REPORT DOCUMENTATION PAGE

AFRL-SR-BL-TR-98-
0602

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	15 April 98	Final Technical Report 15 Jan 96- 14Jan 98

4. TITLE AND SUBTITLE
Optimization of Geometric Structures of New Materials on Parallel Computer: Final Project Report

5. FUNDING NUMBERS
Grant No.: F49620-94-1-0286
Amend: P00003
Project-Task 2303/FS
61102F

6. AUTHOR(S)
PI: John H. Weare, UCSD, Chemistry & Biochemistry 0340
Co-PI: Ryoichi Kawai, UAB, Physics
Co-PI: Beth Ong, UCSD, Math

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
The Regents of the University of California
University of California, San Diego
9500 Gilman Drive
La Jolla, CA 92093-0340
AFOSR's CFDA #12.800

8. PERFORMING ORGANIZATION REPORT NUMBER
31807A

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AFOSR/NL
Dr. Michael R. Berman, Program Manager
110 Duncan Avenue, Room B115
Bolling AFB, DC 20332-8050

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION / AVAILABILITY STATEMENT
This document is approved for public release, its distribution is unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT ***(Maximum 200 Words)***

Ab inito molecular dynamics (AIMD) simulation code based on the planewave pseudopotential local density functional method have been developed. With the new codes, it is possible to simulate geometric, dynamical and electronic properties of molecules and polymers containing more than 200 atoms from first principle. In addition, they allow the use of both periodic and freespace boundary conditions. Extended systems such as polymers can be simulated with periodic boundary conditions, whereas finite-size systems including charged systems can be calculated with freespace boundary conditions. Various benchmark tests demonstrate the high degree of accuracy and efficiency of parallel processing. This code has been applied to the simulation of semiconducting polymers with large repeat cells.

The most significant problem with the application of AIMD to a wider variety of high performance materials is the poor convergence of the planewave basis that is used in all the present methods. For a system that have 2^{nd} period elements nitrogen, oxygen and flourine or the transition metals, AIMD calculations based on plane waves are very inefficient. We have been developing a new method which is based on the direct descretation of space. There are a number of difficulties with the application of this approach, but we believe that we have finally identified a correct path to an efficient algorithm.

14. SUBJECT TERMS

15. NUMBER OF PAGES
18 pages

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

Optimization of Geometric Structures of New Materials on Parallel Computers: Final Report

John H. Weare, PI, UCSD, Chemistry Ryoichi Kawai, CoPI, UAB, Physics
Beth Ong, CoPI, UCSD, Math

1 Summary

Ab initio molecular dynamics (AIMD) simulation codes based on the planewave pseudopotential local density functional method have been developed. The new planewave based code uses significantly smaller memory and disk space. It is now written in CM Fortran and therefore can be run on a number of different massively parallel platforms with minimal change. We have also developed another version based on MPI. This code is running on the SP2 and will be running on the T3E here at the San Diego Supercomputer Center.

With the new codes, it is possible to simulate geometric, dynamical, and electronic properties of molecules and polymers containing more than 200 atoms from first principle. In addition, it allows the use of both periodic and freespace boundary conditions. Extended systems such as polymers can be simulated with periodic boundary conditions, whereas finite-size systems including charged systems can be calculated with freespace boundary conditions. Various benchmark tests demonstrate the high degree of accuracy and efficiency of parallel processing. This code has been applied to the simulation of semiconducting polymers with large repeat cells.

The most significant problem with the application of AIMD to a wider variety of high performance materials is the poor convergence of the planewave basis that is used in all the present methods. For systems that have 2nd period elements nitrogen, oxygen and flourine or the transition metals the atom scattering potentials are very strong causing a rapid variation in the wavefunction and requiring that small wavelengths be included in the basis. This makes AIMD calculations very inefficient. We have been developing a new method which is based on the direct descretation of space (vs. descretation of momentum space). There are a number of difficulties with the application of this approach, but we believe that we have finally identified a correct path to an efficient algorithm.

2 Introduction

For large systems of the structural and compositional complexity that is common to high performance materials the most reliable and efficient approach to first principles calculation

is based on the local density approximation (LDA) [10]. In this approach, the total energy of electrons interacting with the nuclei is described as a functional of electron density and coordinates of the nuclei. Electronic orbital wavefunctions are given by the solutions of the generalized eigenvalue equations:

$$\mathcal{H} \mid \psi_i\rangle = \sum_j \lambda_{ij} \mid \psi_j\rangle \tag{1}$$

where the Hamiltonian is given by

$$\mathcal{H} = \left(\frac{p^2}{2m} + V_{ext} + V_H + V_{xc}\right) \tag{2}$$

and λ_{ij} is a Lagrange's multiplier which maintains the orbital orthonormality constraints, $\langle\psi_i \mid \psi_j\rangle = \delta_{i,j}$. Electron-electron interaction is included in the Hartree potential, V_H, and the exchange correlation potential, V_{xc}. Since V_H and V_{xc} are functionals of the electron density, Eq.(1) must be solved self-consistently.

In order to further implement the theory, it is necessary to expand ψ in a basis set. Usually this basis is chosen with the particular physics of the problem in mind. For example, a natural choice of basis functions would be planewaves for metallic systems. This choice seems inappropriate for nonmetals. For these systems we are developing a new method based on a very localized finite element expansion of the wavefunction. However, for any choice of basis it is important to design the method of solution to match the architecture of the computer. In the following we discuss our progress with both approaches.

3 Accomplishment: Direct minimization of the Kohn-Sham equations using preconditioned conjugate gradients methods

In this project we compared various methods used in the direct minimization of the Kohn-Sham equations. The objective is to compute the lowest energies (eigenvalues) and their corresponding wave functions (eigenvectors) using a planewave basis set. Several variants of preconditioned conjugate gradients are compared. They are distinguished by:

1. the basic iterative method or minimization scheme: steepest descent (SD) versus conjugate gradients (CG)

2. the preconditioner: the preconditioner of Teter, Payne, and Arias (TPA)[15] versus the multilevel nodal basis (MNB)[1] preconditioner. The use of a preconditioner accelerates the convergence of the basic iterative method by first preprocessing the Hamiltonian in the Kohn-Sham equations by its approximate inverse. The Hamiltonian consists of the kinetic energy and potential energy terms. TPA is basically an approximation of the inverse of the kinetic energy term in LDA on the finest/final grid, done in Fourier

space. MNB is an approximation of the inverse of the kinetic energy term on different grid levels, done in either Fourier space or real space. Thus, both preconditioners will work well provided the kinetic energy term remains dominant. The effect of a multilevel approximation to the kinetic energy term (as in MNB) is to annihilate the effect of the mesh size. Thus, MNB is expected to perform better as the mesh size diminishes; that is, as more grid points are used.

3. band-by-band (BB) versus whole band minimization (WB). Band-by-band minimization is a procedure whereby the lowest energies and corresponding wave functions are computed one at a time, beginning with the lowest energy, followed by the second, and so on. The ith wave function is made orthogonal to the previous $i-1$ wave functions. Whole band minimization computes all lowest energies and wave functions simultaneously. At the end of the procedure, the wave functions are orthogonalized through a Gram-Schmidt process.

4. exact energy (EE) versus inexact energy (IE) calculation. Exact energy calculation is when the energy is evaluated using the new iterates of the wave functions during all steps of the computation of the new iterates of the wave functions. Inexact energy calculation is when the energy is calculated using the current iterate of the wave functions during the line search in a particular CG direction. This has the effect of freezing the Hamiltonian in the LDA at the current iterates of the wave functions.

We compare the following methods

1. Method CG, preconditioner MNB, WB minimization, IE calculation
2. Method CG, preconditioner TPA, WB minimization, IE calculation
3. Method CG, no preconditioner, WB minimization, IE calculation
4. Method CG, preconditioner MNB, WB minimization, EE calculation
5. Method CG, preconditioner TPA, WB minimization, EE calculation
6. Method CG, no preconditioner, WB minimization, EE calculation
7. Method CG, preconditioner TPA, BB minimization, IE calculation
8. Method CG, no preconditioner, BB minimization, IE calculation
9. Method SD, no preconditioner, WB minimization, IE calculation

In Table 1, we show the ratios of the CPU times for these different methods for Li_n, where n is the number of atoms, N_g^3 grid points with uniform mesh size, and random initial wave functions. Plane wave basis and fast Fourier transform are used. The CPU time ratios are normalized so that method 1 has value 1.

Method	$Li_{10}, N_g = 32$	$Li_{20}, N_g = 32$	$Li_{40}, N_g = 40$	$Li_{40}, N_g = 64$
1	1.000	1.000	1.000	1.000
2	0.788	1.357	fails	fails
3	2.442	2.863	2.647	3.640
4	8.784	8.250	6.477	7.216
5	5.299	4.980	4.002	4.505
6	11.178	8.553	6.097	10.007
7	7.524	14.692	21.183	no data
8	33.230	51.263	no data	no data
9	13.299	9.508	no data	no data

CPU time ratios

In general,

1. CG performs better than SD
2. WB minimization performs better than BB minimization
3. IE calculation reduces computation time dramatically compared with EE calculation
4. CG with preconditioner performs better than without preconditioner
5. MNB preconditioner performs better than TPA, especially as the number of atoms and grid size increase.

Of the nine methods, method 1 (the CG method with MNB preconditioner, WB minimization, and IE calculation) performs best.

4 Accomplishment: Adaptive Mesh Implementation

The primary disadvantage of planewave methods is that they do not readily support the adaptivity needed to represent the various length scales present in many materials applications. Ideally, our basis set should adapt to local changes in the electronic charge density, such as near atomic centers. Planewave basis functions uniformly cover the entire computational domain and therefore preclude localization.

However, structured adaptive mesh refinement techniques in real space have been shown to efficiently capture the multiple length scales and localized singularities for simple model systems [2]. An adaptive method nonuniformly places computational effort and memory in those portions of the problem domain with the highest error; thus, adaptive codes can target systems that are difficult or infeasible to solve with the planewave approach.

We have developed a prototype LDA/LSD code based on adaptive mesh refinement methods using a finite element basis set. Computational results for some simple diatomic systems are presented below. Our adaptive implementation is not yet competitive with the more mature planewave methods; however, we have identified changes that will improve the accuracy and efficiency of the adaptive approach. This work has been recently been published in two articles [7, 8]

4.1 Finite Element Discretization

We discretize the Kohn-Sham equations using finite element techniques, which has certain advantages over competing discretization methods. Finite elements readily admit local adaptivity. Finite element basis functions are very localized in space, interacting only with their immediate neighbors, and therefore do not suffer from the scaling problems of LCAO methods that use Gaussian basis sets. Discretization approaches such as finite differences or finite volumes do not provide a consistent framework for defining operators on adaptive grid hierarchies, resulting in nonsymmetric operators and complex Kohn-Sham eigenvalues.

The finite element approach expands the wavefunctions ψ in a basis set $\{\phi_i\}$:

$$\psi(x) = \sum_{i=1}^{N} c_i \phi_i(x)$$

and the Kohn-Sham equations are discretized using a Ritz formulation, resulting in the following nonlinear eigenvalue problem:

$$\frac{1}{2}\int \nabla\phi_i(x)\nabla\psi(x) + \int \phi_i(x)\psi(x)V(x) = \epsilon \int \phi_i(x)\psi(x), \quad i = 1, \ldots, N.$$

Note that we have shown only one wavefunction ψ to simplify the notation; the full Kohn-Sham equations involve a set of ψ coupled through the charge density and V. Our current code uses a 3d trilinear basis element ϕ_i and approximates the rightmost two integrals in the above equation using the mid-point integration rule.

4.2 Structured Adaptive Mesh Refinement

Traditionally, finite element calculations have been implemented using *unstructured* data types (e.g., a graph), so called because the data representations do not exploit local structure in the adaptive mesh. Connectivity information must be stored for each unknown at greatly increased cost in memory overheads. Furthermore, unstructured methods make poor use of memory and cache locality and typically do not parallelize well since connectivity information must be distributed across processor memories.

The basic idea behind *structured* adaptive mesh refinement is that if one element requires refinement, then it is likely that neighboring elements will be refined also. Thus, we can exploit this localized structure to reduce memory overheads and improve performance. Structured adaptive mesh refinement methods represent partial differential equations using a hierarchy of nested, locally structured grids. All grids at the same level of the hierarchy have the same mesh spacing, but successive levels have finer spacing than the ones preceding it, providing a more accurate representation of the solution (see Figure 1).

Instead of storing connectivity information for each unknown, structured methods store connectivity information for each refinement patch, which in turn may contain many thousands of unknowns. Refinement patches can be represented in a few tens of bytes; thus, in parallel implementations, structure information is replicated across processor memories, improving parallel performance.

Numerical computations on structured adaptive meshes consist of efficient array-based calculations on refinement patches and "fix-up" computations on the boundaries of the patches. In our particular application, the time spent on boundary computations is only about 10-20% of the time spent on patch interiors. Finally, a structured representation enables us to use highly efficient multilevel solvers such as the FAC (Fast Adaptive Composite) multigrid method [12] (see Section 4.4.1).

4.3 Parallel Software Support

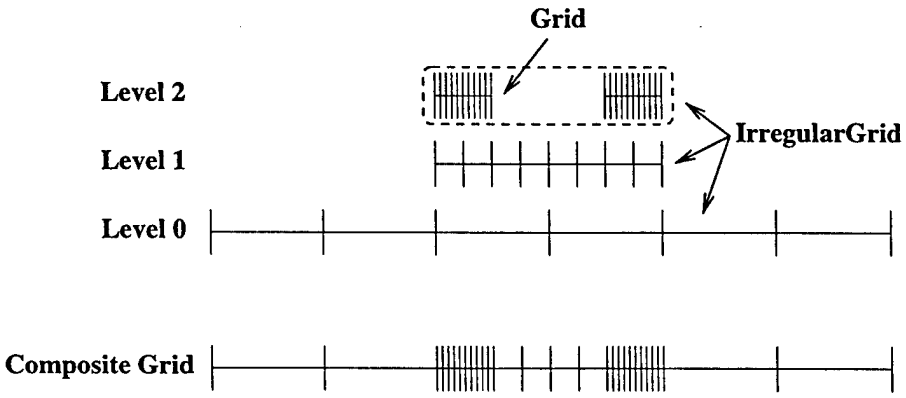
Adaptive mesh methods are difficult to implement on parallel architectures because they rely on dynamic, complicated data structures with irregular communication patterns. On parallel platforms, the programmer is burdened with the responsibility of managing data distributed across processor memories and orchestrating interprocessor communication and synchronization. Refinement regions vary in size and location in the computational space, resulting in complicated geometries. Communication patterns between grid patches and between grid levels are irregular and change as the hierarchy is modified. Because adaptive mesh applications change in response to the dynamics of the problem (e.g., as atoms move during structure optimization), little can be known about the structure of the computation at compile-time. Thus, decisions about data decomposition, the assignment of work to processors, and the calculation of communication patterns must be made at run-time. These implementation difficulties soon become unmanageable and can obscure the mathematics behind the algorithms.

To simplify the development of structured adaptive mesh applications, we have developed an object-oriented adaptive mesh software framework in C++ that provides computational scientists with high-level support for structured adaptive mesh applications. Our framework manages mundane details such as interprocessor communication, parallel execution, load balancing, and grid generation (see Figure 1). We have based our adaptive mesh software on previous work by Kohn and Baden (Department of Computer Science and Engineering, University of California at San Diego) [9]. Our framework allows scientists to concentrate on the high-level expression of mathematical methods rather than being concerned with the underlying parallel implementation details. It enables us to run our applications on a variety of parallel performance computers, including the CRAY T3D, ASCI Blue Machine, IBM SP2, architectures supporting MPI, and networks of workstations.

In addition to simplifying the code development in our own project, our software framework and numerical methods are applicable to other fields of science and engineering of interest to the Air Force where it is important to track localized physical phenomena with high accuracy. We are currently talking with applications scientists at Lawrence Livermore National Laboratories about applying our adaptive software technology to the solution of multiscale problems in computational fluid dynamics and crack propagation.

4.4 Numerical Solvers

The adaptive real-space method is sufficiently different from the planewave approach that new types of numerical algorithms are required. For example, the Hartree calculation (needed to

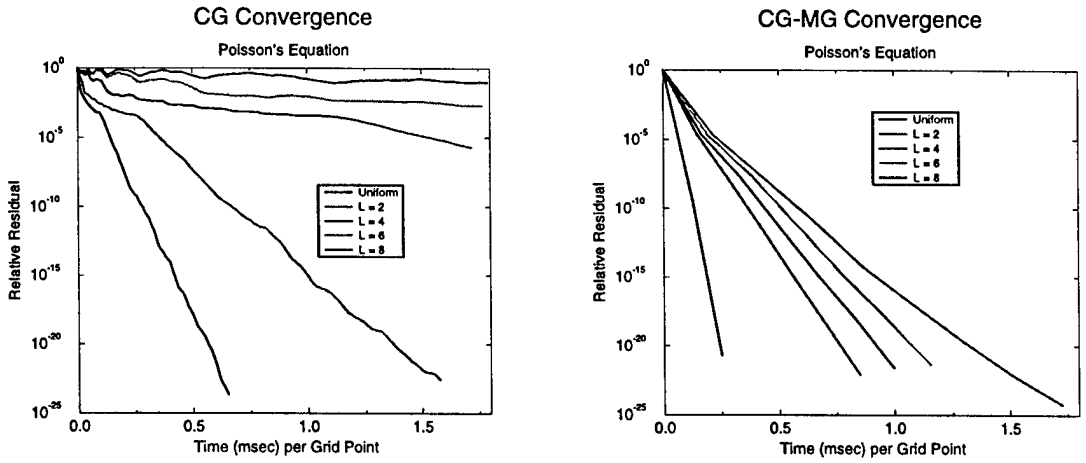


Data Type	Description
`Grid`	`Grid` represents a single refinement patch in the adaptive grid hierarchy. `Grid` computations are typically performed in serial numerical routines.
`IrregularGrid`	`IrregularGrid` represents one level in the adaptive mesh hierarchy. `Grids` in an `IrregularGrid` are distributed across processors, and applications compute over these `Grids` in parallel. `IrregularGrid` provides communication routines to fill boundary cells for `Grids` at the same level of refinement.
`CompositeGrid`	`CompositeGrid` represents the entire adaptive mesh hierarchy. It provides mechanisms to communicate between levels and to create new refinements through error estimation, grid generation, and load balancing.

Figure 1: Our object-oriented adaptive mesh refinement framework represents the structured adaptive hierarchy (a composite grid) using three basic classes: a `Grid`, an `IrregularGrid`, and a `CompositeGrid`. A `CompositeGrid` consists of `IrregularGrid` objects organized into levels. Each `IrregularGrid` is a collection of `Grids`.

compute the $V(x)$ term in the Kohn-Sham equations) is trivial in Fourier space but not straight-forward in real space. In this Section, we describe some of the algorithmic advances we have made in implementing our adaptive solver.

One of the difficulties of the adaptive approach is that the conditioning of the Kohn-Sham equations is dependent on the number of levels of refinement in the adaptive mesh hierarchy. As shown in Figure 2a, iterative methods such as unpreconditioned conjugate gradients require twice as many iterations to converge with each new level of adaptive refinement (assuming a mesh refinement factor of two). Typical adaptive mesh computations such as the ones presented in Section 4.5 need between two and four levels of adaptive refinement, resulting in between two and sixteen times more iterations for a naive solver. Thus, practical and efficient implementations of the adaptive method require more sophisticated numerical algorithms.



(a) (b)

Figure 2: These graphs compare the convergence of (a) an unpreconditioned conjugate gradient method with (b) a multigrid-preconditioned conjugate gradient solver as the number of levels of adaptive refinement is varied. Preconditioning becomes more important with increasing levels of refinement.

4.4.1 FAC Multigrid

The multigrid method is a highly efficient and practical solver for many elliptic partial differential equations. Multigrid is optimal in the sense that it converges in a constant number of iterations independent of the size and conditioning of the linear system of equations.

We have implemented a multigrid preconditioner to accelerate the solution of the Hartree problem. We use a variant of multigrid for structured adaptive mesh hierarchies called FAC (Fast Adaptive Composite) [12]. The advantage of FAC over competing adaptive multigrid methods is that it provides a consistent framework for defining the composite grid operator at interfaces between fine and coarse grids.

Figure 2b illustrates the performance of our Hartree solver with the FAC preconditioner. (Although we could use FAC by itself without CG, the conjugate gradient wrapper provides some extra stability to the iterative solver.) Preconditioning significantly reduces the time to solution, especially for adaptive mesh hierarchies with many levels of refinement. For example, for an adaptive mesh with six levels, the FAC solver reduces the Hartree residual by more than twenty orders of magnitude in the same time that the standard conjugate gradient method reduces it by only two orders of magnitude.

4.4.2 Rayleigh Quotient Minimization

The same types of condition number scaling described in the previous Section for the Hartree equation also apply to the Kohn-Sham eigenvalue problem. A naive iterative method such

as steepest descent would require too many iterations to converge for the adaptive approach. Therefore, we use an eigenvalue solver technique developed by Longsine and McCormick called Simultaneous Rayleigh Quotient Minimization with Subspace Diagonalization [11].

The basic idea behind this approach is to take iterative steps that minimize the Rayleigh Quotient:

$$RQ(\psi) = \frac{\int \psi \mathcal{H} \psi}{\int \psi \psi}$$

where $\mathcal{H}$ is the Hamiltonian of the Kohn-Sham equations. At each step of the algorithm, we choose one wavefunction ψ_i and take a step $\psi_i^{new} \leftarrow \psi_i + \alpha d$, where α minimizes the Rayleigh Quotient for that wavefunction:

$$\min_{\alpha} RQ(\psi_i + \alpha d).$$

If we assume that the Hamiltonian operator is approximately linear about the location ψ_i, then we can compute the step size α efficiently without a nonlinear search. The step directions d are generated via a CG-like process.

4.5 Computational Results

To validate the adaptive mesh refinement approach, we have applied our adaptive techniques to some simple diatomic problems whose LSD solutions are known. Figure 3 illustrates LSD results and Morse fits for Be_2, Li_2, BeF, and F_2. All computations were performed using unfiltered Hamann pseudopotentials.

The Be_2 and Li_2 systems are easily calculated using the planewave approach, and our results match the planewave solutions. BeF is an example of a material with two very disparate length scales: the Be pseudopotential is very soft and delocalized whereas the F pseudopotential is very stiff and localized about the nucleus. Computations with an unfiltered Hamann fluorine pseudopotential would require grids of size 128^3 or larger for the planewave method as compared to an equivalent of about 70^3 for the adaptive method.

The oscillations in the solution about the Morse fit for BeF and F_2 are due to accuracy limitations in our current implementation of the adaptive method. We are currently using only second order finite elements, and our mid-point integration scheme does not preserve the variational nature of the finite element formalism. In the following Section, we discuss future development efforts that will improve the accuracy of our adaptive code and reduce these spurious oscillations.

4.6 Analysis and Future Research

We have implemented an adaptive mesh refinement real-space code that solves the LDA/LSD equations for materials design. In doing so, we have developed a reusable object-oriented software framework for parallel adaptive methods and have employed several sophisticated numerical techniques. We do not yet believe that our current adaptive code is competitive with the best LCAO or planewave methods; however, below we identify changes that will improve the accuracy and computational speed of our adaptive approach.

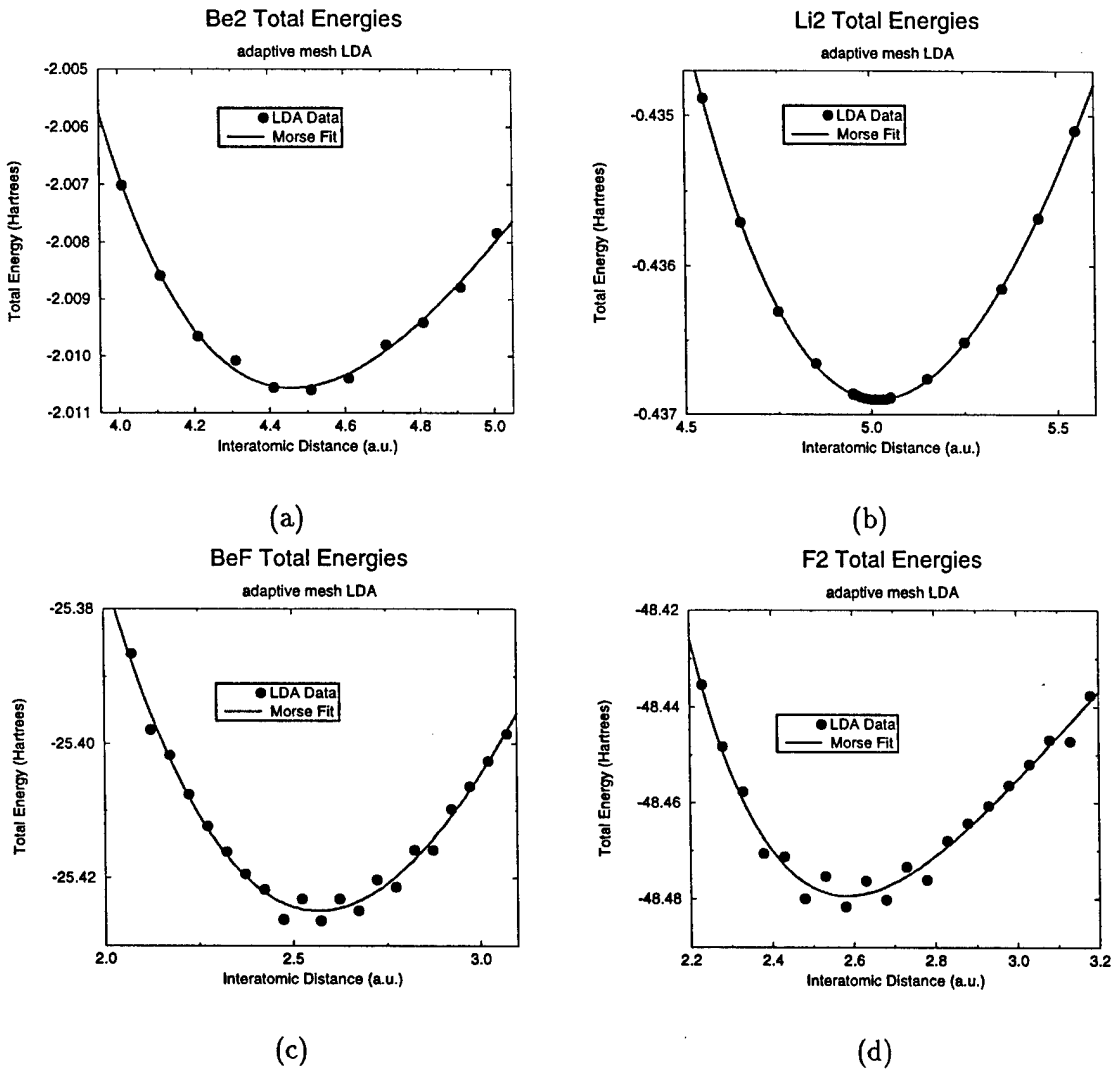


Figure 3: Sample adaptive LSD calculations for various diatomic molecules: (a) Be_2, (b) Li_2, (c) BeF, and (d) F_2. The Morse curve values were calculated by taking a least squares fit of the LDA data to the standard diatomic Morse energy profile.

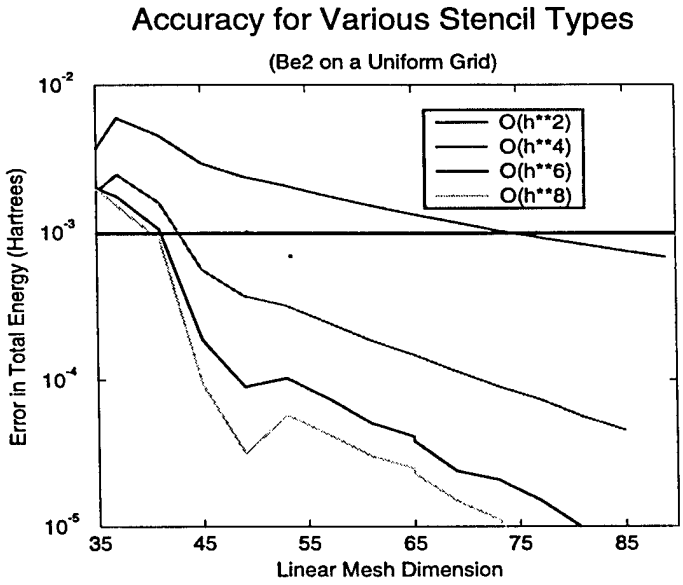


Figure 4: This graph shows the convergence in total LDA energy as a function of stencil order and mesh spacing for a Be_2 molecule. These results suggest that a sixth order $O(h^6)$ stencil requires approximately half as many points (40 vs. 75 in each dimension) as a second order $O(h^2)$ stencil for the same millihartree accuracy. In three dimensions, this represents an eight-fold reduction in mesh size.

4.6.1 Higher Order Finite Elements

Our current adaptive solver uses 3d trilinear elements that are $O(h^2)$ accurate; these types of elements are commonly used in the finite elements applications community. Unfortunately, this low order means that we must use numerous mesh points to obtain the millihartree or better accuracy desired for materials calculations. Figure 4 illustrates the slow convergence in energy for the second order method as compared to the higher order methods. These results were calculated for Be_2 on a uniform computational grid. For millihartree accuracy, the second order method requires eight times more points (in 3d) than a sixth order method. Equivalently, for the same number of grid points, a sixth order method can provide 0.01 millihartree accuracy as compared to only millihartree accuracy for the second order method.

We are currently developing an adaptive method that employs higher order elements to improve the accuracy of the method and reduce memory requirements. We plan to use either fourth or sixth order Hermite elements. Higher order methods should have the additional benefit of reducing the number of levels of refinement and thus improving the condition number of our numerical problems.

4.6.2 Eigenvalue Solver Preconditioning

The computational results in Section 4.4.1 illustrate how a good preconditioning method can significantly reduce the iteration count and thus time to solution. Although we have been successful in developing good preconditioning techniques for the Hartree calculation, we have not as yet developed an efficient preconditioner for our eigenvalue solver (see Section 4.4.2).

Our Rayleigh Quotient solver is more efficient than a steepest descent approach, but it still suffers from scaling problems with additional refinement levels. We are actively pursuing a multilevel preconditioning technique to reduce the number of iterations needed by the eigenvalue algorithm. We are considering either a multigrid preconditioner or a multilevel nodal basis preconditioner [1]. Experiments by Sung, Ong, and Weare [14] for planewave methods show the effectiveness of multilevel preconditioners for the eigenvalue equations.

5 Accomplishment: Parallel Planewave Implementation of AIMD

Using a planewave basis set, the Kohn-Sham equations, Eq.(1), may be implemented very efficiently. Large numbers of basis functions are required even when pseudopotentials are used. However, the computational cost of this is offset by the high parallelism and efficient vectorization of the algorithm. Broadly speaking the implementation of planewave LDA requires data parallel operations, array reductions and Fast Fourier transforms. Each of these are discussed below.

5.1 Data Parallel Operations

Many mathematical operations in planewave based LDA calculations are data parallel operations such as $\vec{X} = \alpha\vec{X} + \vec{Y}$ where $\vec{X}$ and $\vec{Y}$ are vectors with a large number of components. Since each component can be computed independently, these types of operations are very efficient on almost any platforms. Many RISC workstations (e.g., IBM and Silicon Graphics workstations) and vector machines such as CRAY provide very efficient routines, Basic Linear Algebra Subroutines (BLAS) tuned for their architectures. Our code takes full advantage of these routines. Furthermore, data parallel operations are perfectly parallelisable without any interprocessor communication. This is why the planewave method is easily parallelized.

5.2 Array Reductions

Our implementation of AIMD also uses a lot of array reduction operations such as dotproduct of two large vectors. All reduction operations used in our AIMD are vectorizable on CRAY. The BLAS routines tuned for RISC and vector architectures include these routines. Therefore, these operations are performed very efficiently on RISC and vector machines. On the other hand, array reduction requires interprocessor communications on parallel machines or networks of workstations, which may cost significant cputime on distributed-memory parallel machines. Fortunately, the CM-5 has very efficient routines to carry out these operations in the high performance Fortran language.

5.3 Fast Fourier Transform

The efficiency of the current AIMD code relies on 3D fast Fourier transformation (FFT) method. On serial processor machines such as RISC and vector machines, the multi-dimensional FFT is essentially same as a series of 1D FFT. Very efficient FFT routines are available on these platforms. On massively-parallel distributed-memory machines, 3D FFT is inefficient due to heavy interprocessor communication. However, if the array data is properly distributed, the multi-instance multi-dimensional FFT routines on CM-5 is remarkably fast. Unfortunately, such a fast FFT is not available on other massively parallel computers at present. We are currently developing an efficient 3d FFT for MIMD parallel computers.

5.4 Language Consideration

The serial version of our AIMD code was originally written in standard Fortran 77 whereas the parallel version uses a kind of high performance fortran (Fortran 90 + some extension). It is desirable to write a portable code that runs on any advanced vector computers, massively parallel machines and networks of workstations without major modification. We expect that high performance fortran will be soon available on most massively parallel platforms. Since Fortran 90 is close to high performance fortran and the CM Fortran, we are writing new code using standard Fortran 90. Machine-dependent extensions to Fortran 90 will be used if they are necessary to archive required computational efficiency. The new code has been tested on the Connection Machine model 500 (CM-500) at Naval Research Laboratory (NRL) and model 5 (CM-5) at Army High Performance Computing Resource Center(AHCRC) using CM-fortran 2.3.

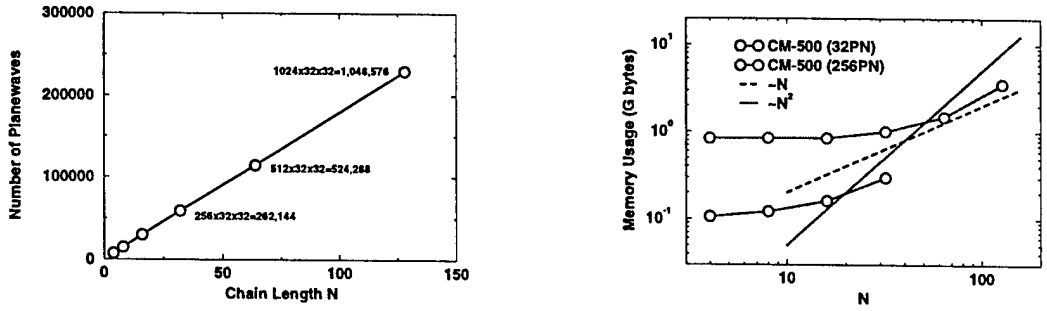
5.5 Performance Tests

The computational efficiency of the new code is tested by computing a chain of polyacetylene using periodic boundary condition. Instead of evaluating density at many k points in the Brillouin zone, more than one repeat units are placed in a large unit cell. This system is computationally one of the worst because a large unit cell is needed to isolate the chain from nearest neighbors. Furthermore, we used radix-2 FFT because mixed-radix FFT is not available on most parallel platforms at present. For a more densely packed system and with mixed-radix FFT available on certain platforms, higher performance can be obtained.

5.5.1 Computational Efficiency: Memory Usage

The size of planewave basis used for this test is shown in Figure 5a. The size of corresponding real space grids is given in the Figure. Previously, both the wavefunction in momentum space and in real space are kept in memory. Since the real space wavefunction consumes significant amount of memory, it was not possible to calculate $[CH]_{64}$ because of memory overflow. Now, it is possible to compute up to $[CH]_{128}$, which involves as many as 360 electrons per spin.

We successfully reduced the memory usage in the new code which requires only 1/3 – 1/2 of memory previously required. Figure 5b displays the amount of memory used on 32-PN and



(a) (b)

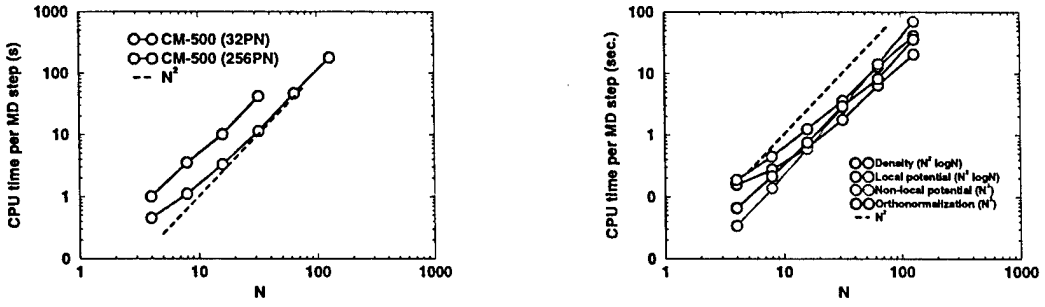
Figure 5: (a) Size of planewaves and (b) memory usage of ab initio molecular dynamics simulation for $[CH]_N$ chains. The numbers in the left figure are the corresponding grid sizes in real space.

256-PN CM-500 as a function of the number of repeat units. For large N, the memory usage should scale N^2 since both the number of basis functions and the number of electrons increase linearly. Figure 5a indicates that at N=128, the memory usage is nearly in proportion to N^2. Therefore, the memory usage becomes a major problem with planewave method above this size. However, chains of these sizes are large enough to simulate the properties of realistic polymers with various defects (kinks, dopants, .etc.) We will carry out such simulation in the near future.

5.5.2 Computational Efficiency II: Speed

Reduction of memory usage increases the number of interprocessor communications, which may require additional cputime. Figure 6 shows the cputime usage per molecular dynamics time step as a function of the chain length. In order to perform a significant period of simulation the cputime per step must be less than a few minutes. The new code is fast enough to perform dynamical simulation of $[CH]_{64}$. Although further improvement in speed for $[CH]_{128}$ is desired, limited simulation can be done for this size. Using the mixed-radix FFT library routines now available on CM-500, it is possible to reduce the cputime significantly for large systems.

Since the number of floating operations in the planewave method is proportional to N^3, the cputime also scales N^3 at large N. In principle, N^3-scaling is valid on parallel machines when N is sufficiently large. However, because of interprocessor communication and other complications in massively parallel operations, our new code does not scale N^3 but nearly N^2



(a) (b)

Figure 6: CPU time usage of ab initio molecular dynamics simulation for $[CH]_N$ chains.

for the size of systems we are interested in. This is because for small systems routines scaled as $N^2 \log N$ are dominant.

5.5.3 Degree of Accuracy

The high degree of accuracy of our methods has been discussed in the literature. [4, 5, 6, 13, 3] Our previous code has been independently tested at Wright-Patterson Air Force Base and shown to be very accurate. To demonstrate the accuracy of our new capability to calculate charged species we calculated the ionization potential (I) of Beryllium atom from the energies of neutral atom and positive ion. Our result of .330au compares well with the experimental value of .343au and with the other calculated values in the literature. We also calculated the electron affinity (A) of Chlorine atom. We obtained $A = 3.9eV$. Agreement with the experimental value (3.6eV) is marginal but accurate enough to predict many chemical processes. Generally speaking, the calculation of negative ions is harder than that of neutral atoms or positive ions. We have now implemented a generalized gradient correction method which is expected to improve the electron affinity calculation.

5.6 Applications

Developing highly conducting polymers holds a high priority in Air Force Materials Research. Therefore, we will apply our AIMD method to conducting polymers. We begin with polyacetylene and a polymer based on squarelene. However, our applications will be extended to many other chemical systems. In the following, we show preliminary results of squarelene-based polymers.

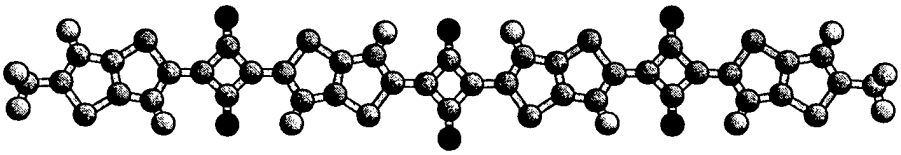


Figure 7: Structure of the trimer, $CH_3 - [C_{10}H_2O_2S_2]_3 - C_7H_5S_2$

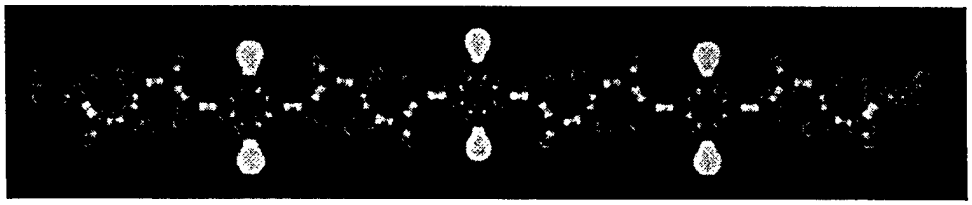


Figure 8: Valence electron density of the trimer, $CH_3 - [C_{10}H_2O_2S_2]_3 - C_7H_5S_2$

5.6.1 Narrow-Gap Polymers based on Squarelene

A polymer based on a squarelene and fused thiophenes shown in Fig.7 is a candidate of narrow-gap semi-conducting polymer. An accurate band-gap as a function of polymer length is important to understand the electronic properties of these polymers. However, because of its large repeat unit, it is too expensive to apply an accurate CI method. On the other hand, the Hartree-Fock method may not provide sufficient accuracy. If the band gap is evaluated from HOMO-LUMO gap, HF overestimates the gap substantially whereas LSD significantly underestimates it. The singlet-triplet excitation energy approaches to the band gap as the size increases, if many-body effects do not play a significant role. Since LSD accurately predicts the singlet-triplet excitation energy, we estimate the band gap by calculating both singlet and triplet state energy.

Oligomers up to trimer (Fig. 7) are calculated. The largest system contains 250 electrons. A supercell of $77 \times 16 \times 10$ box and $256 \times 64 \times 32$ grid points are used. About 200,000 basis functions per orbital is needed to get converged results. The electron density on the molecular plane (Fig. 8) indicates no clear alternation in bond order. Singlet-triplet excitation energy plotted in Fig. 9 suggests that the band gap of an infinite chain will be less than 0.5eV.

References

[1] J. H. Bramble, J. E. Pasciak, and J. Xu. Parallel multilevel methods. *Math. Comp.*, 55:1–22, 1990.

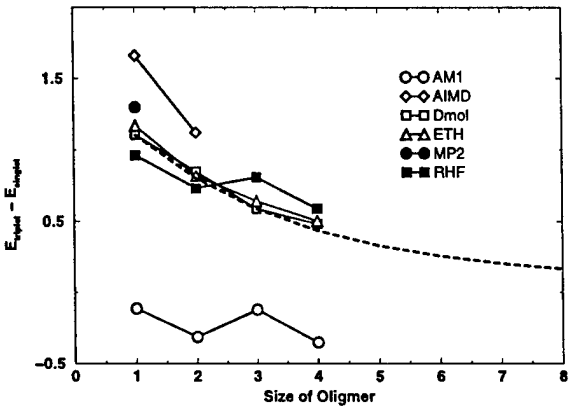


Figure 9: Various theoretical calculation of singlet-triplet energy gap as a function of oligomer length

[2] Eric J. Bylaska, Scott R. Kohn, Scott B. Baden, Alan Edelman, Ryoichi Kawai, M. Elizabeth G. Ong, and John H. Weare. Scalable parallel numerical methods and software tools for material design. In *Proceedings of the Seventh SIAM Conference on Parallel Processing for Scientific Computing*, pages 219–224, San Francisco, CA, February 1995. SIAM.

[3] R. Kawai, J. F. Tombrello, and J. H. Weare. Li_5 as a pseudorotating planar cluster. *Phys. Rev. A*, 49:4236, 1994.

[4] R. Kawai and J. H. Weare. From van der waals to metallic bonding: The growth of be clusters. *Phys. Rev. Lett.*, 64:81, 1990.

[5] R. Kawai and J. H. Weare. Instability of the b_{12} icosahedral cluster: Rearrangement to a lower energy. *J. Chem. Phys.*, 95:1151, 1991.

[6] R. Kawai and J. H. Weare. Anomalous stability of b_{13}^+ clusters. *Chem. Phys. Lett.*, 191:311, 1992.

[7] Scott Kohn, John Weare, Elizabeth Ong, and Scott Baden. Parallel adaptive mesh refinements for electronic structure calculations. In *Proceedings of the Eighth SIAM Conference on Parallel Processing for Scientific Computing*, Minneapolis, Mn, March 14-17 1997. SIAM.

[8] Scott Kohn, John Weare, Elizabeth Ong, and Scott Baden. Software abstraction and computational issues in parallel structured adaptive mesh methods for electronic calculations. In *Proceedings of the SIAM Workshop on Structured Adaptive Mesh Refinement Grid Methods*, Minneapolis, Mn, March 12-13 1997. SIAM.

[9] Scott R. Kohn. *A Parallel Software Infrastructure for Dynamic Block-Irregular Scientific Calculations.* PhD thesis, University of California at San Diego, June 1995.

[10] W. Kohn and L. J. Sham. *Phys. Rev.*, 140:A1133, 1965.

[11] D. E. Longsine and S. F. McCormick. Simultaneous Rayleigh quotient minimization methods for $Ax = \lambda Bx$. *Linear Algebra and Its Applications*, 34:195–234, 1980.

[12] Steve F. McCormick, editor. *Multilevel Adaptive Methods for Partial Differential Equations.* SIAM, Philadelphia, 1989.

[13] M.-W. Sung, R. Kawai, and J. H. Weare. Packing transition in nanosized li clusters. *Phys. Rev. Lett.*, 73:3552, 1994.

[14] Ming-Wen Sung, Maria Elizabeth G. Ong, and John H. Weare. Direct minimization of the Kohn-Sham equations using preconditioned conjugate gradient methods, March 1995. American Physical Society March Meeting.

[15] M. P. Teter, M. C. Payne, and D. C. Allan. Solution of schroedinger equation for large systems. *Phys. Rev. B*, 40:12255, 1989.