CASTEP: Quantum Mechanical Atomistic Simulation Code

Dr Ben Jesson,
CSAR Optimisation Support

May 2000 saw the release to the UK academic community of version 4.2 of the quantum mechanical atomistic simulation computer code, "CASTEP". The release of this code, which has been keenly awaited by many CSAR users, is the result of an extensive collaboration, over several months, between the academic developers of the code, the Daresbury HPCI centre, and the CSAR applications and optimisation support teams.

"CASTEP" (CAmbridge Serial Total Energy Package), originally written in the 1980's by Mike Payne at Cambridge University, is now developed by the UK Car-Parrinello (UKCP) consortium of academic research groups (1) and a commercial partner, Molecular Simulation Inc (MSI) (2). MSI markets and sells the code to commercial users world-wide, but under the terms of a UKCP-MSI agreement, CASTEP is available for free to all UK universities (CSAR users wanting to use CASTEP should contact Dr. Phil Lindan at the Daresbury Laboratory (3)).

For a fuller description of the purpose of CASTEP, its functionality and the science that underpins it, the reader is referred to the web pages referenced in the footnotes, and the publications listed therein.

In essence, however, CASTEP uses density functional theory (DFT) (specifically, using plane-waves and pseudopotentials) to solve approximately the Schrödinger equation for periodic systems of atoms, yielding the total energy, atomic forces and internal stresses in the system, as well as interesting electronic properties (the electron wavefunction, charge density distribution, density of electronic states, etc).

CSAR's role

The development and optimisation support provided by CSAR for CASTEP was carried out by Ben Jesson and Stephen Pickles from the CSAR team, working in conjunction with Rob Allan and Ian Bush of the Daresbury HPCI centre, who have previously worked substantially on the CASTEP code.

The first task of this team was the parallelisation of some key parts of the CASTEP code. CASTEP is a large and complex piece of software, consisting of approximately 120,000 lines of Fortran 77/90. It has been developed over more than a decade by a number of people, and during that time much of the older parts of the code have been parallelised using the Message Passing Interface (MPI) library. However, some of the most important new functionality of the previous CASTEP release, version 3.9 (including the 'density mixing' electronic minimiser and the ability to use 'ultrasoft' pseudopotentials), had not been parallelised, dramatically limiting the utility of CASTEP for CSAR users.

Further, even when this parallelisation work was complete, it was still not immediately possible to perform the very large CASTEP calculations that members of the UKCP were hoping to run on the CSAR T3E. This was due to the memory usage of some CASTEP parallel algorithms, which was such that, regardless of the number of processors, the memory required on each one would always exceed that available (ie 256 Mb on the CSAR T3E).

The second phase of CSAR's development work was therefore to address this problem, specifically for the Cray T3E (the other CSAR computers, an Origin2000 and VPP300, have more memory available to each processor and so do not suffer so acutely from these problems).

Finally, the substantial and in-depth knowledge of the CASTEP code that has been gained through this development work has been applied to the optimisation of the code, particularly aimed at the Cray T3E.

Results

It is rather hard to illustrate the results of the initial parallelisation work performed by CSAR since, for example, the precise parallel speed-ups obtained will depend strongly on the nature of the calculation performed and, in any case, will be affected by the ongoing optimisation work. Nevertheless, good parallel performance was obtained for the test case calculations used in the work.

The improvement in the memory usage of the parallel code was acheived through the implementation of a number of alternative parallel algorithms for which the memory requirement could be scaled down by using more processors. In the case of some of CASTEP's file input and output routines, this necessitated the use of certain Cray-specific I/O routines. The results of this work are illustrated for one medium-sized test case, a 37-atom simulation of an aluminium impurity in silica, in figure 1.

As can be seen, the memory usage per processor, even when many processors are used, tends to a particular lower limit below which it cannot further be reduced. For large systems, this lower limit was greater than the 256 Mbytes available on the T3E.

However, following the implementation of alternative parallel algorithms, the limiting memory usage has been substantially reduced. For larger systems it is likely that this reduction will be even more dramatic than that shown here, and indeed much larger CASTEP calculations can now be performed on the Cray T3E than were hitherto possible.

The development work already described has been aimed primarily at getting the latest CASTEP functionality working on the T3E for large calculations. Now that this has largely been achieved, we are beginning to turn our attention to the optimisation of the code, so that these calculations can be performed as efficiently as possible. This task is complicated by the fact that optimisation needs of the code depend strongly on the nature of the calculation being performed. However, some initial optimisation results have been obtained, which can be expected to make some significant improvement to the performance of the code for a wide range of calculations.

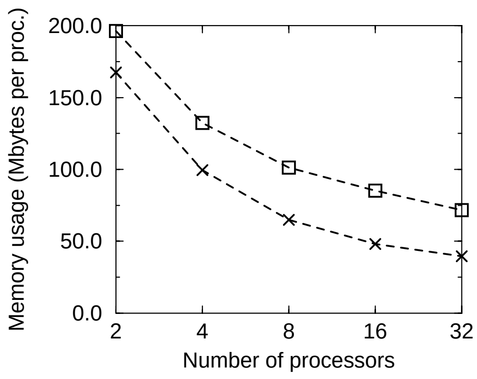

Figure 1 : Per-processor memory usage for a typical medium-sized CASTEP calculation. Squares and crosses show memory usage before and after the CSAR work.

These particular optimisations focus on the implementation of the Fast Fourier Transform (FFT) algorithm in the code, which for many calculations can account for a very significant part of the total execution time. CASTEP uses a three-dimensional FFT on a distributed data grid, which involves three sets of non-distributed one-dimensional FFTs, interspersed with intra-processor and inter-processor data copy operations.

Various optimisations have been applied to this part of the code, the results of which are indicated in figure 2. This plot shows CPU times per FFT, for a range of grid sizes, before and after optimisation. The squares indicate timings taken from the original unoptimised routine, and the crosses show the result of applying a rather general cache-based optimisation to the 1-D FFT part of the code, which should show a s similar benefit an all cache-based computer architectures (including, for example, an SGI Origin2000 such as fermat). The diamonds show timings obtained using a Cray-specific library routine for the 1-D FFT, and the triangles show the effect of also using a Cray-specific routine for the intra-processor data copy operations. These results illustrate rather well the general principle that, while some generic optimisations can make a very significant difference to the performance of a piece of code, often vendor-specific library routines can provide much greater performance for relatively little programmer effort.

The timings shown in figure 2 focus exclusively on the FFT routine, and therefore in a real example the improvement in performance may not be quite as dramatic as it suggests. However, for the medium-sized calculation described above, the overall execution time of the code on 16 processors drops from 312 seconds with the original FFT to just 183 seconds when using the Cray-specific routines, a 70% improvement in performance!

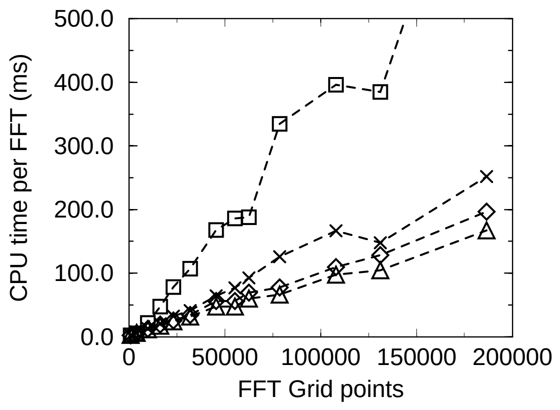

Figure 2 : CPU times per FFT as a function of grid size, before and after various optimisations as described in the text.

Conclusions

The parallelisation and optimisation support provided by CSAR to the UKCP consortium in connection with the CASTEP code is clearly of an unusually extensive and in-depth nature. However, it serves as a good illustration of the enormous benefits that can accrue from such a relationship. For example, although CSAR's support for CASTEP has continued over a number of months, the optimisation work described above took place over only a small fraction of that time, and it is not unreasonable to expect that other codes could similarly benefit from the investment of just a few support tokens!

Footnotes

(1) http://www.cse.clrc.ac.uk/Activity/UKCP

(2) http://www.msi.com/materials/cerius2/castep.html

(3) e-mail: ukcp@dl.ac.uk