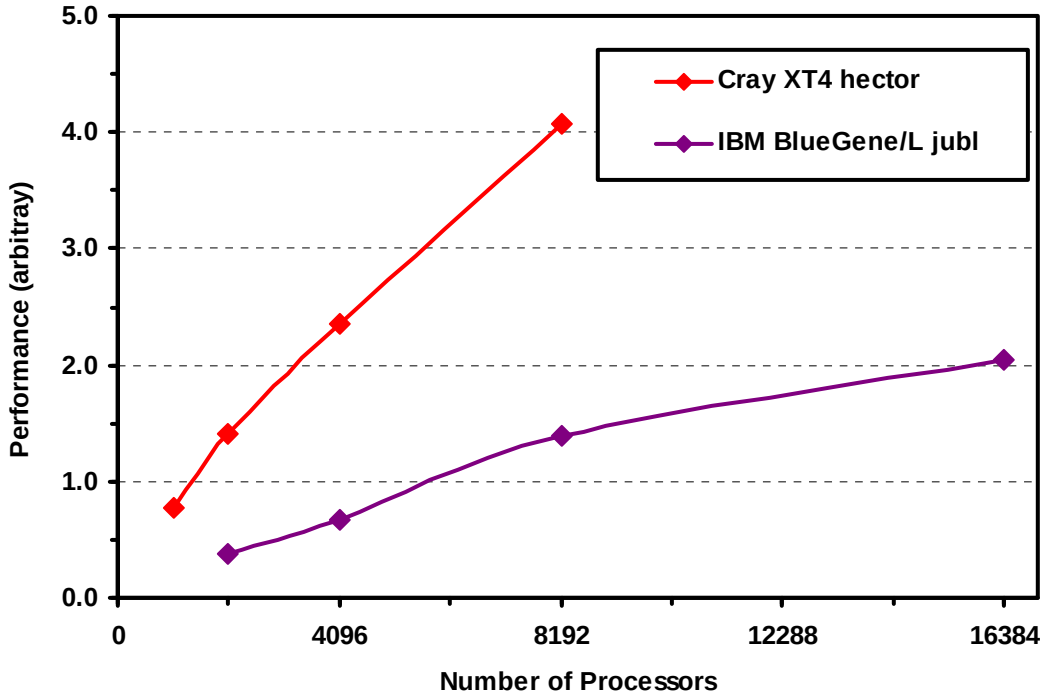

Figure 5. Performance of DL_POLY_3 for a 14.6 million particle system on the HECToR Cray XT4 and the Jülich IBM BlueGene/L

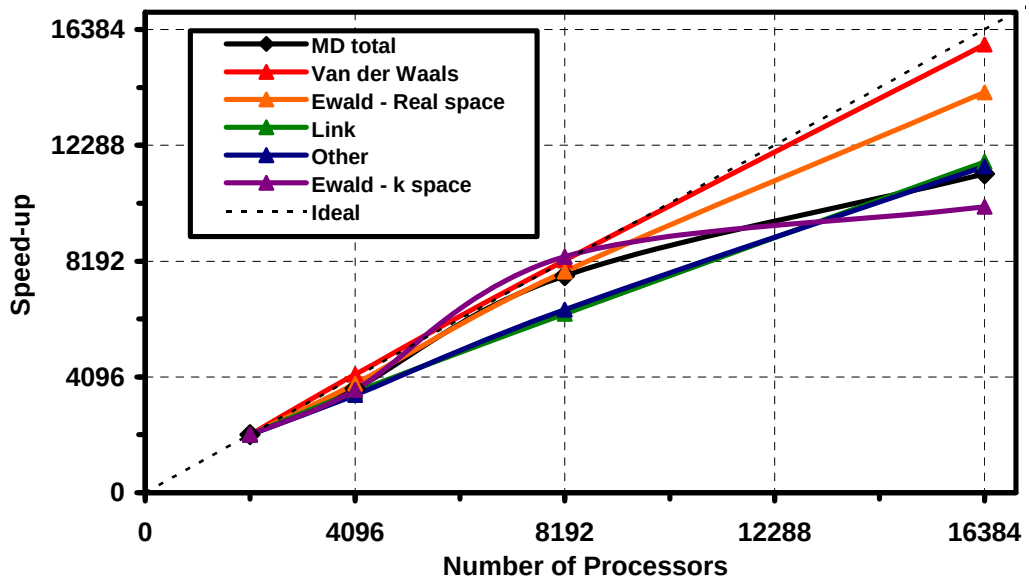

Figure 6. Speed-up of various components of the DL_POLY_3 code for a 14.6 million particle system on the Jülich IBM BlueGene/L

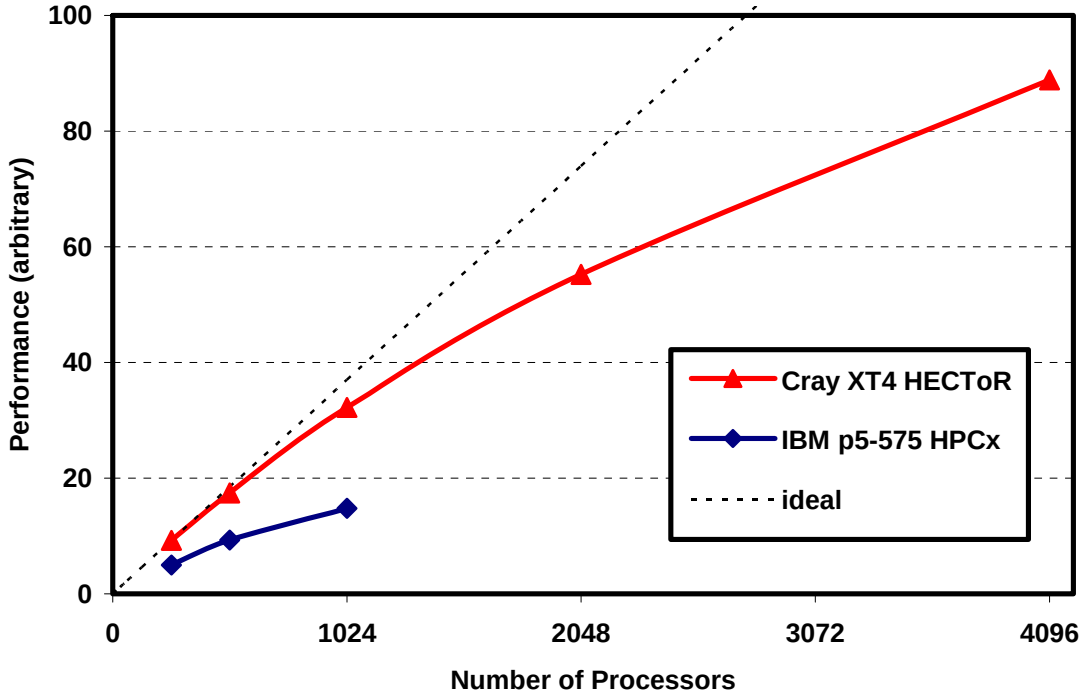

Figure 7. Performance of CRYSTAL for the 1737 atom, 23268 basis function test case on the HECToR Cray XT4 and the HPCx IBM p5-575 systems

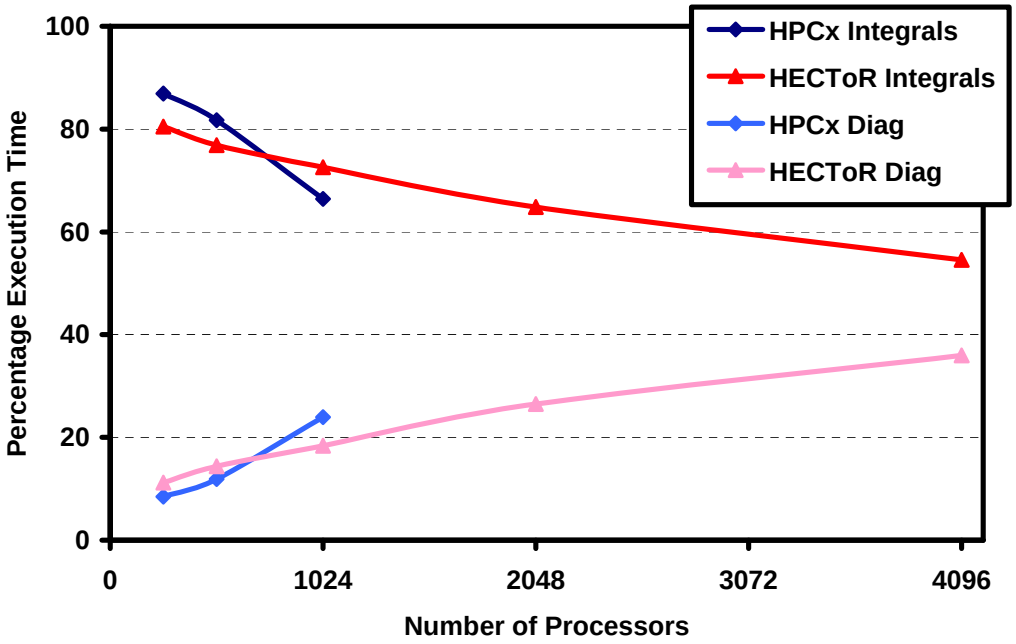

Figure 8. Percentage execution time spent in the Diagonaliser and the Integral calculations in CRYSTAL for the 1737 atom, 23268 basis function test case on the HECToR Cray XT4 and the HPCx IBM p5-575 systems

For both time to solution and memory usage the scaling of these two components is very different. The building of the Kohn-Sham matrix involves the evaluation of a very large number of integrals, each of which is independent. Therefore the time to solution for the problem scales very well. However the access patterns in the data objects required for this portion of the code is very irregular, and hence difficult to distribute. Therefore at present CRYSTAL replicates these objects (though it does store them in sparse format), so leading to a very poor scaling in memory usage.

On the other hand the data objects used in the dense linear algebra are relatively easy to distribute, and standard libraries (e.g. ScaLaPACK) exist that can use the distributed form. However the resulting ratio of computation to communication is not as good as one might hope for, especially in the diagonalisation phase, resulting in relatively poor scaling of the time to solution.

Thus the scaling of the time to solution is a competition between these two phases in the calculation. This can be seen in Figure 8 which shows the execution time spent in the diagonaliser and in the integral calculations for this problem on the HECToR Cray XT4 and the HPCx IBM p5-575 systems. At low processor counts the building of the matrix dominates, however, due to its inherently very good scaling, at high processor counts it becomes less important, and the diagonalisation starts to dominate.

For scaling to very large numbers of processors it will almost certainly be necessary to re-engineer both parts of this scheme. To exploit such large numbers of processors it will be necessary to use very large system sizes, but the replicated data in the matrix build will probably put unsuitably tight limits on the size of system that might be studied. We note that the memory per processor on HECToR is three times that of the initial HPCx installation (which had 1GB per processor), while the number of processors has increased by roughly a factor of ten. On IBM BlueGene systems which are typically limited to 512 MB per processor, the memory limitation is even more severe.

Once this is solved then it will probably be necessary to address the poor scaling of the diagonaliser. While "diagonalisation-less" methods of solution are known, which use direct minimization methods to find the lowest energy, they have been very much less used in Quantum Chemistry than the standard method and research on how best to implement them is required. For instance how to treat metals is not clear, and one should note that this problem potentially affects all calculations for while the final solution may not be metallic, intermediate stages may be.

5. Conclusions

We have looked at the scalability of five large-scale application codes up to the largest processor counts available to us at this time; 16384 processors on an IBM BlueGene/L and 8192 processors on a Cray XT4. The two CFD codes, SBLI and Code_Saturne, scale well given sufficiently large problem sizes. POLCOMS has known scaling problems associated with the load balance obtained from the land/sea distribution but we believe this is not a fundamental limitation to its scalability. DL_POLY_3 scales well for large problem sizes if care is taken with the method used for the long-range force field. CRYSTAL scales reasonably well for large problems with known issues surrounding the replicated memory used in the integrals evaluation and the scalability of the diagonalisation. The issue of the scalability of diagonalisers is further explored by Sunderland [16]. The prospects for all these codes look good to exploit higher processor counts.

In addition to the need for large problem sizes, we have identified a requirement for further research in efficient parallel I/O, parallel partitioning for unstructured mesh codes and diagonalisation-less methods for quantum chemistry. The need for parallel I/O is particularly urgent as none of the five benchmark cases reported here includes I/O.

Of course there are other issues besides scalability. The efficiency of processor utilisation is key to exploiting large-scale systems, especially as nodes move to multi-core chips; quad-core and beyond. The current programming model of Fortran plus MPI continues to work well, though it is possible that hybrid MPI-OpenMP methods may assist in the exploitation of multi-core nodes. This reduces the total number of MPI tasks, with a consequent reduction in communications overheads particularly for collective operations, and also allows more efficient use of memory as all cores on the chip have access to shared memory structures. Furthermore global address space languages may provide an efficient high-level interface to single-sided communication protocols.

Acknowledgments

Access to HECToR and HPCx has been facilitated through the Service Level Agreement between CSED and the Engineering and Physical Sciences Research Council. We are grateful to CSCS for access to the Cray XT3 and to Forschungszentrum Jülich for access to their IBM BlueGene/L as part of the Blue Gene Scaling Workshop, 5^{th}-8^{th} Dec 2006.

References

[1] *High performance computing and computational aerodynamics in the UK*, D.R. Emerson, K. Badcock, A.G. Sunderland and M. Ashworth, The Aeronautical Journal, 111, (1117), 125-132.

[2] *A Strategic Framework for High-End Computing*, The High End Computing Strategic

Framework Working Group, 2006, http://www.epsrc.ac.uk/CMSWeb/Downloads/Other/2006HECStrategicFramework.pdf

[3] *High Performance Computing beyond the Peta Scale in Japan*, Toichi Sakata, Proceedings of SciDAC 2007, 24–28 June 2007, Boston, Massachusetts, USA, Journal of Physics: Conference Series 78 (2007) 012059

[4] *Migrating a Scientific Application from MPI to Co-Arrays*, J.V. Ashby, Proceedings of the Cray User Group, 5-8 May 2008, Helsinki

[5] *Application Performance on the UK's New HECToR Service*, F. Reid, M. Ashworth, T. Edwards, A. Gray, J. Hein, P. Knight, A.D. Simpson, K. Stratford and M. Weiland, Proceedings of the Cray User Group, Helsinki, 5-8 May 2008

[6] *Direct Numerical Simulation of Shock/Boundary Layer Interaction*, N.D. Sandham, M. Ashworth and D.R. Emerson, http://www.cse.scitech.ac.uk/ceg/sbli.shtml

[7] *Using Cray Performance Analysis Tools*, S–2376–41, Cray Inc., 2007

[8] *Code_Saturne: A Finite Volume Code for the Computation of Turbulent Incompressible Flows – Industrial Applications*, F. Archambeau, N. Mechitoua and M. Sakiz, International Journal on Finite Volumes, 1 (1), 2004

[9] METIS - Family of Multilevel Partitioning Algorithms http://glaros.dtc.umn.edu/gkhome/views/metis/

[10] SCOTCH: Static Mapping, Graph, Mesh and Hypergraph Partitioning, and Parallel and Sequential Sparse Matrix Ordering Package http://www.labri.fr/perso/pelegrin/scotch/

[11] *Optimization of the POLCOMS Hydrodynamic Code for Terascale High-Performance Computers*, M. Ashworth, J.T. Holt and R. Proctor, Proceedings of the 18th International Parallel & Distributed Processing Symposium, 26-30 April 2004, Santa Fe, New Mexico

[12] *DL_POLY_3: the CCP5 national UK code for molecular-dynamics simulations*, I. T. Todorov and W. Smith, R. Soc. Phil. Trans.: 362, 1835-1851, 2004

[13] *DL_POLY_3: new dimensions in molecular dynamics simulations via massive parallelism*, I. T. Todorov, W. Smith, K. Trachenko and M. T. Dove, J. Mater. Chem., 16, 1911-1918, 2006

[14] *The Need for Parallel I/O in Classical Molecular Dynamics*, I.T. Todorov, Proceedings of the Cray User Group, Helsinki, 5-8 May 2008

[15] *The CRYSTAL Home Page* http://www.crystal.unito.it/

[16] *Investigating the Performance of Parallel Eigensolvers on High-end Systems*, Andrew Sunderland, Proceedings of the Cray User Group, Helsinki, 5-8 May 2008

About the Authors

Mike Ashworth leads the Advanced Research Computing Group in the Computational Science & Engineering Department at STFC's Daresbury Laboratory since 2002. The Group is engaged in the development and optimization of large-scale applications for high-performance systems across a wide range of scientific disciplines. His own work focuses on the development and optimization of environmental modelling and CFD codes, including performance engineering and application of Grid technologies. He also serves on the CUG Board of Directors as Director-at-Large. E-Mail: m.ashworth@dl.ac.uk.

Ian Bush is a Computational Scientist in the Advanced Research Computing Group focussing on the development and optimisations of methods for computational chemistry and materials codes. E-mail: i.j.bush@dl.ac.uk.

Charles Moulinec is a Computational Scientist in the Computational Engineering Group working on the development and optimisation of CFD codes. E-mail: c.moulinec@dl.ac.uk.

Ilian Todorov is a Computational Scientist in the Advanced Research Computing Group working the development and optimisation of molecular dynamics codes. E-mail: i.t.todorov@dl.ac.uk.