Parallel simulation methodological issues for creating 3D nanostructures

D Fülep, I László

***Abstract* —** We introduce our approach to the calculation of the growing 3D carbon nanostructures from graphene nanoribbons.

Our computer simulations face performance and efficiency issues so we make efforts to make our simulation method faster. Density Functional based Tight-Binding (DFTB) Molecular Dynamics (MD) simulations were performed in hybrid multi CPU - multi GPU environment.

Using these self-developed IT tools we get closer to understanding self-organized growing of nanotubes which can be the basic bricks of nano-sized electric circuits in the near future.

Keywords **—** GPU and hybrid computing, parallel simulation techniques, molecular dynamics simulation, carbon nanotube, graphene, Density Functional Tight Binding method

I. Introduction

Although the outstanding electric properties of carbon nanotubes has already been shown is several research [1], the actual use is still just a dream because mass production is still not available [2-4] due to the lack of well controlled reliable construction technology. To increase stability and decrease the effect of free bonds on the edges two nanoribbons worth to be used to construct tubular nanotubes. [8,12]. Nanotubes are the basic bricks of nano-sized electric circuits and can be constructed in several ways [5-7]. It was demonstrated that graphene patterns with atomic accuracy can develop in a self organizing way to the predetermined fullerenes or nanotubes [5, 9-11]. Until mass production initiates, the best way of research is doing computer simulations to explore possibilities of using nanotubes.

Molecular dynamics simulations can predict topological and energetical conditions [13] to be used in future mass production for growing defects-free nanotubes and their junctions [14-15] which can be the base elements of nanonetworks. For these calculations there is a demand for high-performance computing as we calculate with quite large carbon structures and a lot of experimentations in topology are needed to find appropriate shape graphene nanoribbons. Due to high computing demand we cannot use serial code any more, turning to parallel computing is a must. Although development patterns exist for the preparation of serial and parallel molecular dynamics computations, no general tool exists to do whole process of molecular dynamics simulations in every field.

Dávid Fülep: Department of Mathematics and Computational Science, Faculty of Technology Sciences, Széchenyi István University, H-9126 Győr, Hungary (fulep@sze.hu)

István László: Department of Theoretical Physics, Institute of Physics, Budapest University of Technology and Economics, H-1521 Budapest, Hungary (laszlo@eik.bme.hu)

E-mail: fulep@sze.hu, laszlo@eik.bme.hu

II. The Algorithm

The method we used is based on our previous models but has some improvements which ensure more accuracy and helps our model approaching reality. We used molecular dynamics approach to describe position and movement of atoms. In these calculations all interactions between atom pairs are calculated which means N^2 force calculations in each simulation step (N is the number of atoms) if we would use the classical calculations. In addition, the full description of the structure needs 6N parameter (3N position coordinates and 3N speed vector coordinates)

Our model has to capture the typical behavior of the nanostructures but at the same time our model should be as simple as possible to be computationally efficient. If we must choose between physical accuracy and speed of execution, we always choose an accurate representation of reality even if we face performance issues with larger models.

In our simulations it was required to use the maximum level of detail so each atom was represented as an individual particle in the model.

In the beginning of each simulation cycle there is a need to know neighborhood situation between atoms. In our model we used direct summation method where distances between all atom pairs are computed and only for those are stored in the neighbor matrix where the computed distance is less than a predefined r_m is. Distance matrix is a triangular matrix as $d_{ij} = d_{ji}$ for any atom i and j. Distance between two atoms can be computed as

$$d_{ij} = \sqrt{(R_{ix} - R_{jx})^2 + (R_{iy} - R_{jy})^2 + (R_{iz} - R_{jz})^2} \qquad (1)$$

where R_{ix}, R_{iy} and R_{iz} are the coordinates of atom R_i.

Neighborhood of two given atom is computed from interatomic distances:

$$b_{ij} = \begin{cases} 1, if\ d_{ij} < r_m \\ 0, otherwise \end{cases} \qquad (2)$$

Neighbor matrix stores all the indices of neighbors of atom i in its row i:

$$N = \begin{bmatrix} g_{11}\ g_{12}\ \dots\ 0 \\ \vdots \\ g_{n1}\ g_{n2}\ \dots\ 0 \end{bmatrix} \quad (3)$$

where g_{ij} is the number of j_{th} neighbor of atom i. Remaining matrix elements are set to 0: $N_{ij} = 0$ if j is greater than the number of neighbors of atom i.

This method ensures that unnecessarily small forces have also not been taken into account. Direct summation has quadratic runtime simplicity, but still is one of the most efficient algorithms because it is very simple and easy to parallelize.

The interatomic interaction was calculated with the help of Density Functional Tight Binding method [16]. Tight Binding is a method to calculate the electronic band structure of a crystal. Density functional theory is a theory of electronic structure involving the electron density as basic unknown instead of electronic wave function. This theory is of primary importance especially in constructing nanostructures because it decreases calculation demands of MD algorithms.

In the mathematical model energy can be described as follows:

$$E = \sum_{k=1}^{N} \frac{P_k^2}{2M_k} + E_{bond} + E_{rep} \quad (4)$$

$$E_{bond} = \sum_i n_i\ \varepsilon_i \quad (5)$$

where n_i is 0, 1 or 2. The Kohn-Sham orbitals ψ_i of the system are described in terms of atom-centered localized basis functions ϕ_v [18]. The orbitals are as follows:

$$\psi_i(r) = \sum_v^m C_{vi}\phi_v(r - R_{k_v}) \quad (6)$$

ϕ_v (r-Rv) is the basis function around the atom in position R_v, m is the number of basis functions in position R_k, $1 \le k \le N$ where N is number of atoms. R_v equals to one of R_k vectors.

Eigenvalues and eigenvectors are described with:

$$\sum_{v=1}^{m} C_{vi}\left(H_{\mu v} - \varepsilon_i S_{\mu v}\right) = 0 \quad (7)$$

on any μ and i, where

$$H_{\mu v} = \langle \Phi_\mu |H| \Phi_v \rangle \quad (8)$$

and

$$S_{\mu v} = \langle \Phi_\mu | \Phi_v \rangle \quad (9)$$

Diagonal matrix elements $H_{\mu v}$ and $S_{\mu v}$ can be computed by Slater-Koster parameter functions $H_{sp\sigma}(R)$, $H_{pp\sigma}(R)$, $H_{ss\sigma}(R)$, $H_{pp\pi}(R)$, $S_{sp\sigma}(R)$, $S_{pp\sigma}(R)$, $S_{ss\sigma}(R)$, $S_{pp\pi}(R)$. Carbon atoms have orbitals s, p_x , p_y and p_z. The functions can be described with Chebyshev polinomials [18]. Matrix elements are as follows [19]:

$$H_{ss}(R_j - R_i) = H_{ss\sigma}(R_{ij}) \quad (10)$$

$$H_{sx}(R_j - R_i) = \cos(\alpha_x)\, H_{sp\sigma}(R_{ij}) \quad (11)$$

$$H_{xx}(R_j - R_i) = \cos^2(\alpha_x)\, H_{pp\sigma}(R_{ij}) + (1 - \cos^2(\alpha_x))\, H_{pp\pi}(R_{ij}) \quad (12)$$

$$H_{xy}(R_j - R_i) = \cos(\alpha_x)\cos(\alpha_y)\, H_{pp\sigma}(R_{ij}) - \cos(\alpha_x)\cos(\alpha_y)\, H_{pp\pi}(R_{ij}) \quad (13)$$

$$H_{xz}(R_j - R_i) = \cos(\alpha_x)\cos(\alpha_z)\, H_{pp\sigma}(R_{ij}) - \cos(\alpha_x)\cos(\alpha_z)\, H_{pp\pi}(R_{ij}) \quad (14)$$

In our model, parameter functions $H_{sp\sigma}(R)$, $H_{pp\sigma}(R)$, $H_{ss\sigma}(R)$, $H_{pp\pi}(R)$, $S_{sp\sigma}(R)$, $S_{pp\sigma}(R)$, $S_{ss\sigma}(R)$, $S_{pp\pi}(R)$ are implemented as two dimensional matrices. Sizes of matrices are proportional to the number of atoms in the model. We also need to store the derivatives of these parameter functions. The so-called overlapping matrix elements are as follows:

$$S_{ss}(R_j - R_i) = S_{ss\sigma}(R_{ij}) \quad (15)$$

$$S_{sx}(R_j - R_i) = \cos(\alpha_x)\, S_{sp\sigma}(R_{ij}) \quad (16)$$

$$S_{xx}(R_j - R_i) = \cos^2(\alpha_x)\, S_{pp\sigma}(R_{ij}) + (1 - \cos^2(\alpha_x))\, S_{pp\pi}(R_{ij}) \quad (17)$$

$$S_{xy}(R_j - R_i) = \cos(\alpha_x)\cos(\alpha_y)\, S_{pp\sigma}(R_{ij}) - \cos(\alpha_x)\cos(\alpha_y)\, S_{pp\pi}(R_{ij}) \quad (18)$$

$$S_{xz}(R_j - R_i) = \cos(\alpha_x)\cos(\alpha_z)\, S_{pp\sigma}(R_{ij}) - \cos(\alpha_x)\cos(\alpha_z)\, S_{pp\pi}(R_{ij}) \quad (19)$$

for basis functions on different atoms. If two basis functions refer to same atoms, $S_{ss} = S_{xx} = S_{yy} = S_{zz} = 1$, if μ=ν and $S_{\mu v} = 0$ (μ≠ν).

E_{rep} repulsive potential is also computed with Chebyshev polinomials [18].

Constant temperature granted by Nosé-Hoover (NH) thermostat [16-18, 21, 22]. Atomic force F_k^{NH} acting on the atom in position R_k is computed using Hellmann-Feynman theorem:

$$F_k^{NH} = \sum_i \sum_{\mu,\nu}^{m} n_i \left(C_{i\mu} C_{i\nu} \left[\frac{\partial H_{\mu\nu}}{\partial \mathbf{R_k}} - \varepsilon_i \frac{\partial S_{\mu\nu}}{\partial \mathbf{R_k}} \right] \right) - \frac{\partial E_{rep}}{\partial \mathbf{R_k}} - \xi P_k \quad (20)$$

$$\xi = \frac{f}{Q} k_B (T - T_{env}) \quad (21)$$

where ξ is coefficient of friction, k_B is Boltzmann constant, Q is thermic inertia parameter, f is degree of freedom (f = 3), T_{env} is environmental temperature. Kinetic temperature T can be written in the following form:

$$T = \frac{1}{3Nk_B}\sum_{k=1}^{N}\frac{P_k^2}{2M_k} \quad (22)$$

where P_k is the momentum of the atom number k and M_k its atomic weight.

The nanoribbons were cut out from a graphene sheet of standardized interatomic distance r = 1.42 Å. The two nanoribbons are placed facing each other in parallel position. During the molecular dynamics calculation constant environmental temperature was provided [17-18].

Verlet algorithm [20] was used to calculate velocity:

$$V_k(t) = \frac{R_k(t+\Delta t)-R_k(t-\Delta t)}{2\Delta t} \quad (23)$$

$$P_k(t) = M_k V_k(t) \quad (24)$$

$$R_k(t+\Delta t) = 2R_k(t) - R_k(t-\Delta t) + \frac{F_k^{NH}(t)}{M_k}\Delta t^2 \quad (25)$$

Δt is time step (Δt = 0.7 fs), R_k is the position of the given atom, V is velocity, and F_k^{NH} is the force which act on atom in position R_k at the constant temperature granted by Nosé-Hoover (NH) thermostat [17-18, 21-22]. The initial atomic displacements during the time steps were sorted randomly and they gave the initial velocities by appropriate scaling. In this scaling we supposed an initial kinetic temperature T_{init}. This initial temperature was chosen from the range of T_{init} = 1000 K and 1100 K. As the formation of new bonds decreased the potential energy and increased the kinetic energy we had to keep the temperature constant. In a constant energy calculation the kinetic energy obtained by forming new bonds destroyed other bonds of the structure. It is evident that in the Nosé-Hoover thermostat there is an oscillation of the temperature but it cannot destroy the structure formation. In the following the temperature of the calculation will represent the temperature of the thermostat. If the constant temperature were realized with the help of random scaling of the kinetic energy we could not distinguish the temperature of the environment and the structure.

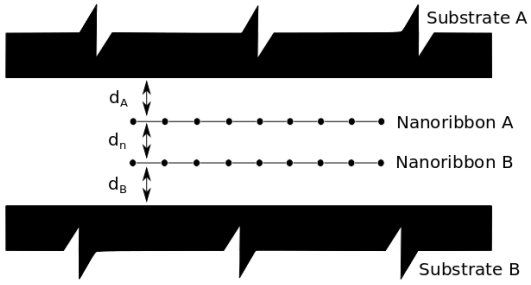

Figure 1. Initial model can be placed between two blocks of graphite substrates.

The graphene nanoribbons are placed between two blocks of graphite substrates as we can see on Figure 1. This improvement was inspired by possible production technology and aims to achieve better control of growing perfect nanotubes. Our models contain two parallel nanoribbons of d_n distance. The distance of the model is d_A from Substrate A and d_B from Substrate B. Usually $d_A = d_B = d_n$. The long-range Van der Waals interaction between the substrates and the ribbons is characterized by Lennard-Jones (LJ) term [23]. The LJ potential is as follows:

$$U_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6}\right] \quad (26)$$

Parameter *r* is the distance of the given model atom and the substrate atom. The parameters were calculated as σ = 2.9845 Å and ε = 0.002 eV.

While creating the IT representation of the algorithm we studied the performance of the method and we found it strongly resource intensive. Large matrices are in use which can be the size of 1 GB in practical simulation parameters. The limiting behavior of the whole algorithm can be described with $O(N^2)$ to $O(N^3)$ which predicts long running times.

To run parallelized the whole mathematical algorithm were disassembled into elementary pieces to find possibilities to change algorithm described in literature and found parts to run parallel. In our approach a hybrid HPC (High Performance Computer) machine is being used where we can use traditional processors (CPUs) and General Purpose Graphic Processing Units (GPGPUs). CPUs typically consists of 1 to 16 cores, GPUs consists one or two orders of magnitute more as can be seen on Figure 2.

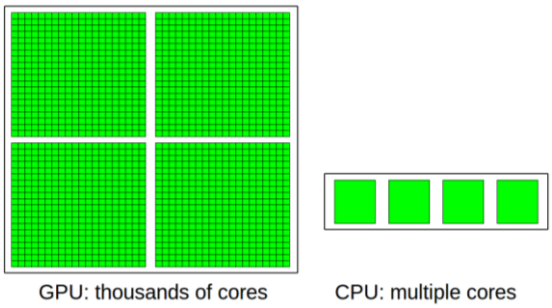

Figure 2. Initial model can be placed between two blocks of graphite substrates.

CPUs can be used for really general purpose, as it is usual in traditional sequential programming. The few cores optimized for serial processing. GPGPUs, contrary to their names, cannot be used general, we can say they can do only a narrow slice of the whole job, but they can do that extremely fast [24]. GPUs, originally developed for rendering real-time effects in 3D engineering applications and computer games now are suitable for performing data-parallel scientific computations. GPUs has a massively parallel architecture consisting hundreds or thousands of smaller, lighter, simpler,

therefore more efficient cores designed for handling multiple tasks simultaneously. Input-output operations can be a bottleneck here as GPUs use own memory for their calculations so all data has to be transferred from to main memory of the computer to the memory of the GPU board. Implementing efficient algorithms on a GPU requires deep understanding of the hardware and in many cases we have got the feeling that GPUs are a large step back in time if we examine the software development process itself. Besides that, we can point out many tasks which cannot work efficient on a GPU. That is why is makes no sense to realize everything on the GPU, only move some well-defined parts of the code there.

In our simulations a hybrid machine is used: a significant part of the code, including I/O, processing initial conditions and all the code parts where the running time is irrelevant can be processed on the CPUs. The most compute-intensive portions of the algorithm are sent to the GPUs.

Methodology was worked out to utilize possibilities of different hardware and software resources. The code was rewritten in latest version of FORTRAN and OpenMP and PGI OpenACC is used to make the algorithm parallel [25-27]. These standards aim speed up software development while try to stay near the speed of MPI (Message Passing Interface) or CUDA (Compute Unified Device Architecture). OpenMP target shared memory systems where processors share the same main memory. In contrast, MPI target both distributed as well shared memory system. In our simulations we decided on aiming shared memory systems so MPI was not used. It is because if a problem fits into a single machine (to be considered as an SMP – Symmetric MultiProcessing machine) if performs better than a DMP (Distributed Memory Parallel) architecture where we should face into the effect of the network [28, 29]. In our case a cluster machine with more simulations running at the same time made casual experimentation possible. The code, which is running on a cluster of slightly different servers, is able to discover its environment and decides how much processors (processor cores) to use and if a GPU is found then parts of the code is placed to the NVIDIA GPGPU accelerator devices. The so-called pragmas give suggestions and instructions to the FORTRAN compiler, for example:

```
!$omp do
  !$acc data copyin(a,b) copy(c)
  !$acc kernels
    (part of code to run parallelized)
  !$acc end kernels
  !$acc end data
!$omp enddo
```

Pragmas provide additional information to the compiler, beyond what is conveyed in the "host" programming language itself. They are written as a comment in the host language so they can be simply ignored when not compiled with special parallel compiler. This implicitly means that the code should be written in a way to be able to run serial.

Verification and validation of the code can be justified with real-life simulations of molecular dynamic problems of carbon nanostructures. Depending on input data, overall system performance achieved a speedup of 70 which make larger studying structures than ever before (Table 1). Though, we still see many possibilities on optimize the code in the future, thus further speedup is expected. We face to a problem that GPUs have relatively small memory on board (eg. 3GB on Tesla 2050) so we have to fit all data there. There is a large overhead associated with data transfer between central CPU memory and GPU device on-board memory. Data transfer can completely destroy the speedup effect of the GPU. For considerable larger models our computation methodology should be supplemented with data segmentation of large model data.

Hardware type	Speedup
Single CPU (1 core, serial) Intel Xeon E5-X5650	1
Multicore (12 threads) Intel Xeon E5-X5650	4
GPU (448 cores) Tesla 2050	59
GPU (512 cores) Tesla 2090	70

Table 1. Speedup on different hardware

Special verification software was created to compare output of different versions of the algorithm during development phase. The results of the simulation must be independent from the hardware resources the simulation was running on. As all the simulations are computed with a large but limited precision, the results cannot be exactly the same but differences in results can be small enough:

$$\Delta R_i < \varepsilon \tag{27}$$

where $R_i \in R$ is the i_{th} value of result set R and ε is a sufficiently small number. For verification, each atom coordinates of all simulation steps were computed using both our original serial code and the parallel code being verified. All the corresponding atom coordinates were compared and the simulation was accepted only if all the differences in coordinates were bellow 10^{-7} Å.

In addition to verifying result data with software, another usual method of validation is visualizing data (an example can be seen in Figure 3). It is extremely important as the result set of each simulation is a large set of numbers which represent the state of the nanostructure in a given time step. Thousands of time steps are calculated for each simulation, of course. All of these represent too much information to interpret in mind and if all the states of simulation meet the previous expectations in visual we can be sure that the calculations are valid.

III. Results

An IT methodology was worked out which is suitable for designing the construction of carbon nanotubes and forecasting its properties as well as circumstances of their emergence.

Based on the methodology, a serial Tight Binding Molecular Dynamics algorithm was implemented for a hybrid CPU-GPGPU HPC environment. As an important part of the methodology the algorithm was optimized and parallelized aiming better running speeds. Some parts of the algorithm achieved the speedup of 230 and other parts remained serial. A speedup of 70 reached for the algorithm as a whole. We were studying armchair and zigzag nanotubes, Y junctions and other 3D structures. We examined all the cases of different orientations and nanotube diameters to know, how the existence of the substrates influences the self-organized growing of nanotubes. The initial structure contained two parallel (coincident or similar size) graphene nanoribbons d_n = 3.35 Å from each other. We also set the initial distance of the nanoribbons and the substrates the same value: $d_A = d_B$ = 3.35 Å. We calculated the interatomic forces between the carbon atoms considering repulsive and attractive Van der Waals forces of the substrates.

It was expected that larger models would show similar behavior so we started creating nanotubes due to diameter of critical size and above. Figure 3 shows such a large model. Figure 3.a shows initial model and Figure 3.b shows the developed structure after 2 ps. It can be seen easily that the structure fits compressed is the two substrates. Then we started to pull apart the two substrates so the structure reached its final shape as seen on Figure 3.c. For better understanding Figure 3.d shows initial and the flattened model between the substrates. Note that on the figure only one graphite layer can be seen instead of each block of graphite substrates.

We made several experiments with different widths of nanoribbons. We can discover the tendency to form a graphene sheet. Even with using substrates self-organized growing of perfect nanotubes of small diameter under a certain size could not be done.

In the case of zigzag nanotubes the critical curvature energy is less, the critical ribbon width is greater than the same value at the armchair nanotubes. These simulations were performed again with and without using substrates. The initial structure must be above a certain critical size depending on the existence of the substrates. Larger models are easier to construct.

While we have defined critical widths of graphene nanoribbons for different lattice orientations we discovered not all the cases of simulations lead successfully built nanostructure. Thus, we examined the reproducibility of the full process of self-organized growing of 3D nanostructures.

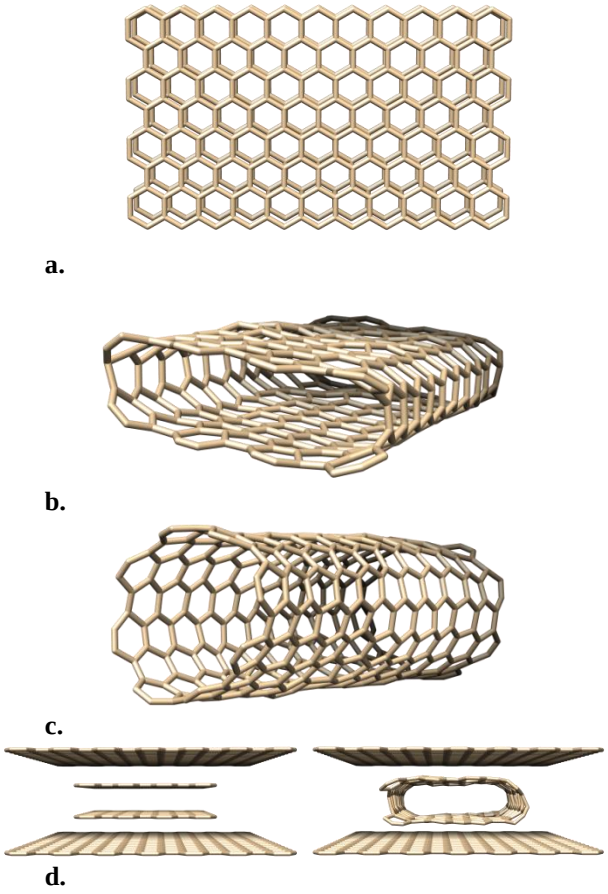

Figure 3. Simulation of carbon nanotube.
a. The initial model: two parallel nanoribbons
b. The flattened shape model at 2 ps. (Flattening caused by the two substrates)
c. The final shape of the model after the substrates were removed
d. The initial and the flattened model between the substrates (For the sake of simplicity, one graphite layer of each substrate block is shown.)

We recognized simulations started with the same parameters also can produce different outcome. The only difference is the random atomic movement of the atoms in the model which is set by pseudo-random numbers in the initialization phase of the simulation. Therefore we run several tests to analyze how our findings on critical width depend on random atomic movements. These tests should be done on each separate model but the results are similar in different cases. On Figure 4 statistics of a straight armchair nanotube is shown. We have found that using nanoribbons bellow critical width is also possible but with very low chances of success. Larger width provides high chances so using critical width is still an important quantitative data of carbon nanostructures which can be formed in future production. Our experiences with straight zigzag nanotubes are similar in tendency – but with higher ribbon widths.

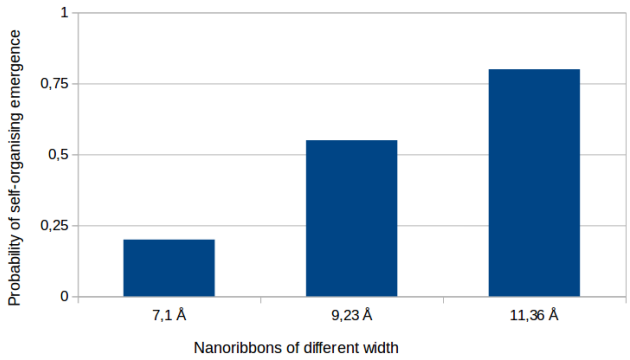

Figure 4. Probability of perfect self-organising growth of AC carbon nanotubes depends on the width of the graphene nanoribbons.

IV. Conclusion

An IT methodology was worked out for designing the construction of carbon nanotubes from parallel graphene nanoribbon. The implementation, which based on previous models and utilizes theory of molecular dynamics takes into account all the physical details of atomic behavior (interacting forces, movements etc.) of the model.

The molecular dynamics code was rewritten using the latest parallel programming techniques available for CPUs and GPUs in order to increase performance. A high performance cluster of hybrid servers were used to run the simulations. These high performance computers only performs well when the code running on them are especially designed to utilize the hardware capabilities. We achieved a speedup of 70 in our methodology compared to our old serial algorithm running. With the advanced algorithm we achieved new results in the field of materials science.

From our molecular dynamics simulations we obtained the following conditions for straight nanotube formation from two parallel nanoribbons placed between two graphite substrate blocks: For armchair nanotubes the critical ribbon width is 7.10 Å, for zigzag nanotubes is 13.53 Å. These numbers refer the case of using substrates in the simulations. Long range Van der Waals forces were also taken into consideration in computing acting forces. Both critical widths are significantly narrower than critical widths without using substrates.

These experiences add lots of useful information but still not enough to describe self-organized building of Y junctions. Y shape graphene sheets in these initial models show high sensitivity in small changes of topology and positioning. Further circumstances of creating Y junctions are the subject of our further research. In case of Y junctions we need to do several systematic test simulations to find suitable conditions and parameters.

As our IT implementation develops, we can move towards even larger 3D nano structures to study.

References

[1] Avouris P 2002 "Molecular electronics with carbon nanotubes" *Accounts Chem. Res.* 35 1026-1034

[2] Tans S J, Verschueren A R M, Dekker C 1998 "Room-temperature transistor based on a single carbon nanotube" *Nature* 393 49-52

[3] Yao Z, Postma H W C, Balents L, Dekker C 1999 "Carbon nanotube intramolecular junctions" *Nature* 402 273-276

[4] Keren K, Berman R S, Buchstab E, Sivan U, Braun E 2003 "DNA-templated carbon nanotube field-effect transistor" *Science* 302 1380-1382

[5] Fülep D, Zsoldos I, László I "Molecular dynamics simulations for lithographic production of carbon nanotube structures from graphene" 2015 Mathematics in Computer Science and Engineering Series 42, pp 253-256

[6] Tapaszto L, Dobrik G, Lambin P, Biro L P 2008 "Tailoring the atomic structure of graphene nanoribbons by scanning tunnelling microscope lithography" *Nature Nanotechnology* 3 397-401

[7] Nemes-Incze P, Magda G, Kamarás K, Biró L P 2010 "Crystallographically selective nanopatterning of graphene on SiO_2" *Nano Research* 3 110-116

[8] Han S S, Lee K S, Lee H M 2004 "Nucleation mechanism of carbon nanotube" *Chemical Physics Letters* 383 321-325

[9] László I, Zsoldos I 2012 "Graphene-based molecular dynamics nanolithography of fullerenes, nanotubes and other carbon structures" *Europhysics Letters* 99 63001

[10] László I, Zsoldos I 2012 "Molecular dynamics simulation of carbon nanostructures: The C_{60} buckminsterfullerene" *Phys. Status Solidi B* 249 2616-2619

[11] László I, Zsoldos I 2014 "Molecular dynamics simulation of carbon nanostructures: The D_{5h} C_{70} fullerene" *Physica E* 56 427-430

[12] He L, Lu J Q, Jiang H 2009 "Controlled Carbon-Nanotube Junctions Self-Assembled from Graphene Nanoribbons" *Small* 5 2802-2806

[13] D Fülep, I Zsoldos, I László: 2015 "Topological and energetic conditions for lithographic production of carbon nanotubes from graphene" *Hindawi Publishing, Journal of Nanomaterials, Volume 2015, Article ID 379563 doi 10.1155 /2015/379563*

[14] I Zsoldos: Planar trivalent polygonal networks constructed by carbon nanotube Y-junctions, Journal of Geometry and Physics 61:(1) pp. 37-45. (2011)

[15] I Zsoldos I, Gy Kakuk: New formation of carbon nanotube junctions, Modelling and Simulation in Materials Science and Engineering 15: pp. 739-747. (2007)

[16] Allen M P, Tildesley D J 1996 "Computer Simulation of Liquids" Clarendon Press, Oxford

[17] Frenkel D, Smit B 1996 "Understanding Molecular Simulation – From Algoriths to Applications" Academic Press, San Diego

[18] Porezag D, Frauenheim T, Köhler T, Seifert G and Kaschner R 1995 "Construction of tight-binding-like potentials on the basis of density-functional theory: Application to Carbon" *Phys. Rev. B* 51 12947-12957

[19] J.C. Slater, G.F. Koster 1954 "Simplified LCAO Method for the Periodic Potential Problem", Phys Rev 94:1498–1524

[20] Verlet L, 1967 "Computer experiments on classical fluids. I. Thermodynamical properties of Lennard-Jones molecules" *Phys. Rev.* 159 98-103

[21] Nosé S, 1984 "A molecular dynamics method for simulation in the canonical ensemble" *Mol. Phys.* 52 255-268

[22] Hoover W G 1985 "Canonical dynamics: Equilibrium phase-space distributions" *Phys. Rev. A* 31 1695-1697

[23] Xian-Hong Meng, Ming Li, Zhan Kang, Jian-Liang Xiao 2014 "Folding of multi-layer graphene sheets induced by van der Waals interaction" Acta Mechanica Sinica 30(3): 410-417

[24] Parallel Programming and Computing Platform, NVIDIA Corporation, 2015. Available at https://developer.nvidia.com/cuda-zone

[25] The OpenMP Architecture Review Board, 2015. Available at http://openmp.org/wp/

[26] OpenACC Directives for Accelerators, 2015. Available at http://www.openacc.org/

[27] PGI Accelerator Programming Model, The Portland Group, 2015. Available at https://www.pgroup.com/resources/docs.htm

[28] Moni Naor, Larry Stockmeyer 1995 "What can be computed locally" SIAM Journal on Computing 24 (6) 1259-1277, doi: 10.1137/S0097539793254571

[29] Hagit Attiya, Jennifer Welch 2004 "Distributed computing: fundamentals, simulations, and advanced topics" Hoboken, NJ Wiley ISBN 0-471-45324-2