Monte Carlo Simulation of Ultrafast Carrier Transport: Scalability Study

Aneta Karaivanova\textsuperscript{a}, Emanuil Atanassov\textsuperscript{a}, Todor Gurov\textsuperscript{a}

\textsuperscript{a}IICT-BAS, Acad. G. Bonchev St., Bl. 25A, Sofia 1113, Bulgaria

Abstract

In this work we consider Monte Carlo methods and algorithms for solving quantum-kinetic integral equations which describe the electron transport in semiconductors. Here we study the scalability of the presented algorithms using HPC resources in South-Eastern Europe. Numerical results for parallel efficiency and computational cost are also presented. In addition we discuss the coordinated use of heterogeneous HPC resources from one and the same application in order to achieve a good performance.

Keywords: electron transport, Monte Carlo algorithms, scalability, parallel efficiency, high-performance computations

1. Introduction

The Monte Carlo Methods (MCMs) provide approximate solutions to a variety of mathematical problems by performing statistical sampling experiments on a computer [1, 2]. They are based on the simulation of random variables whose mathematical expectations are equal to a given functional of the solution of the problem under consideration. By sampling sufficient number of realizations of the chosen random variable, one can obtain both, an estimate of the desired quantity (solution), and an estimate of the error. This allows to define a confidence interval for the solution with certain probability. In this way, MCM can be used for solving problems with uncertainties. Improving the MCM means to decrease the size of the confidence interval also taking into account the computational time.

Many problems in a transport theory and related areas can be described mathematically by a second kind integral equation:

$$f = K(f) + \phi.$$  

(1)

In general, the physical quantities of interest are determined by functionals of the type:

$$J_g(f) \equiv (g, f) = \int_G g(x)f(x)dx,$$  

(2)

where the domain $G \subset \mathbb{R}^d$ and $\mathbb{R}^d$ is the $d$-dimensional Euclidean space. The functions $f(x)$ and $g(x)$ belong to any Banach space $X$ and to the adjoint space $X^*$, respectively, and $f(x)$ is the solution of (1).
The mathematical concept of the MC approach is based on the iterative expansion of the solution of (1):

$$f_s = K(f_{s-1}) + \phi, \ s = 1, 2, \ldots ,$$  \hspace{1cm} (3)

where $s$ is the number of iterations. In fact (3) defines a Neumann series

$$f_s = \phi + K(\phi) + \ldots + K^{s-1}(\phi) + K^s(\phi_0), \ s > 1,$$

where $K^s$ means the $s$-th iteration of $K$. If the corresponding infinite series converges then the sum is an element $f$ from the space $X$ which satisfies (1).

The replacement of $f$ by the Neumann series in (2), gives rise to a sum of consecutive terms which are evaluated by a MC method with the help of random estimators.

We define a random variable $\xi$ such that its mathematical expectation is equal to $J(f)$: $E\xi = J(f)$.

Then we can define a MC method

$$\bar{\xi} = \frac{1}{N} \sum_{i=1}^{N} \xi^{(i)} \xrightarrow{P} J_g(f),$$  \hspace{1cm} (4)

where $\xi^{(1)}, \ldots, \xi^{(N)}$ are independent values of $\xi$ and $\xrightarrow{P}$ means stochastic convergence as $N \rightarrow \infty$. The rate of convergence is evaluated by the “law of the three sigmas”, [1, 3]:

$$P\left( |\bar{\xi} - J_g(f)| < 3 \frac{\sqrt{\text{Var}(\xi)}}{\sqrt{N}} \right) \approx 0.997.$$

Here $\text{Var}(\xi) = E\xi^2 - E^2\xi$ is the variance of the MC estimator. Thus, a peculiarity of any MC estimator is that the result is obtained with a statistical error [1, 3, 4]. As $N$ increases, the statistical error decreases proportionally to $N^{-1/2}$.

Thus, there are two types of errors – systematic (a truncation error) and stochastic (a probability error) [4, 5]. The systematic error depends on the number of iterations of the used iterative method, while the stochastic error is related to the the probabilistic nature of the MC method. From (1) and (3) one can get the value of the truncation error. If $f_0 = \phi$ then

$$f_s - f = K^s(\phi - f).$$

The relation (4) still does not determine the computational MC algorithm: we must specify the modeling function (called sampling rule) for the random variable $\xi$.

$$\Theta = F(\beta_1, \beta_2, \ldots ,),$$  \hspace{1cm} (5)

where $\beta_1, \beta_2, \ldots ,$ are uniformly distributed random numbers in the interval (0, 1). It is known that random number generators are used to produce such sequences of numbers. They are based on specific mathematical algorithms. Now both relations (4) and (5) define a MC algorithm for estimating $J_g(f)$. The case when $g = \delta(x - x_0)$ is of special interest, because it is used for calculating the value of $f$ at $x_0$, where $x_0 \in G$ is a fixed point.

Every iterative algorithm uses a finite number of iterations $s$. In practice we define a MC estimator $\xi_s$ for computing the functional $J_g(f_s)$ with a statistical error. On the other hand $\xi_s$ is a biased estimator for the functional $J_g(f)$ with stochastic and truncation errors [4, 5]. The number of iterations can be a random variable when an $\varepsilon$-criterion is used to truncate the Neumann series or the corresponding Markov chain in the MC algorithm.

The presented numerical results are obtained at the Bulgarian HPC infrastructure which consists of several small HPC clusters with Intel CPUs and Infiniband interconnection (total more than 1400 logical cores) grouped around the powerful supercomputer - BlueGene/P with 8192 CPU cores.

The paper is organized as follows. In Section 2 the quantum-kinetic equation is derived from a physical model describing electron transport in quantum wires. An integral form of the equation is obtained by reducing the dimensionality of space and momentum coordinates. The MC approach and corresponding MC algorithm are presented in Section 3. The numerical results using Bulgarian HPC resources are discussed in Section 4. Summary and directions for future work are given in Section 5.
2. The Quantum kinetic integral equation

In the general case a Wigner equation for nanometer and femtosecond transport regime is derived from a three equations set model based on the generalized Wigner function [7]. The complete Wigner equation poses serious numerical challenges. Two limiting versions of the equation corresponding to simplified physical conditions are considered in few works, namely, the Wigner-Boltzmann equation [8] and the homogeneous Levinson (or Barker-Ferry) equation [9, 10]. These equations are analyzed with various MCMs using spherical and cylindrical transformations to reduce the dimensions in the momentum space [11, 12]. The computer power of the European Grid infrastructure (EGI) in some cases is used to investigate above problems [13, 14, 15].

Here we consider a highly non-equilibrium electron distribution which propagates in a quantum semiconductor wire [16]. The electrons, which can be initially injected or optically generated in the wire, begin to interact with three-dimensional phonons. This is third limiting case, where the electron-phonon interaction is described on the quantum-kinetic level by the Levinson equation [17, 18], but the evolution problem becomes inhomogeneous due to the spatial dependence of the initial condition. The direction of the wire is chosen to be $z$, the corresponding component of the wave vector is $k_z$. The electrons are in the ground state $\Psi(r_\perp)$ in the plane normal to the wire, which is an assumption consistent at low temperatures. The initial carrier distribution is assumed Gaussian both in energy and space coordinates, and an electric field can be applied along the wire.

The integral representation of the quantum kinetic equation for the electron Wigner function $f_w$ in this case has the form [19]:

$$f_w(z, k_z, t) = f_w(z - \frac{\hbar k_z}{m} t + \frac{h F}{2m} t^2, k_z, 0) +$$

$$\int_0^t dt' \int_0^t dt' \int dq'_z \int dk'_z \left[ S(k'_z, k_z, t', t', q'_z) f_w \left(z - \frac{\hbar k_z}{m} (t - t') + \frac{h F}{2m} (t^2 - t'^2) + \frac{\hbar q'_z}{2m} (t' - t''), k_z, t' \right) -
S(k_z, k'_z, t', t'' q'_z) f_w \left(z - \frac{\hbar k_z}{m} (t - t'') + \frac{h F}{2m} (t^2 - t''^2) - \frac{\hbar q'_z}{2m} (t' - t''), k_z, t'' \right) \right] \right].$$

where

$$S(k'_z, k_z, t', t'', q'_z) =$$

$$\frac{2V}{(2\pi)^3} \left| G(q'_z) \mathcal{F}(q'_z, k_z - k'_z) \right|^2 \left[ (n(q') + 1) \cos \left( \frac{e(k_z) - e(k'_z) + \hbar \omega q'}{\hbar} (t' - t'') + \frac{h}{2m} F q'_z (t'^2 - t''^2) \right) +
\right.$$  

$$\left. n(q') \cos \left( \frac{e(k_z) - e(k'_z) - \hbar \omega q'}{\hbar} (t' - t'') + \frac{h}{2m} F q'_z (t'^2 - t''^2) \right) \right].$$

Here, $f(z, k_z, t)$ is the Wigner function described in the 2D phase space of the carrier wave vector $k_z$ and the position $z$, and $t$ is the evolution time.

The electric force $F$ depends on the electric field $E$ as follows: $F = eE / \hbar$, where the electric field is along the direction of the wire, $e$ being the electron charge and $\hbar$ - the Plank’s constant.

$n_q = 1/(\exp(h\omega_q / kT) - 1)$ is the Bose function, where $k$ is the Boltzmann constant and $T$ is the temperature of the crystal, corresponds to an equilibrium distributed phonon bath.

$h\omega_q$ is the phonon energy which generally depends on $q' = q'_z + q'_z = q'_z + (k_z - k'_z)$, and $e(k_z) = (\hbar^2 k_z^2)/(2m)$ is the electron energy.

$\mathcal{F}$ is obtained from the Fröhlich electron-phonon coupling by recalling the factor $i\hbar$ in the interaction Hamiltonian:

$$\mathcal{F}(q'_z, k_z - k'_z) = - \left[ \frac{2\pi e^2 \omega q'}{h V} \left( \frac{1}{\epsilon_\infty} - \frac{1}{\epsilon_s} \right) \right]^{1/2},$$

where $(\epsilon_\infty)$ and $(\epsilon_s)$ are the optical and static dielectric constants. The shape of the wire affects the electron-phonon coupling through the factor

$$G(q'_z) = \int d\mathbf{r}_\perp e^{i q'_z \cdot \mathbf{r}_\perp} |\Psi(r_\perp)|^2,$$
where \( \Psi \) is the ground state of the electron system in the plane normal to the wire. If the cross-section of the wire is chosen to be a square with side \( a \) than we obtain:

\[
|G(q'_\perp)|^2 = |G(q'_\perp)G(q'_z)|^2 = \left(\frac{4\pi}{q'_z a (q'_z a^2 - 4\pi^2)}\right)^2 4 \sin^2(aq'_z/2) \times \left(\frac{4\pi}{q'_z a (q'_z a^2 - 4\pi^2)}\right)^2 4 \sin^2(aq'_z/2).
\]

3. Monte Carlo approach

The equation (6) can be rewritten in the form:

\[
f_w(z, k_z, t) = f_w(z - z(k_z, t), k_z, 0) + \int_0^t dt' \int_{t'}^t dt'' \int_G d^3k' K_1(k_z, k_z', t', t'') \times f_w \left( z + h(k_z, q'_z, t', t'', F), k_z', t'' \right) + \int_0^t dt' \int_{t'}^t dt'' \int_G d^3k' K_2(k_z, k_z', t', t'') \times f_w \left( z + h(k_z, -q'_z, t', t'', F), k_z, t'' \right),
\]

where

\[
z(k_z, t) = \frac{h k_z}{m} t - \frac{h F}{2m} t^2,
\]

\[
h(k_z, q'_z, t, t', t'', F) = -\frac{h k_z}{m} (t - t'') + \frac{h F}{2m} (t^2 - t''^2) + \frac{h q'_z}{2m} (t - t'),
\]

\[
K_1(k_z, k_z', t', t'') = S(k_z', k_z, t', t'', q'_\perp) = -K_2(k_z', k_z, t', t''),
\]

and

\[
\int_G d^3k' = \int d^2q'_\perp \int_{-Q_z}^{Q_z} dk_z.
\]

The values of the physical quantities are expressed by the following general functional of the solution of (7):

\[
J_g(f) = \int_0^T \int_D g(z, k_z, t) f_w(z, k_z, t) dz dk_z dt.
\]

Here we specify that the phase space point \((z, k_z)\) belongs to a rectangular domain \(D = (-Q_z, Q_z) \times (-Q_z, Q_z)\), and \(t \in (0, T)\).

The function \(g(z, k_z, t)\) depends on the quantity of interest. Here, we are going to estimate by MC approach the Wigner function (6), the wave vector (and respectively the energy) \(f(k_z, t)\), and the density distribution \(n(z, t)\). The last two functions are given by the integrals

\[
f(k_z, t) = \int \frac{dz}{2\pi} f_w(z, k_z, t) \quad \text{and} \quad n(z, t) = \int \frac{dk_z}{2\pi} f_w(z, k_z, t).
\]

The MC estimator for evaluating the functional (8) using backward time evolution of the numerical trajectories can be constructed in the following way:

\[
\hat{\xi}_n[J_g(f)] = \frac{g(z, k_z, t)}{p_m(z, k_z, t)} W_0 f_w(., k_z, 0) + \frac{g(z, k_z, t)}{p_m(z, k_z, t)} \sum_{j=1}^s W_{ij} f_w(., k^{\alpha}_{z,j}, t_j).
\]

Here

\[
f_w(., k^{\alpha}_{z,j}, t_j) = \begin{cases} f_w \left( z + h(k_{z,j-1}, k_{z,j-1} - k_{z,j}, t_{j-1}, t_j, t, F), k_{z,j}, t_j \right), \\ f_w \left( z + h(k_{z,j-1}, k_{z,j} - k_{z,j-1}, t_{j-1}, t_j, t, F), k_{z,j}, t_j \right) \end{cases}
\]

where \(\alpha = 1\), in the first case, and \(\alpha = 2\) in the second one;
The probabilities \( p_\alpha(z, k, t) \) and the transition density \( p_{tr}(k, k', t', t'') \) are chosen to be tolerant\(^1\) to the function \( g(z, k, t) \) and the kernels, respectively. The initial point \((z, k, t_0)\) in the Markov chain is chosen using the initial density, where \( k_0 \) is the third coordinate of the wave vector \( \mathbf{k}_0 \). Next points \((k_j, t_j, t_j) \in (-Q_2, Q_2) \times (t_j, t_{j-1}) \times (0, t_{j-1})\) of the Markov chain:

\[
(k_{0}, t_0) \rightarrow (k_1', t_1, t_1) \rightarrow \ldots \rightarrow (k_{j}', t_{j}', t_{j}) \rightarrow \ldots ,
\]

where \( j = 1, 2, \ldots, s \) do not depend on the position \( z \) of the electrons. They are sampled using the transition density \( p_{tr}(k, k', t', t'') \) as we take only the \( z \)-coordinate of the wave vector \( k \). Note the time \( t_j' \) conditionally depends on the selected time \( t_j \). The Markov chain terminates in time \( t_s < \varepsilon_1 \), where \( \varepsilon_1 \) is a fixed small positive number called a truncation parameter.

In order to evaluate the functional (8) by \( N \) independent samples of the estimator (9), we define a Monte Carlo method

\[
\frac{1}{N} \sum_{i=1}^{N} (\xi_i)_{i} \xrightarrow{P} J_g(f_s) \approx J_g(f),
\]

where \( \xrightarrow{P} \) means stochastic convergence as \( N \to \infty \); \( f_s \) is the iterative solution obtained by the Neumann series of (7), and \( s \) is the number of iterations.

The relation (10) still does not determine the computational algorithm. To define a MC algorithm we have to specify the initial and transition densities, as well the modeling function (or sampling rule). The modeling function describes the rule needed to calculate the states of the Markov chain by using uniformly distributed random numbers in the interval \((0, 1)\). In our case we use SPRNG library [6].

Here, the transition density is chosen:

\[
p_{tr}(k, k', t', t'') = p(k'/k)p(t, t', t''),
\]

where

\[
p(t, t', t'') = p(t, t'')p(t' / t'') = \frac{1}{t(t - t'')}
\]

and

\[
p(k'/k) = c_1/(k' - k)^2
\]

\((c_1 \text{ is the normalized constant). Thus, if we know } t, \text{ the next times } t'' \text{ and } t' \text{ are computed by using the inverse-transformation rule.}

The wave vectors \( k' \) are sampled in the following algorithm:

1. Sample a random unit vector \( \omega = (\sin \theta \cos \varphi, \sin \theta \sin \varphi, \cos \theta) \) as \( \sin \theta = 2 \sqrt{\beta_1 - \beta_1^2}, \cos \theta = 2 \beta_1 - 1, \) and \( \varphi = 2 \pi \beta_2 \) where \( \beta_1 \) and \( \beta_2 \) are uniformly distributed numbers in \((0, 1)\);

2. Calculate \( l(\omega) = -\omega \cdot k + (Q_2^2 + (\omega \cdot k)^2 - k_0^2)^{1/2} \), where \( \omega \cdot k \) denotes a scalar product between two vectors;

3. Sample \( \rho = l(\omega) \beta_3 \), where \( \beta_3 \) is an uniformly distributed number in \((0, 1)\);

4. Calculate \( k' = k + \rho \omega \).

We note that we have to compute all three coordinates of the wave vector although we need only the third one. The choice of \( p_{in}(z, k, t) \) depends on the function \( g(z, k, t) \). The cases when

\[
(i) \quad g(z, k, t) = \delta(z - z_0) \delta(k_z - k_{z,0}) \delta(t - t_0),
\]

\[
(ii) \quad g(z, k, t) = \frac{1}{2\pi} \delta(k_z - k_{z,0}) \delta(t - t_0),
\]

\[
(iii) \quad g(z, k, t) = \frac{1}{2\pi} \delta(z - z_0) \delta(t - t_0),
\]

\(^1\)\( r(x) \) is tolerant of \( g(x) \) if \( r(x) > 0 \) when \( g(x) \neq 0 \) and \( r(x) \geq 0 \) when \( g(x) = 0 \).
are of special interest, because they estimate the values of the Wigner function, wave vector and density distribution in fixed points.

4. Parallel implementation and numerical results

The stochastic error for the (homogeneous) Levinson or Barker-Ferry models has order $O(\exp(c_2t)N^{-1/2})$, where $t$ is the evolution time and $c_2$ is a constant depending on the kernels of the obtained quantum kinetic equation [11, 12]. Using the same mathematical techniques as in [11], we can prove that the stochastic error of the MC estimator under consideration has order $O(\exp(c_3t^2/N^{1/2})$. The factor $\exp(c_3t^2)$ contains the term $t^2$ because there is a double integration over the evolution time in the quantum kinetic equation (6). The estimate shows that when $t$ is fixed and $N \to \infty$ the error decreases, but for large $t$ the factor $\exp(c_3t^2)$ looks ominous. Therefore, the MC algorithm described above solves an NP-hard problem concerning the evolution time. The suggested importance sampling technique, which overcomes the singularity in the kernels, is not enough to solve the problem for long evolution time with small stochastic error. In order to decrease the stochastic error we have to increase $N$ - the number of Markov chain realizations. For this aim, a lot of CPU power is needed for achieving acceptable accuracy at evolution times above 100 femtoseconds.

It is known that the MC algorithms are perceived as computationally intensive and naturally parallel [20]. They can usually be implemented via the so-called dynamic bag-of-work model [21]. In this model, a large MC task is split into smaller independent subtasks, which are then executed separately. One process or thread is designated as “master” and is responsible for the communications with the “slave” processes or threads, which perform the actual computations. Then, the partial results are collected and used to assemble an accumulated result with smaller variance than that of a single copy. The inherent characteristics of MC algorithms and the dynamic bag-of-work model make them a natural fit for the parallel architectures.

Our numerical results are obtained using the following HPC platforms:

(i) The biggest HPC resource in Bulgaria is the supercomputer BlueGene/P which is deployed at the Executive Agency "Electronic Communications Networks and Information Systems". It has two racks with 2048 PowerPC 450 processors (32 bits, 850 MHz), 8192 processor cores and a total of 4TB random access memory. The theoretical peak performance is 27.85 Tflops.

(ii) The other HPC platform is the HPC cluster deployed at the institute of information and communication technologies of the Bulgarian academy of sciences. This cluster consists of two racks which contain HP Cluster Platform Express 7000 enclosures with 36 blades BL 280c with dual Intel Xeon X5560 @ 2.8Ghz (total 576 cores), 24 GB RAM per blade. There are 8 storage and management controlling nodes 8 HP DL 380 G6 with dual Intel X5560 @ 2.8 Ghz and 32 GB RAM. All these servers are interconnected via non-blocking DDR Infiniband interconnect at 20Gbps line speed. The theoretical peak performance is 3.23 Tflops. The HPC cluster was upgraded with an HP SL390s G7 4U Lift Half Tray Server with four NVIDIA Tesla M2090 6GB Modules, included in ProLiant SL6500 Scalable System Rack. The GPU cards have 2048 CUDA cores. The peak GPU computing performance exceeds the value of 2.66 Tflops in double precision or 5.32 Tflops in single precision. The GPU computing modules are connected to the HPCG blade cluster with QDR InfiniBand cards.

Both HPC resources are connected with 1Gbps Ethernet fiber optics and all Bulgarian researchers can have access to them. Parallel programming paradigms supported by these HPC resources are Message passing, supporting several implementations of MPI: MVAPICH1/2, OpenMPI, OpenMP.

By using the Bulgarian HPC resources we were able to reduce the computing time of the MC algorithm under consideration. The simulations of the Markov chain are parallelized on the the above HPC platforms by splitting the underlying random number sequences from the SPRNG library. In our research, the MC algorithm has been implemented in C++ language. The MPI implementation was MVAPICH1.

The scalability results presented in Figures 1-2 are obtained for the problem with GaAs material parameters: the electron effective mass is 0.063, the optimal phonon energy is 36 meV, the static and optical dielectric constants are $\epsilon_s = 12.9$ and $\epsilon_{oo} = 10.92$. The initial condition is a product of two Gaussian distributions of the energy and space. The $k_z^2$ distribution corresponds to a generating laser pulse with an excess energy of about 150 meV. The $z$ distribution is centered around zero. The side of the wire is chosen to be 10 nanometers.
The values of the Wigner function \( f(z, k_z, t) \) are estimated in a rectangular domain \((-Q_1, Q_1) \times (-Q_2, Q_2)\), where \( Q_1 = 400 \text{ nm} \) and \( Q_2 = 0.66 \text{ nm}^{-1} \) consisting of \( 800 \times 260 \) points. The stochastic error for this case is relatively large. The relative mean squared error is in order of \( 10^{-3} \).

The timing results for evolution time \( t = 180 \text{ fs} \) and for all \( 800 \times 260 \) points, are shown in Table 1. The number of the Markov chain’s trajectories is 1 billion. The results are obtained on the HPC cluster deployed at IICT-BAS. Tests with hyper-threading switched on and off were performed and the results show that the use of all logical cores, which are twice as many as physical cores, improves the speed with 33% – 38%, which means that it is practical to turn hyperthreading on for these kinds of problems. In previous implementations of this code on desktop computers we have achieved higher improvements from hyperthreading. Thus we believe that the reason for the relatively small improvement is the improved efficiency of the current code, which means that more of the computations use floating point units of the processor and since these units are shared between threads, the hyperthreading can not offer more substantial gains. The parts of the code related to generation of pseudorandom numbers contain more integer operations and gain more from hyperthreading.

The results shown in Table 2 are obtained on IBM BlueGene/P. The solution again is estimated for evolution time \( t = 180 \text{ fs} \) and for all \( 800 \times 260 \) points, as the number of the Markov chain’s trajectories again is 1 billion. Both timing results demonstrate a very good speed-up and parallel efficiency.

We also implemented our algorithm using CUDA and tested it on our GPU-based resources. The random number generator that we used was the default CURAND generator from the CUDA SDK. The parallelization of
Table 2. The CPU time (seconds) for all 800 × 260 points, the speed-up, and the parallel efficiency.

<table>
<thead>
<tr>
<th>Cores</th>
<th>CPU Time (s)</th>
<th>Speed-up</th>
<th>Parallel Efficiency</th>
</tr>
</thead>
<tbody>
<tr>
<td>1024</td>
<td>23498</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>2048</td>
<td>12082</td>
<td>1.9449</td>
<td>0.97245</td>
</tr>
<tr>
<td>4096</td>
<td>6091</td>
<td>3.8769</td>
<td>0.96923</td>
</tr>
</tbody>
</table>

The code using CUDA was achieved without major rewrite of the code or changes to the program logic. The work is first split into blocks of trajectories to be computed. The master process sends the work to the slave processes, which initialize the respective GPU device and repeatedly execute the respective GPU kernel and return the results.

The same computation as above was performed in 67701 seconds on one NVIDIA M2090 card, which means that one card’s performance is comparable to that of 3 blades with hyperthreading turned off. We believe that this result can be improved, because there could be some warp divergence due to logical statements in the code. This issue can be mitigated by changes in the way the samples are computed by the threads, to make sure that the divergence is limited. We also tested the algorithm when running on several GPU cards in parallel. When 6 Nvidia M2090 cards from the same server were used to compute $10^7$ trajectories, we obtained about 93 % parallel efficiency. For such relatively small number of trajectories, the main source of inefficiency is the time spent in the cudaSetDevice call in the beginning of the computations.

5. Conclusions and future work

A quantum-kinetic model for the evolution of an initial electron distribution in a quantum wire has been introduced in terms of the electron Wigner function. The physical quantities, expressed as functionals of the Wigner function are evaluated within a MC approach. The developed MC method is characterized by the typical for quantum algorithms computational demands. The numerical results were obtained on two HPC platforms - IBM BlueGene/p and HP blade cluster. The test results show excellent parallel efficiency. The heterogeneous nature of the hardware resources that are available to the research team suggest the need to develop software that can use all those resources simultaneously. For such kinds of Monte Carlo or quasi-Monte Carlo codes it is possible to extend a dynamic load balancing scheme, that has already been developed by the team for simpler problems. Since both the cluster and the supercomputer are positioned behind a firewall, a proxy mechanism will have to be employed in order to connect the super-master of the scheme with the working nodes and the user client machine. When the full information about the Wigner function is to be collected, there will be certain strain on the external network connectivity of the cluster or the supercomputer. Thus our next step will be the development and testing of the proxy module, that can also aggregate the results before sending them upstream. We have already developed code for parallel computations using several GPU cards, in a possibly heterogeneous configuration, but
we believe we should concentrate to achieve better performance from the GPU version by some code refactoring to make the jumps more predictable.

Acknowledgment

The research work reported in the paper is partly supported by the project AComIn “Advanced Computing for Innovation”, grant 316087, funded by the FP7 Capacity Programme (Research Potential of Convergence Regions), and by the Bulgarian Ministry of Education, Youth and Science under Grant DNS7FP-02/1.

References