Advanced Algorithms for *Ab-initio* Device Simulations

Mathieu Luisier*, Fabian Ducry*, Mohammad Hossein* Bani-Hashemian*, Sascha Brück*,
Mauro Calderara*, and Olaf Schenk†
*Integrated Systems Laboratory, ETH Zürich, 8092 Zürich, Switzerland
†Institute of Computational Science, University of Lugano, 6900 Lugano,Switzerland

***Abstract*—Numerical algorithms dedicated to large-scale quantum transport problems from first-principles are presented in this paper. They can be decomposed into three main categories: (i) the calculation of the open boundary conditions that connect the simulation domain and its environment, (ii) the solution of the resulting Schrödinger equation in the ballistic limit of transport, and (iii) the extension of this case to situations involving scattering, e.g. electron-phonon interactions. It will be shown that *ab-initio* device simulations require algorithms specifically developed for that purpose and that graphics processing units (GPUs) can bring significant speed ups as compared to solvers based on CPUs only. As an illustration, the computational times coming from the investigation of a realistic conductive bridging random access memory cell will be reported.**

***Index Terms*—quantum transport, algorithms, DFT, NEGF**

I. Introduction

Over the last 20 to 25 years, the capabilities of quantum transport calculations have witnessed a tremendous evolution. The state-of-the-art has rapidly gone from one-dimensional ballistic simulations in the effective mass approximation (EMA) [1] to multi-dimensional geometries [2], the inclusion of complex scattering mechanisms in an empirical full-band basis [3], and the consideration of *ab-initio* bandstructure models [4]. These progresses have been made possible partly thanks to hardware improvements, but mostly thanks to algorithmic innovations. The latter have allowed to move from proof-of-concept demonstrations to the investigation of realistic nano-devices that look almost like the manufactured ones.

This is especially true in the nanoelectronics research area. When the first quantum transport simulations of transistors were proposed at the beginning of the years 2000's, quantum mechanical effects were not playing a role as important as today: tunneling through the oxide layer and from the gate to the drain regions were causing non-negligible leakage currents, while the quantization of the channel states had a noticeable influence on carrier transport from source to drain. However, these phenomena could be accounted for by slightly adapting the classical drift-diffusion (DD) equations [5], the modeling standard in the semiconductor industry.

This work was supported by the Werner Siemens Stiftung, by ETH-Research Grant ETH-35 15-2, by the Platform for Advanced Scientific Computing in Switzerland (ANSWERS), by the European Research Council under Grant Agreement No 335684-E-MOBILE, and by two grants from the Swiss National Supercomputing Centre under Project No. s714 and pr28.

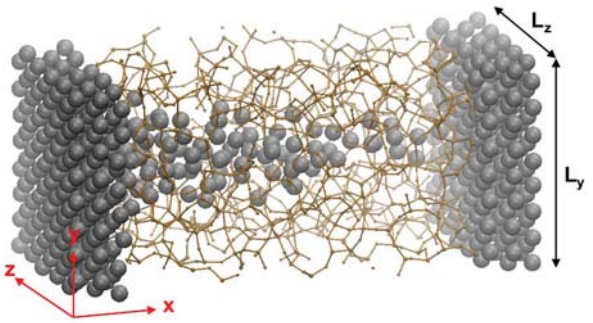

Fig. 1. Three-dimensional atomic structure of a conductive bridging random access memory (CBRAM) cell made of two Cu electrodes (one active and one inert) separated by an amorphous SiO_2 layer through which a nano-filament can grow and dissolve. Electron transport occurs along the x direction, while the y and z axes are assumed periodic. The considered cross section in the y-z plane is equal to 2.1×2.1 nm^2. The simulation domain (larger than the one shown here) is composed of 4449 atoms.

Even today, it could be argued that, in many cases, the drift-diffusion model is still sufficient to explain the characteristics of newly fabricated nano-transistors or to design a future generations of components, provided that the available DD-based simulation tools have been first properly calibrated. This by no way means that quantum transport solvers are unnecessary, simply that the performance of conventional metal-oxide-semiconductor logic switches can, to a certain extend, still be optimized with the help of a classical technology computer aided design (TCAD) software. The loss of accuracy induced by the usage of DD is largely compensated by the significantly lower computational burden as compared to a quantum transport package. With the advent of always more powerful computing units, things might rapidly change.

There are other classes of nano-devices where the utilization of a tool implementing quantum mechanical concepts is essential to capture the underlying physics. This is the case of non-volatile conductive bridging random access memories (CBRAMs), also called electro-chemical metallization (ECM) cells [6], whose electronic properties strongly depend on ionic motions, geometrical confinement, quantization, transport via hopping/tunneling, and electron-phonon interactions. All these effects cannot be properly described within the framework of a classical simulation approach. Accurate results require a quantum transport solver from first-principles.

978-1-5386-6790-3/18/$31.00 ©2018 IEEE

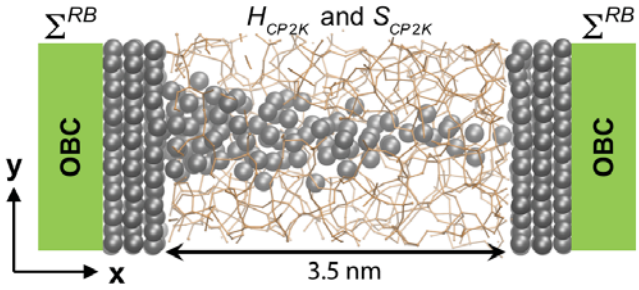

Fig. 2. Projection of the CBRAM cell in Fig. 1 onto the x-y plane. The simulation domain is described by a Hamiltonian H_{CP2K} and overlap S_{CP2K} matrices that are directly imported from the CP2K package [9]. The applied open boundary conditions (OBCs) connect the device structure to two semi-infinite leads via two retarded boundary self-energies labeled Σ^{RB}.

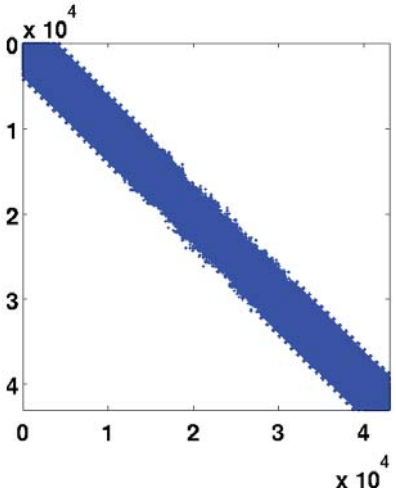

Fig. 3. Sparsity pattern of the Hamiltonian and overlap matrices corresponding to the CBRAM cell from Fig. 1 with extended Cu electrodes. The matrix size is equal to N=43020 with a filling of 3.57%. A sparse band with a maximum width of about 8000 can be observed along the diagonal of the matrix.

A Cu-SiO_2 CBRAM cell is schematically represented in Fig. 1. Its 3-D structure is resolved at the atomic level. It is composed of two metallic electrodes, one active and one inert (here Cu is used for both), separated by a SiO_2 oxide matrix through which a nano-filament can grow and dissolve. Here, we will show how the electrical current flowing through such a device can be simulated with quantum transport (QT), both in the ballistic limit and in the presence of scattering. To do that, open boundary conditions (OBCs) must be introduced. They drive the electron population out-of-equilibrium. The paper is organized as follows: in Section II, algorithms to compute the OBCs and to solve the resulting QT problem with Non-equilibrium Green's Functions (NEGF) are presented. Their application to the CBRAM cell from Fig. 1 is discussed in Section III. Finally, conclusions are drawn in Section IV.

II. Modeling Approach

Figure 2 depicts the simulation domain of a CBRAM cell in the low resistance state where the two metallic plates are short-circuited by a Cu nano-filament that grew between them. Within the NEGF formalism, the following equations must be solved to obtain the transport properties of this 3-D device:

$$\left(E \cdot S_{CP2K} - H_{CP2K} - \Sigma^{RB} - \Sigma^{RS}\right) \cdot G^R(E) = I, \quad (1)$$

$$G^{\gtrless}(E) = G^R(E) \cdot \left(\Sigma^{\gtrless B} + \Sigma^{\gtrless S}\right) \cdot G^A(E), \quad (2)$$

where G^R, G^A, $G^<$, and $G^>$ are the retarded, advanced, lesser, and greater Green's Functions at energy E, respectively, with the corresponding self-energies $\Sigma^{R,A,<,>}$. The index B (S) in Σ refers to the boundary (scattering) self-energy. Due to the metallic electrodes and the amorphous nature of the SiO_2 layer, parameterizing an empirical tight-binding model to simulate the considered CBRAM is not really practical. Performing a density-functional theory (DFT) [7] with a plane-wave code and then transforming the output into a set of maximally localized Wannier functions [8] does not appear more suitable due to the size of the system. The most convenient solution consists in using a DFT code relying on a localized basis set, e.g. the CP2K tool and its contracted Gaussian-type orbitals (GTO) [9]. Hence, the overlap S_{CP2K} and Hamiltonian H_{CP2K} matrices can be directly produced by CP2K and imported into a QT code. Their sparsity pattern can be seen in Fig. 3. Different techniques to compute the Σ^B's and solve Eqs. (1) and (2) are now reviewed.

Open Boundary Conditions: Iterative schemes like the Sancho-Rubio algorithm [10] are very popular to determine the OBC self-energies. They involve several matrix multiplications and inversions with the same size as Σ. Since several iterations (10 or more) must be executed to reach convergence, the application of such methods to large-scale *ab-initio* quantum transport problems is usually counter-indicated. As an alternative, the calculation of the OBCs can be transformed into a normal eigenvalue (EV) problem [11] whose solution time depends cubically on the system size. It should be noted that neither Sancho-Rubio nor the EV approach can be efficiently parallelized. More recently, contour integral methods have been developed. With them, only the most relevant eigenvalues needed to derive the OBC matrices can be computed [12], [13], as can be seen in Fig. 4. The key advantage of these techniques is that they can be parallelized. Furthermore, they mostly require the solution of linear systems of equations with a limited number of right-hand-sides. We have found that the so-called Beyn algorithm [13] works the best in our case.

Ballistic Transport: Once that the Σ^B's are known, the solution of Eqs. (1) and (2) can start. In fact, only G^R is needed for ballistic transport. To obtain it, it has been demonstrated that the mode space approximation can be successfully applied to DFT+NEGF situations [14]. However, this is only possible if the simulation domain is made of repeatable unit cells, a condition that breaks down in the presence of amorphous layers, as encountered here. As a consequence, the most common approach to deal with Eq. (1) remains the recursive Green's Function (RGF) algorithm where the (off-)diagonal blocks of the Green's Functions are constructed step-by-step

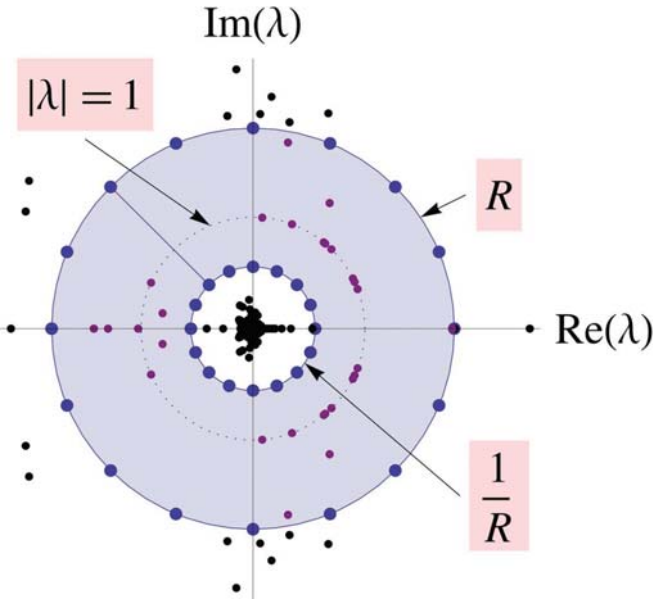

Fig. 4. Selected contour in the complex plane to enclose only the m eigenvalues λ corresponding to propagating and slow decaying modes of the contacts (magenta dots). The black dots refer to eigenvalues with $|\lambda|$ <1/R and $|\lambda|$ >R. They can be neglected as their contribution to the OBCs is minimal. Here, R is a cut-off radius with typical values in the range of 10 to 100 to ensure accurate results.

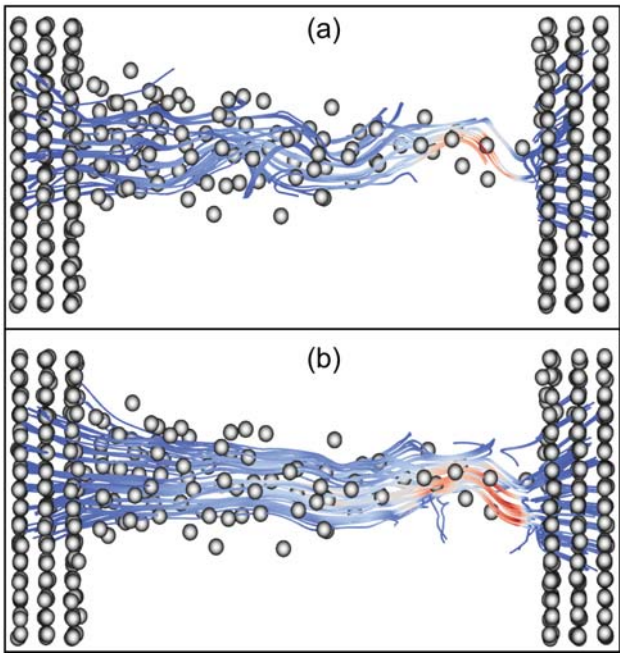

Fig. 5. Spatial current distribution through the CBRAM cell from Fig. 1 taken (a) in the ballistic limit of transport and (b) in the presence of electron-phonon scattering. Red indicates in both cases high current concentrations, blue lower ones. For simplicity, the Si and O atoms are not shown.

going from one side of the Hamiltonian matrix to the other. Again, due to the necessity to perform a high number of matrix operations, RGF tends to be a limiting factor in *ab-initio* QT problems. The computational load can be reduced if NEGF is replaced by the quantum transmitting boundary method (QTBM), also known as Wave Function (WF) formalism, whose governing equation can be written as [11]

$$\left(E \cdot S_{CP2K} - H_{CP2K} - \Sigma^{RB}\right) \cdot \Psi(E) = Inj. \qquad (3)$$

This is a sparse linear system of equations "Ax=b" where the right-hand-side Inj accounts for all modes injected into the simulation domain. The wave function $\Psi(E)$ instead of the retarded Green's Function $G^R(E)$ is computed. Eq. (3) can be ideally handled by a parallel sparse linear solver such as MUMPS [16]. We have recently established that by using general-purpose graphics processing units (GPUs), implementing our own algorithm called SplitSolve, and interleaving the calculation of the OBC and of Eq. (3), the computational time can be greatly decreased in the context of device simulations from first-principles [17].

Transport with Scattering: When scattering is included through self-energies Σ^S, Eq. (3) is no more adapted and Eqs. (1-2) must be recalled, the retarded and greater/lesser Green's Functions being requested. Besides the RGF algorithm that can be run on CPUs only or accelerated by GPUs, a selected inversion method can be utilized [18]. Based on a LU decomposition of the Hamiltonian matrix, selected entries of G^R and $G^{\gtrless}$ can be computed, i.e. those corresponding to the sparsity pattern of H_{CP2K}+Σ^{RB}+Σ^{RS}. This approach

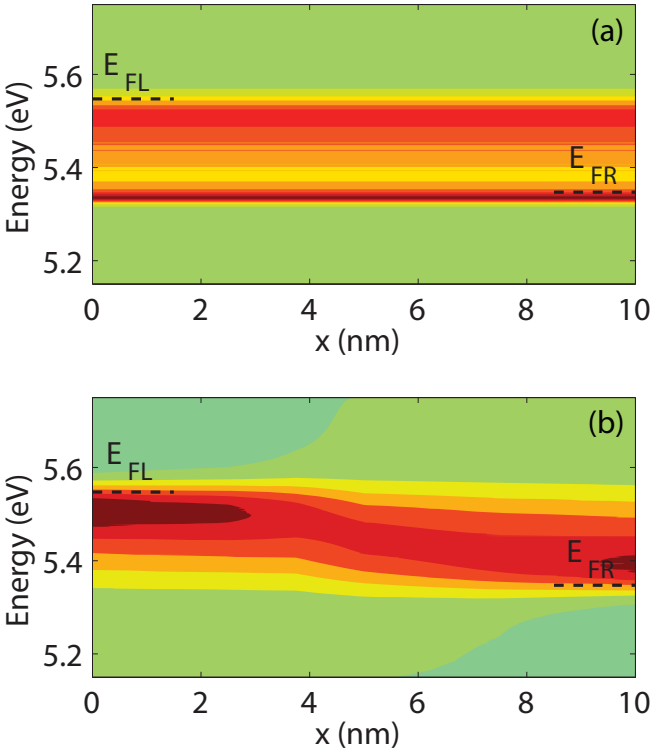

Fig. 6. Spectral current through the same device as in Fig. 5, again (a) in the ballistic limit and (b) with dissipative scattering. Regions with high current densities are plotted in red, those without any current in green. The blue line refers to the electrostatic potential. The left (E_{FL}) and right (E_{FR}) Fermi levels are indicated by the dashed lines.

Method	1 core	2 cores	4 cores	8 cores	12 cores
EV	2467	1235	-	-	-
Beyn	117	58.6	34	20.3	18.9

TABLE I

TIME (IN SECONDS) TO COMPUTE THE OBCS FOR THE CBRAM CELL FROM FIG. 1 WITH AN EIGENVALUE (EV) METHOD [11] AND WITH THE BEYN ALGORITHM [13], [17]. THE RESULTS ARE REPORTED AS A FUNCTION OF THE NUMBER OF CORES. ALL OF THEM ARE LOCATED ON THE SAME HYBRID NODE COMPOSED OF AN INTEL XEON E5-2690 V3 CPU (2.60 GHZ, 12 CORES) AND A NVIDIA TESLA P100 GPU.

Method	1 core	2 cores	4 cores	8 cores	12 cores
RGF (NEGF)	1743	874	-	-	-
MUMPS (WF)	266	150.4	116.2	100.1	-
SS (WF)*	-	6.96	-	-	-

TABLE II

TIME (IN SECONDS) TO SOLVE THE BALLISTIC NEGF OR WF EQUATION FOR ONE ENERGY POINT OF THE CBRAM CELL FROM FIG. 1 WITH THE RGF ALGORITHM [15], MUMPS SPARSE LINEAR SOLVER [16], AND SPLITSOLVE (SS) APPROACH [17]. THE SAME HARDWARE AS IN TABLE I IS USED. THE * INDICATES THE USAGE OF THE AVAILABLE GPU.

takes advantage of the PARDISO library (direct sparse linear solver) and its shared memory parallelization [19].

III. RESULTS

Before diving into numerical considerations, simulation results are presented in Figs. 5 and 6: the spatial and spectral distributions of the electrical currents flowing through the CBRAM cell from Fig. 1 are plotted in the ballistic limit of transport and in the presence of electron-phonon scattering [20]. A smoothening of the electron trajectories as well as energy relaxation characterize the inclusion of dissipative interactions. These data could not have been generated without the implementation of novel parallel algorithms stressing not only CPUs, but also GPUs. The timing experiments below should clearly highlight their benefit. They have all been executed on either one single (OBCs and ballistic limit) or two (scattering) nodes of the Piz Daint supercomputer at the Swiss National Supercomputing Center [21]. Each of these nodes is made of one Intel Xeon E5-2690 v3 CPU (2.60 GHz, 12 cores) and one NVIDIA Tesla P100 GPU. Only the time to solve either Eqs. (1-2) or Eq. (3) for one energy point is given. Due to memory constraints, it is not possible to simultaneously treat more than one energy point per node, except for the scattering case where two nodes per energy point are required.

Table I reports the time to compute the OBCs of the CBRAM cell from Fig. 1 with the EV and Beyn methods. Sancho-Rubio has not been tested as it would have taken even more time than the EV approach. Using a single core, Beyn is already more than 20$\times$ faster than EV. If all available cores per node are leveraged, Beyn becomes 65$\times$ more efficient than EV (18.9 sec. on 12 cores vs. 1235 sec. on 2 cores), with almost exactly the same numerical results. In approximately 0.1% of the cases, the Beyn algorithm does not return the correct contact eigenvalues. This is not an intrinsic issue: it is rather caused by the choice of not suitable setting parameters, e.g. the cut-off radius R in Fig. 4.

Method	1 core	2 cores	4 cores	8 cores	12 cores
RGF CPU	2001	1214	-	-	-
RGF GPU*	81.3	54.5	-	-	-
SINV	-	3045	1777	1147	992

TABLE III

TIME (IN SECONDS) TO SOLVE THE NEGF EQUATIONS IN PRESENCE OF SCATTERING FOR ONE ENERGY POINT OF THE CBRAM CELL FROM FIG. 1 WITH THE RGF ALGORITHM [15] (CPU AND CPU+GPU VERSION) AND THE SINV APPROACH FROM PARDISO [18]. TWO HYBRID NODES WITH THE SAME PROPERTIES AS IN TABLES II AND III ARE EMPLOYED.

Type	Ball. CPU	Ball. GPU	Scatt. CPU	Scatt. GPU
Best Time (s)	120.4	21	1026	88.5
Speed Up	1$\times$	**5.7**$\times$	1$\times$	**11.6**$\times$
OBC Solver	Beyn	Beyn	Beyn	Beyn
Schröd. Solver	MUMPS	SS	SINV	RGF
#Cores	8	10	2$\times$12	2$\times$2
Parallelization	MPI	MPI+GPU	MPI+OMP	MPI+GPU

TABLE IV

SUMMARY OF THE SHORTEST TIMES (IN SECONDS) OBTAINED TO SOLVE THE OBCS AND RESULTING SCHRÖDINGER EQUATION FOR ONE ENERGY POINT OF THE SAME CBRAM CELL AS IN FIG. 1. THE BALLISTIC AND SCATTERING CASES ARE CONSIDERED. THE SPEED UP BROUGHT BY GPUS, THE USED SOLVERS, THE CORE CONFIGURATION, AND THE PARALLELIZATION TYPE ARE ALSO REPORTED.

In the ballistic limit of transport, the solution of Eq. (1) with the RGF algorithm is equal to 874 sec. on two cores, as detailed in Table II. Going to a higher number of cores with RGF calls for massive code modifications without real advantage in terms of efficiency for short device structures as the considered CBRAM cell. Replacing the NEGF in Eq. (1) by the WF in Eq. (3) leads to a substantial speed up, the computing time decreasing from 874 sec. with RGF on 2 cores down to 100.1 sec with MUMPS on 8 cores. In the latter solver, the parallelization is achieved via the Message Passing Interface (MPI). A factor of 8 is gained in the process. More impressive is the reduction of the simulation time enabled by SplitSolve: with 2 cores supported by the available GPU, Eq. (3) is solved in less than 7 sec, 14 (100) times faster than with MUMPS (RGF).

As soon as scattering is turned on (Table III), the calculation of the lesser/greater Green's Functions cannot be avoided anymore. Two versions of the RGF algorithm have been implemented for that purpose, one running only on CPUs and one off-loading the computationally most intensive numerical operations to the GPUs. This relatively straightforward trick speeds up the simulations by a factor larger than 20, which is really attractive for practical applications. The selected inversion technique (SINV), currently restricted to CPUs only, is at first slower than its RGF counterpart, but as the number of cores (threads) per node increases, it starts to slightly outperform it (1214 sec. on 2 cores for RGF on CPUs vs. 992 sec. on 12 cores for SINV, speed up of 1.2).

The best results on CPUs and CPUs+GPUs are summarized in Table IV for the ballistic and scattering cases. It should be emphasized that one of the key features of the SplitSolve algorithm is that it can start working on the solution of Eq. (3) even before the boundary self-energies are known. Hence, the

CPUs and GPUs are stressed out at the same time, which optimizes the computational performance and the simulation time. If we assume that the presented CPU-based algorithms represent the state-of-the-art, it can be deduced from Table IV that GPUs can further reduce the computing time by one order of magnitude. This factor has paved the way for the results in Figs. 5 and 6. Without it, the inclusion of electron-phonon scattering and self-heating effects in the study of realistic CBRAM cells would not be doable at this time [20].

IV. Conclusion

A series of parallel numerical algorithms has been reviewed in this paper to shed light on the quantum transport properties of nanoscale devices with large dimensions. Significant speed up factors are obtained as compared to standard solution schemes, both for the calculation of the open boundary conditions as well as for the solution of the resulting NEGF or WF equations. In the ballistic limit of transport, the proposed algorithms operate close to the theoretical peak performance of the machine they are running on so that further important improvements will be rather challenging. The situation is very different for situations requiring the presence of scattering self-energies. In this case, a parallel, fully GPU-based RGF algorithm can be envisioned. It would probably bring an additional speed up of 2 or more with respect to a simulator or a code running only on CPUs.

References

[1] S. Datta, "Electronic Transport in Mesoscopic Systems", Cambridge University Press, Cambridge (1995).

[2] J. Wang, E. Polizzi, and M. Lundstrom, J. Appl. Phys. 96, 2192 (2004).

[3] R. Lake, G. Klimeck, R. C. Bowen, and D. Jovanovic, J. Appl. Phys. 81, 7845 (1997).

[4] M. Brandbyge, J.-L. Mozos, P. Ordejon, J. Taylor, and K. Stokbro, Phys. Rev. B 65, 165401 (2002).

[5] S. Selberherr, A. Schutz, and H. W. Potzl, IEEE Trans. Elec. Dev. 27, 1540 (1980).

[6] R. Waser and M. Aono, Nature Materials 6, 833 (2007).

[7] W. Kohn and L. J. Sham, Phys. Rev. 140, A1133 (1965).

[8] N. Marzari and D. Vanderbilt, Phys. Rev. B 56, 12847 (1997).

[9] J. VandeVondele, M. Krack, F. Mohamed, M. Parrinello, T. Chassaing, and J. Hutter, Comput. Phys. Commun. 167, 103 (2005).

[10] M. P. L. Sancho, J. M. L. Sancho, J. M. L. Sancho, and J. Rubio, J. Phys. F: Met. Phys. 15, 851 (1985).

[11] M. Luisier, A. Schenk, W. Fichtner, and G. Klimeck, Phys. Rev. B 74, 205323 (2006).

[12] E. Polizzi, Phys. Rev. B 79, 115112 (2009).

[13] W.-J. Beyn, Linear Algebra Appl. 436, 3839 (2012).

[14] M. Shin, W. J. Jeong, and J. Lee, J. Appl. Phys. 119, 154505 (2016).

[15] A. Svizhenko, M. P. Anantram, T. R. Govindan, B. Biegel, and R. Venugopal, J. Appl. Phys. 91, 2343 (2002).

[16] P. Amestoy, I. Duff, and J.-Y. L'Excellent, Comput. Methods Appl. Mech. Eng. 184, 501 (2000).

[17] S. Brück, M. Calderara, M. H. Bani-Hashemian1, J. VandeVondele, and M. Luisier, J. Chem. Phys. 147, 074116 (2017).

[18] A. Kuzmin, M. Luisier, and O. Schenk, in Euro-Par 2013 Parallel Processing (vol. 8097 of Lecture Notes in Computer Science), F. Wolf, B. Mohr, and D. Mey, Eds. Berlin, Germany: Springer, pp. 533544 (2013).

[19] O. Schenk and K. Gärtner, J. Future Generation Computer Systems 20, 475 (2004).

[20] F. Ducry, A. Emboras, S. Andermatt, M. H. Bani-Hashemian, N. Cheng, J. Leuthold, and M. Luisier, Proceedings of the IEDM 2017, pp. 4.2.1-4.2.4 (2017).

[21] http://www.cscs.ch