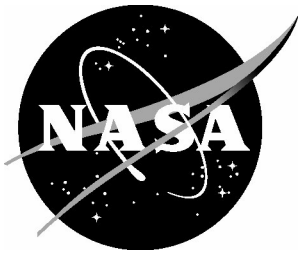
NASA/CR–2016-219202



Parallel Grand Canonical Monte Carlo (ParaGrandMC) Simulation Code

Vesselin I. Yamakov
National Institute of Aerospace, Hampton, Virginia

May 2016

NASA STI Program . . . in Profile

Since its founding, NASA has been dedicated to the advancement of aeronautics and space science. The NASA scientific and technical information (STI) program plays a key part in helping NASA maintain this important role.

The NASA STI program operates under the auspices of the Agency Chief Information Officer. It collects, organizes, provides for archiving, and disseminates NASA's STI. The NASA STI program provides access to the NTRS Registered and its public interface, the NASA Technical Reports Server, thus providing one of the largest collections of aeronautical and space science STI in the world. Results are published in both non-NASA channels and by NASA in the NASA STI Report Series, which includes the following report types:

- TECHNICAL PUBLICATION. Reports of completed research or a major significant phase of research that present the results of NASA Programs and include extensive data or theoretical analysis. Includes compilations of significant scientific and technical data and information deemed to be of continuing reference value. NASA counter-part of peer-reviewed formal professional papers but has less stringent limitations on manuscript length and extent of graphic presentations.

- TECHNICAL MEMORANDUM. Scientific and technical findings that are preliminary or of specialized interest, e.g., quick release reports, working papers, and bibliographies that contain minimal annotation. Does not contain extensive analysis.

- CONTRACTOR REPORT. Scientific and technical findings by NASA-sponsored contractors and grantees.

- CONFERENCE PUBLICATION. Collected papers from scientific and technical conferences, symposia, seminars, or other meetings sponsored or co-sponsored by NASA.

- SPECIAL PUBLICATION. Scientific, technical, or historical information from NASA programs, projects, and missions, often concerned with subjects having substantial public interest.

- TECHNICAL TRANSLATION. English-language translations of foreign scientific and technical material pertinent to NASA's mission.

Specialized services also include organizing and publishing research results, distributing specialized research announcements and feeds, providing information desk and personal search support, and enabling data exchange services.

For more information about the NASA STI program, see the following:

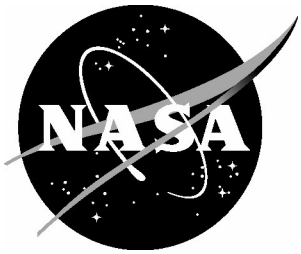
- Access the NASA STI program home page at http://www.sti.nasa.gov

- E-mail your question to help@sti.nasa.gov

- Phone the NASA STI Information Desk at 757-864-9658

- Write to:
  NASA STI Information Desk
  Mail Stop 148
  NASA Langley Research Center
  Hampton, VA 23681-2199

NASA/CR–2016-219202



Parallel Grand Canonical Monte Carlo (ParaGrandMC) Simulation Code

Vesselin I. Yamakov
National Institute of Aerospace, Hampton, Virginia

National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23681-2199

May 2016

Acknowledgements

The development of this software was initiated through funding from the NASA Aeronautics Research Institute Seeding Fund, and through cooperative agreement NCC-1-02043 with the National Institute of Aerospace. The author is especially grateful to Yuri Mishin from George Mason University for in depth discussions and helpful ideas provided through out this project.

Available from:

NASA STI Program / Mail Stop 148
NASA Langley Research Center
Hampton, VA 23681-2199
Fax: 757-864-6500

Abstract

This report provides an overview of the Parallel Grand Canonical Monte Carlo (ParaGrandMC) simulation code. This is a highly scalable parallel FORTRAN code for simulating the thermodynamic evolution of metal alloy systems at the atomic level, and predicting the thermodynamic state, phase diagram, chemical composition and mechanical properties. The code is designed to simulate multi-component alloy systems, predict solid-state phase transformations such as austenite-martensite transformations, precipitate formation, recrystallization, capillary effects at interfaces, surface absorption, etc., which can aid the design of novel metallic alloys. While the software is mainly tailored for modeling metal alloys, it can also be used for other types of solid-state systems, and to some degree for liquid or gaseous systems, including multiphase systems forming solid-liquid-gas interfaces.

1. Introduction

The objective of the Parallel Grand Canonical Monte Carlo (ParaGrandMC) simulation code is to provide a flexible computational tool to model solid-state systems, such as metal alloys, from physics based principles at the atomic level. This modeling gives insightful understanding of the physical processes that govern material properties of an alloy at the thermal and loading conditions representative for the manufacturing and processing stages, as well as in service. Such understanding increases the design space beyond what is achievable through constitutive models obtained from empirical data. As such, ParaGrandMC helps to guide the design process and to predict the functionality of the alloy in specific applications.

The code is designed to study the thermodynamic properties, phase diagram, chemical composition, and mechanical properties of condensed matter systems. The approach taken is based on evolving an initially given atomic system (defined through a list of atomic coordinates of all participating atoms) using the Monte Carlo (MC) algorithm [1]. The algorithm is based on repeated random sampling, so called Metropolis sampling [2], of the configuration space of the system using the classical Boltzmann probability distribution. In the process, internal variables of the system, such as system energy, internal pressure, system size (strain), and others, are reported continuously during the simulation. Atomic configurations, in terms of coordinates of all atoms, are stored at predefined time intervals for a post-processing analysis, such as phase identification, lattice parameter estimates, free energy integration, etc. A recently suggested parallelization algorithm [3] based on a specific domain decomposition is used. The algorithm is implemented by using Message Passing Interface (MPI) to work on distributed multi-core Central Processing Unit (CPU) platforms allowing simulations of multimillion atom systems.

The simulation of the system evolution in ParaGrandMC can be performed in the following two basic statistical thermodynamic ensembles: (i) canonical – a simulation for a constant number of atoms and fixed chemical composition; or (ii) semi-grand canonical – a simulation for a constant number of atoms, but of varying chemical composition,

which is controlled through the chemical potential of each element present in the system. Both ensembles are simulated under given constant temperature. Different loading regimes can be applied by varying the system size in all three spatial dimensions synchronously or independently to sustain constant hydrostatic pressure or constant selected stress tensor components.

Verification of ParaGrandMC code was performed in two ways. First, by comparing the results with a serial MC code, "SOLD", built by Yuri Mishin from George Mason University and tested against the broadly used Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) package [4], and second, by comparing with molecular dynamics simulations of identical systems.

The goal of this report is to provide an overview of ParaGrandMC's features and capabilities and to demonstrate its parallel performance. The theoretical foundation of the simulation methodology is given in Section 2, followed by an explanation of the parallelization technique implemented in the code in Section 3. A description of the required input and the resultant output data of the code is provided in Section 4. Analysis of the parallel performance is given in Section 5 before the document is concluded in the summary section.

2. Theoretical Foundation

2.1. General Overview

ParaGrandMC uses a variation of a lattice-free Semi-Grand canonical Monte Carlo (SGMC) simulation technique [5]. In such a simulation, a system of atoms representing a material system is evolved through a repeated random sampling of the configuration space of the system (a $3N$-dimensional space for a system of N atoms with 3 positional degrees of freedom for each atom in the three-dimensional (3D) system volume) using the classical Boltzmann probability distribution at a chosen temperature T. The algorithm consists of generating random trial moves in the configuration space that are accepted or rejected according to the Metropolis sampling scheme [2]. In this scheme, the trial move is accepted with the probability

$$P = \begin{cases} 1 & , \ \Delta\Phi \leq 0 \\ exp(-\Delta\Phi/k_B T), & \Delta\Phi > 0 \end{cases}, \tag{1}$$

where $\Delta\Phi$ is the change in the thermodynamic potential of the system during the trial move, and k_B is the Boltzmann constant. Depending on the simulated thermodynamic ensemble, $\Delta\Phi$ can take different forms, which will be described separately. Irrespective of the form, $\Delta\Phi$ always contains the change in the potential energy ΔE of the system, defined as the energy of interaction among all atoms. Interatomic potentials of Embedded Atom Method (EAM) type [6,7] and Angular Dependent Potential (ADP) types [8] are used to calculate ΔE. Their functional formulation is described in Section 4. If the move is accepted, the system state is updated to the new state. If the move is rejected, the system state remains unchanged. Execution of a number of trial moves equal to the number of atoms N, represents 1 Monte Carlo Step (MCS).

As a result, starting from an initial given configuration, the system evolves through a Markov chain of events towards a state of thermodynamic equilibrium. Once reached, thermodynamic equilibrium is maintained throughout the simulation.

The simulation of the system evolution can be performed under various constraints, depending on which type of thermodynamic ensemble is being simulated. These constraints also specify different forms of the thermodynamic potential difference $\Delta\Phi$ in Eq. (1). The following ensembles, and their variations are possible:

2.2. Canonical Monte Carlo (CMC) constant volume (NVT) ensemble

In the canonical *NVT* ensemble, the number of particles, N_α, of each chemical element, α, are kept constant, the system volume is fixed (system dimensions do not change), and the system evolution is simulated under constant temperature:

$$\begin{aligned} &N_\alpha = const. \\ &V = const. \\ &T = const. \end{aligned} \quad . \qquad (2a)$$

The trial move is defined by selecting a random atom and displacing it by a small random amount in a random direction. The move is accepted with the probability defined by Eq. (1), where $\Delta\Phi$ is equal to the change of the total potential energy of the system ΔE, resulting from the trial move

$$\Delta\Phi = \Delta E. \qquad (2b)$$

2.3. Canonical Monte Carlo (CMC) constant pressure (NPT) ensemble

In the canonical NPT ensemble, the number of particles, N_α, of each chemical element, α, are kept constant, but the system volume is allowed to fluctuate while maintaining zero internal pressure at constant *T*:

$$\begin{aligned} &N_\alpha = const. \\ &P = const. = 0. \\ &T = const. \end{aligned} \qquad (3a)$$

The volume fluctuation is achieved by an additional trial move executed once per MCS. This trial move applies a small random amount of homogeneous strain in one or several directions. The move is accepted with the probability given by Eq. (1), with

$$\Delta\Phi = \Delta E - Nk_BTln(V'/V) \qquad (3b)$$

where ΔE is the change of the potential energy due to the applied strain, and V' and V are the new and old volumes of the system, respectively.

2.4. Semi-Grand canonical Monte Carlo (SGMC) constant volume (μVT) and constant pressure (μPT) ensembles

In the semi-grand canonical ensemble, the total number of particles, *N*, is kept constant, but the number of particles for each chemical element, α, can vary, depending on a given chemical potential μ_α. The system constraints are defined as follows.

For μ*VT* ensemble:

$$\begin{aligned} &\textstyle\sum_\alpha N_\alpha = N = const. \\ &\mu_\alpha = const. \\ &V = const. \\ &T = const. \end{aligned} \quad . \tag{4a}$$

For μ*PT* ensemble:

$$\begin{aligned} &\textstyle\sum_\alpha N_\alpha = N = const. \\ &\mu_\alpha = const. \\ &P = const. = 0 \\ &T = const. \end{aligned} \quad . \tag{4b}$$

The μ*PT* ensemble involves a variation of the system size in the same way as in the canonical constant pressure (*NPT*) ensemble.

Fixing the chemical potential allows the system to achieve thermodynamically equilibrated chemical composition, rather than being constrained to a fixed composition as in the canonical ensemble. Variation of the chemical composition is achieved by modifying the trial move in the canonical ensemble where, after displacing a random atom in a random direction, its chemical type is also changed in random among all of the existing chemical element types in the system. The move is accepted with the probability defined by Eq. (1), where

$$\Delta\Phi = \Delta\mathrm{E} + \Delta\mu_{\alpha\beta} + \frac{3}{2}k_B T ln\frac{m_\alpha}{m_\beta}. \tag{4c}$$

Here, Δ*E* includes the total energy change from the combined displacement and chemical change trial move. The symbol $\Delta\mu_{\alpha\beta} = \mu_\alpha - \mu_\beta$, is the chemical potential difference between the newly assigned and the old chemical elements, α and β, of the chosen atom of atomic mass m_α and m_β, respectively. Note that when α≡β, Eq. (4c) transforms into Eq. (2b).

2.5. Semi-grand canonical Monte Carlo with feedback (FB-SGMC)

In the semi-grand canonical ensemble, described in Section 2.4, the chemical composition is not directly defined. It comes as a result of the MC equilibration procedure at a fixed set of chemical potentials $\{\mu_\alpha\}$. The desired chemical composition

has to be adjusted by choosing the appropriate chemical potentials, μ_α, which is done by trial and error. This is time consuming and not very convenient when a specific composition is targeted.

FB-SGMC allows for achieving a predefined concentration of element α, by evolving μ_α through an iterative feedback loop. In this loop, the change of μ_α after each MCS is proportional to the deviation of the current atomic fraction, c_α, of element α from a preset value, c_α^0. The iterative equation is given as

$$\mu_\alpha^{(n)} = \begin{cases} \mu_\alpha^{(n-1)} - a_\alpha \left(\frac{c_\alpha^{(n-1)} + c_\alpha^{(n-2)}}{2} - c_\alpha^0 \right), & for\ n \geq 2 \\ \mu_\alpha^{(0)}, & for\ n < 2 \end{cases}, \tag{5}$$

where index *n* labels Monte Carlo steps, and a_α is an adjustable parameter that affects the computational efficiency but not the equilibrium end-state. Once the system has reached equilibrium, both the chemical composition and the chemical potential fluctuate around their equilibrium values. The size of the fluctuations depends on a_α. Increasing a_α decreases the fluctuations**.**

The rest of the Monte Carlo scheme proceeds as the semi-grand canonical ensemble with fixed μ_α, as described in Section 2.4. Since the total number of atoms is fixed ($\sum_\alpha N_\alpha = N = const.$), only the differences between the chemical potentials, $\Delta\mu_{\alpha\beta} = \mu_\alpha - \mu_\beta$, affect the composition. Therefore, without a loss of generality, one of the chemical potentials can be set to zero and the remainder can be evolved through Eq. (5).

2.6. Variance-constrained semi-grand canonical Monte Carlo (VC-SGMC)

VC-SGMC [5] is similar to FB-SGMC as described in Section 2.5, the only difference being the choice of the update function for μ_α. Instead of using Eq. (5), μ_α is updated according to the following equation

$$\mu_\alpha^{(n)} = \begin{cases} \mu_\alpha^{(n-1)} - 2b_\alpha \frac{c_\alpha^{(n-1)} - c_\alpha^{(n-3)}}{2}, & for\ n \geq 3 \\ \mu_\alpha^{(0)}, & for\ n < 3 \end{cases}, \tag{6}$$

where b_α is an adjustable parameter that affects computational efficiency but not the equilibrium end-state. Note that in Eq. (6) there is no targeted concentration, c_α^0, and the resulting equilibrium concentration depends on the initial state of the system at $n = 0$ defined by $c_\alpha^{(0)}$ and $\mu_\alpha^{(0)}$, together with the external parameters, such as pressure and temperature. Since it can be proven [5] that Eq. (6) evolves the system towards equilibrium, then if the system starts from equilibrium, i.e., $c_\alpha^{(0)}$=c_α^0, and $\mu_\alpha^{(0)} = \mu_\alpha^0$, equilibrium will be maintained during the simulation. Thus, VC-SGMC is used to maintain equilibrium, while FB-SGMC is used to reach equilibrium.

Both FB-SGMC and VC-SGMC, can be considered as special cases of a so-called Gaussian ensemble [5,9]. Gaussian ensemble is a theoretical construction where the system, instead of being in contact with an infinite reservoir of particles, as in the grand

canonical ensemble, is in contact with a finite reservoir. In the limiting case, when the reservoir cannot supply or take away any particles, Gaussian ensemble is transferred into the canonical ensemble. Thus, Gaussian ensemble can be considered as an intermediate case between canonical and grand canonical ensembles. Being in contact with a *finite* reservoir of particles, means that the fluctuations in c_α will have a Gaussian distribution around the equilibrium concentration, with a dispersion that can be controlled by the size of the reservoir, or in the cases considered here, by the adjustable parameters a_α and b_α in Eq. (5) and Eq. (6), respectively.

The advantages of the Gaussian ensemble with controlled system fluctuations is in simulating multicomponent systems during phase transformation or within the phase coexistence region in the phase diagram for immiscible systems. During a first order phase transition, the chemical potential has a discontinuity and becomes different in the two (old and new) phases. Thus, for a system with phase coexistence, or in the process of phase transition, a global chemical potential for the whole system is not defined. Consequently, fixing the chemical potential in (μVT) or (μPT) ensembles, as described in Section 2.4, makes it impossible to simulate phase coexistence and phase interfaces. Allowing for μ to vary in the FB-SGMC or VC-SGMC methods makes it possible to simulate phase coexistence.

At present, there is no clear argument which of the two methods, FB-SGMC or VC-SGMC, is more advantageous in simulating phase coexistence. A detailed comparison study for the case of the γ/γ′ phase coexistence in NiAl alloy system is given in [9], which shows that both methods are comparable under a suitable choice of the a_α and b_α parameters. The ability to explicitly prescribe a target chemical composition in the FB-SGMC method gives it a certain practical advantage over the VC-SGMC.

2.7. Canonical ensemble with atom swap moves

Considering the problems with a non-defined global chemical potential during the state of phase coexistence, as discussed in Section 2.6, a method to simulate phase coexistence is to use canonical ensemble with fixed composition. The problem with using CMC in (NVT) or (NPT) versions, as described in Sections 2.2 and 2.3, is the inability to change the chemical type of an atom. Thus, relying only on displacement moves alone, takes a prohibitively long time to intermix the chemical elements.

One way to introduce chemical change of an atom without changing the overall chemical composition and the total number of atoms is to use swap moves. In a swap move, a pair of atoms is selected at random and their chemical type is swapped, so that the chemical element of the first atom is assigned to the second atom, and vice versa. The move is accepted with the probability defined by Eq. (1), where

$$\Delta\Phi = \Delta E_{\alpha\beta} + \Delta\mu_{\alpha\beta} + \Delta\mu_{\beta\alpha} + \frac{3}{2}k_B T\left(ln\frac{m_\alpha}{m_\beta} + ln\frac{m_\beta}{m_\alpha}\right) = \Delta E_{\alpha\beta}. \qquad (7)$$

Above, $\Delta E_{\alpha\beta}$ is the potential energy difference due to the swap of the chemical types α, and β, between the two atoms, while their atomic positions remain fixed. Since

$\Delta\mu_{\alpha\beta} = -\Delta\mu_{\beta\alpha}$, and $ln\frac{m_\alpha}{m_\beta} = -ln\frac{m_\beta}{m_\alpha}$, the only non-zero term left in Eq. (7) is $\Delta E_{\alpha\beta}$. If the two atoms are of the same chemical type ($\alpha \equiv \beta$), then there is no change in the system state (i.e., $\Delta E_{\alpha\alpha} = 0$), but statistically, the attempt is counted as accepted, since in this case $\Delta\Phi = 0$ and $P = 1$ in Eq. (1). This regime simulates the same canonical ensembles as described in Section 2.2 and Section 2.3 with the constraints defined by Eq. (2a) and Eq. (3a) for NVT and NPT ensembles, respectively.

In combination with the displacement MC moves, described in Section 2, the swap move tremendously accelerates the intermixing of atoms of different chemical types in a multicomponent system. Its efficiency in exploring a large volume of the configuration space is similar to the SGMC methods described in Sections 5 and Section 6. Note also, that there is no need of defining a chemical potential as an additional parameter, because the chemical potential terms in Eq. (7) cancel each other. Because of the fixed chemical composition, the canonical ensemble with swap moves is highly efficient in simulating phase coexistence in multicomponent multiphase systems.

The disadvantage of implementing a swap MC move is in the difficulty of its parallelization. At present, there is no efficient way to execute a series of random swap moves in parallel. For large systems, where a constant composition needs to be maintained, the preferred methods are the FB-SGMC and the VC-SGMC methods, because of the possibility for their parallelization. In such cases, the constant chemical composition is achieved by setting high enough values for the coefficients a_α and b_α in Eq. (5) and Eq. (6), respectively, so that the fluctuations around that composition can be made sufficiently small [9].

In ParaGrandMC, the execution of the swap MC move has been optimized significantly and partially parallelized. For example, while the swap move can be executed together with the displacement moves of the two pair atoms, which is usually the case in serial MC codes, in ParaGrandMC the swap move is performed separately. At each MCS, the code first executes *N* displacement trial moves, and then another series of *N* swap MC moves. The separation of the two moves allows for speed up by: (i) applying full parallelization on the displacement moves; (ii) taking full advantage of the fact that the swap moves do not include change in the atomic positions, but only of the chemical types of the atoms, which can significantly speed up the calculation of ΔE; and (iii) some partial parallelization can still be applied for stand alone swap moves, depending on the relative positions of the two paired atoms. In spite of that, the tests presented in Section 5 of this report show that this regime is the least efficient of all methods examined and requires further development.

3. Parallel Implementation of the Monte Carlo Method

From a computational standpoint, the Monte Carlo method requires sequential random memory access, which prevents efficient parallelization. Until recently, parallelization schemes were implemented only in terms of replicating the simulated system with some random variations on the available number of processor cores, or processing elements (PEs), and executing independent MC simulations on each processor for obtaining better

and faster statistical sampling. This strategy is mostly efficient for relatively small systems in equilibrium, which are thermodynamically equivalent. On the other hand, the replica method does not take advantage of parallelizing the computation of a single system and is limited when the system size is too large to be handled efficiently by a single PE. In addition, many systems require long simulation time to reach thermodynamic equilibrium before being replicated. A particular example is the case of multicomponent metal alloys of non-stoichiometric composition, which require large atomic systems to be simulated for representative statistics.

ParaGrandMC utilizes a recently suggested algorithm [3] for parallelization based on a special kind of domain decomposition. The domain decomposition is implemented in combination with a link-cell algorithm used for fast search of nearest neighbors in the interaction range of a given particle [1].

In brief, the link-cell algorithm divides the entire system volume into link-cells (Figure 1), and each link-cell keeps a list of the atoms that are placed inside it. The link-cells are of close to cubic shape with a size a_{cell} slightly larger than the interaction range of the potential. The search for a neighbor in the interaction range of a particle does not have to exceed the nearest neighbor link-cells. In a 3D system the search range includes a cube of $3^3 = 27$ link-cells, rather than the whole system.

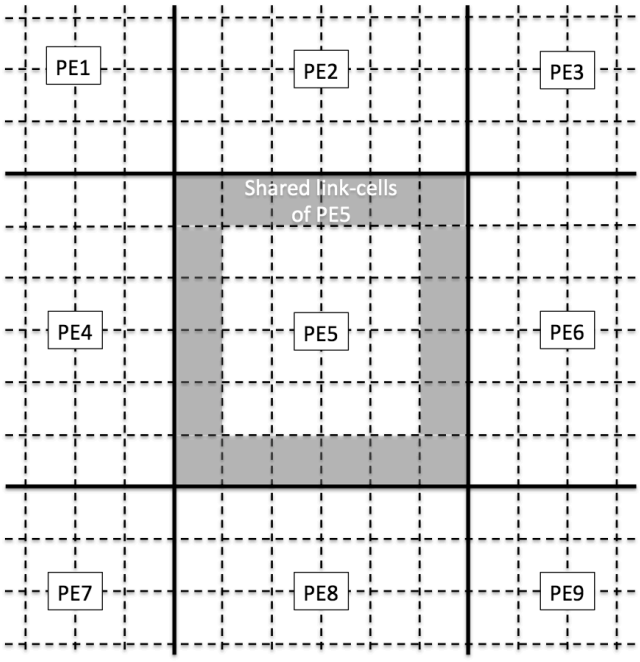


Figure 1: Schematic representation of the link-cell spatial decomposition of the volume of a simulated system among a number of Processing Elements (PEs) for parallel computation. Bold lines indicate regions for different PEs. Dotted lines indicate the link-cell mesh with the cells in grey indicated as shared cells with the neighboring PE.

Each PE processes a part of the link-cell mesh assigned to it following a uniform spatial decomposition of the whole system volume. The PE shares the states of the atoms in the link-cells neighboring other PEs. In ParaGrandMC, the spatial decomposition is

performed only in the y- and z-directions, resulting in a two-dimensional (2D) grid arrangement of the PE units (while the link-cell mesh is 3D). For systems of moderate size of less than 10^7 particles, a 2D PE arrangement was found to be optimal from a performance standpoint, compared to a 3D arrangement, which requires more internode communications. The y-z 2D PE arrangement achieves best performance when the x-dimension is chosen to be the shortest system dimension, because it is not decomposed over multiple PEs. For optimal performance, the spatial decomposition parallelization requires an even number of PEs in each dimension. Thus, for the 2D decomposition, the total number of requested PEs must be divisible by 4.

The parallelization of the MC simulation explores the fact that if two MC moves are independent of each other (meaning that the acceptance or rejection of the first does not affect the acceptance or rejection of the second, and vice versa), they can be executed simultaneously. In the case of a short range interaction, MC moves on particles that are further apart than the interaction range are independent and can be executed simultaneously. Thus, the MC algorithm for executing 1 MCS (consisted of N MC trial moves and N = number of atoms in the system) proceeds by first, generating the random list of N trial atom selections* needed for the entire MCS. Next, the algorithm identifies groups of atoms in the list that are all outside of each others interaction range. Trial moves on the atoms of one such group can be executed in parallel by the available PEs. The same steps are repeated for the next group, until all N trial moves are conducted, before repeating the procedure for the next MCS.

The key point in this algorithm is how to efficiently identify the groups of non-interacting atoms. The procedure takes advantage of the already created link-cell mesh for the system. For simplicity, the procedure is illustrated in Figure 2 for a two-dimensional system, but its 3D implementation is straightforward. The link-cells are grouped in 2x2 arrays and are indexed inside each array from 1 to 4 in a clockwise direction. All link-cells of the same index form a set. As a result, atoms from different link-cells of one and the same set (for example, atoms **A** and **B** marked in Figure (2)) are guaranteed to be outside of their interaction range and can be executed simultaneously if they belong to separate PEs. The algorithm for the execution of one MCS is illustrated through the flowchart in Figure 3.

* Selecting an atom several times in the list is allowed.

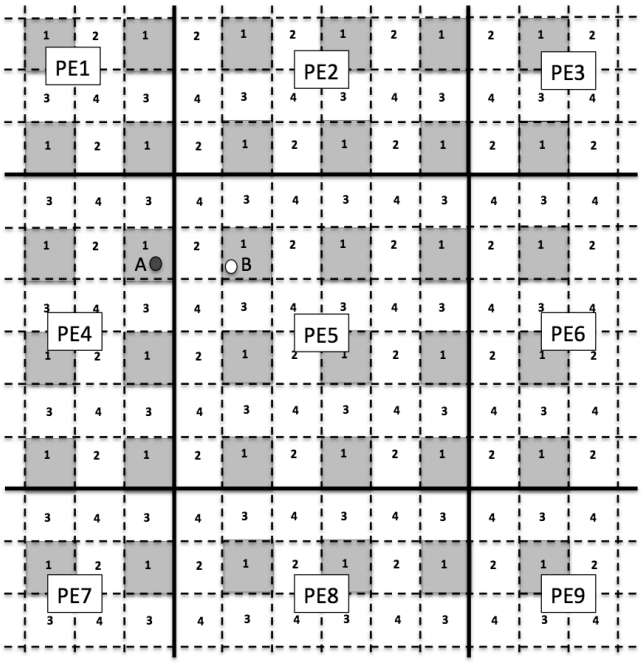


Figure 2: Two-dimensional partitioning of the link-cell mesh in sets of four cells, numbered from 1 to 4. For demonstration, the link-cells of set number 1 are shown in gray. Events on atoms of the same set, but of different cells (for example on atoms **A** and **B)**, can be performed simultaneously if they belong to different PEs.

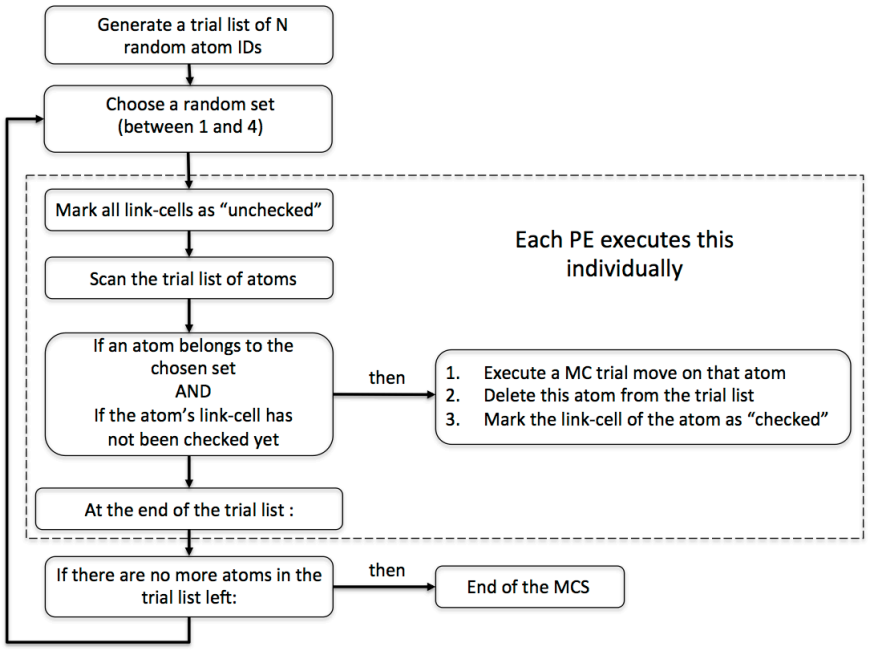


Figure 3: Flowchart summarizing the parallel execution of one Monte Carlo Step (MCS) for a system of N atoms. The operations inside the dashed rectangle are executed by each PE independently. The operations outside the dashed rectangle are executed by the master PE.

4. Input and Output Data

4.1. Input data

ParaGrandMC needs the following input data: (i) an input model configuration file; (ii) a command file; and (iii) a set of files describing the implemented interatomic potential. A detailed description of each file format is given in the ParaGrandMC User's Guide, which is included in the distribution package of the software. Here, a brief description of these files is given.

The input model configuration file, named "structure.plt" contains a header with the dimensions of the system and the number of atoms it contains, followed by a list of all atoms. Each atom is described by its identification (ID) number, atomic (x, y, z) coordinates in Å, a number identifying the associated chemical element, and a code number for the constraint degrees of freedom for this atom, if any.

The command file, named "pgmc.com", contains a list of commands with parameters that define the simulation conditions for various regimes describing the whole simulation procedure. A complete list of the commands and their implementation is given in the User Guide.

The interatomic potential is presented in the form as published in the NIST repository for interatomic potentials [10]. The potential is represented as a series of tabulated functions given in text file format in *.dat files stored in one directory. A list of the potential files and their ordering is given in the file "pot.dat", which is used by ParaGrandMC to access the files.

Interatomic potentials of EAM type [6,7] and ADP type [8] are currently implemented in ParaGrandMC. Their functional form, expressed through the total system potential energy *E*, is given as follows. For the EAM potential:

$$E = \frac{1}{2}\sum_{i,j(i\neq j)} \Phi_{ij}\left(r_{ij}\right) + \sum_i \mathrm{F}_i(\bar{\rho}_i), \tag{8a}$$

and for the ADP potential:

$$E = \frac{1}{2}\sum_{i,j(i\neq j)} \Phi_{ij}\left(r_{ij}\right) + \sum_i \mathrm{F}_i(\bar{\rho}_i) + \frac{1}{2}\sum_{i,\alpha}(\mu_i^{\alpha})^2 + \frac{1}{2}\sum_{i,\alpha,\beta}\left(\lambda_i^{\alpha\beta}\right)^2 - \frac{1}{6}\sum_i {\nu_i}^2. \tag{8b}$$

Here, indices *i* and *j* enumerate interacting atoms at a distance r_{ij}, and the superscripts $\alpha, \beta = 1, 2, 3$ refer to the Cartesian coordinates. The first term in Eqs. (8a and b) represents pair interactions, $\Phi_{ij}\left(r_{ij}\right)$ being the pair-interaction potential between atoms *i* and *j*. The term $\mathrm{F}_i(\bar{\rho}_i)$ is the embedding energy of atom *i* as a function of the collective electron density $\bar{\rho}_i$ created at site *i* by all other atoms in the system

$$\bar{\rho}_i = \sum_{j\neq i} \rho_j\left(r_{ij}\right), \tag{9}$$

where $\rho_j(r_{ij})$ is the electron density coming from atom *j*, placed at a distance r_{ij} from atom *i*. The $\Phi_{ij}(r_{ij})$ and $F_i(\bar{\rho}_i)$ terms are both centrosymmetric in the sense that they depend only on the distance r_{ij} between atoms *i* and *j*, but not on their relative orientation with respect to other atoms in the system. Only these terms constitute the EAM potential in Eq. (8a).

In a number of cases, a centrosymmetric potential is not enough to describe the interatomic forces. Some examples include transition metals, or systems with more pronounced covalent bonding. To address this deficiency, the EAM form Eq. (8a) has been extended to include orientation dependent vector and tensor terms in Eq. (8b) in the form of

$$\mu_i^{\alpha} = \sum_{j \neq i} u(r_{ij}) r_{ij}^{\alpha}, \tag{10a}$$

$$\lambda_i^{\alpha\beta} = \sum_{j \neq i} w(r_{ij}) r_{ij}^{\alpha} r_{ij}^{\beta}, \tag{10b}$$

and the trace of λ

$$\nu_i = \sum_{\alpha} \lambda_i^{\alpha\alpha}. \tag{10c}$$

Eqs. (10a-c) require the definition of the two additional functions $u(r)$ and $w(r)$, which express the anisotropy of the interactions in terms of dipole and quadrupole expansion of the potential energy. These additional functions enable a description of a direction-dependent interactions in the ADP potential, but at a computational expense that is twice the EAM potential.

ParaGrandMC can also utilize model configuration and potential files in the format defined by the broadly used Large-scale Atomic/Molecular Massively Parallel Simulator (LAMMPS) package [4].

4.2. Output data

As a result of the simulation, ParaGrandMC produces the following output files: (i) an output configuration file, (ii) an output data file, and (iii) a log file.

The output configuration file has the same format as the input configuration file "structure.plt", but its name is given in the pgmc.com command file keeping the extension "plt". The output file can be directly used as an input file, after renaming to "structure.plt", so that a follow up simulation can be started where the first simulation has been interrupted. Because output configurations can be produced repeatedly during the simulation, each output configuration file has the number of the performed MCS as a suffix to its name. For example, "filename.00123456.plt" stores the system configuration produced after 123456 MCS. A converter from "plt" to "imd" format for use with the opens source code OVITO [11] for visualization and analysis purposes is available. The output configuration file can also be written in LAMMPS format [4].

The output data file has the name of the configuration output file, as given in the pgmc.com file, but with extension "dat", and without the MCS number (e.g.,

"filename.dat"). This file stores measurements of various parameters of the system, such as system energy, system dimensions, chemical composition, etc. that are taken periodically during the simulation as instructed in the pgmc.com file. The output data file is continuously appended as the simulation proceeds. The data are stored in text format in columns and can be used directly with a number of plotting programs, such as xmgrace [12], gnuplot [13], or Microsoft Excel®.

The log file, called "pgmc.out", stores log reports produced periodically from the code. Its purpose is mostly to help debugging if something has gone wrong during the simulation. As such, the pgmc.out file contains more detailed information about each step in the simulation process. Some examples are: the parameters of the interatomic potential, such as the range of the interactions and the type of the interacting chemical elements, information on the spatial decomposition of the system on different PEs, and how many atoms were on each PE.

5. Parallel Performance and Scalability

5.1. Model description

Several benchmark tests were performed with ParaGrandMC to demonstrate its performance and scalability. The tests were performed on a Beowulf Linux cluster consisting of nodes containing 4 CPUs each. The CPUs are 8-core AMD Opteron™ 6320 series processors rated at 2.8 GHz, which makes a total of 32 PEs per node. The code was compiled using Intel® Composer XE compiler. The tests were performed on a single node using up to 32 PEs, and on two nodes, using up to 64 PEs.

The tests used both types of interatomic potentials as described in Section 4 for two alloys. The EAM potential [7] describes a $Ni_{85}Al_{15}$ alloy of 85 atomic percent Ni and 15 atomic percent Al, and the ADP potential [8] describes a $Ni_{85}Fe_{15}$ alloy. The two alloys are structurally very similar forming the γ' phase as $L1_2$–structure in Ni_3Al and Ni_3Fe accommodating a few percent off-stoichiometry. The off-stoichiometry was chosen intentionally in order to allow for higher dynamics in the chemical variations in the systems. In the case of $Ni_{85}Al_{15}$, a coexistence with an atomically disordered γ phase can be observed within this chemical composition [9]. To make the tests for the two potentials comparable, the shorter interaction range of the Ni-Fe ADP potential of 5.2 Å was artificially increased to match the 6.7 Å of the Ni-Al EAM potential. This increase was performed by setting the potential energy in the additional range at zero, so that the atomic interaction is not altered, while the number of the examined neighbors, which substantially effects the computational speed, is made equal for both potentials.

The atomic systems for both alloys were constructed from a startup cubic system, containing N=4000 atoms simulated under periodic boundary conditions in all three x-, y-, and z-directions. After equilibration at the simulation temperature of T = 1000K, the size of the system box was established at 35.7 Å for the $Ni_{85}Al_{15}$ alloy, and at 35.8 Å for the $Ni_{85}Fe_{15}$ alloy. After equilibration, the two startup systems were scaled up by replication in the y- and z- directions to N=16000 (2x2), 64000 (4x4), and 256000 (8x8) atoms. The size in the x-direction was not replicated, because the spatial decomposition for

parallelization is performed only in the y- and z-directions, as described in Section 3.

A series of test simulations of 1000 MCS each were performed and the processor time was clocked for the MC cycles only, excluding input-output operations. All tests were performed at constant volume, fixed at the T=1000K equilibration temperature. In addition, a reference simulation on a single PE in serial mode (excluding all of the PE-to-PE communication overhead) was performed for all tested regimes.

5.2. Test results

Figure 4 shows the wall clock time taken for one MC simulation of 1000 MCS as a function of the number of PEs requested. The results given are for the EAM Ni-Al potential model. Except for the smallest system of 4000 atoms, the strongest decrease in simulation time is at the transition from a serial execution on 1 PE to a parallel execution on 4 PEs. For the 4000 atom system, the communication overhead required for the parallel execution becomes comparable to the computation time and the benefit of the parallelization is small.

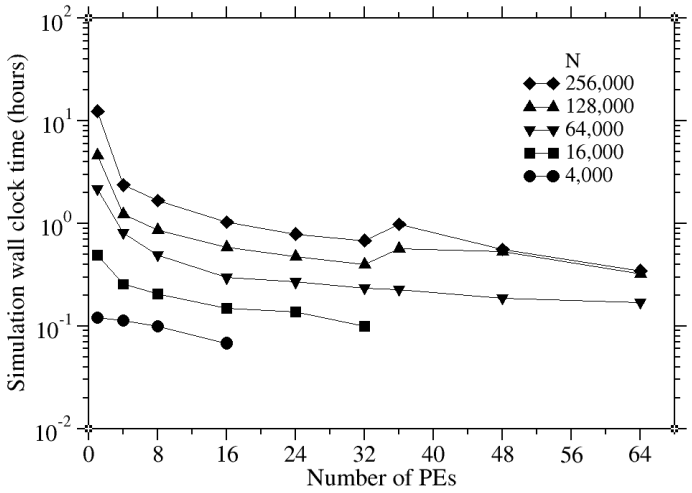


Figure 4: Wall clock time for executing 1000 MCS on systems of different number of atoms *N*, as indicated in the figure, as a function of the requested number of PEs. The data are for the CMC (NVT) ensemble (Section 2.2).

A notable increase in the wall clock time is observed at the transition from 32 to 36 PEs for the N=128000 and N=256000 atom systems. This increase reflects the transition from the execution on a single node (of 32 PEs) to multiple nodes, which require inter-node communication through an external router. This increase shows the sensitivity of the parallel MC scheme to the rate of data transfer between nodes.

Since the cost of computer time is usually given per CPU hour, Figures 5(a,b) show the simulation time in terms of CPU hours (equal to the wall clock time multiplied by the number of requested PEs) as a function of the PE number for the EAM and the ADP potentials, respectively. In general, increasing the number of PEs results in an increase of the overall CPU time (and respectively, the cost of the simulation), which is due to the slower than proportional increase of the speed as the number of PEs is increased. Overall,

the ADP model is approximately twice more costly than the EAM model, which is as expected [8].

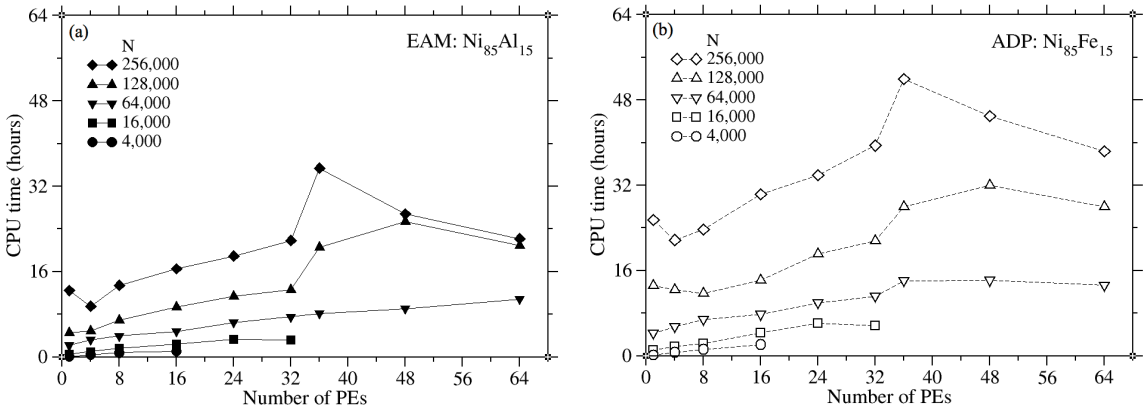


Figure 5: Total CPU time for executing 1000 MCS on systems of different number of atoms *N*, as indicated, as a function of the requested number of PEs, using (a) EAM potential, and (b) ADP potential. The data are for the CMC (NVT) ensemble (Section 2.2).

A notable observation in Figure 5(a) is the appearance of a dip in the spent CPU time when going from 1 to 4 PEs for the largest systems of N=256000 atoms for the EAM potential. Similar dips are also present for the ADP systems of N=128000 and N=256000 atoms in Figure 5(b). These local minima indicate a threshold in the system size, where parallelization becomes highly beneficial. One reason for such a threshold is the higher than linear decrease of the computation time due to the decrease of the allocated memory volume per PE as the atoms become distributed over multiple PEs. Due to the random memory access in the MC algorithm, the simulation time is highly sensitive to the allocated memory. For large enough systems, distributing the system volume to more PEs becomes cost effective.

A notable peak increase of the CPU time is observed when the PE number exceeds 32 PEs for both potentials. As mentioned already, this increase is due to requesting PEs from two nodes (of 32 PEs each) and the need for inter-node communication. The demonstrated behavior underscores the substantial role of the hardware architecture on the effectiveness of the parallel MC algorithm.

The scaling plots shown in Figure 6 illustrate how many times faster (speed up) the MC simulation runs on multiple PEs compared to a single PE for both potentials. The data show that there is a steady increase of the speed up as the number of PEs increases. Both potentials show similar trends in speed up relative to a single PE execution with the ADP potential showing systematically higher values. The higher computational demand for the ADP potential while having the same communication overhead compared to the EAM potential favors parallelization. Larger systems experience notably larger speed up due to the higher load per PE, which benefits more from parallelization. Again, there is a noticeable dip in the speed up as the PE number exceeds 32 PEs and the inter-node communication slows down the performance.

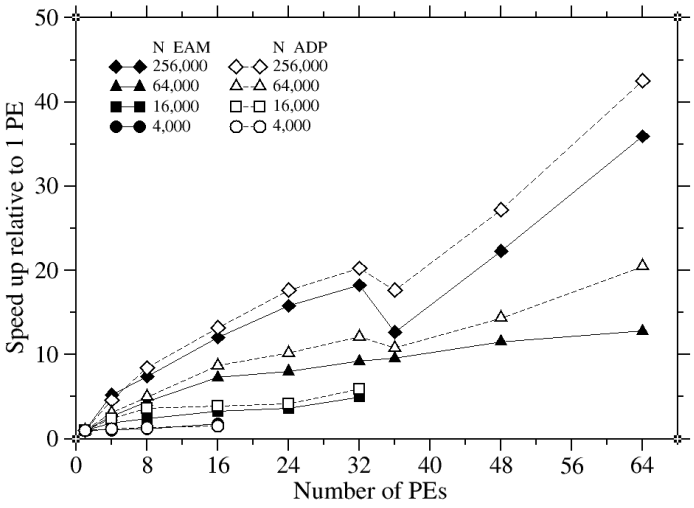


Figure 6: Speed up of a parallel execution relative to a serial 1 PE execution on systems of different sizes for the two implemented types of interatomic potentials, EAM and ADP. The data are for the CMC (NVT) ensemble (Section 2.2).

A question of interest is how MC simulations using the different ensembles, presented in Section 2, compare in terms of speed up. Figure 7 shows this comparison for the largest examined system of 256000 atoms. Figure 7a shows that in the case of the EAM potential, all three versions of the SGMC ensemble are equivalent in performance. The CMC (NVT) ensemble is slightly slower, which may be counterintuitive, since it is the simplest model of all. This ensemble exploits only single atom displacement MC moves, without change of the chemical type. The actual reason for its lower speed up is the higher acceptance rate of these moves, which is almost twice as high, compared to all of the SGMC ensembles (79% vs. 43% acceptance, respectively). Accepted trial moves involve more computations, because of the required update of the particle state in the system. In principle, the acceptance rate can be controlled through adjusting the maximum value of the random displacement move, which is an input parameter in the simulations.

The lowest speed up is found for the canonical Monte Carlo with swap moves. As it has been discussed in Section 2.7, the swap MC moves are most difficult to parallelize, because they involve the simultaneous change of a pair of particles randomly selected from around the system. This regime is only partially parallelized, and its speed up is least affected by the number of PEs. For parallelization on more than 32 PEs, this speed up is practically non-existent. The ADP model, presented in Figure 7b, shows similar behavior as the EAM potential model with slightly higher speed up numbers, due to its more computationally demanding functional form, which favors parallelization.

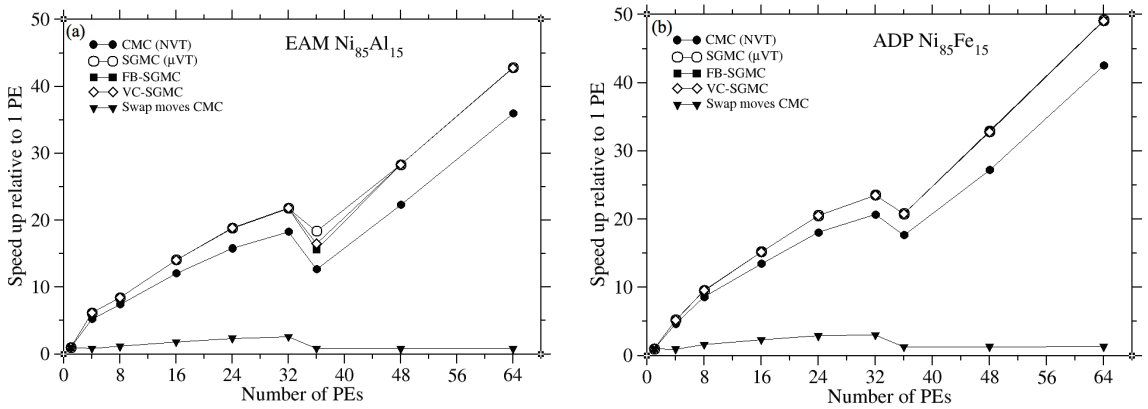


Figure 7: A comparison of the parallel execution relative to a serial execution speed up for the various ensemble types, as discussed in Section 2, on a system of N=256000 atoms, for the two implemented types of interatomic potentials, (a) EAM and (b) ADP.

6. Summary

This report presents the basic concepts of the ParaGrandMC code, together with the physics behind it. To help potential users understand the advantages of using ParaGrandMC, results of several parallel performance tests are provided. Overall, the analysis shows good scalability of the performance with the number of requested PEs. The computational efficiency of the code improves with the increase of the computational load. Larger systems using more computationally demanding calculations benefit more from parallelization. Parallelization for systems of less than 10000 atoms does not improve the performance substantially, and are best executed in a serial mode on 1 PE.

A strong sensitivity of the performance on the communication speed between PEs is observed. Going from a shared memory intra-node to a distributed memory inter-node communication results in a noticeable decrease in performance.

Differences in performance for the various simulated regimes implementing different kinds of MC trial moves are discussed. With the exception of the swap-moves canonical Monte Carlo regime, which is only partially parallelized, no significant performance differences are found between the rest of the regimes. The swap move CMC shows some small speed up when executed on a single multicore multi-CPU platform. In order to compensate for the low parallel efficiency of the swap move CMC, ParaGrandMC provides a number of options to constrain the fluctuations in the chemical composition for the FB-SGMC and VC-SGMC ensembles, making them a good substitution to the swap-move CMC for large system simulations.

The ParaGrandMC code is available through https://software.nasa.gov/ .

References

[1] Frenkel, B., Smit, B., Understanding Molecular Simulation (Academic Press, London, 2001).

[2] Metropolis, N., Rosenbluth, A.W., Rosenbluth, M.N., Teller, A.H., Teller, E., "Equations of state calculations by fast computing machines," Journal of Chemical Physics 21, (1953) 1087-1092.

[3] Plimpton, S., Battaile, C., Chandross, M., Holm, L., Thompson, A., Tikare, V., Wagner, G., Webb, E., Zhou, X., Cardona, CG., Slepoy, A., "Crossing the mesoscale no-man's land via parallel kinetic Monte Carlo," SANDIA report: SAND2009-6226, October (2009).

[4] LAMMPS Molecular Dynamics Simulator: http://lammps.sandia.gov/

[5] Sadigh, B., Erhart, P., Stukowski, A., Caro, A., Martinez, E., Zepeda-Ruiz, L., "Scalable parallel Monte Carlo algorithm for atomistic simulations of precipitation in alloys," Phys Rev B 85 (2012) 184203-1-11.

[6] Daw, M.S., Baskes, M.I., "Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals," Phys. Rev. B 29, (1984) 6443-6453.

[7] Mishin, Y., "Atomistic modeling of the γ and γ′ phases of the Ni–Al system," Acta Mater. 52 (2004) 1451–67.

[8] Mishin, Y., Mehl, M.J., Papaconstantopoulos, D.A., "Phase stability in the Fe-Ni system: Investigation by first-principles calculations and atomistic simulations," Acta Mat. 53 (2005) 4029-4041.

[9] Mishin, Y., "Calculation of the γ/γ′ interface free energy in the Ni-Al system by the capillary fluctuation method," Modelling Simul. Mater. Sci. Eng. 22 (2014) 045001-1-16.

[10] NIST Interatomic Potentials Repository: www.ctcms.nist.gov/potentials/

[11] Stukowski, A., "Visualization and analysis of atomistic simulation data with OVITO-the Open Visualization Tool," Modell. Simul. Mater. Sci. Eng. 18 (2010) 015012.

[12] http://plasma-gate.weizmann.ac.il/Grace/

[13] http://www.gnuplot.info/