Scalable Parallel Electronic Structure Calculations on the IBM SP2

Stefan Goedecker[1], Adolfy Hoisie[2]

[1] Max-Planck Institute for Solid State Research, Stuttgart, Germany
[2] Cornell Theory Center, Ithaca, USA
email: goedeck@prr.mpi-stuttgart.mpg.de

Abstract

We have developed an highly efficient and scalable electronic structure code for parallel computers using message passing. The algorithm takes advantage of the natural parallelism in quantum chemistry problems to obtain very high performance even on a large number of processors. Most of the terms which scale cubically with respect to the number of atoms have been eliminated allowing the treatment of very large systems. It uses one of the most precise versions of Density Functional Theory, namely Self-Interaction Corrected Density Functional Theory.

1 Density Functional Theory

The laws of Quantum Mechanics describe matter such as atoms, molecules and solids on a microscopic level. These laws are therefore not only the central laws of chemistry, but also of many fields of physics, materials sciences and biology. Their solution is numerically very expensive. Rather crude approximations based on classical mechanics are therefore frequently used to get a rough understanding of the structure of molecular systems. These classical force fields cannot describe such important phenomena as the breaking of bonds. There is therefore a enormous interest to develop accurate computational methods, that use the correct quantum mechanical equations. The quantum mechanical calculation of systems containing more than a 100 atoms on vector supercomputers has only very recently become possible, both by algorithmic advances [1, 4] and hardware progress. The treatment of even larger systems will only be possible on massively parallel machines because both of memory and speed requirements.

Among the many methods for electronic structure calculations (i.e. the solution of the equations of Quantum Mechanics), Density Functional theory (DFT) [2] is one of the most popular one. It is at the same time highly accurate and faster than traditional quantum chemistry methods such as Hartree Fock (HF) theory. Recently, a lot of interest focused on improving the standard Local Density Approximation (LDA) of density functional theory. Gradient corrected DFT is one such scheme. Very encouraging results have been reported that demonstrate accuracy comparable to good quantum chemistry methods. Gradient corrected schemes cancel most of the non-physical self-interactions of the LDA approximation, which are responsible for most of its deficiencies with respect to accuracy.

Another approach to improve upon LDA is to introduce the self-interaction corrections [3] (SIC) directly. This leads to a numerically more complicated scheme since now each electron orbital feels a different potential. Therefore, it is no longer possible to view the electronic structure problem as an eigenvalue problem in a external potential that has to satisfy a self-consistency condition. Using a preconditioned gradient minimization approach [4], with an DIIS convergence accelerator [5], we are able to find the electronic ground state in a number of iterations not much larger than the number of iterations necessary in ordinary LDA theory. Plane waves are used as basis set in this calculation. This necessitates pseudo-potentials to eliminate the highly localized electronic core states, that are impossible to describe by a reasonable number of plane waves.

2 General Outline of the Computational Tasks

An electronic structure program of this type consists of a fairly large number of subtasks performing different calculations that vary widely with respect to their computational needs (e.g., memory bandwidth or floating point performance). Since the IBM Power 2 architecture is well balanced it is possible to obtain very good performance on all these subtasks. Several new algorithmic techniques were developed, that reformulate the computational subtasks in such a way as to obtain very high performance on modern RISC architectures such as the RS/6000 Power 2 architecture. Details will be given in the next section. In this section we will discuss the general outline of the main computational tasks.

The program is based on the orbital parallel paradigm, i.e. each processor calculates one or several localized orbitals. Physical interactions between the orbitals lead to inter-processor communication. The two basic interactions involved are related to the fact that the orbitals repel each other electrostatically and that they have to be mutually orthogonal to satisfy the Pauli exclusion principle. Since the calculation of one localized orbital and its potential takes most of the CPU time, this parallelization scheme leads to rather modest communication requirements and high parallel performance.

The electrostatic potential acting on a given orbital is obtained by Fast Fourier Transform (FFT) techniques. The charge density giving rise to this potential is obtained by summing over the charge densities of all the other orbitals in the system. In practice, this is effectuated by doing first a global inter-processor reduction sum over all the orbitals using the MPI ALLREDUCE routine and then subtracting the charge density of the orbital on which the potential is acting. If one uses FFT techniques to solve Poisson's equation the total charge (i.e. electronic plus ionic charge) has to be zero. Due to the electrostatic self-interaction correction, this is not the case. Therefore, one has to add a localized neutralizing charge distribution, that is analytically representable, and whose electrostatic potential is known analytically as well. The simplest charge-potential pair satisfying these requirements is a Gaussian charge distribution that gives rise to a potential of the form $\frac{erf(r/a)}{r}$, where erf is the error function. The centre of the Gaussian charge distribution is chosen to coincide with the centre of the localized orbitals in order to preserve conservation of crystal momentum in periodic structures. The calculations of the exchange correlation potential in the SIC-LDA scheme requires again the total electronic density and the individual orbital densities for evaluating the exchange correlation self-interaction correction. Most of the published exchange correlation potential forms contain many special functions such as logarithms and square roots, that are very slow to calculate. We used a new rational functional form, that is an order of magnitude faster to evaluate than these other forms.

Traditionally, pseudo-potentials are applied in Fourier space rather than real space /citereal. This leads first to large cubically scaling terms (that prevent one from calculating very large atomic systems) and second to large memory requirements. In our approach, we have implemented a new pseudopotential with optimal properties for real space implementation [7]. The computational load due to this form of pseudopotential scales only quadratically with the molecular system size. Also, since it has a very simple analytical form, it can be evaluated on the fly (i.e. it is recalculated every time it is needed) and has therefore very small memory requirements. This is essential in bringing down the memory requirements for large runs.

The kinetic energy of each orbital is calculated in Fourier space, where the kinetic energy operator is a diagonal matrix. In this part FFTs are thus again involved. Full advantage is taken of the fact that the orbitals are real (leading to real-to-complex FFT) and of the fact that in the initial stages of the FFT most of the Fourier coefficients are zero.

Several new techniques are utilized for the orthogonalization of the orbitals. For a symmetric Löwdin orthogonalization, one needs the overlap matrix between the occupied orbitals. Optimal load balancing can be obtained only if each processor calculates some fraction of each element of this overlap matrix S. This computation requires a data structure that is entirely different from the basic orbital parallel data structure. We have developed a data transformation, consisting of two data transpositions. One of them is an on-processor transposition, whereas the second one is a global inter-processor transposition handled by a single call to the all-to-all MPI routine. The sequence of these two transformation reorders the data in the required way. The calculation of the matrix $S^{-1/2}$ is done by a fixed point iteration, allowing full parallelization of this part as well. This orthogonalization part is the only part that leads to cubically scaling terms. In addition it also leads to some quadratically growing communication terms. Because it is fully parallelized the cubic computational part is negligible even for 500 atom systems. However the quadratic communication part becomes the major bottleneck of the program for large systems as will be discussed below. In principle this bottleneck could easily be removed by taking advantage of the sparsity of the overlap matrix in big systems. Reducing the scaling would however introduce additional overhead for medium size systems which are usually dealt with in production runs and was therefore not done.

3 Detailed description of the main subroutines

3.1 Fast Fourier routines

Fast Fourier Transformations have a large ratio of load/stores to floating point operations. On RISC processors, FFT performance is usually limited by the speed at which data can be fed into the CPU. We have therefore implemented a data access pattern [9] that leads to mainly stride one data access and long inner loops. A feature that turns out to be very useful to boost the performance of FFT's is the RS6000/POWER 2 quadruple load/store instruction that can load/store two double precision numbers within one cycle. Since each of the two fixed point units can issue this instruction simultaneously, one can actually load or store 4 double precision numbers within one cycle. In addition, we use an FFT kernel which has an optimal balance of multiplications and additions [10] and runs therefore virtually at peak speed in cases where the data is available from cache. Under these circumstances, our kernel is actually slightly faster than the kernel of the FFT's in the highly optimized IBM ESSL library. To get this high performance we again take advantage of a special RS/6000 instruction, namely the Floating point Multiply Add (FMA) instruction. This instruction calculates an expression of the type $x = a \times b + c$

with a repeat frequency of one cycle and the result is available after one additional latency cycle.

3.2 Calculation of the exchange correlation potential

The exchange correlation functional is given by

$$\epsilon_{xc} = -\frac{a_0 + a_1 r_s + a_2 r_s^2 + a_3 r_s^3}{b_1 r_s + b_2 r_s^2 + b_3 r_s^3 + b_4 r_s^4}$$

where $r_s \propto \rho^{-1/3}$ and the coefficients a and b are given in reference [7] . The calculation of r_s from the charge density ρ is numerically much more expensive than the evaluation of the rational function. Unfortunately, the calculation of r_s can not be avoided, since only a rational function of r_s gives a correct asymptotic behaviour of $1/r_s$ at low densities. To speed up this part of the calculation, we compute r_s by a Newton iteration of the equation $r_s^3 = 1/\rho$. We take advantage of the fact that the exchange correlation potential is not needed to full machine precision. We also make use of the smooth variation of the charge density on the grid which ensures that the solution of the Newton iteration on one grid point is a very good initial guess for the Newton iteration of the next grid point. Using all these tricks, the calculation of r_s is speeded up by a factor of 3 compared to a straightforward evaluation of $\rho^{-1/3}$. Because of the Newton iteration divisions are an important part in the calculation of the exchange correlation potential. They need 14 to 15 cycles to execute on the RS/6000 architecture. This part runs virtually at the highest possible speed for the combination of division and FMA's found in this section of the code.

3.3 Electrostatic mono-pole correction

As mentioned above, the mono-pole correction requires the addition of a neutralizing Gaussian charge density to the total charge density and subtracting a $\frac{erf(r/a)}{r}$ potential from the solution obtained by the FFT techniques. A direct evaluation of a Gaussian and error function on all the grid points is extremely expensive. We therefore approximate both of them by a Chebychev expansion of order 50, which gives close to machine precision accuracy for the values of r encountered in the computational problem. We schedule the calculation of both special functions such that only one evaluation of the Chebychev recursion is needed to evaluate both of them. For the determination of the centre of mass of each localized orbital, we multiply the orbital charge density by a smooth cutoff function, that is also related to an error function. This function is also expanded in a Chebychev polynomial of degree 50. Because the pipeline of the RS6000 architecture is very shallow (just 2 cycles for a FMA), it is not difficult to avoid dependencies in the Chebychev recursion by some modest loop unrolling and this part runs practically at peak speed as well.

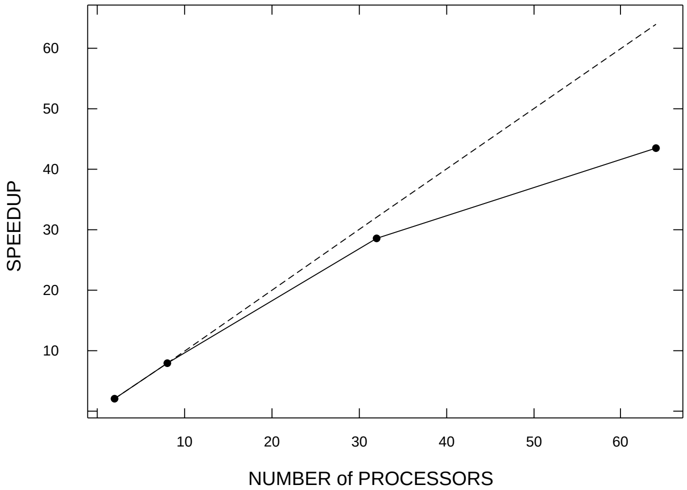

Figure 1: *Speedup of this electronic structure code for a 64 atom Silicon system on a IBM SP2. The solid line is the measured speedup, the dashed line the ideal speedup.*

3.4 Communication subroutines

In addition to the IBM MPL message passing library, two other highly efficient implementations of message passing libraries are available on the the SP2 series, namely the public domain PVM library and the recently adopted new message passing standard MPI. Because MPI offers the largest choice of collective message passing routines and because MPI implementations are now available on most parallel computers we used MPI in this work. Because we used an intrinsically parallel algorithm [8], it turned out that the three basic communication steps could be done by just using two high level MPI subroutines, namely ALLREDUCE for the calculation of the charge density and the build up of the overlap matrix and ALLTOALL for the orbital rotations. As a result, the mapping of the physical methods into a parallel algorithm was natural and straightforward.

In large applications containing several hundred atoms that we run on a regular basis on a 512 node SP2 in Cornell, the minimum number of processors is usually dictated by memory requirements. Strong scalability can therefore only be demonstrated for small systems. For the case of a 64 atom Silicon system this is shown in Figure 1.

Since one SP2 node is a very powerful number crunching engine, that can calculate one or a few orbitals in very little time (a few minutes), weak scalability is more important in practical applications, where one wants to treat big systems. The communication requirements per node for the transposition and charge reduction sum grow slightly faster than linear (like $N log_2(N)$, where N is the size of the system). Since the computational load grows in the same way the fraction of the total time going into these two communication parts is nearly constant (Table 1) leading to a perfect weak scaling behaviour. It has to be pointed out that this excellent scaling behaviour of the ALLTOALL routine is directly related to the topology of the High-Performance Switch. SP2's omega-network allows simultaneous communications from each node to any other processor. As mentioned above, the communication part to build up the overlap matrix grows however faster than all the other parts in the present implementation and causes deviations from the otherwise perfect weak scaling behaviour for very large systems.

Table 1: Fraction of the total elapsed time that goes into communication for several systems containing varying numbers of Silicon (Si) atoms on configurations containing between 64 and 256 processors.

	64 procs, 128 Si	128 procs, 256 Si	256 procs, 512 Si
orbital rotation	.04	.03	.03
reduction sum charge density	.07	.07	.07
reduction sum overlap matrix	.09	.14	.31

4 Conclusions

We have developed a scalable and highly optimized electronic structure program, that allows highly accurate calculations for large systems using the fundamental laws of Quantum Mechanics. The SP2 parallel computer turned out to be the most appropriate platform for these calculations. Running at roughly half its theoretical peak speed, one processor can rapidly calculate a few orbitals belonging to one atom. By simultaneously increasing the number of atoms and the number of processors we can scale up these calculations to several hundred atoms. For large runs with up to 256 processors, the fraction of communication from the transposition and reduction sums remains nearly constant. It is to expected, that the combination of large parallel supercomputers and efficient algorithms adapted to these architectures will make significant contributions to many fields of science and engineering. This application to electronic structure of materials using Self-Interaction corrected Density Functional Theory is a powerful example.

Acknowledgements

We thank J. Hutter, M. Teter and C. Umrigar for interesting discussions and M. Parrinello for useful comments on the manuscript. The support of the Cornell Theory Centre and the access to their SP2 high performance parallel computer was essential during the whole development of the program.

References

[1] R. Car and M. Parrinello, Phys. Rev. Lett. **55**, 2471 (1989)

[2] R. G. Parr and W. Yang, "Density-Functional Theory of Atoms and molecules", Oxford University Press,1989

[3] J. P. Perdew and A. Zunger, Phys. Rev. B **23**, 5048 (1981)

[4] M. P. Teter, M. C. Payne and D. C. Allan, Phys. Rev. B **40**, 255 (1989)

[5] J. Hutter, H.P. Lüthi and M. Parrinello, Comp. Mat. Sci. **2** 244 (1994)

[6] R. D. King-Smith, M. C. Payne, and J. S. Lin, Phys. Rev. B **44**, 13063 (1991)

[7] S. Goedecker, M. Teter and J. Hutter to be published in Phys. Rev. B

[8] S. Goedecker, J. of Comp. Phys. **118**, 261 (1995)

[9] S. Goedecker, Comp. Phys. Commun. **76**, 294 (1993)

[10] S. Goedecker, SIAM, J. of Sc. Comp. to be published