# Parallel Numerical Simulation of Boltzmann Transport in Single-Walled Carbon Nanotubes

Zlatan Aksamija
Electrical and Computer Engineering
University of Wisconsin-Madison

## CONTENTS



## I. OUTLINE

This document deals with the numerical simulation of electronic and thermal transport in nanostructures and the numerical methods for their simulation. The approach is based on the Boltzmann transport equation which is used to describe the behavior of both electrons (charge current carriers) and phonons (thermal current carriers) and their interaction with each other, as well as their interaction with external potentials by coupling the transport equations with the Poisson equation. The method of solving the transport equations and simulating charge and thermal transport are developed and utilized. For one-dimensional systems like single-walled carbon nanotubes (SWCNTs),

the equations for transport and potentials are discretized in space, momentum, and time, and solved by an explicit upwind method. Methods presented herein are applicable to a wide range of semiconductor nanostructures including silicon/germanium, carbon-based, and compound semiconductors, layered heterostructures, nanocomposites, as well as many geometries such as nanomembranes and nanowires.

## II. Introduction and Motivation

For many years the computer industry has relied for progress on the stellar rate of scaling in integrated circuits. The usual expectation, based on Moore's law, is that the number of transistors packed on a very large scale of integration (VLSI) chip doubles roughly every eighteen months. This requires aggressive research into the numerous bottlenecks that threaten to slow down this amazing rate. Much research has gone into photolithography needed to produce such dense circuits, into device structures that would allow smaller channel lengths, and many other issues that help this trend to continue. Now there is an emerging issue that threatens to impose an absolute limit on how many transistors can be packed onto a die. This is the issue of heat.

The amount of heat generated by the current technology continues to scale with the number of transistors in an integrated circuit, a number which is now nearing a billion. Furthermore, the introduction of multi-core architectures, now ubiquitous, has scaled the heat dissipation further, making thermal issues even more prominent. In order to stay within the allowable envelope for thermal dissipation, manufacturers most often have to resort to scaling the clock frequency back, a move which degrades performance since the number of instructions which can be executed per unit time depends directly on the operating frequency of the processor. For this reason, we can say that thermal issues are already impacting integrated circuit performance, and will continue to do so for many years to come. Indeed, the power density of modern integrated circuit chips has also followed an exponential trend, increasing by an order of magnitude roughly every eight years, based on data reported in [1]. Based on this data, on-chip power density has already exceeded that of a nuclear reactor at $100$ W/cm$^2$.

Therefore, the heat produced by a chip will very soon exceed temperatures that silicon can withstand without a phase change. When a current of electrons flows through a device, electrons collide with the crystal lattice. These collisions allow the electron to either absorb or dissipate energy in the form of a packet of thermal vibration of the lattice known as a *phonon*. In order to probe the issue of heat we must probe the mechanisms of heat generation in the next generation of nanostructures. One way to do this is by extending the capabilities of current simulation tools to include the ability to accurately simulate the generation of phonons. This allows us to examine where and how heat is generated, and thereby explore possible ways to minimize heat within the given constraints of a transistor device. In order to simulate electron-phonon interactions, we must take into account the complex details of the phonon dispersion—the relationship between phonon energy and momentum. With a more detailed knowledge of heat generation in a device, we can look at the causes of and simulate possible solutions to the problem of heat generation, including the use of nanostructures such as carbon nanotubes (CNTs).

## III. Simulation of Charge and Thermal Transport

### A. Introduction

In order to study charge and thermal transport at the nanoscale, we must consider not just the classical formulation of transport in the spatial domain, but instead we must take together both real space and reciprocal, or momentum, space aspects of charge and thermal transport. In order to do this, we use the Boltzmann transport equation (BTE). This is a semiclassical formulation of transport, which is capable of including self-consistently the transport of both electrons and phonons, as well as externally applied electric fields, electron-phonon interaction, anharmonic phonon decay, and many other types of scattering to a desired level of accuracy and detail. The principle behind the BTE is the simple idea of charge conservation. Each particle, in the present case either electron or phonon, is assumed to occupy a spatial and a momentum coordinate, which is the classical aspect of the BTE. A distribution function then counts the number of particles occupying each set of coordinates in space and momentum. Then conservation of particles in both space and momentum is enforced by equating the total rate of change in time to the change of the distribution function due to various scattering mechanisms.

### B. Boltzmann Transport Equation for Electrons

In the semiclassical formulation [2], particle position and momentum are both independent and functions only of time, so we can expand the gradient and apply the chain rule (1).

$$\frac{df(x,k,t)}{dt} = \frac{\partial f}{\partial t} + \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial k}\frac{dk}{dt} = \left(\frac{df}{dt}\right)|_{scat} \tag{1}$$

Furthermore, we can identify the derivative of position as the electron velocity $dx/dt = v(k)$. Velocity is computed from the gradient of electronic band-structure (2).

$$v(k) = \frac{1}{\hbar}\frac{dE(k)}{dk} \tag{2}$$

The derivative of momentum can be identified as the effect of the field

$$\frac{dk}{dt} = \frac{eF(x)}{\hbar}$$

where $e$ is the electron charge, and $F(x)$ is the electric field strength at a given position. Finally, we obtain the final form of the BTE (3), which is exactly the BTE that can be derived from more detailed considerations of electron conservation [2].

$$\frac{\partial f}{\partial t} + v(k)\frac{\partial f}{\partial x} + \frac{eF(x)}{\hbar}\frac{\partial f}{\partial k} = \delta f|_{scat} \tag{3}$$

This form shows us that we have a first-order system of partial differential equations to solve in dimensions of time, space, and momentum. The advantage of solving for a distribution function in this formulation is that we can now obtain the total charge along the tube quite simply by integrating the distribution function over the momentum variable and multiplying by the electron charge (4).

$$\rho(t,x) = e\int f(x,k,t)dk \tag{4}$$

The charge density can now be used to determine the strength of the electric field along the tube. Similarly, the current can be determined by integrating the velocities of all the momentum states weighted by their distribution (5).

$$I(x,t) = e \int v(k) f(x,k,t) dk \tag{5}$$

Expression (5) allows us to obtain a value for the current once the steady-state distribution function $f(x,k,t)$ is known. The coupling of external applied potentials and the electronic transport in the nanotube is achieved through the Poisson equation (6).

$$\nabla^2 V(x) = \frac{d^2 V(x)}{dx^2} = \frac{\rho(x)}{\epsilon} \tag{6}$$

### *C. Electron Scattering Rates*

As shown in (3), the forcing term of the equation is the change in the distribution function due to collisions. Electrons can interact with quantized physical vibrations, called phonons, as well as impurities and other electrons in the nanotube lattice. In the absence of scattering, the distribution would advect in space and momentum under the influence of electron velocity and applied field. Scattering serves to limit this process and bring the distribution function to a steady-state value after some initial transient time. The scattering integral can be derived by considering the probabilities of transitions from one momentum state $k$ to another state $k\prime$, represented by $S(k,k\prime)$, and reverse $S(k\prime,k)$. The probability of occupying those states is given by the distribution functions $f(k)$ and $f(k\prime)$. Then the scattering-out term is $S(k,k\prime)f(k)$ and the scattering-into of a state $k$ is $S(k\prime,k)f(k\prime)$. Taking the difference and integrating over all the states $k\prime$ gives the total change in the distribution function due to collisions (7).

$$\frac{df(x,k,t)}{dt} = \int dk\prime \left[ S(k,k\prime) f(x,k,t) - S(k\prime,k) f(x,k\prime,t) \right] \tag{7}$$

The scattering probabilities $S(k,k\prime)$ can be derived from quantum-mechanical perturbation theory, thereby including the correct scattering statistics [2]. We can also extend this calculation to include Pauli's exclusion principle, which states that no more than one electron can occupy the same quantum-mechanical state. This translates directly into an additional factor which counts the probability of a final state not being available, represented by $(1 - f(k))$. Inserting this factor complicates slightly the expression for the scattering integral (8).

$$\delta f|_{scat} = \int dk\prime \left[ S(k,k\prime) f(k)(1 - f(k\prime)) - S(k\prime,k)(1 - f(k)) f(k\prime) \right] \tag{8}$$

A slightly simpler form can be derived from (7) by assuming that the distribution function is only slightly perturbed from equilibrium by a small function, which we label $\delta f(x,k,t)$ (9).

$$\frac{d}{dt}\left(f_{eq} + \delta f\right) = \int dk\prime \left[ S(k,k\prime)\left(f_{eq} + \delta f\right) - S(k\prime,k)\left(f_{eq}\prime + \delta f\prime\right) \right] \tag{9}$$

Since the equilibrium distribution, given by the familiar Fermi-Dirac statistics (10), is in a steady state, the overall scattering integral in equilibrium is zero (11) [3]. This

principle is referred to as detailed balance, since each process is exactly balanced out by an opposite but equal counterpart, causing no net change to the distribution function.

$$f_{eq}(x,k,t) = \left[exp\left(\frac{E(k) - E_F(x)}{k_B T} + 1\right)\right]^{-1} \tag{10}$$

$$\frac{df_{eq}(x,k,t)}{dt} = \int dk\prime \left[S(k,k\prime) f_{eq}(x,k,t) - S(k\prime,k) f_{eq}(x,k\prime,t)\right] = 0 \tag{11}$$

This leaves only an expression involving $\delta f(x,k,t)$ (12).

$$\frac{d\delta f(x,k,t)}{dt} = \int dk\prime \left[S(k,k\prime)\delta f(x,k,t) - S(k\prime,k)\delta f(x,k\prime,t)\right] \tag{12}$$

The perturbed portion, $\delta f$, is an odd function of momentum $\delta f(x,k,t) = -\delta f(x,-k,t)$, while the scattering probabilities are even functions $S(k,k\prime) = S(-k,k\prime)$, so the second term in the integral cancels, leaving us with a simplified form (13).

$$\frac{d\delta f(x,k,t)}{dt} = \delta f(x,k,t) \int S(k,k\prime) dk\prime \tag{13}$$

The integral over all the scattering probabilities produces the total scattering rate $\Gamma(k)$, which is the inverse of the momentum relaxation time $\tau_m(k)$ (14).

$$\Gamma(k) = \frac{1}{\tau_m(k)} = \int S(k,k\prime) dk\prime \tag{14}$$

To first order, this probability would be calculated from Fermi's golden rule [2], [4], [5], where *H* is the matrix element, and $\omega(q)$ is the phonon frequency. The minus and plus signs refer to emission and absorption, respectively. Equation (15) is simplified somewhat by using the rigid pseudo-ion approximation [6] for the matrix element.

$$\begin{aligned} S(k,k\prime) &= \frac{2\pi}{\hbar}|H(k,k\prime)|^2 \delta(E(k) - E(k\prime) \pm \hbar\omega(q)) \\ &= \frac{2\pi}{\hbar}\frac{|D(q)|^2 I^2(q)}{\rho\omega(q)}\left(N(x,q,t) + \frac{1}{2} \pm \frac{1}{2}\right)\delta(E(k) - E(k\prime) \pm \hbar\omega(q)) \end{aligned} \tag{15}$$

Implementing this formula requires numerical integration over the whole momentum space and use of the complete phonon and electron dispersion relationships. This would then be repeated on some fine grid, and for each point we would have a numerical integration to perform [7], [8]. This would give us a probability to scatter from each position in momentum space.

$$\frac{df(x,k,t)}{dt} = \frac{\delta f(x,k,t)}{\tau_m(k)} = \frac{f(x,k,t) - f_{eq}(x,k,t)}{\tau_m(k)} \tag{16}$$

We can see from (16) that any disturbance to the equilibrium distribution relaxes back to equilibrium with a time-constant equal to the relaxation time. The time dependent solution of the BTE then represents a balance between drift and diffusion acting to displace the distribution function from equilibrium, and scattering acting to return it back to equilibrium through the relaxation time. The relaxation time can be precomputed by appropriate methods and stored for use in the simulation.

### D. Phonon Boltzmann Transport Equation

The principle of conservation of particles applies equally to phonons. Therefore, the same equation governs their transport, with some modifications. The key difference between electrons and phonons is that phonons have no charge and therefore do not interact directly with each other, or with any applied potentials. Their only interaction is through the atomic potential, leading to spontaneous decay if the potential is not perfectly quadratic, a property often termed *anharmonic* [9]. The phonon BTE will therefore have one less term on the left-hand side of (17) since the only change of the distribution function is due to the motion of phonons in real space.

$$\frac{\partial N(x,q,t)}{\partial t} + v(q)\frac{\partial N(x,q,t)}{\partial x} = \delta N(x,q,t)|_{el-ph} - \delta N(x,q,t)|_{anharm.} \tag{17}$$

The right-hand side of the phonon BTE is considerably more involved than its electron counterpart. The phonon distribution function, usually notated by $N(x,q,t)$, increases every time an electron-phonon scattering event takes place, and decreases for every anharmonic phonon decay. Therefore we will have two terms, one owing to the interactions with electrons, and the other due to anharmonicity. The first term is derived by rearranging the corresponding term for electrons. We notice that for every transition an electron makes from an initial state $k$ to a final state $k\prime$, a phonon with a momentum $q = \pm(k - k\prime)$ is either absorbed or emitted, depending on the energies of the two electron states. Every emission increases the phonon occupancy and adds one to $N(x,q,t)$, while every phonon absorption does the opposite, and decreases $N(x,q,t)$ by unity [10]. The electron states are weighted by their probability of occupancy, given by the electron distribution function $f(x,k,t)$. The total rate of change of the phonon distribution function is obtained by summing over all the electron states, as in (18).

$$\begin{aligned}\frac{dN(q)}{dt} &= \frac{2\pi}{\Omega}\int dk \frac{|D(q)|^2 I^2(q)}{\rho\hbar\omega(q)} \\ &\quad \{(N(x,q,t)+1)\, f(x,k\pm q,t)\delta(E(k\pm q) - E(k) \pm \hbar\omega(q)) \\ &\quad -N(x,q,t) f(x,k,t)\delta(E(k) - E(k\pm q) \pm \hbar\omega(q))\}\end{aligned} \tag{18}$$

The second term in the scattering portion of the phonon BTE (17) is due to coupling with other phonon modes through the anharmonic part of the atomic potential. This gives rise to a decay process for phonons which is especially important in optical branches, which, due to their higher energies, have ample opportunity to decay into pairs of acoustic phonons. Because such processes involve three phonons, they are also known as three-phonon decay or scattering. There are also higher order processes involving four or more phonons, but these are less directly applicable to our study. Also, owing to the higher order of four- and five-phonon process, the strength of the interaction is smaller by an order of magnitude or more because the atomic potential is smooth and does not give rise to significant quadric or quintic terms.

### E. Phonon Scattering and Anharmonic Decay

The anharmonic phonon decay can be thought of as very similar to the electron-phonon interaction, in the sense that we view the scattering event as one phonon entering with an initial state $q$ and two phonons exiting with final states $q\prime$ and $q \pm q\prime$. Energy and momentum are conserved, governed by the expressions $q = q\prime \pm q$" and $\omega(q) =$

$\omega(q\prime) \pm \omega(q \pm q\prime)$, similar to the electron case. This produces an expression for the rate of change of the phonon distribution function (19) that depends on the distributions at the final states in a nonlinear fashion, again much like the electron-phonon interaction.

$$\begin{aligned}\frac{dN(x,q,t)}{dt} &= \frac{2\pi}{\Omega}\int dq\prime \frac{c^2(q,q\prime,q\pm q\prime)}{\rho\hbar\omega(q)\omega(q\prime)\omega(q\pm q\prime)}\delta(\omega(q)-\omega(q\prime)\pm\omega(q\pm q\prime)) \\ &\quad \{N(x,q,t)\,(N(x,q\prime,t)+1)\,(N(x,q\pm q\prime,t)+1) \\ &\quad -(N(x,q,t)+1)\,N(x,q\prime,t)N(x,q\pm q\prime,t)\} \end{aligned} \tag{19}$$

The interaction potential is derived from the cubic portion of the atomic potential, but can be simplified considerably by assuming it depends only on an average parameter $\gamma$ called the $Grüneisen$ constant, which is determined experimentally from thermal expansion and compressibility [9]. This allows a simple relationship to be derived for the anharmonic coupling potential (20).

$$c(q,q\prime,q\pm q\prime) = -\frac{i}{\sqrt{N}}\frac{2M}{\sqrt{3}v(q)}\gamma\omega(q)\omega(q\prime)\omega(q\pm q\prime) \tag{20}$$

As for the electron case, a relaxation time expression can be derived for phonons from the BTE (18) by considering only a perturbation $\delta N$ to the equilibrium distribution function.

$$\begin{aligned}\frac{d\,(\delta N)}{dt} &= \frac{2\pi}{\Omega}\int dq\prime \frac{4M^2\gamma}{3Nv(q)}\omega(q)\omega(q\prime)\omega(q\pm q\prime) \\ &\quad \{(N_{eq}(x,q,t)+\delta N)\,(N_{eq}(x,q\prime,t)+1)\,(N_{eq}(x,q\pm q\prime,t)+1) \\ &\quad -(N_{eq}(x,q,t)+\delta N+1)\,N_{eq}(x,q\prime,t)N_{eq}(x,q\pm q\prime,t)\} \\ &\quad \delta(\omega(q)-\omega(q\prime)\pm\omega(q\pm q\prime)) \end{aligned} \tag{21}$$

The expression in (21) is simplified by considering the equilibrium condition (22), again based on detailed balance.

$$\begin{aligned}\int dq\prime\{&N_{eq}(x,q,t)\,(N_{eq}(x,q\prime,t)+1)\,(N_{eq}(x,q\pm q\prime,t)+1) \\ &-(N_{eq}(x,q,t)+1)\,N_{eq}(x,q\prime,t)N_{eq}(x,q\pm q\prime,t)\} \\ &\delta(\omega(q)-\omega(q\prime)\pm\omega(q\pm q\prime))=0 \end{aligned} \tag{22}$$

Finally we arrive at a relaxation time expression for phonons (23) which is valid as long as the final modes are near equilibrium.

$$\begin{aligned}\frac{dN(x,q,t)}{dt} &= \frac{N(x,q,t)-N_{eq}(x,q,t)}{\tau(q)} \\ &= \frac{2\pi}{\Omega}\int dq\prime \frac{4M^2\gamma}{3Nv(q)}\omega(q)\omega(q\prime)\omega(q\pm q\prime) \\ &\quad \{\delta N(x,q,t)\,(N(x,q\prime,t)+1)\,(N(x,q\pm q\prime,t)+1) \\ &\quad -\delta N(x,q,t)N(x,q\prime,t)N(x,q\pm q\prime,t)\} \\ &\quad \delta(\omega(q)-\omega(q\prime)\pm\omega(q\pm q\prime)) \end{aligned} \tag{23}$$

From expression (23) we can factor out the perturbed portion $\delta N$ and divide by it in order to obtain an expression for the three-phonon scattering rate, or the inverse of the relaxation time (24).

$$\begin{aligned}\frac{1}{\tau(q)} &= \frac{d\delta N(x,q,t)}{dt}\frac{1}{\delta N(x,q,t)} \\ &= \frac{2\pi}{\Omega}\int dq\prime \frac{4M^2\gamma\omega(q)\omega(q\prime)\omega(q\pm q\prime)}{3Nv(q)}\{N(x,q\prime,t)+N(x,q\pm q\prime,t)+1\} \\ &\quad \delta(\omega(q)-\omega(q\prime)\pm\omega(q\pm q\prime)) \end{aligned} \tag{24}$$

This integral can be evaluated numerically on a grid in momentum space and the relaxation time stored for use in simulation.

# IV. Boltzmann Transport Simulation of Single-Walled Carbon Nanotubes

## A. Introduction

Carbon nanotubes have emerged in the last decade as a potential rival to the established CMOS technology. They can be metallic or semiconducting, depending on their physical structure and radius, and therefore have been suggested as potential candidates for both device and interconnect applications. Immense research efforts have gone into understanding their chemical and physical properties. Several experiments have led to measurements of current-voltage relationships (I-V curves) in selected nanotubes, thereby shedding some light on the current transport in them [11], [12]. Several Monte Carlo studies also succeeded at elucidating some of the electron-phonon scattering properties and their relationship to transport, but applied only to the momentum space aspect of transport, by assuming a fixed field and an infinitely long tube [13], [14]. This study aims to fill the gap and study transport in both real space, where effects of the electric field are applied, and in momentum space, where electron-phonon scattering can be included accurately by directly discretizing the Boltzmann transport equation (BTE). Electron-phonon scattering is the dominant mechanism because it gives rise to heat and limits the amount of current a nanotube can transport [11], [15]. Therefore, an accurate description of electron-phonon coupling is essential. The electric field is then included self-consistently by coupling with the Poisson equation. In order to make the problem tractable for fast simulation, the entire problem is divided up in the spatial domain into strips to be solved simultaneously on a parallel computer. This technique is called "domain decomposition".

## B. Upwind Discretization of the Boltzmann Transport Equation

Under the relaxation time assumption, the BTE (3) becomes a linear parabolic equation coupled to an elliptic Poisson equation (6) through the charge density (4) and field. The boundaries of the solution domain make a rectangular box in space, time, and momentum coordinates, so the use of a finite-difference discretization is appropriate and convenient. We divide the time interval from $t_0 = 0$ up to $t_F$ into $N_n$ intervals of length $\Delta t = t_F/N_n$. Space and momentum are divided also into $N_j$ and $N_k$ segments, respectively, each of length $\Delta x = L/N_j$ and $\Delta k = K/N_k$, where $L$ is the nanotube length, $K = 2\pi/a$ is the length of the first Brillouin zone in momentum space, and $a$ is the atomic spacing of the unit cell in the particular nanotube being considered. The derivative in space can be approximated by a forward difference (25), while the derivatives in space and momentum can be either forward or backward differences (26), (27).

$$\frac{\partial}{\partial t} f(x,k,t) = \frac{f^{n+1}_{j,k} - f^{n}_{j,k}}{\Delta t} + O(\Delta t^2) \tag{25}$$

$$\begin{aligned} \frac{\partial}{\partial x} f(x,k,t) &= \frac{f^{n}_{j+1,k} - f^{n}_{j,k}}{\Delta x} + O(\Delta x^2) \\ &= \frac{f^{n}_{j,k} - f^{n}_{j-1,k}}{\Delta x} + O(\Delta x^2) \end{aligned} \tag{26}$$

$$\begin{aligned} \frac{\partial}{\partial k} f(x,k,t) &= \frac{f^{n}_{j,k+1} - f^{n}_{j,k}}{\Delta k} + O(\Delta k^2) \\ &= \frac{f^{n}_{j,k} - f^{n}_{j,k-1}}{\Delta k} + O(\Delta k^2) \end{aligned} \tag{27}$$

The derivatives in time, space, and momentum are first order, but both field and velocity can be either positive or negative in various positions in space and momentum. In order to treat this coupled set of equations, we have to use an upwind discretization scheme for the BTE, and choose the direction of differencing based on the sign of the local velocity and field [16]. Using an explicit time-stepping scheme also allows a straightforward inclusion of scattering. The update equation for the discretized distribution function can then be given as (28). Similar expressions will be derived for the phonon BTE (34), only omitting the interaction with the electric field.

$$\begin{aligned} f_{j,k}^{n+1} &= f_{j,k}^{n} + \frac{f_{j,k}^{n} - f_{eq,j,k}^{n}}{\tau_k} \\ &\quad - \frac{1 + sgn(v_k)}{2} \nu_k (f_{j,k}^{n} - f_{j,k-1}^{n}) \\ &\quad - \frac{1 - sgn(v_k)}{2} \nu_k (f_{j,k+1}^{n} - f_{j,k}^{n}) \\ &\quad - \frac{1 + sgn(F_j)}{2} \nu_j (f_{j,k}^{n} - f_{j-1,k}^{n}) \\ &\quad - \frac{1 - sgn(F_j)}{2} \nu_j (f_{j+1,k}^{n} - f_{j,k}^{n}) \end{aligned} \tag{28}$$

Here $\nu_j = |F(j\Delta x)|\frac{\Delta t}{\Delta x} \leq 1$ in space, and for momentum $\nu_k = |v(k)|\frac{\Delta t}{\Delta k} \leq 1$ for stability of the numerical scheme. These conditions, known as the Courant-Friedrichs-Lewy (CFL) conditions, restrict the maximum time step depending on the maximum field and velocity expected, as well as the mesh in space and momentum.

Furthermore, we can identify the derivative of position as the electron velocity $dx/dt = v(k)$. Velocity is computed from the gradient of electronic band-structure (2), which is obtained by zone-folding graphene data according to Equation (29). Graphene band-structure is computed using tight-binding by solving the secular Equation (30) where the matrices $H$ and $S$ are given in Equation (31). Parameters $s$ and $t$ are called the overlap and transfer integrals, and are computed from first-principles calculations. Typical values are: $\epsilon = 0$, $s = 0.129$, and $t$=-3.033 eV. Parameter $f(k)$ is the form factor of the lattice given by the positions of the atoms in space according to Equation (32). Taking only the three nearest neighbours simplifies calculation considerably. Then we obtain the final form presented in Equation (33). The resulting band-structure is computed from this equation and plotted in Fig. 1 for a (10,10) metallic carbon nanotube. Electron velocity for a (10,10) metallic single-walled carbon nanotube (SWNT) is shown in Fig. 2.

$$\mathbf{k}_{zf} = k\frac{\mathbf{K}_2}{\|\mathbf{K}_2\|} + \mu\mathbf{K}_1 \qquad \mu = 0, 1, \ldots, N-1 \tag{29}$$

$$det\,[H - ES] = 0 \tag{30}$$

$$H = \begin{pmatrix} \epsilon_{2p} & tf(k) \\ tf(k)^* & \epsilon_{2p} \end{pmatrix} \quad , \quad S = \begin{pmatrix} 1 & sf(k) \\ sf(k)^* & 1 \end{pmatrix} \tag{31}$$

$$f(\mathbf{k}) = e^{i\mathbf{k}\cdot\mathbf{R}_j} \quad , \quad j = 1, \ldots, 3 \tag{32}$$

$$\begin{aligned} E_{g2D}(\mathbf{k}) &= \frac{\epsilon_{2p} \pm tw(\mathbf{k})}{1 \pm sw(\mathbf{k})} \\ w(\mathbf{k}) = \sqrt{\|f(\mathbf{k})\|^2} &= \sqrt{1 + 4\cos\frac{\sqrt{3}k_x a}{2}\cos\frac{k_y a}{2} + 4\cos^2\frac{k_y a}{2}} \end{aligned} \tag{33}$$

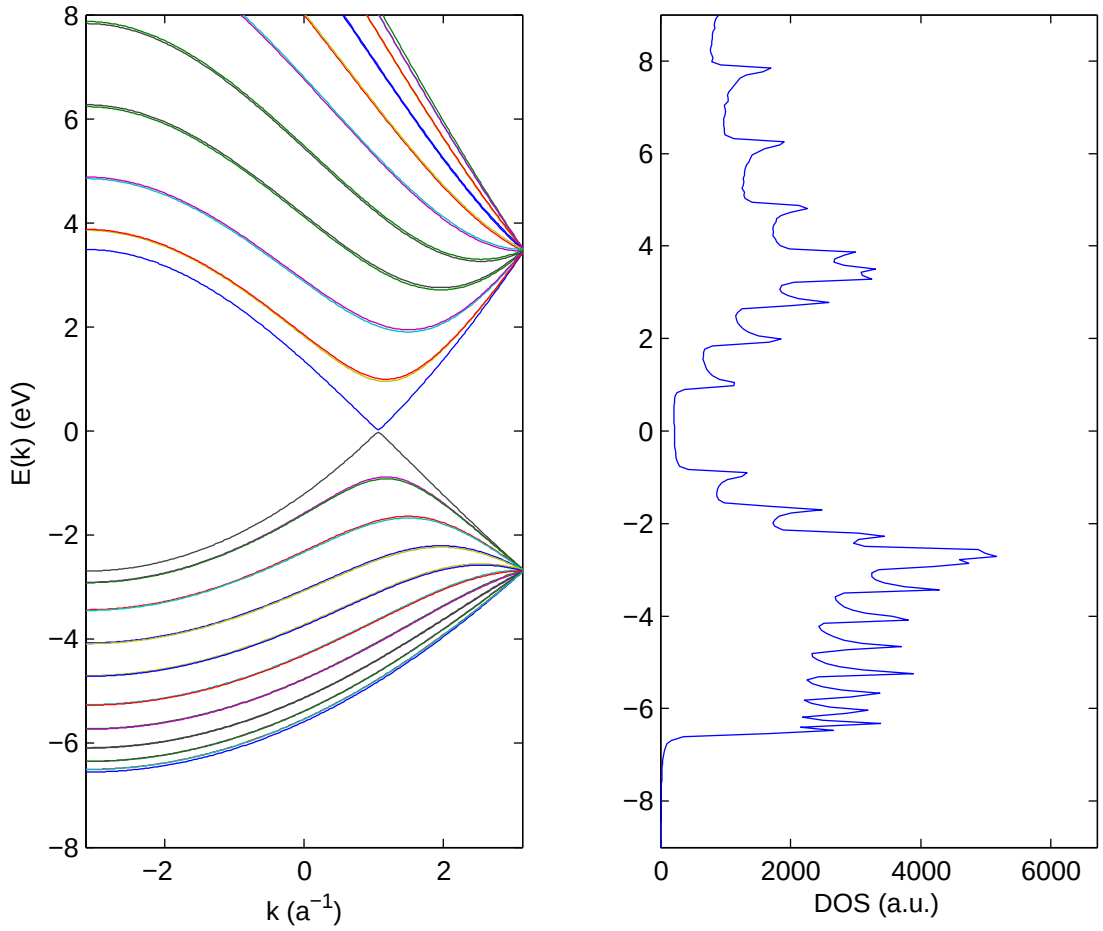


Fig. 1. Electron energy versus momentum $E(k)$ for the various bands of a (10,10) metallic nanotube, and the corresponding density-of-states (DOS). The electronic band-structure varies depending on the radius and structure of the nanotube, and determines strongly the electronic transport properties by dictating the electron velocity and influencing electron-phonon scattering rates. Two bands cross at zero energy (Fermi level) thereby making this a metallic nanotube.

In order to understand self-heating in SWNTs, we can include phonon transport in the same way as the electron transport, using the upwind discretization scheme for the BTE. Coupling between electrons and phonons can then be achieved self-consistently by solving both problems in lockstep and using the solutions for both distributions to update the subsequent time step. Electrons and phonons then influence each other through the scattering integrals (8), which have to be recomputed at each time step reflecting the

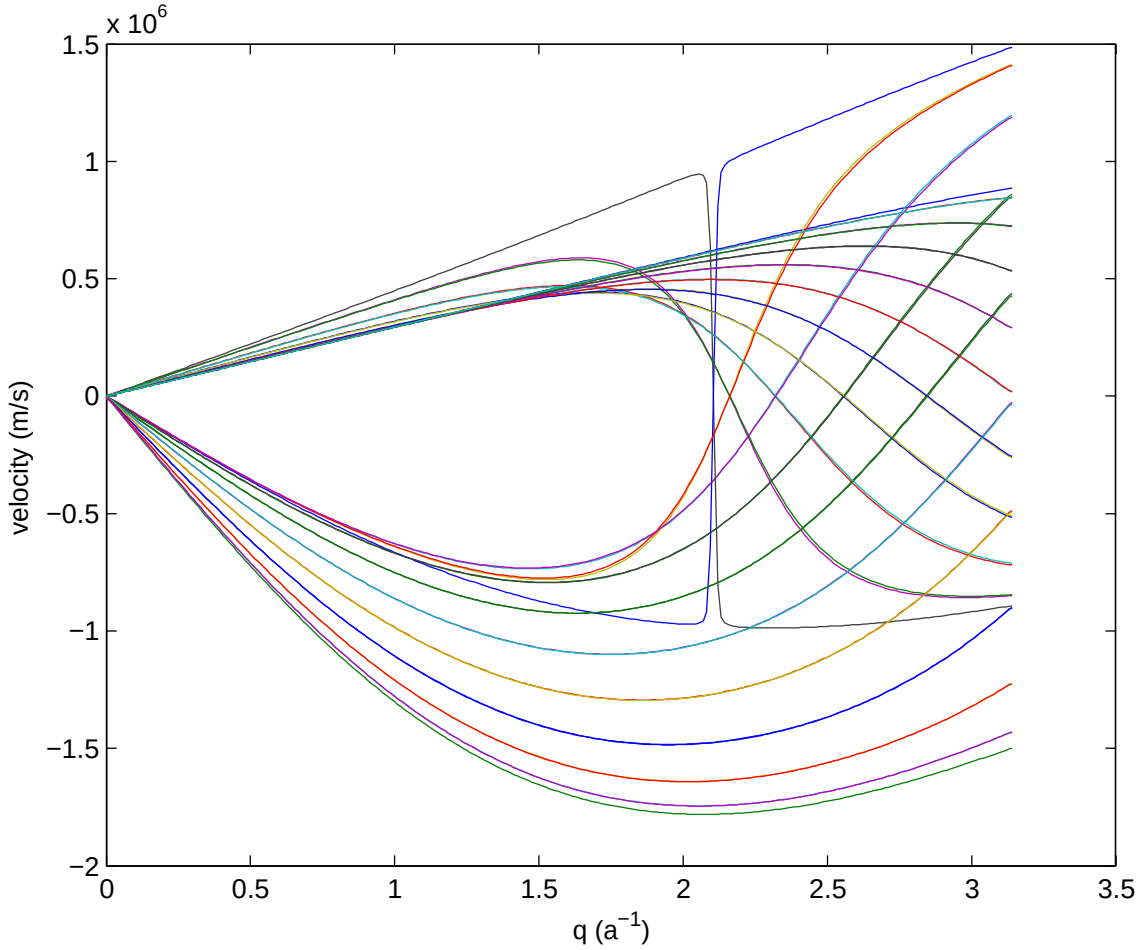


Fig. 2. Electron velocity versus momentum $v(k)$ for the various bands of a (10,10) metallic nanotube. Electron velocity is obtained as the gradient of the electronic band-structure. Electron velocity is highest for the pair of bands that cross at the Fermi level, where it reaches around $8.5 \times 10^5$ ms$^{-1}$.

updated solutions for the two distribution functions. This process can allow a detailed simulation of the complex dynamic interaction between electrons and phonons, and achieve a full coupling leading to an accurate determination of non-equilibrium phonon distributions. Phonon dispersion is also obtained from the zone-folding procedure on the graphene data shown in Fig. 3. The dispersion of a (10,10) metallic tube is plotted in Fig. 4, while the phonon velocities, obtained from the gradient of the dispersion, are shown in Fig. 5 for a few select acoustic and optical branches in order to illustrate the difference in the velocity of the acoustic and optical phonon branches, leading to a much lower contribution of optical branches to thermal transport.

$$\begin{aligned} N_{l,q}^{n+1} &= N_{l,q}^{n} + \frac{N_{l,q}^{n} - N_{eq,l,q}^{n}}{\tau_q} \\ &\quad - \frac{1 + sgn(v_q)}{2} \nu_q (N_{l,q}^{n} - N_{l,q-1}^{n}) \\ &\quad - \frac{1 - sgn(v_q)}{2} \nu_q (N_{l,q+1}^{n} - N_{l,q}^{n}) \end{aligned} \tag{34}$$

The stability condition for phonons is only restricted by the phonon velocity $\nu_q =$

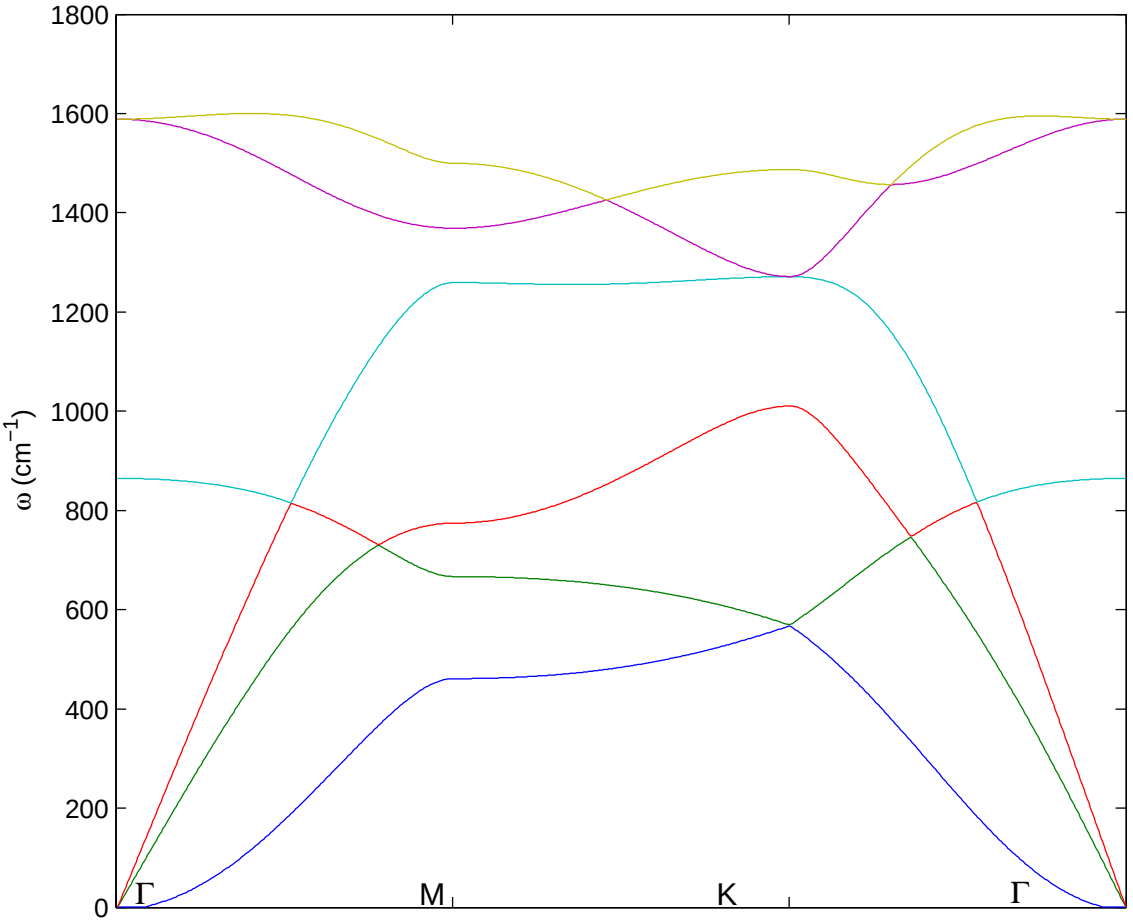


Fig. 3. Phonon dispersion relationship of graphene computed from a force constant model by considering interactions up to the fourth nearest neighbor.

$|v(q)|\frac{\Delta t}{\Delta q}$. Here once again we have for momentum $\nu_q = |v(q)|\frac{\Delta t}{\Delta k} \leq 1$ for stability of the numerical scheme.

The Poisson problem for the potential (6) can be treated on the same grid in space using finite-differences. Having only one dimension in space simplifies the problem greatly and produces a simple scheme (35).

$$V_{j+1}^n - 2V_j^n + V_{j-1}^n = \frac{\Delta x^2 \rho_j^n}{\epsilon} \tag{35}$$

The electric field strength at any point is then given by a simple derivative of the potential $V(x)$. Now the field strength at any point in space can be determined from the derivative of the potential and the result coupled back into (3). The boundary conditions at either end of the nanotube are equal to the applied external potentials. We can assume that one end is at zero voltage, say the $x = 0$ end, and the other at some voltage $V(x = L) = V_L$ which will be taken as an input parameter to the simulation.

### C. Simulation Results

By adjusting the input parameters for nanotube length and applied voltage, we can obtain a series of current-voltage characteristics for a range of lengths. We chose values

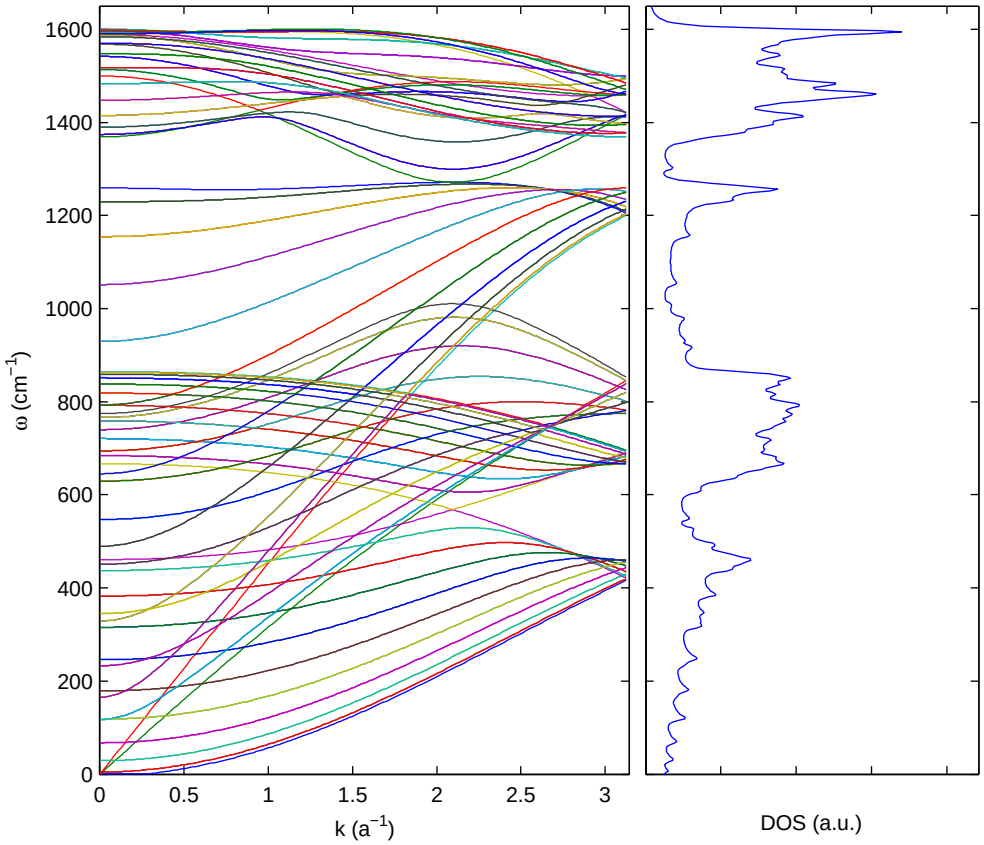


Fig. 4. Phonon dispersion versus momentum $\omega(q)$ for the various branches of a (10,10) metallic nanotube, and the corresponding lattice-vibrational density-of-states (DOS). The phonon dispersion was obtained by applying the zone-folding procedure to the dispersion of graphene. The vibrational DOS shows a strong peak in the optical modes, highlighting their contribution to the electron-phonon interaction.

ranging from 50 nm up to 1 $\mu$m, representing a range that may be encountered in future applications. The I-V curves all show the characteristic current saturation which has been confirmed experimentally on several occasions, as in Fig. 6. This is a consequence of the relaxation term increasing as the applied voltage, and consequently field, is increased, resulting in a non-linear relationship between applied voltage and the resulting current. At low bias, the current-voltage relationship is linear, thereby making the current scale also linearly with the applied electric field. Figure 7 shows that the results conform well to a line with a slope close to 80 k$\Omega$ per micron of length. Metallic carbon nanotubes, such as the (10,10) tube simulated here, are well suited for interconnect applications, where their predictable resistivity can be easily engineered.

### *D. Conclusions*

This section treated electron and transport in SWNTs in the BTE formalism. The BTE is solved self-consistently with the Poisson equation and iterated in time using an upwind finite-difference scheme until a steady state is reached. Phonon scattering is included in the relaxation time approximation. Current-voltage characteristics of small diameter metallic nanotubes are explored with lengths ranging from 50 nm up to 1 $\mu$m.

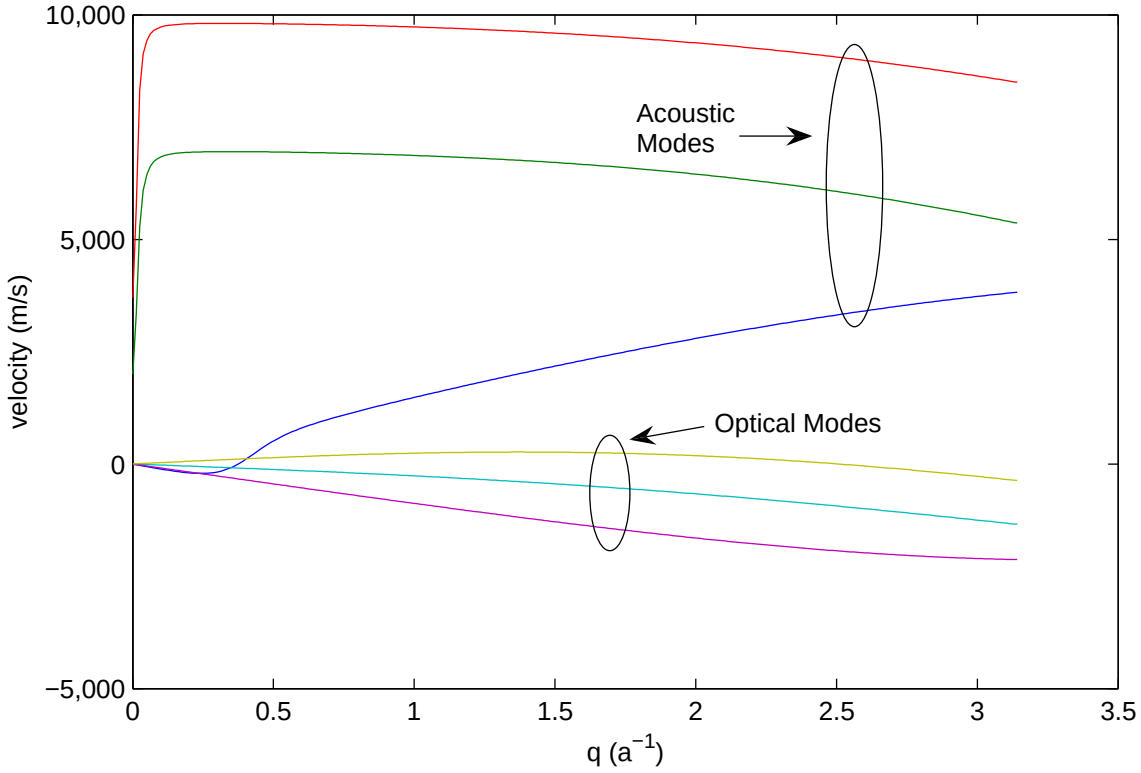


Fig. 5. Phonon group velocity versus momentum $v(q)$ for the various branches of a (10,10) metallic nanotube. Phonon velocity is low for optical modes and much higher for acoustic modes, leading to a small contribution of optical modes to thermal transport.

Current-voltage results show saturation at high bias near the value of 25 $\mu$A. Resistance of the tubes is shown to scale linearly with their length in the low bias regime with a slope of 80 k$\Omega/\mu$m.

### *E. Assessment*

Use the sequential Matlab implementation of the BTE code, or go on-line and access the interactive version of the code on the NanoHub [17] at:

`http://nanohub.org/resources/cntbte`

to explore the answers to the following questions:

1. The relaxation time was shown to encapsulate the response of the system of particles to perturbations away from equilibrium, such as the application of an external voltage. Change the relaxation time $\tau$ from its default value to a few larger and smaller values. Plot the transient current response with the same fixed voltage applied to the nanotube. Describe the effect of changing the relaxation time $\tau$ on the timing and magnitude of the current.

2. The discretization used in this code has some inherent limitations. The numerical solution has to be able to track the propagating wavefront in both the real space (as given by particle velocity) and momentum space (as determined by the strength of the applied forces acting on particles). Vary the timestep by increasing and decreasing it away from its default value. Which direction (increasing or decreasing) leads to

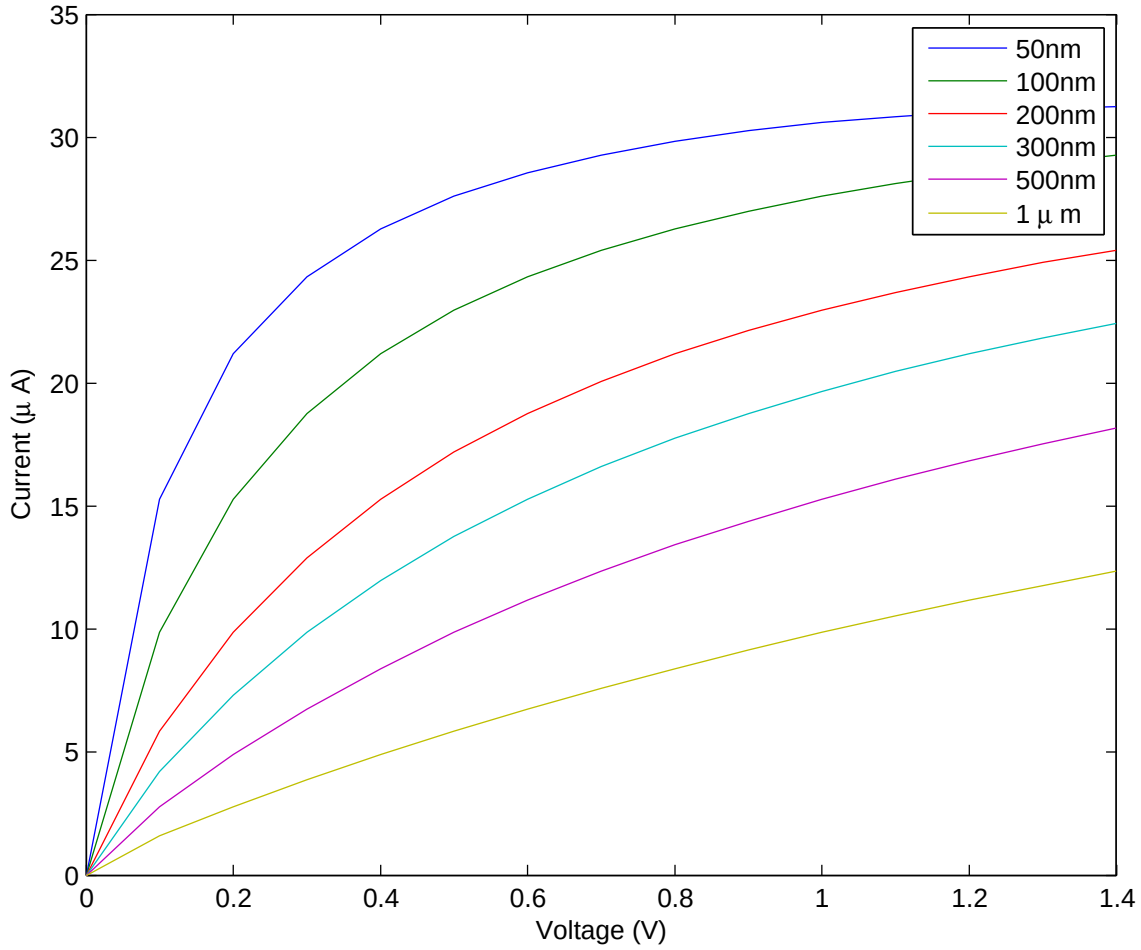


Fig. 6. Current-voltage characteristics of metallic (10,10) tubes of various length ranging from 50 nm up to 1 $\mu$m. All curves show saturation at high bias.

numerical instability (instability can be seen from a rapid oscillatory solution)? Explain why using the CFL stability criteria provided in the previous chapter.

3. Now vary the size of the discretization mesh in momentum space $\delta k$. What happens to the stability of numerical solution when $\delta k$ is decreased?

4. Vary the size of the spatial discretization mesh $\delta x$ employed in the code. What happens to the numerical stability of the solution as $\delta x$ is reduced? Now increase the applied voltage (this has the effect of "speeding up" the system in momentum space due to the larger electrostatic force experienced by the particles) and observe what happens to the stability when $\delta k$ is decreased. Is the stability criterion easier or harder to satisfy (in other words, when you increase the applied voltage, does the critical $\delta k$ at the onset of numerical instability increase or decrease)? Use the CFL stability criteria to explain your answer.

5. Based on the answers to the previous three questions, which of the two CFL conditions (one on the real space discretization $\delta x$ and the other on the momentum space $\delta k$) is more stringent (i.e. which of the two makes it easier to make the numerical solution unstable) in this particular example of carbon nanotubes? Explain why.

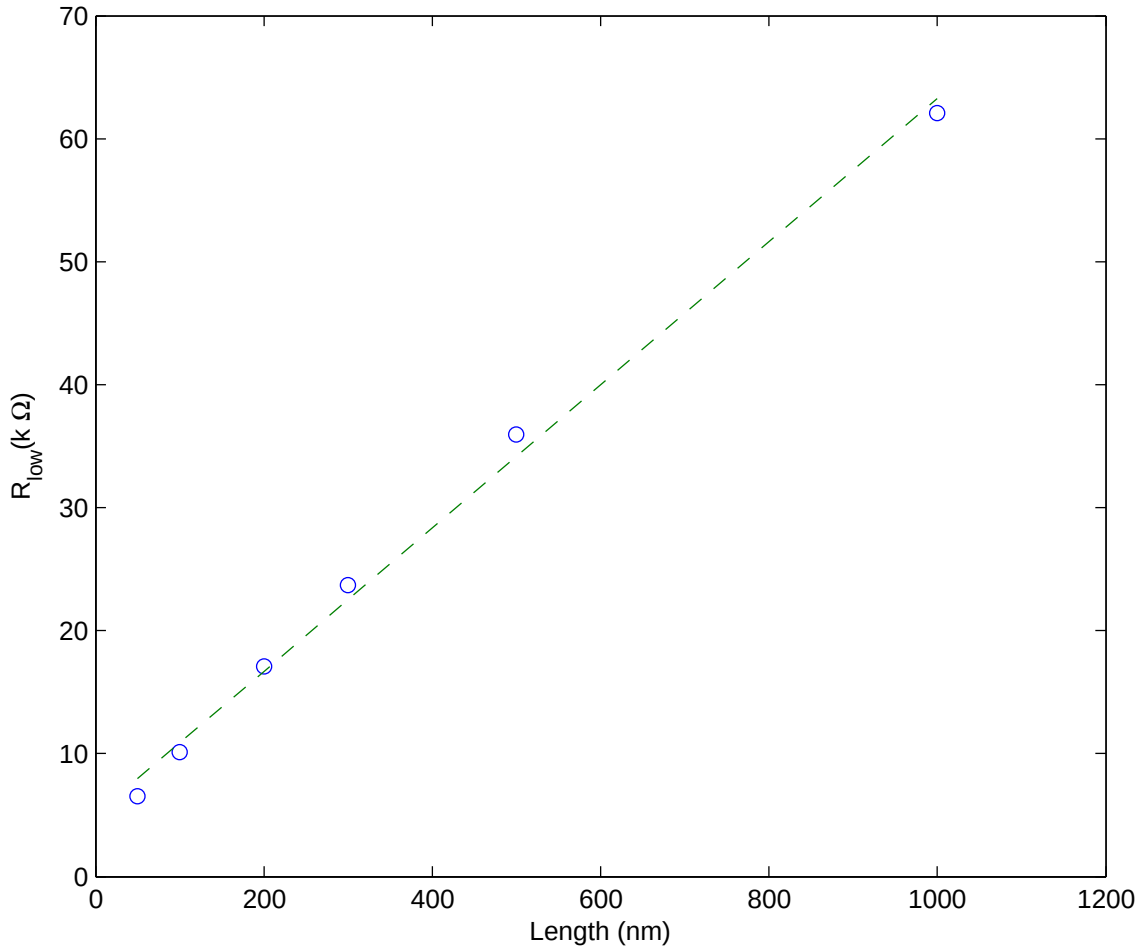


Fig. 7. Resistance versus length for a (10,10) tube at low bias (100 mV). The length dependence of the resistance is linear with a slope of around 80 kΩ per micrometer of length.

# V. Parallel Programming With The Message Passing Interface

## A. Introduction

With the advent of new and more affordable computer technologies in the last few decades, parallel processing has emerged as a powerful new technology. However, programming multi/many processor machines remains a major hurdle in the utilization of high-performance and massively parallel resources. The software hurdle must be overcome before the technology can be used effectively. The Message Passing Interface (MPI) is a language-independent computer communications descriptive application programming interface for message-passing on a parallel computer system. MPI's goals are high performance, scalability, and portability. MPI is generally considered low-level. It expresses parallelism explicitly rather than implicitly. Most MPI implementations consist of specific set of routines callable from FORTRAN, C, or C++, and any other languages capable interfacing with such routine libraries. MPI has for long stood out as the only message passing library which can be considered a standard. It is supported on virtually all HPC platforms. There is no need to modify the code when porting applications to different platform architectures. Deployment of existing code is equally possible on both shared and distributed memory hardware. Nonetheless, the target platform for MPI codes has been mainly distributed memory systems including clusters.

MPI is relatively easy to learn and implement, although perhaps not as easy as some other implicit parallel languages like OpenMP. Nonetheless, all parallelism in MPI codes is explicit: the programmer is responsible for correctly identifying parallelism and implementing the resulting algorithm using MPI constructs. For this reason, MPI has been considered successful in achieving high performance and high portability, but it has often been criticized for its low-level qualities. At present there is no effective replacement to MPI and its worldwide practical acceptance. There are several MPI implementations, mostly open source. Examples are MPI-1.2, MPI-2, MPICH, LAM, and several others that can easily be located on the internet. MPI-2 is primarily designed for C++ and FORTRAN 90 and includes new features such as scalable I/O, dynamic process management and collective communications.

## B. Basic MPI Commands

MPI programs are based on the *Single Program Multiple Data (SPMD)* paradigm. Although the details of what happens when an MPI program is executed vary from machine to machine, the essentials are the same on all machines, provided we run one process on each processor. The user issues a directive to the operating system which has the effect of placing a copy of the executable program on each processor. Each processor begins execution of its copy of the executable. Different processes can execute different statements by branching within the program. Typically the branching will be based on process ranks. Every MPI program must contain the preprocessor directive `#include "mpi.h"`. This file contains the definitions, macros and function prototypes necessary for compiling an MPI program. Before any other MPI functions can be called, the function `MPI_Init` must be called, and it should only be called once. Its arguments are pointers to the main function's parameters `argc` and `argv`. It allows systems to do any special set up so that the MPI library can be used. After a program has finished using the MPI library, it must call `MPI_Finalize`. This cleans up any unfinished business left by MPI, such as pending receives that were never completed. Note that FORTRAN uses slightly different syntax from C.

MPI provides the function `MPI_Comm_rank`, which returns the rank of a process in its second argument. Its syntax is

```
int MPI_Comm_rank(MPI_Comm comm, int rank)
```

The first argument is a *communicator*: a collection of processes that can send messages to each other. For basic programs, the only communicator needed is the general "catch-all" communicator contained in the variable `MPI_COMM_WORLD`. It is predefined in MPI and consists of all the processes running when program execution begins. Many of the constructs in our programs also depend on the number of processes executing the program, so MPI provides the function `MPI_Comm_size` for determining this. Its first argument is a communicator, and it returns the number of processes in that communicator in its second argument. Its syntax is

```
int MPI_Comm_size(MPI_Comm comm, int size)
```

The actual message passing in a parallel program is carried out by the MPI functions `MPI_Send` and `MPI_Recv`. The first command sends a message to a designated process, and second receives a message from a process. These are the most basic message passing commands in MPI. In order for the message to be successfully communicated the system must append some information to the data that the application program wishes to transmit. This additional information forms the envelope of the message, which contains the following information: the rank of the receiver, the rank of the sender, a tag, and a communicator. These items can be used by the receiver to distinguish among incoming messages. The source argument can be used to distinguish messages received from different processes. The tag is a user-specified integer that can be used to distinguish messages received from a single process.

The syntax of the send and receive commands is as follows:

```
int MPI_Send(void* message, int count, MPI_Datatype datatype,
             int dest, int tag, MPI_Comm comm)
int MPI_Recv(void* message, int count, MPI_Datatype datatype,
  int source, int tag, MPI_Comm comm, MPI_Status* status)
```

Like most functions in the standard C library most MPI functions return an integer error code; too often ignored, these error codes can be made use of for error checking purposes. With these six basic MPI commands, it is possible to set up and communicate messages between processors using the MPI interface. This will suffice for the implementation of most basic scientific parallel codes. Further improvements are possible by using overlapped and collective communication commands. However, these are not necessary to get started in implementing basic parallel solvers, such as the BTE solver example explored in the following section.

### *C. Parallel Efficiency*

The primary goal of a parallel implementation is often efficiency. Parallelism may also be introduced in order to conduct simulations at higher resolution with the same time to solution, or to take advantage of the fact that there is typically much more memory available when running on many processors than when running on a single processor. We rely on high-performance computing (HPC) resources in order to separate the work done in the program into as many independent components as possible so that the overall

execution time is reduced by executing these components simultaneously. Therefore, the ultimate goal is to make the best possible use of HPC resources by creating efficient parallel programs. Efficiency of a parallel program is measured by its effectiveness relative to the serial counterpart. It may not be immediately obvious, but there is typically more work to be done in a parallel program due to the additional overhead of dividing up the work into as many independent subtasks as possible, and the additional work necessary to synchronize the parallel processes and assemble the final solution. How is efficiency achieved in a parallel code? It is achieved by utilizing as many simultaneous processes as the problem and the hardware allow. There are several factors impacting parallel efficiency. The first is the *overhead* already mentioned, which is the additional work not present in the serial computation. The second is the *concurrency*, which is the ability to utilize many processors working alongside at the same time. The third is the *load balance*, or even distribution of work among the processors. Without the ability to perform the computation concurrently and evenly with many processors, efficiency cannot be achieved in a parallel program.

We can define the efficiency of $p$ processors working together more precisely as the ratio of the serial cost to the parallel cost $E_p = T_s/pT_s$, where $T_s$ and $T_p$ are the serial and parallel execution times. From this definition we can see that efficiency will rarely exceed unity due to the overhead incurred in setting up the parallel execution, dividing the work among $p$ processors, and assembling the final solution. We can also define the *Speedup* due to parallel execution as the ratio of serial and parallel execution times $S_p = T_s/T_p = pE_p$. Ideal efficiency of unity and ideal speedup equal to the number of processors working in parallel therefore mean the same thing: perfect load balance, maximum concurrency, and zero overhead. As we increase the number of processors working in parallel, it becomes increasingly difficult to achieve good load balance, concurrency, and low overhead. Therefore, we often talk about the *scalability* of an algorithm as its ability to maintain reasonable efficiency as the number of processors increases. We can also consider *algebraic density* as the ratio of computation to be performed on each individual portion of the data to the amount of communication required with other processes to obtain their data. Problems with large algebraic density require less interaction between processors relative to the overall work to be accomplished and make it easier to achieve high efficiency. The extreme case are those problems where each data point or subset of the data is manipulated and computed on independently, without having to interact with other data on other processes. Such problems are called *embarrassingly parallel*; some examples are simple image manipulation and some Monte Carlo integration algorithms.

One simple and popular measure of efficiency is expressed by *Amdahl's Law*. Let us divide the work in a given problem into two distinct portions: a fraction $s(0 \leq s < 1)$ which is inherently serial and cannot be broken down into independent parts, and another fraction $(1 - s)$ which is $p$-fold parallel. Then the parallel time, efficiency, and speedup can be expressed in terms of the fraction of serial work $s$ and number of processors $p$

as

$$T_p = sT_s + (1-s)\frac{T_s}{p}$$
$$E_p = \frac{T_s}{pT_p} = \frac{1}{sp + (1-s)}$$
$$S_p = \frac{T_s}{T_p} = \frac{p}{sp + (1-s)}. \tag{36}$$

From Amdahl's Law we can see that the serial fraction can severely limit parallel efficiency and scaling of a parallel algorithm. In fact, for any $s > 0$, the limit of efficiency, as we increase the number of processors $p$, goes to zero. Therefore, if as little as one percent of the work cannot be parallelized, we can never achieve speedup better than $S_p = 100$ and use more than one hundred processors efficiently. The ultimate goal of any parallel implementation is to minimize the serial fraction of the work in order to make the best use of as many processors as the hardware makes available. As future supercomputers come on-line with more and more computational power, the number of processors available to us continues to increase; this makes our work in writing efficient parallel code more challenging.

### *D. Parallel Implementation*

Since the problem discretization (28) has both space and momentum dependence at each time update, it lends itself to parallelization quite naturally. We can divide the space-momentum domain into strips either along the spatial or the momentum coordinate. Then adjacent strips couple together either by drift in momentum space or diffusion in real space. After each update, we have to sum over the grid in momentum to obtain the charge density (4), and then solve the Poisson equation on the grid in space. If we divide the domain into strips in space, then the sums on the momentum grid will not require any communication, as each grid point in space is an independent sum over all the momentum grid points. But the Poisson equation will require communication with the neighbors to solve it, as will the update equation for the distribution function (28). This means we need to pass $2N_k + 2$ doubles per processor at each iteration.

If the domain is divided into strips in momentum, then adding up the distribution to obtain the charge will require communication to perform a reduction, and the Poisson problem will then be solved on only one processor, but the update will not require any communication. Therefore we have to communicate $N_j$ doubles per processor to obtain the sum for the charge density. A possible disadvantage is that the Poisson problem is not parallelized because all the spatial dependence of the charge density ends up on one processor after the sum over the momentum grid. Using rectangular patches to divide the domain will result in an approach somewhere between the two aforementioned approaches.

The idea that appears the most natural is the division into strips in space, depicted in Fig. 8. This allows each processor to perform the sum in momentum space and obtain the charge density independently. Then the Poisson problem can be solved efficiently because it requires only communication with the neighbors, but this communication requires only two doubles to be passed per processor. Dividing the domain into rectangular patches may also be efficient for the relaxation time formulation, but might complicate more elaborate and detailed schemes for computing the scattering integral.

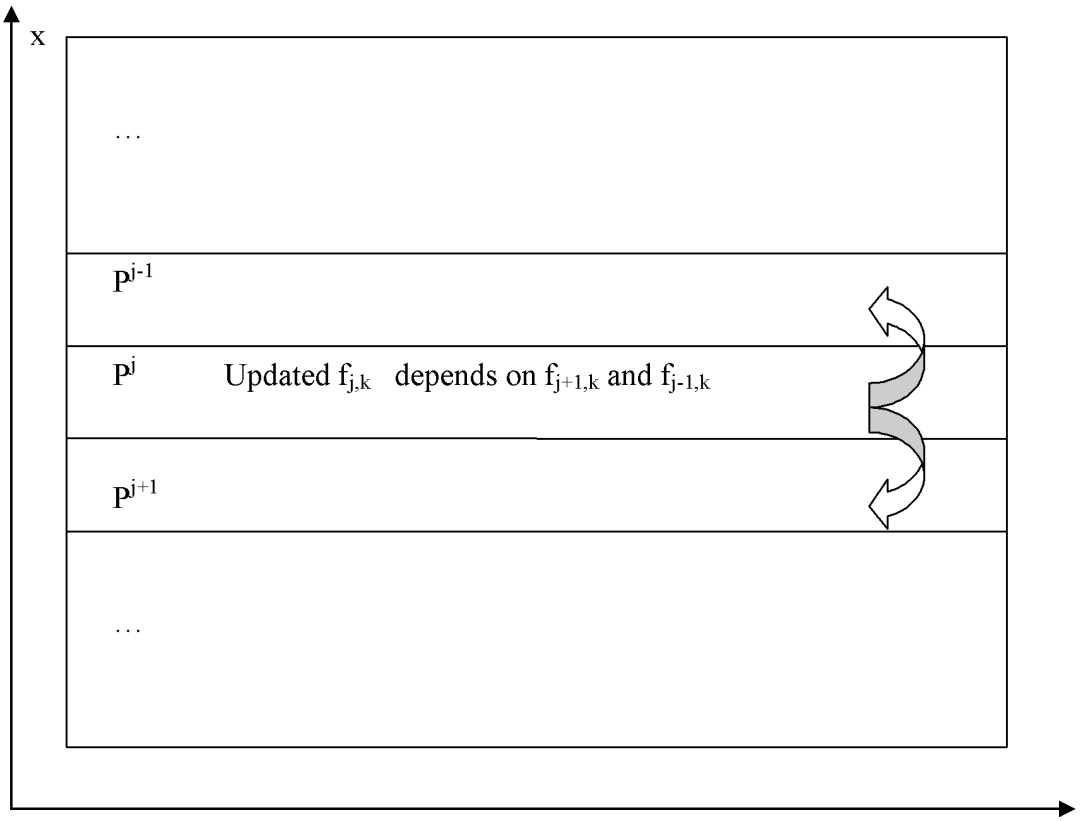


Fig. 8. Division of the space-time domain into strips in space. The update then requires communication with the immediate neighbors, as depicted. The computation of charge density in each strip is independent, and does not require any communication. Finally, solving the Poisson equation in parallel also requires communication with the neighbors. This approach makes it easy to extend and include more detailed scattering, phonon transport, and even changing properties along the nanotube in the future.

Nonetheless, for large domains and large numbers of processors, the scheme based on rectangular patches, or the two dimensional domain decomposition, will be the most flexible approach.

### *E. Parallel Performance*

An important property of a parallel implementation is the ability to scale with large numbers of processors, in order to facilitate its application to large-scale scientific problems. We tested our implementation on a range of numbers of processors on the Turing cluster in the Computational Science program at the University of Illinois and collected the running times, plotted in Fig. 9. A logarithmic plot of the running times shows a slope of $-0.81$, Fig. 10. This is close to ideal speedup, with a slope of $-1$, and allows us to predict the running time for various numbers of processors. Figure 11 shows the speedup, defined as the ratio of the sequential to the parallel running time. This curve confirms our prediction that the parallel times will scale close to linear speedup. This departure is due to additional overhead due to communication, as well as some overhead due to loading input files containing bandstructure and velocity data

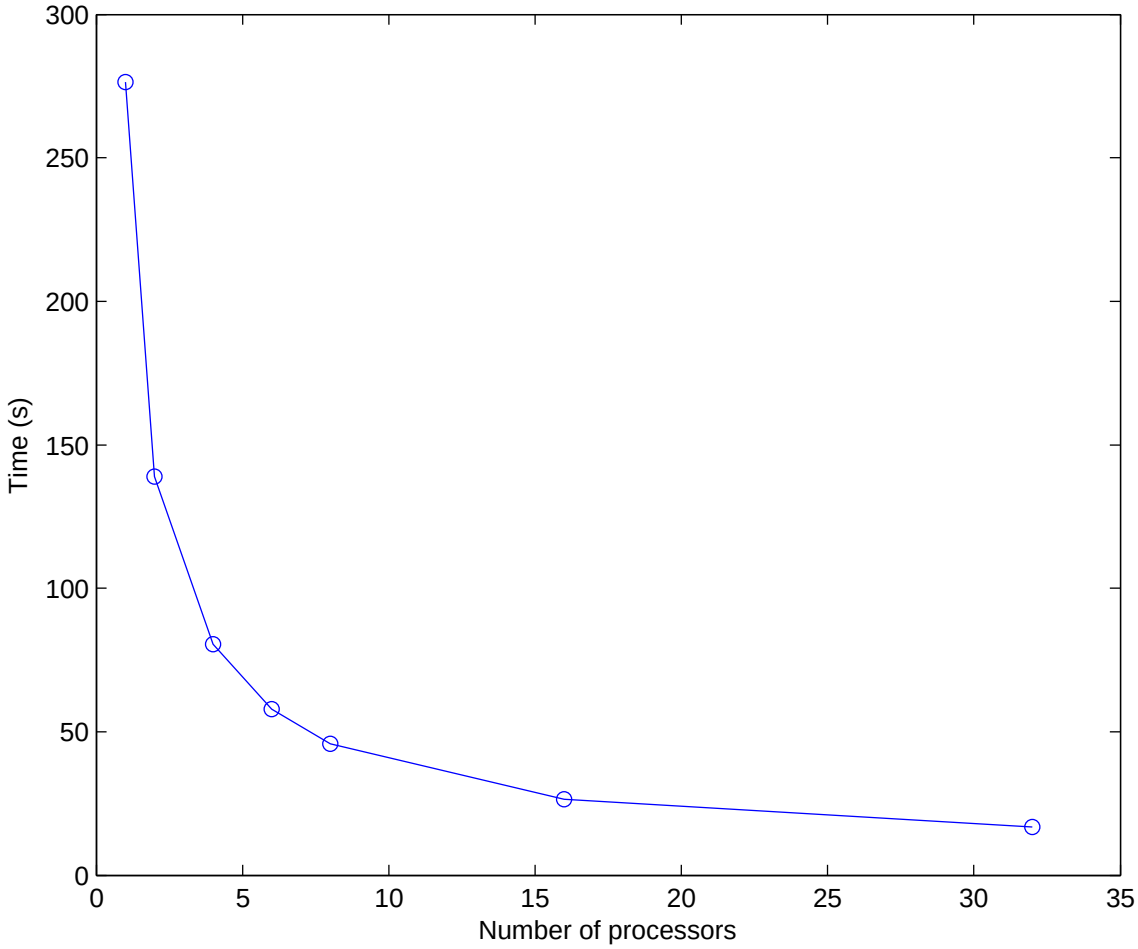


Fig. 9. Plot of running times over a range of processor numbers, showing a marked decrease in the running time of the implementation when run on a large number of processors.

which is needed for updates. Overall, this implementation is suitable for large parallel machines, and enables simulation and analysis of future nanoscale electronics to be performed in reasonable time-frames.

### *F. Assessment*

1. Use Amdahl's law to show that the efficiency of a parallel implementation can never exceed 1. What is the value of $p$ where $E_p = 1$ occurs?

2. Show that, as the number of processors increases, the speedup approaches $1/s$, the inverse of the serial fraction. (Hint: take the limit $p \to \infty$ and eliminate all terms that are small relative to $p$ i.e. all terms not involving $p$).

3. Figure 11 shows that for $p = 32$, $S_p \approx 16$. What does this tell us about the fraction of sequential work in the program? (Hint: calculate $s$ from the given speed up $S_p = 16$).

4. Assume that the sequential work for a 2-dimensional Boltzmann transport problem of dimension $N_x$ by $N_k$ is $T_s = N_x N_k$. If we divide the problem into $p$ 1-dimensional strips in space, each of dimension $N_x N_k/p$, then we add an additional communication overhead of $2N_k + 2$ per processor. Assuming that the total cost of computation does not change, calculate the resulting speedup due to the parallel implementation.

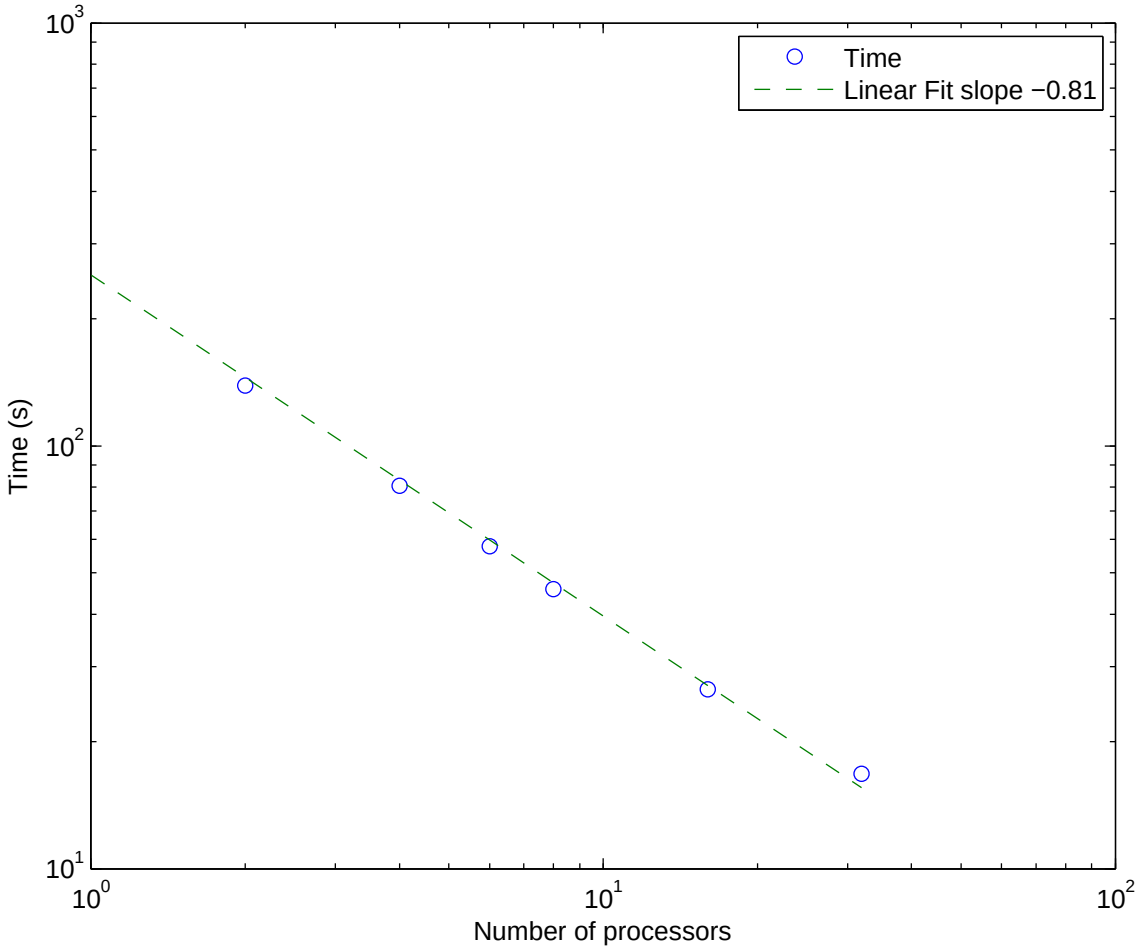


Fig. 10. Logarithmic plot of the running times (circles) and a linear fit to the data (dashed line). The slope of the linear fit is -0.81.

5. Starting with the same assumptions as in problem 4, let's divide up the problem described in the preceding chapter into $p$ equally sized rectangular patches of dimensions $N_x/\sqrt{p}$ by $N_k/\sqrt{p}$ instead of the 1-dimensional strips. Now assume the cost of communication between neighboring patches will be equal to the length of the perimeter of each patch (as it would be when we need to communicate just those points which are shared by adjacent neighbors in the problem domain) which gives a total communication cost per processor of $2N_x/\sqrt{p} + 2N_k/\sqrt{p} + 2$. Calculate the resulting speedup.

6. Plot the speedups obtained in problems 4 and 5 for values of $p$ ranging from 1 all the way up to 1024. Discuss which of the two possible decompositions of the problem gives better speedup, and over what range up $p$.

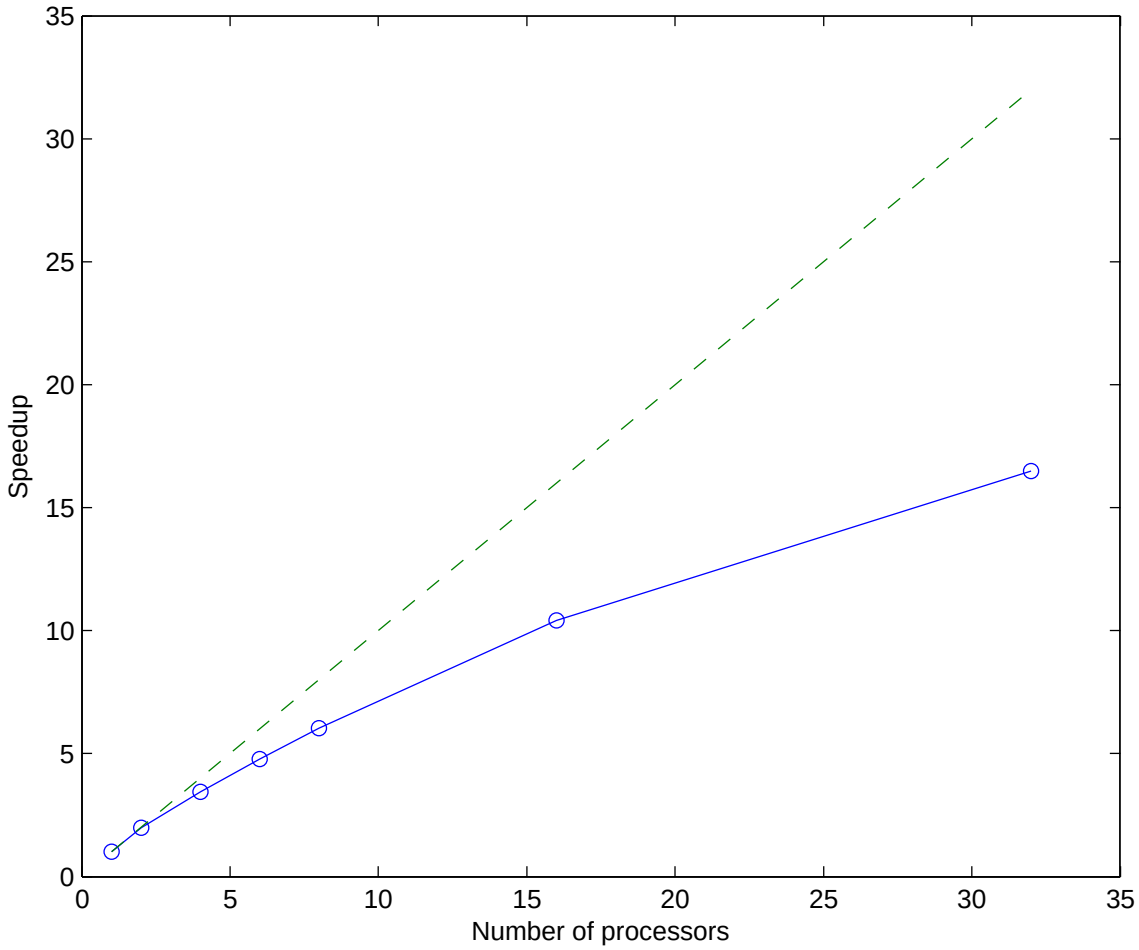


Fig. 11. Plot of speedup, or ratio of sequential to parallel running times (solid line) along with perfect linear speedup (dashed line).

## REFERENCES


[1] E. Pop, S. Sinha, and K.E. Goodson, "Heat generation and transport in nanometer-scale transistors," in *Proceedings of the IEEE*, August 2006, vol. 94, pp. 1587–1601.

[2] K. Hess, *Advanced Theory of Semiconductor Devices*, IEEE Press, New York, NY, 2000.

[3] C. Kittel, *Introduction to Solid State Physics*, John Wiley and Sons, Inc., New York, NY, 2005.

[4] P. H. Nguyen, K. R. Hofmann, and G. Paasch, "Comparative full-band Monte Carlo study of Si and Ge with screened pseudopotential-based phonon scattering rates," *Journal of Applied Physics*, vol. 94, no. 1, pp. 375–386, July 2003.

[5] C. Jacoboni and L. Reggiani, "The Monte Carlo method for the solution of charge transport in semiconductors with applications to covalent materials," *Reviews of Modern Physics*, vol. 55, no. 3, pp. 645–705, July 1983.

[6] P. H. Nguyen, K. R. Hofmann, and G. Paasch, "Full-band Monte Carlo model with screened pseudopotential based phonon scattering rates for a lattice with basis," *Journal of Applied Physics*, vol. 92, no. 9, pp. 5359–5370, November 2002.

[7] T. Kunikiyo, M. Takenaka, Y. Kamakura, M. Yamaji, H. Mizuno, M. Morifuji, K. Taniguchi, and C. Hamaguchi, "A Monte Carlo simulation of anisotropic electron transport in silicon including full band structure and anisotropic impact-ionization model," *Journal of Applied Physics*, vol. 75, no. 1, pp. 297–312, January 1994.

[8] P. D. Yoder, J. M. Higman, J. Bude, and K. Hess, "Monte Carlo simulation of hot electron transport in Si using a unified pseudopotential description of the crystal," *Semiconductor Science and Technology*, vol. 7, pp. B357–B359, 1992.

[9] F. Seitz and D. Turnbull, Eds., *Solid State Physics, vol. 7*, Academic Press, New York, NY, 1970.

[10] Esther M. Conwell, *High Field Transport in Semiconductors*, Academic Press, Inc., New York, NY, 1967.

[11] J.-Y. Park, S. Rosenblatt, Y. Yaish, V. Sazonova, H. Ustunel, S. Braig, T.A. Arias, P.W. Brouwer, and P.L. McEuen, "Electron-phonon scattering in metallic single-walled carbon nanotubes," *Nano Letters*, vol. 4, no. 3, pp. 517–520, 2004.

[12] Ali Javey, Jing Guo, Magnus Paulsson, Qian Wang, David Mann, Mark Lundstrom, and Hongjie Dai, "High-field quasiballistic transport in short carbon nanotubes," *Physical Review Letters*, vol. 92, no. 10, pp. 106804, 2004.

[13] A. Verma, M. Z. Kauser, and P. P. Ruden, "Ensemble Monte Carlo transport simulations for semiconducting carbon nanotubes," *Journal of Applied Physics*, vol. 97, no. 11, pp. 114319, 2005.

[14] G. Pennington and N. Goldsman, "Semiclassical transport and phonon scattering of electrons in semiconducting carbon nanotubes," *Physical Review B*, vol. 68, no. 4, pp. 045426, Jul 2003.

[15] Michele Lazzeri, S. Piscanec, Francesco Mauri, A. C. Ferrari, and J. Robertson, "Electron transport and hot phonons in carbon nanotubes," *Physical Review Letters*, vol. 95, no. 23, pp. 236802, 2005.

[16] K. W. Morton and D. F. Myers, *Numerical Solution of Partial Differential Equations*, University Press, Cambridge, UK, 1994.

[17] R. Kumar and R. Singh, "Simulation of carrier transport through single wall carbon nanotubes)," *International Journal Of Engineering Science*, vol. 3, no. 7, pp. 5635–5640, 2011.


## VI. AUTHOR'S BIOGRAPHY

Zlatan Aksamija graduated from the University of Illinois, Urbana-Champaign, with a B.S. in Computer Engineering in 2003 and a M.S in Electrical Engineering in 2005. He graduated with Highest Honors and as an Edmund C. James Scholar. Zlatan received his Ph.D. from the University of Illinois in 2009 in the Computational Multiscale Nanostructures group under the guidance of Prof. Umberto Ravaioli. Zlatan's research focused on the mechanisms of heat generation and transfer in nanoscale semiconductor devices, and efficient numerical algorithms for their simulation. He is the recipient of several awards, including the Gregory M. Stillman memorial graduate research award from the University of Illinois ECE department, and the First Place and Outstanding Paper awards from the IEEE Region 4 Committee for the 2007 Electro-Information Technology conference. Zlatan is also the recipient of several fellowships, including the prestigious Computational Science Graduate Fellowship from the Department of Energy and the Micron Foundation Graduate Fellowship. After the completion of his doctoral studies in 2009, Zlatan pursued a post-doctoral appointment in the Nanoelectronics Theory group at the University of Wisconsin-Madison, working on simulation of thermoelectric devices with the support from the Computing Innovation Postdoctoral Fellowship from the Computing Research Association and the "Transformative Computational Science

through Cyber-Infrastructure" (CI TraCS) Postdoctoral Fellowship from the National Science Foundation.