Heterogeneous hardware implementation of Molecular Static method for modelling of interatomic behaviour

Lukasz Rauch*

Akademia Górniczo-Hutnicza, Al. Mickiewicza 30, 30-059 Krakow, Poland

Abstract

Heterogeneous hardware architectures and increasing range of their practical applications in recent years influence development of parallel and distributed algorithms. In the paper special attention is put on numerical simulations of interatomic behavior, which are widely used in multiscale algorithms. The algorithms implemented in this work are based on molecular static interactions, applied in simulation of nanostructural defects in metallic materials. Two aspects, i.e. qualitative and quantitative, are analysed within the paper. The first aspect is responsible for reliable simulations of interactions between nano particles on the basis of Lennard-Jones and Sutton-Chen potentials. The quantitative results present comparison of proposed approach performance for different computing devices. The results obtained for both aspects are presented in the paper and discussed in details.

© 2013 The Authors. Published by Elsevier B.V.
Selection and/or peer-review under responsibility of the organizers of the 2013 International Conference on Computational Science

Keywords: heterogeneous hardware, molecular static, modelling

1. Modeling of interatomic behavior

An unstable state of a system of atoms is usually an effect of structural defects occurring in real materials. They are simulated to obtain reliable explicit representation of material microstructure with assumed dislocation densities characteristic for specific materials. Defects in metallic materials can be divided into the following categories: point, linear and planar defects. The most common point defects are vacancies, interstitial atoms or impurities, which are met in alloys. Linear defects, usually called dislocations, are categorized into two groups i.e. edge and screw defects, while planar defects are stacking faults, twin boundaries and grain

* Corresponding author. Tel.: +48126173875; fax: +48126172921
E-mail address: lrauch@agh.edu.pl
boundaries in polycrystals. They are present in almost all kinds of metallic materials. Examples of described defects are presented in Fig. 1.

a) [Diagram of a defect in metallic material nanostructures – vacancy (a), self-interstitial atom (b), linear defect (c)]

Fig. 1. Defects in metallic material nanostructures – vacancy (a), self-interstitial atom (b), linear defect (c)

All defects in material nanostructure cause increase of a potential between particles, which have to be minimized by modification of atoms locations. Various models of interatomic potentials can be used to describe common atomic interactions. However, the main problem is to apply a potential function, which is able to describe physical and chemical properties of materials realistically. These properties depend on final atomic structure, which can be simulated numerically.

Numerical simulations of nanostructural defects require calculation of influence between large amount of atoms for relatively small sample of material. Theoretically, all atoms inside analysed structure decide about another atoms location and influence each another. In practice, atoms, which are located in significant distance, are omitted in calculations, due to a threshold applied on interatomic potentials (threshold is defined in algorithms as a cut-off radius). The potentials, e.g. Lennard-Jones or Morse, are usually implemented for testing purposes or to simulate behaviour of gases. However, in the case of metallic structures Sutton-Chen potential is much more appropriate to describe multi-atomic common influence. Differences between these potentials are visible in number of formula parameters as well as complexity of calculations, which is directly related to the complexity of applied neighbourhood. Molecular Static (MS) is one of such numerical methods, which is based on analysis of potentials and minimization of atoms energy. It aims to obtain the state of atoms equilibrium, which is achieved by changing the positions of atoms in each time step, so that the value of the global energy is reduced. The global energy of the system is calculated on the basis of a sum of atoms potentials and information about velocities received indirectly from interatomic driving forces, according to the following equation:

$$F_{ij} = -\frac{\partial \phi(r_{ij})}{\partial r_{ij}} r_{ij}$$

$$F_{ij} = -F_{ij}$$

(1)

where $F_{ij}$ is a force of $i^{th}$ particle acting on $j^{th}$ particle, $r_{ij}$ is a distance between these particles and $\phi(r_{ij})$ is a potential function dependent on distances between particles. The potential energy of $N$ atoms is obtained on the basis of atomic potentials as well and is described by the equation:

$$V(t_1, t_2, \ldots, t_N) = \sum \phi_1(t_i) + \frac{1}{2} \sum_{i,j>i} \phi_2(t_i, t_j) + \frac{1}{3} \sum_{i,j>k>l} \phi_3(t_i, t_j, t_k) + \ldots$$

(2)

where $\phi_i$ is an $i^{th}$ element potential, $r_i$ is the position vector of an $i^{th}$ atom. The first component of the formula $\phi_1$ expresses the external forces of the system, which is ignored in this paper. Atoms interactions described by
\( \phi_2 \) and three \( \phi_3 \) are applied. Many models have been presented and discussed in detail by Elizondo [1] and Liu et al. [2,3]. In this paper, Lennard-Jones and Sutton-Chen potentials are implemented (Fig 2).

Diatomic Lennard-Jones potential model [4] was presented in 1924 in the form of the following equation:

\[
\phi^{l-j}(r_{ij}) = 4\varepsilon \left[ \left( \frac{\sigma}{r_{ij}} \right)^{12} - \left( \frac{\sigma}{r_{ij}} \right)^6 \right]
\]  

(3)

where, parameter \( \varepsilon \) is the minimum value of the potential function and \( \sigma \) defines zero potential. Lennard-Jones potential considers only two body interaction, reliable mainly in application to weak interactions between molecules of noble gases (such as argon). It is also used in modelling of solids, because of its simplicity and low computational effort [5]. Component \( \left( \frac{\sigma}{r_{ij}} \right)^{12} \) (in equation 3) describes the short-range \( (r_{ij} < r_d) \) ionic repulsion to prevent penetration between them. Component \( \left( \frac{\sigma}{r_{ij}} \right)^6 \) describes van der Waals interaction (attraction between two electric dipoles for \( (r_{ij} < r_d) \)). The following properties of Lennard-Jones potential can be observed:

- \( r_{ij} = r_d \) : \( \phi^{l-j} = -\varepsilon = \Phi_{\min} \),
- \( r_{ij} = \sigma \) : \( \phi^{l-j} = 0 \),
- \( r_{ij} \to 0 \) : \( \phi^{l-j} \to \infty \),
- \( r_{ij} \to \infty \) : \( \phi^{l-j} \to 0 \).

A force acting on particular atoms can be calculated from differentiation of equation (3) as follows:

\[
f_{ij} = \frac{\varepsilon \alpha}{\sigma} \left[ 2 \left( \frac{\sigma}{r_{ij}} \right)^{13} - \left( \frac{\sigma}{r_{ij}} \right)^7 \right] \frac{r_{ij}}{r_{ij}^2}
\]  

(4)

In case of Sutton-Chen potential, a force of \( i^{th} \) atom influencing \( j^{th} \) atom is given by derivative of the potential regarding an interatomic distance \( r_{ij} \) as well. However, this potential is much more sophisticated than in Lennard-Jones equation. Sutton-Chen potential for \( i^{th} \) atom is a complex function of the positions of its neighbouring atoms and their neighbours. It is given by the following equation:

\[
\phi^{s-ch} = \varepsilon \left[ \frac{1}{2} \sum_{j \neq i} \left( \frac{\alpha}{r_{ij}} \right)^n - c \sqrt{g_i} \right]
\]  

(5)

where
\[ p_i = \sum_j \left( \frac{a}{r_{ij}} \right)^m \]  

Unitless parameters \( c, n \) and \( m \) in (5) and (6), are specific for each material. Variable \( \varepsilon \) in (5) is a scale factor of energy and \( a \) is the lattice constant. The force acting between particular atoms is transformed and written as follows:

\[ f_{ij} = -\varepsilon \left[ n \left( \frac{a}{r_{ij}} \right)^n - \frac{cm}{2} \left( \frac{1}{\sqrt[2]{r_{ij}}} + \frac{1}{\sqrt[2]{r_{ji}}} \right) \left( \frac{a}{r_{ij}} \right)^m \right] \left( \frac{1}{r_{ij}} \right) \]  

Atomic interactions for solids are more complex than for gases. Therefore, multi-element Sutton-Chen potential given in (5) is much more reliable to describe behaviour of solids [6], especially metallic materials with A1 (fcc) nanostructure. For the purposes of this work two different primitive cells were implemented i.e. A1 and A2 (bcc), which are presented in Fig. 3.

![Fig. 3. Implemented crystallographic primitive cells – A1 (a) and A2 (b).](image)

A1 structure is face-centered cubic form, typical for metals like Cu, Ag, Au, Al. The primitive cell is built of fourteen atoms placed in corners and in central points of faces. Besides metal, this structure is also present in gases like Ne, Ar, Kr, Xe. A2 structure, called also body-centered cubic, contains nine atoms placed in corners and in central point of primitive cell. It occurs in metals like Fe, Mo, W, Na [7]. Both these cells, i.e. A1 and A2, are implemented and duplicated to obtain complex nanostructures with large number of atoms. Afterwards, specific number of defects is introduced randomly and stabilization of structure is performed. Detailed description of numerical procedure is presented in section 3.

2. Motivation and objective of the work

Development of various hardware architectures [8] improved computational efficiency of many devices including non-conventional processors and computational devices like Cell Broadband Engine Architecture (CBEA) or General-Purpose Computing on Graphics Processing Units (GPGPUs). Even new projects in this area of science were started e.g. Calxeda (www.calxeda.com), Tilera (www.tilera.com) or Parallela (www.kickstarter.com/projects/adapteva/parallela-a-supercomputer-for-everyone), aiming at creation of new kind of integrated computational units. The architectures of these devices are usually based on many low frequency cores, which guarantee high computational performance as well as power consumption efficiency. These devices are usually a part of cluster computing nodes equipped with conventional CPUs, thus the...
hardware architecture is characterized by heterogeneity of internal units. Such hardware offers multi-level parallelism starting from mentioned computing nodes. On the node level the problem can be divided onto devices installed inside the node e.g. CPUs and GPGPUs. On the device level the problem can be parallelized further between cores (in multicore CPU) or between Streaming Processors (in GPGPU). The last level of parallelism is division of the problem between cores available in Streaming Processors.

This extraordinary capabilities of modern computing clusters can be efficiently applied for both types of multiscale simulations i.e. concurrent and upscaling. Algorithms, responsible for calculations in different scales, are characterized by different computational behaviour. They are calculated with different performance on different devices. Therefore, optimal efficiency can be obtain only by application of specific problem division and load balancing approaches for given problem and known target hardware.

Due to the progress of new algorithm development, hardware development as well as availability of many different hardware architectures, the need of unique software implementation technology appeared. The algorithms has to be designed to fulfil requirements of a state-of-the-art hardware, but keeping in mind that implemented methods can be run as in standalone mode or in cooperation with other numerical algorithms as complex multiscale approach [9]. Similar situation is observed in the case of nanostructural solutions like numerical simulations of defects presented in this work on the basis of Molecular Static algorithm. Therefore, the main objective of this work is to design and implement this complex approach by using unique implementation technology, which allows to test created software by using various devices. This justifies usage of OpenCL approach (http://www.khronos.org/opencl/), which is supported by a group of the most important hardware providers in the world and which can be used on the great majority of devices available on the market.

3. Details of implementation

The algorithm presented in this work is composed of four major steps, described below in details. The first step is generation of nanostructure, which is executed on a host side. This means that all computational cost of this procedure is usually handled by CPU. This cost is negligible in the case of large scale computing, because much larger computing overhead is handled by a device side. Generating of nanostructure is based on various primitive cells with respect to translational symmetry, which can be done both in 2D and 3D. The distances between each generated pair of atoms do not guarantee immediate stability of the system. The function responsible for generation of the nanostructure takes three arguments as input – one of these parameters is \( m \), the number of primitive structures in each dimension. Therefore, the function returns a rectangular block of atoms in form of one of the nanostructural primitive cells multiplied \( m \) times in each dimension. Additionally, program implements functionality allowing modification of atoms, thus there exists possibility to add extra or remove existing atom before stabilization. Afterwards, the data are allocated in the memory of computing devices and kernels are initialized. Kernels are parallel programs, which implements Embedded Atom Method (EAM). Typical parallel implementation of EAM, based on MPI, can be found in many publications [10,11] as well as in several commercial and open source molecular simulations frameworks e.g. LAMMPS (http://lammps.sandia.gov/) or NAMD (http://charm.cs.uiuc.edu/research/moladyn). The most of these solutions use distributed memory models, nevertheless the analysis of EAM performance on shared memory model with multi-core processors can be also found in the literature [12-15]. Implementations presented in these works use OpenMP, thus they are not dedicated on heterogeneous architectures. However, the work [15] proposed by Hu et al. is especially interesting, while it presents general approach to efficient implementation of parallel EAM based on spatial decomposition method with short range interactions. The method is also implemented in this paper by using OpenCL technology. The main idea of spatial decomposition, which is mapped onto the device architecture is presented in Fig. 4. The list of atom neighbours determines a size of data synchronized between particular kernels.
Kernels are the main solvers of the problem, responsible for stabilization of the moving atoms. Each kernel calculates new positions of $n/GWS$ atoms, where $n$ is total number of atoms in the structure and $GWS$ is a Global Work Size, which determines a number of all kernels executed in the device. GWS as well as LWS (Local Work Size) are specific parameters of computing devices. LWS is a number of kernels executed on single Streaming Multiprocessor. The analysis of GWS and LWS influencing the computational performance of the program is presented in section 5. The process of stabilization starts in two cases i.e. with and without defects. The first case is executed after generation of initial nanostructure, while the second one is launched each time the defect is introduced into the structure. The large number of atoms often requires high computational efforts, therefore the procedure solving each move of atoms is implemented in parallel form according to spatial decomposition (Fig. 4). Each atom has specific number of neighbours influencing its behaviour as well as final computational efficiency. This number can be determined in three different ways, which are presented in Fig 5 i.e. by a list of neighbours, by application of additional mesh or by a cut-off radius. For the purposes of this work the cut-off radius is introduced. Due to this solution the computational cost can be highly decreased, especially in case of large atomic systems. The cut-off radius is justified by applied potentials (Fig 2), which for a pair of very outlying atoms can be negligible. To calculate the force acting on analysed atom the following formula has to be used:

$$F_k = \sum_{i=1,i\neq k}^{n} F(k, i)$$  \hspace{1cm} (8)

where $k$ is an index of analysed atom, $i$ are indexes and $n$ is a number of atoms located in the cut-off radius distance, $F(k, i)$ is a force acting on atom $k$. 

![Fig. 4. Spatial domain decomposition mapped on the architecture of NVIDIA GTX 570.](image)

![Fig. 5. Different approaches for determination of neighboring atoms – list of neighbors (a), additional mesh (b), cut-off radius (c).](image)
After stabilization of nanostructure introduction of nanostructural defects is applied. This causes movement of atoms from their equilibrium positions. Defects in material structures distort the periodicity of structure and influence properties of materials. All the defects are applied manually or automatically by random applications of defects on the basis of material physical characteristics e.g. dislocation density.

4. Qualitative results

The software was tested to check reliability of proposed parallel implementation. The tests were performed for both A1 and A2 primitive cells. Example of obtained results are presented in Fig 6 for A2 nanostructure with mixed (point and linear defects). Stabilization in this case was performed without additional boundary conditions.

![Stable nanostructure A2 without defects (a), stable nanostructure after introduction of defects and stabilization (b)](image)

The qualitative results obtained for proposed software were compared to simulations by using LAMMPS, which is currently one of the most popular framework for modelling of molecular dynamics. The comparison was performed for isothermal conditions and proved high reliability of the solution proposed in this work.

5. Performance analysis

The analysis of computational efficiency was performed on the following three hardware configurations:

- 2 x Intel Xeon X5650 (12M Cache; 2.66@3.06 GHz) [2x (6cores +Hyper Threading )], Nvidia Tesla M2090, 16 GB DDR3, Linux Scientific Operating System,
- Intel Core i7 2600K (8M Cache; 3.4@3.8 GHz) [4 cores +Hyper Threading], Nvidia GTX 570, 16 GB DDR3, Linux Scientific Operating System,
- Intel Core i3 M350 (3M Cache; 2.26 GHz) [2 cores + Hyper Threading], Radeon HD5650 M, 4 GB DDR2, Linux Scientific Operating System.

The procedure of performance analysis in the case of OpenCL framework starts with determination of optimal number of LWS, which is crucial in reliable estimation of speedup and scalability. LWS is determined on the basis of computing time analysis, while LWS and GWS are equal and their value is changed in the range of admissible values for specific device. This test aims to obtain the highest performance on single Streaming Multiprocessor on GPGPU, which guarantees the best usage of computational accelerator. NVIDIA SDK offers tools for determination of the best LWS in the form of Excel sheet, nevertheless experiments performed on
various algorithms show that the best LWS strongly depends on character of the numerical procedure. In most cases the highest possible value of LWS for particular device offers the best efficiency. Such results were also obtained in the case of implementation proposed in this work for Tesla M2090 and GTX570 (Fig. 7). The test was performed for 29660 atoms in 5 iterations. However, it could not be performed on third configuration containing Radeon HD 5650, since the card was unable to handle this number of atoms. Therefore, maximum \( LWS = 256 \) was established for further calculations on this device.

Speedups were calculated for CPUs as well as for GPGPUs, excluding Intel i3, which does not support OpenCL framework. The parameters of performed numerical tests are presented in table 1.

Table 1. The parameters of calculations established for determination of speedup.

<table>
<thead>
<tr>
<th>Computational device</th>
<th>LWS</th>
<th>Range of GWS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel Xeon 5650 (2 connected devices)</td>
<td>2</td>
<td>2 – 56</td>
</tr>
<tr>
<td>Intel i7</td>
<td>2</td>
<td>2 – 16</td>
</tr>
<tr>
<td>Tesla M2090</td>
<td>1024</td>
<td>1024 – 34816</td>
</tr>
<tr>
<td>NVIDIA GTX 570</td>
<td>1024</td>
<td>1024 – 15360</td>
</tr>
<tr>
<td>Radeon HD 5650</td>
<td>256</td>
<td>256 – 2048</td>
</tr>
</tbody>
</table>

The time obtained for single Streaming Multiprocessor was used as sequential time for calculation of speedup. The results from these numerical tests are presented in Fig. 8 for CPUs and GPGPUs separately. Speedups for CPUs possess conventional character of the plots. From these data and according to Amdahl’s law it can be found that parallelization of proposed solution for Intel Xeon and i7 equals 94.17% and 83.49% respectively. The results obtained for two computational accelerators are extremely high.
Fig. 8. Comparison of speedups obtained for GPGPUs and CPUs

Fig. 9. Power consumption for Tesla M2090 (nominal power 225W) in a function of utilized streaming multiprocessors.
Additionally, the analysis of power consumption was performed by using tools delivered by NVIDIA SDK (Fig 9). The main objective was to verify if there is any possibility to setup computational strategy, aiming at saving energy and minimizing potential costs of devices maintenance, for distributed calculations by using clusters with GPGPUs. Nevertheless, the measured times of particular calculations, the scalability of proposed approach and, finally, the power consumption of the device showed that the best strategy is to use maximum computational power of GPGPU to solve the problem in the shortest time. This guarantees minimal power consumption, since for any other configuration the time of calculations multiplied by measured power usage gives higher cost of cluster maintenance.

6. Conclusions

The paper presents implementation of Molecular Static method for heterogeneous hardware architectures. The proposed software was implemented in OpenCL framework and tested on various computing devices for simulation of defects in metallic material nanostructure. The qualitative results obtained during experiments proved good reliability of the solution in comparison to other available programs. The quantitative results showed that parallel implementation is very efficient for different kinds of devices, including CPUs and GPGPUs, for which final speedup highly exceeded ideal expectations. The analysis was extended with measurements of power consumption during simulations, which led to conclusions that from this aspect point of view optimal load balancing should consider maximal usage of GPGPUs before adding another device. This can be applied to computing strategies on clusters of graphic cards aiming at minimization of maintenance costs as well as computational times.

Two main directions of the approach development are considered:
- Determination of optimal load balancing and distribution of calculations for mixed configuration of hardware e.g. for two devices available in computing node simultaneously like Xeon and Tesla on one motherboard. Initial calculations show that optimal load balancing can be obtained in such case.
- Adjustment of source codes implementation for usage as a part of multiscale approach. Analysis of performance in such a sophisticated solution will give answers on how to perform the simulations of nanostructural defects in standalone and coupled modes.

Acknowledgement

Financial assistance of the NCN project no 2011/01/D/ST6/02023 is acknowledged.
References