Computation of Si Nanowire Bandstructures on Parallel Machines Through Domain Decomposition

Tao Li[1], Ximeng Guan[1], Zhiping Yu[1], and Wei Xue[2]

[1] Institute of Microelectronics, Tsinghua University, Beijing 100084, China
{litaofrank99, gxm}@mails.tsinghua.edu.cn,
yuzhip@tsinghua.edu.cn
[2] Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
xuewei@tsinghua.edu.cn

Abstract. This paper presents a methodology for calculating silicon nanowire (SiNW) bandstructures on parallel machines. A partition scheme is developed through domain decomposition and loading balance is considered for scheduling jobs on machines with different efficiencies. Using $sp^3d^5s^*$ tight-binding model, the Hamiltonian matrix of the SiNW is constructed and its eigenvalues are extracted with the Implicitly Restarted Arnoldi Method. Parallel calculation performance is tested on identical machines finally and a linear speedup is gained as the number of nodes increases.

1 Introduction

The future's scaling down of device size based on conventional silicon technology is predicted to face fundamental physical limits in the near future. In order to achieve the device miniaturization into nanometer scale, it is expected that conventional device structures should be modified or totally altered. With the rapid progress in nanofabrication technology, SiNWs with small diameters (<20 nm) have been synthesized and extensively studied for potential applications in nanoelectronics [1][2]. Due to the strong quantum confinement (QC) in ultrathin SiNWs, atomic bandstructure effects [3] are expected to play an important role on their device characteristics.

For modelling the nanoscale device characteristics, any realistic electronic model, whatever the underlying basis, must accurately reproduce the experimentally verified band gaps and effective masses for the relevant bands. The strength of tight-binding (TB) techniques is their ability to properly model such crucial material properties with a localized, atomistic orbital basis [4]. TB models have therefore developed into the primary choice for many researchers interested in the quantitative modelling of electronic structure on a nanometer scale. For the accurate description of the conduction subbands of the nanowire, it is necessary to include higher orbitals beyond the minimal sp^3 basis [4] and to employ the

V.N. Alexandrov et al. (Eds.): ICCS 2006, Part I, LNCS 3991, pp. 250–257, 2006.

$sp^3d^5s^*$ TB model, which was developed by Jancu *et al.* [5] and was found to describe quite well up to the two lowest conduction bands and as well as the valance bands of bulk Si.

Application of $sp^3d^5s^*$ TB model on SiNW bandstructures computation covers a broad spectrum of Brillouin zone and energy bands of interests [6]. In many practical cases, physical domains, which require a huge number of finite difference computational grids, are inevitable, and a complex nanowire cross-section is needed. In such circumstances, not only the PC memory size is too small to carry out the computations, but also a lot of CPU time is required. To facilitate such computational demand, tasks can be distributed to several machines so that each node is responsible for a smaller sub-task only. This idea underlies the present work of parallelizing the calculation of SiNW bandstructures. The implementation of the proposed parallel algorithm through domain decomposition on the Itanium-2 cluster is done by employing the commonly used message passing interface (MPI) library [7].

This paper is organized as follows. Section 2 presents the partition scheme through domain decomposition, followed by Section 3 which gives the algorithm for scheduling jobs on parallel machines with different efficiencies. In section 4, the computation for energy levels with $sp^3d^5s^*$ TB model on a single node machine is explained in detail. In Section 5, the bandstructures for SiNWs oriented along [112] direction with different cross-section shapes are computed and the performance is evaluated. Finally, a conclusion is given to close this paper.

2 Partition Scheme

A detailed calculation of the SiNW bandstructures in Brillouin zone needs a large number of finite difference computational grids, i.e. energy levels on a long list of $\boldsymbol{k}$ values are needed. Once the energy levels on each $\boldsymbol{k}$ point are carried out, the whole bandstructure could be charted by connecting the corresponding energy level respectively. Considering this, the partition scheme is straightforward developed through domain decomposition, and a master/slave program architecture is adopted for the parallel implementation.

On the first stage, a list of $\boldsymbol{k}$ values in the Brillouin zone are selected by the master node, then they are decomposed and distributed to the slave nodes in the cluster. When the energy levels computation is finished on one slave node, the results are sent back to the master and a new $\boldsymbol{k}$ value will be assigned. As shown in Fig. 1, this procedure continues until all points have been sent out and then the whole bandstructure is charted. In this partition scheme, communication occurs only between the master and slave nodes during the parallel computation, consequently, the partition scheme has a good scalability when grid size increases.

Another important aspect in parallel calculation is loading balance: for an urgent need for speed, jobs should be assigned to the slave nodes according to their computation abilities. In Sect. 3, We will show that this optimizing problem is polynomially solvable in our circumstance.

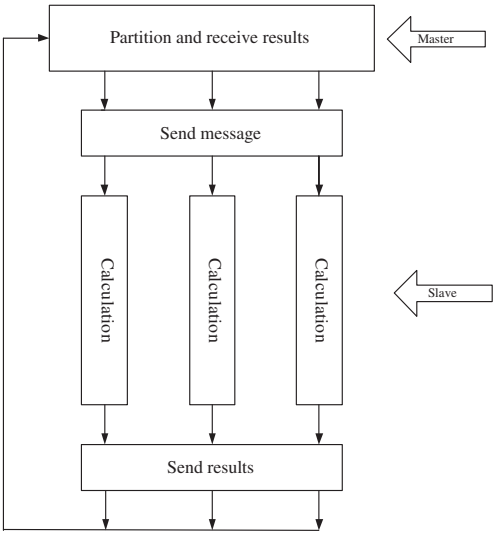

Fig. 1. Overview of the parallel calculation process

3 Scheduling Jobs on Parallel Machines

In this section, we consider the scheduling problem where n jobs $J_1, \ldots, J_n$ have to be processed by m machines with different efficiencies. Each job is described by a release date r_i, a processing time p_i and a cost function $f_i(t)$. This function represents the cost induced by J_i when it is finished at time t. The problem of minimizing the sum of the f_i functions consists of finding a set of completion times C_i for each job J_i such that:

- jobs start after their release date, i.e., $\forall t, C_i - p_i \geq r_i$,
- no more than m machines are used at any time t, i.e., $\forall t, |\{J_i | | C_i - p_i \leq t < C_i\}| \leq m$,
- the objective function $\sum_i f_i(C_i)$ is minimal.

This problem, denoted as the $(P|r_i \sum f_i(C_i))$ in the standard scheduling terminology (e.g., [8]) is a generalization of several NP-hard scheduling problems. However, we refer to Brucker and Knust [9] for up to date complexity results on machine scheduling. As shown below, we study the specially practical case where:

- jobs are identical, i.e. differences between calculating energy levels of different $\boldsymbol{k}$ values are neglected,
- job release dates are the same and equal to zero,
- jobs are scheduled on uniform parallel machines (i.e., on machines that do not run at the same speed).

This optimizing problem is solved in this paper with a greedy algorithm. The initial list, which contains the time required to complete one job for each machine, is first established and the times are sorted from earliest to latest, then a job is assigned to the machine that corresponds to the first completion time in the list. This completion time is then increased by the time to complete an additional job on the machine, and the list is stored. This process is repeated until all the jobs have been sent out. When the procedure is finished, we successfully schedule these n jobs on m machines for the earliest completion time. This scheme satisfies the minimality property described in [10] and the algorithm is listed below. A time complexity analysis of this algorithm for scheduling a set of n jobs on m machines leads to $\Theta(mn)$.

```
algorithm GreedySchedule
   const
     n = the quantity of jobs;
     m = the quantity of nodes;
     Time: real array; {An array of size m, in which Time[i] stands
     for the time required to complete one job on machine i}
   Var
     Scheme: integer array; {An array of size m}
     Buffer: real array; {An array of size m}
     i, j: integer;
   begin
   i := 0;
   j := 0;
   repeat
     i := i + 1;
     Scheme[i] := 0;
     Buffer[i] := Time[i];
   until i = m;
   i := 0;
   repeat
     i := i + 1;
     j := FindMin(Buffer); {FindMin(Buffer) finds the smallest element
     in Buffer, and returns its index}
     Buffer[j] := Buffer[j] + Time[j];
     Scheme[j] := Scheme[j] + 1;
   until i = n;
   {Scheme is the array we want, in which Scheme[i] stands for the
    number of jobs assigned to machine i}
end.
```

4 Calculating Energy Levels with TB Model

In this section, the calculation of energy levels with $sp^3d^5s^*$ TB model on a single node machine is explained in detail. The Hamiltonian matrix is constructed first

with TB model, and then the data structure is optimized to hold the Hamiltonian matrix which has billions of entries. At last, the eigenvalues of the Hamiltonian matrix, i.e. the energy levels, are extracted with the Implicitly Restarted Arnoldi Method.

4.1 Building Hamiltonian Matrix

In $sp^3d^5s^*$ TB model, the basis wave function of a crystal with a 3-D traditional symmetry is formed as a Bloch sum [6]:

$$\Phi_{\alpha l \boldsymbol{k}} = \frac{1}{\sqrt[2]{N}} \sum_j e^{i(\boldsymbol{R}_j + \boldsymbol{r}_l)\cdot \boldsymbol{k}} \phi_\alpha(\boldsymbol{r} - \boldsymbol{R}_j - \boldsymbol{r}_l), \tag{1}$$

where ϕ_α is the atomic orbital of the state indexed by α, $\boldsymbol{k}$ is the 3-D wave vector, $\boldsymbol{R}_j$ denotes the position of the jth primitive cell and $\boldsymbol{r}_l$ is the relative position of the lth atom within the primitive cell [11]. The eigenfunction is then expressed as a linear combination of $\Phi_{\alpha l \boldsymbol{k}}$:

$$\Psi_{\boldsymbol{k}} = \sum_{\alpha, l} C_{\alpha l} \Phi_{\alpha l \boldsymbol{k}}. \tag{2}$$

The Hamiltonian of the system is expanded upon $\Phi_{\alpha l \boldsymbol{k}}$ to form a matrix [6].

Due to the nearest neighbor approximation introduced by the TB method, the Hamiltonian matrices of the SiNWs are always sparse, which usually range from 50000×50000 to 500000×500000. In this work, such a Hamiltonian matrix is constructed with basic sub-blocks in a pattern according to atom arrangement of a specific nanowire. As a result, we record the matrix in sub-blocks rather than other traditional ways of sparse matrix storage. As an example, we consider a [112] oriented SiNW with a cross-section size of 7.5nm×7.5nm. The size of the Hamiltonian matrix is 38400×38400. The number of nonzero entries of this gigantic matrix is approximately 1.92×10^6. If the sparse storage is applied, the memory needed to hold all the nonzero elements is 30M bytes. However, if the representation of this matrix is stored in sub-blocks, only a constant memory of 70k bytes is required at each node.

4.2 Eigenvalue Computation

Eigenvalues of the Hamiltonian matrix are solved using the Implicitly Restarted Arnoldi Method working on the Krylov subspace, and its related software package ARPACK, which is a collection of Fortran77 subroutines designed for large scale eigenvalue problems [12], is employed in this work. During the Arnoldi procedure, a matrix-vector multiplication is needed for solving the eigenvalues. However, there is no limit on the data structure for the matrix. In this work, a subblock-vector multiplication scheme is developed. As shown in Fig. 2, sub-blocks and the corresponding parts of the vector are multiplied, and then the results are collected and assembled into a new vector. Level 2 Basic Linear Algebra Subprograms (BLAS) are included to achieve an optimum performance for the multiplication.

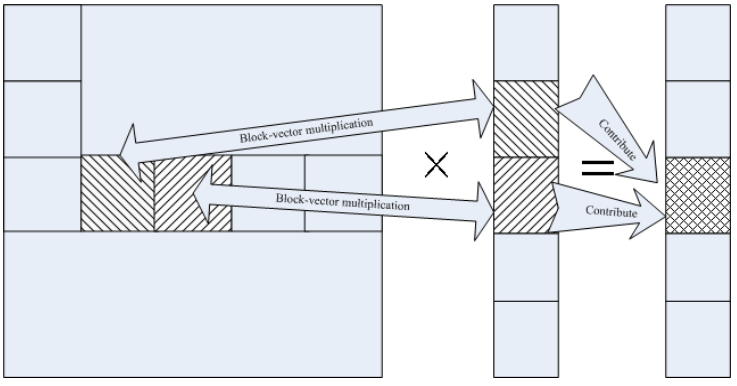

Fig. 2. The scheme of subblock-vector multiplication

As typically, only the energy levels near the bandgap (i.e., eigenvalues near the zero point) are cared in device simulation, the Hamiltonian matrix is not necessary to be diagonalized completely. However, the number of positive eigenvalues may not balance with the number of negative ones. This is not a satisfying solution as the information of both the conduction band and valance band is needed. To solve this problem, a matrix shift is carried out before matrix diagonalization. The calculated eigenvalues are then shifted back and stored.

5 Results

The cluster used for the bandstructure calculation consists of eight 1.0GHz Itanium-2 processors in four 2-CPU nodes connected through dual gigabit Ethernet. The MPI software used on this platform is MPICH-1.2.6 and the compilers are Intel Fortran and C/C++ compiler 9.0. The compilers switches used are: $-O2 - mcpu = itanium2 - mtune = itanium2$. Additionally, the Intel Math Kernel Library is included.

Bandstructures for three different SiNWs structures, including 56, 80, and 150 atoms in the supercell respectively, are computed. 24 points of $\boldsymbol{k}$ are chosen in the Brillouin zone, for each one of which 200 energy levels are selected. Finally, 100 energy levels on each $\boldsymbol{k}$ point are chosen and the bandstructures are charted in Fig. 3.

Fig. 4 shows the speedup and efficiency of the parallel calculation for different numbers of processors used. Here, the speedup is defined as the ratio of the run time using a single grid to the parallel run-time for calculating the bandstructures. From Fig. 4 we can see that a linear speedup is obtained and the overall performance of the parallel calculation is very good. Finally the machine time consumed for these three structures with different grid sizes is listed in Table 1.

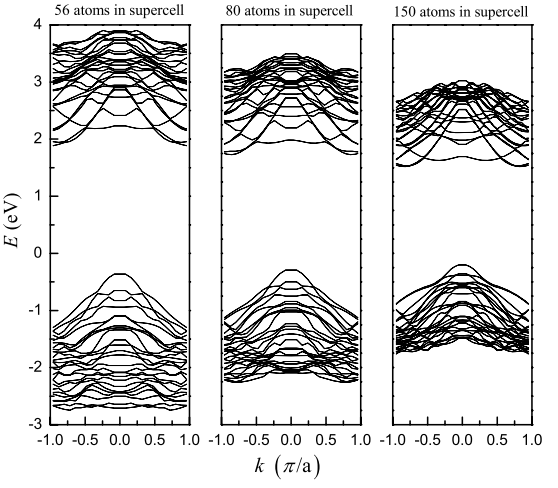

Fig. 3. Bandstructures of three [112] oriented SiNWs with different dimensions

Table 1. Machine time for bandstructure calculation

56 atoms		80 atoms		150 atoms	
#CPUs	Time (s)	#CPUs	Time (s)	#CPUs	Time (s)
1	1866	1	3384	1	7296
2	888	2	1573	2	3666
4	460	4	862	4	1819
8	299	8	421	8	906

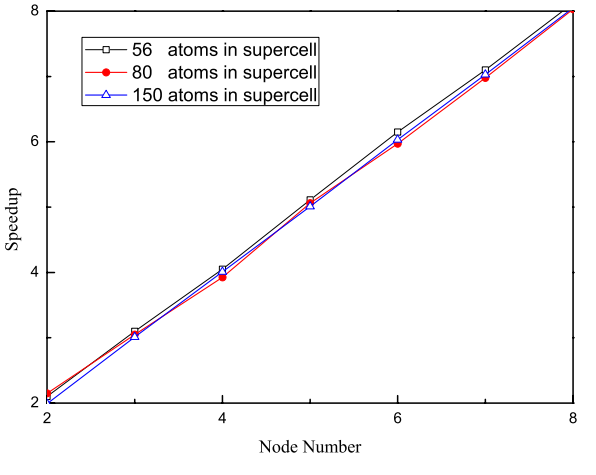

Fig. 4. Parallel computational speedup/efficiency

6 Conclusion

We have presented a parallel methodology of bandstructure calculation of SiNWs through domain decomposition in this paper. An algorithm has been developed for loading balance on heterogenous machines. Data structure has been optimized for storing the Hamiltonian matrix generated with $sp^3d^5s^*$ TB model, and the eigenvalues have been extracted with the Implicitly Restarted Arnoldi Method. Performance of the parallel methodology has been demonstrated on a cluster of identical parallel machines, and a linear speedup has been obtained.

Acknowledgement

This work is partially supported by a grant for Ph.D. education program from the Ministry of Education in China.

References

1. Cui, Y., Zhong, Z., Wang, D., Wang, W.U., Lieber, C.M.: High Performance Silicon Nanowire Field Effect Transistors. Nano Lett., Vol. 3 (2003) 149-152
2. Ma, D.D.D., Lee, C.S., Au, F.C.K., Tong, S.Y., Lee, S.T.: Small-Diameter Silicon Nanowire Surfaces. Science, (2003) 1874-1876
3. Zhao, X., Wei, C., Yang, L., Chou, M.Y.: Quantum Confinement and Electronic Properties of Silicon Nanowires. Phys. Rev. Lett., Vol. 92 (2004) 236805
4. Ko, Y.J., Shin, M., Lee, S., Park, K.W.: Effects of Atomistic Defects on Coherent Electron Transmission in Si Nanowires: Full Band Calculations. J. Appl. Phys., Vol. 89 (2001) 374-380
5. Jancu, J.-M., Scholz, R., Beltram, F., Bassani, F.: Empirical $spds^*$ Tight-binding Calculation for Cubic Semiconductors: General Method and Material Marameters. Phys. Rev. B., Vol. 57 (1998) 6493-6507
6. Guan, X., Yu, Z.: Supercell Approach in Tight-Binding Calculation of Si and Ge Nanowire Bandstructures. Chin. Phys. Lett., Vol. 22 (2005) 2651-2654
7. Snir, M., Otto, S.W., Lederman, S.H-, Walker, D.W., Dongarra, J.: MPI The complete Reference. The MIT Press Cambridge, Massachussets (1996)
8. Cormen, T.H., Leiserson, C.E., Rivest, R.L., Stein, C.: Introduction to Algorithms. 2nd edn. The MIT Press, Massachussets (2001)
9. Brucker, P., Knust, S.: Complexity Results of Scheduling Problems. URL: http://www.mathematik.uni-osnabrueck.de/research/OR/class
10. Dessouky, M.I., Lageweg, B.J., Lenstra, J.K., van de Velde, S.L.: Scheduling Identical Jobs on Uniform Parallel Machines. Statist. Neerlandica, Vol. 44 (1990) 115-123
11. Yu, P.Y., Cadona, M.: Fundamentals of Semicondutors: Physics and Materials Properties. 3rd edn. New York: Springer, New York (2001)
12. Lehoucq, R., Maschhof, K.,Sorensen, D., Yang, C.: ARPACK. URL: http://www.cs.ucdavis.edu/ bai/ET/arnoldi_methods/overview_ARPACK.html