Journal of Computational Physics **160,** 29–41 (2000)
doi:10.1006/jcph.2000.6440, available online at http://www.idealibrary.com on IDEAL®

Parallel Empirical Pseudopotential Electronic Structure Calculations for Million Atom Systems

A. Canning,* L. W. Wang,*,† A. Williamson,† and A. Zunger†

*NERSC, Lawrence Berkeley National Laboratory, Berkeley, California 94720; and † National Renewable Energy Laboratory, Golden, Colorado 80401

Received November 1, 1999

We present a parallel implementation of the previously developed folded spectrum method for empirical pseudopotential electronic structure calculations. With the parallel implementation we can calculate a small number of electronic states for systems of up to one million atoms. A plane-wave basis is used to expand the wavefunctions in the same way as is commonly used in *ab initio* calculations, but the potential is a fixed external potential generated using atomistic empirical pseudopotentials. Two techniques allow the calculation to scale to million atom systems. First, the previously developed folded spectrum method allows us to calculate directly a few electronic states of interest around the gap. This makes the scaling of the calculation $O(N)$ for an N atom system and a fixed number of electronic states. Second, we have now developed an efficient parallel implementation of the algorithm that scales up to hundreds of processors, giving us the memory and computer power to simulate one million atoms. The program's performance is demonstrated for many large semiconductor nanostructure systems.

Key Words: electronic structure; density functional theory; conjugate gradients; pseudopotential; quantum dots.

I. INTRODUCTION

Most electronic structure calculations are based on solving an effective single-particle Schrödinger equation

$$\hat{H}\psi_i(\mathbf{r}) \equiv \left[-\frac{1}{2}\nabla^2 + V(\mathbf{r})\right]\psi_i(\mathbf{r}) = \epsilon_i\psi_i(\mathbf{r}), \tag{1}$$

where $\{\psi(\mathbf{r})\}$ are orthogonal single particle wavefunctions and $V(\mathbf{r})$ is the total potential of

0021-9991/00 $35.00

the system. There are two main strategies for solving Eq. (1), reflecting two distinct types of physical problems:

The first class of physical problems requires the calculation of all N_{occ} eigensolutions of Eq. (1) corresponding to the N_{occ} occupied states. For example, if Eq. (1) corresponds to a 1000 atom Si cluster, there are $N_{\text{occ}} = 1000 \times 2$ doubly occupied levels for this four valence electron/atom system. The need to solve for all occupied levels stems from the need to determine the atomic positions or vibrational properties. Since atomic positions (vibrations) depend on the first (second) derivatives of the total energy with respect to displacements, and since the total energy depends on all N_{occ} eigensolutions of Eq. (1), these types of physical problems require a "full solution" of Eq. (1).

One also needs to calculate all N_{occ} eigensolutions to determine equilibrium atomic positions in those problems where the geometry is difficult to guess or approximate at the outset. This includes molecular shapes, surface reconstruction patterns in semiconductors, atomic rebonding next to defects in insulators, and structural phase transitions under pressure. For this class of physical problems one needs in fact to solve Eq. (1) iteratively, since $V(r)$ is unknown at the outset, and since it depends functionally on all N_{occ} solutions $\{\psi_i\}$ via $V(r) = V[\{\psi_i, i = 1, \ldots, N_{\text{occ}}\}]$. The two leading methodologies that specify this functional relationship are the Kohn–Sham [1] approximation to the density functional theory [2] and the Hartree–Fock method [3], both of which provide a specific functional relationship between $\{\psi_i, i = 1, \ldots, N_{\text{occ}}\}$ and $V(r)$ of Eq. (1). Since all N_{occ} eigensolutions are needed, and since the orthogonalization of N_{occ} solutions scales as N_{occ}^3, the computational effort here scales as N^3, where $N \propto N_{\text{occ}}$ is the number of atoms. This limits the applicability of this approach to systems containing only a few hundred atoms.

The second class of physical problems requires the eigensolutions of Eq. (1) in only a small energy range, e.g., just below and above the Fermi energy. This is appropriate for those physical problems where the total energy is not needed, because the atomic positions and the potential $V(r)$ can be modeled at the outset or are given. For example, free-standing nanostructures made of >1000 atoms often possess bulk-like interatomic distances which are known; in strained nanostructures one can accurately evaluate the structural parameters from continuum elasticity [4] or atomistic elasticity [4, 5] (e.g., the valence force field, VFF) without having to solve Eq. (1). For these types of problems, it would be wasteful to solve for all eigensolutions of Eq. (1), as the physical interest often lies within a narrow energy range (e.g., the optical transitions across a band gap). Naturally, solving Eq. (1) for only a subset of the eigensolutions means that we cannot construct $V(r) = V[\{\psi_i, i = 1, \ldots, N_{\text{occ}}\}]$, but must know $V(r)$ at the outset. This is the case when $V(r)$ is some external potential (e.g., the "confining potential" in nanostructures), when it can be constructed as a superposition of screened ionic pseudopotentials [6] (the empirical pseudopotential method [7]), or when one can approximate $V(r)$ by first solving Eq. (1) self-consistently for smaller subsystems and then assembling them together.

Very often, the system size in this class of problems is very large, ranging from a few thousand atoms (e.g., free standing quantum dots [8–11], composition modulations in alloys [12], random superlattices [13], short range order [14], impurities [15], ordered alloys [16]) to a few million (e.g., embedded quantum dots [17]. Two techniques enable us to solve such large-scale problems:

(i) The folded spectrum method (FSM) [18, 19] allows us to calculate directly the band edge states without calculating all the other states below the band edge. As a result,

the method has $O(N)$ scaling for a fixed number of states. Section II of this paper describes the computational aspects of the folded spectrum method in detail. This method has been applied to the study of free-standing [8–11] and embedded [17] quantum dots, as well as to various inhomogeneous alloys [12, 14, 16], superlattices, [13] and impurities [15].

(ii) The other technique involves an efficient parallel implementation of the algorithm using a specialized parallel fast Fourier transformation (FFT). To simulate a system containing a million atoms, parallel computation becomes necessary to obtain the required memory and computer power to perform the calculation on a reasonable timescale. Since the wavefunctions $\psi_i(\mathbf{r})$ are expanded in a plane-wave basis, the most time-consuming operation to solve Eq. (1) is the FFT. In our current ESCAN (Energy SCAN) code, an efficient parallel FFT subroutine is used to transform the wavefunction from plane-wave representation to a real space grid. The parallel FFT is described in detail in Section III.

In Section IV, we discuss speed-up curves for a few different-sized heterostructure systems on different numbers of processors of a Cray T3E computer. In Section V we present an example application of this code to the calculation of an embedded quantum dot system of one-half million atoms.

II. THE FOLDED SPECTRUM METHOD

The central idea of the folded spectrum method [18, 19] is that an eigensolution (ϵ, ψ) (we drop the subscript i) of Eq. (1) also satisfies

$$(\hat{H} - \epsilon_{\text{ref}})^2 \psi = (\epsilon - \epsilon_{\text{ref}})^2 \psi, \tag{2}$$

where ϵ_{ref} is a fixed reference energy. Furthermore, the N_{occ}th ($N_{\text{occ}} + 1$th) state, counted from the bottom of the energy spectrum of $\hat{H}$, becomes the lowest state in the spectrum of $(\hat{H} - \epsilon_{\text{ref}})^2$ if ϵ_{ref} is placed inside the band gap and close to the top of the valence band (bottom of the conduction band). This is illustrated in Fig. 1. (The idea of using the squared Hamiltonian was used in the residual minimization method approach to iteratively diagonalize large matrices [20].) This fact is used in the FSM to calculate the band edge

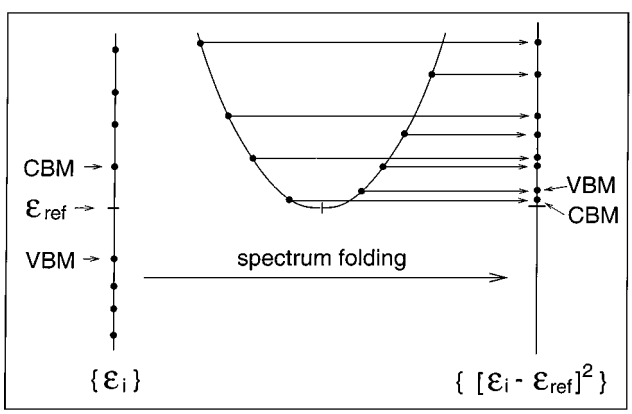

FIG. 1. Folded spectrum energy scheme. The spectrum of $\hat{H}$ has been folded at ϵ_{ref} into the spectrum of $(\hat{H} - \epsilon_{\text{ref}})^2$. The valence band maximum (VBM) state and the conduction band minimum (CBM) states have become the lowest states in the spectrum of $(\hat{H} - \epsilon_{\text{ref}})^2$.

state $\{\psi\}$ by seeking the variational minimum of

$$F = \langle\psi|(\hat{H} - \epsilon_{\text{ref}})^2|\psi\rangle. \tag{3}$$

Unlike the variational minimum of $\langle\psi|\hat{H}|\psi\rangle$, which yields the lowest energy state of $\hat{H}$, the minimum solution of F gives the band edge states if ϵ_{ref} is placed inside the band gap. Although simple algorithmically, the real challenge of the FSM is to develop a scheme which finds ψ efficiently from the minimization of F. This is not an easy task because changing H to F significantly increases the condition number of the linear operator (the matrix) [21]. We have used the conjugate gradient method to solve the variational minimum of F (this is described in detail in Ref. [19]). The Lanczos method is also appropriate for this type of problem, and a study of a variant of the method applied to the FSM is presented in [22].

To explain how we use the conjugate gradient method for this problem we must go into more detail of the plane-wave expansion for the wavefunctions. In a plane-wave representation the wavefunctions can be written as

$$\psi(\mathbf{r}) = \sum_{\mathbf{q}} a(\mathbf{g}) e^{i\mathbf{g}\cdot\mathbf{r}}. \tag{4}$$

The selection of the number of plane-waves is determined by a cutoff E_{cut} in the plane-wave kinetic energy $\frac{1}{2}|\mathbf{g}|^2$, where $\{\mathbf{g}\}$ are the reciprocal lattice vectors. The wavefunction ψ is stored in reciprocal space by its coefficients $a(\mathbf{g})$. It is transformed onto a real space grid $\psi(\mathbf{r})$ by applying a parallel FFT, which will be described in the next section.

Application of $(\hat{H} - \epsilon_{\text{ref}})^2$ to ψ is carried out [19] by twice applying $[-\frac{1}{2}\nabla^2 + V(\mathbf{r}) - \epsilon_{\text{ref}}]$ to ψ. The term $-\frac{1}{2}\nabla^2\psi$ is computed in reciprocal space, while $V(\mathbf{r})\psi(\mathbf{r})$ is obtained by using an FFT to transform $a(\mathbf{g})$ to real space, $\psi(\mathbf{r})$, then applying $V(\mathbf{r})$ to $\psi(\mathbf{r})$ and transforming the product back to $\mathbf{g}$ space. The result can be cast in the same form as $\sum_{\mathbf{g}} c(\mathbf{g}) e^{i\mathbf{g}\cdot\mathbf{r}}$ (with the same energy cutoff for $\{\mathbf{g}\}$). Then $[-\frac{1}{2}\nabla^2 + V(\mathbf{r}) - \epsilon_{\text{ref}}]$ is applied again to this function to get the final result F. Once F is obtained, we minimize it with respect to the variational wavefunction coefficients $a(\mathbf{g})$, using the preconditioned conjugate gradient method [23]. The conjugate gradient method is defined as a series (indexed by superscript $\{j\}$) of sequential line minimizations of the task function F. A line minimization implies adding a search wavefunction $P_j(\mathbf{r})$ to the current wavefunction $\psi^j(\mathbf{r})$ and constructing a new wavefunction $\psi^{j+1}(\mathbf{r})$,

$$\psi^{j+1}(\mathbf{r}) = \psi^j(\mathbf{r})\cos(\theta) + P_j(\mathbf{r})\sin(\theta), \tag{5}$$

which minimizes F at a value of θ. In this procedure, the search function $P_j(\mathbf{r})$ is made orthogonal to $\psi^j(\mathbf{r})$. The next search direction P_{j+1} is given by

$$P_{j+1}(\mathbf{g}) = A(\mathbf{g})\chi_{j+1}(\mathbf{g}) + \beta_j P_j(\mathbf{g}), \tag{6}$$

where

$$\chi_{j+1}(\mathbf{r}) \equiv \frac{\partial F}{\partial \psi^{j+1}(\mathbf{r})} = \left[-\frac{1}{2}\nabla^2 + V(\mathbf{r}) - \epsilon_{\text{ref}}\right]^2 \psi^{j+1}(\mathbf{r}). \tag{7}$$

The preconditioner $A(\mathbf{g})$ is a $\mathbf{g}$-space function

$$A(\mathbf{g}) = \frac{E_k^2}{\left(\frac{1}{2}g^2 + V_0 - \epsilon_{\text{ref}}\right)^2 + E_k^2}, \tag{8}$$

where V_0 is the average potential and E_k is the average kinetic energy of the wavefunction ψ. The β_j in Eq. (6) is determined using the Polak–Ribiere formula [21]:

$$\beta_j = \frac{\sum_{\mathbf{g}} A(\mathbf{g})[\chi_{j+1}(\mathbf{g}) - \chi_j(\mathbf{g})]\chi_{j+1}(\mathbf{g})}{\sum_{\mathbf{g}} A(\mathbf{g})\chi_j(\mathbf{g})\chi_j(\mathbf{g})}. \tag{9}$$

Usually, a few wavefunctions $\{\psi_i\}$ are minimized simultaneously while being kept mutually orthogonal. N_l line minimization steps are carried out for each wavefunction followed by a subspace diagonalization based on $(\hat{H} - \epsilon_{\text{ref}})^2$. Then, we start another sequence of line minimization iterations. This forms an outside loop. This algorithmic structure is the same as in the minimization of $\langle\psi|\hat{H}|\psi\rangle$ in the conventional conjugate gradient method [23]. At the end of the minimization, to unambiguously obtain the eigenenergies $\{\epsilon_i\}$ of $\hat{H}$, a subspace diagonalization based on $\hat{H}$ is carried out. We use $N_l \sim 100$ (the square of typical N_l values used in conventional conjugate gradient methods based on $\hat{H}$). The number of outside loops is typically 5–10, to converge to an accuracy of 10^{-6} Ryd in the Raleigh quotient of the wavefunction. This is about the same number of outside iterations as is used in conventional methods. Following the above procedure, the computational effort to solve for each wavefunction $\psi_i(r)$ scales linearly with the system size, N (more exactly, it scales as $N \ln N$ due to the FFT). Thus if we increase the size of the system, always calculating the same number of wavefunctions, then the computational cost scales linearly with the system size.

III. PARALLELIZATION STRATEGY

The two most important criteria driving the choice of any parallelization strategy are equal division of the computational workload among the processors (load balancing) and minimization of the communications. Due to the plane-wave expansion used in the ESCAN code the parallelization scheme we have chosen has strong similarities to those used for *ab initio* plane-wave codes [24, 25] but there are some differences. In the case of the ESCAN code, since we calculate a small number of electronic states, the FFT will dominate the calculation time for all system sizes (typically it takes more than 80% of the total run time). In *ab initio* plane-wave codes the N^3 scaling of the orthogonalization step will dominate as we go to large systems while the FFT part of the calculation (scaling as $N^2 \ln N$) dominates for smaller systems. Therefore, in *ab initio* codes, load-balancing schemes are typically driven by the orthogonalization part of the calculation, distributing the same number of wavefunction coefficients to each processor. This is done either by giving as near as possible an equal number of bands to each processor [25] or by dividing up the g vectors for each band among the processors, giving as closely as possible an equal number of g vectors to each processor. Distribution by bands is not possible for the ESCAN code, as we normally calculate only a small number of bands, typically four to six, so we must divide the g vectors among the processors. While parallelization over the k points is another possibility, we are typically using only the $k = 0$, Γ point in the ESCAN calculations. The main difference between a load-balancing scheme based on a division of g vectors for ESCAN and a self-consistent code is that the former is driven by the requirement of load balancing the

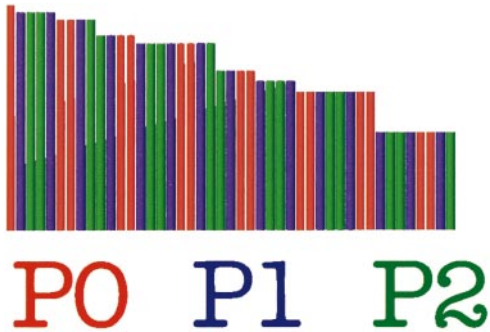

FIG. 2. Distribution of g vector columns to three processors. The g vector columns are produced by dividing the sphere of g vectors into z direction columns (see Fig. 3a). The columns are then ordered by length and assigned to the three processors as shown, with processor zero being assigned all the red columns, processor one the blue columns, and processor two the green columns.

three-dimensional distributed FFT rather than the orthogonalization part. Distributed 3d FFTs are among the most demanding algorithms on the interconnect system of a parallel computer as the communications are global (there is no physical locality in the communication structure) and of significant size (the complete data set is communicated). We will now describe in more detail our approach to parallelizing the FFT to load balance the calculation and minimize the communications. Our approach is similar to the method used in Ref. [24] for *ab initio* plane-wave codes.

In reciprocal space the data set representing a band is a sphere of points or more generally an ellipsoid. In this section, for simplicity, we will assume the data set is a sphere and the grid in real space is a cube. Our parallelization scheme is easily extended to the more general case of an ellipsoid. To perform a three-dimensional FFT it is necessary to perform one-dimensional FFTs in each of the three dimensions. We work with a data order of (x, y, z) in real space and (z, y, x) in g space. Library FFTs usually return the data in the same order as provided but in our application it is not necessary for the data to have the same order. This avoids a global transpose and the associated communications. In order to load balance for the first step in the 3d FFT, where we perform one-dimensional FFTs on all the z-columns, it is necessary to have the same number of z-columns of g vectors on each processor. A second weaker constraint to the load balancing is that the number of g vector coefficients should not vary too much between the processors, as this would result in some processors requiring much more memory than others and hence reduce the size of systems that can be run on a given number of processors. In order to satisfy these two constraints we used the following load-balancing scheme (see Fig. 2):

1. Divide the sphere of g vectors into columns in the z direction; each column is defined by a (g_x, g_y) index (see Fig. 3a).
2. Order these columns in descending order according to their length.
3. Assign these columns to the processors as shown in Fig. 2.

In this way we give, as closely as possible, an equal number of columns to each processor while still having approximately the same number of g vectors on each processor. Now that we have determined the g space data layout we can describe the distributed three-dimensional FFT. All g space calculations performed in the code are done with this data distribution.

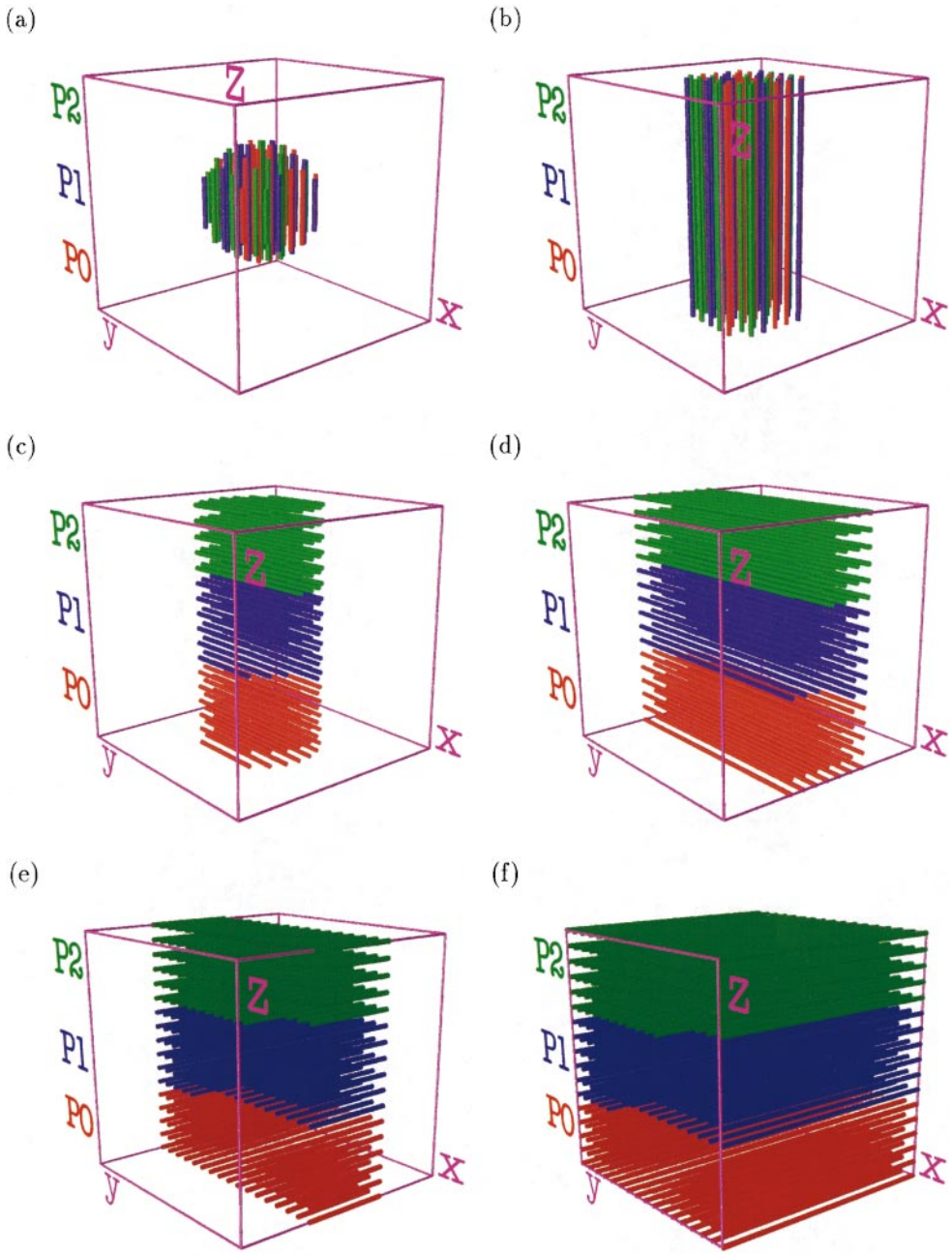

FIG. 3. Parallel three-dimensional FFT. This shows which processors deal with which part of the grid during the three dimensional FFT. The colors red, blue, and green correspond to the part of the grid that resides on processors zero, one, and two (for more details see text in Section III).

The size of the real space grid for the potential is typically taken to be twice the diameter of the *g* space sphere. A large saving in computations and communications can be made by not performing FFTs on the zero columns and not transforming the zero columns. At each step of the distributed 3d FFT, the data we are working on are expanding out to fill the full real space grid. For this reason it is inefficient to use FFT parallel libraries, which

cannot take advantage of this saving and often have very restricted data layouts such as the requirement that each processor must hold a complete plane of data. Starting with the z-column distribution of g vectors described above the three-dimensional FFT on a single band is performed in the following way (see Fig. 3):

1. Each processor pads out the ends of each of the z-columns of g vector coefficients that it holds with zeros to form full length z-columns on each processor. The complete data set is now a cylinder of length $2d$ and diameter d, where d is the diameter of the original g vector sphere and $2d$ is the cube size (see Fig. 3b).
2. Each processor performs one-dimensional FFTs on its set of z-columns.
3. The cylinder of data is now reorganized from z-columns to y-columns (ordered by their x, z indices) with each processor now holding a contiguous set of y-columns. Global data redistribution is required at this step (i.e., going from Fig. 3b to Fig. 3c), as can be seen by the changes in color of the data elements. Each processor is given as closely as possible the same number of y-columns.
4. The y-columns (which are sections through the cylinder) are now padded with zeros at the ends to form full-length columns. The complete data set is now a slab of dimension d in the x direction and $2d$ in the other directions (see Fig. 3d).
5. Each processor performs one-dimensional FFTs on its set of y-columns.
6. The slab of data is now transformed from y-columns (x, z ordered) to x-columns (y, z ordered) with each processor now having a set of contiguous x-columns (i.e., going from Fig. 3d to Fig. 3e). Each processor is given as closely as possible the same number of x-columns. Communications are minimized at this step since most of the transformations are local to the processor with only data at the interfaces of the colored blocks being communicated. In the ideal case where there are complete (y, x) planes on each processor the transpose can be done locally on each processor and there are no communications. Due to our choice of data layouts in the FFT the main communications are in step 3, where the data set (the cylinder) is much smaller than the slab.
7. The x-columns are now padded at the ends with zeros so the global data set is now the complete cube of side $2d$ (see Fig. 3f).
8. Each processor performs one-dimensional FFTs on its set of x-columns, producing the final distributed real space representation of the wavefunction.

The reverse of this process is performed to go from real space to g space. In the communications routines in the FFTs, to reduce latency effects as much as possible, each processor gathers the data it has to send to each of the other processors before sending. In the case of a Γ point calculation we follow the same procedure but work with a half sphere in g space.

IV. CODE PERFORMANCE

We have tested the speed of our program on an InAs/GaAs quantum dot system. The potential $V(\mathbf{r})$ is constructed as a superposition of screened atomic empirical pseudopotentials. More explicitly, we have

$$V(\mathbf{r}) = \sum_{R_\alpha} v_\alpha(|\mathbf{r} - \mathbf{R}_\alpha|), \tag{10}$$

where $\{R_\alpha\}$ are the atomic positions of atom type α. The spherical atomic empirical pseudopotentials $v_\alpha(r)$ are obtained via a fit to the bulk band structure of the constituent materials [6]. The empirical pseudopotentials of Ref. [26] are used for InAs and GaAs. Using this method, the potential $V(\mathbf{r})$ of a million atom system can be readily constructed. The main task here is to calculate the wavefunctions $\psi_i(\mathbf{r})$ near the band edge of the energy spectrum (i.e., i near N_{occ}). A 5 Ryd cutoff energy is used for the plane-wave basis in Eq. (4).

To test the speed of the code for different numbers of processors and different system sizes we chose three InAs/GaAs quantum dot systems containing 8000, 97,336, and 1,000,000 atoms (see Fig. 4). The real space grid sizes for the potential for these systems are 128^3, 288^3, and 576^3. The number of processors we tested ranges from 2 to 512. For reasons of memory the 97,336 atom system does not run on fewer than 8 processors and the 1,000,000 atom system does not run on fewer than 128 processors. The timings are shown for one state, with 50 line minimizations and without spin–orbit interactions. The speedup of the code is extremely good, with only the smaller systems showing poor performance for very large processor counts that they would typically not be run on. As the number of processors increases, the ratio of communications to calculations increases and the speedup becomes less linear. When there is less than a full (y, z) plane on each processor this also causes greatly increased communications as is the case of the 8000 atom system on 256 and 512 processors (real space grid size is 128^3). The results presented here are for a Fortran90 and SHMEM (the low-level communication library on Cray and SGI machines) version of the code. MPI versions of the FFTs have a less linear speedup on the Cray T3E.

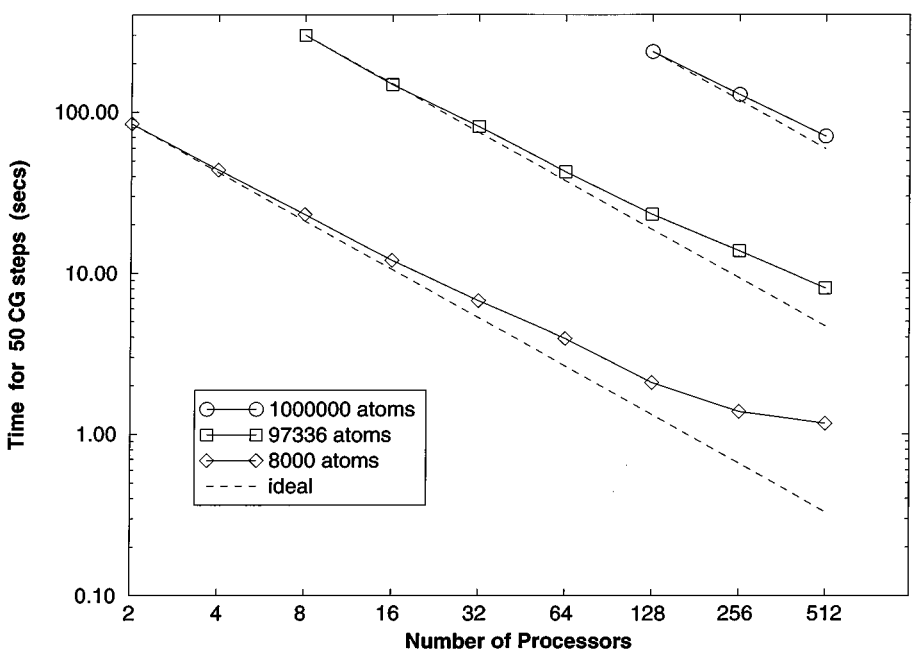

FIG. 4. Speedup curves on the Cray-T3E900 parallel supercomputer for three different sizes of InAs/GaAs quantum dots.

V. APPLICATION: ELECTRONIC STATES OF A PYRAMIDAL QUANTUM DOT

The ESCAN code is particularly suited to studying quantum dots, as it is reasonably straightforward to construct accurate empirical pseudopotentials for these systems, and the number of atoms, while being beyond the size possible with *ab initio* codes, is within the reach of the parallel ESCAN code. In this section, to illustrate the applicability of the ESCAN code to these types of problems, we will present some results for a self-assembled pyramidal quantum dot with one quarter million atoms.

Self-assembled semiconductor quantum dots have received a lot of attention recently [27]. The most interesting properties of these quantum dots is the change in the band gap as a function of the quantum dot sizes and the physical confinement of the band edge electronic states inside the quantum dot. We have calculated a pyramidal shaped InAs quantum dot embedded in a GaAs matrix. The base length of the pyramid is $20a$, where $a = 5.653$ Å is the GaAs lattice constant. The height of the pyramid is $10a$. This particular dot was discussed in Ref. [26]. The atomic positions are relaxed from the ideal zinc blende structure due to the lattice mismatch between InAs and GaAs. This relaxation is calculated using a atomistic valence force field model [5]. The resulting $\{R_\alpha\}$ is used in Eq. (10) to calculate $V(\mathbf{r})$. Spin–orbit interaction is represented as an atomic nonlocal part in the

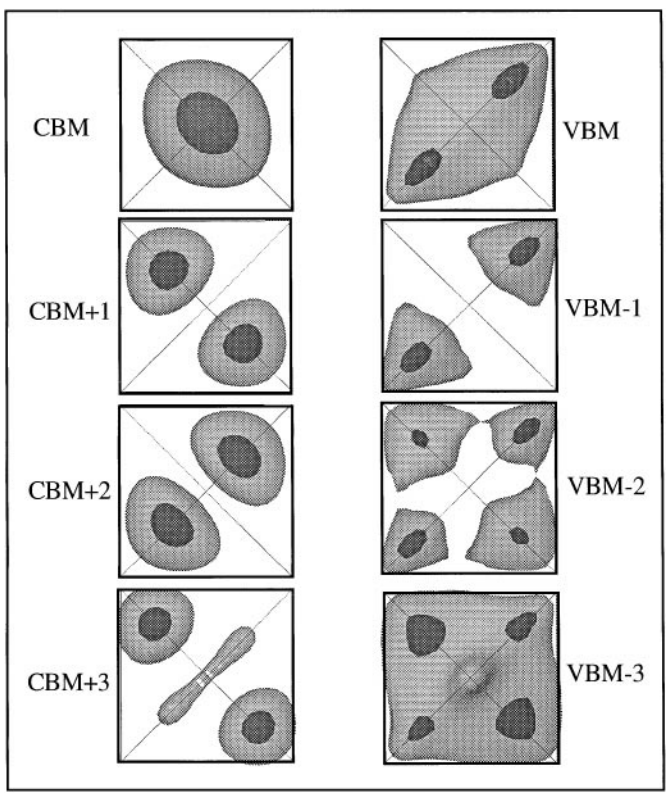

FIG. 5. Charge density isosurface plots of the CBM and VBM states for a pyramidal quantum dot (InAs pyramid placed inside a GaAs matrix) of base size 113 Å. The complete system contains about 250,000 atoms.

potential. As a result, the wavefunction $\psi_i(\mathbf{r})$ has spin-up and spin-down components. The relaxed InAs pyramid is placed inside a $28a \times 28a \times 30a$ GaAs matrix periodic supercell. The resulting system contains one quarter million atoms. We have calculated the four conduction band minimum (CBM) states and four valence band maximum (VBM) states. The real space grid size is $448 \times 448 \times 480$. On a 128-processor run, each processor holds 298 z-columns and holds on average 42,300 g coefficients for each wavefunction. The number of g coefficients varies from 42,284 to 42,310, which balances the memory usage on each processor. The whole calculation takes about 20 h on 128 processors of a Cray T3E computer.

The band gap increases from 0.41 eV for bulk InAs to 0.96 eV in the quantum dot. The charge density of the four CBM states and four VBM states are plotted in Fig. 5. They are all localized inside the quantum dot. A more detailed account of this work has been reported in [26].

VI. CONCLUSION

We have introduced an atomistic approach to calculating the electronic states of systems up to one million atoms. In this approach, the wavefunction is expanded using a plane-wave basis, as in conventional *ab initio* calculation, but the potential is constructed non-self-consistently from an atomistic empirical pseudopotential. The calculation of large systems is made possible by two techniques: (1) the folded spectrum method introduced previously [18, 19], which reduces the calculation effort from the conventional $O(N^3)$ scaling to $O(N)$ scaling, for a fixed number of electronic states; and (2) an efficient parallel implementation of the algorithm using specialized FFTs, which is introduced here, allowing the code to scale to hundreds of processors. With these techniques, we demonstrated that direct million atom electronic structure calculations are possible. As the technological and scientific importance of semiconductor nanostructures increase rapidly, such large-scale accurate numerical calculations become more and more necessary. The empirical pseudopotential calculation is much faster than the self-consistent *ab initio* calculations, and it avoids the band gap problem in such calculations [28] by fitting to the experimental band structure. On the other hand, it is much more accurate and provides possibilities for further improvements on atomistic details compared to phenomenological continuum theories like the k.p. model [29]. Thus it provides a useful middle ground for large-scale electronic structure calculations for the near future. This approach is not limited to the empirical pseudopotential method. It is applicable to any fixed, Fourier transformable potential $V(r)$.

ACKNOWLEDGMENTS

Work at NREL was supported by DOE-Basic Energy Sciences, Division of Material Science under Contract DE-AC36-98-GO10337. Work at NERSC was supported by DOE Office of Science, Office of Laboratory Policy and Infrastructure Management under Contract DE-AC03-76SF00098. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the DOE Office of Energy Research. Andrew Canning has benefited from useful discussions with B. Pfrommer and A. De Vita in the development of the FFTs for this code.

REFERENCES

1. W. Kohn and L. J. Sham, Self-consistent equations including exchange and correlation effects, *Phys. Rev.* **140**, 1133A (1965).
2. P. Hohenberg and W. Kohn, Inhomogeneous electron gas, *Phys. Rev.* **136**, 864B (1964).
3. L. A. Curtiss, P. C. Redfern, K. Raghavachari, and J. A. Pople, Assessment of Gaussian-2 and density functional theories for the computation of ionization potentials and electron affinities, *J. Chem. Phys.* **109**, 42 (1998).
4. C. Pryor, J. Kim, L. W. Wang, A. Williamson, and A. Zunger, Comparison of two methods for describing the strain profiles in quantum dots, *J. Appl. Phys.* **83**, 2548 (1998).
5. P. Keating, Effects of invariance requirements on the elastic strain energy of crystals with applications to the diamond structure, *Phys. Rev.* **145**, 637 (1966).
6. L. W. Wang and A. Zunger, Local-density-derived semiempirical pseudopotentials, *Phys. Rev. B* **51**, 17398 (1995); H. Fu and A. Zunger, Local-density-derived semiempirical nonlocal pseudopotentials for InP with applications to large quantum dots, *Phys. Rev. B* **55**, 1642 (1997).
7. M. L. Cohen and T. K. Bergstresser, Band structures and pseudopotential form factors for fourteen semiconductors of the diamond and zinc-blende structures, *Phys. Rev.* **141**, 789 (1964).
8. D. J. Norris, Al. L. Efros, M. Rosen, and M. G. Bawendi, *Phys. Rev. B* **53**, 16347 (1996).
9. C. Y. Yeh, S. B. Zhang, and A. Zunger, Confinement, surface, and chemisorption effects on the optical properties of Si quantum wires, *Phys. Rev. B* **50**, 14405 (1994).
10. H. Fu and A. Zunger, InP quantum dots: Electronic structure, surface effects, and the redshifted emission, *Phys. Rev. B* **56**, 1496 (1997).
11. A. Franceschetti and A. Zunger, Direct pseudopotential calculation of exciton coulomb and exchange energies in semiconductor quantum dots, *Phys. Rev. Lett.* **78**, 915 (1997).
12. T. Mattila, L. W. Wang, and A. Zunger, Electronic consequences of lateral composition modulation in semiconductor alloys, *Phys. Rev. B* **59**, 5678 (1999).
13. K. A. Mader, L. W. Wang, and A. Zunger, Electronic structure of intentionally disordered AlAs/GaAs superlattices, *Phys. Rev. Lett.* **74**, 2555 (1995).
14. L. Bellaiche and A. Zunger, Effects of atomic short-range order on the electronic and optical properties of GaAsN, GaInN, and GaInAs alloys, *Phys. Rev. B* **57**, 4425 (1998).
15. L. Bellaiche, S. H. Wei, and A. Zunger, Band gaps of GaPN and GaAsN alloys, *Appl. Phys. Lett.* **70**, 3558 (1997).
16. T. Mattila, S. H. Wei, and A. Zunger, Electronic structure of sequence mutations in ordered GaInP2, *Phys. Rev. Lett.* **83**, 2010 (1999).
17. J. Kim, L. W. Wang, and A. Zunger, Comparison of the electronic structure of InAs/GaAs pyramidal quantum dots with different facet orientations, *Phys. Rev. B* **57**, R9408 (1998).
18. L. W. Wang and A. Zunger, Solving Schroedinger's equation around a desired energy: Application to silicon quantum dots, *J. Chem. Phys.* **100**, 2394 (1994).
19. L. W. Wang and A. Zunger, in *Semiconductor Nanoclusters*, edited by P. V. Kamat and D. Meisel (Elsevier, Amsterdam, 1996).
20. D. M. Wood and A. Zunger, A new method for diagonalising large matrices, *J. Phys. A* **18**, 1343 (1985).
21. G. H. Golub and C. F. Van Loan, *Matrix Computations* (John Hopkins Univ. Press, Baltimore, 1989).
22. K. Wu, A. Canning, H. D. Simon, and L.-W. Wang, Thick-restart Lanczos method for electronic structure calculations, *J. Comput. Phys.* **154**, 156 (1999).
23. M. Payne, M. P. Teter, D. C. Allan, T. A. Arias, and J. D. Joannopoulos, Iterative minimization techniques for *ab initio* total-energy calculations: Molecular dynamics and conjugate gradients, *Rev. Mod. Phys.* **64**, 1045 (1992).
24. L. J. Clarke, I. Štich, and M. C. Payne, Large-scale *ab initio* energy calculations on parallel computers, *Comput. Phys. Comm.* **72**, 14 (1992).
25. J. Wiggs and H. Jónsson, A parallel implementation of the Car–Parrinello method by orbital decomposition, *Comput. Phys. Comm.* **81**, 1 (1994).

26. L. W. Wang, J. Kim, and A. Zunger, Electronic structures of [110]-faceted self-assembled pyramidal InAs/GaAs quantum dots, *Phys. Rev. B* **59**, 5678 (1999).
27. A. Zunger, Electronic structure theory of semiconductor quantum dots, *MRS Bull.* **23**, 35 (1998).
28. M. S. Hybertsen and S. G. Louie, Electron correlation in semiconductors and insulators: Band gaps and quasiparticle energies, *Phys. Rev. B* **34**, 5390 (1986); R. W. Godby, M. Schluter, and L. J. Sham, Self-energy operators and exchange-correlation potentials in semi-conductors, *Phys. Rev. B* **37**, 10159 (1988).
29. C. Pryor, Geometry and material parameter dependence of InAs/GaAs quantum dot electronic structure, *Phys. Rev. B* **60**, 2869 (1999).