Time-dependent Density Functional-based Tight-bind Method Efficiently Implemented with OpenMP Parallel and GPU Acceleration†

Guo-hong Fan, Ke-li Han∗, Guo-zhong He

State Key Laboratory of Molecular Reaction Dynamics, Dalian Institute of Chemical Physics, Chinese Academy of Sciences, Dalian 116023, China

(Dated: Received on August 16, 2013; Accepted on October 29, 2013)

The time-dependent density functional-based tight-bind (TD-DFTB) method is implemented on the multi-core and the graphical processing unit (GPU) system for excited state calculations of large system with hundreds or thousands of atoms. Sparse matrix and OpenMP multithreaded are used for building the Hamiltonian matrix. The diagonal of the eigenvalue problem in the ground state is implemented on the GPUs with double precision. The GPU-based acceleration fully preserves all the properties, and a considerable total speedup of 8.73 can be achieved. A Krylov-space-based algorithm with the OpenMP parallel and GPU acceleration is used for finding the lowest eigenvalue and eigenvector of the large TDDFT matrix, which greatly reduces the iterations taken and the time spent on the excited states eigenvalue problem. The Krylov solver with the GPU acceleration of matrix-vector product can converge quickly to obtain the final result and a notable speed-up of 206 times can be observed for system size of 812 atoms. The calculations on serials of small and large systems show that the fast TD-DFTB code can obtain reasonable result with a much cheaper computational requirement compared with the first-principle results of CIS and full TDDFT calculation.

Key words: Density-functional theory, Tight-binding method, Time-dependent density functional theory, Excited state, Graphical processing unit, Krylov iterative algorithm, Sparse matrix, OpenMP

I. INTRODUCTION

Density functional theory (DFT) [1] is the most widely used first principle method among all electronic structure methods that requires fewer computer resources than other wave function-based theories and provides accurate results. For electronically excited states, time-dependent density-functional theory (TDDFT) [2–7] is widely used. TDDFT, a time-dependent extension of DFT, is an efficient method for the optical properties of finite systems and bulk materials. TDDFT generally scales O (N^4) along with the total basis size of the system, and the computation cost increases rapidly as the system size increases. For large systems, such as nano-sized systems of technical interest and biology systems with thousands of atoms, the DFT method becomes too time-consuming to be used, even if the resolution of the identity [8, 9] approximation is used to transfer the four-center integral into three-center integral. In recent years, the GPU device, which is formerly used for video games, has begun to capture many interests among scientists, as massive parallelization can be achieved within a single GPU device or some parallelized GPU device. The newly developed TeraChem program [10] is an example that is designed to utilize the GPU device to accelerate the most time-consuming part of DFT and TDDFT calculations. Large speed-up compared with a CPU-run code can be achieved which makes large-scale TDDFT calculations available. GPU-acceleration is successfully used in first principles methods for large-scale calculations. Semi-empirical methods, such as MNDO [11], AM1 [12], and PM3 [13] that generally fit to experimental results, are another option to be used for large systems. Although the semi-empirical methods are not as accurate as DFT and other ab initio methods, they are generally two to three orders of magnitude fast and are an alternative choice for large-scale simulations. However, the uses of these semi-empirical methods are limited because of their low accuracy or poor transference among different systems.

Trying to combine the accuracy of the first principle method and the speed of the semi-empirical method, while avoiding its defects such as poor accuracy and low transferences, the self-consistent charge density-
functional tight-binding (SCC-DFTB) method [14–20] which is derived from pure DFT is then developed. SCC-DFTB can be considered as a stationary approximation to the DFT [21–23] with all the parameters [24–26] fitted to DFT. Charge consistency is achieved with the definition of Mullikan charges which allows the charge rearrangement between various elements and improves the intermediate molecular charge transfer. The detailed derivation and formalism of the SCC-DFTB method have been addressed [14, 16, 27], many benchmark studies [14, 28–32] showing the great success and limitations of this method were conducted. The DFTB method can be extended to the time-dependent case using Casida’s linear response theory [4] to treat the excited state. The DFTB method which is derived from pure DFT method also inherits the limitations and defects of the pure DFT method. TD-DFTB method has limitations in dealing with high-lying electronic states of strong Rydberg character, intermolecular charge-transfer transitions [33], double excitations [34], and optical spectrum of solids [35]. Several of these problems have been solved by the newly proposed exchange-correlation (XC) functional, such as the range separated exchange-correlation functional [36–40] to treat the charge-transfer excitations. Other failures still need more sophisticated solutions. Despite these limitations, the TD-DFTB method can reach the accuracy close to that of pure TDDFT and can be two- to three-orders of magnitude faster than DFT method which makes the TD-DFTB method a promising approach for the excited states properties of large system.

In this work, we present a detailed implementation of the TD-DFTB method for excited properties of large molecular systems. A short communication has been given previously just to briefly show some of the results [41]. We briefly presented the DFTB method used for the ground state and the TD-DFTB method which is extended with Casida’s [3, 42] linear response approach for the excited states. To utilize the tight-binding nature of the tight-binding method and obtain high efficiency of the program, OpenMP multi-core parallelism and sparse matrix technique are used to build the Hamiltonian matrix elements. Given the massive parallelization capability of the GPU device, the most time-consuming part of the ground state to diagonalize the eigenvalue equation is accelerated with the GPU device. The large TDDFT eigenvalue problem of the excited state is implemented with a Krylov-space-based algorithm [43], which greatly reduces the number of the iterations taken for the eigenvalue problem. The large TDDFT matrix vector product which is the most time-consuming part of the total excited state run is parallelized with OpenMP and GPU acceleration to achieve the maxima speed-up. Comparison with experiment and the first-principle results on a series of small molecules are conducted to test the correctness of the method. Benchmarking tests on serials of large systems are conducted to examine the performances of the program in large systems.

II. METHODOLOGY
A. DFTB for the ground states

A detailed derivation of DFTB from pure DFT have been given in a number of papers including several reviews [14, 16, 27]. A brief description is also given in our pervious communication [41]. SCC-DFTB is derived using a Taylor series expansion of the DFT [1] total energy with respect to the charge density fluctuations around a given reference charge density and then truncated. The energy expansion can then be written as the parts that depend only on the reference charge density \( n_{\sigma} \) and the parts that depend on the density charge fluctuations \( \delta n_{\sigma} \) [14]:

\[
E_{\text{tot}} \approx \sum_{\sigma=\uparrow,\downarrow} \sum_{\text{oct}} n_{\sigma} \left< \Psi_{\sigma} \right| \hat{H}_0 \left| n_0, 0 \right\rangle \left< n_0, 0 \right| \Psi_{\sigma} \right> + \frac{1}{2} \int \int \left\{ \frac{1}{\mid \mathbf{r} - \mathbf{r}' \mid} + \frac{\delta^2 E_{\text{XC}}}{\delta \delta n^2} \right\} \delta n_{\sigma} \delta n'_{\sigma} d^3 \mathbf{r} d^3 \mathbf{r}' + O \left( \delta n^3 \right) + E_{\text{rep}} \tag{1}
\]

In the equation, \( H_0 \) is the Hamiltonian relating only to the reference density, \( E_{\text{XC}} \) is the exchange-correlation energy and \( E_{\text{rep}} \) is the repulsive energy. The first energy term is called the non self-consistent charge term (NONSCC), with parts of the energy expansion relating only to the reference density combined together. The second term is called the coulomb interaction and contains the exchange correlation contributions. All the remaining parts of the expansion are combined together to the third repulsive term [14]:

\[
E_{\text{rep}} = E_{jk} + E_{\text{XC}} \left[ n_{\uparrow}, 0 \right] - \int V_{\text{XC}} \left[ n_0, 0 \right] n_{\uparrow} d^3 \mathbf{r} - \frac{1}{2} \int \int \frac{n_{\uparrow} n_{\downarrow}}{\mid \mathbf{r} - \mathbf{r}' \mid} d^3 \mathbf{r} d^3 \mathbf{r}' \tag{2}
\]

Here, total energy expansion is cut off at the second order. The third order term can be kept which can substantially improve the description of the charged systems, especially with regard to hydrogen binding energies and proton affinities useful in bimolecular systems [20].

From the full Kohn-Sham DFT expressed on a local basis, SCC-DFTB undergoes several main approximations [44]. The DFT energy is expanded up to the second order with respect to charge density fluctuations around a given reference density. All three-center interaction integrals are neglected, and the two-center integrals are evaluated and tabulated as a function of interatomic distance. The molecular orbital is expressed in a minimal atomic basis set, and only valence electrons are considered. The short distance repulsive potential is fitted as a function of two body interactions. The
The Kohn-Sham energy difference to obtain the excited state energy can be estimated as an approximation of the excited analytical form of the gamma functional. With the above approximations the total energy of DFT can be finally briefly written as follows [14]:

\[ E = \sum_{\mu} \sum_{i} c_{\mu i} E_{\mu i} + \frac{1}{2} \sum_{AB} \Delta q_{AB} \gamma_{AB} \Delta q + E_{\text{cusp}} \]  

(3)

Applying the variation principle, the eigenvalue equation is then written as [14]:

\[ \sum_{\nu} (H_{\mu \nu} c_{\nu i} - \varepsilon_{i} S_{\mu \nu} c_{\nu i}) = 0 \]  

(4)

\[ H_{\mu \nu} = H_{\mu \nu}^0 + \frac{1}{2} \sum_{C} (\gamma_{AC} + \gamma_{BC}) \Delta q_{C} \]  

\[ \mu \in A, \quad \nu \in B \]  

A. Sparse matrix and GPU speed-up in the ground state

For large systems, a large part of the matrix becomes zero and a sparse matrix algorithm can be used to reduce the memory and CPU cost. The Hamiltonian matrix and the overlap matrix in the ground state eigenvalue problem are built and updated in a compressed sparse-row format. The matrix is transformed to a square form during the diagonalization of the eigenvalue problem. To speed up the tight-binding code, the OpenMP multi-threaded parallel technique is used to build up the sparse matrix in the multi-core architecture. This reduces the computational cost of building the Hamiltonian matrix element into a small portion of the total self-consistent ground-state run. The main computational time is then spent on the diagonalization of the eigenvalue equation Eqs.(4) and Eqs.(5). As the system size increases, it becomes the most time-consuming. The MAGMA library is a dense linear algebra library similar to Linear Algebra Package (LAPACK) [48], but it is used for the current “Multi-core+GPU” heterogeneous/hybrid architectures. To exploit the acceleration capability of GPU, the diagonal LAPACK routine DSYGVD is replaced with the MAGMA diagonal routine MAGMADSYGVD, which runs on a hybrid parallel architecture with the GPU device. Double precision is used throughout the program and all the properties can be preserved.

B. Krylov-space-based algorithm to find the lowest eigenvalues and eigenvectors of the large TDDFT matrix

For large systems, the dimension of the TDDFT matrix in Eq.(6) scales as \( N^4 \). Specifically, the dimension of the matrix is \( (N_{\text{occ}}N_{\text{Vac}})^2 \), where \( N_{\text{occ}} \) is the number of the selected occupied Kohn-Sham orbitals and \( N_{\text{Vac}} \) is the number of virtual orbitals. The detailed derivation of the approach can be found in Casida’s study [3, 42]. The resulting equation is as follows:

\[ \sum_{j b} \left[ \omega_{ia}^2 \delta_{ij} \delta_{ab} + 4 \omega_{ia} K_{ia,jb}^{\Sigma} \right] \psi^{I \Sigma}_{j b} = \frac{\Omega_{I \Sigma}}{\Omega_{I \Sigma}} F^{I \Sigma}_{ia} \]  

(6)

In these equations, \( \gamma_{\mu \nu} \) is the same gamma function as that in the ground case, \( m_{\mu \nu} \) is a quantity similar to \( \gamma_{\mu \nu} \). Both quantities can be calculated from atomic DFT calculations. Eq.(6) is diagonalized to obtain the singlet and triplet excited state energies. An ordinary diagonalization method requires a number of operations proportional to \( N^2 \), where \( N \) denotes the number of basis functions. However, as only low-lying excited states are of common interest, a kind of iterative diagonal method such as the Davidson method [46] can be used as the eigensolver.
ground state and $N_{\text{occ}}$ is the number of the selected unoccupied orbitals. An ordinary diagonal method, such as that in the LAPACK, requires the whole matrix to be stored in the memory and a number of operations proportional to $N^6$. As the system becomes larger, the dimension of the problem becomes huge, and it takes too long time for the eigenvalue problem. For the TDDFT eigenvalue problem, only several lowest eigenvalues and eigenvectors are usually required. A simple iterative method is preferred to calculate the eigenvectors without explicitly constructing the TDDFT matrices. In the following, a Krylov-space-based algorithm is used to solve the eigenvalue problem.

The eigenvalue Eq.(6) can be written as

$$HF_I = \lambda_I F_I$$

which is a kind of matrix-vector product with a large dimension, $\lambda = \Omega^2_{I \Sigma}$ is the eigenvalue and $F_I$ is the eigenvector. The matrix vector-product can be further written as

$$F_I = \omega_{ib} \delta_{ib} + 4 \sqrt{\omega_{ia}} K^\Sigma_{ia,jb} \sqrt{\omega_{jb}} F^\Sigma_{jb}$$

where $K^\Sigma = K_{\Sigma}^\Sigma$. The Krylov-space algorithm [43, 46, 49] finds several lowest eigenvalues of the TDDFT matrix $K$ by starting with an arbitrary eigenspace with dimension $l (l \leq N)$ and constructing the orthogonal basis $(K_1, K_2, \ldots, K_l)$ [41]. The trial vector and the eigenspace at some iteration are expanded in a linear combination of the orthogonal vectors.

$$K_n = \sum_{i=0}^{n-1} \alpha_i K_i$$

The linear combination coefficients $\alpha_i$ in the above equation are found using the Ritz variational procedure, leading to the projected eigenvalue equation

$$K_n H K_n \alpha = \lambda \alpha$$

This equation guarantees to yield the best approximation to the lowest exact eigenvalue of matrix that belongs to a Krylov subspace [49]. The subspace ($K_n$) spanned by the vectors $K_0, \ldots, K_n$, approximates an invariant subspace of the whole matrix $H$ with increasing accuracy as the number of vectors increases. The dimension of the eigenvalue equation is much lower than the original TDDFT eigenvalue problem, and it can be solved more efficiently. The projected eigenvalue equation is then solved, and the eigenvalue is considered closer to the true eigenvalue. New subspace vectors orthogonalized to the previous subspace vectors are added to the subspace. The whole procedure is repeated until the convergence is reached [41].

The algorithm requires the formation of the matrix-vector product in each iteration step. For large matrices, this step is the most time-consuming part of the whole procedure, and it can be parallelized to obtain maximum performance. As only the subspace vectors need to be stored in the memory, the memory required for the Krylov-space-based algorithm is much lower than that for the traditional diagonal method. A kind of subspace collapse technique similar to the restart scheme used in the Lanczos method [50] is further used to reduce the memory requirement. The idea is to choose the optimal approximate eigenvectors already obtained and to restart the procedure with an initial subspace expanded by these eigenvectors. Another important aspect of the iterative algorithm is the choice of the start initial guess vector which may have large impact on the convergence rate of the algorithm. The algorithm may converge to a wrong result or take a large number of iterations to converge if improper initial vectors are used. A random guess vector can be used as the initial vector. For the TDDFT eigenvalue problem, as the major contribution of the transition frequencies comes from some filled and unoccupied molecular orbitals, the Kohn-Sham orbital energy difference can be used as an effective initial value for the excited state. The Kohn-Sham orbital energy difference is then sorted, and the lowest is chosen as the initial guess eigenvalue. The initial eigenvectors are set as one component of the initial eigenvector to one, and the others set to zero. The diagonal approximation corresponds to the HOMO-LUMO approximation for the transition. Comparison with the commonly used package Arpack [51] for several eigenvalues of large matrix is shown in the performance section.

C. OpenMP and GPU implementation of the TDDFT matrix vector product

The OpenMP implementation of the matrix vector product in Eq.(6) is simple. The second part of the matrix vector product in the equation is conducted as follows [41]. Step 1: the temp product with the initial eigenvector for each atom, which is calculated as,

$$\tau_B = \sum_{jB} q^b_{jB} \sqrt{\omega_{jb}} F_{jb}$$

Step 2: the product of the gamma matrix $\gamma_{AB}$ with temp vector $\tau_B$ is produced,

$$\zeta_A = \sum_B \gamma_{AB} \times \tau_B$$

Step 3: the dot-like product of $\sqrt{\omega_{ia}} \sum_A Q^i_A \zeta_A$ is conducted for each $ia$. 

DOI:10.1063/1674-0068/26/06/635-645

©2013 Chinese Physical Society
The first step for the OpenMP can be parallelized by adding the OpenMP parallel reduction directive before the \( \text{j} \) loop. The parallel of the matrix vector product of step 2 is implemented by the OpenMP version of the LAPACK routine DSYMV in the Intel Math Kernel Library. The parallelization of the last step of the matrix vector product is implemented using the OpenMP parallel do loop directive. We have also tried to utilize GPU to accelerate the matrix vector product. For the parallel reduction of step 1, the GPU acceleration becomes something complicated. The reduction is split into two step sum. The first step sum is a partial sum over \( \text{j} \) loop with a CUDA (NVidia Compute Unified Device Architecture) kernel launched with a proper block and grid size. Each thread in the block does a step in the \( \text{j} \) loop for each atom. For easy implementation the block size is chosen as the power of 2 and 512 can be a good choice. With the result of each thread stored in the shared memory of the block, a kind of tree reduction for each block adds the result to a partial sum. The sum of the partial sum of each block is then done in the second step with a kernel launch of only one block. The block size in the second step has to be the same as the grid size in the first step. The matrix vector product of step 2 is implemented by the GPU LAPACK routine CUBLASDSYMYV in CUBLAS [52]. For the last step of the GPU implementation we just have to launch another kernel with proper block and grid size over the \( \text{i} \) loop to utilize the acceleration capability of GPU. However, the large matrix vector is memory read and writes intensive, not computational intensive. In system size with thousands of atoms, the dimension of the Mullikan transition charge becomes too huge to be stored in the GPU device memory. The transition charge is then calculated whenever it is required. A number of matrix elements for the matrix vector product have to be fetched from the device memory which makes the matrix vector product a bank conflict problem. For the bank conflict reason the GPU matrix vector product can accelerate the matrix vector product by 4–5 times compared with the serial code run on an Intel Xeon E5620 CPU.

In the Krylov-space-based algorithm, the original large TD-DFT eigenvalue problem is projected onto the “Krylov subspace”. The dimension of the projected eigenvalue problem is much lower than the original TD-DFT eigenvalue problem. Only the projected eigenvalue problem has to be solved on the GPU device. A limited size of the excited state eigenvector (several times of the number of the excited states required), the overlap matrix, Hamiltonian matrix and the density matrix of the ground state has to be stored in the GPU memory. For example, for a molecular with \( 10^3 \) atoms, the size of the ground state Hamiltonian matrix is about \( (4 \times 10^3)^2 \). The size of an excited state eigenvector is about \( (2 \times 10^3)^2 \), and the whole excited state matrix is in the huge size of \( (2 \times 10^3)^2 \times (2 \times 10^3)^2 \). Because the whole excited state matrix needs not to be stored in the GPU memory, the GPU implemented TD-DFTB Krylov-space-based algorithm can efficiently reduce the memory requirement and handle the excited state problem.

IV. TD-DFTB CALCULATIONS OF SMALL AND LARGER MOLECULES

To test the accuracy of the program, we implement the method on a series of small organic molecules with well-known experimental absorption spectra and first principle results. As there are numerous tests for the ground-state DFTB method, the current study only presents the results for the excited states. The molecular structures in the calculations below are optimized at the SCC-DFTB level. The self-consistency charge error of the tests is set to \( 1.0 \times 10^{-6} \). The molecular geometry is considered at equilibrium with the force error of \( 1.0 \times 10^{-5} \). The Mio-0-1 [14] parameter set is used for all calculations. The calculation for these systems takes less than one second, whereas the first-principle calculation takes tens of minutes. Table I shows the excited state energy differences. Results for the experiment and the first principle are obtained from Ref.[45]. The mean absolute error of the first-principle TDDFT result with respect to the experimental data is 0.36 and 0.37 eV for the singlet states and triplet states. The error is slightly larger for the TD-DFTB method, but the difference is small, with 0.40 and 0.67 eV respectively. The sign error is \(-0.24 \) and \(-0.36 \) for the TDDFT method and 0.06 and 0.61 eV for the TD-DFTB method. The sign error of the triplet states is larger than that of the singlet state. Table I also indicates that transition \( n \rightarrow \pi^* \) has a larger error than transition \( \pi \rightarrow \pi^* \).

We then implement the TD-DFTB code on the excited state energy of system with hundreds and thousands of atoms to investigate the capability of the TD-DFTB code for excited properties of large systems. To compare the full TDDFT result in large systems, we need an efficient TDDFT code. As massive parallel can be achieved with the GPU device, the full TDDFT result of large systems is obtained from Isborn et al.’s result [10] which use GPU to accelerate the full TDDFT calculation. Their full \( \text{ab initio} \) excited-state calculations are conducted with a development version of the GPU-based TeraChem software package using the configuration interaction singles (CIS) and the adiabatic linear response of Tamm-Dancoff time-dependent DFT [53]. The GPU-based TeraChem calculations by Isborn et al. are run on platform with dual quad-core Intel Xeon X5570 CPUs, 72 GB RAM, and 8 Tesla C1060 GPUs. The 6-31G basis set is used in the \( \text{ab initio} \) calculation. No d-orbitals are included as the d-function GPU integral routine is not yet implemented. In what follows, we apply our TD-DFTB code to the same systems calculated by the GPU TDDFT. The following tests are evaluated on a variety of systems [10]: four generations of oligothiophene den-
TABLE I The excited state energies calculated by the TDDFT method which provides the first-principles result [45] and the TD-DFTB method (this work) in comparison with experimental results [45]. The signed mean error and the absolute mean error with respect to the experimental data are shown at the bottom of the table.

<table>
<thead>
<tr>
<th>Compound</th>
<th>State</th>
<th>Excited state energy/eV</th>
<th>Exp.</th>
<th>TDDFT</th>
<th>This work</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ethylene</td>
<td>$^3$B$_{1u}(\pi \rightarrow \pi^*)$</td>
<td>4.40</td>
<td>4.16</td>
<td>5.45</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$B$_{1u}(\pi \rightarrow \pi^*)$</td>
<td>7.65</td>
<td>7.44</td>
<td>7.79</td>
<td></td>
</tr>
<tr>
<td>Propene</td>
<td>$^3$A$^\prime$(\pi \rightarrow \pi^*)</td>
<td>3.95</td>
<td>5.13</td>
<td>5.13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^1$A$^\prime'(\pi \rightarrow \pi^*)$</td>
<td>6.36</td>
<td>6.96</td>
<td>7.13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^1$A$^\prime$(\pi \rightarrow \pi^*)</td>
<td>7.19</td>
<td>6.69</td>
<td>7.13</td>
<td></td>
</tr>
<tr>
<td>Butadiene</td>
<td>$^3$B$_{u}(\pi \rightarrow \pi^*)$</td>
<td>3.20</td>
<td>2.83</td>
<td>3.66</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$Ag(\pi \rightarrow \pi^*)</td>
<td>4.95</td>
<td>4.91</td>
<td>5.61</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$B$_{u}(\pi \rightarrow \pi^*)$</td>
<td>5.92</td>
<td>5.43</td>
<td>5.52</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$B$_{g}(\pi \rightarrow \pi^*)$</td>
<td>6.08</td>
<td>6.41</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^1$A$_{u}(\pi \rightarrow \pi^*)$</td>
<td>5.80</td>
<td>6.16</td>
<td>6.43</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^1$A$_{g}(\pi \rightarrow \pi^*)$</td>
<td>6.36</td>
<td>5.57</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cyclopropene</td>
<td>$^3$B$_{2u}(\pi \rightarrow \pi^*)$</td>
<td>4.16</td>
<td>3.77</td>
<td>4.78</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^1$B$_{2u}(\pi \rightarrow \pi^*)$</td>
<td>6.45</td>
<td>6.00</td>
<td>6.14</td>
<td></td>
</tr>
<tr>
<td>Benzene</td>
<td>$^3$B$_{1u}(\pi \rightarrow \pi^*)$</td>
<td>3.89</td>
<td>3.90</td>
<td>4.72</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$E$_{1u}(\pi \rightarrow \pi^*)$</td>
<td>4.85</td>
<td>4.55</td>
<td>5.06</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$B$_{2u}(\pi \rightarrow \pi^*)$</td>
<td>5.69</td>
<td>4.87</td>
<td>5.30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$B$_{2u}(\pi \rightarrow \pi^*)$</td>
<td>4.89</td>
<td>5.17</td>
<td>5.30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$^3$B$_{1u}(\pi \rightarrow \pi^*)$</td>
<td>6.20</td>
<td>5.91</td>
<td>5.68</td>
<td></td>
</tr>
</tbody>
</table>

| Pyridine     | $^3$B$_1(n \rightarrow \pi^*)$ | 4.10 | 3.71 | 4.48 | |
|              | $^3$B$_2(n \rightarrow \pi^*)$ | 4.84 | 4.36 | 4.92 | |
| Formaldehyde | $^1$B$_1(\pi \rightarrow \pi^*)$ | 4.51 | 4.37 | 4.48 | |
|              | $^1$B$_2(\pi \rightarrow \pi^*)$ | 5.00 | 5.28 | 5.39 | |
| Ketene       | $^1$A$_1(\pi \rightarrow \pi^*)$ | 6.45 | 5.94 | 5.81 | |
|              | $^1$A$_2(\pi \rightarrow \pi^*)$ | 7.23 | 6.48 | 7.01 | |
| Propynal     | $^3$A$^\prime'(n \rightarrow \pi^*)$ | 2.99 | 2.74 | 4.09 | |
| Glyoxal      | $^3$A$_{u}(n \rightarrow \pi^*)$ | 3.56 | 3.37 | 4.09 | |
|              | $^3$A$^\prime'(n \rightarrow \pi^*)$ | 2.38 | 1.53 | 2.45 | |
|              | $^3$A$_{u}(n \rightarrow \pi^*)$ | 2.73 | 2.14 | 2.45 | |

Abs. err. singlet | 0.36 | 0.40 | |
Sign err. singlet | -0.24 | 0.06 | |

Abs. err. triplet | 0.37 | 0.67 | |
Sign err. triplet | -0.36 | 0.61 | |

run with a charge error of $1.0 \times 10^{-6}$ converge in a number of circles less than 20. However, some molecules such as the test case S15 and S16 also have an over underestimated excited state energy. This may be caused by the general problem of TDDFT unable to correctly describe the transitions with charge transfer characteristic [60]. The TD-DFTB result generally lies between the result of the CIS and that of the TD-B3LYP. The DFTB method employs Slater-Koaster parameters that are fitted to pure DFT with some kind of exchange correlation functional. The mio parameter set we used are fitted to DFT using the Perdew-Burke-Ernzerhof (PBE) [61] functional. Similar to other conventional local exchange-correlation functionals, the PBE functional may exhibit an incorrect asymptotic behavior. The TD-DFTB result being lower than that of TD-B3LYP may be explained as the TDDFT with a standard non-hybrid exchange-correlation functional generally underestimates excitation energies, especially for Rydberg states. While the incorporation of a hybrid functional into DFTB is not straightforward, the situation may be improved by the addition of a range of characteristics in the TD-DFTB method, such as the range-separated exchange functional that separates the exchange portion of the DFT functional into long and short-range contributions [36–40]. As the system size grows, the TD-DFTB excited energy has the same decreasing tendency as the widely used

DOI:10.1063/1674-0068/26/06/635-645 ©2013 Chinese Physical Society
FIG. 1 Structures of four generations of oligothiophene dendrimers S1–S4 and the number of atoms.

FIG. 2 Structure of a PYP chromophore in vacuum (S6) and five solvated PYP structure of the test with an increasing number of solvated waters [41].

B3LYP functional. The obtained accuracy is generally higher than that of the first-principle CIS calculations. TD-DFTB can reach an excited state energy close to that of the TD-B3LYP, with much less time and memory requirement.

V. PROGRAM PERFORMANCE AND BENCHMARKING

We conduct some benchmark tests on different systems with different programming techniques to test the program performance. The sparse matrix and OpenMP implementation have been described above. The OpenMP parallel implementation of the square matrix diagonal is conducted using the dynamically linked Intel Math Kernel Library to provide the required BLAS (Basic Linear Algebra Subprograms) and LAPACK functions. Figure 3 shows the program performance on a machine with an 8-core Intel Xeon E5620 CPU and an NVIDIA Tesla C2050 GPU card [41]. Various programming techniques are employed for the ground state. The calculation is a ground-state run of three self-consistency steps on a 270 angstrom long carbon nanotube with 2560 atoms. As shown in Fig.3, different speeds can be obtained by enabling various
program skills. The timing result indicates the importance of a quick DFTB implementation. The best speed-up code is achieved with the GPU eigensolver, sparse matrix and OpenMP parallel. The speed-up is 8.73 compared with the serial code on CPU without sparse matrix. A speed-up of 5.52 is obtained when the GPU is not accelerated, and a speed-up of 4.43 is obtained without the GPU and sparse matrix. It also shows that each programming technique contributes to the total speed-up of 8.73. Tests using other combinations of program skills can provide more assessments. To test the linear scaling properties of the ground-state DFTB calculation as the system size increase, we conduct some tests on the carbon nanotube with different
FIG. 5 Timing information of the excited-state calculation with OpenMP parallel compared with the serial code on CPU. The tests are run on a platform with an 8-core Intel Xeon E5530 CPU and 24 GB RAM.

TABLE III Time $T$ spent on the TD-DFTB excited state eigenvalue problem and the corresponding number of matrix-vector product $N$ taken for molecules from S7 to S13. The Arpack solver is run on an 8 core Intel(R) Xeon(R) CPU E5620 with the OpenMP parallel. The Krylov solver is run on an NVIDIA Tesla C2050 GPU card for GPU acceleration. The speed-up of Arpack solver compared with the Krylov solver is shown in the last column.

<table>
<thead>
<tr>
<th>System (atoms)</th>
<th>$T$ (s)</th>
<th>$N$</th>
<th>Speed-up</th>
</tr>
</thead>
<tbody>
<tr>
<td>S7(71)</td>
<td>2.8</td>
<td>1233</td>
<td>344</td>
</tr>
<tr>
<td>S8(149)</td>
<td>28.6</td>
<td>2447</td>
<td>231</td>
</tr>
<tr>
<td>S9(218)</td>
<td>75.6</td>
<td>2344</td>
<td>32</td>
</tr>
<tr>
<td>S10(332)</td>
<td>504.1</td>
<td>4750</td>
<td>38</td>
</tr>
<tr>
<td>S11(467)</td>
<td>1205.5</td>
<td>3691</td>
<td>26</td>
</tr>
<tr>
<td>S12(605)</td>
<td>3413.2</td>
<td>3937</td>
<td>27</td>
</tr>
<tr>
<td>S13(812)</td>
<td>17797.3</td>
<td>7442</td>
<td>21</td>
</tr>
</tbody>
</table>

FIG. 6 The timing report and speed-up of the first three excited-state calculations for S7–S13 with different iterative methods. (a) Speed-up, (b) Arpack solver run time with 8 core E5620 CPU, and (c) Krylov solver run time on Tesla C2050 card.

The most time-consuming part of the matrix vector product for the excited-state calculation in the TD-DFTB implementation is accelerated with the multi-core OpenMP parallelization. We conduct tests while running the program on different numbers of CPU cores using the OpenMP matrix vector parallel and Krylov-space-based iterative methods. Figure 5 shows the OpenMP multi-threading acceleration property of the program. The test system is the oligothiophene dimers molecular S4, which is shown in Fig.1(a) and contains 636 atoms. The total excited-state run time with different numbers of CPU cores is shown in Fig.5. Compared with the serial run, the parallel speed-up is generally linear as the number of core increases. The best multi-core speed up is 6.9, which is achieved using 8 CPU cores. The scaling property remains linear at 8 cores and shows no declinance. More speed-ups can be expected with more than 8 cores, indicating that OpenMP is efficient in parallelizing the matrix vector product. The excited-state calculation can be greatly speed-up by the multi-core multi-threading programming.

The Arpack package is generally used for the computation of the lowest or largest eigenvalues of large matrices, which uses the implicitly restarted Arnoldi/Lanczos method [51]. Table III, Fig.6, and Fig.7 shows the time spent on the TD-DFTB excited state eigenvalue problem and the corresponding number of matrix-vector product for the first three excited-state of the series of molecules from S7 to S13 using the Krylov-space-based method and the Arpack package. The convergence thresholds (norm of the residual vector) of both iterative solvers are set to $10^{-5}$. The Arpack solver is run on an 8 core Intel(R) Xeon(R) CPU E5620 with the OpenMP parallel. The Krylov solver is run on an NVIDIA Tesla C2050 GPU card for GPU acceleration of the matrix vector product. The speed-up of Arpack solver compared with the Krylov solver is also shown in Table III. A remarkable speed-up of about 206 times can be observed for the Krylov solver with the GPU acceleration for system with 812 atoms. The speed-up comes from both the reduction of number of matrix-vector product with the Krylov solver and the parallelization with the GPU device. Compared with the commonly used Arpack package, the block Davison method uses fewer iterations and matrix vector products. The TDDFT eigenvalue equation corrects the Kohn-Sham orbital energy difference KS with the cou-
The corresponding number of matrix vector products taken for TD-DFTB excited states of system from S7 to S13 with different iterative methods. The convergence threshold (norm of the residual vector) of both methods is set to $10^{-5}$.

Sampling matrix in Eq.(10) and Eq.(11). The initial guess vector of the HOMO-LOMO approximation that represents the most dominant transition presents to be a good choice. This is another cause of the reduction in the number of matrix vector products. Figure 8 shows the scaling property of the program as the basis size increases [41]. The number of atoms for the test molecules from S7 to S13 increases with the addition of solvated waters. As the systems size increases, the time for the Krylov solver increases slowly, whereas the time for the Arpack solver increases rapidly. The time for the excited state increases only in the quadratic trend as the number of basis size increases. The TD-DFTB excited state with the Krylov-space-based method can obtain excited energy within tens of minutes for systems with thousands of atoms, much faster than the GPU-CIS calculation that uses the minimal 6-31G basis set.

VI. CONCLUSION

In this work, the implementation of the TD-DFTB method and performance for large systems are presented. A new computational code that implements the DFTB/TD-DFTB method is developed. The ground-state DFTB is implemented with the OpenMP and sparse matrix technique, which minimize the time required to build the Hamiltonian matrix and the density matrix. The most time-consuming part in DFTB calculation to diagonalize the eigenvalue problem is implemented on GPU, which shows good performance over the serial code on CPU. The calculation of a 270 Å carbon nanotube with 2560 atoms shows that the sparse matrix and the OpenMP multithreading accelerate the program by 5.52 times on an 8-core CPU while the GPU acceleration can further accelerate the code to 8.73. For the excited states, the Krylov-space-based iteration method with the GPU accelerated matrix-vector product can converge quickly to obtain the final result.

A notable speed-up of 206 times can be observed for system of 812 atoms. The whole scale property of the program can increase in the quadratic trend as the number of basis size increases, which enables the excited-state calculation of hundreds or even thousands atoms to be feasible. The implementation of the TD-DFTB method on a series of large systems shows that TD-DFTB can obtain excited energy with better accuracy than CIS and a reasonable result close to that of B3LYP and BLYP. The excited-state result of the solvated PYP chromophore also indicates the well-known problem of the TD-DFT in treating charge-transfer excitations. As the system becomes larger, the problem becomes more severe. The excited state energies calculated by the TD-DFTB method come close to the HOMO-LOMO energy gap of the corresponding Kohn-Sham molecular orbitals as the system size increases. This may originate from the problems of TD-DFT itself or the simplification made in the tight-binding approximation. Further research is still required to resolve these problems.

Supplementary material: The geometry Cartesian coordinates, the GPU version of CIS and the full TD-DFT result in TABLE II taken from Ref.[10] for comparison are shown.
