DFTB+, a Sparse Matrix-Based Implementation of the DFTB Method†

B. Aradi,*,‡ B. Hourahine,§ and Th. Frauenheim‡

Bremen Center for Computational Materials Science, Universität Bremen, Otto-Hahn-Alle 1, 28359 Bremen, Germany, and SUPA, Department of Physics, The University of Strathclyde, John Anderson Building, 107 Rottenrow, Glasgow G4 0NG, United Kingdom

Received: January 9, 2007; In Final Form: May 4, 2007

A new Fortran 95 implementation of the DFTB (density functional-based tight binding) method has been developed, where the sparsity of the DFTB system of equations has been exploited. Conventional dense algebra is used only to evaluate the eigenproblems of the system and long-range Coulombic terms, but drop-in $O(N)$ or $O(N^2)$ modules are planned to replace the small code sections that these entail. The developed sparse storage structure is discussed in detail, and a short overview of other features of the new code is given.

I. Introduction

The density functional-based tight binding (DFTB) method is a very efficient tool for carrying out quantum mechanical simulations. Several contributions in this special issue describe the mathematical and physical basis of the method as well as the broad range of systems where it delivers reliable results. The current contribution focuses on a technical aspect of the DFTB method, namely, the sparsity of the corresponding equations and how this can exploited to build an efficient DFTB implementation. The general DFTB equations are not derived here. Interested readers should consult the appropriate contributions in this volume or the literature.[1−5] As a proof of concept, the sparse technique described here has been implemented in the DFTB+ code.

There has been extensive work on developing sparse structures for quantum chemistry, and it is not our intention to review the substantial literature of sparse matrices here. A recent discussion of some of the techniques is presented in ref 6. Also, the DFTB+ project is not the first attempt to create a DFTB implementation, which exploits the sparsity of the underlying equations. Zhang et al. used the SLEPc external library to solve the eigenvalue problem in a sparse format.[7] Yang et al. replaced the diagonalization with the $O(N)$ divide and conquer technique,[8] which delivers only the density matrix for the investigated system but not the eigenvalues and the eigenvectors. However, a detailed description of the storage type of the sparse matrices was not given in those cases. Sternberg et al. also created a DFTB-based $O(N)$ method[9] with a rather complex special storage format.

Regarding the creation of the density matrix, the DFTB+ code is not as developed as in the earlier attempts. Currently it uses conventional dense diagonalization algorithms, but the extension of the code is planned to incorporate diagonalization and/or density matrix creation schemes. At the end of this article, a brief list of additional features of this code is given.

† Part of the "DFTB Special Section".
* Corresponding author.
‡ Universität Bremen.
§ The University of Strathclyde.

II. Sparsity of the DFTB equations

In the DFTB approach, every one-electron wavefunction $\psi(\mathbf{r})$ is expressed as a linear combination of atomic orbitals $\phi_\mu(\mathbf{r})$

$$\psi(\mathbf{r}) = \sum_\mu c_\mu \phi_\mu(\mathbf{r}) \tag{1}$$

(Throughout this article, the convention is used that summations on Greek letters run over all basis functions.) The main task is to solve the eigenproblem

$$\sum_\nu H^\sigma_{\mu\nu} c^\sigma_\nu = \epsilon^\sigma_\mu \sum_\nu S_{\mu\nu} c^\sigma_\nu$$

where

$$H^\sigma_{\mu\nu} = \langle \phi_\mu | \hat{H}^\sigma | \phi_\nu \rangle \text{ and } S_{\mu\nu} = \langle \phi_\mu | \phi_\nu \rangle \tag{2}$$

are the Hamiltonian and the overlap matrices, respectively. The index σ indicates the spin state ($\uparrow$ or $\downarrow$).

If orbitals ϕ_μ and ϕ_ν are located on atoms A and B, respectively, then the Hamiltonian matrix element $H^\sigma_{\mu\nu}$ for the spin-polarized self-consistent charge (scc) case can be written as

$$H^\sigma_{\mu\nu} = \langle \phi_\mu | \hat{H}_0 | \phi_\nu \rangle + \frac{1}{2} S_{\mu\nu} \sum_C \sum_{l'' \in C} (\gamma_{Al,Cl''} + \gamma_{Bl',Cl''}) \Delta q_{Cl''} \pm \frac{1}{2} S_{\mu\nu} \left(\sum_{l'' \in A} W_{All''} m_{Al''} + \sum_{l'' \in B} W_{Bl'l''} m_{Bl''} \right) \tag{3}$$

The first term is the matrix element of the non-scc DFTB Hamiltonian. The second term is the scc contribution, where the sum runs over all shells (l'') of all atoms (C) in the system. A "shift vector'', $\gamma_{Al,Cl''}\Delta q_{Cl''}$, containing the scc potential at site Al can be constructed. γ itself consists of a long-range pure Coulombic term and a short-range term S

$$\gamma_{Al,Cl''} = \frac{1}{R_{AC}} - S(U_{A,l}, U_{C,l''}, R_{AC}) \tag{4}$$

10.1021/jp070186p CCC: $37.00
Published on Web 06/14/2007

where R_{AC} is the distance between atoms A and C and $U_{A,l}$ and $U_{C,l''}$ are the Hubbard parameters for the appropriate atoms and shells. The charge difference $\Delta q_{Cl''}$ is the difference between the sum of the Mulliken charges on shell l of atom C and the total charge on that shell in an isolated atom.

The last term of eq 3 contains the spin contribution. Its sign depends on the spin channel, σ. The interaction between the spins is determined by the spin coupling constants $W_{All'}$, where l and l' are shells on the same atom. The spin polarization m_{Al} of shell l on atom A is the difference between the sum of the Mulliken charges on the spin up and spin down orbitals in that shell

$$m_{Al} = q_{Al\uparrow} - q_{Al\downarrow} \tag{5}$$

The matrix elements of the non-scc Hamiltonian and overlap matrices vanish with increasing distance between the atoms so that in large systems both matrices contain mostly zeros. The pattern of the possible nonzero elements is determined by the geometry of the system to be calculated (as well as the order of the numbering used for the atoms). As shown in eq 4, the scc shift contribution contains both short- and long-range terms. Nevertheless, the total scc contribution is screened by the overlap matrix so that the pattern of the matrix elements with nonvanishing scc contributions is the same as the overlap matrix. Because the non-scc Hamiltonian and the overlap matrix are tabulated on the same grid up to the same cutoff, the patterns of the nonzero scc and the nonzero non-scc matrix elements are the same. Similarly, the spin contribution is also multiplied by the overlap matrix, giving rise to the same pattern. By determining this pattern at the start (or during a calculation if the geometry has changed), it can be ensured that only the relevant matrix elements are evaluated and stored.

III. Sparse Storage

The storage scheme implemented in the DFTB+ code is dictated by the physical nature of the system rather than by abstract mathematical criteria as would be the case with many common sparse formats (e.g., compressed column/row format). This simplifies the source code and also promises better cache-line reuse behavior because inside the loops the elements are accessed in the order of their storage in memory. The implented technique is similar to that of Challacombe's atom-blocked sparse storage[10] extended to periodic boundary conditions with arbitrary k-point sampling. The next two sections contain detailed descriptions of the storage scheme for nonperiodic and periodic systems.

A. Nonperiodic Systems. To establish sparse storage, the Hamiltonian and the overlap matrices are first partitioned into atomic blocks, with each block containing all of the matrix elements related to the interaction of a certain atom pair. The elements inside the nonzero blocks are stored in a 1D vector. Starting with the leftmost column of blocks, the blocks are stored top to bottom column after column, whereby elements of each block are stored in column major order. Figure 1 shows the storage order of the elements for a fictitious system consisting of four atoms with four orbitals on each, as shown in Figure 2. Because the Hamiltonian and the overlap matrices are symmetric, only blocks in the lower triangle are stored. For the on-site blocks, it would be sufficient to store only one triangle of each block, but to keep the storage scheme simple, both triangles are stored, as for all of the other off-site blocks.

The current scheme allows a simple and transparent way of accessing the matrix elements. A neighbor map is built at the beginning of the calculation (and after every geometry update) containing information about the interacting atoms. Because of the symmetry of the matrices, only the neighbor relationships where the index of the first interacting atom is less than or equal to the index of the second interacting atom are stored.

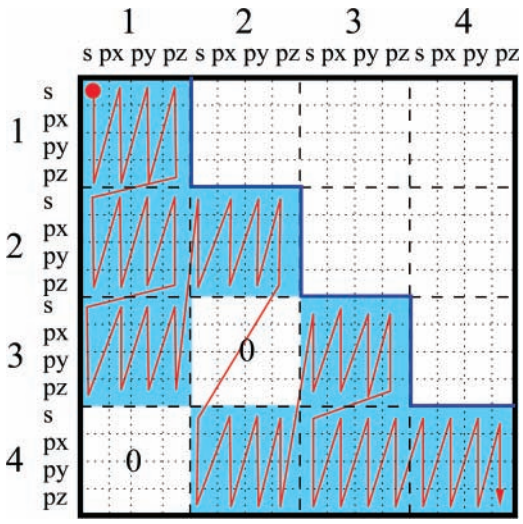

Figure 1. Storage order of the matrix elements for the sparse storage of the Hamiltonian and the overlap matrices for the fictious system shown in Figure 2. The numbers indicate the atoms, and s, px, py, and pz are the orbitals of the individual atoms. The red line with the arrow indicates the storage order. Dashed lines indicate atomic blocks, and dotted lines indicate the rows and columns of the matrix. Only nonzero blocks in the lower triangle (shaded background) are stored.

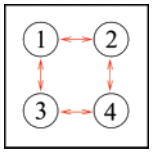

Figure 2. Fictitious nonperiodic system of four atoms with first neighbor interaction only. The red arrows indicate the interactions between the atoms.

B. Periodic Systems. The sparse storage introduced in the previous section can be extended to periodic structures as well. However, one has to consider that in the periodic case the basis functions are Bloch functions, with each of them built from an atomic orbital ϕ_μ as

$$\beta_\mu^{\mathbf{k}}(\mathbf{r}) = \frac{1}{\sqrt{N}} \sum_{\mathbf{R}} \phi_\mu(\mathbf{r} - \mathbf{R}) e^{i\mathbf{kR}} \tag{6}$$

where the summation runs over all possible translation vectors $\mathbf{R}$ of the periodic system. Such Bloch functions can be constructed for every orbital ϕ_μ and for every point $\mathbf{k}$ in the Brillouin zone of the periodic system. The number N specifies the number of cells in the repeating region.

As shown in ref 11, the matrix elements of an arbitrary operator $\hat{O}$ with the translational symmetry of the lattice can be written as

$$O_{\mu\nu}(\mathbf{k}) = \langle \beta_\mu^{\mathbf{k}} | \hat{O} | \beta_\nu^{\mathbf{k}} \rangle = \sum_{\mathbf{R}} e^{i\mathbf{kR}} O_{\mu\nu}(\mathbf{R}) \text{ where } O_{\mu\nu}(\mathbf{R}) = \langle \phi_\mu(\mathbf{r}) | \hat{O} | \phi_\nu(\mathbf{r} - \mathbf{R}) \rangle \tag{7}$$

One can think about $O_{\mu\nu}(\mathbf{R})$ as a rectangularly shaped matrix built from square submatrices belonging to specific values of $\mathbf{R}$. For the Hamiltonian and the overlap matrices, only a finite number of submatrices must be considered because the matrix

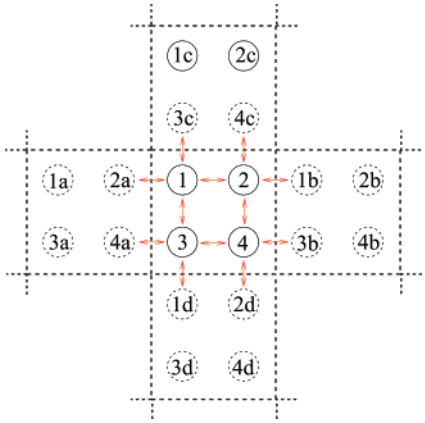

Figure 3. Fictitious periodic system of four atoms with first neighbor interaction only. The red arrows indicate the interactions between the atoms. Dashed lines delimit the periodic images of the central cell. Only images with nonvanishing interactions with the central cell are shown.

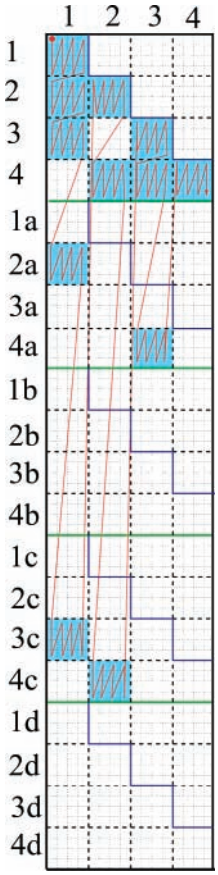

Figure 4. Storage order of the matrix elements for the sparse storage of the Hamiltonian and the overlap matrices for the fictitious system shown in Figure 3. Numbers indicate the atoms. The red line with the arrow shows the storage order of the elements. Dashed lines indicate atomic blocks, and dotted lines indicate the rows and columns of the matrix belonging to a specific orbital of the current atom (orbital names omitted). Only nonzero blocks in the lower triangles (shaded background) are stored.

elements go to zero with increasing length of **R**. Figure 3 shows a fictitious periodic system of four atoms with first neighbor interactions only. The structure of the appropriate rectangular Hamiltonian or overlap matrix is shown in Figure 4, assuming that each atom has only four orbitals.

The elements of the rectangular matrices are stored in a similar way as for the nonperiodic case. Starting with the leftmost column of atomic blocks, those with nonzero elements are stored from top to bottom, whereby the elements inside a block are written to the storage vector in column major order. Analogous to the nonperiodic case, only one triangle of each square matrix must be considered. Because the matrix elements of the rectangular matrices are all real, the following relation is valid for an operator $\hat{O}$ having the same translational symmetry as the lattice:

$$O_{\mu\nu}(\mathbf{R}_0) = \langle\phi_\mu(\mathbf{r})|\hat{O}|\phi_\nu(\mathbf{r}-\mathbf{R}_0)\rangle = \langle\phi_\nu(\mathbf{r}-\mathbf{R}_0)|\hat{O}|\phi_\mu(\mathbf{r})\rangle^* = \langle\phi_\nu(\mathbf{r}-\mathbf{R}_0)|\hat{O}|\phi_\mu(\mathbf{r})\rangle$$
$$= \langle\phi_\nu(\mathbf{r})|\hat{O}|\phi_\mu(\mathbf{r}+\mathbf{R}_0)\rangle = O_{\nu\mu}(-\mathbf{R}_0) \quad (8)$$

Accordingly, every element in the upper triangle of a certain submatrix $O_{\mu\nu}(\mathbf{R}_0)$ is equal to the transposed element in the lower triangle of the submatrix $O_{\nu\mu}(-\mathbf{R}_0)$. For example, for the system in Figure 3, there is no need to store the interaction between an arbitrary orbital on atom 2 and another orbital on periodic image 1b of atom 1 if the equivalent interaction between the same orbitals on atoms 1 and 2a has already been stored.

The rectangular Hamiltonian and the overlap matrix are real, and the number of nonzero atomic blocks scales as the number of atoms for large systems, so it is advantageous to store those matrices in that form. If the square (complex) form is needed (e.g., for solving the eigenproblem), then the rectangular form can be very easily folded back by considering the appropriate phase factors:

$$O_{\mu\nu}(\mathbf{k}) = \sum_{\mathbf{R}} e^{i\mathbf{k}\mathbf{R}} O_{\mu\nu}(\mathbf{R}) \quad (9)$$

An analogous reverse transform can be defined as

$$O_{\mu\nu}(\mathbf{R}) = \sum_{l} \omega_l e^{-i\mathbf{k}_l\mathbf{R}} O_{\mu\nu}(\mathbf{k}_l) \quad (10)$$

where the sum runs over the k points used for the calculation. The factor ω_l is the weight of the k point $\mathbf{k}_l$. If the quantity, which is transformed back, is used only in expressions where it is multiplied by the overlap (or the Hamiltonian) matrix, then it is sufficient to consider only those translation vectors **R** that made nonzero contributions to the overlap (or the Hamiltonian). The next section describes this requirement in more detail.

The strategy chosen for storing and handling the matrices for periodic systems is a natural extension of the strategy for nonperiodic systems. This has the advantage that the implementing routines can be used for both cases without the need for case differentiation. Only the neighbor maps must be constructed with regard to the applied boundary conditions.

IV. Analysis in Real Space

As pointed out in the previous section, any square matrix can be transformed into a rectangular form with the help of eq 10. This is especially useful for those quantities that are multiplied by the overlap matrix in all the expressions where they occur. In that case, it is sufficient to use only those values of **R** for the transformation that were also used to construct the rectangular overlap matrix. Similarly, it is enough to store those atomic blocks from the resulting rectangular matrix, where the corresponding blocks in the overlap matrix have nonzero elements (and had been, therefore, also stored). The expressions can then be evaluated rapidly using the sparse form. As examples, the Mulliken charge analysis and the non-scc force calculation are demonstrated below. Because in the rectangular

matrix formalism the nonperiodic matrices are special cases of the more general periodic ones, both examples are shown for periodic boundary conditions. It should be also noted that although for the sake of better visualization we always refer to rectangular matrices, in the actual implementation only the nonzero blocks of those matrices need to be stored and used in the evaluation of the expressions. This assures scaling behavior that is linear in the number of atoms.

A. Mulliken Analysis. When solving the eigenproblem in a certain k point **k**, the resulting one-electron wave functions $\psi_i^{\mathbf{k}}(\mathbf{r})$ are obtained as a linear combination of the Bloch functions defined in eq 6:

$$\psi_i^{\mathbf{k}}(\mathbf{r}) = \sum_{\mu} c_{i\mu}^{\mathbf{k}} \beta_{\mu}^{\mathbf{k}}(\mathbf{r}) \tag{11}$$

The number of electrons in level i in k-point **k** can be then written as

$$q_i^{\mathbf{k}} = n_i^{\mathbf{k}} |\psi_i^{\mathbf{k}}|^2 = n_i^{\mathbf{k}} \sum_{\mu} \sum_{\nu} c_{i\mu}^{\mathbf{k}*} c_{i\nu}^{\mathbf{k}} \langle \beta_{\mu}^{\mathbf{k}} | \beta_{\nu}^{\mathbf{k}} \rangle \tag{12}$$

where $n_i^{\mathbf{k}}$ is the occupation number for the selected eigenlevel in the given k point. By substituting the identity operator into eq 7, the overlap of two Bloch functions can be written as

$$\langle \beta_{\mu}^{\mathbf{k}} | \beta_{\nu}^{\mathbf{k}} \rangle = \sum_{\mathbf{R}} \mathrm{e}^{\mathrm{i}\mathbf{k}\mathbf{R}} S_{\mu\nu}(\mathbf{R}) \text{ where } S_{\mu\nu}(\mathbf{R}) = \langle \phi_{\mu}(\mathbf{r}) | \phi_{\nu}(\mathbf{r} - \mathbf{R}) \rangle \tag{13}$$

The total number of electrons N_e in the system can be obtained by summing up $q_i^{\mathbf{k}}$ from all the eigenlevels and k points, whereby the number of electrons in each k point must be multiplied by the weight of the given k point:

$$N_\mathrm{e} = \sum_l \omega_l \sum_i q_i^{\mathbf{k}_l} \tag{14}$$

$$= \sum_i \sum_l \omega_l n_i^{\mathbf{k}_l} \sum_{\mu} \sum_{\nu} c_{i\mu}^{\mathbf{k}_l *} c_{i\nu}^{\mathbf{k}_l} \sum_{\mathbf{R}} \mathrm{e}^{\mathrm{i}\mathbf{k}_l\mathbf{R}} S_{\mu\nu}(\mathbf{R}) \tag{15}$$

The summation for l goes from 1 to the number of k points, and the summation for i goes from 1 to the number of eigenlevels in each k point. By introducing the density matrix for k point k_l as

$$P_{\mu\nu}(\mathbf{k}_l) = \sum_i n_i^{\mathbf{k}_l} c_{i\mu}^{\mathbf{k}_l *} c_{i\nu}^{\mathbf{k}_l} \tag{16}$$

the previous equations can be written as

$$N_\mathrm{e} = \sum_{\mathbf{R}} \sum_{\mu} \sum_{\nu} S_{\mu\nu}(\mathbf{R}) \sum_l \omega_l P_{\mu\nu}(\mathbf{k}_l) \mathrm{e}^{\mathrm{i}\mathbf{k}_l\mathbf{R}} \tag{17}$$

Accordingly, a packed (real space) density matrix can be introduced as

$$P_{\mu\nu}(\mathbf{R}) = \sum_l \omega_l P_{\mu\nu}(\mathbf{k}_l) \mathrm{e}^{-\mathrm{i}\mathbf{k}_l\mathbf{R}} \tag{18}$$

so that using one obtains

$$\sum_l \omega_l P_{\mu\nu}(\mathbf{k}_l) \mathrm{e}^{\mathrm{i}\mathbf{k}_l\mathbf{R}} = P_{\mu\nu}(-\mathbf{R}) = P_{\nu\mu}(\mathbf{R}) \tag{19}$$

Putting this back in, the total charge (or the charge of a certain orbital q_ν) can be calculated as

$$N_\mathrm{e} = \sum_{\mathbf{R}} \sum_{\mu} \sum_{\nu} S_{\mu\nu}(\mathbf{R}) P_{\nu\mu}(\mathbf{R}) = \sum_{\nu} q_\nu \tag{20}$$

with

$$q_\nu = \sum_{\mathbf{R}} \sum_{\nu} S_{\mu\nu}(\mathbf{R}) P_{\nu\mu}(\mathbf{R}) \tag{21}$$

being the charge on orbital ϕ_ν. The expression for q_ν contains only an element-wise multiplication of rectangular matrices $S_{\mu\nu}(\mathbf{R})$ and $P_{\mu\nu}(\mathbf{R})$. Because only the lower triangles are stored for submatrices $S_{\mu\nu}(\mathbf{R})$ and $P_{\mu\nu}(\mathbf{R})$ as a result of the symmetry, additional care must be taken so that the contribution from orbital ϕ_μ to the charge q_ν must also be accounted for by q_μ (as the contribution from orbital ϕ_ν).

B. Non-scc Forces. The non-scc contribution to the forces on atom C can be calculated in the DFTB method as

$$\mathbf{F}_C^0 = \sum_l \omega_l \sum_i n_i^{\mathbf{k}_l} \sum_{\mu} \sum_{\nu} c_{i\mu}^{\mathbf{k}_l *} c_{i\nu}^{\mathbf{k}_l} \left[\frac{\partial H_{\mu\nu}^0(\mathbf{k}_l)}{\partial \mathbf{R}_C} - \epsilon_i \frac{\partial S_{\mu\nu}(\mathbf{k}_l)}{\partial \mathbf{R}_C} \right]$$

where

$$H_{\mu\nu}^0(\mathbf{k}_l) = \langle \beta_{\mu}^{\mathbf{k}_l} | \hat{H}_0 | \beta_{\nu}^{\mathbf{k}_l} \rangle \text{ and } S_{\mu\nu}(\mathbf{k}_l) = \langle \beta_{\mu}^{\mathbf{k}_l} | \beta_{\nu}^{\mathbf{k}_l} \rangle$$

By using eq 7, the expression for the non-scc force can be rewritten as

$$\mathrm{F}_C^0 = \sum_l \omega_l \sum_i n_i^{\mathbf{k}_l} \sum_{\mu} \sum_{\nu} c_{i\mu}^{\mathbf{k}_l *} c_{i\nu}^{\mathbf{k}_l} \sum_{\mathbf{R}} \mathrm{e}^{\mathrm{i}\mathbf{k}_l\mathbf{R}} \left[\frac{\partial H_{\mu\nu}^0(\mathbf{R})}{\partial \mathbf{R}_C} - \epsilon_i \frac{\partial S_{\mu\nu}(\mathbf{R})}{\partial \mathbf{R}_C} \right] \tag{22}$$

By introducing the energy-weighted density matrix

$$P_{\mu\nu}^\mathrm{e}(\mathbf{k}_l) = \sum_i \epsilon_i^{\mathbf{k}_l} n_i^{\mathbf{k}_l} c_{i\mu}^{\mathbf{k}_l *} c_{i\nu}^{\mathbf{k}_l}$$

and similar to eq 18 its real space rectangular equivalent

$$P_{\mu\nu}^\mathrm{e}(\mathbf{R}) = \sum_l \omega_l P_{\mu\nu}^\mathrm{e}(\mathbf{k}_l) \mathrm{e}^{-\mathrm{i}\mathbf{k}_l\mathbf{R}}$$

eq 22 can be rewritten as

$$\mathbf{F}_C^0 = \sum_{\mathbf{R}} \sum_{\mu} \sum_{\nu} \left[P_{\nu\mu}(\mathbf{R}) \frac{\partial H_{\mu\nu}^0(\mathbf{R})}{\partial \mathbf{R}_C} - P_{\nu\mu}^\mathrm{e}(\mathbf{R}) \frac{\partial S_{\mu\nu}(\mathbf{R})}{\partial \mathbf{R}_C} \right]$$

Similar to the Mulliken analysis, this expression can also be fully evaluated in the sparse form. After calculating the derivatives with respect to $\mathbf{R}_C$ for the stored (nonzero) blocks, only an element-wise multiplication of matrix elements is required (while allowing for the skew symmetry of derivatives of H^0 and S).

V. Performance

The sparse storage technique described in the previous sections had been implemented in the DFTB+ code. This code is the successor of previous DFTB implementations DFTB and DYLAX. During its design, special attention was paid to the

sparsity of the DFTB equations. Almost all operations are accomplished in the sparse format and scale linearly in time and memory with the number of atomic orbitals in the system.

A. Testing Environment. To compare the efficiency of the sparse algorithms versus their dense version, we investigated their performance for the cases of a rather sparse and a rather dense system. For the former, we chose cylindrically shaped unrelaxed silicon nanowires (SiNW) along the ⟨110⟩ direction with various supercell lengths along the periodic axis of the wire. The diameter of each wire was the same (1.5 nm). The dense system was represented by supercells of various sizes containing a cut from bulk silicon with a vacancy (V_{Si}) in the middle. The vacancy was needed to induce charge transfer in the system to test the performance of the algorithms during the scc loops. The silicon was described with an sp basis using the parametrization from the pbc-0-1 parameter set.[21] For every system, we made benchmarks with periodic as well as with nonperiodic boundary conditions. Because of the missing functionality of arbitrary k-point sampling in the old DFTB code, all periodic calculations were made using the Γ point as only the k point.

The tests for the sparse algorithms were executed using the official 1.0 release of the DFTB+ code. Because this code does not contain the dense versions of the tested algorithms, we compared it against the old official (now obsoleted) DFTB code, making sure that both codes use the same diagonalizer (LAPACK[12] divide and conquer) so that comparisons of total running times are fair. Both codes were compiled with the Intel Fortran compiler (version 9.1 for the EM64T architecture) using exactly the same (aggressive) optimization options and the same LAPACK library (Intel MKL 8.1.1). The tests were executed on Pentium Dual Core machines (CPU frequency 2.6 GHz, cache size 4096 KB). The test runs always used only one thread. The benchmark data was calculated by averaging over timing results of five subsequent runs. The timing information was obtained by the cpu_{time} Fortran call, except for the total execution time where the user time for the process (as returned by the Unix command time) was used.

B. Sparse versus Dense Algorithms. We tested both the sparse algorithm for the Mullikan population analysis and the sparse algorithm for the non-scc force calculation in DFTB+ versus their dense counterparts in the old DFTB code. The sparse version of the Mullikan population analysis turned out to be too fast for reliable timing data because even for systems with more than 1000 atoms its execution took less than 10^{-2} s. Therefore, only the benchmark data for the non-scc force calculations are presented here.

Figures 5 and 6 show the execution times for the silicon vacancy and the silicon nanowire, respectively, using different system sizes. It can be clearly seen that the sparse version of the algorithm scales linearly with the number of atoms (N) in the system and the dense algorithm shows a scaling behavior of N^2. The sparse algorithm handles periodic and non-periodic systems in a unified framework; therefore, the performance for the same system with and without a periodic boundary condition is nearly identical. In the dense implementation, the periodic boundary condition causes a significant time overhead for the calculation of the non-scc forces.

C. Total Running Times. The performance scaling of an entire calculation is, for large systems, determined by the component with the worst scaling behavior. In DFTB+, there are currently two major components that are not implemented using linear scaling algorithms: diagonalization of the Hamiltonian and summation of the Coulomb interaction.

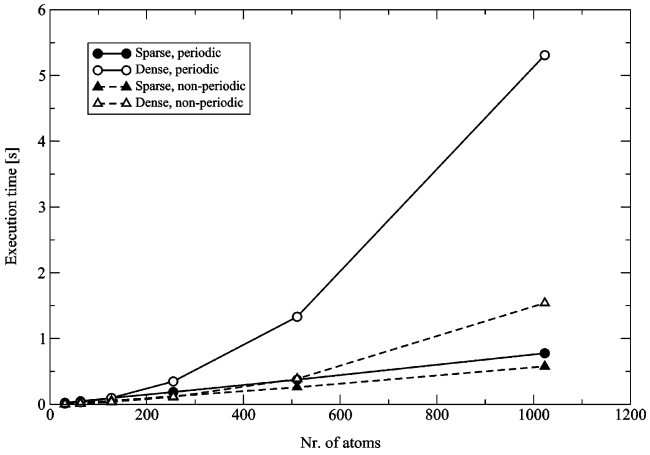

Figure 5. Performance of the non-scc force calculation for a silicon vacancy with different numbers of atoms around it. The filled circles and triangles indicate the results made with the sparse algorithm using periodic and nonperiodic boundary conditions, respectively. The empty circles and triangles show the results using the dense algorithm.

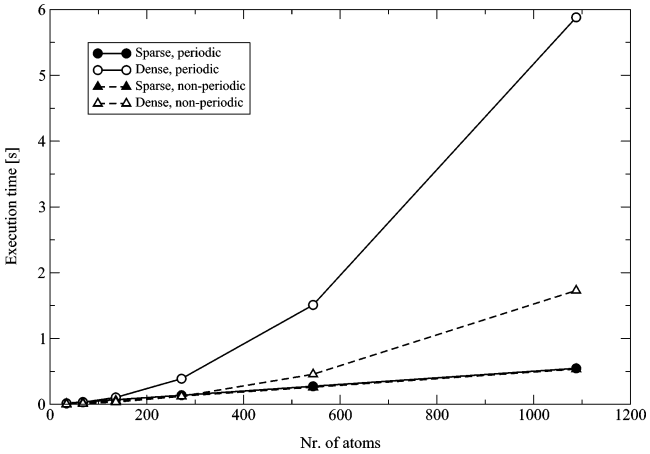

Figure 6. Performance of the non-scc force calculation for unrelaxed silicon nanowires containing different numbers of atoms. The diameter of the nanowires is the same (1.5 nm) in all calculations. The filled circles and triangles indicate the results made with the sparse algorithm using periodic and nonperiodic boundary conditions, respectively. The empty circles and triangles show the results using the dense algorithm. (The curve for the periodic sparse timing data overlays the nonperiodic sparse one.)

Diagonalization is currently performed using conventional square matrix techniques, as implemented in LAPACK. The sparse matrices are converted into the square form before diagonalization. Because the $O(N^3)$ cost of diagonalizing the Hamiltonian is the worst scaling step (in time) during a DFTB calculation, alternative approaches will also be offered in a near-term implementation. We plan to provide interfaces to various external libraries utilizing diagonalization in the sparse format. This should give better perfomance for extremely sparse systems. (However, bad scaling and the issue of quadratically scaling storage for the eigenvectors still remain.) Additionally, the alternative approach of replacing the diagonalization with an order-N method is also underway, with the divide and conquer[13] method being initially considered. Because this method had already been applied to DFTB,[8] we do not expect any difficulties with its implementation.

As shown in eqs 3 and 4, the scc contribution to the Hamiltonian matrix contains a sum with pure Coulombic $1/R$ terms. To calculate this contribution, the conventional Ewald summation is used. The scaling behavior for the construction of the potential is $O(N^2)$. Although the diagonalization, not the

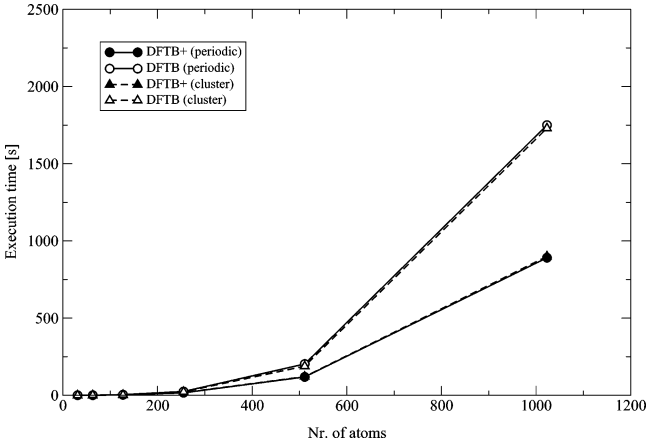

Figure 7. Performance of total energy calculations for a silicon vacancy in bulk silicon with different numbers of atoms around it. The filled circles and triangles indicate the results made with DFTB+ using periodic and nonperiodic boundary conditions, respectively. The empty circles and triangles show the results using the old DFTB code.

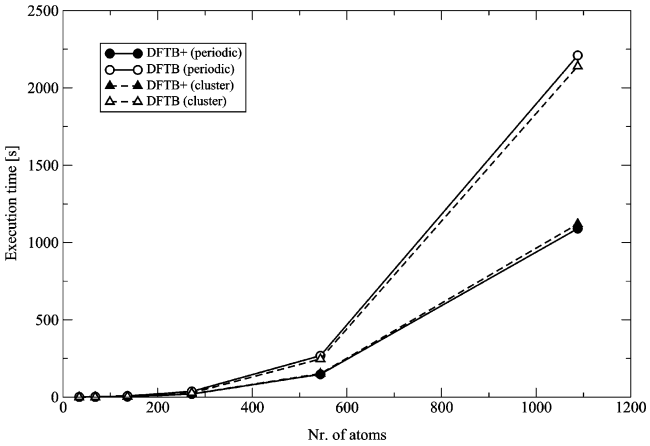

Figure 8. Performance of total energy calculations for unrelaxed silicon nanowires containing different numbers of atoms. The diameter of the nanowires is the same (1.5 nm) in all calculations. The filled circles and triangles indicate the results made with DFTB+ using periodic and nonperiodic boundary conditions, respectively. The empty circles and triangles show the results using the old DFTB code.

Ewald summation, is the real bottleneck in the DFTB+ code at the moment, a replacement of the current Ewald scheme with an $O(N \log N)$ scheme[14] is considered.

Despite the fact that the run time of the DFTB+ code for large systems is dominated by the diagonalization, the new structure of the code and its linear scaling components should already account for a significant performance improvement compared to that of the old DFTB code. In addition to the performance of the individual code components, we also investigated the running times for complete calculations. The test systems and the test environment were the same as for the non-scc force calculation tests. The calculations had been performed using self-consistent charges (scc) by using the same mixing algorithm with both codes (modified Broyden mixing[15]). Both codes performed 10 scc loops for every test case, followed by a subsequent (scc) force calculation. Figures 7 and 8 show the timing results for the silicon vacancy and the silicon nanowire, respectively.

The scaling of the total run time of both codes is similar. However, because of more efficient algorithms (apart from the diagonalization), DFTB+ heavily outperforms the old DFTB code for systems larger than 50−60 atoms. We re-emphasize that the tests presented here concentrated only on the performance improvements due to the algorithmic differences between the two implementations. We modified the old version to use the same diagonalizer as DFTB+; otherwise, the timings for the older code have been increased by almost 100%.

TABLE 1: Sparsity of the Relevant Matrices for Silicon Vacancies in Bulk Silicon and for Silicon Nanowires[a]

		sparsity	
system	no. of atoms	periodic	nonperiodic
V_{Si}	31	0.750	0.307
	63	0.375	0.182
	127	0.188	0.110
	255	0.094	0.063
	511	0.047	0.034
	1023	0.023	0.018
SiNW	34	0.531	0.531
	68	0.251	0.251
	136	0.125	0.125
	272	0.058	0.058
	544	0.030	0.030
	1088	0.015	0.015

[a] The sparsity is calculated as the size ratio of the sparse storage format and the conventional square matrix for the given system.

VI. Additional Features

An additional advantage of the sparse technique is the reduced storage for the Hamiltonian, overlap, and density matrices. This is especially crucial in those cases where many instances of those matrices must be stored at the same time. Typical examples are those cases where a mixing of the density matrices of subsequent iterations is required (e.g., LDA+U correction). In the case of a Broyden mixing scheme, the density matrices of all iterations must be stored, yielding a huge difference in the memory requirement between the conventional square and the sparse matrices. Table 1 shows the ratio between the size of the sparse storage format in DFTB+ and the size of the conventional square matrix for the test systems (for one matrix).

Besides the use of sparse matrices, the DFTB+ code offers several other features, some of them being unique among current DFTB implementations. It allows calculations with elements having angular momenta up to f. The code also allows collinear spin-polarized calculations (using the more recent spin formalism[5]) and arbitrary k-point sampling. To circumvent the deficiencies of LDA and GGA in treating elements with partially occupied d or f shells, the LDA+U method had been derived for the DFTB formalism and implemented using the sparse matrix structures presented here.[16] The code can also apply corrections for van der Waals and dispersion interactions,[17,22] which are also not properly described by LDA or GGA.

VII. Summary

We have developed a technique to store the matrices used in the DFTB method in a sparse format. The size of the matrices scales, for large systems, linearly with the number of atoms in the system. The stored matrices all are real for periodic as well as nonperiodic calculations. Many operations during a DFTB calculation can be directly evaluated in this sparse format, giving rise to a substantial speed gain. Additionally, the current formalism allows identical treatment of most operations for periodic and nonperiodic systems so that most of the routines need not account for the boundary conditions of the system. The applicability of the developed technique has been demonstrated in the DFTB+ code.

Acknowledgment. We thank Patrick Briddon (University of Newcastle) for fruitful discussions about the sparse storage format and Thomas Niehaus (University of Bremen) for helpful comments on the derivations. This work was partially funded by the European Commission through the RENIBEL (Rare Earths Doped Nitrides for High Brightness Electroluminescence Emitters) Network (contract no. HPRN-CT-2001-00297). B.H. gratefully acknowledges the Royal Society of Edinburgh BP Research Fellowship for funding.

References and Notes

(1) Porezag, D.; Frauenheim, T.; Köhler, T.; Seifert, G.; Kaschner, R. *Phys. Rev. B* **1995**, *51*, 12947.

(2) Seifert, G.; Porezag, D.; Frauenheim, T. *Int. J. Quant. Chem.* **1996**, *58*, 185.

(3) Elstner, M.; Porezag, D.; Jungnickel, G.; Elsner, J.; Haugk, M.; Frauenheim, T.; Suhai, S.; Seifert, G. *Phys. Rev. B* **1998**, *58*, 7260.

(4) Frauenheim, T.; Seifert, G.; Elstner, M.; Hajnal, Z.; Jungnickel, G.; Porezag, D.; Suhai, S.; Scholz, R. *Phys. Status Solidi B* **2000**, *217*, 41.

(5) Köhler, C.; Seifert, G.; Frauenheim, T. *Chem. Phys.* **2005**, *309*, 23.

(6) Saravan, C.; Shao, Y.; Baer, R.; Ross, P. N.; Head-Gordon, M. *J. Comput. Chem.* **2003**, *24*, 618.

(7) Zhang, H.; Smith, B.; Sternberg, M.; Zapol, P. *ACM Transactions on Mathematical Software*; Association for Computing Machinery: New York, 2007; p 33.

(8) Liu, H. Y.; Elstner, M.; Kaxiras, E.; Frauenheim, T.; Hermans, J.; Yang, W. T. *Proteins* **2001**, *44*, 484.

(9) Sternberg, M.; Galli, G.; Frauenheim, T. *Comput. Phys. Commun.* **1999**, *118*, 200.

(10) Challacombe, M. *J. Chem. Phys.* **1999**, *110*, 2332.

(11) Slater, J. C.; Koster, G. F. *Phys. Rev.* **1954**, *94*, 1498.

(12) Anderson, E.; Bai, Z.; Bischof, C.; Blackford, S.; Demmel, J.; Dongarra, J.; Du Croz, J.; Greenbaum, A.; Hammarling, S.; McKenney, A.; Sorensen, D. *LAPACK Users' Guide*, 3rd ed.; Society for Industrial and Applied Mathematics: Philadelphia, PA, 1999.

(13) Yang, W.; Lee, T.-S. *J. Chem. Phys.* **1995**, *103*, 5674−5678.

(14) Darden, T.; York, D.; Pederson, L. *J. Chem. Phys.* **1993**, *98*, 10089.

(15) Johnson, D. D. *Phys. Rev. B* **1988**, *38*, 12807.

(16) Hourahine, B.; Sanna, A.; Aradi, B.; Köhler, C.; Niehaus, T.; Frauenheim, T. *J. Phys. Chem. A*, submitted for publication, 2007.

(17) Elstner, M.; Hobza, P.; Frauenheim, T.; Suhai, S.; Kaxiras, E. *J. Chem. Phys.* **2001**, *114*, 5149.

(18) Karasawa, N.; Goddard, W. A., III. *J. Phys. Chem.* **1989**, *93*, 7320−7327.

(19) Chen, Z.-M.; Cagin, T.; Goddard, W. A., III. *J. Comput. Chem.* **1997**, *18*, 1365.

(20) DFTB+ is a DFTB implementation, which is free for noncommercial use. For details, see http://www.dftb-plus.info.

(21) The pbc-0-1 parameter set can be obtained from http://www.dftb.org.

(22) In periodic calculations, this is implemented using a Ewald method for long-range interactions.[18,19]