

Programming Paradigms, Tools and Algorithms for the Spectral Solution of the Electronic Schrödinger Equation on Non-Uniform-Memory Parallel Processors

Background

We propose to develop software tools and methods that are appropriate for current and future generations of large scale shared memory computer systems. Our purpose is to enable a more productive utilization of these architectures for scientific computation. Work undertaken by our team will extend and drive the development of programming paradigms, tools and algorithms for this important machine class, with particular emphasis on assessing the influence of memory architecture on code performance. We will specifically focus on algorithms for solving differential equations appropriate to quantum chemistry, which are both computationally and memory intensive. To accomplish our goals we bring together a team of world class scientists taken from academia, the developers of the *Gaussian* computational chemistry code (Gaussian Inc.), and an innovative multinational computer hardware/software company (Sun Microsystems).

Our emphasis on shared memory parallel processors recognizes the fact that, in the last five years, this environment has become increasingly prevalent and available. Dramatic increases in single processor performance that continue to track Moore's Law [1] have decreased the need for massive parallelism for many applications. Moreover, improvements in memory system hardware, such as the development of non-uniform memory access (NUMA) technologies, have permitted shared address space architectures to scale to processor numbers that were previously the sole domain of distributed memory machines. However, maximizing performance on large shared memory processors without resorting to an essentially distributed memory programming model presents substantial challenges in the design of all levels of software; from compilers to operating systems, and from operating system related middleware to the end user's application code. Addressing these challenges, while also furthering the development of an exciting new category of computational chemistry methods, is the aim of this proposal.

In researching different parallel algorithms for solving differential equations we will use both libraries for accessing on-chip performance monitoring counters (e.g. [2, 3]) and complete machine simulation [4]. The former are useful for obtaining statistics, e.g. the ratio of cache misses on code sections of interest, while the latter can yield more detailed information, e.g. cache miss ratios broken down in terms of their various causes (mapping conflicts, misses arising from previous coherency-related invalidations, etc). Simulation also permits performance to be evaluated on non-existing variants of a machine, for example, to predict performance changes resulting from changing the cache line size. The downside of simulation is that it is usually slow. Also the majority of simulators are expensive commercial products beyond the means of most software developers. Thus, to date simulation has not been widely used as a tool to aid in algorithm design and analysis. This project aims to demonstrate this utility in the design of scalable algorithms for solving differential equations. We will build on the substantial work already undertaken at the Australian National University (ANU) to develop a comparatively fast, freely available SPARC based full machine simulator [5]. Our work will involve extending the existing simulator (e.g. to include a variety of memory models), normalizing the results of simulation to those obtained using on-chip counters on existing hardware, predicting performance on as yet un-built machines, and exploring operating system changes (such as modifications to the page allocation and migration algorithms [6]).

As already noted, our project will focus on algorithms for solving differential equations appropriate to quantum chemistry. Specifically we will use the *Gaussian* computational chemistry package. This is the market leading quantum chemistry code that is used extensively throughout the world [7]. At the time of writing the program comprises over a million lines of code and is available for virtually all current generation computing platforms. While the code can perform a variety of different computations, the majority solve the Schrödinger equation for a system of nuclei and electrons. The implementation in *Gaussian*, like most chemistry applications, relies on the well known fixed nuclei approximation where the differential equations are solved for the motion of the electrons moving in the field of the fixed nuclei. Solving this problem is referred to as solving the electronic Schrödinger equation, or solving the electronic structure problem. However, even within the confines of fixed nuclei, further approximations are required. Different approximations give rise to different electronic structure methods, and invariably these all result in a set of coupled partial differential equations that must be solved. Differential equations can be solved either on a physical grid, or by using a spectral expansion. *Gaussian* derives its name from the fact that a spectral approach is used, expanding the solution(s) in a basis of Gaussian functions located on each nuclear centre.

Under the broad umbrella of electronic structure methods, our focus will be linear scaling and related methods [8]. As the name suggests these algorithms are designed to show asymptotic linear scaling with system size, i.e. with the number of atoms in the computation. This is a dramatic reduction in cost from the cubic or higher order scaling that was previously the norm (see Significance and Innovation). This area has experienced rapid development in the last five years, in part because increases in computer power now permit systems of sufficient size to be studied that the idea at the core of linear scaling methods, namely locality of interactions, becomes meaningful. Linear scaling electronic structure methods promise to revolutionize computation, opening up the possibility of first principles quantum calculations on nanoscale systems, such as enzymes, ion channels, nanotubes, molecular electronic devices etc. This will be especially true if, as is the objective of this proposal, we can develop highly scalable parallel implementations for shared memory computers.

Implementing efficient parallel algorithms for these new methods presents a number of problems. Two critical issues are computational load balance, and optimal partitioning and placement of the large sparse arrays that are involved. Load balancing is crucial, since the computational tasks at the core of the new methods depend greatly on the distance between the atoms involved. Essentially, atoms that are close give rise to much work; those separated by long distances give rise to little. Data partitioning is important since a truly scalable algorithm should not replicate any $O(N)$ data structure, while data placement is critical to ensure that frequently-used data is “close” to where it is required. Given these difficulties, performance simulation will be a vital tool in our research into parallel implementations of these methods.

In addition to the technical difficulties of developing efficient parallel algorithms, non-trivial software engineering issues are associated with implementing and maintaining these algorithms within a large commercial package like *Gaussian*. Our work will be of little merit if the code we develop falls into disrepair because it is too costly to maintain. To have maximum impact our software must be readable, extensible and maintainable, and should not deviate substantially from the standard code which is continuously being developed by people outside of this project. For these reasons the ultimate goal of this project is to work within the OpenMP programming paradigm [9]. This is a standard that, in contrast to other shared memory programming standards like pthreads [10], results in minimal difference between the sequential and parallel source code. We state this as our ultimate goal, because currently OpenMP does not, for example, include any model for task or work queues [11], or any concept of locality in data and process placement. An

objective of this proposal is to demonstrate the utility of these (and other) concepts, then working through our industrial partner Sun, seek to put forward ideas for how the OpenMP standard might usefully evolve. In this respect we note that some extensions to account for NUMA characteristics have already been proposed by some hardware vendors [12]. However there is no general consensus on what these extensions should be, and very little research on the utility of these proposed extensions for real life application codes. Our work will fill this void.

In summary this project brings together experts in computer science, software engineering and electronic structure methods. By working as a team we will be able to produce advances that have a long term practical impact on the field. The project contains elements of mathematical modeling, software engineering and software-hardware co-design; key components of the “Complex Systems” ARC priority research area. Moreover the computational methods that are the target of this work are immediately applicable to the ARC priority research area of “Nano-Materials and Bio-Materials”. Success of this project will be measured in several ways, including (but not limited to):

- The ability to realistically simulate the performance of a highly complex real world computational science application code, and to use this information to aid the design of parallel software.
- The ability to predict the effect of architectural and operating system changes on the performance of a complex application, and thereby give guidance to the hardware and software groups within Sun as to how they may want to change their products.
- Significant advances in the development of efficient NUMA-aware parallel electronic structure algorithms for linear scaling and related methods.
- Proposals for extensions to the OpenMP standard, particularly in the area of task or work queues and NUMA aware bindings.

Significance and Innovation

This proposal targets electronic structure methods for “large” systems, and their efficient implementation on shared memory parallel processors. What is a “large” system requires some discussion. Essentially the asymptotic computational cost of all spectral based electronic structure methods is dependent on the number of functions used in the spectral decomposition. This in turn is linearly related to the number of atoms in the system (as the functions are atom centered). Without pre-screening methods such as Hartree-Fock (HF) or Density Functional Theory (DFT), scale with the fourth power of the number N of functions/atoms – nominally $O(N^4)$. Careful use of screening reduces this to $O(N^3)$ or slightly less [13]. In this context, depending on the computer, implementation etc, a “large” calculation might comprise a few hundred atoms. Other methods, such as the most commonly used coupled-cluster approach [14], scales as $O(N^7)$. As a consequence, a “large” coupled-cluster calculation may have only a few tens of atoms.

More recent approaches for HF and DFT calculations have formally reduced the scaling of the time dominant part of the computation from $O(N^4)$ to $O(N)$ [15, 16]. When coupled with an efficient parallel implementation, these improved algorithms will enable routine calculations on systems with many thousands of atoms. For the more expensive coupled-cluster methods, achieving $O(N)$ scaling is considerably more complicated, although still technically possible [17]. For these methods our short term strategy will be to pursue a well parallelized conventional coupled-cluster method, and then to gradually include linear scaling ideas. This will still be extremely valuable as coupled-cluster methods provide the most accurate, generally applicable, electronic structure method that is currently available [14]. The method is used as a “gold standard”, to which other results are compared. It is widely used for computing accurate heats of

formations that are then used as input data for other computational models, e.g. combustion models. Thus increasing the number of atoms that can be treated with these “expensive” methods, even slightly, is still very significant.

For the average computational scientist, our emphasis on the shared memory parallel environment will be beneficial since small-scale shared memory parallel systems are the norm in many research groups. This is not surprising, since the cost of adding a second processor is usually minimal compared to the cost of the base system. Beyond their local environment this research group is likely to have access to a larger shared memory parallel system. Be that a machine manufactured by Sun, Compaq, SGI etc, beyond a certain size virtually all such machines display some NUMA characteristics. Thus our focus on NUMA parallel implementations is both widely applicable and of much current interest. For example, at this time there is no consensus on the best programming paradigm using NUMA machines. Compaq has proposed a series of extensions to the OpenMP model for use on their GS product line [12], SGI has a different set of ideas for use on their Origin systems [18]. In both cases very little data is available to demonstrate the utility, or otherwise, of their proposed schemes. Our work would fill this gap, and would be extremely timely as the community is beginning to consider how the OpenMP standard may evolve beyond version 2.

This project brings together experts from both the computational chemistry and computer science areas with the aim of dramatically improving the parallel performance of an extremely widely used electronic structure code. To achieve this goal we will use the recently developed approach of (execution-driven) complete machine simulation [19]. This approach will yield the greatest accuracy in the ordering of memory interactions between processors and also capture all interactions and effects such as page mapping policies that arise from the operating system. It has the added advantage that the application code can be simulated unmodified. Such an approach will allow us to analyze in detail the memory access patterns of the code and thereby determine optimal data placement on NUMA architectures. Through simulation we will be able to assess performance on new architecture types without necessitating the building of new hardware. Thus, as well modifying the application to better fit the hardware, we will be in the unique position of being able to suggest hardware modifications that improve application performance.

This project will also advance the field of computer simulation. Traditional simulation techniques require detailed simulation of the CPU pipeline in order to obtain realistic ordering of memory events in the shared memory context. However, this may decrease simulation speed by up to two orders of magnitude. Working closely with our industrial partner, Sun, we will explore computationally inexpensive methods of improving timing accuracy that do not require full pipeline simulation. Optimizing and exploring performance-accuracy tradeoffs in the modeling of NUMA memory systems is also expected to produce novel work. The use of a non-trivial and important application such as Gaussian is essential to validate this work.

References

1. G.V. Post, "Computer technology changes and purchasing strategies", *Adv. Computers*, **54**, 183 (2001).
2. S. Browne, J. Dongarra, N. Garner, G. Ho, P. Mucci, "A portable programming interface for performance evaluation on modern processors", *Int. J. High Performance Computing Applications*, **14**, 189 (2000).
3. R.P. Garg, I.A. Sharapov, "Techniques for optimizing applications: High performance computing", Prentice Hall 2001 (ISBN 0130934763).
4. S.S. Mukherjee, S.V. Adve, T. Austin, J. Emer, P.S. Magnusson, "Performance simulation tools", *IEEE Computer*, **35**, 38 (2002).
5. B. Clarke, A. Czezowski, P. Strazdins, "Implementation aspects of SPARC V9 complete machine simulator", *Proc. Australasian Comp. Sys. Arch. Conf.*, 2002 (cap.anu.edu.au/cap/projects/sulima)
6. D.S. Nikolopoulos, T.S. Papatheodorou, C.D. Polychronopoulos, J. Labarta, E. Ayguade, "Leveraging transparent data distribution in OpenMP via user-level dynamic page migration", *High Performance Computing, Proceedings Lecture Notes in Computer Science*, **1940**, 415 (2000).
7. D.B. Boyd, "The computational chemistry literature", *Reviews in Computational Chemistry*, VCH Publishers, New York, **2**, 461 (1991).
8. S. Goedecker, "Linear scaling electronic structure methods", *Rev. Mod. Phys.*, **71**, 1085 (1999).
9. L. Dagum, R. Menon, "OpenMP: An industry standard API for shared-memory programming", *IEEE Comput. Sci. and Eng.*, **5**, 46 (1998).
10. B. Kuhn, P. Peterson, E. O'Toole, "OpenMP versus threading in C/C plus", *Concurrency – Practice and Experience*, **12**, 1165 (2000).
11. S. Shah, G. Haab, P. Petersen, J. Throop, "Flexible control structures for parallelism in OpenMP", *Concurrency – Practice and Experience*, **12**, 1219 (2000).
12. J. Bircsak, P. Craig, R. Crowell, Z. Cvetanovic, J. Harris, C.A. Nelson, C.D. Offner, "Extending OpenMP for NUMA machines", *Proc. SC2000* (www.supercomp.org/sc2000/Proceedings/)
13. M. Häser, R. Ahlrichs, "Improvements on the direct SCF method", *J. Comput. Chem.*, **10**, 104 (1989).
14. P.R. Taylor, "Accurate calculations and calibration", *Lecture Notes in Chemistry*, **58**, 325 (1992).
15. M.C. Strain, G.E. Scuseria, M.J. Frisch, "Achieving Linear Scaling for the Electronic Quantum Coulomb Problem", *Science*, **271**, 51-53 (1996).
16. G.E. Scuseria, "Linear scaling density functional calculations with Gaussian orbitals", *J. Phys. Chem. A*, **103**, 4782 (1999).
17. G.E. Scuseria, P.Y. Ayala, "Linear scaling coupled cluster and perturbation theories in the atomic orbital basis", *J. Chem. Phys.*, **111**, 8330 (1999).
18. G.R. Luecke, W.H. Lin, "Scalability and performance of OpenMP and MPI on a 128-processor SGI Origin 2000", *Concurrency and Computation-Practice and Experience*, **13**, 905 (2001).
19. M. Rosenblum, S.A. Herrod, E. Witchel, A. Gupta, "Complete computer-system simulation - the SIMOS approach", *IEEE Parallel and Distributed Technology*, **3**, 34 (1995).
20. R. Kobayashi, A.P. Rendell, "A direct coupled-cluster algorithm for massively parallel computers", *Chem. Phys. Lett.* **265**, 1 (1997).
21. A.P. Rendell, T.J. Lee, "Coupled-cluster theory employing approximate integrals: An approach to avoid the input/output and storage bottlenecks", *J. Chem. Phys.*, **101**, 400 (1994).
22. P. Constans, P.Y. Ayala, G.E. Scuseria, "Scaling reduction of the perturbative triples correction (T) to coupled-cluster theory via Laplace transform formalism", *J. Chem. Phys.*, **113**, 10451 (2000).