High-Performance Computational Chemistry: Hartree-Fock Electronic Structure Calculations on Massively Parallel Processors

Jeffrey L. Tilson, Mike Minkoff, Albert F. Wagner,
Ron Shepard, and Paul Sutton
Argonne National Laboratory
Argonne, IL 60439

Robert J. Harrison, Ricky A. Kendall and Adrian T. Wong
Environmental and Molecular Sciences Laboratory
Pacific Northwest National Laboratory
Richland,WA 99352

October 7, 1998

Abstract

The parallel performance of the NWChem version 1.2α parallel direct-SCF code has been characterized on five massively parallel supercomputers (IBM SP, Kendall Square KSR-2, Cray T3D and T3E, and Intel Touchstone Delta) using single-point energy calculations on seven molecules of varying size (up to 389 atoms) and composition (first-row atoms, halogens, and transition metals). We compare the performance using both replicated-data and distributed-data algorithms and the original McMurchie-Davidson and recently incorporated TEXAS integrals packages.

1 Introduction

In recent years, there has been considerable effort in adapting computational chemistry applications to parallel computer architectures, including many parallel implementations of the direct self-consistent-field [1] (direct-SCF) electronic structure method as described in a recent review [2]. Initial work in this area greatly reduced the direct-SCF time to solution by utilizing efficient parallelism in the integral generation step; one common limitation, however, was the use of replicated-data structures. On modestly parallel computers, this limitation is of little practical consequence. On distributed-memory massively parallel processors (MPPs) with hundreds of nodes, however, the replicated-data model

restricts the problem size to fit within the available memory of a single processor. This restriction is very severe given that the memory of a single node may be several orders of magnitude smaller than the aggregate memory.

More recent efforts at constructing parallel direct-SCF codes, distribute all sizable data structures. Colvin et al. [3] were the first to demonstrate such a code. Based on a systolic loop, their algorithm is memory efficient but requires computing integrals up to three separate times each iteration and can suffer from load imbalance. Furlani and King [4] used dynamic blocking and distributed-data techniques to create a parallel direct-SCF code that efficiently computes the integrals. They used dynamic load balancing and obtained performance results on up to 16 processors of the Intel IPSC/2.

We have published several algorithms for fully distributed direct-SCF calculations [5]. The simplest, an *atom-blocked* algorithm, was implemented [6] in an early version of NWChem (version 1.0α) using a simple dynamic task-scheduling technique and a distributed-data toolkit called *Global Arrays* (GA) [7, 8]. Generation of integral blocks and all significant linear algebra operations are performed in parallel. Data communications within the GA library use the most efficient mechanism available for each target machine. A newer algorithm that dynamically blocks over multiple atoms was implemented in the next version of NWChem (version 1.2α). The more general blocking reduces the total communication and is more tolerant of high communication latency. NWChem version 1.2α also includes other code optimizations that improve performance for small molecules and speed convergence.

In this work, we characterize the performance of our original direct-SCF implementation on several MPPs using different data-distribution models and integral-package combinations, with single point energy calculations on several benchmark molecules. These molecules were selected to sample different areas of chemical interests. The characterization will be based on the time to solution and parallel speedup.

The objectives of this benchmark effort include

- understanding the scalability of distributed and replicated direct-SCF algorithms in terms of the hardware characteristics of different computer platforms and architectures,
- investigating the impact of integral evaluation speed on parallel scalability and effective use of parallel computers,
- demonstrating the benefits of distributed-data versus replicated-date algorithms, and
- assessing the performance of differing data-distribution schemes and computer architectures for several kinds of molecules.

The rest of this article is divided as follows. Section 2 describes the algorithm; Section 3 compares the specific MPP machines; Section 4 describes the benchmark molecules and shows the results of the calculations; and Section 5 summarizes results.

2 NWChem SCF Algorithm

The distributed-data Fock matrix construction algorithm within NWChem (version 1.2α) is essentially as described in [6] with several additional algorithmic improvements and an improved version of the quadratically convergent algorithm described in reference [9]. A replicated-data model is also provided within the NWChem package and is useful for small to medium-sized calculations.

Improvements in NWChem (version 1.2α) permitted far better performance on a wider variety of molecules as compared with NWChem (version 1.0α). These improvements include

- dynamic blocking in the distributed-data Fock matrix build similar to that of Furlani and King [4],
- reduced linear algebra overhead, which improves scalability for small systems, and
- more robust convergence resulting from tightening of tolerances, an improved approximation to the matrix exponential, and miscellaneous code optimizations.

We have also included a significantly faster integral generation scheme.

Briefly, the distributed-data algorithm is modeled after that of a blocked matrix-multiply algorithm for a high-performance workstation with a small, fast memory cache. The cost of accessing an element of the distributed density or Fock matrices must be offset by using this element multiple times. To achieve this reuse of data, we stripmine over the fourfold loop of basis function indices. Since the computation is a quartic function of the block size while the communication is only a quadratic function, the block size may be adjusted to ensure that the computation time dominates the communication time.

We previously stripmined by grouping basis functions according to their (usually atomic) centers. This approach had the advantage that the resulting geometrical sparsity can be exploited in the outer stripmining loops. However, the granularity was fixed and was inadequate for calculations on large molecules in small basis sets or for machines with very high latencies. The revised algorithm groups atoms into blocks, looping over quartets of blocks of atoms, and then evaluating all integrals within a quartet of blocks of atoms. The blocksize is maximized subject to the constraint that there are sufficient tasks to ensure good load balance. This form of blocking permits sparsity to still be exploited in the outer loops. The significant increase in overall performance results from a substantially decreased sensitivity to message-passing latency and the ability to vectorize integral computation over many identical shell quartets.

All calculations benefit from the potential vectorization of integral evaluation, but most calculations benefit by less than 10% from the reduction in communication. However, the time to solution (see below) for our ZSM-5 zeolite fragment test case (389 atoms, 2053 function, STO-3G basis, no symmetry) was reduced by nearly a factor of seven [10]. This exceptional reduction resulted because the minimal basis set and high latency of the IBM SPn were inconsistent with obtaining optimal performance with the old algorithm. With the new code, the Fock-build is completely dominated by integral computation and Fock-matrix construction.

3 Hardware and Software Comparisons

3.1 Hardware Comparisons

Five different computers were used in this study: the IBM SP1 in the Mathematics and Computer Science Division at Argonne National Laboratory, the Kendall Square KSR-2 in the Environmental and Molecular Sciences Laboratory at Pacific Northwest National Laboratory, the Intel Touchstone Delta at the California Institute of Technology, and the CRAY T3D and T3E housed at the National Energy Research Supercomputer Center at Lawrence Livermore National Laboratory. These five computers are distributed-memory MPPs.

All of the machines support the message-passing programming paradigm, and on the IBM SP and Intel Delta, it is the only mechanism for communicating data between nodes; the lowest level of the GA library uses interrupt-driven message passing on these machines. The KSR-2, T3D, and T3E also have nonuniform memory-access (NUMA) characteristics in that they both provide mechanisms to directly address both local and nonlocal memory. The KSR-2 provides a uniform address space and dynamically migrates data. The T3D and T3E provides a nonuniform address space and static assignment of data to processors. On these machines GA is implemented using the fastest available shared-memory primitives. The latency and bandwidths reported are those measured for the GA get operation, and may differ signifcantly from other operations (e.g., the common "ping" message-passing benchmark).

Five characteristics of these specific computers are relevant to this study.

1. the number of nodes,
2. the computational power per node in billions (10^9) of floating point operations per second (GFLOPS/s) estimated from the LINPACK report [12],
3. the aggregate memory of the computer in gigabytes (GB),
4. the latency (measured in μsec) of remote memory reference, and
5. the bandwidth (measured in megabytes per second, MB/s) for remote memory reference.

Degradation from peak performance in serial computers is generally a complicated relationship between the number and type of arithmetic operations per clock cycle and the numbers and types of memory references (i.e., local cache, local RAM, disk access, etc.) [13]. The parallel environment can be understood in much the same way by modeling the additional accesses to nonlocal memory and the amount of load imbalance during the calculation.

In Table 1 we compare the five characteristics for the computers used in this study.

3.2 Software Description

For single-point, direct-SCF energy calculation, the program must

Table 1: Capabilities of the computers used in this study. Bandwidth and latency values are for GA remote get operations. CRAY T3D and T3E address memory in words. GFLOPS values estimated from the LINPACK report and refer to 64-bit arithmetic.

Computer	Number of Nodes	Peak GFLOPS/node	Aggregate Memory (Gb)	GA Latency (μsec)	GA Bandwidth (Mb/s)
IBM SP1	128	0.125	16	517	14
KSR-2	80	0.06	2.6	37	28
CRAY T3D	256	0.148	16	21	34
CRAY T3E	128	0.6	32	6	230
Intel DELTA	512	0.039	8	350	8

1. construct an initial guess for the molecular orbitals (MOs),
2. form the density matrix from the current MOs,
3. compute integrals and contract them with the density matrix to form the Fock matrix, and
4. determine an update to the wavefunction and check for convergence.

To update the molecular orbital coefficients, NWChem avoids the use of diagonalization, which historically has been inefficient on massively parallel computers [14], by using instead a preconditioned conjugate-gradient (PCG) algorithm [9] that is a refinement of an approach suggested in this context by one of us [15] and related to the SCF algorithm of Bacskay [16]. In the early iterations, preconditioning is performed with a one-electron approximation to the orbital Hessian (which is related to an approximate diagonaliztion scheme), but once the maximum element of the orbital gradient (or off-diagonal Fock-matrix element) falls below some threshold, preconditioning is performed with the exact orbital Hessian. These equations are solved iteratively to the minimum precision required to maintain quadratic convergence; this approach permits very aggressive screening to be used with no loss of accuracy. The line searches inherent in the conjugate-gradient algorithm are performed to a minimally sufficient accuracy, and additional search steps are typically made in the opening iteration only when very large rotations are being applied. The approximate line search provides stability and indeed a guarantee of (eventual) convergence. Once the quadratically convergent region is entered, no line searches are necessary, though checks are still made to ensure downhill progress.

Explicit diagonalization is still used to canonicalize the orbitals (at the beginning and end of the SCF procedure) and may also be used to render the Hessian more diagonally dominant, reducing the number of micro-iterations. The total number of explicit diagonalizations is typically three. The diagonalization software used in NWChem, called PeIGS [17], executes in parallel and was developed, in part, for this project. The performance of parallel diagonalization has improved greatly in the timeframe of our research project, and elimination of the diagonalization step is no longer a sufficient justification

for the additional complexity of the PCG algorithm. However, the more significant benefit of this approach is to expose the structure of the algorithm and enable 50% or more of the Fock-matrix constructions to be made to low precision. Thus, it is not sufficient to measure the expense of the second-order scheme simply by the number of Fock-matrix constructions, since they may vary by up to twofold in cost.

The calculation of integrals within NWChem may be performed in either a segmented or generalized contracted basis using a McMurchie-Davidson (McMD) algorithm [11], unless only s, p, or L shells are involved, in which case the rotated-axes algorithm of Pople et al. [18] is used. Recently the TEXAS integral package [19] has been incorporated into NWChem, resulting in a marked decrease in the overall time to solution, and permitting analysis of the parallel performance under conditions of much greater relative communications cost.

Several starting guess options are available for NWChem. In all the reported calculations, a superposition of atomic SCF densities is used to construct a Fock matrix that is explicitly diagonalized to form the initial MOs.

4 Benchmarks

In this section, we present the results of single-point direct-SCF energy calculations for seven molecules using the codes and computers described above. The characteristics of these molecules are discussed first, followed by the parallel performance as measured by scalability Ideal scalability has $T(P)$ inversely proportional to P, while deviations indicate that communications are having an impact on the algorithm.

4.1 Molecules

Our benchmark molecules were chosen from the chemistry of polymers, combustion, and catalysts. Each molecule may be characterized by the number of atoms and the number of basis functions. The number of basis functions squared gives the total data storage requirements for each persistent matrix (distributed-data model) or the per-node memory requirements using the replicated-data model. Depending on the type of wavefunction optimization and code, enough memory must be available to store upwards of $\mathcal{O}(10)$ such matrices. Table 2 lists the molecules and their characteristics. The labels listed in the table will be used to refer to specific molecules throughout the rest of this paper. Figures 1 and 2 provide images of all the molecules except C_{60}, whose geometry is well known, and $C_{20}H_{42}$, which has a geometry that is an obvious extension of $C_{10}H_{22}$ in Figure 1. In Figure 2, only the largest molecule, labeled zeolite, is shown. In this molecule, a pyradine impurity, is located in the upper left-hand zeolite channel and is responsible for this channel's being the most constricted in the figure. The atoms in this channel are shown in lighter shading. The Co-cat test case used a 4-31G basis except for Co, which had a minimal basis. Ti-cat used a 6-31G basis for all atoms except Ti, which was of DZP quality. The selected zeolite basis was STO-3G, while the biphenyl and n-alkanes basis sets were of DZP quality. C_{60} was a DZP basis with an added diffuse s-function. Geometries were obtained from varying sources. The n-alkanes were optimized at the

SCF/STO-3G level subject to being in the normal conformation. The zeolite geometry represents the normal zeolite unit cell; Co-cat and Ti-cat were optimized at a minimal basis level. The biphenyl geometry was optimized using the DZP basis. The C_{60} geometry is from a DFT(VWN) geometry optimization procedure.

Table 2: Characteristics of benchmark molecules as described in the text. The data size is the memory required to hold a single square matrix of dimension the number of basis functions. All calculations were performed under a C_1 point group except Co-cat(C_s) and $C_{60}(I_h)$

Label	Formula	Number of Atoms	Number of Basis fns	Data Size (MB)
Co-cat	$(Cp)Co(NO)(CH_3)$	17	114	0.10
Ti-cat	$(Cp)_2(CH_2)Ti(Cl_2)$	24	174	0.24
$C_{10}H_{22}$	$C_{10}H_{22}$	32	260	0.54
Biphenyl	$(C_6H_4CF_3)_2$	28	324	0.84
$C_{20}H_{42}$	$C_{20}H_{42}$	62	510	2.08
C_{60}	C_{60}	60	1020	8.32
Zeolite	$C_5H_5NSi_{108}O_{194}H_{76}$	389	2053	33.72

The wavefunctions were converged to at least 10^{-6} in the energy, except for zeolite (10^{-5}). The minimum integral screening tolerance for all calculations was set to 10^{-10}. All calculations were performed without symmetry except for Co-cat(C_s) and $C_{60}(I_h)$. The results are reported in Table 3 and in Figures 3, 4, and 5.

4.2 Time to Solution

The absolute times to solution reported in this section must be qualified in three ways. First, in an MPP environment, many factors influence the time to solution, for example, the blocking parameters, the angular momentum distribution in the basis set (which influences integral generation time), and the integral screening parameters. The limited number of test problems precludes our ability to distinguish these influences for each molecule. Instead, the seven benchmark molecules are diverse enough so that the overall behavior of the times to solution are representative of performance for general-purpose calculations. Second, results for both versions 1.0α and 1.2α of NWChem will be shown here in order to illustrate behavior on the widest range of MPPs (several MPPs were no longer in operation when version 1.2α was completed). The emphasis is not on the very fastest time to solution tuned for a particular molecule but on how the method performs for a wide range of architectures. Third, new versions of NWChem incorporate modifications that can dramatically reduce the time to solution from that shown here by, among other things, storing subsets of the integrals (e.g., semi-direct methods). Thus, the results are representative of a particular method of obtaining SCF energies, not the only — nor necessarily the fastest — method.

The time in hours for almost all the benchmark molecules using NWChem (version 1.2α) on the SP, T3E, and KSR-2 are listed in Table 3.

Table 3: Time to solution in hours for NWChem (version 1.2α) using 32 nodes on the IBM SP1, CRAY T3E, KSR (KSR times are estimated from 36-node results as $T(36)*36/32$). For C_{60} on IBM SP1, results are estimates from one half the value for 64 nodes.

Molecule	Machine	NWChem v1.2α McMD-integrals	NWChem v1.2α TEXAS-integrals
$(Cp)Co(NO)(CH_3)$	SP1	0.24	0.24
	KSR	1.15	
	T3E	0.07	
	C90		
$(Cp)_2(CH_2)Ti(Cl_2)$	SP1	0.68	0.68
	KSR	1.31	
	T3E	0.26	
	C90		
$C_{10}H_{22}$	SP1	0.77	0.30
	KSR	1.48	
	T3E	0.29	
	C90		
$(C_6H_4CF_3)_2$	SP1	2.06	1.18
	T3E	0.86	
	C90		
$C_{20}H_{42}$	SP1	4.44	1.86
	T3E	1.77	
	C90		
C_{60}	SP1	0.68	
	T3E		
$C_5H_5NSi_{108}O_{194}H_{76}$	SP1	36.72	
	T3E		

Consider first the growth of the time to solution (T) with the number of basis functions N. The entries in the table are ordered with increasing N, which changes from top to bottom by about a factor of 20. The high symmetry of C_{60} makes its T anomolously low. The relatively high T for the transition metal-containing species illustrates the difficulty of convergence and the importance of high angular momentum basis functions in transition metals (the top two entries in the table). By contrast, the third entry in the table, $C_{10}H_{22}$, is relatively low even though N has noticeably increased. Nonetheless, overall, a fit of T to an N^a form reasonably well represents the results in the table, with a ranging from 1.3 to 2.0 depending on the integrals package and the machine. This is much less than the formal N^4 dependence because of screening.

Table 3 also shows that the TEXAS integrals package makes a significant difference for the larger cases but essentially no difference for the two smallest cases. This is an expected result, as the larger cases included relatively high levels of angular momentum.

All the results in the table are for the distributed-data structures. We have also conducted similar experiments using the replicated-date structure models. These runs were made for the first five entries in Table 3 on the IBM SP1 with the TEXAS integrals package. In general, the resulting times to solution are smaller but with only a less than 20% scatter about the average of the distributed and replicated values of T. In one sense, the development of a distributed-data algorithm is of no consequence in doing the first five entries of the table. However, replicated-data model is inherently not scalable. The last entry in the table has data structures too large to be replicated on the SP1 used in this work. In this sense, the distributed data algorithm incurs no penalty in handling small problems, while at the same time becomes the only method for handling large problems.

In Figure 3, the dependence of T on P is displayed for six of the seven benchmark molecules on the SP1. Both results for the TEXAS and McMD methods are presented. The following four features are apparent. First, T is nearly inversely proportional to P, hence resulting in straight lines of slope near -1.0 in the figure. This is equivalent to nearly ideal values of S(P) over the range of P displayed and is largely independent of the integral method.

Second, for either integral method, the smallest molecule, Co-cat, is too small to be efficiently computed on moderately large P due to an insufficient amount of work for efficient processing on large numbers of nodes. In effect the calculation is limited by communications and load imbalances. The decay of S(P) from ideal for Co-cat can be ameliorated by carrying out the calculations on computers with faster communications relative to computational power. This will be discussed shortly. The decay cannot be improved by a replicated data mode for NWChem. Although not shown in the figure, T in this mode is quite similar to the results in the figure for all molecules. This similarity indicates that while the pattern of communications is very different for the two modes, its effect on performance for smaller problems at larger values of P is about the same.

Third, the major effect of the TEXAS integrals package is a reduction of T that is largely independent of P. The reduction is largest for basis sets with lower angular momentum components. The fourth and last feature in the figure is the surprisingly large flattening out of T(P) for large P in the case of biphenyl for both integrals packages. This trend is not observed for either $C_{10}H_{22}$ or $C_{20}H_{42}$, which bracket in basis set size the biphenyl case. Note that biphenyl, being more compact that $C_{10}H_{22}$ and $C_{20}H_{42}$, which are linear chains, is less affected by screening. This fact implies that adjustment of the blocking parameters as a function of P may influence performance. Computational experiments with biphenyl demonstrated some sensitivity to screening, but no thorough study was carried out in this direction.

In Figure 4, T is plotted with respect to P for two different MPPs. In both cases, the McMD integrals package is used. As the figure clearly shows, there is a generally uniform factor of about 2.5 reduction in T(P) from the SP1 to the T3E. This is due to the factor of almost 4 increase in peak FLOPS/node in going from the SP1 to the T3E (see Table 1). A secondary trend is that T(P) does not flatten out so readily at large P on the T3E. This is especially apparent on the smallest molecule, Co-cat. However, the onset, if the not the

degree, of nonideal scaling for Co-cat occurs for both MPPs at about the same number of processors, suggesting that load imbalance is also a significant effect. All calculations would eventually show the same behavior on large enough numbers of processors, but since the number of tasks is proportional to at least the square of the number of basis functions, relatively modest increases in the number of atoms in all the other benchmark molecules over that of Co-cat are large enough to more efficiently exploit the parallelism of the largest available MPPs. NWChem on the T3E, with the highest bandwidth, is least affected by the overall communications and, hence, performs better in the intermediate processor count region where deviation from ideal becomes significant.

In Figure 5, T(P) is displayed for only two molecules, Co-cat and biphenyl. These calculations are for NWChem (version 1.0α), an earlier version first discussed in [6] that was the only version available to run on the Intel DELTA and the CRAY T3D. Results for these two machines are shown in the figure along with results for the SP1. All calculations in the figure were done with the McMD integrals package available in version 1.0α. In conjuction with Figure 4, this figure indicates that only the SP1 is severely limited by communications for the Co-cat problem. The T3E, T3D, and DELTA certainly differ in the magnitude of T(P) as a result of changes in the GFLOPS rating but tend to show ideal speedup to larger values of P because of superior communications relative to computation speed. However, as discussed with regard to the previous figure, all MPPs tend to show deviation from the ideal at approximately the same number of processors. The Fock matrix build of NWChem (version 1.0α) is executed in $O(N_{atom}^2)$ tasks in a fixed blocking over atoms. Therefore, load imbalance will become apparent at a comparable number of processors on all MPPs. Direct plots of S(P) for the Fock matrix build alone on all the rest of the benchmark molecules not shown in the figure confirm this universal behavior. In version 1.2, there is dynamic blocking, and the universal behavior is not as obvious, but still quite apparent for Co-cat, as seen in the previous figure.

Figures 4 and 5 allow a comparsion in performance of the version 1.0α and 1.2α on the SP1. The newer version is faster on each problem by a factor of about 1.7 and exhibits nearly ideal speedup to higher values of P. This speedup is a reflection of tighter tolerances, dynamic blocking in data distribution, and reduced linear algebra overhead in the newer version.

The results in all the figures are largely dominated by the Fock matrix build. However, NWChem can be run on the T3D for the medium-sized molecules in the figures in a replicated data mode with all integrals computed once and cached in-core. This is representative of the limiting case of very fast integral evaluation and therefore tests the scalability of the entire code, rather than just the Fock matrix build. The nearly ideal speed-up found in the figures for medium size molecules is retained, indicating that NWChem is pervasively scalable.

As discussed earlier, the absolute timings reported here for NWChem for these earlier versions are subject to many factors, not the least of which is that NWChem, and its tuning, has evolved. However, the times presented here are reasonably fast, and the focus is on the performance of the algorithm on distributed-memory architectures. The results demonstrate that the general scalability of NWChem and its distributed nature have not come at the price of rapid time to solution.

5 Conclusions

The performance of a fully distributed, parallel direct-SCF algorithm [6] has been characterized on five MPP computers using single-point energy calculations on seven molecules of widely varying size and composition. As described here and elsewhere, the NWChem code has four features particularly motivated by the desire for full data distribution and parallelism: (1) the GA library for efficient sharing of distributed data, (2) blocking of the integral computation over groups of atoms, (3) fully dynamic load-balancing, and (4) a second-order convergent wavefunction optimization scheme with variable screening tolerances.

The benchmark results from the DELTA, T3D, T3E, SP1, and the KSR-2 computers show that the larger benchmark molecules (i.e., those with basis sets of more than several hundred functions) scale well with NWChem. However, for smaller benchmark molecules, the decreased number of tasks favor the use of fewer processors.

The NWChem performance is noticeably improved with the faster TEXAS integrals, and scalability is only slightly degraded for large molecules, as shown in Figure 3. The nearly ideal speedup of the fully-in-core calculation on the T3D demonstrate that NWChem will fully exploit fast integral evaluation methods, and continued improvements in integral computation technology are being developed. More recent versions of NWChem have been further enhanced by incorporation of semi-direct methods using advanced I/O capabilities.

The most important advantage of a fully distributed code is that the problem size is restricted only by the *aggregate* memory of the MPP. Consider, for example, the calculation on the ZSM-5 cluster using the Intel DELTA: it is not possible to hold even one lower-triangular matrix on one node. This aggregate memory is almost always larger, and certainly more cost effective, than the memory of even the largest shared-memory computers.

Acknowledgments

This work was supported through the U.S. Department of Energy by the Mathematical, Information, and Computational Science Division of the Office of Computational and Technology Research; by the Chemical Sciences Division of the Office of Basic Energy Sciences; and by the Office of Health and Environmental Research, which funds the Pacific Northwest Laboratory Environmental Molecular Sciences Laboratory Project D-384. This work was performed under contract W-31-109-Eng-38 (Argonne National Laboratory) and under contract DE-AC06-76RLO 1830 with Battelle Memorial Institute (Pacific Northwest National Laboratory).

This research was performed in part using the Intel Touchstone Delta System operated by Caltech on behalf of the Concurrent Supercomputing Consortium. Access to this facility was provided by Argonne National Laboratory.

The authors gratefully acknowledge use of the Argonne Center for Computational Science and Technology.

References

[1] Almlöf, J.; Faegri, K.; Korsell, K., Principles for a Direct SCF Approach to LCAO-MO Ab-Initio Calculations. *J. Comp. Chem.* 1982, 3, 385–399.

[2] Harrison, R. J.; Shepard, R. Ab Initio Molecular Electronic Structure on Parallel Computers. *Annu. Rev. Chem.* 1994, 45, 623–658.

[3] Colvin, M. E.; Janssen, C. L.; Whiteside, R. A.; Tong, C. H. Parallel Direct SCF for Large-Scale Calculations. *Theor. Chim. Acta.* 1993, 84, 301–314.

[4] Furlani, T. R.; King, H. F. Implementation of a Parallel Direct SCF Algorithm on Distributed Memory Computers. *J. Comp. Chem.* 1995, 16, 91–104.

[5] Foster, I. T.; Tilson, J. L.; Shepard, R.; Wagner, A. F.; Harrison, R. J.; Kendall, R. A.; Littlefield, R. L. Toward High-Performance Computational Chemistry: I. Scalable Fock Matrix Construction Algorithms. *J. Comp. Chem.*, 1995, 17, 109–123.

[6] Harrison, R. J.; Guest, M. F.; Kendall, R. A.; Bernholdt, D. E.; Wong, A. T.; Stave, M.; Anchell, J.; Hess, A. C.; Littlefield, R. L.; Fann, G. L.; Nieplocha, J.; Thomas, G. S.; Elwood, D.; Tilson, J. L. Shepard, R.; Wagner, A. F.; Foster, I. T.; Lusk, E.; Stevens, R. Toward High Performance Computational Chemistry: II. A Scalable SCF Program. *J. Comp. Chem.*, 1995, 17, 124–132.

[7] Harrison, R. Moving beyond Message Passing. Experiments with a Distributed-Data Model. *Theo. Chim. Acta.* 1993, 84, 363–375.

[8] Nieplocha, J.; Harrison, R. J.; Littlefield, R. J. Global Arrays: A Portable "Shared-Memory" Programming Model for Distributed Memory Computers. Institute of Electrical and Electronics Engineers and Association for Computing Machinery IEEE Computer Society, Los Alamitos, *Supercomputing'94*, 1994

[9] Wong, A. T.; Harrison, R. J. Approaches to Large-scale Parallel Self-Consistent Field Calculations. *J. Comp. Chem.* 1995, 16, 1291–1300.

[10] Tilson, J. L. Massively Parallel Self-Consistent-Field Calculations. *I&EC RESEARCH*, 1995, 34, 4161–4165.

[11] McMurchie, L. E.; Davidson, E. R. One- and Two-Electron Integrals over Cartesian Gaussian Functions. *J. Comp. Phys.*, 1978, 26, 218–231.

[12] Dongarra, J. J. Linpack Benchmark, http://performance.netlib.org/performance/html/linpack-peak.data.col0.html, 1998.

[13] Dowd, K. In *High Performance Computing.*, O'Reilly and Associates. Inc. 1995.

[14] Littlefield, R.; Maschhoff, K. Investigating The Performance of Parallel Eigensolvers for Large Processor Counts. *Theor. Chim. Acta* 1993, 84, 457–473.

[15] Shepard, R. Elimination of the Diagonalization Bottleneck in Parallel Direct-SCF Methods. *Theor. Chim. Acta* 1993, 84, 343–351.

[16] Bacskay, G. B. A Quadratically Convergent Hartree-Fock (QC-SCF) Method. Application to Closed Shell Systems. *Chem. Phys.* 1981, 61, 385–404.

[17] Fann, G. I.; Littlefield, R. J.; Elwood, D. M. Performance of a Fully Parallel Dense Real Symmetric Eigensolver in Quantum Chemistry Applications., In *High Performance Computing 1995*, Grand Challenges in Computer Simulation. Tentner, A., ed.; The Society for Computer Simulation, 1995, pp. 329–336.

[18] Pople, J. A.; Hehre, W. J. Computation of Electron Repulsion Integrals Involving Contracted Gaussian Basis Functions. *J. Comp. Phys.* 1978, 27, 161–168.

[19] Wolinski, K.; Pulay, P.;Hamilton, T. Integral package from TEXAS 95 extended for use within NWChem including analytic gradients by Harrison, R. J.; Kendall, R. A.; and Wolinski, K. 1997.

[20] Pulay, P. Convergence Acceleration of Iterative Sequences. The Case of SCF Iteration. *Chem. Phys. Lett.* 1980, 15, 393–398.

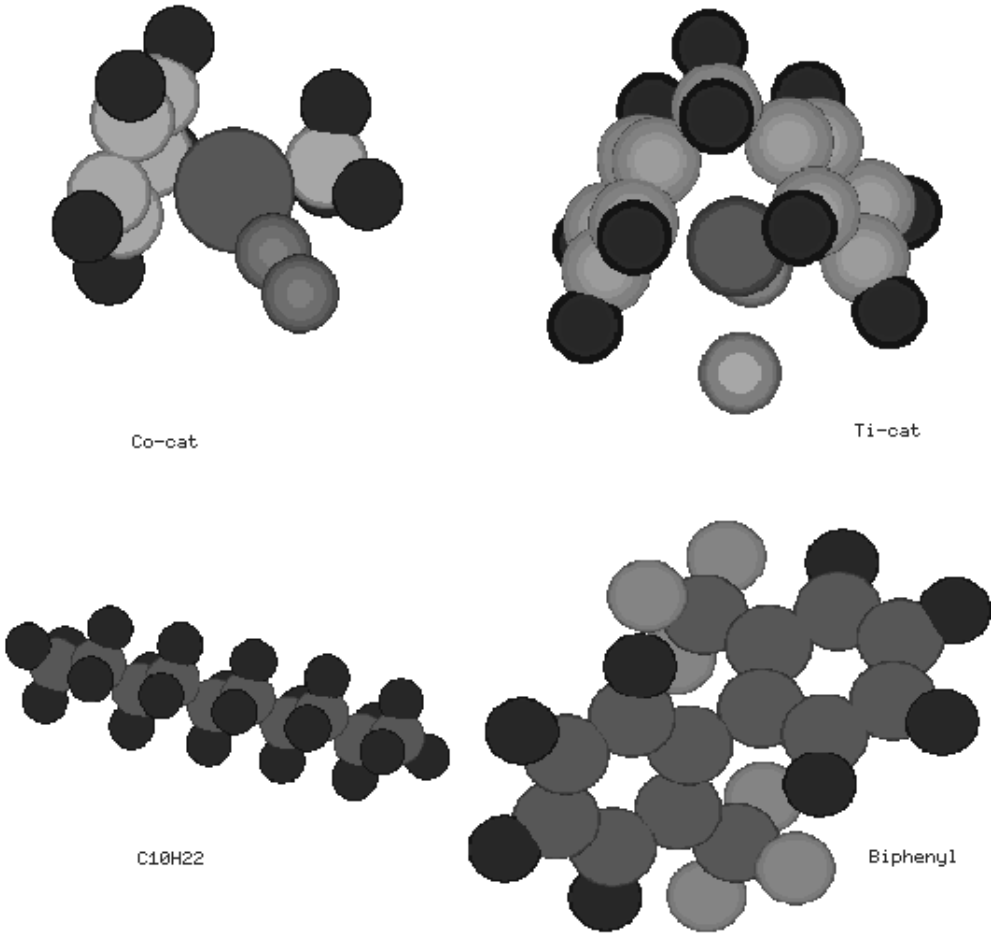

Figure 1: Figures of the Co-cat, Ti-cat, $C_{10}H_{22}$, and biphenyl molecules.

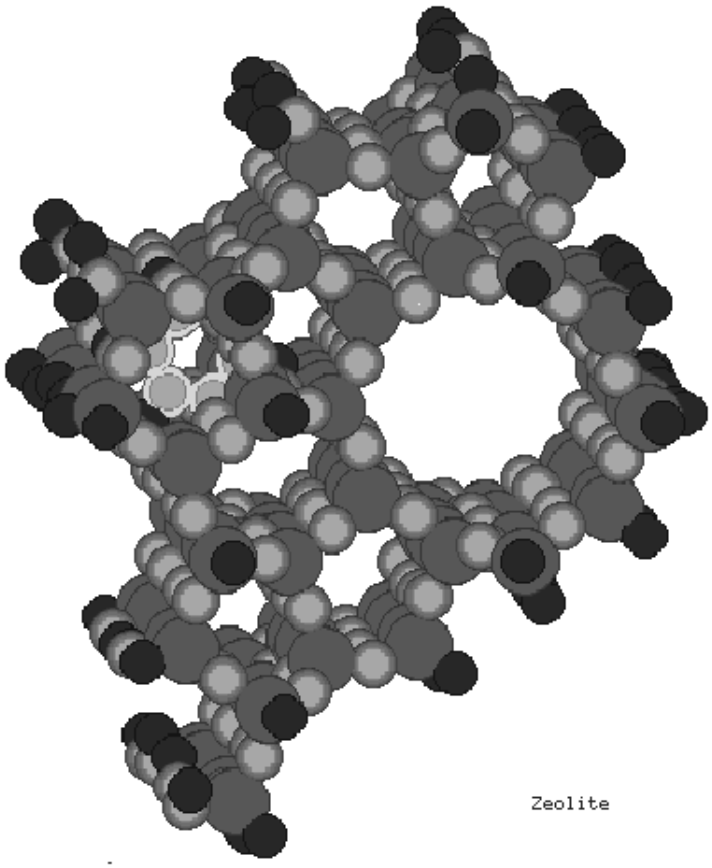

Figure 2: Zeolite molecule.

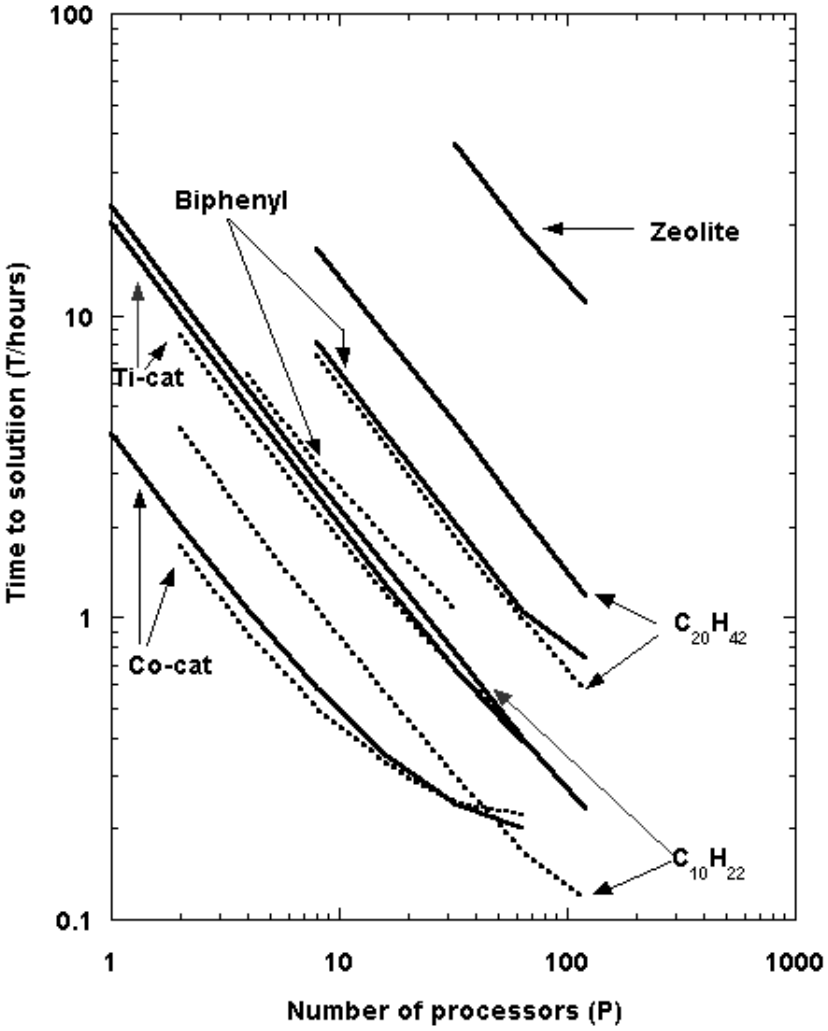

Figure 3: Time to solution (hours) versus number of processors (P) for NWChem v1.2α with McMD and TEXAS integrals packages on the SP1. TEXAS results are indicated with dashed lines.

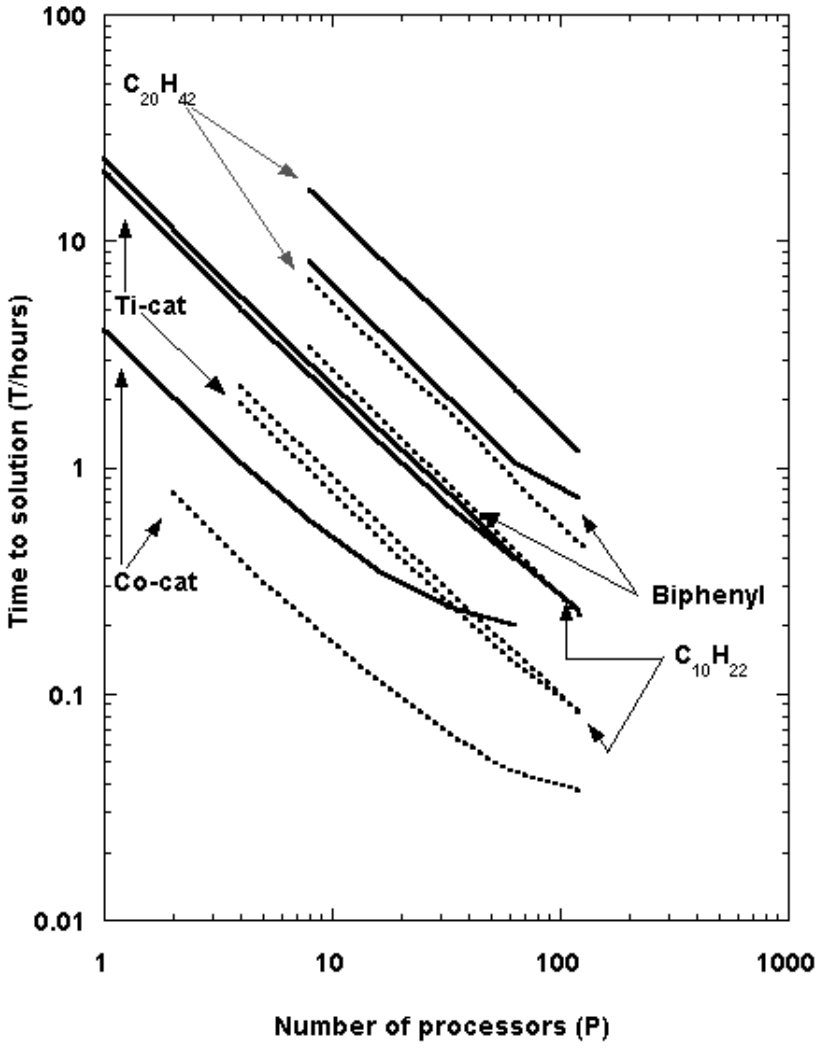

Figure 4: Time to solution (hours) versus number of processors (P) for NWChem v1.2α with the McMD integrals package on the SP1 and T3E. T3E results are indicated with dashed lines.

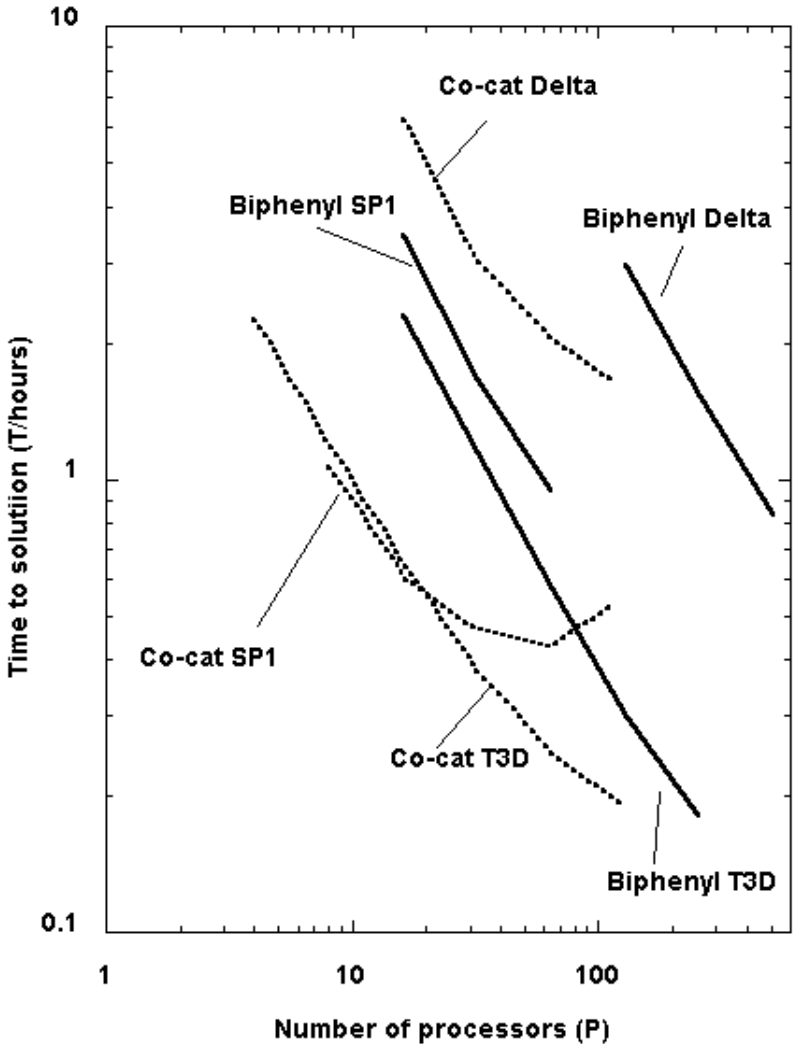

Figure 5: Time to solution (hours) versus number of processors for NWChem v1.0α with McMD integrals package on the SP1, T3D, and DELTA.